Chap 5 策略优化

2022年1月28日 17:05

- 策略优化
    - 构造一个 带可学习参数 θ 的 策略概率函数
        - 两种：softmax 或 高斯法 等
        - Softmax
        - 高斯法，就是 假设 policy 是 正态分布的，期望，是 状态 的线性和 （θ 就是权重），方差 可以是固定的，亦可以作为参数的一部分
    - 目标是收获的收益最大
        - 不可微分的，一般是黑箱的 函数
            - 注意，这里的 目标函数 J(θ) 一般是通过 实际在环境中跑一圈（或蒙特卡洛法）得到，与 可微分法 是不同的
            - cross-entropy 方法
                - 查看 my_learning_agent.py
                - Derivative-free Method: Cross-Entropy Method

                    1. $\theta^* = \arg\max J(\theta)$
                    2. Treat $J(\theta)$ as a black box score function (not differentiable)

                    Algorithm 1 CEM for black-box function optimization

                    ```
                    for iter i = 1 to N do
                        C = {}
                        for parameter set e = 1 to N do
                            sample θ^(e) ~ P_{μ^(i)}(θ)
                            execute roll-outs under θ^(e) to evaluate J(θ^(e))
                            store (θ^e, J(θ^(e))) in C
                        end for
                        μ^(i+1) = arg max_μ Σ_{k∈Ĉ} log P_μ(θ^(k))
                        where Ĉ are the top 10% of C ranked by J(θ^(e))
                    end for
                    ```

                    3. Example of CEM for a simple RL problem:https://github.com/cuhkrlcourse/RLexample/blob/master/my_learning_agent.py
            - 梯度估计 Finite Difference
                - Theta 的 每个分量 分别 加一个 极小值，来估计 微商
        - 可微分的（理论模型）
            - 一步收益 per-step 的
                - $$\begin{aligned} J(\theta) &= \mathbb{E}_{\pi_\theta}[r] \\ &= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) r \end{aligned}$$
                - d 是以 状态 的分布概率，理论上 应该是用马尔科夫链来计算出来的
            - 路径收益

Policy Gradient for Multi-step MDPs

1. Denote a state-action trajectory from one episode as $\tau = (s_0, a_0, r_1, \ldots s_{T-1}, a_{T-1}, r_T, s_T) \sim (\pi_\theta, P(s_{t+1}|s_t, a_t))$
2. Denote $R(\tau) = \sum_{t=0}^{T-1} R(s_t, a_t)$ as the sum of rewards over a trajectory τ
3. The policy objective is

- $$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} R(s_t, a_t)\right] = \sum_\tau P(\tau;\theta)R(\tau)$$

  where $P(\tau;\theta) = \mu(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$ denotes the probability over trajectories when executing the policy π_θ

  4. Then our goal is to find the policy parameter θ

  $$\theta^* = \arg\max_\theta J(\theta) = \arg\max_\theta \sum_\tau P(\tau;\theta)R(\tau)$$

- τ 理论上 指的是 所有可能的路径中的一个， μ(s0) 指的是以s0作为初始状态的概率，最终 J(θ) 理论上是在所有可能的路径下 收益的 期望
- 当然，理论 是 不可能的，所以通过 采样来实现

  - 计算技巧
    - Likelihood ratio：乘一个，除一个，转化为log的 梯度

      Likelihood ratios exploit the following tricks

      $$\begin{aligned}\nabla_\theta\pi_\theta(s,a) &= \pi_\theta(s,a)\frac{\nabla_\theta\pi_\theta(s,a)}{\pi_\theta(s,a)} \\ &= \pi_\theta(s,a)\nabla_\theta\log\pi_\theta(s,a)\end{aligned}$$

    - 首先，J 的梯度，主要 就是 P 的梯度

      $$\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta\sum_\tau P(\tau;\theta)R(\tau) \\ &= \sum_\tau \nabla_\theta P(\tau;\theta)R(\tau) \\ &= \sum_\tau \frac{P(\tau;\theta)}{P(\tau;\theta)}\nabla_\theta P(\tau;\theta)R(\tau) \\ &= \sum_\tau P(\tau;\theta)R(\tau)\frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)} \\ &= \sum_\tau P(\tau;\theta)R(\tau)\nabla_\theta\log P(\tau;\theta)\end{aligned}$$

    - P 的梯度，通过 likelihood ratio 等于 policy 的 梯度，直接摆脱了 难算的 MDP 动态转移概率

      $$\begin{aligned}\nabla_\theta\log P(\tau;\theta) &= \nabla_\theta\log\left[\mu(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t,a_t)\right] \\ &= \nabla_\theta\left[\log\mu(s_0) + \sum_{t=0}^{T-1}\log\pi_\theta(a_t|s_t) + \log p(s_{t+1}|s_t,a_t)\right] \\ &= \sum_{t=0}^{T-1}\nabla_\theta\log\pi_\theta(a_t|s_t)\end{aligned}$$

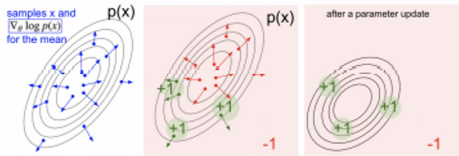
- 策略优化 policy gradient
  - 理解：本质上是 类似于 加权的 极大似然估计（但不是）
    - 首先会有一个 初始的 θ 参数，有一个基于 θ 的策略概率分布，
    - 但初始，策略概率 最大的 不一定是 受益最大的
    - Policy gradient 和 极大似然估计 一样都是样本理论概率的求和， 但 policy gradient 由于 每一种 路径 都乘了 收益，因此 是按照收益加权平均，（即对概率分布进行了一定的扭曲变换）
    - 一次迭代后，策略分布 会更靠近 收益大的 区域

Understanding Score Function Gradient Estimator

1. Consider the generic form of $E_{\tau\sim\pi_\theta}[R(\tau)]$ as

$$\nabla_\theta \mathbb{E}_{p(x;\theta)}[f(x)] = \mathbb{E}_{p(x;\theta)}[f(x)\nabla_\theta \log p(x;\theta)]$$
$$\approx \frac{1}{S}\sum_{s=1}^{S} f(x_s)\nabla_\theta \log p(x_s;\theta), \text{where } x_s \sim p(x;\theta)$$

   1. compute the gradient of an expectation of a function $f(x)$
2. The above gradient can be understood as:
   1. Shift the distribution p through its parameter θ to let its future samples x achieve higher scores as judged by $f(x)$
   2. The direction of $f(x)\nabla_\theta \log p(x;\theta)$ pushes up the log likelihood of the sample, in proportion to how good it is



- 证明参考：https://danieltakeshi.github.io/2017/03/28/going-deeper-into-reinforcement-learning-fundamentals-of-policy-gradients/
  - 第一步构造 $J(\theta)$

    The policy objective is

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} R(s_t, a_t)\right] = \sum_{\tau} P(\tau;\theta)R(\tau)$$

    where $P(\tau;\theta) = \mu(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t,a_t)$ denotes the probability over trajectories when executing the policy π_θ
  - 求梯度，likelihood ratio，将 $J(\theta)$ 化简为只与 策略概率函数、reward 相关
  - 引入 时序因果关系 causality
  - 引入 baseline
  - actor-critic
  - 实际实现中，Gt 的 期望 是 Q(s,a) ， baseline 的期望是 V(s)，因此可以写成：

1. Advantage function: combine Q with baseline V

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$

2. Then the policy gradient becomes:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)A^{\pi_\theta}(s,a)]$$

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a)A^{\pi_\theta}(s,a)]$$
$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$$

1. Two function approximators contain two sets of parameter vectors,

$$V_{\mathbf{v}}(s) \approx V^\pi(s)$$
$$Q_{\mathbf{w}}(s,a) \approx Q^\pi(s,a)$$