Subject: Westlake University, Reinforce Learning, Lecture 10, Actor-Critic Methods

Date: from February 16, 2025 to February 19, 2025

Contents

Lecture 10, Actor-Critic Methods

Bilibili:Lecture 10, Actor-Critic Methods

Introduction

Actor-Critic Methods 仍然是 policy gradient method，实际上其将 value function approximation 结合进了 policy gradient method。这里 actor 指的是 policy update，因为 policy 需要用于实施行动，就是一个 actor 的角色； critic 指的是 policy evaluation / value estimation，因为 action / state value 需要估计，就是一个 critic 的角色.

QAC: the simplest actor-critic

QAC

首先回顾一下 policy gradient 的 basic idea:

1. 定义 scalar metric $J(\theta)$，其可以为 $\bar{v}_\pi$ 或 $\bar{r}_\pi$
2. 最大化 $J(\theta)$ 的 gradient ascent algorithm 理论上为：

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\theta_t) = \theta_t + \alpha \mathbb{E}_{S\sim\eta, A\sim\pi}\left[\nabla_\theta \ln \pi(A|S, \theta_t) q_\pi(S, A)\right]$$

3. 将理论上的期望计算转化为采样的 stocahstic 版本：

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t, \theta_t) q_t(s_t, a_t)$$

从第 3 点可以看出，整个式子就是 ”actor”，因为改变 θ 实际上就是在 update policy； $q_t(s_t, a_t)$ 的算法就是 ”critic”，即评判.

不同的估计 action value q_t 的方法也就产生了不同的算法：

1. 若使用 MC 的方法，那么对应的算法就称为 REINFORCE 或者 Monte Carlo policy gradient
2. 若使用 TD 的方法，那么对应的算法就称为 actor-critic

Algorithm 1 The Simplest Actor-Critic Algorithm (QAC)

Aim: Search for an optimal policy by maximizing $J(\theta)$.

for each time step t in each episode **do**

Generate a_t following $\pi(a \mid s_t, \theta_t)$, observe r_{t+1}, s_{t+1}, and then generate a_{t+1} following $\pi(a \mid s_{t+1}, \theta_t)$. ▷ Generate the experience sample $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1})$ for SARSA

Critic (value update):

$$w_{t+1} = w_t + \alpha_w \left[r_{t+1} + \gamma q(s_{t+1}, a_{t+1}, w_t) - q(s_t, a_t, w_t)\right] \nabla_w q(s_t, a_t, w_t)$$

Actor (policy update):

$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \ln \pi(a_t \mid s_t, \theta_t) q(s_t, a_t, w_{t+1})$$

end for

1. **critic:** SARSA + value function approximation
2. **actor:** the policy update algorithm
3. **on-policy:** 由于理论上进行梯度上升时 $\mathbb{E}_{S\sim\eta,A\sim\pi}$ 中 $\theta\sim\pi$，这里的 π 既是 behavior policy(要按照这个采样)，也是 target policy
4. 虽然是 on-policy 的，但是由于算法本身具有随机性（探索能力），因此无需使用 $\epsilon-$greedy
5. 此算法是最简单的 actor-critic 算法，Q 是指 action value $q-$value

Advantage actor-critic(A2C)

A2C: Baseline invariance

Advantage actor-critic(A2C) 是QAC 的一个推广，因为名称有两个 ”a” 所以简称 A2C。其核心 idea 是引入一个 baseline / 偏置量已减小估计的方差：

$$\nabla_\theta J(\theta)=\mathbb{E}_{S\sim\eta,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)(q_\pi(S,A)-b(S))\right] \tag{1}$$

其中 $b(S)$ 为 S 的标量函数，被称为 baseline.

下面证明 baseline 有如下两个特点：

1. **Valid:** 引入 baseline 不会对 $\nabla_\theta J(\theta)$ 的值造成影响
2. **Useful:** 引入 baseline 可以降低 variance，提高采样结果稳定性

Valid: 引入 baseline 不会对 $\nabla_\theta J(\theta)$ 的值造成影响

Proof. 为证明

$$\mathbb{E}_{S\sim\eta,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)q_\pi(S,A)\right]=\mathbb{E}_{S\sim\eta,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)(q_\pi(S,A)-b(S))\right]$$

即要证明

$$\mathbb{E}_{S\sim\eta,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)b(S)\right]=0$$

将上式展开即可得到结论：

$$\begin{aligned}\mathbb{E}_{S\sim\eta,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)b(S)\right]&=\sum_{s\in\mathcal{S}}\eta(s)\sum_{a\in\mathcal{A}}\pi(a|s,\theta_t)\nabla_\theta\ln\pi(a|s,\theta_t)b(s)\\&=\sum_{s\in\mathcal{S}}\eta(s)\sum_{a\in\mathcal{A}}\nabla_\theta\pi(a|s,\theta_t)b(s)=\sum_{s\in\mathcal{S}}\eta(s)b(s)\sum_{a\in\mathcal{A}}\nabla_\theta\pi(a|s,\theta_t)\\&=\sum_{s\in\mathcal{S}}\eta(s)b(s)\nabla_\theta\sum_{a\in\mathcal{A}}\pi(a|s,\theta_t)=\sum_{s\in\mathcal{S}}\eta(s)b(s)\nabla_\theta 1=0\end{aligned}$$

□

Useful: 引入 baseline 可以降低 variance，提高采样结果稳定性

Proof. 下面证明 $b(S)$ 会对 $\mathrm{var}(X)$ 造成影响，为此我们考虑 $\mathrm{tr}[\mathrm{var}(X)] = \mathbb{E}[X^TX] - \bar{x}^T\bar{x}$，就有

$$\begin{aligned}\mathbb{E}[X^TX] &= \mathbb{E}\left[(\nabla_\theta \ln \pi)^T(\nabla_\theta \ln \pi)(q(S,A)-b(S))^2\right] \\ &= \mathbb{E}\left[\|\nabla_\theta \ln \pi\|^2(q(S,A)-b(S))^2\right]\end{aligned}$$

(**see details in textbook**) □

因此可以看出，方差较小的算法会使得在采样时采样得到的 x 接近 $\mathbb{E}[X]$，所以我们希望通过改变 baseline 来尽量降低方差，这就需要选择最优的 b.

Optimal baseline: 实际上最优的 baseline 为(**see details in textbook**)

$$b^*(s) = \frac{\mathbb{E}_{A\sim\pi}[\|\nabla_\theta \ln \pi(A|s,\theta_t)\|^2 q(s,A)]}{\mathbb{E}_{A\sim\pi}[\|\nabla_\theta \ln \pi(A|s,\theta_t)\|^2]} \tag{2}$$

但是由于其有些复杂，在平常的算法中常使用如下 baseline:

$$b(s) = \mathbb{E}_{A\sim\pi}[q(s,A)] = v_\pi(s) \tag{3}$$

A2C: algorithm

根据前面的分析，我们令 $b(s) = v_\pi(s)$，这样得到的 gradient-ascent algorithm 为

$$\begin{aligned}\theta_{t+1} &= \theta_t + \alpha\mathbb{E}\left[\nabla_\theta \ln \pi(A|S,\theta_t)[q_\pi(S,A) - v_\pi(S)]\right] \\ &\doteq \theta_t + \alpha\mathbb{E}\left[\nabla_\theta \ln \pi(A|S,\theta_t)\delta_\pi(S,A)\right]\end{aligned} \tag{4}$$

其中 $\delta_\pi(S,A) \doteq q_\pi(S,A) - v_\pi(S)$ 为优势函数(advantage function)，称此名称是因为根据定义 $v_\pi(S) = \sum\limits_a \pi(a|S)q_\pi$，因此若 $\delta_\pi > 0$，就说明此 action 比平均的 action 要好，是有优势的.

这样得到梯度上升算法为：

$$\begin{aligned}\theta_{t+1} &= \theta_t + \alpha\nabla_\theta \ln \pi(a_t|s_t,\theta_t)[q_t(s_t,a_t) - v_t(s_t)] = \theta_t + \alpha\nabla_\theta \ln \pi(a_t|s_t,\theta_t)\delta_t(s_t,a_t) \\ &= \theta_t + \alpha\frac{\nabla_\theta\pi(a_t|s_t,\theta_t)}{{}_*\pi(a_t|s_t,\theta_t)}\delta_t(s_t,a_t) = \theta_t + \alpha\underbrace{\left(\frac{\delta_t(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}\right)}_{\text{step size}}\nabla_\theta\pi(a_t|s_t,\theta_t)\end{aligned}$$

同上一节一样，step size 此时是探索(exploration) 和剥削(exploitation)的一个平衡。并且由于我们更关心相对值，因此上一节的 q_t 没有这一节的 δ_t 好.

Trick: 这里有一个小的技巧是将 δ_t 进行替换:

$$\delta_t = q_t(s_t,a_t) - v_t(s_t) \to r_{t+1} + \gamma v_t(s_{t+1}) - v_t(s_t)$$

由于在期望上这二者相同

$$\mathbb{E}[q_\pi(S,A) - v_\pi(S)|S=s_t, A=a_t] = \mathbb{E}\left[R + \gamma v_\pi(S') - v_\pi(S)|S=s_t,A=a_t\right]$$

因此这样替换是合理可行的。并且此时 δ_t 只需要一个神经网络近似 v_t 即可，原先需要两个网络分别近似 q_t 和 v_t.

Algorithm 2 Advantage Actor-Critic (A2C) or TD Actor-Critic(on-policy)

Aim: Search for an optimal policy by maximizing $J(\theta)$.
for each time step t in each episode **do**
  Generate a_t following $\pi(a \mid s_t, \theta_t)$ and then observe r_{t+1}, s_{t+1}.
  TD error (advantage function):

$$\delta_t = r_{t+1} + \gamma v(s_{t+1}, w_t) - v(s_t, w_t)$$

  Critic (value update):

$$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w v(s_t, w_t)$$

  Actor (policy update):

$$\theta_{t+1} = \theta_t + \alpha_\theta \delta_t \nabla_\theta \ln \pi(a_t \mid s_t, \theta_t)$$

end for

Off-policy actor-critic

Review

由于计算梯度时理论期望要使用 $A \sim \pi$(见下式)，因此 behavior policy 和 target policy 相同.

$$\nabla_\theta J(\theta) = \mathbb{E}_{S\sim\eta, A\sim\pi}[*]$$

因此前面介绍的两种 AC 算法都是 off-policy 的，如果想使用一些既有经验，就需要将算法改进为 off-policy 的，方法就是使用 **importance sampling**[a].

[a]importance sampling 也可以在别的地方例如 MC、TD 使用

Importance sampling

Motivating example 首先给出一个例子: 我们已经知道根据大数定律，当未知一个概率分布时可以根据采样对其进行估算，例如想要估计概率分布 p_0，就可以依 p_0 采样得到 samples $\{x_i\}$，然后得到

$$\bar{x} = \sum_{i=1}^{n} \frac{1}{n} x_i \to \mathbb{E}_{X\sim p_0}[X]$$

但是我们希望实现依据一个已知但是不同于 p_0 的分布 p_1 采样，最后估算出 p_0 的期望，这也就对应了 off-policy 中使用与 target policy 不同的 behavior policy 产生策略，但是最后估计的是 target policy

Importance sampling 注意到我们实际上可以使用 p_1 代替 p_0 进行采样，即:

$$\mathbb{E}_{X\sim p_0}[X] = \sum_x p_0(x)x = \sum_x p_1(x) \underbrace{\frac{p_0(x)}{p_1(x)} x}_{f(x)} = \mathbb{E}_{X\sim p_1}[f(X)]$$

这样就可以使用 p_1 采样，计算 $\bar{f}$

$$\bar{f} \doteq \frac{1}{n}\sum_{i=1}^{n} f(x_i), \quad \text{where } x_i \sim p_1$$

最后得到

$$\mathbb{E}_{X\sim p_0}[X] \approx \bar{f} = \frac{1}{n}\sum_{i=1}^{n} f(x_i) = \frac{1}{n}\sum_{i=1}^{n} \frac{p_0(x_i)}{p_1(x_i)} x_i$$

其中 $\frac{p_0(x_i)}{p_1(x_i)}$ 被称为重要性权重(importance weight)，一个直观的解释是若某样本点 x_i 在 p_0 下很容易被采到但是在 p_1 下很难被采到，那么就要很珍惜这个样本点，直观上来说这个样本点的权重应该比较大，这与 $\frac{p_0(x_i)}{p_1(x_i)}$ 此时很大刚好对应.

Note 1. 这里的逻辑应该是这样的：如果可以使用 p_0 进行采样，那么直接根据大数定律使用 $\bar{x} = \sum_{i=1}^{n} \frac{1}{n} x_i \to \mathbb{E}_{X\sim p_0}[X]$ 即可，但是现在我们只能根据 p_1 进行采样，并且同时对于采出来的样本点 $\{x_i\}$，只知道这些离散点的 $p_0(x_i)$（同时知道所有点的 $p_1(x)$），例如使用一个神经网络逼近 p_0，但是此时 p_0 都没有表达式。由于定义式 $\mathbb{E}_{X\sim p_0}[X] = \sum_x p_0(x)x$ 需要知道所有点的 p_0 即连续情形，因此无法计算，最终只能通过转化成 p_1 下有限个点来计算 p_0 的期望。总的来说，就是将本该在 p_0 下使用无限个点（连续情形）才能计算的期望转化为在 p_1 下使用有限个点（离散情形）逼近的期望（大数定律保证结果）.

Off-policy policy gradient

考虑 off-policy 情形，即 behavior policy 不同于 target policy π 时的情况，设此时 begavior policy 为 β。此时 metric 定义为

$$J(\theta) = \sum_{s\in\mathcal{S}} d_\beta(s) v_\pi(s) = \mathbb{E}_{S\sim d_\beta}[v_\pi(S)]$$

其中 d_β 为 policy β 的 stationary distribution.

直接给出 $J(\theta)$ 的梯度如下：

$$\nabla_\theta J(\theta) = \mathbb{E}_{S\sim\rho, A\sim\beta}\left[\frac{\pi(A|S,\theta)}{\beta(A|S)} \nabla_\theta \ln \pi(A|S,\theta) q_\pi(S,A)\right] \tag{5}$$

其中 $\gamma \in (0,1)$ 为 discounted rate, ρ 为 state distribution.(**see details in textbook**)

此时仍然加上 baseline $b(s)$，得到

$$\nabla_\theta J(\theta) = \mathbb{E}_{S\sim\rho, A\sim\beta}\left[\frac{\pi(A|S,\theta)}{\beta(A|S)} \nabla_\theta \ln \pi(A|S,\theta) \left(q_\pi(S,A) - b(S)\right)\right]$$

根据前面的结构，令 $b(s) = v_\pi(s)$，得到

$$\nabla_\theta J(\theta) = \mathbb{E}\left[\frac{\pi(A|S,\theta)}{\beta(A|S)} \nabla_\theta \ln \pi(A|S,\theta) \left(q_\pi(S,A) - v_\pi(S)\right)\right]$$

对应的 stocahtisc 版本即

$$\theta_{t+1} = \theta_t + \alpha_\theta \frac{\pi(a_t|s_t,\theta_t)}{\beta(a_t|s_t)} \nabla_\theta \ln \pi(a_t|s_t,\theta_t) \left(q_t(s_t,a_t) - v_t(s_t)\right)$$

与 on-policy 类似地有

$$q_t(s_t, a_t) - v_t(s_t) \approx r_{t+1} + \gamma v_t(s_{t+1}) - v_t(s_t) \doteq \delta_t(s_t, a_t)$$

如此得到算法

$$\begin{aligned}\theta_{t+1} &= \theta_t + \alpha_\theta \frac{\pi(a_t|s_t, \theta_t)}{\beta(a_t|s_t)} \nabla_\theta \ln \pi(a_t|s_t, \theta_t) \delta_t(s_t, a_t) \\ &= \theta_t + \alpha_\theta \left(\frac{\delta_t(s_t, a_t)}{\beta(a_t|s_t)} \right) \nabla_\theta \pi(a_t|s_t, \theta_t)\end{aligned}$$

Algorithm 3 Off-policy Actor-Critic Based on Importance Sampling

1: **Initialization:** A given behavior policy $\beta(a \mid s)$. A target policy $\pi(a \mid s, \theta_0)$ where θ_0 is the initial parameter vector. A value function $v(s, w_0)$ where w_0 is the initial parameter vector.
2: **Aim:** Search for an optimal policy by maximizing $J(\theta)$.
3: **for** each time step t in each episode **do**
4: Generate a_t following $\beta(s_t)$ and then observe r_{t+1}, s_{t+1}.
5: **TD error (advantage function):**

$$\delta_t = r_{t+1} + \gamma v(s_{t+1}, w_t) - v(s_t, w_t)$$

6: **Critic (value update):**

$$w_{t+1} = w_t + \alpha_w \frac{\pi(a_t \mid s_t, \theta_t)}{\beta(a_t \mid s_t)} \delta_t \nabla_w v(s_t, w_t)$$

7: **Actor (policy update):**

$$\theta_{t+1} = \theta_t + \alpha_\theta \frac{\pi(a_t \mid s_t, \theta_t)}{\beta(a_t \mid s_t)} \delta_t \nabla_\theta \ln \pi(a_t \mid s_t, \theta_t)$$

8: **end for**

Deterministic Actor-Critic(DPG)

Deterministic Actor-Critic

值得注意的是前面介绍的所有的算法的 $\pi(a|s,\theta) > 0$，也就是说都带有 exploration 性，都是 stochastic 策略。但是 $\pi(a|s,\theta) > 0$ 有 action 个数是有限个的隐含要求，因为 $\sum_a \pi(a|s;\theta) = 1$，因此无法处理 action 无限即 continuous action 的情形.

下面考虑 deterministic 的情形，此时重新定义一个 deterministic policy

$$a = \mu(s, \theta) \doteq \mu(s)$$

这与 $\pi(a|s;\theta)$ 不同的是直接输出 action，而不是采取某些 action 对应的概率值.

1. μ 是状态空间到动作空间的映射：$\mu : \mathcal{S} \to \mathcal{A}$
2. μ 实际中可以用参数为 θ 的神经网络替代，输入 s 输出 a

与之前的推导过程类似，首先定义 discounted case 下的 metric

$$J(\theta) = \mathbb{E}[v_\mu(s)] = \sum_{s\in\mathcal{S}} d_0(s) v_\mu(s)$$

其中 $d_0(s)$ 为概率分布，满足 $\sum_{s\in\mathcal{S}} d_0(s) = 1$，最简单的情形是与 μ 不相关（例如只有某 s_0 的 $d_0(s_0) = 1$，或 behavior policy 的 stationary distribution）.

直接给出 discounted case 下的梯度表达式为

$$\begin{aligned}
\nabla_\theta J(\theta) &= \sum_{s\in\mathcal{S}} \rho_\mu(s) \nabla_\theta \mu(s) \left(\nabla_a q_\mu(s,a)\right)|_{a=\mu(s)} \\
&= \mathbb{E}_{S\sim\rho_\mu}\left[\nabla_\theta \mu(S) \left(\nabla_a q_\mu(S,a)\right)|_{a=\mu(S)}\right]
\end{aligned}$$

其中 $\gamma \in (0,1)$ 为 discounted rate, ρ_μ 为 state distribution.(**see details in textbook**)

注意在此梯度的表达式中最后已经没有 a，即对 target policy 的更新已经不再和 behavior policy 相关，因此天然是 off-policy 的.

再代入梯度上升算法中：

$$\theta_{t+1} = \theta_t + \alpha_\theta \mathbb{E}_{S\sim\rho_\mu}\left[\nabla_{\theta\mu(S)} \left(\nabla_a q_\mu(S,a)\right)|_{a=\mu(S)}\right]$$

使用 stochastic 版本:

$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu(s_t) \left(\nabla_a q_\mu(s_t,a)\right)|_{a=\mu(s_t)}$$

Algorithm 4 Deterministic Actor-Critic Algorithm

Initialization: A given behavior policy $\beta(a \mid s)$. A deterministic target policy $\mu(s,\theta_0)$ where θ_0 is the initial parameter vector. A value function $v(s,w_0)$ where w_0 is the initial parameter vector.
Aim: Search for an optimal policy by maximizing $J(\theta)$.
for each time step t in each episode **do**
    Generate a_t following β and then observe r_{t+1}, s_{t+1}.
    TD error:

$$\delta_t = r_{t+1} + \gamma q(s_{t+1}, \mu(s_{t+1},\theta_t), w_t) - q(s_t,a_t,w_t)$$

    Critic (value update):

$$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w q(s_t,a_t,w_t)$$

    Actor (policy update):

$$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu(s_t,\theta_t) \left(\nabla_a q(s_t,a,w_{t+1})\right)\Big|_{a=\mu(s_t)}$$

end for

Note 2. 1. *DPG 算法是 off-policy 的，behavior policy β 可以与 target policy μ 不同*

2. *β 也可以是 μ+noise，这样就有了探索性（此时就是 on-policy 的实现方式）*

3. 当使用线性函数逼近 $q(s,a,w)$ 时，即 $q(s,a,w)=\phi^T(s,a)$，此时算法为 *DPG*（原文中也是如此）
4. 当使用神经网络逼近 $q(s,a,w)$ 时，被称为 *deep deterministic policy gradient(DDPG)*

First updated: February 19, 2025
Last updated: October 3, 2025

References