Subject: Westlake University, Reinforce Learning, Lecture 9, Policy Gradient Methods

Date: from February 14, 2025 to February 16, 2025

Contents

Lecture 9, Policy Gradient Methods

Bilibili:Lecture 9, Policy Gradient Methods

Outline

本节将 policy-based 算法的学习，直接建立一个 policy 的目标函数，通过优化此函数就可以直接得到最优的策略；此前所学习的算法都是 value-based，核心是进行 evaluation 后，根据此 action / state value 不断迭代选择更好的 policy.

value-based methods → policy-based methods
value function approximation → policy function approximation

Basic idea of policy gradient

在此前，policy 都是以表格形式存储的，即不同 state 下采取某 action 的概率都存储在如下表格中：

	a_1	a_2	a_3	a_4	a_5
s_1	$\pi(s_1, a_1)$	$\pi(s_1, a_2)$	$\pi(s_1, a_3)$	$\pi(s_1, a_4)$	$\pi(s_1, a_5)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_9	$\pi(s_9, a_1)$	$\pi(s_9, a_2)$	$\pi(s_9, a_3)$	$\pi(s_9, a_4)$	$\pi(s_9, a_5)$

Table 1: Table of probability of all states

现在我们参考 value function approximation 的想法，也将 policy $\pi(s, a)$ 函数化，将其记为 $\pi(a|s;\theta)$，其中 $\theta \in \mathbb{R}^d$ 为参数向量，也可以记其为 $\pi_\theta(a, s)$ 和 $\pi_\theta(a|s)$。目前常见的做法是使用神经网络函数化 $\pi(s, a)$，网络的参数为 θ，输入为 s，输出为 $\pi(a_1, s; \theta), \cdots$.

Advantage: 1. storage; 2. generalization

Basic idea

1. 定义一个 metric / objective function $J(\theta)$ 以定义最优策略（**How to define this metric?**）
2. （梯度上升）优化目标函数（**How to compute the gradient?**）

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\theta_t)$$

Differences between tabular and function representations

1. 最优策略 π^* 的定义不同
   (a) 表格形式下：策略 π^* 满足 $v_{\pi^*}(s) \geqslant v_\pi(s), \forall s$
   (b) 函数形式下：策略 π^* 最大化了一个预先定义的函数（作为标量度量，scalar metric）
2. 获得某 action 的概率 $\pi(s, a)$ 的方式不同
   (a) 表格形式下：直接查表
   (b) 函数形式下：需要重新计算一遍

3. 如何更新某个概率 $\pi(s,a)$
   (a) 表格形式下：直接更改相应的概率值
   (b) 函数形式下：直接改变参数，从而间接改变相应地概率值

Metrics to define optimal policies

Metric 1: Average value

最简单和直观的想法就是对所有的 state value 取加权平均，这样的 metric 被称为 average state value / average value：

$$\bar{v}_\pi = \sum_{s\in\mathcal{S}} d(s)v_\pi(s) \tag{1}$$

其中 $\bar{v}_\pi$ 为所有 state value 的加权平均，$d(s) \geqslant 0$, $\forall s$，由于 $\sum_{s\in\mathcal{S}} d(s) = 1$ 因此若将 $d \sim S$，那么就有

$$\bar{v}_\pi = \mathbb{E}[v_\pi(S)] = \sum_s p(s)v_\pi(s)$$

将上述公式写成向量形式（方便计算 $\bar{v}_\pi$ 梯度），就有

$$\bar{v}_\pi = \sum_{s\in\mathcal{S}} d(s)v_\pi(s) = d^T v_\pi$$

其中

$$v_\pi = [\ldots, v_\pi(s), \ldots]^T \in \mathbb{R}^{|\mathcal{S}|}, \quad d = [\ldots, d(s), \ldots]^T \in \mathbb{R}^{|\mathcal{S}|}.$$

Case 1: Independent on policy 最简单的情况是 d 的选择不依赖于 π，这样在计算梯度时就不需要对 d 的部分进行计算，此时将其记为 d_0。此时 d_0 也有不同的选择：

1. 将所有的 state 平等对待，即 $d_0(s) = 1/|\mathcal{S}|$
2. 可能只关心某一个状态 s_0（例如固定 s_0 处出发），此时可以赋值 $d_0(s_0) = 1, \quad d_0(s \neq s_0) = 0$.

Case 2: Dependent on policy 第二种情况是 d 的选择依赖于 π。一个常见的做法是 $d = d_\pi(s)$，即 policy π 的 stationary distribution（满足 $d_\pi^T P_\pi = d_\pi^T$）

Metric 2: Average reward

第二种 metric 为 **average one-step reward / average reward**，即 immediate reward 的加权平均：

$$\bar{r}_\pi \doteq \sum_{s\in\mathcal{S}} d_\pi(s)r_\pi(s) = \mathbb{E}[r_\pi(S)] \tag{2}$$

其中 $S \sim d_\pi$，为 stationary distribution，$r_\pi(s)$ 为 state s 处的 immediate reward：

$$r_\pi(s) \doteq \sum_{a\in\mathcal{A}} \pi(a|s)r(s,a)$$

其中

$$r(s,a) = \mathbb{E}[R|s,a] = \sum_r rp(r|s,a)$$

求解过程为：

$$r(s,a) \to r_\pi(s) \to \bar{r}_\pi$$

Usual form: 在论文中，常见的形式是对于一个给定策略 π，已经生成了一个 trajectory，且 reward 为 $(R_{t+1}, R_{t+2}, \ldots)$，此时此 trajectory 的 single-step reward 为

$$\lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[R_{t+1} + R_{t+2} + \cdots + R_{t+n} | S_t = s_0\right] = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\sum_{k=1}^{n} R_{t+k} | S_t = s_0\right]$$

其中 s_0 为初始 state.

值得注意的是无穷多步时初始位置已不再有意义（**see details in textbook**）：

$$\lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\sum_{k=1}^{n} R_{t+k} | S_t = s_0\right] = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\sum_{k=1}^{n} R_{t+k}\right] \overset{\text{大数定律}}{=} \sum_{s} d_\pi(s) r_\pi(s) = \bar{r}_\pi$$

Note 1. *1. 由于上述的 π 都是被 θ 参数化的，因此定义的 metric 都是 θ 的函数*

*2. 实际上我们只介绍了 discounted case $\gamma \in [0,1)$，没有介绍 undiscounted case，虽然在 immediate reward 下二者的 metric 相同，但后者事实上更复杂(**see details in textbook**)*

3. 上面介绍的两种 metric 实际上在 discounted case 下有如下关系：

$$\bar{r}_\pi = (1-\gamma)\bar{v}_\pi$$

因此优化他们的效果是一样的

*4. **Test***

$$\begin{aligned} J(\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1}\right] &= \sum_{s\in\mathcal{S}} d(s)\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} | S_0 = s\right] \\ &= \sum_{s\in\mathcal{S}} d(s) v_\pi(s) = \bar{v}_\pi \end{aligned}$$

Small Summary: 关于两种 metric 都有两种等价的定义

1. **Average value:**

   (a)
   $$J(\theta) = \sum_{s\in\mathcal{S}} d(s) v_\pi(s)$$

   (b)
   $$J(\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1}\right]$$

2. **Average Reward:**

   (a)
   $$J(\theta) = \sum_{s\in\mathcal{S}} d_\pi(s) r_\pi(s)$$

(b)

$$J(\theta) = \lim_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\sum_{k=1}^{n} R_{t+k}|S_t = s_0\right]$$

Gradients of the metrics

Example

事实上计算 metric 的梯度是 policy gradient methods 中最复杂的部分，因为有很多的情况，例如

1. 区分使用的 metric 是 $\bar{v}_\pi, \bar{r}_\pi, \bar{v}_\pi^0$
2. 区分是否是 discounted case

此处仅做简要介绍(**see details in textbook**).

我们直接给出梯度的公式：

$$\nabla_\theta J(\theta) = \sum_{s\in\mathcal{S}} \eta(s) \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a) \tag{3}$$

其中 $J(\theta)$ 可以为 $\bar{v}_\pi, \bar{r}_\pi$ 和 $\bar{v}_\pi^0$；$=$ 可以为严格相等(=)、约等于（$\approx$）与成比例（$\propto$）；η 为分布/ state 的权重.

事实上有

$$\nabla_\theta \bar{r}_\pi \simeq \sum_s d_\pi(s) \sum_a \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$
$$\nabla_\theta \bar{v}_\pi = \frac{1}{1-\gamma} \nabla_\theta \bar{r}_\pi \quad \text{(in discounted case)}$$
$$\nabla_\theta \bar{v}_\pi^0 = \sum_{s\in\mathcal{S}} \rho_\pi(s) \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a)$$

其中第一个式子中 discounted case 是约等于，undiscounted case 是严格等于.

根据 $\nabla_\theta \ln \pi(a|s,\theta) = \frac{\nabla_\theta \pi(a|s,\theta)}{\pi(a|s,\theta)}$ 可以得到如下非常有用的形式

$$\begin{aligned}
\nabla_\theta J(\theta) &= \sum_{s\in\mathcal{S}} \eta(s) \sum_{a\in\mathcal{A}} \nabla_\theta \pi(a|s,\theta) q_\pi(s,a) \\
&= \sum_s d(s) \sum_a \pi(a \mid s,\theta) \nabla_\theta \ln \pi(a \mid s,\theta) q_\pi(s,a) \\
&= \mathbb{E}_{S\sim d}\left[\sum_a \pi(a|S,\theta) \nabla_\theta \ln \pi(a|S,\theta) q_\pi(S,a)\right] \\
&= \mathbb{E}\left[\nabla_\theta \ln \pi(A|S,\theta) q_\pi(S,A)\right] \\
&\approx \nabla_\theta \ln \pi(a|s,\theta) q_\pi(s,a)
\end{aligned}$$

这样做的目的是求期望可以用采样来替代（变成 stochastic gradient）.

Note 2. 为保证 $\ln \pi(a|s,\theta)$ 合理，需要保证 $\pi(a|s,\theta) > 0$，因此 *greedy* 的方式是不可以的，可以使用 *softmax function* $\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_{a'\in\mathcal{A}} e^{h(s,a',\theta)}}$，其中 $h(s,a',\theta)$ 为另外的函数，例如神经网络，直接将其输出层定为 *softmax*.

Gradient-ascent algorithm(REINFORCE)

Gradient-ascent algorithm

最大化 $J(\theta)$ 的梯度上升算法理论上为

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

但是求期望时需要预先知道分布，这可能并不知道，所以就要使用如下随机版本（stochastic gradient）

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

但仍然需要对 q_π （未知）进行替换，得到如下梯度

$$\theta_{t+1}=\theta_t+\alpha\nabla_\theta\ln\pi(a_t|s_t,\theta_t)q_t(s_t,a_t) \tag{4}$$

How to do sampling? 由于我们需要通过采样将期望进行替换，即

$$\mathbb{E}_{S\sim d,A\sim\pi}\left[\nabla_\theta\ln\pi(A|S,\theta_t)q_\pi(S,A)\right]\longrightarrow\nabla_\theta\ln\pi(a|s,\theta_t)q_\pi(s,a)$$

1. 采样 S 时，$S\sim d$，其中 d 是 π 的 long run behavior，但是一般不会这么做，有数据就不错了，不会等那么久
2. 采样 A 时，$A\sim\pi(A|S,\theta)$，a_t 应当来自当前策略 $\pi(\theta_t)$ 下的 state s_t，因此 policy gradient method 是 on-policy 的[a].

How to do interpret this algorithm? 由于 $\nabla_\theta\ln\pi(a_t|s_t,\theta_t)=\frac{\nabla_\theta\pi(a_t|s_t,\theta_t)}{\pi(a_t|s_t,\theta_t)}$，因此式(4)可以写为：

$$\begin{aligned}\theta_{t+1}&=\theta_t+\alpha\nabla_\theta\ln\pi(a_t|s_t,\theta)q_t(s_t,a_t)=\theta_t+\alpha\underbrace{\left(\frac{q_t(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}\right)}_{\beta_t}\nabla_\theta\pi(a_t|s_t,\theta_t)\\&\triangleq\theta_t+\alpha\beta_t\nabla_\theta\pi(a_t|s_t,\theta_t)\end{aligned}\tag{5}$$

这样 $\alpha\beta_t$ 就可以作为新的步长。并且可以看出，实际上式(5)是在优化 π，因为将 $\pi(a_t|s_t,\theta_t)$ 看作是 $f(\theta_t)$，那么显然这就是牛顿法。这就要保证新的步长 $\alpha\beta_t$ 需要很小.

根据微分，当 $\theta_{t+1}-\theta_t$ 非常小时，有

$$\pi(a_t|s_t,\theta_{t+1})\approx\pi(a_t|s_t,\theta_t)+(\nabla_\theta\pi(a_t|s_t,\theta_t))^T(\theta_{t+1}-\theta_t)=\pi(a_t|s_t,\theta_t)+\alpha\beta_t\|\nabla_\theta\pi(a_t|s_t,\theta_t)\|^2$$

因此 $\beta_t>0$ 则 $\pi(a_t|s_t,\theta_{t+1})>\pi(a_t|s_t,\theta_t)$；若 $\beta_t<0$ 则 $\pi(a_t|s_t,\theta_{t+1})<\pi(a_t|s_t,\theta_t)$.

Details about β_t: 事实上 $\beta_t=\frac{q_t(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}$ 是在平衡探索(exploration) 和剥削(exploitation)

1. **exploitation:** 可以看到 β_t 与 $q_t(s_t,a_t)$ 成正比，因此

   $$q_t\text{大}\rightarrow\beta_t\text{大}\rightarrow\pi(a_t|s_t)\text{大}$$

   也就是说对于 action value 更大的 action (q_t 更大)，当前的 policy 倾向于选择它（$\pi_t(a_t|s_t)$）大，因此是一种 exploitation

2. **exploration:** 可以看到 β_t 与 $\pi(a_t|s_t, \theta_t)$ 成反比，因此

$$\pi(a_t|s_t)\text{小} \rightarrow \beta_t\text{大} \rightarrow \pi(a_t|s_t)\text{大}$$

   也就是说当前时间步 t 选择某个 action 的概率小，那么下一时间步 $t+1$ 选择该 action 的概率会变大，这就是一种 exploration.

[a]也有 off-policy 的情况，但是需要额外技巧

REINFORCE

在式(4)中，如果使用了 MC 的方法得到 $q_t(s_t, a_t)$，即采样得到一个 episode，计算相应的 return(g_t)，赋值给 q_t，那么就称其为 **REINFORCE** 算法，其是最早也是最简单的 policy gradient 的算法.

Algorithm 1 Policy Gradient by Monte Carlo (REINFORCE)

Initialization: A parameterized function $\pi(a \mid s, \theta)$, $\gamma \in (0, 1)$, and $\alpha > 0$.
Aim: Search for an optimal policy maximizing $J(\theta)$.
for the kth iteration **do**
  Select s_0 and generate an episode following $\pi(\theta_k)$. Suppose the episode is $\{s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T\}$. ▷ **offline**(because need a full episode)
  for $t = 0, 1, \ldots, T-1$ **do**
    Value update:

$$q_t(s_t, a_t) = \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$$

    Policy update:

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

  end for
  Set $\theta_k = \theta_T$
end for

First updated: February 16, 2025
Second updated: February 18, 2025
Last updated: October 3, 2025

References