第八章 带基线的策略梯度方法

上一章推导出策略梯度，并介绍了两种策略梯度方法——REINFORCE 和 Actor-Critic。虽然上一章的方法在理论上是正确的，但是在实践中效果并不理想。本章介绍的带基线的策略梯度 (Policy Gradient with Baseline) 可以大幅提升策略梯度方法的表现。使用基线 (Baseline) 之后，REINFORCE 变成 REINFORCE with Baseline，Actor-Critic 变成 Advantage Actor-Critic (A2C)。

8.1 策略梯度中的基线

首先回顾上一章的内容。策略学习通过最大化目标函数 $J(\boldsymbol{\theta}) = \mathbb{E}_S[V_\pi(S)]$，训练出策略网络 $\pi(a|s;\boldsymbol{\theta})$。可以用策略梯度 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 来更新参数 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta}_{\text{new}} \;\leftarrow\; \boldsymbol{\theta}_{\text{now}} + \beta \cdot \nabla_{\boldsymbol{\theta}}\, J\big(\boldsymbol{\theta}_{\text{now}}\big).$$

策略梯度定理证明：

$$\nabla_{\boldsymbol{\theta}}\, J\big(\boldsymbol{\theta}\big) \;=\; \mathbb{E}_S\Big[\mathbb{E}_{A\sim\pi(\cdot|S;\boldsymbol{\theta})}\Big[Q_\pi\big(S,A\big) \,\cdot\, \nabla_{\boldsymbol{\theta}}\, \ln\pi\big(A\,\big|\,S;\,\boldsymbol{\theta}\big)\Big]\Big]. \tag{8.1}$$

REINFORCE 和 Actor-Critic 都是通过对策略梯度 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 做近似推导出的；两种方法区别在于具体如何做近似。

8.1.1 基线 (Baseline)

基于策略梯度公式 (8.1) 得出的 REINFORCE 和 Actor-Critic 方法效果通常不好。但是只需对策略梯度公式 (8.1) 做一个微小的改动，就能大幅提升表现：把 b 作为动作价值函数 $Q_\pi(S,A)$ 的基线 (Baseline)，用 $Q_\pi(S,A) - b$ 替换掉 Q_π。设 b 是任意的函数，只要不依赖于动作 A 就可以；例如，b 可以是状态价值函数 $V_\pi(S)$。

定理 8.1. 带基线的策略梯度定理

设 b 是任意的函数，但是 b 不能依赖于 A。把 b 作为动作价值函数 $Q_\pi(S,A)$ 的基线，对策略梯度没有影响：

$$\nabla_{\boldsymbol{\theta}}\, J\big(\boldsymbol{\theta}\big) \;=\; \mathbb{E}_S\Big[\mathbb{E}_{A\sim\pi(\cdot|S;\boldsymbol{\theta})}\Big[\Big(Q_\pi\big(S,A\big) \;-\; b\Big) \,\cdot\, \nabla_{\boldsymbol{\theta}}\, \ln\pi\big(A|S;\,\boldsymbol{\theta}\big)\Big]\Big].$$

♡

定理 8.1 说明 b 的取值不影响策略梯度的正确性。不论是让 $b=0$ 还是让 $b = V_\pi(S)$，对期望的结果毫无影响，期望的结果都会等于 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$。其原因在于

$$\mathbb{E}_S\Big[\mathbb{E}_{A\sim\pi(\cdot|S;\boldsymbol{\theta})}\Big[b \,\cdot\, \nabla_{\boldsymbol{\theta}}\, \ln\pi\big(A|S;\,\boldsymbol{\theta}\big)\Big]\Big] \;=\; 0.$$

定理的证明放到第 8.4 节，对数学感兴趣的读者可以阅读。

定理中的策略梯度表示成了期望的形式，我们对期望做蒙特卡洛近似。从环境中观

测到一个状态 s，然后根据策略网络抽样得到 $a \sim \pi(\cdot|s;\boldsymbol{\theta})$。那么策略梯度 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 可以近似为下面的随机梯度：

$$\boxed{\boldsymbol{g}_b(s,a;\boldsymbol{\theta}) \;=\; \Big[Q_\pi(S,A) \;-\; b\Big] \cdot \nabla_{\boldsymbol{\theta}} \ln \pi(A\,|\,S;\,\boldsymbol{\theta}).}$$

不论 b 的取值是 0 还是 $V_\pi(s)$，得到的随机梯度 $\boldsymbol{g}_b(s,a;\boldsymbol{\theta})$ 都是 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 的无偏估计：

$$\mathsf{Bias} \;=\; \mathbb{E}_{S,A}\Big[\boldsymbol{g}_b(S,A;\,\boldsymbol{\theta})\Big] \;-\; \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \;=\; \mathbf{0}.$$

虽然 b 的取值对 $\mathbb{E}_{S,A}\big[\boldsymbol{g}_b(S,A;\boldsymbol{\theta})\big]$ 毫无影响，但是 b 对随机梯度 $\boldsymbol{g}_b(s,a;\boldsymbol{\theta})$ 是有影响的。用不同的 b，得到的方差

$$\mathsf{Var} \;=\; \mathbb{E}_{S,A}\left[\Big\|\boldsymbol{g}_b(S,A;\,\boldsymbol{\theta}) \;-\; \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\Big\|^2\right]$$

会有所不同。如果 b 很接近 $Q_\pi(s,a)$ 关于 a 的均值，那么方差会比较小。所以 $b = V_\pi(s)$ 是很好的基线。

8.1.2 基线的直观解释

策略梯度公式 (8.1) 期望中的 $Q_\pi(S,A) \cdot \nabla_{\boldsymbol{\theta}} \ln \pi(A|S;\boldsymbol{\theta})$ 的意义是什么呢？以图 8.1 中的左图为例。给定状态 s_t，动作空间是 $\mathcal{A} = \{左, 右, 上\}$，动作价值函数给每个动作打分：

$$Q_\pi\big(s_t, 左\big) \;=\; 80, \qquad Q_\pi\big(s_t, 右\big) \;=\; -20, \qquad Q_\pi\big(s_t, 上\big) \;=\; 180,$$

这些分值会乘到梯度 $\nabla_{\boldsymbol{\theta}} \ln \pi(A|S;\boldsymbol{\theta})$ 上。在做完梯度上升之后，新的策略会倾向于分值高的动作。

- 动作价值 $Q_\pi(s_t, 上) = 180$ 很大，说明基于状态 s_t 选择动作“上”是很好的决策。让梯度 $\nabla_{\boldsymbol{\theta}} \ln \pi(上\,|\,s_t;\boldsymbol{\theta})$ 乘以大的系数 $Q_\pi(s_t, 上) = 180$，那么做梯度上升更新 $\boldsymbol{\theta}$ 之后，会让 $\pi(上\,|\,s_t;\boldsymbol{\theta})$ 变大，在状态 s_t 的情况下更倾向于动作“上”。
- 相反，$Q_\pi(s_t, 右) = -20$ 说明基于状态 s_t 选择动作“右”是糟糕的决策。让梯度 $\nabla_{\boldsymbol{\theta}} \ln \pi(右\,|\,s_t;\boldsymbol{\theta})$ 乘以负的系数 $Q_\pi(s_t, 右) = -20$，那么做梯度上升更新 $\boldsymbol{\theta}$ 之后，会让 $\pi(右\,|\,s_t;\boldsymbol{\theta})$ 变小，在状态 s_t 的情况下选择动作“右”的概率更小。

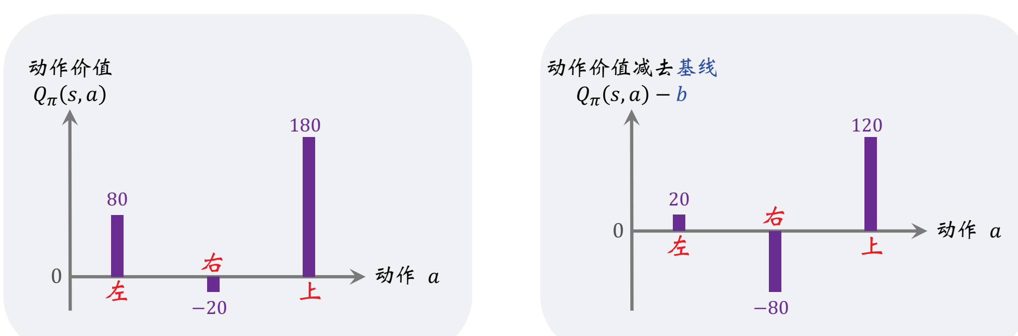

图 8.1: 动作空间是 $\mathcal{A} = \{左, 右, 上\}$。给定状态 s。左图纵轴表示动作价值 $Q_\pi(s,a)$。右图纵轴表示动作价值减去基线 $Q_\pi(s,a) - b$，其中基线 $b = 60$。

根据上述分析，我们在乎的是动作价值 $Q_\pi(s_t, \text{左})$、$Q_\pi(s_t, \text{右})$、$Q_\pi(s_t, \text{上})$ 三者的相对大小，而非绝对大小。如果给三者都减去 $b = 60$，那么三者的相对大小是不变的；动作“上”仍然是最好的，动作“右”仍然是最差的。见图 8.1 中的右图。因此

$$\Big[Q_\pi\big(s_t, a_t\big) - b\Big] \cdot \nabla_{\boldsymbol{\theta}} \ln \pi\big(A \,\big|\, S; \boldsymbol{\theta}\big)$$

依然能指导 $\boldsymbol{\theta}$ 做调整，使得 $\pi(\text{上} \,|\, s_t; \boldsymbol{\theta})$ 变大，而 $\pi(\text{右} \,|\, s_t; \boldsymbol{\theta})$ 变小。

8.2 带基线的 REINFORCE 算法

上一节推导出了带基线的策略梯度，并且对策略梯度做了蒙特卡洛近似。本节中，我们使用状态价值 $V_\pi(s)$ 作基线，得到策略梯度的一个无偏估计：

$$\boldsymbol{g}(s,a;\boldsymbol{\theta}) \;=\; \Big[Q_\pi(s,a) \;-\; V_\pi(s)\Big] \cdot \nabla_{\boldsymbol{\theta}} \ln \pi\big(a \,\big|\, s;\, \boldsymbol{\theta}\big).$$

我们在第 7.4 节中学过 REINFORCE，它使用实际观测的回报 u 来代替动作价值 $Q_\pi(s,a)$。此处我们同样用 u 代替 $Q_\pi(s,a)$。此外，我们还用一个神经网络 $v(s;\boldsymbol{w})$ 近似状态价值函数 $V_\pi(s)$。这样一来，$\boldsymbol{g}(s,a;\boldsymbol{\theta})$ 就被近似成了：

$$\tilde{\boldsymbol{g}}(s,a;\boldsymbol{\theta}) \;=\; \Big[u \;-\; v(s;\,\boldsymbol{w})\Big] \cdot \nabla_{\boldsymbol{\theta}} \ln \pi\big(a \,\big|\, s;\, \boldsymbol{\theta}\big).$$

可以用 $\tilde{\boldsymbol{g}}(s,a;\boldsymbol{\theta})$ 作为策略梯度 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 的近似，更新策略网络参数：

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} \,+\, \beta \cdot \tilde{\boldsymbol{g}}(s,a;\,\boldsymbol{\theta}).$$

8.2.1 策略网络和价值网络

带基线的 REINFORCE 需要两个神经网络：策略网络 $\pi(a|s;\boldsymbol{\theta})$ 和价值网络 $v(s;\boldsymbol{w})$；神经网络结构如图 8.2 和 8.3 所示。策略网络与之前章节一样：输入是状态 s，输出是一个向量，每个元素表示一个动作的概率。

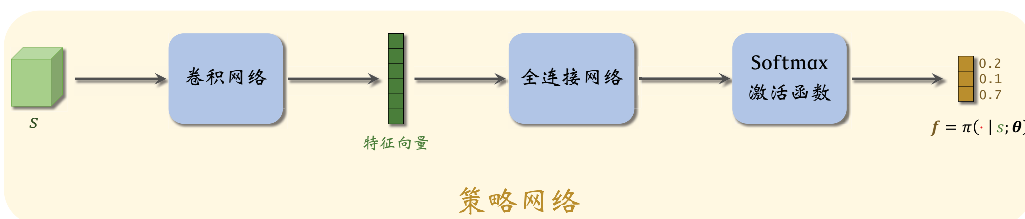

图 8.2: 策略网络 $\pi(a|s;\boldsymbol{\theta})$ 的神经网络结构。输入是状态 s，输出是动作空间中每个动作的概率值。举个例子，动作空间是 $\mathcal{A} = \{左, 右, 上\}$，策略网络的输出是三个概率值：$\pi(左|s;\boldsymbol{\theta}) = 0.2$，$\pi(右|s;\boldsymbol{\theta}) = 0.1$，$\pi(上|s;\boldsymbol{\theta}) = 0.7$。

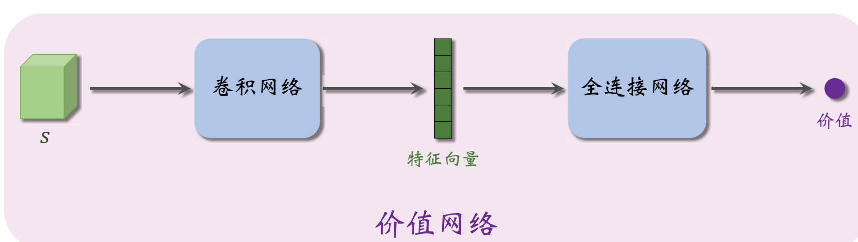

图 8.3: 价值网络 $v(s;\boldsymbol{w})$ 的结构。输入是状态 s；输出是状态的价值。

此处的价值网络 $v(s;\boldsymbol{w})$ 与之前使用的价值网络 $q(s,a;\boldsymbol{w})$ 区别较大。此处的 $v(s;\boldsymbol{w})$ 是对状态价值 V_π 的近似，而非对动作价值 Q_π 的近似。$v(s;\boldsymbol{w})$ 的输入是状态 s，输出是

一个实数，作为基线。策略网络和价值网络的输入都是状态 s，因此可以让两个神经网络共享卷积网络的参数，这是编程实现中常用的技巧。

虽然带基线的 REINFORCE 有一个策略网络和一个价值网络，但是这种方法不是 Actor-Critic。价值网络没有起到"评委"的作用，只是作为基线而已真正帮助策略网络（演员）改进参数 $\boldsymbol{\theta}$（演员的演技）的不是价值网络，而是实际观测到的回报 u。

8.2.2 算法的推导

训练策略网络的方法是近似的策略梯度上升。从 t 时刻开始，智能体完成一局游戏，观测到全部奖励 $r_t, r_{t+1}, \cdots, r_n$，然后计算回报 $u_t = \sum_{k=t}^{n} \gamma^{k-t} \cdot r_k$。让价值网络做出预测 $\widehat{v}_t = v(s_t; \boldsymbol{w})$，作为基线。这样就得到了带基线的策略梯度：

$$\tilde{\boldsymbol{g}}(s_t, a_t; \boldsymbol{\theta}) \;=\; \big(u_t - \widehat{v}_t\big) \cdot \nabla_{\boldsymbol{\theta}} \ln \pi\big(a_t \,\big|\, s_t; \boldsymbol{\theta}\big).$$

它是策略梯度 $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ 的近似。最后做梯度上升更新 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} + \beta \cdot \tilde{\boldsymbol{g}}\big(s_t, a_t; \boldsymbol{\theta}\big).$$

这样可以让目标函数 $J(\boldsymbol{\theta})$ 逐渐增大。

训练价值网络的方法是回归 (Regression)。回忆一下，状态价值是回报的期望：

$$V_\pi(s_t) \;=\; \mathbb{E}\big[U_t \,\big|\, S_t = s_t\big],$$

期望消掉了动作 $A_t, A_{t+1}, \cdots, A_n$ 和状态 $S_{t+1}, \cdots, S_n$。训练价值网络的目的是让 $v(s_t; \boldsymbol{w})$ 拟合 $V_\pi(s_t)$，即拟合 u_t 的期望。定义损失函数：

$$L(\boldsymbol{w}) \;=\; \frac{1}{2n} \sum_{t=1}^{n} \frac{1}{2} \big[v(s_t; \boldsymbol{w}) \;-\; u_t\big]^2.$$

设 $\widehat{v}_t = v(s_t; \boldsymbol{w})$。损失函数的梯度是：

$$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}) \;=\; \frac{1}{n} \sum_{t=1}^{n} \big(\widehat{v}_t - u_t\big) \cdot \nabla_{\boldsymbol{w}} v(s_t; \boldsymbol{w}).$$

做一次梯度下降更新 $\boldsymbol{w}$：

$$\boldsymbol{w} \;\leftarrow\; \boldsymbol{w} \;-\; \alpha \cdot \nabla_{\boldsymbol{w}} L(\boldsymbol{w}).$$

8.2.3 训练流程

当前策略网络的参数是 $\boldsymbol{\theta}_{\text{now}}$，价值网络的参数是 $\boldsymbol{w}_{\text{now}}$。执行下面的步骤，对参数做一轮更新。

1. 用策略网络 $\boldsymbol{\theta}_{\text{now}}$ 控制智能体从头开始玩一局游戏，得到一条轨迹 (Trajectory)：

$$s_1, a_1, r_1, \quad s_2, a_2, r_2, \quad \cdots, \quad s_n, a_n, r_n.$$

2. 计算所有的回报：

$$u_t \;=\; \sum_{k=t}^{n} \gamma^{k-t} \cdot r_k, \qquad \forall\, t = 1, \cdots, n.$$

3. 让价值网络做预测：
$$\widehat{v}_t \;=\; v\big(s_t;\boldsymbol{w}_{\text{now}}\big), \qquad \forall\, t=1,\cdots,n.$$
4. 计算误差 $\delta_t=\widehat{v}_t-u_t$，$\forall\, t=1,\cdots,n$。
5. 用 $\{s_t\}_{t=1}^n$ 作为价值网络输入，做反向传播计算：
$$\nabla_{\boldsymbol{w}}\, v\big(s_t;\, \boldsymbol{w}_{\text{now}}\big), \qquad \forall\, t=1,\cdots,n.$$
6. 更新价值网络参数：
$$\boldsymbol{w}_{\text{new}} \;\leftarrow\; \boldsymbol{w}_{\text{now}} \;-\; \alpha\cdot\sum_{t=1}^{n}\delta_t\cdot\nabla_{\boldsymbol{w}}\, v\big(s_t;\, \boldsymbol{w}_{\text{now}}\big).$$
7. 用 $\{(s_t,a_t)\}_{t=1}^n$ 作为数据，做反向传播计算：
$$\nabla_{\boldsymbol{\theta}}\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}_{\text{now}}\big), \qquad \forall\, t=1,\cdots,n.$$
8. 做随机梯度上升更新策略网络参数：
$$\boldsymbol{\theta}_{\text{new}} \;\leftarrow\; \boldsymbol{\theta}_{\text{now}} \;+\; \beta\;\cdot\;\sum_{t=1}^{n}\gamma^{t-1}\;\cdot\;\underbrace{\delta_t\;\cdot\;\nabla_{\boldsymbol{\theta}}\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}_{\text{now}}\big)}_{\text{即近似梯度 }\tilde{\boldsymbol{g}}(s_t,a_t;\boldsymbol{\theta}_{\text{now}})}.$$

8.3 Advantage Actor-Critic (A2C)

之前我们推导出了带基线的策略梯度，并且对策略梯度做了蒙特卡洛近似，得到得到策略梯度的一个无偏估计：

$$\boldsymbol{g}(s,a;\boldsymbol{\theta}) \;=\; \Big[\underbrace{Q_\pi(s,a) \;-\; V_\pi(s)}_{\text{优势函数}}\Big] \cdot \nabla_{\boldsymbol{\theta}} \ln \pi(a \,|\, s;\boldsymbol{\theta}). \tag{8.2}$$

公式中的 $Q_\pi - V_\pi$ 被称作优势函数 (Advantage Function)。因此，基于上面公式得到的 Actor-Critic 方法被称为 Advantage Actor-Critic，缩写 A2C。

A2C 属于 Actor-Critic 方法。有一个策略网络 $\pi(a|s;\boldsymbol{\theta})$，相当于演员，用于控制智能体运动。还有一个价值网络 $v(s;\boldsymbol{w})$，相当于评委，他的评分可以帮助策略网络（演员）改进技术。两个神经网络的结构与上一节中的完全相同，但是本节和上一节用不同的方法训练两个神经网络。

8.3.1 算法推导

训练价值网络： 训练价值网络 $v(s;\boldsymbol{w})$ 的算法是从贝尔曼公式来的：

$$V_\pi(s_t) \;=\; \mathbb{E}_{A_t\sim\pi(\cdot|s_t;\boldsymbol{\theta})}\Big[\mathbb{E}_{S_{t+1}\sim p(\cdot|s_t,A_t)}\Big[R_t \;+\; \gamma\cdot V_\pi(S_{t+1})\Big]\Big].$$

我们对贝尔曼方程左右两边做近似：

- 方程左边的 $V_\pi(s_t)$ 可以近似成 $v(s_t;\boldsymbol{w})$。$v(s_t;\boldsymbol{w})$ 是价值网络在 t 时刻对 $V_\pi(s_t)$ 做出的估计。
- 方程右边的期望是关于当前时刻动作 A_t 与下一时刻状态 S_{t+1} 求的。给定当前状态 s_t，智能体执行动作 a_t，环境会给出奖励 r_t 和新的状态 s_{t+1}。用观测到的 r_t、s_{t+1} 对期望做蒙特卡洛近似，得到：

$$r_t + \gamma\cdot V_\pi(s_{t+1}). \tag{8.3}$$

- 进一步把公式 (8.3) 中的 $V_\pi(s_{t+1})$ 近似成 $v(s_{t+1};\boldsymbol{w})$，得到

$$\widehat{y}_t \;\triangleq\; r_t + \gamma\cdot v(s_{t+1};\boldsymbol{w}).$$

把它称作 TD 目标。它是价值网络在 $t+1$ 时刻对 $V_\pi(s_t)$ 做出的估计。

$v(s_t;\boldsymbol{w})$ 和 $\widehat{y}_t$ 都是对动作价值 $V_\pi(s_t)$ 的估计。由于 $\widehat{y}_t$ 部分基于真实观测到的奖励 r_t，我们认为 $\widehat{y}_t$ 比 $v(s_t;\boldsymbol{w})$ 更可靠。所以把 $\widehat{y}_t$ 固定住，更新 $\boldsymbol{w}$，使得 $v(s_t;\boldsymbol{w})$ 更接近 $\widehat{y}_t$。

具体这样更新价值网络参数 $\boldsymbol{w}$。定义损失函数

$$L(\boldsymbol{w}) \;\triangleq\; \frac{1}{2}\Big[v(s_t;\boldsymbol{w}) \;-\; y_t\Big]^2.$$

设 $\widehat{v}_t \triangleq v(s_t;\boldsymbol{w})$。损失函数的梯度是：

$$\nabla_{\boldsymbol{w}}\, L(\boldsymbol{w}) \;=\; \underbrace{\big(\widehat{v}_t - \widehat{y}_t\big)}_{\text{TD 误差 } \delta_t}\cdot \nabla_{\boldsymbol{w}}\, v(s_t;\boldsymbol{w}).$$

定义 TD 误差为 $\delta_t \triangleq \widehat{v}_t - \widehat{y}_t$。做一轮梯度下降更新 $\boldsymbol{w}$：

$$\boxed{\boldsymbol{w} \;\leftarrow\; \boldsymbol{w} \;-\; \alpha \cdot \delta_t \;\cdot\; \nabla_{\boldsymbol{w}}\, v\big(s_t;\boldsymbol{w}\big).}$$

这样可以让价值网络的预测 $v(s_t;\boldsymbol{w})$ 更接近 $\widehat{y}_t$。

训练策略网络： A2C 从公式 (8.2) 出发，对 $\boldsymbol{g}(s,a;\boldsymbol{\theta})$ 做近似，记作 $\tilde{\boldsymbol{g}}$，然后用 $\tilde{\boldsymbol{g}}$ 更新策略网络参数 $\boldsymbol{\theta}$。下面我们做数学推导。回忆一下贝尔曼公式：

$$Q_\pi\big(s_t,a_t\big) \;=\; \mathbb{E}_{S_{t+1}\sim p(\cdot|s_t,a_t)}\Big[\,R_t \;+\; \gamma\cdot V_\pi\big(S_{t+1}\big)\Big].$$

把近似策略梯度 $\boldsymbol{g}(s_t,a_t;\boldsymbol{\theta})$ 中的 $Q_\pi(s_t,a_t)$ 替换成上面的期望，得到：

$$\begin{aligned}\boldsymbol{g}\big(s_t,a_t;\,\boldsymbol{\theta}\big) \;&=\; \Big[\,Q_\pi\big(s_t,a_t\big) \;-\; V_\pi\big(s_t\big)\Big] \;\cdot\; \nabla_{\boldsymbol{\theta}}\,\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}\big)\\ &=\; \Big[\mathbb{E}_{S_{t+1}}\Big[\,R_t \;+\; \gamma\cdot V_\pi\big(S_{t+1}\big)\Big] \;-\; V_\pi\big(s_t\big)\Big] \;\cdot\; \nabla_{\boldsymbol{\theta}}\,\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}\big).\end{aligned}$$

当智能体执行动作 a_t 之后，环境给出新的状态 s_{t+1} 和奖励 r_t；利用 s_{t+1} 和 r_t 对上面的期望做蒙特卡洛近似，得到：

$$\boldsymbol{g}\big(s_t,a_t;\,\boldsymbol{\theta}\big) \;\approx\; \Big[\,r_t \;+\; \gamma\cdot V_\pi\big(s_{t+1}\big) \;-\; V_\pi\big(s_t\big)\Big] \;\cdot\; \nabla_{\boldsymbol{\theta}}\,\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}\big).$$

进一步把状态价值函数 $V_\pi(s)$ 替换成价值网络 $v(s;\boldsymbol{w})$，得到：

$$\tilde{\boldsymbol{g}}\big(s_t,a_t;\,\boldsymbol{\theta}\big) \;\triangleq\; \Big[\,\underbrace{r_t \;+\; \gamma\cdot v\big(s_{t+1};\,\boldsymbol{w}\big)}_{\text{TD 目标 }\widehat{y}_t} - v\big(s_t;\,\boldsymbol{w}\big)\Big] \;\cdot\; \nabla_{\boldsymbol{\theta}}\,\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}\big).$$

前面定义了 TD 目标和 TD 误差：

$$\widehat{y}_t \;\triangleq\; r_t \;+\; \gamma\cdot v\big(s_{t+1};\,\boldsymbol{w}\big) \qquad \text{和} \qquad \delta_t \;\triangleq\; v\big(s_{t+1};\,\boldsymbol{w}\big) \;-\; \widehat{y}_t.$$

因此，可以把 $\tilde{\boldsymbol{g}}$ 写成：

$$\boxed{\tilde{\boldsymbol{g}}\big(s_t,a_t;\,\boldsymbol{\theta}\big) \;\triangleq\; -\,\delta_t \;\cdot\; \nabla_{\boldsymbol{\theta}}\,\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}\big).}$$

$\tilde{\boldsymbol{g}}$ 是 $\boldsymbol{g}$ 的近似，所以也是策略梯度 $\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta})$ 的近似。用 $\tilde{\boldsymbol{g}}$ 更新策略网络参数 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} \;+\; \beta\cdot\tilde{\boldsymbol{g}}\big(s_t,a_t;\,\boldsymbol{\theta}\big).$$

这样可以让目标函数 $J(\boldsymbol{\theta})$ 变大。

策略网络与价值网络的关系： A2C 中策略网络（演员）和价值网络（评委）的关系如图 8.4 所示。智能体由策略网络 π 控制，与环境交互，并收集状态、动作、奖励。策略网络（演员）基于状态 s_t 做出动作 a_t。价值网络（评委）基于 s_t、s_{t+1}、r_t 算出 TD 误差 δ_t。策略网络（演员）依靠 δ_t 来判断自己动作的好坏，从而改进自己的演技（即参数 $\boldsymbol{\theta}$）。

读者可能会有疑问：价值网络 v 只知道两个状态 s_t、s_{t+1}，而并不知道动作 a_t，那么价值网络为什么能评价 a_t 的好坏呢？价值网络 v 告诉策略网络 π 的唯一信息是 δ_t。回顾一下 δ_t 的定义：

$$-\delta_t \;=\; \underbrace{r_t \;+\; \gamma\cdot v\big(s_{t+1};\,\boldsymbol{w}\big)}_{\text{TD 误差 }\widehat{y}_t} \;-\; \underbrace{v\big(s_t;\,\boldsymbol{w}\big)}_{\text{基线}}.$$

基线 $v(s_t;\boldsymbol{w})$ 是价值网络在 t 时刻对 $\mathbb{E}[U_t]$ 的估计；此时智能体尚未执行动作 a_t。而 TD 目标 $\widehat{y}_t$ 是价值网络在 $t+1$ 时刻对 $\mathbb{E}[U_t]$ 的估计；此时智能体已经执行动作 a_t。

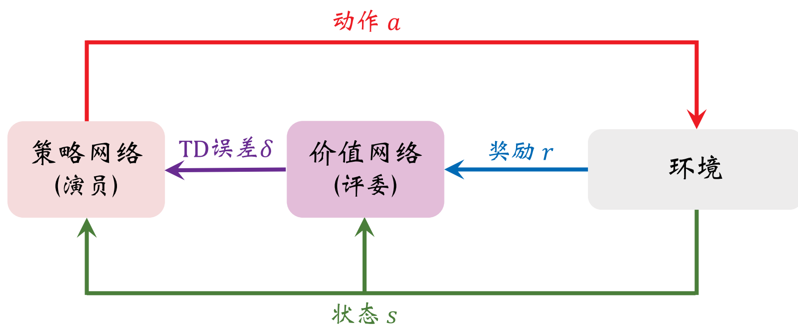

图 8.4: A2C 中策略网络（演员）和价值网络（评委）的关系图。

- 如果 $\widehat{y}_t > v(s_t;\boldsymbol{w})$，说明动作 a_t 很好，使得奖励 r_t 超出预期，或者新的状态 s_{t+1} 比预期好；这种情况下应该更新 $\boldsymbol{\theta}$，使得 $\pi(a_t|s_t;\boldsymbol{\theta})$ 变大。
- 如果 $\widehat{y}_t < v(s_t;\boldsymbol{w})$，说明动作 a_t 不好，导致奖励 r_t 不及预期，或者新的状态 s_{t+1} 比预期差；这种情况下应该更新 $\boldsymbol{\theta}$，使得 $\pi(a_t|s_t;\boldsymbol{\theta})$ 减小。

综上所述，δ_t 中虽然不包含动作 a_t，但是 δ_t 可以间接反映出动作 a_t 的好坏，可以帮助策略网络（演员）改进演技。

8.3.2 训练流程

下面概括 A2C 训练流程。设当前策略网络参数是 $\boldsymbol{\theta}_{\text{now}}$，价值网络参数是 $\boldsymbol{w}_{\text{now}}$。执行下面的步骤，将参数更新成 $\boldsymbol{\theta}_{\text{new}}$ 和 $\boldsymbol{w}_{\text{new}}$：

1. 观测到当前状态 s_t，根据策略网络做决策：$a_t \sim \pi(\cdot\,|\,s_t;\,\boldsymbol{\theta}_{\text{now}})$，并让智能体执行动作 a_t。
2. 从环境中观测到奖励 r_t 和新的状态 s_{t+1}。
3. 让价值网络打分：
$$\widehat{v}_t \;=\; v\big(s_t;\,\boldsymbol{w}_{\text{now}}\big) \qquad \text{和} \qquad \widehat{v}_{t+1} \;=\; v\big(s_{t+1};\,\boldsymbol{w}_{\text{now}}\big)$$
4. 计算 TD 目标和 TD 误差：
$$\widehat{y}_t \;=\; r_t + \gamma\cdot\widehat{v}_{t+1} \qquad \text{和} \qquad \delta_t \;=\; \widehat{v}_t - \widehat{y}_t.$$
5. 更新价值网络：
$$\boldsymbol{w}_{\text{new}} \;\leftarrow\; \boldsymbol{w}_{\text{now}} \;-\; \alpha\cdot\delta_t\cdot\nabla_{\boldsymbol{w}} v\big(s_t;\,\boldsymbol{w}_{\text{now}}\big).$$
6. 更新策略网络：
$$\boldsymbol{\theta}_{\text{new}} \;\leftarrow\; \boldsymbol{\theta}_{\text{now}} \;-\; \beta\cdot\delta_t\cdot\nabla_{\boldsymbol{\theta}}\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}_{\text{now}}\big).$$

注 此处训练策略网络和价值网络的方法属于**同策略** (On-policy)，要求行为策略 (Behavior Policy) 与目标策略 (Target Policy) 相同，都是最新的策略网络 $\pi(a|s;\boldsymbol{\theta}_{\text{now}})$。不能使用经验回放，因为经验回放数组中的数据是用旧的策略网络 $\pi(a|s;\boldsymbol{\theta}_{\text{old}})$ 获取的，不能在当前重复利用。

8.3.3 用目标网络改进训练

上述训练价值网络的算法存在自举——即用价值网络自己的估值 $\widehat{v}_{t+1}$ 去更新价值网络自己。为了缓解自举造成的偏差，可以使用目标网络 (Target Network) 计算 TD 目标。把目标网络记作 $v(s;\boldsymbol{w}^{-})$，它的结构与价值网络的结构相同，但是参数不同。使用目标网络计算 TD 目标，那么 A2C 的训练就变成了：

1. 观测到当前状态 s_t，根据策略网络做决策：$a_t \sim \pi(\cdot\,|\,s_t;\,\boldsymbol{\theta}_{\text{now}})$，并让智能体执行动作 a_t。
2. 从环境中观测到奖励 r_t 和新的状态 s_{t+1}。
3. 让价值网络给 s_t 打分：
$$\widehat{v}_t \;=\; v\big(s_t;\,\boldsymbol{w}_{\text{now}}\big).$$
4. 让目标网络给 s_{t+1} 打分：
$$\widehat{v_{t+1}^{-}} \;=\; v\big(s_{t+1};\,\boldsymbol{w}_{\text{now}}^{-}\big).$$
5. 计算 TD 目标和 TD 误差：
$$\widehat{y_t^{-}} \;=\; r_t + \gamma\cdot\widehat{v_{t+1}^{-}} \qquad \text{和} \qquad \delta_t \;=\; \widehat{v}_t - \widehat{y_t^{-}}\,.$$
6. 更新价值网络：
$$\boldsymbol{w}_{\text{new}} \;\leftarrow\; \boldsymbol{w}_{\text{now}} \;-\; \alpha\cdot\delta_t\cdot\nabla_{\boldsymbol{w}} v\big(s_t;\,\boldsymbol{w}_{\text{now}}\big).$$
7. 更新策略网络：
$$\boldsymbol{\theta}_{\text{new}} \;\leftarrow\; \boldsymbol{\theta}_{\text{now}} \;-\; \beta\cdot\delta_t\cdot\nabla_{\boldsymbol{\theta}}\ln\pi\big(a_t\,\big|\,s_t;\,\boldsymbol{\theta}_{\text{now}}\big).$$
8. 设 $\tau\in(0,1)$ 是需要手动调的超参数。做加权平均更新目标网络的参数：
$$\boldsymbol{w}_{\text{new}}^{-} \;\leftarrow\; \tau\cdot\boldsymbol{w}_{\text{new}} \;+\; \big(1-\tau\big)\cdot\boldsymbol{w}_{\text{now}}^{-}.$$

8.4 证明带基线的策略梯度定理

本节证明带基线的策略梯度定理 8.1。将定理 7.1 与引理 8.2 相结合，即可证得定理 8.1。

引理 8.2

设 b 是任意函数，b 依赖于 A。那么对于任意的 s，

$$\mathbb{E}_{A\sim\pi(\cdot|s;\boldsymbol{\theta})}\left[b\cdot\frac{\partial\ln\pi(A|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right] \;=\; 0.$$

♡

证明 由于基线 b 不依赖于动作 A，可以把 b 提取到期望外面：

$$\begin{aligned}
\mathbb{E}_{A\sim\pi(\cdot|s;\boldsymbol{\theta})}\left[b\cdot\frac{\partial\ln\pi(A|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right] &= b\cdot\mathbb{E}_{A\sim\pi(\cdot|s;\boldsymbol{\theta})}\left[\frac{\partial\ln\pi(A|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right]\\
&= b\cdot\sum_{a\in\mathcal{A}}\pi\big(a\,\big|\,s;\boldsymbol{\theta}\big)\cdot\frac{\partial\ln\pi(a|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\\
&= b\cdot\sum_{a\in\mathcal{A}}\pi\big(a\,\big|\,s;\boldsymbol{\theta}\big)\cdot\frac{1}{\pi(a|s;\boldsymbol{\theta})}\cdot\frac{\partial\pi(a|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\\
&= b\cdot\sum_{a\in\mathcal{A}}\frac{\partial\pi(a|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}.
\end{aligned}$$

上式最右边的连加是关于 a 求的，而偏导是关于 $\boldsymbol{\theta}$ 求的，因此可以把连加放入偏导内部：

$$\mathbb{E}_{A\sim\pi(\cdot|s;\boldsymbol{\theta})}\left[b\cdot\frac{\partial\ln\pi(A|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right] \;=\; b\cdot\frac{\partial}{\partial\boldsymbol{\theta}}\underbrace{\sum_{a\in\mathcal{A}}\pi(a|s;\boldsymbol{\theta})}_{\text{恒等于 }1}.$$

因此

$$\mathbb{E}_{A\sim\pi(\cdot|s;\boldsymbol{\theta})}\left[b\cdot\frac{\partial\ln\pi(A|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right] \;=\; b\cdot\frac{\partial\,1}{\partial\boldsymbol{\theta}} \;=\; 0.$$

□