第十七章 模仿学习

模仿学习 (Imitation Learning) 不是强化学习，而是强化学习的一种替代品。模仿学习与强化学习有相同的目的：两者的目的都是学习策略网络，从而控制智能体。模仿学习与强化学习有不同的原理：模仿学习向人类专家学习，目标是让策略网络做出的决策与人类专家相同；而强化学习利用环境反馈的奖励改进策略，目标是让累计奖励（即回报）最大化。

本章介绍三种常见的模仿学习方法：行为克隆 (Behavior Cloning)、逆向强化学习 (Inverse Reinforcement Learning)、生成判别模仿学习 (GAIL)。行为克隆不需要让智能体与环境交互，因此学习的“成本”很低；而逆向强化学习、生成判别模仿学习则需要让智能体与环境交互。

17.1 行为克隆

行为克隆 (Behavior Cloning) 是最简单的模仿学习。行为克隆的目的是模仿人的动作，学出一个随机策略网络 $\pi(a|s;\boldsymbol{\theta})$ 或者确定策略网络 $\mu(s;\boldsymbol{\theta})$。虽然行为克隆的目的与强化学习中的策略学习类似，但是行为克隆的本质是监督学习（分类或者回归），而不是强化学习。行为克隆通过模仿人类专家的动作来学习策略，而强化学习则是从奖励中学习策略。

模仿学习需要一个事先准备好的数据集，由（状态，动作）这样的二元组构成，记作：

$$\mathcal{X} = \Big\{ (s_1, a_1), \cdots, (s_n, a_n) \Big\}.$$

其中 s_j 是一个状态，而对应的 a_j 是人类专家基于状态 s_j 做出的动作。可以把 s_j 和 a_j 分别视作监督学习中的输入和标签。

17.1.1 连续控制问题

连续控制的意思是动作空间 $\mathcal{A}$ 是连续集合，比如 $\mathcal{A} = [0, 360] \times [0, 180]$。我们搭建类似图 17.2 的确定策略网络，记作 $\boldsymbol{\mu}(s;\boldsymbol{\theta})$。输入是状态 s，输出是动作向量 $\boldsymbol{a}$，它的维度 d 是控制问题的自由度。

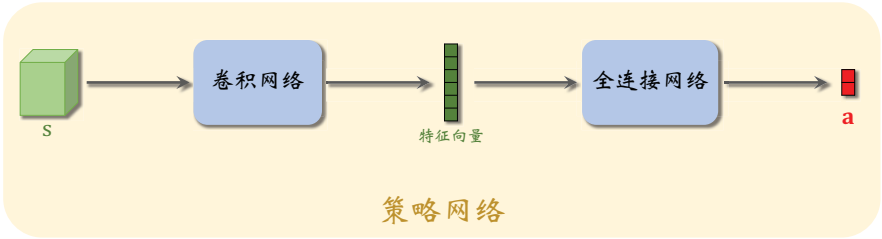


图 17.1: 确定策略网络 $\boldsymbol{\mu}(s;\boldsymbol{\theta})$ 的结构。输入是状态 s，输出是动作 $\boldsymbol{a}$。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

行为克隆用回归的方法训练确定策略网络。训练数据集 $\mathcal{X}$ 中的二元组 $(s, \boldsymbol{a})$ 的意思是基于状态 s，人做出动作 $\boldsymbol{a}$。行为克隆鼓励策略网络的决策 $\boldsymbol{\mu}(s;\boldsymbol{\theta})$ 接近人做出的动作 $\boldsymbol{a}$。定义损失函数

$$L(s, \boldsymbol{a}; \boldsymbol{\theta}) \triangleq \frac{1}{2}\Big[\boldsymbol{\mu}(s;\boldsymbol{\theta}) - \boldsymbol{a}\Big]^2.$$

损失函数越小，说明策略网络的决策越接近人的动作。用梯度更新 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \cdot \nabla_{\boldsymbol{\theta}} L(s, \boldsymbol{a}; \boldsymbol{\theta}),$$

这样可以让 $\boldsymbol{\mu}(s;\boldsymbol{\theta})$ 更接近 $\boldsymbol{a}$。

训练流程： 给定数据集 $\mathcal{X} = \{(s_j, \boldsymbol{a}_j)\}_{j=1}^n$。重复下面的随机梯度下降，直到算法收敛：

1. 从序号 $\{1, \cdots, n\}$ 中做均匀随机抽样，把抽到的序号记作 j。
2. 设当前策略网络参数为 $\boldsymbol{\theta}_{\text{now}}$。把 s_j、$\boldsymbol{a}_j$ 作为输入，做反向传播计算梯度，然后用梯度更新 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta}_{\text{new}} \leftarrow \boldsymbol{\theta}_{\text{now}} - \beta \cdot \nabla_{\boldsymbol{\theta}} L(s_j, \boldsymbol{a}_j; \boldsymbol{\theta}_{\text{now}}).$$

17.1.2 离散控制问题

离散控制的意思是动作空间 $\mathcal{A}$ 是离散集合，例如 $\mathcal{A} = \{\text{左}, \text{右}, \text{上}\}$。我们搭建类似图 17.2 的策略网络，记作 $\pi(a|s;\boldsymbol{\theta})$。输入是状态 s，输出记作向量 $\boldsymbol{f}$。$\boldsymbol{f}$ 的维度是 $|\mathcal{A}|$，它的每个元素对应一个动作，表示选择该动作的概率值。比如给定状态 s，策略网络输出：

$$\begin{aligned} f_1 &= \pi(\text{左} \,|\, s; \boldsymbol{\theta}) = 0.2, \\ f_2 &= \pi(\text{右} \,|\, s; \boldsymbol{\theta}) = 0.1, \\ f_3 &= \pi(\text{上} \,|\, s; \boldsymbol{\theta}) = 0.7. \end{aligned}$$

也就是说向量 $\boldsymbol{f} = [0.2, 0.1, 0.7]^T$。

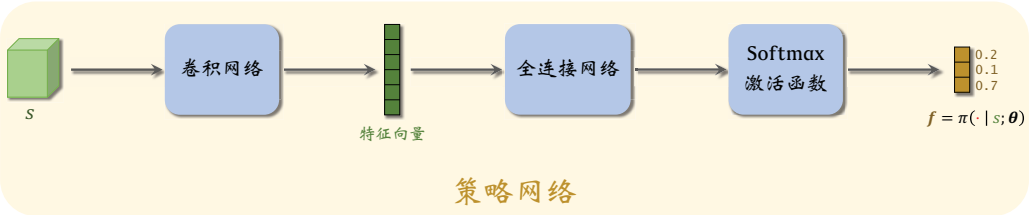


图 17.2: 策略网络 $\pi(a|s;\boldsymbol{\theta})$ 的神经网络结构。

行为克隆把策略网络 $\pi(a|s;\boldsymbol{\theta})$ 看做一个多类别分类器，用监督学习的方法训练这个分类器。把训练数据集 $\mathcal{X}$ 中的动作 a 看做类别标签，用于训练分类器。需要对类别标签 a 做 One-Hot 编码，得到 $|\mathcal{A}|$ 维的向量，记作粗体字母 $\bar{\boldsymbol{a}}$。例如 $\mathcal{A} = \{\text{左}, \text{右}, \text{上}\}$，那么

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

对动作的 One-Hot 编码就是：

$$\begin{aligned} a = \text{左} \quad &\Longrightarrow \quad \bar{\boldsymbol{a}} = \big[1; 0; 0\big], \\ a = \text{右} \quad &\Longrightarrow \quad \bar{\boldsymbol{a}} = \big[0; 1; 0\big], \\ a = \text{上} \quad &\Longrightarrow \quad \bar{\boldsymbol{a}} = \big[0; 0; 1\big]. \end{aligned}$$

向量 $\bar{\boldsymbol{a}}$ 与 $\boldsymbol{f}$ 都可以看做是离散的概率分布，可以用交叉熵 (Cross Entropy) 衡量两个分布的区别。交叉熵的定义是：

$$H\big(\bar{\boldsymbol{a}}, \boldsymbol{f}\big) \triangleq -\sum_{i=1}^{|\mathcal{A}|} \bar{a}_i \cdot \ln f_i.$$

向量 $\bar{\boldsymbol{a}}$ 与 $\boldsymbol{f}$ 越接近，它们的交叉熵越小。用交叉熵作为损失函数：

$$H\Big[\bar{\boldsymbol{a}}, \pi(\cdot \,|\, s; \boldsymbol{\theta})\Big],$$

用梯度更新参数 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \cdot \nabla_{\boldsymbol{\theta}} H\Big[\bar{\boldsymbol{a}}, \pi(\cdot \,|\, s; \boldsymbol{\theta})\Big].$$

这样可以使交叉熵减小，也就是说策略网络做出的决策 $\boldsymbol{f}$ 更接近人的动作 $\bar{\boldsymbol{a}}$。

训练流程： 给定数据集 $\mathcal{X} = \{(s_j, a_j)\}_{j=1}^{n}$，对所有的 a_j 做 One-Hot 编码，变成向量 $\bar{\boldsymbol{a}}_j$。重复下面的随机梯度下降，直到算法收敛：

1. 从序号 $\{1, \cdots, n\}$ 中做均匀随机抽样，把抽到的序号记作 j。
2. 设当前策略网络的参数是 $\boldsymbol{\theta}_{\text{now}}$。把 s_j、$\bar{\boldsymbol{a}}_j$ 作为输入，做反向传播计算梯度，然后用梯度更新 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta}_{\text{new}} \leftarrow \boldsymbol{\theta}_{\text{now}} - \beta \cdot \nabla_{\boldsymbol{\theta}} H\Big[\bar{\boldsymbol{a}}_j, \pi(\cdot \,|\, s_j; \boldsymbol{\theta}_{\text{now}})\Big].$$

17.1.3 行为克隆与强化学习的对比

行为克隆不是强化学习。强化学习让智能体与环境交互，用环境反馈的奖励指导策略网络的改进，目的是最大化回报的期望。而行为克隆不需要与环境交互，而是利用事先准备好的数据集，用人类的动作指导策略网络的改进，目的是让策略网络的决策更像人类的决策。行为克隆的本质是监督学习（分类或者回归），而不是强化学习，因为行为克隆不需要与环境交互。

行为克隆训练出的策略网络通常效果不佳。人类不会探索奇怪的状态和动作，因此数据集上的状态和动作缺乏多样性。在数据集上做完行为克隆之后，智能体面对真实的环境，可能会见到陌生的状态，智能体的决策可能会很糟糕。行为克隆存在“错误累加”的缺陷。假如当前智能体的决策 a_t 不够好。那么下一时刻的状态 s_{t+1} 可能会比较罕见，于是智能体的决策 a_{t+1} 会很差；这又导致状态 s_{t+2} 非常奇怪，使得决策 a_{t+2} 更糟糕。行为克隆训练出的策略常会进入这种恶性循环。

强化学习效果通常优于行为克隆。如果用强化学习，那么智能体探索过各种各样的状态，尝试过各种各样的动作，知道面对各种状态时应该做什么决策。智能体通过探索，各种状态都见过，比行为克隆有更多的“人生经验”，因此表现会更好。强化学习在围棋、

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

电子游戏上的表现可以远超顶级人类玩家，而行为克隆却很难超越人类高手。

强化学习的一个缺点在于需要与环境交互，需要探索，而且会改变环境。举个例子，假如把强化学习应用到手术机器人，从随机初始化开始训练策略网络，至少要致死、致残几万个病人才能训练好策略网络。假如把强化学习应用到无人车，从随机初始化开始训练策略网络，至少要撞毁几万辆无人车才能训练好策略网络。假如把强化学习应用到广告投放，那么从初始化到训练好策略网络期间需要做探索，投放的广告会很随机，会严重降低广告收入。如果在真实物理世界应用强化学习，要考虑初始化和探索带来的成本。

行为克隆的优势在于离线训练，可以避免与真实环境的交互，不会对环境产生影响。假如用行为克隆训练手术机器人，只需要把人类医生的观测和动作记录下来，离线训练手术机器人，而不需要真的在病人身上做实验。尽管行为克隆效果不如强化学习，但是行为克隆的成本低。可以先用行为克隆初始化策略网络，而不是随机初始化，然后再做强化学习，这样可以减小对物理世界的有害影响。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

17.2 逆向强化学习

逆向强化学习 (Inverse Reinforcement Learning，缩写 IRL) 非常有名，但是在今天已经不常用了。下一节介绍的 GAIL 更简单，效果更好。本节只简单介绍 IRL 的主要思想，而不深入讲解其数学原理。

IRL 的基本设定： 第一，IRL 假设智能体可以与环境交互[1]，环境会根据智能体的动作更新状态，但是不会给出奖励。智能体与环境交互的轨迹是这样的：

$$s_1, a_1, \quad s_2, a_2, \quad s_3, a_3, \quad \cdots, \quad s_n, a_n.$$

这种设定非常符合物理世界的实际情况。比如人类驾驶汽车，与物理环境交互，根据观测做出决策，得到上面公式中轨迹，轨迹中没有奖励。是不是汽车驾驶问题中没有奖励呢？其实是有奖励的。避免碰撞、遵守交通规则、尽快到达目的地，这些操作背后都有隐含的奖励，只是环境不会直接把奖励告诉我们而已。把奖励看做 (s_t, a_t) 的函数，记作 $R^\star(s_t, a_t)$。

第二，IRL 假设我们可以把人类专家的策略 $\pi^\star(a|s)$ 作为一个黑箱调用。黑箱的意思是我们不知道策略的解析表达式，但是可以使用黑箱策略控制智能体与环境交互，生成轨迹。IRL 假设人类学习策略 $\pi^\star$ 的方式与强化学习相同，都是最大化回报（即累计奖励）的期望，即

$$\pi^\star \quad = \quad \max_{\pi} \mathbb{E}_{S_t, A_t, \cdots, S_n, A_n} \left[\sum_{k=t}^{n} \gamma^{k-t} \cdot R^\star(S_k, A_k) \right]. \tag{17.1}$$

因为 $\pi^\star$ 与奖励函数 $R^\star(s, a)$ 密切相关，所以可以从 $\pi^\star$ 反推出 $R^\star(s, a)$。

IRL 的基本思想： IRL 的目的是学到一个策略网络 $\pi(a|s; \boldsymbol{\theta})$，模仿人类专家的黑箱策略 $\pi^\star(a|s)$。如图 17.3 所示，IRL 首先从 $\pi^\star(a|s)$ 中学习其隐含的奖励函数 $R^\star$，然后利用奖励函数做强化学习，得到策略网络的参数 $\boldsymbol{\theta}$。我们用神经网络 $R(s, a; \boldsymbol{\rho})$ 来近似奖励函数 $R^\star$。神经网络 R 的输入是 s 和 a，输出是实数；我们需要学习它的参数 $\boldsymbol{\rho}$。

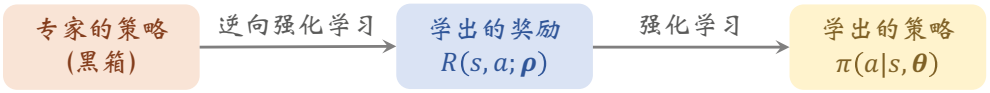


图 17.3

从黑箱策略反推奖励： 假设人类专家的黑箱策略 $\pi^\star(a|s)$ 满足公式 (17.1)，即 $\pi^\star$ 是应对奖励函数 $R^\star$ 的最优策略。对于不同的奖励函数 $R^\star$，则会有不同的 $\pi^\star(a|s)$。是否能由 $\pi^\star$ 的决策反推出 $R^\star$ 呢？举个例子，图 17.4 是走格子的游戏，动作空间是 $\mathcal{A} = \{$上, 下, 左, 右$\}$。两个表格表示两局游戏的状态，蓝色的箭头表示 $\pi^\star$ 做出的决策。请读者仔细观察，尝试推断游戏的奖励函数 $R^\star$。

既然蓝色箭头是最优策略做出的决策，那么沿着蓝色箭头走，可以最大化回报。我

[1]注意，上一节的行为克隆无需智能体与环境交互。

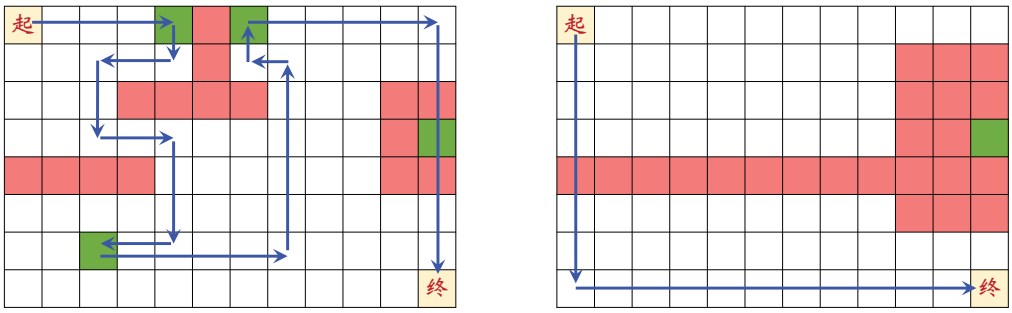
《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

们不难做出以下推断：

- 到达绿色格子有正奖励 r_+，原因是智能体尽量通过绿色格子。到达绿色格子的奖励只能被收集一次，否则智能体会反复回到绿色格子。
- 到达红色格子有负奖励 $-r_-$，因为智能体尽量避开红色格子。由于左图中智能体穿越两个红色格子去收集绿色奖励，说明 $r_+ \gtrsim 2r_-$。由于右图中智能体没有穿越四个红格子去收集绿色奖励，而是穿越一个红格子，说明 $r_+ \lesssim 3r_-$。
- 到达终点有正奖励 r_*，因为智能体会尽力走到终点。由于右图中的智能体穿过红色格子，说明 $r_* > r_-$。
- 智能体尽量走最短路，说明每走一步，有一个负奖励 $-r_\rightarrow$。但是 $r_\rightarrow$ 比较小，否则智能体不会绕路去收集绿色奖励。

注意，从智能体的轨迹中，只能大致推断出奖励函数，但是不可能推断出奖励 r_+、$-r_-$、r_*、$r_\rightarrow$ 具体的大小。把四个奖励的数值同时乘以 10，根据新的奖励训练策略，最终学出的最优策略跟原来相同；这说明最优策略对应的奖励函数是不唯一的。



(a) (b)

图 17.4: 左右两张图表示走格子游戏的两个状态，图中蓝色箭头表示智能体的轨迹。

用奖励函数训练策略网络： 假设我们已经学到了奖励函数 $R(s,a;\boldsymbol{\rho})$，那么就可以用它来训练一个策略网络。。用策略网络 $\pi(a|s;\boldsymbol{\theta}_{\text{now}})$ 控制智能体与环境交互，每次得到这样一条轨迹：

$$s_1, a_1, \quad s_2, a_2, \quad s_3, a_3, \quad \cdots, \quad s_n, a_n,$$

轨迹中没有没有奖励。比如用策略网络控制无人车驾驶，得到的就是这样一条没有奖励的轨迹。好在我们已经从人类专家身上学到了奖励函数 $R(s,a;\boldsymbol{\rho})$，可以用 R 算出奖励：

$$\widehat{r}_t \;=\; R\big(s_t, a_t; \boldsymbol{\rho}\big), \qquad \forall\, t = 1, \cdots, n.$$

可以用任意策略学习方法更新策略网络参数 $\boldsymbol{\theta}$，比如用 REINFORCE：

$$\boldsymbol{\theta}_{\text{new}} \;\leftarrow\; \boldsymbol{\theta}_{\text{now}} \;+\; \beta \cdot \sum_{t=1}^{n} \gamma^{t-1} \cdot \widehat{u}_t \cdot \nabla_{\boldsymbol{\theta}} \ln \pi\big(a|s;\boldsymbol{\theta}_{\text{now}}\big).$$

公式中的 $\widehat{u}_t \triangleq \sum_{k=t}^{n} \gamma^{k-t} \cdot \widehat{r}_k$ 是近似回报。

更新奖励函数： 具体该如何学习奖励函数 $R(s,a;\boldsymbol{\rho})$ 呢？因为我们用 $R(s,a;\boldsymbol{\rho})$ 来训练策略网络 $\pi(a|s;\boldsymbol{\theta})$，所以 $\pi(a|s;\boldsymbol{\theta})$ 依赖于 $\boldsymbol{\rho}$。IRL 的目标是让 $\pi(a|s;\boldsymbol{\theta})$ 尽量接近人类

专家的策略 $\pi^\star(a|s)$。因此要寻找参数 $\boldsymbol{\rho}$ 使得学到的 $\pi(a|s;\boldsymbol{\theta})$ 最接近 $\pi^\star(a|s)$。学习 $\boldsymbol{\rho}$ 的方法有很多种，本书不具体介绍了，有兴趣的读者可以阅读相关的文献。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

17.3 生成判别模仿学习 (GAIL)

生成判别模仿学习 (Generative Adversarial Imitation Learning，缩写 GAIL) 需要让智能体与环境交互，但是无法从环境获得奖励[2]。GAIL 还需要收集人类专家的决策记录（即很多条轨迹）。GAIL 的目标是学习一个策略网络，使得判别器无法区分一条轨迹是策略网络的决策还是人类专家的决策。

17.3.1 生成判别网络 (GAN)

GAIL 的设计基于生成判别网络 (Generative Adversarial Network，缩写 GAN)。本小节简单介绍 GAN 的基础知识。生成器 (Generator) 和判别器 (Discriminator) 各是一个神经网络。生成器负责生成假的样本，而判别器负责判定一个样本是真是假。举个例子，在人脸数据集上训练生成器和判别器，那么生成器的目标是生成假的人脸图片，可以骗过判别器；而判别器的目标是判断一张图片是真实的还是生成的。理想情况下，当训练结束的时候，判别器的分类准确率是 50%，意味着生成器的输出已经以假乱真。

生成器记作 $a = G(s;\boldsymbol{\theta})$，其中 $\boldsymbol{\theta}$ 是参数。它的输入是向量 s，向量的每一个元素从均匀分布 $\mathcal{U}(-1,1)$ 或标准正态分布 $\mathcal{N}(0,1)$ 中抽取。生成器的输出是数据（比如图片）x。生成器通常是一个深度神经网络，其中可能包含卷积层 (Convolution)、反卷积层 (Transposed Convolution)、上采样层 (Upsampling)、全连接层 (Dense) 等。生成器的具体实现取决于具体的问题。

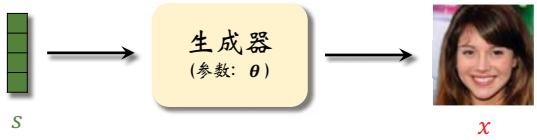


图 17.5: 生成器 $a = G(s;\boldsymbol{\theta})$。

判别器记作 $\hat{p} = D(x;\boldsymbol{\phi})$，其中 $\boldsymbol{\phi}$ 是参数。它的输入是图片 x；输出 $\hat{p}$ 是介于 0 到 1 之间的概率值，0 表示“假的”，1 表示“真的”。判别器的功能是二分类器，实现方法很简单。判别器主要由卷积层、池化层 (Pooling)、全连接层等组成。

图 17.6: 判别器 $\hat{p} = D(x;\boldsymbol{\phi})$。

图 17.7: 训练生成器 $G(s;\boldsymbol{\theta})$。

训练生成器： 将生成器与判别器相连，如图 17.7 所示。固定住判别器的参数，只更

[2]GAIL 和 IRL 都需要让智能体与环境交互，而行为克隆不需要。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

新生成器的参数 $\boldsymbol{\theta}$，使得生成的图片 $x = G(s;\boldsymbol{\theta})$ 在判别器的眼里更像真的。对于任意一个随机生成的向量 s，应该改变 $\boldsymbol{\theta}$，使得判别器的输出 $\widehat{p} = D(x;\boldsymbol{\phi})$ 尽量接近 1。可以用交叉熵作为损失函数：

$$E(s;\boldsymbol{\theta}) \;=\; \ln\Big[1 - \underbrace{D(x;\boldsymbol{\phi})}_{\text{越大越好}}\Big]; \qquad \text{s.t. } x = G(s;\boldsymbol{\theta}).$$

判别器的输出 $\widehat{p} = D(x;\boldsymbol{\phi})$ 是介于 0 到 1 之间的数。$\widehat{p}$ 越接近 1，则损失函数 $E(s;\boldsymbol{\theta}) = \ln(1-\widehat{p})$ 越小。训练生成器参数 $\boldsymbol{\theta}$ 的时候，我们希望 $\widehat{p}$ 尽量接近 1，所以应当更新 $\boldsymbol{\theta}$ 使得 $E(s;\boldsymbol{\theta})$ 减小。做一次梯度下降更新 $\boldsymbol{\theta}$：

$$\boldsymbol{\theta} \;\leftarrow\; \boldsymbol{\theta} - \beta \cdot \nabla_{\boldsymbol{\theta}} E(s;\boldsymbol{\theta}).$$

此处的 β 是学习率，需要用户手动调。

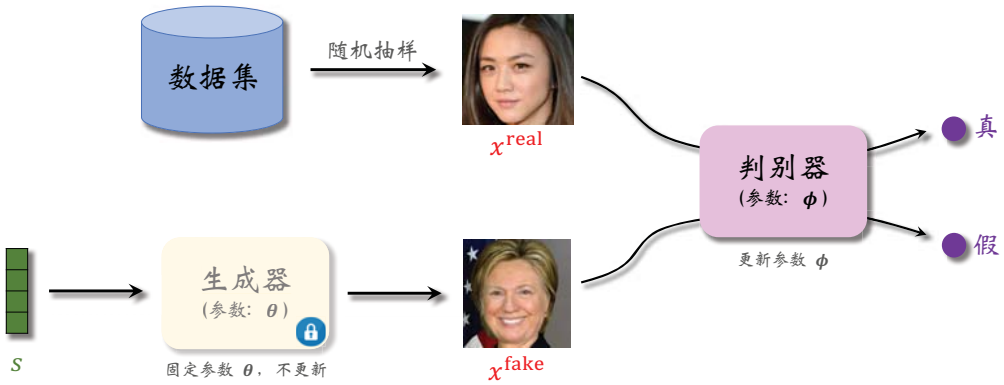


图 17.8: 训练判别器 $D(x;\boldsymbol{\phi})$。

训练判别器： 判别器的本质是个二分类器，它的输出值 $\widehat{p} = D(x;\boldsymbol{\phi})$ 表示对真伪的预测；$\widehat{p}$ 接近 1 表示“真”，$\widehat{p}$ 接近 0 表示“假”。判别器的训练如图 17.8 所示。从真实数据集中抽取一个样本，记作 x^{real}。再随机生成一个向量 s，用生成器生成 $x^{\text{fake}} = G(s;\boldsymbol{\theta})$。训练判别器的目标是改进参数 $\boldsymbol{\phi}$，让 $D(x^{\text{real}};\boldsymbol{\phi})$ 更接近 1（真），让 $D(x^{\text{fake}};\boldsymbol{\phi})$ 更接近 0（假）。也就是说让判别器的分类结果更准确，更好区分真实图片和生成的假图片。可以用交叉熵作为损失函数：

$$F(x^{\text{real}}, x^{\text{fake}};\boldsymbol{\phi}) \;=\; \ln\Big[1 - \underbrace{D(x^{\text{real}};\boldsymbol{\phi})}_{\text{越大越好}}\Big] + \underbrace{\ln D(x^{\text{fake}};\boldsymbol{\phi})}_{\text{越小越好}}.$$

判别器的判断越准确，则损失函数 $F(\boldsymbol{\theta})$ 越小。为什么呢?

- 判别器越相信 x^{real} 为真，则 $D(x^{\text{real}};\boldsymbol{\phi})$ 越大，那么等式右边第一项 $\ln[1-D(x^{\text{real}};\boldsymbol{\phi})]$ 越小。
- 判别器越相信 x^{fake} 为假，则 $D(x^{\text{fake}};\boldsymbol{\phi})$ 越小，那么等式右边第二项 $\ln D(x^{\text{real}};\boldsymbol{\phi})$ 也越小。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

为了减小损失函数 $F(\boldsymbol{\phi})$，可以做一次梯度下降更新判别器参数 $\boldsymbol{\phi}$：

$$\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \eta \cdot \nabla_{\boldsymbol{\phi}} F\big(x^{\text{real}}, x^{\text{fake}}; \boldsymbol{\phi}\big).$$

此处的 η 是学习率，需要用户手动调。

批量随机梯度 (Mini-Batch SGD)： 上述训练生成器和判别器的方式其实是随机梯度下降 (SGD)，每次只用一个样本。实践中，应该每次用一个批量 (Batch) 的样本，比如用 $b = 16$ 个，那么会计算出 b 个梯度。用 b 个梯度的平均去更新生成器和判别器。

训练流程： 实践中，要同时训练生成器和判别器，让两者同时进步。[3] 每一轮要更新一次生成器，更新一次判别器。设当前生成器、判别器的参数分别为 $\boldsymbol{\theta}_{\text{now}}$ 和 $\boldsymbol{\phi}_{\text{now}}$。

1. （从均匀分布或正态分布中）随机抽样 b 个向量：$s_1, \cdots, s_b$。
2. 用生成器生成假样本：$x_j^{\text{fake}} = G\big(s_j; \boldsymbol{\theta}_{\text{now}}\big)$，$\forall\, j = 1, \cdots, b$。
3. 从训练数据集中随机抽样 b 个真样本：$x_1^{\text{real}}, \cdots, x_b^{\text{real}}$。
4. 更新生成器 $G(s; \boldsymbol{\theta})$ 的参数：
   (a). 计算平均梯度：
   $$\boldsymbol{g}_{\theta} = \frac{1}{b}\sum_{j=1}^{b} \nabla_{\boldsymbol{\theta}} E\Big(s_j; \boldsymbol{\theta}_{\text{now}}\Big).$$
   (b). 做梯度下降更新生成器参数：$\boldsymbol{\theta}_{\text{new}} \leftarrow \boldsymbol{\theta}_{\text{now}} - \beta \cdot \boldsymbol{g}_{\theta}$。
5. 更新判别器 $D(x; \boldsymbol{\phi})$ 的参数：
   (a). 计算平均梯度：
   $$\boldsymbol{g}_{\phi} = \frac{1}{b}\sum_{j=1}^{b} \nabla_{\boldsymbol{\phi}} F\Big(x_j^{\text{real}}, x_j^{\text{fake}}; \boldsymbol{\phi}_{\text{now}}\Big).$$
   (b). 做梯度下降更新判别器参数：$\boldsymbol{\phi}_{\text{new}} \leftarrow \boldsymbol{\phi}_{\text{now}} - \eta \cdot \boldsymbol{g}_{\phi}$。

17.3.2 GAIL 的生成器和判别器

训练数据： GAIL 的训练数据是被模仿的对象（比如人类专家）操作智能体得到的轨迹，记作

$$\tau = \big[s_1, a_1, s_2, a_2, \cdots, s_m, a_m\big].$$

数据集中有 k 条轨迹，把数据集记作：

$$\mathcal{X} = \big\{\tau^{(1)}, \tau^{(2)}, \cdots, \tau^{(k)}\big\}.$$

生成器： 上一小节中的 GAN 的生成器记作 $x = G(s; \boldsymbol{\theta})$，它的输入 s 是个随机抽取的向量，输出 x 是一个数据点（比如一张图片）。本小节的 GAIL 的生成器是策略网络 $\pi(a|s; \boldsymbol{\theta})$，如图 17.9 所示。策略网络的输入是状态 s，输出是一个向量：

$$\boldsymbol{f} = \pi\big(\,\cdot\,\big|\, s; \boldsymbol{\theta}\big).$$

输出向量 $\boldsymbol{f}$ 的维度是动作空间的大小 $\mathcal{A}$，它的每个元素对应一个动作，表示执行该动作

[3] 不能让判别器比生成器进步快太多，否则训练会失败。假如判别器的准确率是 100%，那么无论生成器的输出 x 是什么，总被判别为“假”，那么生成器就不知道什么样的 x 更像真的，因而无从改进。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

的概率。给定初始状态 s_1，并让智能体与环境交互，可以得到一条轨迹：

$$\tau \quad = \quad [s_1, a_1, s_2, a_2, \cdots, s_n, a_n].$$

其中动作是根据策略网络抽样得到的：$a_t \sim \pi(\cdot|s_t;\boldsymbol{\theta})$，$\forall t = 1, \cdots, n$；下一时刻的状态是环境根据状态转移函数计算出来的：$s_{t+1} \sim p(\cdot|s_t, a_t)$，$\forall t = 1, \cdots, n$。

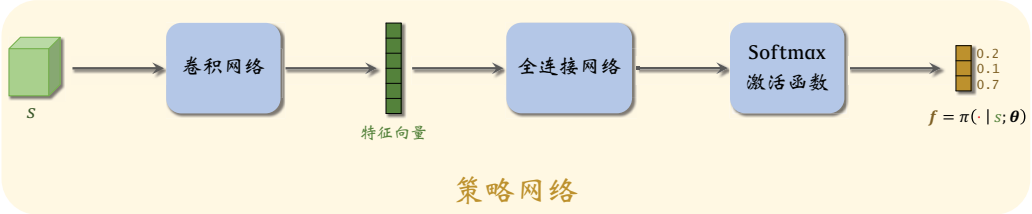


图 17.9: 策略网络 $\pi(a|s;\boldsymbol{\theta})$ 的神经网络结构。输入是状态 s，输出是动作空间 $\mathcal{A}$ 中每个动作的概率值。

判别器： GAIL 的判别器记作 $D(s, a;\boldsymbol{\phi})$，它的结构如图 17.10 所示。判别器的输入是状态 s，输出是一个向量：

$$\widehat{\boldsymbol{p}} \quad = \quad D\big(s, \cdot \,\big|\, \boldsymbol{\phi}\big).$$

输出向量 $\widehat{\boldsymbol{p}}$ 的维度是动作空间的大小 $\mathcal{A}$，它的每个元素对应一个动作 a，把一个元素记作：

$$\widehat{p}_a \quad = \quad D\big(s, a;\boldsymbol{\phi}\big) \quad \in \quad (0, 1), \qquad \forall\, a \,\in\, \mathcal{A}.$$

$\widehat{p}_a$ 接近 1 表示 (s, a) 为“真”，即动作 a 是人类专家做的。$\widehat{p}_a$ 接近 0 表示 (s, a) 为“假”，即策略网络生成的。

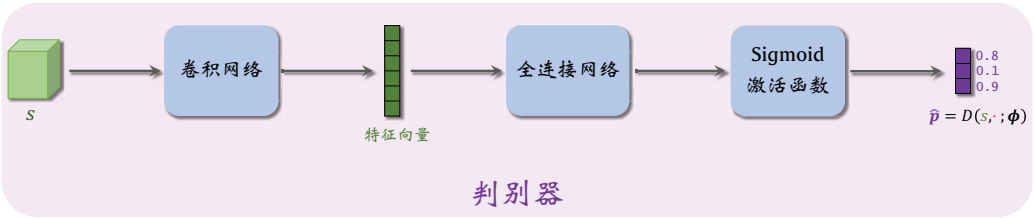


图 17.10: 判别器 $D(s, a;\boldsymbol{\phi})$ 的神经网络结构。输入是状态 s。输出向量的维度等于 $|\mathcal{A}|$，每个元素对应一个动作，每个值都是介于 0 到 1 之间的概率。

17.3.3 GAIL 的训练

训练的目的是让生成器（即策略网络）生成的轨迹与数据集中的轨迹（即被模仿对象的轨迹）一样好，当判别器无法区分生成的轨迹与数据集里的轨迹。

训练生成器： 设 $\boldsymbol{\theta}_{\text{now}}$ 是当前策略网络的参数。用策略网络 $\pi(a|s;\boldsymbol{\theta}_{\text{now}})$ 控制智能体与环境交互，得到一条轨迹：

$$\tau \quad = \quad [s_1, a_1, s_2, a_2, \cdots, s_n, a_n].$$

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang @ stevens.edu

判别器可以评价 (s_t, a_t) 有多真实；$D(s_t, a_t; \boldsymbol{\phi})$ 越大，说明 (s_t, a_t) 在判别器的眼里越真实。把

$$u_t \;=\; \ln D\big(s_t, a_t;\, \boldsymbol{\phi}\big)$$

作为第 t 步的回报；u_t 越大，则说明 (s_t, a_t) 越真实。我们有这样一条轨迹：

$$s_1, a_1, u_1, \quad s_2, a_2, u_2, \quad \cdots, \quad s_n, a_n, u_n.$$

于是可以用 TRPO 来更新策略网络。设当前策略网络的参数为 $\boldsymbol{\theta}_{\text{now}}$。定义目标函数：

$$\tilde{L}\big(\boldsymbol{\theta} \,\big|\, \boldsymbol{\theta}_{\text{now}}\big) \;\triangleq\; \frac{1}{n}\sum_{t=1}^{n} \frac{\pi(a_t|s_t;\boldsymbol{\theta})}{\pi(a_t|s_t;\boldsymbol{\theta}_{\text{now}})} \cdot u_t.$$

求解下面的带约束的最大化问题，得到新的参数：

$$\boldsymbol{\theta}_{\text{new}} \;=\; \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\; \tilde{L}\big(\boldsymbol{\theta} \,\big|\, \boldsymbol{\theta}_{\text{now}}\big); \qquad \text{s.t. dist}\,\big(\boldsymbol{\theta}_{\text{now}},\, \boldsymbol{\theta}\big) \le \Delta. \tag{17.2}$$

此处的 dist 衡量 $\boldsymbol{\theta}_{\text{now}}$ 与 $\boldsymbol{\theta}$ 的区别，Δ 是一个需要调的超参数。TRPO 的详细解释见第 9.1 节。

训练判别器： 训练判别器的目的是让它能区分真的轨迹与生成的轨迹。从训练数据集中均匀抽样一条轨迹，记作

$$\tau^{\text{real}} \;=\; \Big[s_1^{\text{real}}, a_1^{\text{real}}, \cdots, s_m^{\text{real}}, a_m^{\text{real}}\Big].$$

用策略网络控制智能体与环境交互，得到一条轨迹，记作

$$\tau^{\text{fake}} \;=\; \Big[s_1^{\text{fake}}, a_1^{\text{fake}}, \cdots, s_n^{\text{fake}}, a_n^{\text{fake}}\Big].$$

公式中的 m、n 分别是两条轨迹的长度。

训练判别器的时候，要鼓励判别器做出准确的判断。我们希望判别器知道 $\big(s_t^{\text{real}}, a_t^{\text{real}}\big)$ 是真的，所以应该鼓励 $D\big(s_t^{\text{real}}, a_t^{\text{real}};\, \boldsymbol{\phi}\big)$ 尽量大。我们希望判别器知道 $\big(s_t^{\text{fake}}, a_t^{\text{fake}}\big)$ 是假的，所以应该鼓励 $D\big(s_t^{\text{fake}}, a_t^{\text{fake}};\, \boldsymbol{\phi}\big)$ 尽量小。定义损失函数

$$F\big(\tau^{\text{real}}, \tau^{\text{fake}};\, \boldsymbol{\phi}\big) \;=\; \underbrace{\frac{1}{m}\sum_{t=1}^{m} \ln\Big[1 - D\big(s_t^{\text{real}}, a_t^{\text{real}};\, \boldsymbol{\phi}\big)\Big]}_{D\text{ 的输出越大，这一项越小}} + \underbrace{\frac{1}{n}\sum_{t=1}^{n} \ln D\big(s_t^{\text{fake}}, a_t^{\text{fake}};\, \boldsymbol{\phi}\big)}_{D\text{ 的输出越小，这一项越小}}.$$

我们希望损失函数尽量小，也就是说判别器能区分开真假轨迹。可以做梯度下降来更新参数 $\boldsymbol{\phi}$：

$$\boldsymbol{\phi} \;\leftarrow\; \boldsymbol{\phi} \;-\; \eta \cdot \nabla_{\boldsymbol{\phi}}\, F\big(\tau^{\text{real}}, \tau^{\text{fake}};\, \boldsymbol{\phi}\big). \tag{17.3}$$

这样可以让损失函数减小，让判别器更能区分开真假轨迹。

训练流程： 每一轮训练更新一个生成器，更新一次判别器。训练重复以下步骤，直到收敛。设当前生成器和判别器的参数分别为 $\boldsymbol{\theta}_{\text{now}}$ 和 $\boldsymbol{\phi}_{\text{now}}$。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

1. 从训练数据集中均匀抽样一条轨迹，记作

$$\tau^{\text{real}} \quad = \quad \left[s_1^{\text{real}}, a_1^{\text{real}}, \cdots, s_m^{\text{real}}, a_m^{\text{real}}\right],$$

2. 用策略网络 $\pi(a|s;\boldsymbol{\theta}_{\text{now}})$ 控制智能体与环境交互，得到一条轨迹，记作

$$\tau^{\text{fake}} \quad = \quad \left[s_1^{\text{fake}}, a_1^{\text{fake}}, \cdots, s_n^{\text{fake}}, a_n^{\text{fake}}\right],$$

3. 用判别器评价策略网络的决策是否真实：

$$u_t \; = \; \ln D\Big(s_t^{\text{fake}}, a_t^{\text{fake}}; \boldsymbol{\phi}_{\text{now}}\Big), \qquad \forall\, t = 1, \cdots, n.$$

4. 把 τ^{fake} 和 $u_1, \cdots, u_n$ 作为输入，用公式 (17.2) 更新策略网络参数，得到 $\boldsymbol{\theta}_{\text{new}}$。
5. 把 τ^{real} 和 τ^{fake} 作为输入，用公式 (17.3) 更新判别器参数，得到 $\boldsymbol{\phi}_{\text{new}}$。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu

第 十七 章 相关文献

行为克隆 (Behavior Cloning) 这个概念很早就出现在人工智能领域，比如 1995 年的论文 [7]、1997 年的论文 [20]。论文 [80, 99] 研究了行为克隆的理论误差，指出行为克隆会让错误累加。行为克隆也叫做 Learning from Demonstration (LfD) [5]。LfD 这个名字最早由 1997 年的论文提出 [83]。

逆向强化学习 (Inverse Reinforcement Learning) 这个问题首先由 Ng 和 Russell 2000 年的论文提出 [74]。这个问题原本是指“从最优策略中推断出奖励函数”。Abbeel 和 Ng 2004 年的论文 [1] 提出从人类专家的策略中反向学习出奖励函数，然后用奖励函数训练策略函数；这种方法被称作学徒学习 (Apprenticeship Learning)。本书第 17.2 节的内容主要基于学徒学习的思想。逆向强化学习的方法有很多种，比如 [16, 36, 59, 98, 125]。

生成判别模仿学习 (Generative Adversarial Imitation Learning) 由 Ho 和 Ermon 2016 年的论文提出 [47]。它主要基于生成判别网络 (Generative Adversarial Network，缩写 GAN)。GAN 由 Goodfellow 等人 2014 年的论文提出 [41]。

《深度强化学习》2021-02-19 尚未校对，仅供预览。
如发现错误，请告知作者 shusen.wang@stevens.edu