

# 模仿学习简洁教程

许天<sup>1</sup>, 李子牛<sup>2</sup>, 俞扬<sup>1</sup>

1: 南京大学

2: 香港中文大学 (深圳)

`xut@lamda.nju.edu.cn, ziniuli@link.cuhk.edu.cn, yuy@nju.edu.cn`

2021 年 8 月 16 日

# 目录

<b>1 模仿学习：介绍</b>	<b>1</b>
1.1 什么是模仿学习？	1
1.2 谁将从这篇教程中受益？	1
<b>2 背景介绍</b>	<b>2</b>
2.1 马尔可夫决策过程	2
2.1.1 马尔可夫链	2
2.1.2 马尔可夫决策过程	2
2.1.3 无限长度的折扣马尔可夫决策过程	3
2.1.4 有限长度回合制马尔可夫决策过程	4
2.2 模仿学习问题设定	5
第 2 章 练习	5
<b>3 行为克隆</b>	<b>7</b>
3.1 行为克隆算法	7
3.2 复合误差	8
3.3 DAgger 算法	10
3.4 讨论	10
<b>4 对抗式模仿学习</b>	<b>11</b>
4.1 状态-动作分布匹配	11
4.2 极小极大优化建模	12
4.2.1 学徒学习	13
4.2.2 生成对抗式模仿学习	14
4.3 更高级的对抗模仿学习算法	14
4.3.1 TAIL	15
4.3.2 MIMIC-MD	16
4.4 讨论	17
<b>5 环境模仿</b>	<b>20</b>
5.1 环境模仿学习	20
5.2 基于行为克隆的环境模仿	21
5.3 基于对抗式的环境模仿	22
<b>6 总结</b>	<b>23</b>

# 第1章 模仿学习：介绍

## 1.1 什么是模仿学习？

模仿学习 (Imitation Learning) [3, 4, 26, 39]——从专家示例中学习——是一种让智能体（机器人）像人类专家一样能够进行智能决策的方法。在通往通用人工智能的路上，人们发现很难手工地进行编程来教会智能体进行思考，因为这么做涉及到大量的人工工程。比如，在教会车辆自动驾驶的过程中，需要有大量的约束进行考虑（安全驾驶而不发生事故、平稳驾驶而增加舒适感）等等，而针对这些约束设计特定的监督信息信号来引导智能体是一个比较困难的任务。相反之下，人类却能比较容易地完成这些任务，并且为智能体提供大量的示例行为。利用这些专家示例来教会智能体进行智能决策就是模仿学习主要解决的问题。

通用地来讲，任何希望智能体能够像“专家”一样进行决策的系统都能从模仿学习的方法里受益 [14, 18, 74]。事实上，在 20 世纪 80 年代末，卡内基梅隆大学 (Carnegie Mellon University) 的 Pomerleau 教授已经利用人类专家示例成功地训练一个浅层神经网络，使得其控制的无人车能够穿越北美洲 [41]。前几年大放异彩的 AlphaGo，第一个打败人类顶尖选手的机器人，也用到了模仿学习方法对人类棋谱进行学习 [59]。除此之外，在现代的应用中，比如推荐系统 [58]，互联网网约车派单 [56] 等系统中，也用到了模仿学习的方法来构建一个虚拟世界，从而允许智能体在其中自由地试错和学习。

不断涌现的新的任务促使研究者们设计了各种各样的模仿学习算法。其中，普遍认为模仿学习有两大类算法：行为克隆 [8, 40] (Behavioral Cloning) 和对抗式模仿学习 (Adversarial Imitation Learning) [1, 65, 76]。其中，行为克隆算法尝试最小化智能体策略和专家策略的动作差异，把模仿学习任务归约到常见的回归或者分类任务 [48, 66]。而对抗式模仿学习算法则是通过逆强化学习 [36, 50] (Inverse Reinforcement Learning) 来构建一个对抗的奖赏函数，然后最大化这个奖赏函数去模仿专家行为。基于这两种基本的算法设计思路，研究者们提出了各种各样的模仿学习算法变体 [9, 16, 24, 25, 29–31, 43, 67]。

## 1.2 谁将从这篇教程中受益？

虽然已经有一些很好的综述 [4, 26, 39] 和书籍 [3] 介绍了模仿学习算法以及其应用。但是直到最近，模仿学习算法的底层理论才被逐步建立和完善 [44–46, 71, 72]。这些理论很好地解释了模仿学习算法在实际应用中发生的现象，对以后的算法设计、实验分析都很有帮助。但是由于这些理论的前沿性，现在还没有文章来详细地对此进行讲解，这促使作者们写下了这篇教程。

这篇教程着重从统计学习理论的角度来介绍模仿学习算法。在第2章对马尔可夫决策过程、模仿学习、强化学习等背景知识进行了介绍，紧接着在第3章讲解了行为克隆算法及其变体，随后在第4章介绍了对抗式模仿学习算法，最后在第5章讨论了模仿学习算法在环境重构和基于模型强化学习里的应用，一些未被覆盖到的内容和结论性陈述在第6章给出。

作者们希望通过从第2到5章的介绍，能使读者对模仿学习算法有深入浅出的理解。因此，无论是对模仿学习理论感兴趣、希望借助理论来分析已有的实验结果、通过理论来设计更好的算法的人，都会从这个教程里多多少少地受益。我们希望读者有一定机器学习或强化学习或模仿学习基础，但这个不是必须的，因为大多数涉及数学概念的部分都会有具体的例子来帮助更好地理解。

## 第2章 背景介绍

### 内容提要

- 马尔可夫决策过程基本元素
- 值函数

- 状态-动作分布
- 对偶表示

## 2.1 马尔可夫决策过程

### 2.1.1 马尔可夫链

考察一个有限状态的马尔可夫链 (Markov Chain), 其状态空间  $\mathcal{S}$  用整数集合  $\{1, 2, \dots, |\mathcal{S}|\}$  来表示, 这里  $|\mathcal{S}|$  表示总状态个数。这里“马尔可夫性”是指其状态转移与历史无关。具体而言, 我们有

$$P(s_{t+1}|s_t, s_{t-1}, \dots, s_0) = P(s_{t+1}|s_t), \quad \forall s_0, s_1, \dots, s_t, s_{t+1} \in \mathcal{S}. \quad (2.1)$$

因此, 对于一个有限状态的马尔可夫链, 其状态转移可以用一个矩阵  $P \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$  来描述:

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,|\mathcal{S}|} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,|\mathcal{S}|} \\ \cdots & \cdots & \cdots & \cdots \\ p_{|\mathcal{S}|,1} & p_{|\mathcal{S}|,2} & \cdots & p_{|\mathcal{S}|,|\mathcal{S}|} \end{bmatrix}$$

这里一个元素  $P_{i,j}$  表示由状态  $i$  转移到状态  $j$  的概率。为了满足状态转移分布的性质, 我们要求:

$$\sum_{i=1}^{|\mathcal{S}|} P_{i,j} = 1, \quad \forall j \in \mathcal{S}, \quad \text{和} \quad \sum_{j=1}^{|\mathcal{S}|} P_{i,j} = 1, \quad \forall i \in \mathcal{S}.$$

为了完整地表示状态转移过程, 我们需要指定一个初始状态分布  $\rho$ 。这样我们便可以以递归地方式计算某个状态  $i$  在时间  $t$  出现的概率:

$$\mathbb{P}(s_t = i) = \sum_{j \in \mathcal{S}} \mathbb{P}(s_t = j) P(s_t = j | s_{t-1} = j) = \sum_{j \in \mathcal{S}} \mathbb{P}(s_{t-1} = j) p_{j,i}.$$

**例题 2.1 马尔可夫链计算** 考虑如下的马尔可夫链:

$$P = \begin{bmatrix} 0.2 & 0.2 & 0.6 \\ 0.1 & 0.6 & 0.3 \\ 1.0 & 0.0 & 0.0 \end{bmatrix}$$

假设初始状态分布是均匀分布,  $\rho = (1/3, 1/3, 1/3)$ 。那么一个长度为 5 的轨迹  $\text{tr} = (1 \rightarrow 2 \rightarrow 2 \rightarrow 3 \rightarrow 1)$  的发生概率为:

$$\mathbb{P}(\text{tr}) = \frac{1}{3} \times 0.2 \times 0.6 \times 0.3 \times 1.0 = 0.012$$

### 2.1.2 马尔可夫决策过程

马尔可夫决策过程 [42, 63] (Markov Decision Process, MDP) 是马尔可夫链的拓展, 这里我们需要额外考虑两个元素: 动作空间  $\mathcal{A}$  和奖励函数  $r$ 。具体地, 一个马尔可夫决策过程可以由 5 元组  $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \rho, P, r)$  来表示。首先, 动态空间  $\mathcal{A}$  的引入是为了丰富状态转移矩阵的刻画; 在马尔可夫决策过程里, 状态转移不仅受到前一个时刻的状态影响, 还要受到当前动作的影响。数学上, 我们可以表示为:

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \dots, s_0) = P(s_{t+1}|s_t, a_t). \quad (2.2)$$

因此，与马尔可夫链相比，马尔可夫决策过程多了动作对来决定每个时刻的转移概率。除此之外，我们用奖励函数  $r: \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$  来衡量某个状态-动作对的好坏。

为了更加方便地描述上面动作产生的过程，我们引入策略 (policy)  $\pi: \mathcal{S} \mapsto \Delta(\mathcal{A})$ 。具体地，条件分布  $\pi(a|s)$  表示在状态  $s$  处选择动作  $a$  的概率。因为我们最多有  $|\mathcal{S}|$  个状态和  $|\mathcal{A}|$  个动作， $\pi$  也可以用一个  $|\mathcal{S}| \times |\mathcal{A}|$  的矩阵来表示。

根据马尔可夫决策过程的交互规则以及累计回报计算的不同，我们考虑两类特殊的马尔可夫决策过程：无限长度的折扣马尔可夫决策过程和有限长度回合制马尔可夫决策过程。下面进行分别介绍。

### 2.1.3 无限长度的折扣马尔可夫决策过程

一个无限长度的折扣马尔可夫决策过程可以用一个 6 元组来表示  $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \rho, P, r, \gamma)$ ，这里  $\gamma \in (0, 1)$  是折扣因子。首先，我们定义在  $\mathcal{M}$  下的累计（期望）回报：

$$V(\pi) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \sim \rho(\cdot), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t) \right].$$

可以看到  $V(\pi)$  衡量了策略  $\pi$  能够获得的累计奖励的期望。注意到，每一步的奖励会被乘以系数  $\gamma^t$  来计算；这么做，从数学上来讲是为了保证上面的无限求和定义是正确的。更清楚地，假设奖励函数是有界的；不失一般性，假设对于任意的状态-动作对  $(s, a)$ ，我们都有  $r(s, a) \in (0, 1)$ ，那么我们可以发现

$$V(\pi) \leq \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t \right] = \frac{1}{1-\gamma}.$$

除非特殊说明，奖励函数  $r(s, a) \in (0, 1)$  被视作一个默认的假设。此外，定义有效决策长度为  $1/(1-\gamma)$ ，这是因为当把累计回报里的“无限长度”截断到  $\tilde{O}(1/(1-\gamma))$  的量级时，可以很好地用有限长度来代替无限长度：

$$\left| \mathbb{E} \left[ \sum_{t=0}^H \gamma^t r(s_t, a_t) \right] - \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \right| \leq \varepsilon \implies H \geq \frac{1}{1-\gamma} \log \left( \frac{\varepsilon}{1-\gamma} \right).$$

给定  $\mathcal{M}$ ，我们可以求解最优策略  $\pi^*$  使得其累计回报最大：

$$\pi^* \in \operatorname{argmax}_{\pi} V(\pi).$$

强化学习 [63] (Reinforcement Learning) 要解决的问题是在不知道转移概率  $P$  的精确形式但可以与环境交互来获取转移概率信息的情况下，求解最优策略。

由于马尔可夫决策的再生性质，我们可以对任意的起始状态  $s$  来定义其状态价值函数 (state value function)：

$$V^{\pi}(s) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t) \right].$$

类似地，我们可以定义状态-动作价值函数 (state-action value function)：

$$Q^{\pi}(s, a) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t) \right].$$

**例题 2.2 值函数计算** 考虑下面一个简单的马尔可夫决策过程：这里有三个状态  $\mathcal{S} = \{s_0, s_1, s_2\}$  和两个动作  $\mathcal{A} = \{a_1, a_2\}$ （注意这里下标不是表示时间）。

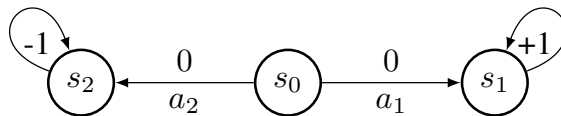


图 2.1: 一个简单的马尔可夫决策过程例子。

考虑初始状态分布  $\rho = (1, 0, 0)$ ，也就是说初始状态总在  $s_0$ 。在状态  $s_0$  有两个动作：向右的动作  $a_1$  和向左

的动作  $a_2$ ；执行  $a_1$  ( $a_2$ ) 后将分别转向  $s_1$  ( $s_2$ )； $s_1$  和  $s_2$  是**吸收态**，意味着采取任何动作都将停留在自己。其状态转移和奖励函数用数学语言表示如下：

$$\begin{aligned} P(s' = s_1 | s = s_0, a = a_1) &= 1, & P(s' = s_2 | s = s_0, a = a_2) &= 1, \\ P(s' = s_1 | s = s_1, a = \cdot) &= 1, & P(s' = s_2 | s = s_2, a = \cdot) &= 1, \\ r(s = s_2, a = \cdot) &= -1, & r(s = s_1, a = \cdot) &= +1, & r(s = s_0, a = \cdot) &= 0 \end{aligned}$$

考虑策略  $\pi(a_1|s_0) = 0.6, \pi(a_2|s_0) = 0.4$ 。根据等比级数求和，下面计算其值函数，

$$\begin{aligned} V^\pi(s_0) &= 0.6 \times \frac{\gamma}{1-\gamma} + 0.4 \times \frac{-\gamma}{1-\gamma} = \frac{0.2\gamma}{1-\gamma}, \\ V^\pi(s_1) &= \frac{1}{1-\gamma}, & V^\pi(s_2) &= \frac{-1}{1-\gamma}, \\ Q^\pi(s_0, a_1) &= \frac{\gamma}{1-\gamma}, & Q^\pi(s_0, a_2) &= \frac{-\gamma}{1-\gamma}. \end{aligned}$$

除此之外，我们还要引入一些与奖励没有关系、只与策略和转移函数有关的变量，这些变量将方便分析后续的模仿学习算法。首先，引入**折扣的状态访问分布**：

$$d^\pi(s) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s).$$


当没有歧义的时候，我们将省去前缀“折扣”，简称  $d^\pi(s)$  为**状态访问分布**。直观来讲， $d^\pi(s)$  刻画了访问状态  $s$  的累计频率。类似地，我们引入（折扣的）**状态-动作访问分布**：

$$d^\pi(s, a) := (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s, a_t = a) = d^\pi(s) \pi(a|s).$$

**注** 状态访问分布定义里的系数  $(1-\gamma)$  是为使其成为一个合理的概率分布。

**例题 2.3 访问分布的计算** 考虑例题2.2中的马尔可夫决策过程和策略。我们可以计算，

$$\begin{aligned} d^\pi(s_0) &= 1-\gamma, & d^\pi(s_1) &= 0.6(1-\gamma) \times \frac{\gamma}{1-\gamma} = 0.6\gamma, & d^\pi(s_2) &= 0.4(1-\gamma) \times \frac{\gamma}{1-\gamma} = 0.4\gamma, \\ d^\pi(s_0, a_1) &= d^\pi(s_0) \times \pi(a_1|s_0) = 0.6(1-\gamma), & d^\pi(s_0, a_2) &= d^\pi(s_0) \times \pi(a_2|s_0) = 0.4(1-\gamma), \\ d^\pi(s_1, \cdot) &= d^\pi(s_1) \times 1 = 0.6\gamma, & d^\pi(s_2, \cdot) &= d^\pi(s_2) \times 1 = 0.4\gamma. \end{aligned}$$

 **笔记** 状态-动作访问分布和值函数的联系可以通过对偶分析得出 [42]：

$$V(\pi) = \frac{1}{1-\gamma} \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} d^\pi(s, a) r(s, a). \quad (2.3)$$

**注** 公式 (2.3) 对于分析模仿学习算法非常重要。由于奖励函数是有界的，因此如果一个策略能够很好地恢复状态-动作访问分布，那么便可以很好地恢复值函数；而这个恢复过程是与环境奖赏没有关系的。

**注** 根据公式 (2.3) 和例题2.2和2.3中的结果，我们可以很容易验证：

$$V(\pi) = \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} d^\pi(s, a) r(s, a) = \frac{1}{1-\gamma} \times (1 \times 0.6\gamma + (-1) \times 0.4\gamma) = \frac{0.2\gamma}{1-\gamma}.$$

### 2.1.4 有限长度回合制马尔可夫决策过程

一个有限长度回合制马尔可夫决策过程可以由一个 6 元祖来表示  $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, H, \rho)$ 。其中  $\mathcal{S}$  和  $\mathcal{A}$  分别是状态和动作空间。不同之前的定义，这里  $\mathcal{P} = \{P_1, \dots, P_H\}$  指定了时变转移函数；具体地， $P_h(s_{h+1}|s_h, a_h)$  表示了时间步  $h$  和状态  $s_h$  上，执行动作  $a_h$ ，转移到状态  $s_{h+1}$  的概率。类似地， $r = \{r_1, \dots, r_H\}$  指定了马尔可夫决策过程的奖赏函数，不失一般性，我们假设  $r_h : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1], \forall h \in [H]$ ，这里符号  $[x]$  表达从 1 到  $x$  的整数集合。为了适应这种时变的概率转移， $\pi = \{\pi_1, \dots, \pi_H\}$  表示了时变的策略，其中  $\pi_h : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ ， $\Delta(\mathcal{A})$  表示在动作空间上的概率单纯形， $\pi_h(a|s)$  表示在时间步  $h$  和状态  $s$  上，执行动作  $a$  的概率。

在  $\mathcal{M}$  下, 策略  $\pi$  的**累积 (期望) 回报**定义如下:

$$V(\pi) := \mathbb{E} \left[ \sum_{h=1}^H r_h(s_h, a_h) \mid s_1 \sim \rho; a_h \sim \pi_h(\cdot | s_h), s_{h+1} \sim P_h(\cdot | s_h, a_h), \forall h \in [H] \right].$$

在有限长度回合制马尔可夫决策过程中, 给定策略  $\pi$ , 我们也可以定义**状态-动作访问分布**:

$$P_h^\pi(s, a) := \mathbb{P}(s_h = s, a_h = a \mid s_1 \sim \rho; a_\ell \sim \pi_h(\cdot | s_\ell), \forall \ell \in [h]).$$


同样地, 在  $\mathcal{M}$  下, 给定策略  $\pi$ , 我们也可以建立累计回报与状态访问分布之间的关系:

$$V(\pi) = \sum_{h=1}^H \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} P_h^\pi(s, a) r_h(s, a). \quad (2.4)$$

## 2.2 模仿学习问题设定

在模仿学习里, 我们假设有一个专家策略是表现不错的, 我们希望智能体能够模仿专家策略进行决策。具体而言, 我们希望智能体的累计回报与专家策略的累计回报比较接近。因此我们可以把模仿学习问题建模成下面的优化问题:

$$\min_{\pi} V(\pi^E) - V(\pi). \quad (2.5)$$

 **笔记** 公式 (2.5) 所展示的优化问题在一定程度上可以视为最大化  $V(\pi)$ , 这个目标和强化学习的目标是一致的。但是与强化学习的区别有: 1) 在模仿学习中, 最大化  $V(\pi)$  的过程里是不知道奖赏函数的; 2) 反而, 智能体被提供专家示例来进行策略优化。

我们假设有一个 (未知的) 专家策略可以为我们提供一些示例, 我们的目的是从这些示例中来恢复专家策略。具体而言, 用  $\pi^E$  来表示专家策略; 专家策略可以和环境进行交互来产生一系列的状态-动作对, 这些状态-动作对就是我们说的示例。通常而言, 这些状态-动作对是一条完整的轨迹被组织起来。经常地, 我们称这个专家示例为 (训练) 数据集  $\mathcal{D}$ 。如果用  $\text{tr}$  来表示一个完整的**轨迹**; 也就是说:  $\text{tr} = \{s_1, a_1, s_2, a_2, \dots, s_H, a_H\}$ , 那么一个由  $m$  条轨迹构成的专家示例  $\mathcal{D}$  可以记为  $\mathcal{D} = \{\text{tr}_i\}_{i=1}^m$ 。

**注** 为了简便, 我们在之后的章节中讲解模仿学习算法的时候一般都是考虑稳态策略  $\pi$ ; 在给出理论分析结果的时候, 我们也会给出有限长度回合制马尔可夫决策过程下时变策略对应的结果。

## 第2章 练习

1. **一般化计算值函数/访问分布的方式 (折扣马尔可夫决策过程)** 对于一个折扣马尔可夫决策过程, 给定状态概率转移等信息的时候, 我们想找到一个通用的方式来计算  $V^\pi(s)$ ,  $Q^\pi(s, a)$ , 以及  $d^\pi(s)$ 。这时, 我们需要用到贝尔曼方程 (Bellman Equation):

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot | s)} \left[ r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^\pi(s') \right] \quad (2.6)$$

定义  $r^\pi : \mathcal{S} \mapsto \mathbb{R}$  为  $r^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot | s)} [r(s, a)]$ ,  $P^\pi : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}$  为  $P^\pi(s' | s) = \sum_a \pi(a | s) P(s' | s, a)$ 。因此我们可以将公式 (2.6) 改写为:

$$V^\pi(s) = (r^\pi + \gamma P^\pi V^\pi)(s) \implies V^\pi = (I - \gamma P^\pi)^{-1} r^\pi. \quad (2.7)$$

因此在已知转移函数和奖励函数的情况下, 我们可以根据公式 (2.7) 来显式地计算  $V^\pi$ 。不难验证 (通过矩阵的级数求和), 公式 (2.7) 里的  $(I - \gamma P^\pi)^{-1}$  就是 (未考虑初始状态分布且未被归一化的) 的状态访问分布  $\tilde{d}^\pi$ :

$$\tilde{d}^\pi = (I - \gamma P^\pi)^{-1}.$$

为了更好的理解这一点, 注意到在对公式 (2.7) 两边同时考虑初始状态分布:

$$V(\pi) = \sum_{s \in \mathcal{S}} \rho(s) V^\pi(s) = \rho^T (I - \gamma P^\pi)^{-1} r^\pi = \frac{1}{1 - \gamma} \langle d^\pi, r^\pi \rangle, \quad (2.8)$$



将上式与公式 (2.3) 进行对比, 不难发现公式 (2.8) 也是  $V(\pi)$  的对偶形式; 而且,

$$d^\pi = (1 - \gamma)(I - \gamma P^{\pi^T})^{-1} \rho. \quad (2.9)$$

在利用公式 (2.9) 计算出  $d^\pi(s)$  后, 很容易计算  $d^\pi(s, a) = d^\pi(s)\pi(a|s)$ 。

2. 一般化计算值函数/访问分布的方式 (有限长度马尔可夫决策过程) 对于一个有限长度的马尔可夫决策过程, 计算  $V_h^\pi(s), Q_h^\pi(s, a)$ ; 其准则就是贝尔曼递归方程:

$$\begin{aligned} V_{H+1}^\pi(s) &= 0, \\ Q_h^\pi(s, a) &= r_h(s, a) + \sum_{s' \in \mathcal{S}} P_h(s'|s, a) V_{h+1}^\pi(s'), \quad \forall h = 1, \dots, H \\ V_h^\pi(s) &= \sum_{a \in \mathcal{A}} \pi_h(a|s) Q_h^\pi(s, a), \quad \forall h = 1, \dots, H. \end{aligned}$$

对于  $P^\pi(s)$  和  $P^\pi(s, a)$  的计算则依赖前向方程:

$$\begin{aligned} P_1^\pi(s) &= \rho(s), \\ P_h^\pi(s, a) &= P_h^\pi(s) \pi_h(a|s), \quad \forall h = 1, 2, \dots, H \\ P_{h+1}^\pi(s') &= \sum_{(s, a) \in \mathcal{S} \times \mathcal{A}} P_h^\pi(s, a) P_h(s'|s, a), \quad \forall h = 1, \dots, H - 1. \end{aligned}$$



## 第3章 行为克隆

### 内容提要

❑ 行为克隆算法 (BC)

❑ 数据增广算法 (DAgger)

❑ 复合误差

这一章主要介绍行为克隆 [8, 40] (Behavioral Cloning) 算法。很难说是某个人发明了行为克隆算法, 但这个算法至少可以追溯到 [40, 41, 51]。其中, Pomerleau 成功训练了一个浅层神经网络来完成自动驾驶里的道路跟随任务。

### 3.1 行为克隆算法

考虑我们已有的数据集  $\mathcal{D}$ , 如何恢复专家策略  $\pi^E$  呢? 行为克隆的想法很简单, 我们直接从数据中估计  $\pi^E$ 。一个经典的估计方法就是最大似然估计 (Maximum Likelihood Estimation)。我们之前提到过一个稳态策略  $\pi$  可以用一个  $|\mathcal{S}| \times |\mathcal{A}|$  的表格 (矩阵) 进行表示; 具体到一个状态  $s$ ,  $\pi(\cdot|s)$  就是动作空间  $\mathcal{A}$  上的单纯型 (可以理解为动作空间  $\mathcal{A}$  上的一个概率分布)。

对于任意给定的一个需要估计的概率模型  $\hat{\pi}_\theta$  ( $\theta$  是被估计的参数), 一个数据样本  $(s, a)$  的似然可以表示为  $\hat{\pi}_\theta(a|s)$ 。因此, 最大化对数似然模型可以表述为:

$$\begin{aligned} \max_{\hat{\pi}_\theta \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}} \quad & \sum_{(s,a) \in \mathcal{D}} \log \pi(a|s) \\ \text{s.t.} \quad & \sum_{a \in \mathcal{A}} \hat{\pi}_\theta(a|s) = 1, \quad \forall s \in \mathcal{S}. \end{aligned} \quad (3.1)$$

可以验证公式 (3.1) 对应的问题是一个凸优化问题, 并且其最优解的形式对于有限状态和动作空间的马尔可夫决策过程问题很简单。具体而言, 最优解可以用“计数”来求解: 对于数据集出现里的状态  $s$ , 我们令

$$\hat{\pi}_\theta(a|s) = \frac{\sum_{(s_i, a_i) \in \mathcal{D}} \mathbb{I}(s_i = s, a_i = a)}{\sum_{(s_i, a_i) \in \mathcal{D}} \mathbb{I}(s_i = s)}, \quad (3.2)$$

这里  $\mathbb{I}(\cdot)$  表示示例函数, 即如果  $\cdot$  为真, 那么  $\mathbb{I}(\cdot) = 1$ ; 否则  $\mathbb{I}(\cdot) = 0$ 。对于数据集里没有出现过的状态  $s$ , 我们可以令其动作分布为一个均匀分布, 也就说  $\pi(a|s) = 1/|\mathcal{A}|$ 。

**注** 这种参数化方法一般称作为直接参数化 (Direct Parameterization), 即每个动作选择概率  $\hat{\pi}_\theta(a|s)$  就是一个参数  $\theta_{s,a}$ , 这里  $\theta \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$ 。

 **笔记** 行为克隆算法可以看做为最小化策略动作差异:

$$\min_{\pi \in \Pi} D_{\text{KL}}(\pi^E \| \pi), \quad (3.3)$$

这里  $D_{\text{KL}}(\pi^E \| \pi) = \mathbb{E}_{s \sim d^{\pi^E}} [D_{\text{KL}}(\pi^E(\cdot|s) \| \pi(\cdot|s))]$ 。由于  $\pi^E$  和  $d^{\pi^E}$  都是未知的, 我们便先利用专家数据估计  $\pi^E$  和  $d^{\pi^E}$ , 然后求解对应的优化问题:

$$\min_{\pi \in \Pi} D_{\text{KL}}(\hat{\pi}^E \| \pi),$$

这里  $\hat{\pi}^E$  可以由最大似然估计得到 (对应公式 (3.2))。当  $\Pi$  包含所有的随机策略的时候, 对应的最优解就是  $\hat{\pi}^E$  本身。

对于非离散空间问题 (状态空间为欧几里得空间, 或动作空间为欧几里得空间, 或者两者都为欧几里得空间), 上面的基于“计数”的求解方案不再可行。特别地, 这时候  $\theta_{s,a} = \hat{\pi}_\theta(a|s)$  不再成立。这时, 公式 (3.1) 一般转化为:

$$\max_{\theta} \sum_{(s,a) \in \mathcal{D}} \log(\hat{\pi}_\theta(a|s)). \quad (3.4)$$

注意到公式 (3.4) 里没有显式地约束策略为单纯型，这是因为通过一些参数化技巧，我们可以很容易使得策略是有效的；见下面的例子。

**例题 3.1** (softmax 参数化) 对于一个  $|\mathcal{A}|$  维的向量  $z$  定义 softmax 函数：

$$\text{softmax}(z)[i] = \frac{\exp(z[i])}{\sum_i \exp(z[i])}.$$

这里  $z[i]$  表示第  $i$  个分量。如果输入的状态是连续空间，我们可以通过特征函数  $g: \mathcal{S} \mapsto \mathbb{R}^d$  将状态  $s$  映射到一个  $d$  维空间的表示 (representation)  $h$ ，然后再通过参数  $\theta \in \mathbb{R}^{d \times |\mathcal{A}|}$  将  $d$  维空间的表示  $h$  映射为动作分布：

$$h = g(s) \in \mathbb{R}^d, z = \theta h \in \mathbb{R}^{|\mathcal{A}|}, \hat{\pi}_\theta(a|s) = \text{softmax}(z)[a].$$

不难发现，这时公式 (3.4) 对应的问题就是常见的分类问题里的交叉熵优化问题。

**例题 3.2** (Gaussian 参数化) 当动作空间是连续动作空间时，我们可以用高斯分布去表示一个策略。具体地，对于每个状态  $s$ ，我们将策略  $\hat{\pi}_\theta(\cdot|s)$  表示为高斯分布  $\mathcal{N}(\mu_\theta(s), \sigma_\theta^2(s))$ ，其中  $\mu_\theta(s)$  和  $\sigma_\theta^2(s)$  分别是高斯分布的均值和方差。这时，公式 (3.4) 转化为

$$\min_{\theta} \sum_{(s,a) \in \mathcal{D}} \frac{(a - \mu_\theta(s))^2}{2\sigma_\theta^2(s)} + \frac{1}{2} \log(2\pi\sigma_\theta^2(s)). \quad (3.5)$$

实验中，我们一般将高斯分布的方差设置为与参数  $\theta$  无关的常量，此时，公式 (3.5) 转化为

$$\min_{\theta} \sum_{(s,a) \in \mathcal{D}} (a - \mu_\theta(s))^2.$$

这便是基于均方差 (Mean Square Error, MSE) 的回归问题。

在之后的陈述中，当不引起混淆的时候，为了简便，我们将  $\hat{\pi}_\theta$  简写为  $\hat{\pi}$ 。

## 3.2 复合误差

行为克隆算法很简单，那么它有什么有缺点呢？答案是有的，行为克隆在实际应用中存在复合误差 (Compounding Error) 的现象。这个观察在 [41] 已经被隐式地点出：“when driving for itself, the network may occasionally stray from the center of road and so must be prepared to recover by steering the vehicle back to the center of the road”。随后，Stéphane Ross 等人正式数学化地阐述了这个问题 [48, 49]。

具体而言，公式 (3.1) 和公式 (3.4) 对应问题的训练数据集都是经策略  $\pi^E$  采集的；但是，我们评估学习到的策略  $\hat{\pi}$  却是基于策略  $\hat{\pi}$  采集到的轨迹。数学上，我们可以回顾值函数的“对偶表示”：

$$V(\pi) = \frac{1}{1-\gamma} \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} d^\pi(s,a) r(s,a).$$

可以看到评估测试时用到策略  $\hat{\pi}$  的状态-动作访问分布和训练数据集的状态-动作分布不太一致。为了分析这个差异，我们注意到两个策略的值函数的差异可以被两个策略的状态-动作访问分布间的差异所界 (bound) 住：

$$|V(\pi^E) - V(\hat{\pi})| \leq \frac{1}{1-\gamma} \cdot \|d^{\pi^E} - d^{\hat{\pi}}\|_1. \quad (3.6)$$

公式 (3.6) 利用了奖励函数处于 0 到 1 之间的假设。那自然而然地一个问题是：如何把  $\|d^{\pi^E} - d^{\hat{\pi}}\|_1$  与行为克隆的目标函数联系起来？

### 引理 3.1 (行为克隆的复合误差 [49, 71])

考虑行为克隆算法，如果以  $\ell_1$ -norm 度量策略恢复的误差。

- 对于无限长度折扣马尔可夫决策过程，如果  $\mathbb{E}_{s \sim d^{\pi^E}} [\|\pi^E(\cdot|s) - \hat{\pi}(\cdot|s)\|_1] \leq \varepsilon$ ，那么我们有  $\|d^{\pi^E} - d^{\hat{\pi}}\|_1 \leq 1/(1-\gamma)\varepsilon$ 。
- 对于有限长度回合制马尔可夫决策过程，如果  $\frac{1}{H} \sum_{h=1}^H \mathbb{E}_{s \sim d_h^{\pi^E}} [\|\pi_h^E(\cdot|s) - \hat{\pi}_h(\cdot|s)\|_1] \leq \varepsilon$ ，那么我们有  $\frac{1}{H} \sum_{h=1}^H \|d_h^{\pi^E} - d_h^{\hat{\pi}}\|_1 \leq H\varepsilon$ 。



**笔记** 虽然引理3.1里是以  $\ell_1$ -norm 在分析策略恢复的误差  $\varepsilon$ ，但是这个指标和行为克隆算法里使用到的 KL 散度是紧密联系的，最终的结果只差一个常数项；具体的结论可以参考 [71]。

**注** 结合公式 (3.6)，我们可以看到，如果行为克隆算法的误差是  $\varepsilon$  的话，其值函数差异的上界将是  $1/(1-\gamma)^2\varepsilon$  或  $H^2\varepsilon$ 。

前面分析了行为克隆的误差界，下面进一步分析其样本复杂度。样本复杂度反应了在有限样本的情况下，行为克隆的算法的效果，比之前基于期望的分析更接近实际情况。

### 定理 3.1 (行为克隆的样本复杂度 [45, 72])

考虑行为克隆算法，假设专家策略是确定性策略，用  $\hat{\pi}$  来表示行为克隆算法得到的策略，

- 对于无限长度折扣马尔可夫决策过程，对于任意的  $\delta \in (0, 1)$  和  $\varepsilon \in (0, 1/(1-\gamma))$ ，以高于  $1-\delta$  的概率，当专家样本数量  $m \geq 2 \frac{|S| \log(|A|/\delta)}{(1-\gamma)^2\varepsilon}$  时，那么我们有  $V(\pi^E) - V(\hat{\pi}) \leq \varepsilon$ 。
- 对于有限长度回合制马尔可夫决策过程，对于任意的  $\delta \in (0, 1)$  和  $\varepsilon \in (0, H)$ ，以高于  $1-\delta$  的概率，当专家样本轨迹  $m \geq \mathcal{O}(\frac{H^2|S|}{\varepsilon} + \frac{\sqrt{|S|H^2 \log(H/\delta)}}{\varepsilon})$  时，那么我们有  $V(\pi^E) - V(\hat{\pi}) \leq \varepsilon$ 。

**笔记** 当忽略常数项和对数项时，在无限长度折扣马尔可夫决策过程和有限长度回合制马尔可夫决策过程下，行为克隆算法的样本复杂度上界分别为  $\tilde{\mathcal{O}}(|S|(1-\gamma)^{-2}/\varepsilon)$  和  $\tilde{\mathcal{O}}(|S|H^2/\varepsilon)$ 。

**笔记** 在环境转移概率未知的设定下，所有模仿学习算法的专家样本复杂度的下界分别为  $\tilde{\Omega}(|S|(1-\gamma)^{-2}/\varepsilon)$  [72] 和  $\tilde{\Omega}(H^2|S|/\varepsilon)$  [45]。样本复杂度下界表示的是，对于任意模仿学习算法，为了获得  $\varepsilon$ -最优的策略，至少需要的专家样本量。可以看到，行为克隆算法的专家样本复杂度上界与下界相吻合，表示行为克隆算法在环境转移概率未知的设定下，是极小极大最优 (Minimax Optimal) 的算法。

**笔记** 行为克隆算法的样本复杂度关于  $1/\varepsilon$  的阶数为 1，这个现象以及其理论分析都是非平凡的。这一重要理论成果对之后要介绍的对抗式模仿学习算法也有重要影响。可以说，这个成果的重要性不亚于复合误差的首次发现。具体而言，Nived Rajaraman 等人 [45] 发现了行为克隆里称之为“缺失质量” [34] (Missing Mass) 的现象，并利用这个性质来改进了对行为克隆算法的样本复杂度分析。

**注** 在定理3.1专家样本复杂度的表述在不同马尔可夫决策过程下表述略有不同。特别地，在无限长度折扣马尔可夫决策过程里，样本复杂度指向转移元组 (transition tuple) 的多少；在有限长度回合制马尔可夫决策过程里则是指完整的轨迹个数。这个差异是由于前者的转移概率函数和对应的策略是时不变的，而后的转移概率函数和对应的策略是时变的。

**例题 3.3** 考虑图3.1里的马尔可夫决策过程。在  $|S|$  个状态里，前  $|S|-1$  的状态都是好的状态，最后一个状态“b”是一个坏的状态。这是因为在前  $|S|-1$  个状态里，采取专家动作（绿颜色）的话，能得到 +1 的奖励，并且状态转移服从初始状态分布；但是，如果采取非专家动作（蓝颜色），那么将确定性地转移到坏状态 b，并且 b 是一个吸收态，意味着在 b 采取任何动作都只能停留在 b 自己，并且得到的奖励为 0。在这个例子上，我们假设初始状态只会出现在前  $|S|-1$  个状态中。

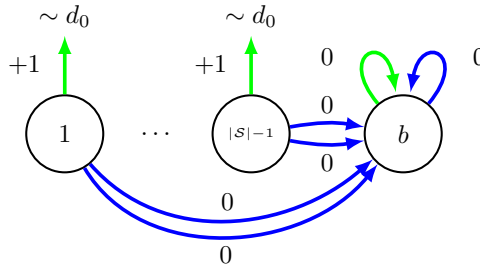


图 3.1: Reset Cliff 马尔可夫决策过程 [45, 72]。

下面以有限长度的马尔可夫决策过程来讲述复合误差（虽然是有限长度的马尔可夫决策过程，但是转移函数是时不变的）。由于专家的数据集很难覆盖所有的状态，因此在某些状态上，是没有专家示例的。根据前面的

介绍，如果在这些未被访问的状态上，BC 会采取一个均匀分布的策略，也就是说，智能体将会以  $1 - 1/|A|$  的概率转移到坏状态  $b$ 。如果这样糟糕的事情发生在第一个时间步，那么智能体在这个回合里得到的奖励便是 0；对比之下，专家得到的奖励则是  $H$ 。这样的现象就是引理3.1和定理3.1里讲述的复合误差。

**注** 图3.1里的马尔可夫决策过程和常用的测试环境 Gym MuJoCo 任务比较相像。相似之处在于 Gym MuJoCo 里的机器人控制任务，如果机器人不小心摔倒了，相当于进入一个坏的吸收态。因此行为克隆的复合误差分析对实际观测到的实验现象有很好的解释。

### 3.3 DAgger 算法

为了解决复合误差问题，Stéphane Ross 等人又提出了 DAgger (Dataset Aggregation) 算法 [49]。这是一个基于在线学习 [54] (Online Learning) 的算法，其把行为克隆得到的策略与环境不断的交互，来产生新的数据。在这些新产生的数据上，DAgger 会向专家策略申请示例；然后在增广后的数据集上，DAgger 会重新使用行为克隆进行训练，然后再与环境交互...；这个过程会不断重复进行。由于数据增广和环境交互，DAgger 算法会大大减小未访问的状态的个数，从而减小复合误差。

---

#### Algorithm 1 DAgger 算法

---

- 1: 初始化数据集  $\mathcal{D}$  和策略  $\hat{\pi}_1$ 。
  - 2: **for** episode  $i = 1, 2, \dots, T$  **do**
  - 3:   设置混合策略  $\pi_i = \beta_i \pi^E + (1 - \beta_i) \hat{\pi}_i$ 。
  - 4:   让  $\pi_i$  与环境交互得到一条新的轨迹  $\text{tr}$ 。
  - 5:   向专家咨询示例动作得到新的数据集  $\mathcal{D}_i = \{(s, \pi^E(s)) : s \in \text{tr}\}$ 。
  - 6:   增广数据集  $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$ 。
  - 7:   在增广后的数据集上重新使用行为克隆训练策略  $\hat{\pi}_{i+1}$
  - 8: **end for**
- 

### 3.4 讨论

行为克隆是一个简单有效的算法。尽管行为克隆存在复合误差的问题，但这个问题可以认为是问题提供的信息有限（也就是说，学习者只被提供了一个包含专家状态和动作的数据集）。在不改变问题设定的前提下，可以从信息论的角度证明，任何算法（包括但不限于行为克隆）在最坏的情况下都会存在复合误差的问题 [45]。

即使针对 DAgger 这个算法，虽然从问题设定上，允许学习者可以与环境交互并且咨询专家。但是由于学习者事前并不知道专家策略的分布，因此对于恢复一个  $\varepsilon$ -最优的策略，DAgger 需要的总专家示例数量未必会比行为克隆需要的总专家示例数量要少；详细地讨论见 [45]。

## 第4章 对抗式模仿学习

### 内容提要

- 状态-动作分布匹配
- 生成对抗式模仿算法实现
- 极小极大优化
- 能更好利用环境信息的对抗式模仿学习算法

这一章主要介绍对抗式模仿学习（Adversarial Imitation Learning）[1, 65, 76]。状态-动作分布匹配是模仿学习的核心思想，我们将讲解为什么这一准则可以缓解复合误差问题。然后基于对偶表示，给出对抗式模仿学习的极小极大化建模[1, 24, 65]。最终，我们将介绍比较前沿的内容：如何能更好地利用转移函数信息来进一步地改善对抗式模仿学习算法[44, 45, 71]。

### 4.1 状态-动作分布匹配

考虑有限长度回合制马尔可夫决策过程。由值函数的对偶表示 ( $V(\pi) = \sum_{h=1}^H \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} P_h^\pi(s,a) r_h(s,a)$ ) 可知，当模仿策略  $\pi$  的状态-动作分布  $P_h^\pi$  和专家策略的状态-动作分布  $P_h^{\pi^E}$  之间的距离很小时，模仿学习的目标  $V^{\pi^E} - V^\pi$  也会很小。基于这一观察，生成对抗式模仿学习[1, 17, 65]的准则是专家策略的“状态-动作分布匹配”：

$$\min_{\pi \in \Pi} \sum_{h=1}^H \psi(P_h^\pi, P_h^{\pi^E}), \quad (4.1)$$

其中  $\psi$  是某种距离度量。

现在在一个自然的问题是，相比于行为克隆中“策略分布匹配”的准则，生成对抗式模仿学习中“状态-动作分布匹配”的准则的好处是什么？我们以图3.1里的马尔可夫决策过程 Reset Cliff 为例来说明。在 Reset Cliff 马尔可夫决策过程中，初始状态只会出现在前  $|\mathcal{S}| - 1$  个状态中，在前  $|\mathcal{S}| - 1$  状态上，执行专家动作之后，状态转移也服从初始状态分布，我们很容易地分析得到专家策略的状态-动作分布为

$$\begin{aligned} P_h^{\pi^E}(s, \text{green}) &= \rho(s), P_h^{\pi^E}(s, \text{blue}) = 0, \forall s \in \{1, 2, \dots, |\mathcal{S}| - 1\}, \forall h \in [H], \\ P_h^{\pi^E}(b, \text{green}) &= P_h^{\pi^E}(b, \text{blue}) = 0, \forall h \in [H]. \end{aligned}$$

其中 green, blue 分别表示专家动作和非专家动作。“状态-动作分布匹配”准则的优势在于：即使在数据集未被访问的状态上，“状态-动作分布匹配”准则也会帮助选择能够接近专家状态-动作分布的动作。具体而言，在 Reset Cliff 马尔可夫决策过程中，专家的状态-动作分布的最大特点是在坏状态“b”上的访问概率为0。基于“状态-动作分布匹配”的准则，模仿策略会避免访问坏状态“b”，进而会在状态  $1, \dots, |\mathcal{S}| - 1$  上去选择执行专家动作 green，避免复合误差的产生（即使这个专家动作可能在数据集里未曾出现）。但是，如果在“策略分布匹配”的准则下，如小节3.2所描述的，在未被访问的状态上，模仿策略会采取一个均匀分布的策略，从而以一定的概率转移到坏状态 b 上，再也不能得到奖赏，产生较大的复合误差。由此可见，相比于行为克隆中“策略分布匹配”的准则，生成对抗式模仿学习中“状态-动作分布匹配”的准则可以降低复合误差，提升模仿策略的累计回报。

考虑优化目标公式(4.1)，在实际中，我们并不知道专家策略的状态-动作分布  $P_h^{\pi^E}$ ，我们需要从有限的专家数据集  $\mathcal{D} = \{\text{tr}_i\}_{i=1}^m, \text{tr}_i = \{s_1^i, a_1^i, s_2^i, a_2^i, \dots, s_H^i, a_H^i\}$  来估计  $P_h^{\pi^E}$ ，一个最直接的估计方法便是最大似然估计[1, 57, 65]：

$$\hat{P}_h^{\pi^E}(s,a) = \frac{\sum_{i=1}^m \mathbb{I}\{s_h^i = s, a_h^i = a\}}{m}, \quad \forall (s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]. \quad (4.2)$$

有了专家策略的状态-动作分布的估计器  $\hat{P}_h^{\pi^E}(s,a)$ ，我们可以得到下面的优化目标：

$$\min_{\pi \in \Pi} \sum_{h=1}^H \psi(P_h^\pi, \hat{P}_h^{\pi^E}). \quad (4.3)$$



**笔记** 根据状态-动作分布匹配原则，学习到的策略与专家策略的动作分布差异（也就是说  $D_{\text{KL}}(\pi^{\text{E}}(\cdot|s), \pi(\cdot|s))$ ）未必是很小的，甚至是不随着训练过程单调递减的；这个现象在专家数据贫乏的时候尤为显现（可以参考 [72] 附录里的图 11 和图 12）。具体而言，智能体会倾向于选择任何可以使得产生的状态-动作分布能够与专家匹配的动作，而这样的动作不一定只是专家动作。因此不能直接把前面提到的行为克隆的损失函数作为对抗式模仿学习的评估准则（Validation Metric）和实验分析的指标。

我们考虑一个简单的基于“状态-动作分布匹配”准则的算法实例。我们将距离度量  $\psi$  选择为 TV-距离：

$$\hat{\pi} = \operatorname{argmin}_{\pi \in \Pi} \sum_{h=1}^H D_{\text{TV}}(P_h^{\pi}, \hat{P}_h^{\pi^{\text{E}}}), \quad (4.4)$$

$$\hat{P}_h^{\pi^{\text{E}}}(s, a) = \frac{\sum_{i=1}^m \mathbb{I}\{s_h^i = s, a_h^i = a\}}{m}, \quad \forall (s, a, h) \in \mathcal{S} \times \mathcal{A} \times [H]. \quad (4.5)$$

我们暂时不考虑公式 (4.4) 里的优化细节，假设可以得到上述优化目标的最优解  $\hat{\pi}$ 。下面的定理给出了这个算法的专家样本复杂度上界：

**定理 4.1 (对抗模仿学习的样本复杂度 [73])**

对于有限长度回合制马尔可夫决策过程，用  $\hat{\pi}$  来表示公式 (4.4) 中定义的策略，对于任意的  $\delta \in (0, 1)$  和  $\varepsilon \in (0, H)$ ，以高于  $1 - \delta$  的概率，当专家样本复杂度  $m \gtrsim \frac{H^2 |\mathcal{S}| \log(H/\delta)}{\varepsilon^2}$  时，那么我们有  $V(\pi^{\text{E}}) - V(\hat{\pi}) \leq \varepsilon$ 。❤

**注** 符号  $\gtrsim$  表示进行比较的时候可以忽略低阶项和常数项。

**笔记** 当忽略常数项和对数项时，公式 (4.4) 定义的算法的样本复杂度为  $\tilde{O}(H^2 |\mathcal{S}| / \varepsilon^2)$ 。当精度要求较高的时候，即  $\varepsilon \in (0, 1)$ ，这样的样本复杂度比行为克隆算法的样本复杂度  $\tilde{O}(H^2 |\mathcal{S}| / \varepsilon)$  要高。而且直接改变距离度量  $\psi$ ，并不会改善样本复杂度；比如后续要介绍的衍生算法 (FEM [1] 和 GTAL [65])，其样本复杂度也基本是  $\tilde{O}(H^2 |\mathcal{S}| / \varepsilon^2)$ 。另外，实验观测到（后面要介绍的图 4.3），这样的样本复杂度是紧的 [73]。这表明，为了得到更好的样本复杂度，我们需要改进算法，而不是改进理论分析技术。

**笔记** 我们看到，相比于行为克隆算法，基于“状态-动作分布匹配”准则进行设计的算法（公式 (4.4)）并没有在最坏情况（Worst-Case）下实现更好的样本复杂度。注意这与前面的分析得出状态-动作匹配能缓解复合误差问题并不矛盾。这里的重要原因是，由公式 (4.5) 定义的最大似然估计器  $\hat{P}_h^{\pi^{\text{E}}}$ ，集中到  $P_h^{\pi^{\text{E}}}$  的速率为  $\tilde{O}(\sqrt{1/m})$ ，即  $\|P_h^{\pi^{\text{E}}} - \hat{P}_h^{\pi^{\text{E}}}\|_1 \leq \tilde{O}(\sqrt{1/m})$  [68]，这会导致  $\tilde{O}(1/\varepsilon^2)$  的样本复杂度。在 4.3 小节，我们将介绍几个从估计器设计的角度来改进对抗模仿学习的算法，其样本复杂度比行为克隆算法会更好。

在本小节，基于专家策略的状态-动作分布的匹配的准则，我们导出了公式 (4.3) 中的目标函数，并阐述了状态-动作分布匹配的准则的优势。下面小节，我们将会介绍如何具体地求解公式 (4.3) 对应的优化问题。

## 4.2 极小极大优化建模

考虑公式 (4.1) 里的目标函数：

$$\min_{\pi \in \Pi} \sum_{h=1}^H \psi(P_h^{\pi}, \hat{P}_h^{\pi^{\text{E}}}).$$

其中  $\psi$  是某种距离度量。常用的距离度量包括  $f$ -距离 [29]。 $f$ -距离的定义为

$$D_f(P, Q) = \sum_x p(x) f\left(\frac{p(x)}{q(x)}\right),$$

其中  $f$  是一个凸函数且满足  $f(0) = 1$ ， $p(x)$  和  $q(x)$  分别为分布  $P$  和  $Q$  的概率密度。当  $f(x) = 1/2 \cdot |x - 1|$  时，我们便得到前面提到的 TV-距离。我们以 TV-距离为例，考虑优化下面的目标函数：

$$\min_{\pi \in \Pi} \sum_{h=1}^H D_{\text{TV}}(P_h^{\pi}, \hat{P}_h^{\pi^{\text{E}}}) = \frac{1}{2} \min_{\pi \in \Pi} \sum_{h=1}^H \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} |P_h^{\pi}(s, a) - \hat{P}_h^{\pi^{\text{E}}}(s, a)|.$$

直接优化上述目标函数比较困难：考虑最常见的梯度法，求解上述目标函数关于优化变量  $\pi$  的梯度就会比较困难。因此，我们考虑利用 TV-距离的对偶表示（即  $\ell_1$ -norm 的对偶范数 [69]），得到下面的极小极大目标：

$$\min_{\pi \in \Pi} \sum_{h=1}^H D_{\text{TV}} \left( P_h^\pi, \hat{P}_h^{\pi^E} \right) = \frac{1}{2} \min_{\pi \in \Pi} \max_{w \in \mathcal{W}} \sum_{(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]} w_h(s,a) \left( \hat{P}_h^{\pi^E}(s,a) - P_h^\pi(s,a) \right),$$

其中  $\mathcal{W} = \{w \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}| \times H} : \|w\|_\infty \leq 1\}$ 。

**注** 对于任何一个  $f$ -距离，我们都可以通过相应的对偶表示，得到对应的极小极大化目标，这一技术在生成对抗式网络中也有应用 [37]。

当把  $w$  看作奖赏函数时，根据值函数的对偶表示，我们可以得到下面的优化目标

$$\min_{\pi \in \Pi} \max_{w \in \mathcal{W}} V_w(\pi^E) - V_w(\pi),$$

其中  $V_w(\pi)$  表示在奖赏函数  $w$  下，策略  $\pi$  的值函数。在上面的极小极大化目标中，给定策略  $\pi$ ，变量  $w$  的目的是找到一个奖赏函数来最大化专家策略和模仿策略的价值差距；给定奖赏函数  $w$ ，变量  $\pi$  的目的是找到一个策略来最大化在奖赏函数  $w$  下的值函数  $V_w^\pi$ ，这与强化学习的目标一样。

我们可以通过梯度下降-上升（Gradient Descent Ascent）来优化极小极大化目标。具体而言，在第  $t+1$  轮迭代时，给定当前模仿策略  $\pi^{(t)}$ ，关于奖赏函数的优化目标为

$$\min_{w \in \mathcal{W}} F^{(t)}(w) := -\frac{1}{2} \sum_{(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]} w_h(s,a) \left( \hat{P}_h^{\pi^E}(s,a) - P_h^{\pi^{(t)}}(s,a) \right).$$

可以在线投影梯度下降 [22] 进行优化：

$$w^{(t+1)} = \mathcal{P}_{\mathcal{W}} \left( w^{(t)} - \eta_w \nabla F^{(t)}(w^{(t)}) \right).$$

其中  $\eta_w$  是步长， $\mathcal{P}_{\mathcal{W}}$  是投影算子： $\mathcal{P}_{\mathcal{W}}(w) = \operatorname{argmin}_{z \in \mathcal{W}} \|w - z\|_2^2$ 。给定奖赏函数  $w^{(t+1)}$ ，关于策略的优化目标为： $\max_{\pi \in \Pi} V_{w^{(t+1)}}(\pi)$ ，我们可以利用基本的求解 MDP 的算法（例如策略迭代 [42]、值迭代 [42] 和策略梯度 [2]）。

### 4.2.1 学徒学习

除了前面用到的 TV-距离，经典的学徒学习（Apprenticeship Learning）算法 Feature Expectation Matching [1]（FEM）和 Game Theoretic Apprenticeship Learning [65]（GTAL）也可以被对抗式模仿学习的框架所描述。在 FEM 和 GTAL 的原始论文里，作者假设真实的奖赏函数可以被一组已知的特征  $\phi : \mathcal{S} \times \mathcal{A} \times [H] \rightarrow [0, 1]^k$  线性表示： $r_h(s,a) = w^{*T} \phi_h(s,a)$ ，其中  $w^* \in \mathbb{R}^k$ 。对于任意一个策略  $\pi$ ，它的特征期望（Feature Expectation）定义为：

$$\mu(\pi) = \mathbb{E} \left[ \sum_{h=1}^H \phi_h(s_h, a_h) \middle| \pi \right] \in \mathbb{R}^k.$$

FEM 和 GTAL 的思想是进行专家的特征期望的匹配。具体地，

$$\min_{\pi} \mu(\pi^E) - \mu(\pi).$$

当我们选取  $k = |\mathcal{S}| \times |\mathcal{A}| \times H$  和特征  $\phi(s', a', h') = (\mathbb{I}(s = s', a = a', h = h'))_{s \in \mathcal{S}, a \in \mathcal{A}, h \in [H]}$  时，策略  $\pi$  的特征期望  $\mu(\pi)$  中的每个元素便是策略  $\pi$  的状态-动作分布。因此，FEM 和 GTAL 也可以看作是专家的状态-动作分布的匹配。FEM 和 GTAL 分别选择  $\ell_2$ -范数和  $\ell_\infty$ -范数作为距离度量，相应的极小极大化目标如下：

$$\text{FEM : } \min_{\pi \in \Pi} \max_{w \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}| \times H} : \|w\|_2 \leq 1} \sum_{(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]} w_h(s,a) \left( \hat{P}_h^{\pi^E}(s,a) - P_h^\pi(s,a) \right), \quad (4.6)$$

$$\text{GTAL : } \min_{\pi \in \Pi} \max_{w \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}| \times H} : \|w\|_1 \leq 1} \sum_{(s,a,h) \in \mathcal{S} \times \mathcal{A} \times [H]} w_h(s,a) \left( \hat{P}_h^{\pi^E}(s,a) - P_h^\pi(s,a) \right). \quad (4.7)$$

这里的  $\hat{P}_h^{\pi^E}(s,a)$  对应公式 (4.2)。FEM 和 GTAL 也是利用之前的梯度下降-上升思想来优化极小极大化目标。具体而言，FEM 会维护着一个状态-动作分布  $P^{\pi^{(t)}}$ ，给定当前的奖赏函数  $w^{(t)}$ ，FEM 先求解在奖赏函数  $w^{(t)}$  下的最优策略  $\tilde{\pi}$ ，通过线搜索的方式来更新状态-动作分布  $P^{\pi^{(t+1)}} = P^{\pi^{(t)}} + \eta(P^{\tilde{\pi}} - P^{\pi^{(t)}})$ ，其中  $\eta \in [0, 1]$  是通过线



搜索确定的步长，即

$$\eta = \operatorname{argmin}_{\eta \in [0,1]} \left\| \hat{P}^{\pi^E} - \left( P^{\pi^{(t)}} + \eta \left( P^{\tilde{\pi}} - P^{\pi^{(t)}} \right) \right) \right\|_2.$$

得到  $P^{\pi^{(t+1)}}$  之后，奖赏函数更新为  $w^{(t+1)} = \hat{P}^{\pi^E} - P^{\pi^{(t+1)}}$ 。

给定奖赏函数  $w^{(t)}$ ，类似地，GTAL 求解在奖赏函数  $w^{(t)}$  下的最优策略得到  $\pi^{(t)}$ 。考虑公式 (4.7)，注意到内层  $\max$  问题的最优解会在边界  $\|w\|_1 = 1$  的地方取到，即奖赏函数为概率单纯形。GTAL 利用了奖赏函数是概率单纯形这一特点，使用在线镜像下降算法（Online Mirror Descent [22]）来优化奖赏函数：

$$w_h^{(t+1)}(s, a) \propto w_h^t(s, a) \cdot \exp \left( \hat{P}_h^{\pi^E}(s, a) - P_h^{\pi^{(t)}}(s, a) \right), \quad \forall (s, a, h) \in \mathcal{S} \times \mathcal{A} \times [H].$$

### 4.2.2 生成对抗式模仿学习

在极小极大化目标下，一个著名模仿学习算法便是生成对抗式模仿学习算法（Generative Adversarial Imitation Learning, GAIL）[24]。为了简便和原始论文保持一致，我们在无限长度的折扣马尔可夫决策过程下介绍这个算法。

在 GAIL 中，作者选择 Jensen-Shannon 距离作为距离度量  $\psi$ ，进行专家策略的状态-动作分布的匹配：

$$\min_{\pi \in \Pi} D_{\text{JS}}(d^\pi, \hat{d}^{\pi^E}),$$

其中  $\hat{d}^{\pi^E}$  是专家的状态-动作分布的估计器。根据 Jensen-Shannon 距离的对偶表示，我们可以得到下面的极小极大化优化目标：

$$\min_{\pi \in \Pi} \max_{D: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}} \mathbb{E}_{(s,a) \sim \hat{d}^{\pi^E}} [\log(D(s,a))] + \mathbb{E}_{(s,a) \sim d^\pi} [\log(1 - D(s,a))]. \quad (4.8)$$

上述目标函数和生成对抗式网络（Generative Adversarial Networks, GAN）[19] 中的目标函数十分相近。其中  $D$  可以看作 GAN 中的判别器，作用是区分某个状态-动作对  $(s, a)$  是由专家策略  $\pi^E$  还是由模仿策略  $\pi$  产生，是一个二分类问题。 $\pi$  是生成器，目标是产生与专家数据相似的状态-动作对，提升判别器的打分，混淆判别器。生成对抗式模仿学习的算法流程如下：

---

#### Algorithm 2 生成对抗式模仿学习

---

**Input:** 专家示例  $\mathcal{D} = \{(s, a)\} \stackrel{i.i.d.}{\sim} d^{\pi^E}$ ，将策略参数和判别器参数分别初始化为  $\theta_0, w_0$ 。

- 1: **for**  $i = 1, 2, \dots, N$  **do**
- 2:   使用策略  $\pi_{\theta_i}$  在环境中采样，得到数据集  $\mathcal{D}_i = \{(s_t, a_t)_t\}$ 。
- 3:   通过下面的梯度公式来更新判别器参数  $w_i \rightarrow w_{i+1}$ ：

$$\sum_{(s,a) \in \mathcal{D}} \nabla_w \log(D_w(s, a)) + \sum_{(s,a) \in \mathcal{D}_i} \nabla_w \log(1 - D_w(s, a)).$$

- 4:   设置奖赏函数  $r(s, a) = -\log(1 - D_{w_{i+1}}(s, a))$ ，利用信赖域策略优化算法 [53] 来更新策略参数  $\theta_i \rightarrow \theta_{i+1}$ 。
- 5: **end for**

**Output:**

---

## 4.3 更高级的对抗模仿学习算法

在4.1小节中，我们举例说明了，相比于 BC 遵循的“策略分布匹配”准则，“状态-动作分布匹配”准则能够降低复合误差。但是，经典的对抗式模仿学习算法在最坏情况下的样本复杂度未必比行为克隆好。在本小节，我们将会介绍两个对抗式模仿学习的改进算法，这两个算法相较于传统的对抗式模仿学习算法具有更好的样本复杂度。在本小节，我们假设算法已知环境的转移概率并且专家策略是一个确定性策略。

### 4.3.1 TAIL

Transition-aware Adversarial Imitation Learning [73] (TAIL) 考虑的是对专家状态-动作分布  $P_h^{\pi^E}(s, a)$  进行更好的估计, 并且基于4.2小节介绍的极小极大化目标进行优化。

我们引入新的符号  $\text{tr}_h$ , 表示截断到时间步  $h$  的轨迹, 即  $\text{tr}_h = \{s_1, a_1, \dots, s_h, a_h\}$ 。根据截断轨迹  $\text{tr}_h$ , 我们考虑对  $P_h^{\pi^E}(s, a)$  进行如下的分解:

$$P_h^{\pi^E}(s, a) = \sum_{\text{tr}_h \in \text{Tr}_h} \mathbb{P}^{\pi^E}(\text{tr}_h) \mathbb{I}\{\text{tr}_h(s_h, a_h) = (s, a)\}.$$

其中  $\text{Tr}_h$  表示截断到时间步  $h$  的全部轨迹,  $\text{tr}_h(s_h, a_h)$  表示阶段轨迹  $\text{tr}_h$  在时间步  $h$  时访问的状态-动作对。即在时间步  $h$  访问状态-动作对  $(s, a)$  的概率为所有包含  $(s, a)$  的 (截断) 轨迹的发生概率。

我们引入截断轨迹的集合  $\text{Tr}_h^{\mathcal{D}} := \{\text{tr}_h : \forall h' \in [h], \text{tr}_h(s_{h'}) \in \mathcal{S}_{h'}(\mathcal{D})\}$ , 其中  $\text{tr}_h(s_{h'})$  表示在截断轨迹  $\text{tr}_h$  在时间步  $h'$  上访问的状态。也就是说,  $\text{Tr}_h^{\mathcal{D}}$  包含的截断轨迹上的任意一个状态, 都被数据集  $\mathcal{D}$  访问过。

我们考虑将数据集  $\mathcal{D}$  随机等分为两份:  $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_1^c$ , 其中  $|\mathcal{D}_1| = |\mathcal{D}_1^c| = m/2$ 。给定  $\mathcal{D}_1$ , 我们对截断轨迹进行划分, 可以得到

$$P_h^{\pi^E}(s, a) = \sum_{\text{tr}_h \in \text{Tr}_h^{\mathcal{D}_1}} \mathbb{P}^{\pi^E}(\text{tr}_h) \mathbb{I}\{\text{tr}_h(s_h, a_h) = (s, a)\} + \sum_{\text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}} \mathbb{P}^{\pi^E}(\text{tr}_h) \mathbb{I}\{\text{tr}_h(s_h, a_h) = (s, a)\}.$$

通过数据集  $\mathcal{D}_1$ , 我们知道被  $\mathcal{D}_1$  访问的状态上的专家动作, 因此我们可以利用马尔可夫决策过程的转移概率, 对上式右侧第一项  $\sum_{\text{tr}_h \in \text{Tr}_h^{\mathcal{D}_1}} \mathbb{P}^{\pi^E}(\text{tr}_h) \mathbb{I}\{\text{tr}_h(s_h, a_h) = (s, a)\}$  进行准确计算。对于上式右侧的第二项, 我们可以利用数据集  $\mathcal{D}_1^c$  进行最大似然估计:

$$\frac{1}{|\mathcal{D}_1^c|} \sum_{\text{tr}_h \in \mathcal{D}_1^c} \mathbb{I}\{\text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a)\}.$$

我们便得到了  $P_h^{\pi^E}(s, a)$  的新的估计器  $\tilde{P}_h^{\pi^E}(s, a)$ :

$$\tilde{P}_h^{\pi^E}(s, a) = \sum_{\text{tr}_h \in \text{Tr}_h^{\mathcal{D}_1}} \mathbb{P}^{\pi^E}(\text{tr}_h) \mathbb{I}\{\text{tr}_h(s_h, a_h) = (s, a)\} + \frac{1}{|\mathcal{D}_1^c|} \sum_{\text{tr}_h \in \mathcal{D}_1^c} \mathbb{I}\{\text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a)\}. \quad (4.9)$$

直观地说, 相比于估计器  $\hat{P}_h^{\pi^E}(s, a)$  (公式(4.5)), 在新估计器  $\tilde{P}_h^{\pi^E}(s, a)$  中, 第一部分是**利用转移概率准确计算**得到的, 第二部分用来估计的样本量 ( $m/2$ ) 与  $\hat{P}_h^{\pi^E}(s, a)$  中的  $m$  为同一量级。可以认为, 在相同样本量 ( $|\mathcal{D}| = m$ ) 下, 新估计器  $\tilde{P}_h^{\pi^E}(s, a)$  的误差比公式(4.2)里的最大似然估计器  $\hat{P}_h^{\pi^E}(s, a)$  的误差更小。

**注** 如果读者仍然对公式(4.9)的具体计算细节困惑, 可以参考[72]的附录C, 那里有详细的例子来说明。

有了新的估计器  $\tilde{P}_h^{\pi^E}(s, a)$  之后, 我们考虑4.2小节中的极小极大优化:

$$\min_{\pi \in \Pi} \sum_{h=1}^H D_{\text{TV}}(P_h^{\pi}, \tilde{P}_h^{\pi^E}) = \min_{\pi \in \Pi} \max_{w \in \mathcal{W}} \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} w_h(s, a) (\tilde{P}_h^{\pi^E}(s, a) - P_h^{\pi}(s, a)). \quad (4.10)$$

对于奖赏函数  $w_h(s, a)$ , 我们利用在线投影梯度法进行优化。对于策略  $\pi$ , 因为已知转移概率和奖赏函数, 我们利用基本的求解 MDP 的算法 (例如策略迭代[42]、值迭代[42]和策略梯度[2]) 得到一个  $\varepsilon_{\text{opt}}$ -最优的策略。TAIL 算法的流程图如下:

#### 定理 4.2 (TAIL 的样本复杂度 [73])

对于任意的  $\varepsilon \in (0, H)$  和  $\delta \in (0, 1)$ ; 假设  $H \geq 5$ . 考虑 TAIL 算法 (Algorithm 3), 令  $\bar{\pi}$  为算法输出的策略, 假设优化误差满足  $\varepsilon_{\text{opt}} \leq \varepsilon/2$ , 当专家样本复杂度、迭代轮数、步长满足

$$m \gtrsim \frac{H^{3/2}|\mathcal{S}|}{\varepsilon} \log\left(\frac{H|\mathcal{S}|}{\delta}\right), \quad T \gtrsim \frac{H^2|\mathcal{S}||\mathcal{A}|}{\varepsilon^2}, \quad \eta^{(t)} := \sqrt{\frac{|\mathcal{S}||\mathcal{A}|}{8T}}.$$

以至少  $1 - \delta$  的概率, 我们有  $V^{\pi^E} - V^{\bar{\pi}} \leq \varepsilon$ .



**笔记** 当忽略常数项和对数项时, TAIL 的专家样本复杂度上界为  $\tilde{O}(H^{3/2}|\mathcal{S}|/\varepsilon)$ 。相比于 BC 的专家样本复杂度  $\tilde{O}(H^2|\mathcal{S}|/\varepsilon)$ , TAIL 提升了  $\tilde{O}(H^{1/2})$ 。相比传统的对抗式模仿学习算法 (FEM 和 GTAL, 甚至包括 GAIL), 这


**Algorithm 3** Transition-aware Adversarial Imitation Learning (TAIL)


**Input:** 专家样本  $\mathcal{D}$ , 迭代轮数  $T$ , 步长  $\eta^{(t)}$ , 初始化奖赏函数  $w^{(1)}$ 。

- 1: 将专家数据集  $\mathcal{D}$  随机等分为两份:  $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_1^c$ , 得到公式 (4.9) 中的估计器  $\tilde{P}_h^{\pi^E}$ 。
- 2: **for**  $t = 1, 2, \dots, T$  **do**
- 3:  $\pi^{(t)} \leftarrow$  求解得到在奖赏函数  $w^{(t)}$  下的  $\varepsilon_{\text{opt}}$ -最优的策略。
- 4: 对于策略  $\pi^{(t)}$  求解对应的状态-动作分布  $P_h^{\pi^{(t)}}(s, a)$ 。
- 5: 利用在线投影梯度法更新奖赏函数:  $w^{(t+1)} := \mathcal{P}_{\mathcal{W}}(w^{(t)} - \eta^{(t)} \nabla F^{(t)}(w^{(t)}))$ , 其中  $F^{(t)}(w) = -\sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} w_h(s, a) (\tilde{P}_h^{\pi^E}(s, a) - P_h^{\pi^{(t)}}(s, a))$ 。
- 6: **end for**
- 7: 求解平均状态-动作分布:  $\bar{P}_h(s, a) = \sum_{t=1}^T P_h^{\pi^{(t)}}(s, a)/T$ 。
- 8: 导出策略  $\pi_h(a|s) \leftarrow \bar{P}_h(s, a) / \sum_a \bar{P}_h(s, a)$ 。

**Output:** 策略  $\pi$ 。

样的对抗式模仿学习算法在最坏情况下的样本复杂度要更好。

 **笔记** TAIL 里的数据集切分是为了利用行为克隆的里的“质量缺失” (Missing Mass) 性质 [45] (见小节 3.1 里的笔记)。正是因为这个性质, TAIL 才能够把定理 4.1 样本复杂度里的  $\tilde{O}(1/\varepsilon^2)$  改进到  $\tilde{O}(1/\varepsilon)$ 。

 **笔记** 这里我们假设环境的转移概率已知, 在每个迭代步  $t$  时, 可以容易地求解  $\varepsilon_{\text{opt}}$ -最优的策略  $\pi^{(t)}$ , 和计算  $\pi^{(t)}$  的状态-动作分布。当环境的转移概率未知时, 我们需要与环境进行交互, 在线地求解策略  $\pi^{(t)}$  和计算相应的状态-动作分布。在这种情形下, 除了专家样本量, 我们还关心与环境交互的样本量。在 [73] 中, 作者给出了当环境的转移概率未知时, 相应的算法和理论结果, 感兴趣的读者可以深入了解。

### 4.3.2 MIMIC-MD

我们介绍另一个基于“状态-动作分布匹配”准则的算法 MIMIC-MD [45]。事实上, MIMIC-MD 是第一个克服样本复杂度难题的对抗式模仿学习算法 (这个关系在 [44] 里被明确点出); 但是为了方便讲述这一原理, 我们先介绍了 TAIL, 后介绍 MIMIC-MD。

给定数据集  $\mathcal{D}$ , 我们定义一个在  $\mathcal{D}$  上, 由行为克隆算法产生的策略的集合  $\Pi_{\text{mimic}}(\mathcal{D})$ 。

$$\Pi_{\text{mimic}}(\mathcal{D}) \triangleq \left\{ \pi \in \Pi_{\text{det}} : \forall h \in [H], s \in \mathcal{S}_h(\mathcal{D}), \pi_h(\cdot | s) = \delta_{\pi_t^E(s)} \right\}, \quad (4.11)$$

其中  $\Pi_{\text{det}}$  表示包含全部确定性策略的集合,  $\mathcal{S}_h(\mathcal{D})$  表示数据集  $\mathcal{D}$  中在时间步  $h$  上访问的状态的集合,  $\delta_{\pi_t^E(s)}$  表示定义在动作空间上的狄拉克分布, 在专家动作  $\pi_t^E(s)$  上赋予 1 的概率, 在其他动作上赋予 0 的概率。也就是说, 策略集合  $\Pi_{\text{mimic}}(\mathcal{D})$  中的每一个策略, 在数据集  $\mathcal{D}$  包含的状态上, 会执行专家动作。

因为我们假设专家策略是确定性策略, MIMIC-MD 的思想就是在已经访问的状态上像行为克隆一样采取与专家一致的行为。这样的策略集合就是公式 (4.11) 里定义的集合。但显然直接做行为克隆并不可取, 所以 MIMIC-MD 还考虑了状态-动作对匹配。特别地, 我们仍把数据集  $\mathcal{D}$  随机等分为两份:  $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_1^c$ , 其中  $|\mathcal{D}_1| = |\mathcal{D}_1^c| = m/2$ 。MIMIC-MD 把待选策略的集合限定为  $\Pi_{\text{mimic}}(\mathcal{D}_1)$ , 这么做的动机是: 最后算法输出策略的性能一定不弱于行为克隆算法输出策略的性能, 因为待选策略的集合被限定为行为克隆输出策略的集合。

我们回顾在 4.3.1 小节中定义的截断轨迹的集合  $\text{Tr}_h^{\mathcal{D}}$ ,  $\text{Tr}_h^{\mathcal{D}}$  包含的截断轨迹上的任意一个状态, 都被数据集  $\mathcal{D}$  访问过。MIMIC-MD 的优化目标为

$$\min_{\pi \in \Pi_{\text{mimic}}(\mathcal{D}_1)} \sum_{h=1}^H \sum_{(s,a) \in \mathcal{S} \times \mathcal{A}} \left| \mathbb{P}^{\pi} \left( \text{tr}_h : \text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a) \right) - \frac{1}{|\mathcal{D}_1^c|} \sum_{\text{tr}_h \in \mathcal{D}_1^c} \mathbb{I} \left\{ \text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a) \right\} \right|. \quad (4.12)$$

给定数据集  $\mathcal{D}^1$ ,  $\mathbb{P}^{\pi}(\text{tr}_h : \text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a))$  表示策略  $\pi$  产生未完全被数据集  $\mathcal{D}_1$  访问, 且在时间步  $h$  上访问  $(s, a)$  的截断轨迹的概率。  $1/|\mathcal{D}_1^c| \cdot \sum_{\text{tr}_h \in \mathcal{D}_1^c} \mathbb{I} \left\{ \text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a) \right\}$  可以看作是在数据集  $\mathcal{D}_1^c$  上, 对概率  $\mathbb{P}^{\pi^E}(\text{tr}_h : \text{tr}_h \notin \text{Tr}_h^{\mathcal{D}_1}, \text{tr}_h(s_h, a_h) = (s, a))$  的最大似然估计。

优化目标 (公式 (4.12)) 可以看作是 TAIL 优化目标 (公式 (4.10)) 下的一个特例。因为限定待选策略集合为  $\Pi_{\text{mimic}}(\mathcal{D}_1)$ , 对于  $\pi \in \Pi_{\text{mimic}}(\mathcal{D}_1)$ ,  $\pi$  在数据集  $\mathcal{D}_1$  访问过的状态上是执行专家动作的, 所以对于集合  $\text{Tr}_h^{\mathcal{D}_1}$  中的任意一个截断轨迹  $\text{tr}_h$ , 都有  $\mathbb{P}^\pi(\text{tr}_h) = \mathbb{P}^{\pi^E}(\text{tr}_h)$ 。因此, 我们只需要匹配那些  $\text{Tr}_h^{\mathcal{D}_1}$  中没有包含的截断轨迹的概率。由此, 我们便可以得到公式 (4.12) 的目标函数。

下面的定理给出了 MIMIC-MD 的样本复杂度上界:

**定理 4.3 (MIMIC-MD 的样本复杂度 [45])**

考虑 MIMIC-MD 算法, 令  $\hat{\pi}$  为公式 (4.12) 下的最优解, 对于任意的  $\varepsilon \in (0, H)$  和  $\delta \in (0, 1)$ , 以至少  $1 - \delta$  的概率, 当专家样本复杂度  $m \gtrsim \frac{H^{3/2}|\mathcal{S}|}{\varepsilon} \log\left(\frac{H|\mathcal{S}|}{\delta}\right)$ ,  $V(\pi^E) - V(\hat{\pi}) \leq \varepsilon$ 。



**笔记** 公式 (4.12) 中定义的优化问题是无法在多项式时间内直接精确求解的。在 [44] 中, 作者将优化问题 (公式 (4.12)) 转化为了一个线性规划问题, 从而可以在多项式时间内精确求解。

**笔记** TAIL 和 MIMIC-MD 的除了解法的区别 (前者是基于梯度下降-上升, 后者是线性规划), 在算法设计上的区别在于 MIMIC-MD 直接将模仿策略的动作在数据集  $\mathcal{D}_1$  里状态上投影到了专家动作, 而 TAIL 并没有这么一个限制, 因此 TAIL 更灵活一些, 这一点对设计后续算法比较重要 [73]。

## 4.4 讨论

在本节里介绍的对抗式模仿学习算法, 之所以能可以克服复合误差问题, 是因为显式地或隐式地利用了转移函数的信息。因此这些算法往往要求转移函数已知, 或者可以与环境进行大量交互来获取转移函数的信息。关于对抗式模仿学习算法与环境交互的复杂度研究相对较少。模仿学习算法样本复杂度的总结可以见表 4.1 [73]。

**表 4.1:** 专家样本和环境交互复杂度总结 [73]。

	转移函数已知	转移函数未知	
	专家样本复杂度	专家样本复杂度	环境交互复杂度
BC [45]	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon}\right)$	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon}\right)$	0
FEM [1]	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon^2} + \frac{H^8 \mathcal{S} ^3 \mathcal{A} }{\varepsilon^5}\right)$	0
GTAL [65]	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon^2} + \frac{H^6 \mathcal{S} ^3 \mathcal{A} }{\varepsilon^3}\right)$	0
MIMIC-MD [45]	$\tilde{O}\left(\frac{H^{3/2} \mathcal{S} }{\varepsilon}\right)$	-	-
TAIL [73]	$\tilde{O}\left(\frac{H^{3/2} \mathcal{S} }{\varepsilon}\right)$	-	-
OAL [57]	-	$\tilde{O}\left(\frac{H^2 \mathcal{S} }{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{H^4 \mathcal{S} ^2 \mathcal{A} }{\varepsilon^2}\right)$
MB-TAIL [73]	-	$\tilde{O}\left(\frac{H^{3/2} \mathcal{S} }{\varepsilon}\right)$	$\tilde{O}\left(\frac{H^3 \mathcal{S} ^2 \mathcal{A} }{\varepsilon^2}\right)$
Lower Bound [44]	$\tilde{\Omega}\left(\frac{H^{3/2}}{\varepsilon}\right)$	$\tilde{\Omega}\left(\frac{H^{3/2}}{\varepsilon}\right)$	-

除此理论分析外, [73] 也进行了实验研究来帮助更好的理解。与之前的 (大规模) 实验性对比 [24, 27, 30] 不同, [73] 里考虑的任务更具有代表性, 能够反应实际情况里遇到的实验观察; 其实验研究结果更具有说服力, 更能揭示算法的本质。

首先, 我们说对于前面提到的满足的 PAC [35, 55] (Probably Approximately Correct) 保证的模仿学习算法, 如果其样本复杂度满足:  $m \gtrsim H^\alpha|\mathcal{S}|/\varepsilon^\beta$ , 那么累计回报差异满足  $V^{\pi^E} - V^\pi \lesssim H^{\alpha/\beta}|\mathcal{S}|^{1/\beta}/m^{1/\beta}$ 。更进一步地:

$$\log(V^{\pi^E} - V^\pi) \lesssim (\alpha/\beta) \log(H) - 1/\beta \log(m) + 1/\beta \log(|\mathcal{S}|).$$

也就是说, 如果以  $\log(H)$  为横轴、 $\log(V(\pi^E) - V(\pi))$  为纵轴的时候, 斜率“最坏”为  $\alpha/\beta$ 。同理, 以  $\log(m)$  为

横轴，纵轴不变的时候，斜率“最坏”为  $1/\beta$ 。“最坏”的情况可以在最难的马尔可夫决策过程上观测的。比如图3.1里的马尔可夫决策过程就是转移函数未知、只知道专家数据集情况下最坏的马尔可夫决策过程 [45]；在这种情况下，我们预期观测到行为克隆算法关于  $\log(H)$  的斜率为  $2/1 = 2$ 。对比之下，对抗式模仿学习算法关于  $\log(H)$  的斜率应该为 1。

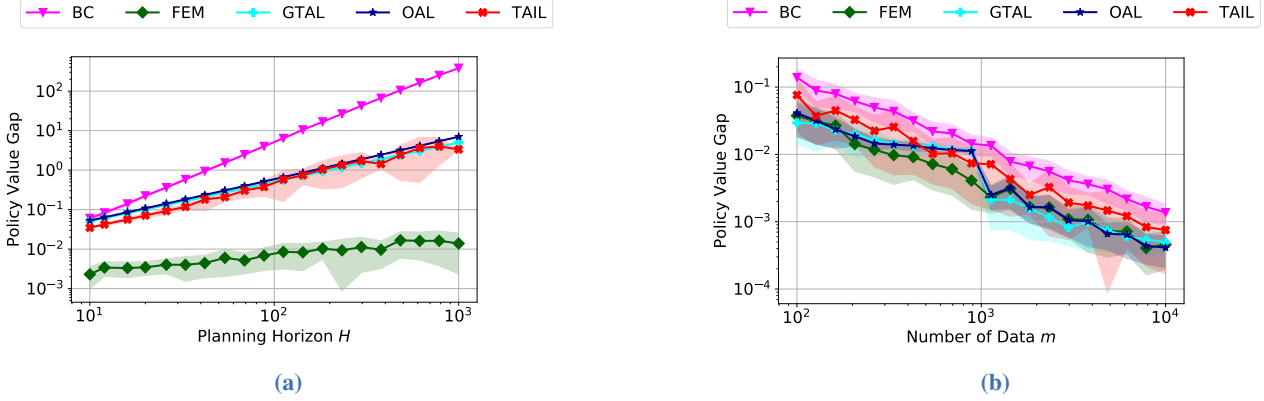


图 4.1: 在 Reset Cliff 马尔可夫决策过程（图 3.1）上模仿学习算法性能关于  $\log(V(\pi^E) - V(\pi))$  的对比 [73]。

Reset Cliff 上的性能对比见图4.1。在图 (a) 里可以看到我们预期的实验结果。在图 (b) 里我们看到几乎所有的模仿学习算法关于  $\log(m)$  的斜率都基本为 1。这对于行为克隆应该说是合理的（因为  $1/\beta = 1/1 = 1$ ）。但是对抗式模仿学习算法比如 FEM 和 GTAL 的斜率不是预期的  $1/\beta = 1/0.5 = 2$ ，这是因为 Reset Cliff 不是统计意义上难的任务。为了进一步地解决这个理论和实际差距的问题，[73] 又考虑了一个称作为 Standard Imitation [45] 的马尔可夫决策过程；见图4.2。

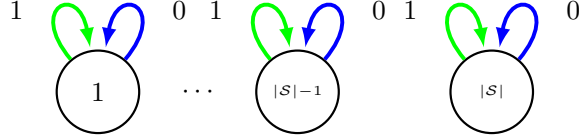


图 4.2: Standard Imitation 马尔可夫决策过程 [45, 73]。

在图4.2所示的马尔可夫决策过程中，每个状态都是一个独立的状态，采取任何动作都将停留在自己。但是，只有采取专家动作green才能获取 +1 的奖励，否则奖励为 0。假设初始状态分布为均匀分布。实验结果见图4.3。

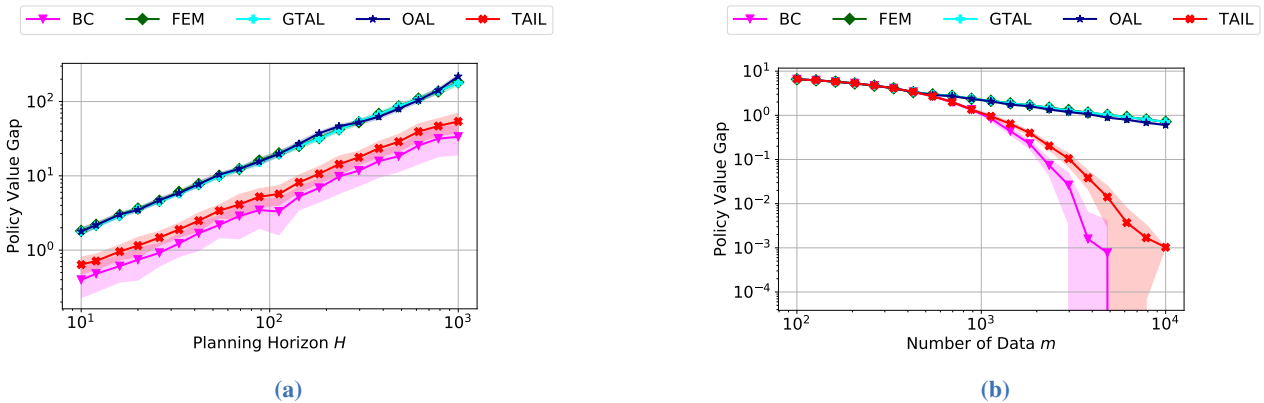


图 4.3: 在 Standard Imitation 马尔可夫决策过程（图 4.2）上模仿学习算法性能关于  $\log(V(\pi^E) - V(\pi))$  的对比 [73]。

考察图4.3。在图 (a) 里我们可以看到，所有算法关于  $\log(H)$  的斜率基本为 1，都不存在复合误差问题。但是



在图 (b) 里, 我们看到 TAIL 和 BC 的性能则更好, 其关于  $\log(m)$  的斜率接近  $1/\beta = 1/1 = 1$  (在  $m \in [10^2, 10^3]$  的区间内), 而传统的对抗式模仿学习算法的斜率则接近  $1/\beta = 1/2 = 0.5$ 。这表明传统的对抗式模仿学习算法在统计估计上存在缺陷, 而 TAIL 因为借助行为克隆算法里的“缺失质量”[45]的性质, 从而有较好地统计性能。

总结一下, 基于统计学习理论的思路, 我们可以更好地分析和理解对抗式模仿学习算法的优势、缺陷和极限, 并且设计出更好的对抗式模仿学习算法。[45] 是模仿学习里利用统计学习理论的先驱工作; 后续的工作包括 [44, 46, 72, 73]。把这些理论拓展到更接近实际的模仿学习任务 (比如多专家、环境未知) 是一项比较意义的工作。此外, 把这些理论算法和神经网络进行结合, 设计更好的实际算法也是一个不错的研究方向。

## 第5章 环境模仿

### 内容提要

- 基于模型的强化学习
- 基于行为克隆的环境模仿

- 基于对抗式的环境模仿

在第3章和第4章中，我们介绍了两大类模仿学习算法，用来模仿专家策略。如图5.1所示，在智能体与环境交互的过程中，除了智能体的策略之外，另一个重要的成分是环境的奖赏和转移函数。如果我们学到近似的奖赏和转移函数，我们便可以在这个近似的环境中进行策略学习，从而降低样本复杂度，这便是基于模仿的强化学习（Model-Based Reinforcement Learning）[63]的主要思想。在这一章，我们将介绍利用模仿学习算法来模仿环境的奖赏和转移函数。

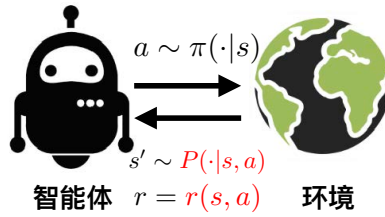


图 5.1: 智能体与环境交互示意图。

### 5.1 环境模仿学习

基于模型的强化学习 [63] 是一类在虚拟的马尔可夫决策过程中进行学习策略的算法。具体而言，基于模型的强化学习首先从与真实环境交互得到的样本中，学习到一个经验环境模型  $\hat{\mathcal{M}} = (\mathcal{S}, \mathcal{A}, \hat{\rho}, \hat{P}, \hat{r}, \gamma)$ ，这个经验模型包含近似的奖赏函数  $\hat{r}$  和转移函数  $\hat{P}$  和初始状态分布  $\hat{\rho}$ ，之后智能体便在这个虚拟的环境模型上进行采样和策略的学习。在基于模型的强化学习中，由于智能体可以在经验环境模型上完成采样，从而降低了在真实环境的样本复杂度，有助于提升强化学习算法的样本效率。基于模型的强化学习算法的一般算法框架如下：

---

#### Algorithm 4 基于模型的强化学习

---

**Input:** 初始化策略  $\pi$ ，奖赏函数  $\hat{r}$ ，转移函数  $\hat{P}$ ，初始状态分布  $\hat{\rho}$ ，和空数据集  $\mathcal{D} = \emptyset$ 。

- 1: **for**  $i = 1, 2, \dots, N$  **do**
- 2:   使用策略  $\pi$  在环境中采样，得到数据集  $\mathcal{D} = \mathcal{D} \cup \{(s_t, a_t, s'_{t+1})\}$ 。
- 3:   通过数据集  $\mathcal{D}$  训练奖赏函数  $\hat{r}$  和转移函数  $\hat{P}$  和初始状态分布  $\hat{\rho}$ 。
- 4:   在经验环境模型中优化策略:  $\pi \leftarrow \arg\max_{\pi'} V_{\hat{\mathcal{M}}}^{\pi'}$ 。
- 5: **end for**

**Output:**

---

环境模型的学习是基于模型的强化学习算法中重要的一步。可以看到，由于策略学习是基于经验模型  $\hat{r}$  和  $\hat{P}$  和  $\hat{\rho}$ ，经验模型的精度直接影响学习到的策略的性能：经验模型的误差越大，学到的策略的性能就会越差。

构建环境模型的过程中，可以把真实环境的奖赏转移函数以及初始状态分布视作“专家”，因此模仿（策略）学习的算法可以被应用到模仿环境模型的任务中。其中，模仿奖励函数和初始状态分布比较简单：对于前者，我们只需要利用回归模型构建一个从  $\mathcal{S} \times \mathcal{A}$  到  $\mathbb{R}$  的映射即可<sup>1</sup>；对于后者，我们只要学习一个生成模型来模拟初始状态分布。而模仿转移函数则比较困难，这是因为转移函数一般是一个高维的条件概率分布。在本章后面的小节中，我们将从模仿学习的角度来考虑如何构建一个高精度的经验转移模型。

---

<sup>1</sup>并且学习奖赏函数的统计复杂度相对于学习转移函数而言往往是一个低阶项 [6, 7]。



## 5.2 基于行为克隆的环境模仿

在这一节，我们考虑利用行为克隆算法进行环境模仿。环境的转移函数的输入是当前的状态-动作对  $(s, a)$ ，输出是下一时刻的状态  $s'$ ，因此可以用一个“对偶智能体”对环境的转移函数进行建模。图5.2展示了智能体与对偶智能体之间的关系图。根据前面的介绍，我们可以利用模仿学习算法来训练“对偶智能体”：专家策略是环境的真实转移函数  $P$ ，对偶智能体的策略是经验转移函数  $\hat{P}_\theta$  ( $\theta$  是经验转移模型的参数)。我们首先考虑利用行为克隆算法来模仿环境。

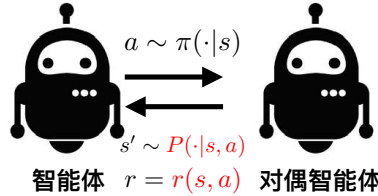


图 5.2: 智能体与对偶智能体示意图。

考虑用某个采样策略  $\pi^D$  与真实环境交互，得到数据集  $\mathcal{D} = \{(s, a, s')\}$ ，其中  $(s, a) \sim d_{\mathcal{M}}^{\pi^D}$ ,  $s' \sim P(\cdot|s, a)$ ， $d_{\mathcal{M}}^{\pi^D}$  表示采样策略  $\pi^D$  在真实环境  $\mathcal{M}$  上的状态-动作分布。在行为克隆里，我们基于最大化似然的准则可以得到下面的目标函数：

$$\begin{aligned} \max_{\hat{P}_\theta \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}| \times |\mathcal{S}|}} \quad & \sum_{(s, a, s') \in \mathcal{D}} \log \left( \hat{P}_\theta(s'|s, a) \right), \\ \text{s.t.} \quad & \sum_{s'} \hat{P}_\theta(s'|s, a) = 1 \quad \forall (s, a) \in \mathcal{S} \times \mathcal{A}. \end{aligned}$$

上述优化问题的对于有限状态和动作空间的马尔可夫决策过程的最优解为

$$\hat{P}_\theta(s'|s, a) = \frac{\sum_{(s_i, a_i, s'_i) \in \mathcal{D}} \mathbb{I}(s_i = s, a_i = a, s'_i = s')}{\sum_{(s_i, a_i, s'_i) \in \mathcal{D}} \mathbb{I}(s_i = s, a_i = a)}.$$

对于状态空间为连续空间、动作空间为连续空间或者两者都为连续空间的情况下，我们可以利用小节3.1里介绍的参数化方法（softmax 参数化、Gaussian 参数化）来进行下面的优化问题：

$$\min_{\theta} \sum_{(s, a, s') \in \mathcal{D}} \log \left( \hat{P}_\theta(s'|s, a) \right).$$

如何评价学习到的转移函数质量呢？最直接的指标是： $\max_{s, a} D_{\text{KL}}(P(\cdot|s, a), \hat{P}_\theta(\cdot|s, a))$ 。但考虑到我们的背景任务是基于模型的强化学习，我们更希望知道在经验模型里估计的值函数与真实环境下的值函数的差异，其对策略优化非常重要。所以，研究者们一般采用这个指标来衡量模型学习的好坏。

### 引理 5.1 (基于行为克隆的环境学习的策略评估误差 [28, 33, 62, 71])

考虑无限长度的折扣马尔可夫决策过程。给定一个数据收集策略  $\pi_D$ ，假设行为克隆算法被用于环境模仿学习，并且有  $\mathbb{E}_{(s, a) \sim d_{\mathcal{M}}^{\pi_D}} [D_{\text{KL}}(P(\cdot|s, a), \hat{P}_\theta(\cdot|s, a))] \leq \varepsilon_m$ 。这时，给定任意一个满足  $\max_s D_{\text{KL}}(\pi(\cdot|s), \pi_D(\cdot|s)) \leq \varepsilon_\pi$  的评估策略  $\pi$ ，策略评估误差的上界为： $|V_{\mathcal{M}}^\pi - V_{\mathcal{M}}^{\pi_D}| \leq \sqrt{2}/(1-\gamma)^2 \sqrt{\varepsilon_m} + 2\sqrt{2}/(1-\gamma)^2 \sqrt{\varepsilon_\pi}$ 。



**笔记** 引理5.1揭示了行为克隆得到的转移模型，在进行策略评估时，关于优化误差的系数是有效决策长度  $1/(1-\gamma)$  的二次方，即环境模仿学习的“复合误差”。

**注** 有一些基于“短视”（short rollout）的方法 [28, 32] 在一定条件下也可以做到评估误差关于  $\sqrt{\varepsilon_m}$  的系数是  $1/(1-\gamma)$  的一次方，但这些方法并没有本质上解决复合误差的问题。

### 5.3 基于对抗式的环境模仿

根据前面的介绍，对抗式模仿学习算法有望在序列决策任务中克服复合误差的难题。因此，我们可以考虑把对抗式策略模仿学习的方法也拓展到环境模仿的任务中。这时，对偶智能体的输入仍然是当前的状态-动作对  $(s, a)$ ，输出是下一时刻的状态  $s'$ 。不同之前策略模仿的是，我们需要将状态-动作分布  $d^\pi(s, a)$  的概念拓展为状态-动作-下一状态的分布  $d^\pi(s, a, s')$ 。相当于说，对偶智能体的状态空间是之前原始的状态空间和动作空间的增广，对偶智能体的动作空间是原始的（下一时刻的）状态空间。

$$d^\pi(s, a, s') = d^\pi(s, a) \cdot P(s'|s, a).$$

利用这种建模方法，我们便可以使用之前的对抗式模仿学习算法，比如 GAIL。以此为例，我们也可以分析策略评估误差。

#### 引理 5.2 (基于对抗式模仿学习的环境学习的策略评估误差 [71])

考虑无限长度的折扣马尔可夫决策过程。给定一个数据收集策略  $\pi_D$ ，假设 GAIL 方法被用于环境模仿学习，并且有  $D_{\text{JS}}(d_{\mathcal{M}}^\pi(s, a, s'), d_{\widehat{\mathcal{M}}}^\pi(s, a, s')) \leq \varepsilon_m$ 。这时，给定任意一个满足  $\max_s D_{\text{KL}}(\pi(\cdot|s), \pi_D(\cdot|s)) \leq \varepsilon_\pi$  的评估策略  $\pi$ ，策略评估误差的上界为： $|V_{\mathcal{M}}^\pi - V_{\widehat{\mathcal{M}}}^\pi| \leq 2\sqrt{2}/(1-\gamma)\sqrt{\varepsilon_m} + 2\sqrt{2}/(1-\gamma)^2\sqrt{\varepsilon_\pi}$ 。



**笔记** 引理5.2表明当 GAIL 的优化误差很小的时候，由于 GAIL 的策略评估误差关于优化误差  $\sqrt{\varepsilon_m}$  的系数是有效决策长度  $1/(1-\gamma)$  的一次方，所以 GAIL 方法在环境模仿时效果可能会更好。关于这方面的实验性工作可以参考 [70, 71]。

**注** 引理5.2里的具体环境模仿学习算法的流程，可以参考 [71] 附录里的算法 1。

## 第6章 总结

这篇教程主要从统计的角度揭示了模仿学习算法的本质。针对行为克隆算法，详细地介绍了复合误差的来源。此后，介绍了对抗式模仿学习是怎么缓解复合误差问题的，并且点出了现在的对抗式模仿学习算法的统计缺陷，以及最新的方法是如何克服这一难题的。最后，介绍了策略模仿学习在环境模仿学习里的应用。我们希望这个教程能帮助更多的人了解、改进和应用模仿学习算法。

由于时间仓促和能力有限，我们意识到一些与模仿学习相关但没有详细陈述的主题并没有详细地介绍。这些主题包括：

- 非有限状态和动作空间的理论结果。当使用神经网络等函数近似的时候，除了统计估计的问题，还有全局优化的问题，这方面的研究进展包括 [10, 21, 75]。
- 除了教程里提交到模仿学习算法，另外一类算法是 AggreVaTe (Aggregate Values to Imitate) 族 [11, 47, 61]。像 DAgger 一样，这也是一类基于在线学习算法的模仿学习算法。
- 模仿学习与强化学习的结合。一些任务中，给定的专家策略可能不是最优的，这时候模仿学习一般是为后续的强化学习做铺垫。把模仿学习和强化学习结合的文章可以参考 [13, 23, 38, 60]。
- VTR 类环境模仿。除了本文提到的行为克隆和对抗式模仿学习在环境模仿里的应用，最近提出来的一类构建经验转移函数的方式称作为：Value-Targeted-Regression (VTR)。简单来讲，这类方法的目标是直接最小化估计的值函数与真实值函数的差异。这类方法的文献可以参考 [5, 15, 20, 52]。
- 模仿学习里的探索问题。针对对抗式模仿学习算法，其需要与环境不断交互来进行模仿，如何在学习的过程中平衡探索和利用（模仿）也是一个非常重要的问题，这部分研究包括 [12, 57, 73]。

## 参考文献

- [1] Pieter Abbeel and Andrew Y. Ng. “Apprenticeship learning via inverse reinforcement learning”. In: *Proceedings of the 21st International Conference on Machine Learning*. 2004, pp. 1–8.
- [2] Alekh Agarwal et al. “Optimality and Approximation with Policy Gradient Methods in Markov Decision Processes”. In: *Proceedings of the 33rd Conference on Learning Theory*. 2020, pp. 64–66.
- [3] Alekh Agarwal et al. *Reinforcement Learning Theory and Algorithms*. 2020. URL: <https://rltheorybook.github.io/>.
- [4] Brenna D Argall et al. “A survey of robot learning from demonstration”. In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.
- [5] Alex Ayoub et al. “Model-Based Reinforcement Learning with Value-Targeted Regression”. In: *Proceedings of the 37th International Conference on Machine Learning*. 2020, pp. 463–474.
- [6] Mohammad Gheshlaghi Azar, Rémi Munos, and Hilbert J. Kappen. “Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model”. In: *Machine Learning* 91.3 (2013), pp. 325–349.
- [7] Mohammad Gheshlaghi Azar, Ian Osband, and Rémi Munos. “Minimax Regret Bounds for Reinforcement Learning”. In: *Proceedings of the 34th International Conference on Machine Learning*. 2017, pp. 263–272.
- [8] Michael Bain and Claude Sammut. “A Framework for Behavioural Cloning”. In: *Machine Intelligence 15: Intelligent Agents*. 1995, pp. 103–129.
- [9] Kianté Brantley, Wen Sun, and Mikael Henaff. “Disagreement-Regularized Imitation Learning”. In: *Proceedings of the 8th International Conference on Learning Representations*. 2020.
- [10] Minshuo Chen et al. “On Computation and Generalization of Generative Adversarial Imitation Learning”. In: *Proceedings of the 8th International Conference on Learning Representations*. 2020.
- [11] Ching-An Cheng and Byron Boots. “Convergence of Value Aggregation for Imitation Learning”. In: *International Conference on Artificial Intelligence and Statistics*. 2018, pp. 1801–1809.
- [12] Ching-An Cheng et al. “Accelerating Imitation Learning with Predictive Models”. In: *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*. 2019, pp. 3187–3196.
- [13] Ching-An Cheng et al. “Fast Policy Learning through Imitation and Reinforcement”. In: *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*. 2018, pp. 845–855.
- [14] Felipe Codevilla et al. “End-to-End Driving Via Conditional Imitation Learning”. In: *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*. 2018, pp. 1–9.
- [15] Amir-massoud Farahmand. “Iterative Value-Aware Model Learning”. In: *Advances in Neural Information Processing Systems* 31. 2018, pp. 9090–9101.
- [16] Justin Fu, Katie Luo, and Sergey Levine. “Learning Robust Rewards with Adversarial Inverse Reinforcement Learning”. In: *Proceedings of the 6th International Conference on Learning Representations*. 2018.
- [17] Seyed Kamyar Seyed Ghasemipour, Richard S. Zemel, and Shixiang Gu. “A Divergence Minimization Perspective on Imitation Learning Methods”. In: *Proceedings of the 3rd Annual Conference on Robot Learning*. 2019, pp. 1259–1277.
- [18] Alessandro Giusti et al. “A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots”. In: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 661–667.

- [19] Ian J. Goodfellow et al. “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems* 27. 2014, pp. 2672–2680.
- [20] Christopher Grimm et al. “The Value Equivalence Principle for Model-Based Reinforcement Learning”. In: *Advances in Neural Information Processing Systems* 33. 2020.
- [21] Ziwei Guan, Tengyu Xu, and Yingbin Liang. “When Will Generative Adversarial Imitation Learning Algorithms Attain Global Convergence”. In: *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*. 2021, pp. 1117–1125.
- [22] Elad Hazan. “Introduction to Online Convex Optimization”. In: *arXiv* 1909.05207 (2019).
- [23] Todd Hester et al. “Deep Q-learning From Demonstrations”. In: *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 2018, pp. 3223–3230.
- [24] Jonathan Ho and Stefano Ermon. “Generative Adversarial Imitation Learning”. In: *Advances in Neural Information Processing Systems* 29. 2016, pp. 4565–4573.
- [25] Jonathan Ho, Jayesh K. Gupta, and Stefano Ermon. “Model-Free Imitation Learning with Policy Optimization”. In: *Proceedings of the 33rd International Conference on Machine Learning*. 2016, pp. 2760–2769.
- [26] Ahmed Hussein et al. “Imitation Learning: A Survey of Learning Methods”. In: *ACM Computing Surveys* 50.2 (2017), pp. 1–35.
- [27] Léonard Hussenot et al. “Hyperparameter Selection for Imitation Learning”. In: *Proceedings of the 38th International Conference on Machine Learning*. 2021, pp. 4511–4522.
- [28] Michael Janner et al. “When to Trust Your Model: Model-Based Policy Optimization”. In: *Advances in Neural Information Processing Systems* 32. 2019, pp. 12498–12509.
- [29] Liyiming Ke et al. “Imitation Learning as f-Divergence Minimization”. In: *arXiv* 1905.12888 (2019).
- [30] Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. “Imitation Learning via Off-Policy Distribution Matching”. In: *Proceedings of the 8th International Conference on Learning Representations*. 2020.
- [31] Ilya Kostrikov et al. “Discriminator-Actor-Critic: Addressing Sample Inefficiency and Reward Bias in Adversarial Imitation Learning”. In: *Proceedings of the 7th International Conference on Learning Representations*. 2019.
- [32] Hang Lai et al. “Bidirectional Model-based Policy Optimization”. In: *Proceedings of the 37th International Conference on Machine Learning*. 2020, pp. 5618–5627.
- [33] Yuping Luo et al. “Algorithmic Framework for Model-based Deep Reinforcement Learning with Theoretical Guarantees”. In: *Proceedings of the 7th International Conference on Learning Representations*. 2019.
- [34] David A. McAllester and Luis E. Ortiz. “Concentration Inequalities for the Missing Mass and for Histogram Rule Error”. In: *Journal of Machine Learning Research* 4 (2003), pp. 895–911.
- [35] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT Press, 2018.
- [36] Andrew Y. Ng and Stuart J. Russell. “Algorithms for Inverse Reinforcement Learning”. In: *Proceedings of the 17th International Conference on Machine Learning*, pp. 663–670.
- [37] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. “f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization”. In: *Advances in Neural Information Processing Systems* 29. 2016, pp. 271–279.
- [38] Junhyuk Oh et al. “Self-Imitation Learning”. In: *Proceedings of the 35th International Conference on Machine Learning*. 2018, pp. 3875–3884.

- [39] Takayuki Osa et al. “An Algorithmic Perspective on Imitation Learning”. In: *Foundations and Trends in Robotic* 7.1-2 (2018), pp. 1–179.
- [40] Dean Pomerleau. “Efficient Training of Artificial Neural Networks for Autonomous Navigation”. In: *Neural Computation* 3.1 (1991), pp. 88–97.
- [41] Dean A Pomerleau. “Alvin: An autonomous land vehicle in a neural network”. In: *Advances in Neural Information Processing System*. 1989.
- [42] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.
- [43] Ahmed Hussain Qureshi, Byron Boots, and Michael C. Yip. “Adversarial Imitation via Variational Inverse Reinforcement Learning”. In: *Proceedings of the 7th International Conference on Learning Representations*. 2019.
- [44] Nived Rajaraman et al. “Provably Breaking the Quadratic Error Compounding Barrier in Imitation Learning, Optimally”. In: *arXiv* 2102.12948 (2021).
- [45] Nived Rajaraman et al. “Toward the Fundamental Limits of Imitation Learning”. In: *Advances in Neural Information Processing Systems* 33. 2020, pp. 2914–2924.
- [46] Paria Rashidinejad et al. “Bridging Offline Reinforcement Learning and Imitation Learning: A Tale of Pessimism”. In: *arXiv* 2103.12021 (2021).
- [47] Stephane Ross and J Andrew Bagnell. “Reinforcement and imitation learning via interactive no-regret learning”. In: *arXiv* 1406.5979 (2014).
- [48] Stéphane Ross and Drew Bagnell. “Efficient reductions for imitation learning”. In: *Proceedings of the 13rd International Conference on Artificial Intelligence and Statistics*. 2010, pp. 661–668.
- [49] Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. “A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning”. In: *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*. 2011, pp. 627–635.
- [50] Stuart J. Russell. “Learning Agents for Uncertain Environments (Extended Abstract)”. In: *Proceedings of the 11st Conference on Learning Theory*. 1998, pp. 101–103.
- [51] Stefan Schaal. “Is imitation learning the route to humanoid robots?” In: *Trends in cognitive sciences* 3.6 (1999), pp. 233–242.
- [52] Julian Schrittwieser et al. “Mastering atari, go, chess and shogi by planning with a learned model”. In: *Nature* 588.7839 (2020), pp. 604–609.
- [53] John Schulman et al. “Trust Region Policy Optimization”. In: *Proceedings of the 32nd International Conference on Machine Learning*. 2015, pp. 1889–1897.
- [54] Shai Shalev-Shwartz. “Online Learning and Online Convex Optimization”. In: *Foundations and Trends in Machine Learning* 4.2 (2012), pp. 107–194.
- [55] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.
- [56] Wenjie Shang et al. “Environment Reconstruction with Hidden Confounders for Reinforcement Learning based Recommendation”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 566–576.
- [57] Lior Shani, Tom Zahavy, and Shie Mannor. “Online Apprenticeship Learning”. In: *arXiv* 2102.06924 (2021).
- [58] Jing-Cheng Shi et al. “Virtual-Taobao: virtualizing Real-World Online Retail Environment for Reinforcement Learning”. In: *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. 2019, pp. 4902–4909.



- [59] David Silver et al. “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587 (2016), pp. 484–489.
- [60] Wen Sun, J. Andrew Bagnell, and Byron Boots. “Truncated horizon Policy Search: Combining Reinforcement Learning & Imitation Learning”. In: *Proceedings of the 6th International Conference on Learning Representations*. 2018.
- [61] Wen Sun et al. “Deeply AggreVaTeD: Differentiable Imitation Learning for Sequential Prediction”. In: *Proceedings of the 34th International Conference on Machine Learning*. 2017, pp. 3309–3318.
- [62] Wen Sun et al. “Dual Policy Iteration”. In: *Advances in Neural Information Processing Systems 31*. 2018, pp. 7059–7069.
- [63] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.
- [64] Richard S. Sutton et al. “Policy Gradient Methods for Reinforcement Learning with Function Approximation”. In: *Advances in Neural Information Processing Systems 12*. 1999, pp. 1057–1063.
- [65] Umar Syed and Robert E. Schapire. “A Game-Theoretic Approach to Apprenticeship Learning”. In: *Advances in Neural Information Processing Systems 20*. 2007, pp. 1449–1456.
- [66] Umar Syed and Robert E. Schapire. “A Reduction from Apprenticeship Learning to Classification”. In: *Advances in Neural Information Processing Systems 23*. 2010, pp. 2253–2261.
- [67] Faraz Torabi, Garrett Warnell, and Peter Stone. “Behavioral Cloning from Observation”. In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 2018, pp. 4950–4957.
- [68] Tsachy Weissman et al. “Inequalities for the L1 deviation of the empirical distribution”. In: *Hewlett-Packard Labs, Tech. Rep* (2003).
- [69] Wikipedia contributors. *Dual norm — Wikipedia, The Free Encyclopedia*. [https://en.wikipedia.org/w/index.php?title=Dual\\_norm&oldid=1029266114](https://en.wikipedia.org/w/index.php?title=Dual_norm&oldid=1029266114). 2021.
- [70] Yueh-Hua Wu et al. “Model imitation for model-based reinforcement learning”. In: *arXiv* 1909.11821 (2019).
- [71] Tian Xu, Ziniu Li, and Yang Yu. “Error Bounds of Imitating Policies and Environments”. In: *Advances in Neural Information Processing Systems 33*. 2020, pp. 15737–15749.
- [72] Tian Xu, Ziniu Li, and Yang Yu. “Error Bounds of Imitating Policies and Environments for Reinforcement Learning”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).
- [73] Tian Xu, Ziniu Li, and Yang Yu. “Nearly Minimax Optimal Adversarial Imitation Learning with Known and Unknown Transitions”. In: *arXiv* 2106.10424 (2021).
- [74] Lantao Yu et al. “SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient”. In: *Proceedings of the 31st AAAI Conference on Artificial Intelligence*. 2017, pp. 2852–2858.
- [75] Yufeng Zhang et al. “Generative Adversarial Imitation Learning with Neural Network Parameterization: Global Optimality and Convergence Rate”. In: *Proceedings of the 37th International Conference on Machine Learning*. 2020, pp. 11044–11054.
- [76] Brian D. Ziebart et al. “Maximum Entropy Inverse Reinforcement Learning”. In: *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*. 2008, pp. 1433–1438.