DS4001-25SP-HW2：强化学习

TA：戴维

2025 年 4 月 11 日

作业说明

本次作业包括书面部分（问题回答）和编程部分（代码填空）。

书面部分需使用 LaTeX 完成，模板文件为 `report_template.tex`。注意：为方便助教批改，你还需要将编程部分完成后的代码复制到模板中的对应位置。书面作业完成后，请导出为 `report.pdf` 文件。

编程部分的完整代码位于 `Project` 文件夹中。你仅需在 `submission.py` 中指定位置完成代码，具体如下：

```
# BEGIN_YOUR_CODE

# END_YOUR_CODE
```

不要对除 `submission.py` 以外的文件进行更改。在完成所有代码编写后，你可以运行 `python learner.py` 学习到 Q 值表，之后运行 `python evaluator.py` 进行评估。

完成所有内容后，请将 `submission.py` 与 `report.pdf` 打包成名为“学号 _ 姓名 _HW2.zip”的压缩文件，并上传至 BB 系统。助教将根据你提交的文件进行评分。

注意事项:

- 本次作业必须独立完成，任何形式的抄袭都不允许。如发现有互相抄袭的情况，抄袭者将平分获得的分数。
- 作业的截止时间为 2024 年 5 月 8 日 23:59:59。迟交 1/2/4/7 天分别会扣除 5/15/40/100 分。
- 如果在做作业过程中遇到问题，可以在 Github Issue 中提问或者线下在答疑课提问，也欢迎同学们相互解答问题。

作业正文

1 问题 1：马尔可夫决策过程 [9%]

考虑以下一个 MDP：

- 状态空间 $S = \{-2, -1, 0, 1, 2\}$；
- 动作空间 $A = \{a_1, a_2\}$；
- 初始状态 $s_0 = 0$，某一时刻 t 的状态 $s_t \in \{-2, 2\}$ 时，该过程结束；
- 在时刻 t，有如下的定义 $P_{s',s,a} = \text{Prob}[s_{t+1} = s' | s_t = s, a_t = a]$。假设在环境噪声的影响下，有转移概率：

$$\begin{cases} P_{i+1,i,a_1} = 0.2, P_{i,i,a_1} = 0.1, P_{i-1,i,a_1} = 0.7 \\ P_{i+1,i,a_2} = 0.3, P_{i,i,a_2} = 0.2, P_{i-1,i,a_2} = 0.5 \end{cases} \tag{1}$$

- 奖励函数

$$\begin{aligned} \mathcal{R}(s, a, s') =& R(s_{t+1} = s' | s_t = s, a_t = a) \\ =& \begin{cases} 20, & s' = 2 \\ 10, & s' = -2 \\ -1, & otherwise \end{cases} \end{aligned}$$

- 衰减因子 $\gamma = 1$。

(a) [6 分] 我们使用 Value Iteration 来评估改 MDP 的状态价值函数 $V(s)$。对于迭代轮次 $i \in \{0, 1, 2\}$，对应的状态价值函数为 $V^{(i)}(s)$。假设 $V^{(0)} \equiv 0$，请你求出所有所有的 $V^{(i)}(s), i \in \{0, 1, 2\}$ 的值。

(b) [3 分] 请根据你计算得到的 $V^{(2)}(s)$ 值，给出当前迭代轮次（$i = 2$）时的最优策略 $\mu(s)$。（你需要列出 3 个 $(s, (s))$ 数值对，两个吸收态不需要考虑）

2 问题 2：Q-Learning[12%]

(a) [3 分] 对于一个马尔可夫决策过程，我们定义一条轨迹 (trajectory) $\tau = (s_t, a_t, r_t, s_{t+1}, a_{t+1}, r_{t+1}, \dots)$ 的回报为 G_t。其表达式为：$G_t = \sum_{k=t}^{+\infty} \gamma^{k-t} \mathcal{R}_k$。我们根据 G_t 可以定义状态价值函数 $V(s) = \mathbb{E}[G_t | s_t = s]$，以及动作价值函数 $Q(s, a) = \mathbb{E}[G_t | s_t = s, a_t = a]$。
基于上面的定义，请推导出 $Q(s, a)$ 的表达形式，用状态价值函数 $V(s')$ 表示。

(b) [6 分] 将估计值的初始值设置为 0。对于定义在状态空间 $S = 0, 1$ 和动作空间 $A = a_1, a_2$ 上的某一轨迹（trajectory）$\tau = \{(s_i, a_i, r_i)\}_4 = \{(0, a_1, 2), (1, a_1, 3), (0, a_2, -1), (1, a_2, 0)\}$，请手动写出估计值 $q(s, a)$ 在时间 $1 \le t \le 4$ 的更新过程（Monte Carlo Q-Learning）。

(c) [3 分] 请简单解释 Q-Learning 算法为什么能够收敛。
（参考材料 https://zhuanlan.zhihu.com/p/365814943，理解并复述文中证明的大致思路，不需要复现其中的数学细节）

3 问题 3：Gobang Programming[53%]

你现在应当初步理解了强化学习的原理。现在，我们可以考虑更加现实的一个游戏——五子棋的简化版，“三子棋”。为方便起见，我们在这里将“三子棋”的规则统一为：

1. 棋盘大小：$n = 3$;
2. 先手规则：黑棋先下;
3. 终局判定：一旦某一方在**行**、**列**、**正反对角线任一方向**上达成连续的三个棋子，则这一方赢。否则，若直到棋盘被填满仍未分出胜负，则双方平局。

在本题中，假定我们要训练的智能体是黑棋棋手，而白棋是一个只会随机落子的小白（因此可以把白棋落子视为环境噪声）；那么我们也可以将其表述为一个规范的 MDP：

- 状态空间 S= 黑棋落子前所有棋盘可能的状态 s（某一位置 $s_{ij} = 0$ 表示无子，1 表示落黑子，2 表示落白子）
- 动作空间 $\mathcal{A} = \{(x, y) \mid s_{xy} = 0\}$;
- 初始状态 $s_0 = 0_{n \times n}$，终局判定达成时，游戏结束。
- 奖励函数 $R(s, a, s') = [L_b^2(s') - L_w^2(s')] - [L_b^2(s) - L_w^2(s)]$，其中 $L_b(s)$ 是状态 s 下黑棋的最长连续长度，$L_w(s)$ 是状态 s 下白棋的最长连线长度。
- 衰减因子 $\gamma = 0.5$。

(a) [33 分] 按照 `submission.py` 中的要求，完成代码填空。按照代码正确性和能否完整运行给分。在本实验中，你需要实现 Q-Learning 算法，训练智能体进行三子棋的游戏。为了减少你的工作量，助教已经实现了三子棋的框架、Q-Learning 的基本代码以及一些辅助代码。你只需要额外补全以下几个函数即可。同时这里再做一些简单的提示：

- `get_next_state` [5 分]，根据当前状态 `self.board` 和动作 `action`，返回下一个状态 `next_board`。你无须在这一部分对 `self.board` 进行修改;
- `sample_noise` [3 分]，根据当前的状态和动作空间，随机生成一个白棋的落子位置;
- `get_connection_and_reward` [5 分]，根据当前状态 `self.board` 和动作 `action`，返回下一个状态的奖励 `reward`，你可以使用 `get_next_state` 以及 `self.count_max_connections` 辅助实现 `get_connection_and_reward`;

- `sample_action_and_noise` [8 分]，根据当前状态 `self.board`，依据 epsilon-greedy 策略输出一个动作 `action`，同时返回一个白棋的落子位置。助教已经完成了调用 `sample_noise` 函数生成白棋的落子位置的返回逻辑，你需要补充实现根据 epsilon-greedy 策略返回 `action` 的部分。若状态-动作对 (s, a) 在 `self.Q` 中未被记录，则应返回一个随机动作。动作确定后，需从动作空间中移除该动作;
- `q_learning_update` [12 分]，你需要按照 Q-Learning 的更新公式来更新 `self.Q`。同时约定以下几个规则：1. 在数学上默认所有 `self.Q[s][a]` 的初始值为 0。另外，状态 s 对应的动作空间为空（即棋盘已满或者游戏已经结束的情况）导致无法在空集上取 max 时，直接以 0 代替即可；2. `self.Q` 被初始化为一个空字典，在后续中记录的 (s, a) Pair 也是有限的（如某一时刻，`self.Q` 的值可能是 $\{s_1 : \{a_1 : 0.5, a_2 : 1.0\}, s_2 : \{a_1 : 0.3\}, s_3 : \{\}\}$ 的形式，而 s_4 尚未被记录）。但是这样已经足以让我们实现正确的 Q-Learning 更新，请你想一想解决办法。

(b) [10 分] 请你运行 `learner.py` 进行 Q Learning，之后运行 `evaluator.py` 进行测试。你需要截取训练以及评估最后的截图，按照复现结果是否合理给分。（建议将评估轮次设置的稍大一些，这样会更准确）

(c) [10 分] 将 `learner.py` 和 `evaluator.py` 中的参数更改为 $n = 4$（其他任何参数和代码均不做改变），然后在 4*4 的棋盘上尝试执行 QLearning 算法，并给出 `evaluator.py` 的运行结果。回答该结果是否符合你的预期，并给出解释。

4 问题 4：Deeper Understanding[16%]

本节我们将从 Bellman 算子和重要性采样（Importance Sampling）进一步加深对强化学习的理解。

4.1 Bellman 算子与压缩映射

强化学习的评估和优化本质上与 Bellman 算子密切相关。

1. 评估：随机策略 $\pi(s, a) \in [0, 1]$ 的价值估计可以用算子 $\mathcal{T}_\pi$ 表示：

$$(\mathcal{T}_\pi v)(s) = \mathbb{E}_{a \in A}\left\{ r_{sa} + \gamma \cdot \sum_{s' \in S} p_{sas'} \cdot v(s') \right\} = \sum_{a \in A} \pi(s, a) \left[r_{sa} + \gamma \cdot \sum_{s' \in S} p_{sas'} \cdot v(s') \right]$$

2. 优化：例如 Value Iteration，实质上就是通过迭代求解最优策略，这个过程同样可以用算子 $\mathcal{T}$ 表示：

$$(\mathcal{T} v)(s) = \max_{a \in A}\left\{ r_{sa} + \gamma \cdot \sum_{s' \in S} p_{sas'} \cdot v(s') \right\}$$

3. 有如上定义之后，$v_\pi, v^* \in \mathbb{R}^{|S|}$ 实际上是对应 Bellman 方程的解：

$$\mathcal{T} v^* = v^*, \quad \mathcal{T}_\pi v_\pi = v_\pi$$

其中

$$v_\pi(s) = \mathbb{E}\left[G_t \mid S_t = s, A \sim \pi \right], \quad v^*(s) = \max_\pi \{ v_\pi(s) \}$$

请根据上面的内容回答下面的问题：

(a) [5 分] 考虑具有有限状态和动作空间的 MDP，以及折扣因子 γ。证明 $\mathcal{T}$ 是压缩映射，即对于 $v_1, v_2 \in \mathbb{R}^{|S|}$，我们恒有 $\|\mathcal{T}v_1 - \mathcal{T}v_2\| \leq \gamma\|v_1 - v_2\|_\infty$。（这表明 Bellman 算子是最大范数中的 “γ-收缩”）

(b) [5 分] 如果 $\mathcal{T}v = v$，则我们称 v 是 $\mathcal{T}$ 的不动点。利用 Bellman 算子是最大范数中的 γ 收缩这一事实，证明 $\mathcal{T}$ 最多有一个不动点，即 Bellman 方程最多有一个解。你可以假设 $\mathcal{T}$ 至少有一个不动点。
提示：在部分 (a) 中证明的结果意味着值迭代几何收敛到最优值函数 $v*$。也就是说，经过 k 次迭代后，v 与 $v*$ 之间的距离最多为 γ^k。

4.2 重要性采样

在异策略（Off-Policy）强化学习中，重要性采样（Importance Sampling）是一种关键方法，用于通过采样策略 π_a 生成的样本估计目标策略 π_b 的期望回报。基于重要性采样，OpenAI 提出了近端策略优化算法（proximal policy optimization，PPO）。在这一节，我们将完成重要性采样的推导。

(a) [2 分] 记 p, q 为关于 x 的两个分布，请证明 $\mathbb{E}_{x\sim p}[f(x)] = \mathbb{E}_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$。
（在强化学习的策略优化中，我们可以将 q 看作采样的策略，p 看作需要优化的目标策略。上式可以理解为：我们通过 q 获得的样本获得了对需要优化的目标函数的无偏估计）

(b) [4 分] 证明 $\mathrm{Var}_{x\sim p}[f(x)] - \mathrm{Var}_{x\sim q}[f(x)\frac{p(x)}{q(x)}] = \mathbb{E}_{x\sim p}[f(x)^2] - \mathbb{E}_{x\sim p}[f(x)^2\frac{p(x)}{q(x)}]$。
提示：$\mathrm{Var}[X] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$，请注意在期望符号右下角标明 x 来自何种分布。
（该结果表明虽然 $f(x)\frac{p(x)}{q(x)}, x \sim q$ 和 $f(x), x \sim p$ 的期望相同，但是他们的方差存在差距。二者的方差相差较大将导致模型训练的不稳定。因此在实践中，我们要求分布 p 与 q 不能相差过大，PPO 中引入了 KL-散度惩罚项来控制这一点，后续为了优化 KL-散度的计算，又出现了基于 clip 操作的近端策略优化裁剪算法）

作业反馈

体验反馈 [10%]

你可以写下任何反馈，包括但不限于以下几个方面：课堂、作业、助教工作等等。

(a) [**必做**] 你在本次作业花费的时间大概是?

(b) [**选做**] 你可以随心吐槽课程不足的方面，或者给出合理的建议。