南開大學
Nankai University

人工智能学院

强化学习实验报告

实验三　MDP 环境搭建

姓名：石若川

学号：2111381

专业：智能科学与技术

2024 年 3 月 17 日

1 实验目的

搭建 MDP 环境，给出状态空间、动作空间、回报函数、状态转移概率，为后续实验奠定基础。

2 实验原理

Gym 库是由 OpenAI 开发的一个用于开发强化学习算法的工具包。它提供了一个统一的接口，使得在不同环境中进行强化学习的研究和开发变得更加简单。

实验中，使用了 Gym 库（0.26.0 版本）中自带的 Taxi-v3 环境。该环境描述了一个出租车载客的问题。网格世界中有四个指定位置，用 R(ed)、G(reen)、Y(ellow) 和 B(lue) 表示。回合开始时，出租车从随机的广场位置出发，乘客也处于随机的位置。出租车行驶至乘客所在位置，接载乘客，行驶至乘客目的地（四个指定地点中的另一个地点），然后送乘客下车。一旦乘客下车，该回合就结束了。

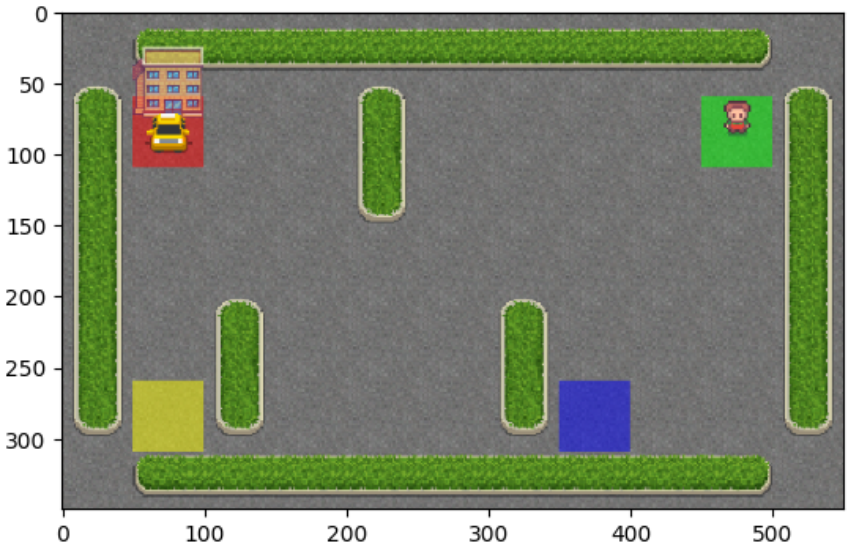

图 2.1: Taxi-v3 环境

- **状态空间：** Taxi-v3 环境的状态空间是离散的，由出租车的位置、乘客的位置、目的地的位置组成。其中出租车的可能位置有 25 个，乘客上车位置有 5 个（0: R(ed)，1: G(reen)，2: Y(ellow)，3: B(lue)，4: in taxi），乘客目的地有 4 个（0: R(ed)，1: G(reen)，2: Y(ellow)，3: B(lue)）。因此该环境的状态空间大小为 500。
- **动作空间：** Taxi-v3 环境的动作空间也是离散的，包含了出租车可以执行四个基本动作（0：向南移动，1：向北移动，2：向东移动，3：向西移动），以及乘客的两个动作（4：乘客上车，5：乘客下车）。因此，动作空间的大小为 6。
- **回报函数：**
  - 移动：-1, 表示每一步都会受到一点惩罚，以鼓励从出发地到目的地走最短的路。
  - 错误运送：-10，表示当乘客被送到到错误的位置时，任务，所以惩罚应该设置大一些。
  - 成功送达：20，表示出租车司机成功完成了任务，应鼓励相应的行为。

- **状态转移概率：** Taxi-v3 环境的状态转移是确定性的，因为环境的动态是完全可控的。也就是说，给定当前状态和采取的动作，环境将会根据相应的规则将出租车移动到下一个状态。具体规则包括：出租车按照动作移动，如果乘客在车内，则在下一个状态中仍然在车内；如果乘客在外面，则在下一个状态中仍然在外面。

3 实验代码

首先导入所需要的库。本实验中使用的 Gym 库为 0.26.0 版本，与 0.24.0 和 0.21.0 版本的函数调用方法有些许不同。

```
import gym
import matplotlib.pyplot as plt
```

使用 gym.make 函数创建环境，其中 mode=”rgb_array” 表示在使用 render 函数渲染环境时，返回当前一帧的 RGB 图像；mode=”rgb_array_list” 表示渲染环境时，返回一个 RGB 图像的列表，包括过去的所有帧；mode=”human” 表示渲染环境时，直接展示可视化界面。下列代码以 rgb_array 模式创建 Taxi-v3 环境，显示了初始化的环境图像，打印出动作空间和状态空间。

```
# 生成环境
env = gym.make('Taxi-v3', render_mode="rgb_array")
# 环境初始化
state = env.reset()
# 渲染环境
pic = env.render()
plt.imshow(pic)
plt.show()

print("Action Space {}".format(env.action_space))
print("State Space {}".format(env.observation_space))
```

另外，实验中还随机从动作空间中选取动作与环境进行交互，利用 “rgb_array_list” 模式记录了智能体与环境的交互过程，代码如下。

```
env = gym.make('Taxi-v3', render_mode="rgb_array_list")
env.reset()
frame = []
# 循环交互
for _ in range(50):
    # 从动作空间随机获取一个动作
    action = env.action_space.sample()

    # agent与环境进行一步交互
    state, reward, terminated, truncated, info = env.step(action)
    print('state = {0}; reward = {1}'.format(state, reward))
```

```
12
13      # 判断当前否完成
14      if terminated:
15          print('done')
16          break
17      # time.sleep(1)
18
19  frame.append(env.render())
20  # 环境结束
21  env.close()
```

接着可以利用 imageio 库将交互过程制作为 gif 动图，便于展示。

```
1  import imageio
2  def compose_gif(frame):
3      imageio.mimsave("gym_Taxi.gif", frames, duration=20)
4
5  compose_gif(frame)
```

4 实验结果

实验中打印的动作空间和状态空间大小如图4.2所示，动作空间大小为 6，状态空间大小为 500，与上文描述一致。

```
Action Space Discrete(6)
State Space Discrete(500)
```

图 4.2: 动作空间和状态空间大小

使用 encode 函数可以指定状态，例如指定 taxi row=3, taxi column=2, passenger index=2, destination index=0 时：

```
1  state = env.encode(3, 1, 2, 0) # (taxi row, taxi column, passenger index,
       destination index)
2  print("State:", state)
3  print(env.P[328])
```

打印结果为如下，State: 328 表示（3,1,2,0）对应的状态为 328。env.P 包括了在当前状态下采取每个动作对应的信息，其字典结构为 action: [(probability, next state, reward, done)]。probability 表示状态转移概率，在本环境下恒为 1；next state 表示采取动作 action 后转移至的状态；reward 表示采取动作 action 的奖励；done 表示回合是否结束。

```
1  State: 328
2  {0: [(1.0, 428, -1, False)], 1: [(1.0, 228, -1, False)],
```

```
2: [(1.0, 348, -1, False)], 3: [(1.0, 328, -1, False)],
4: [(1.0, 328, -10, False)], 5: [(1.0, 328, -10, False)]}
```

实验中利用随机策略循环 6 轮，如图4.3所示。另外，实验中将循环 50 轮的图像制作为动图，参见附件 gym_Taxi.gif。可以发现随机选取动作的情况下，智能体的运动没有目的性，不具备没有完成任务的能力。

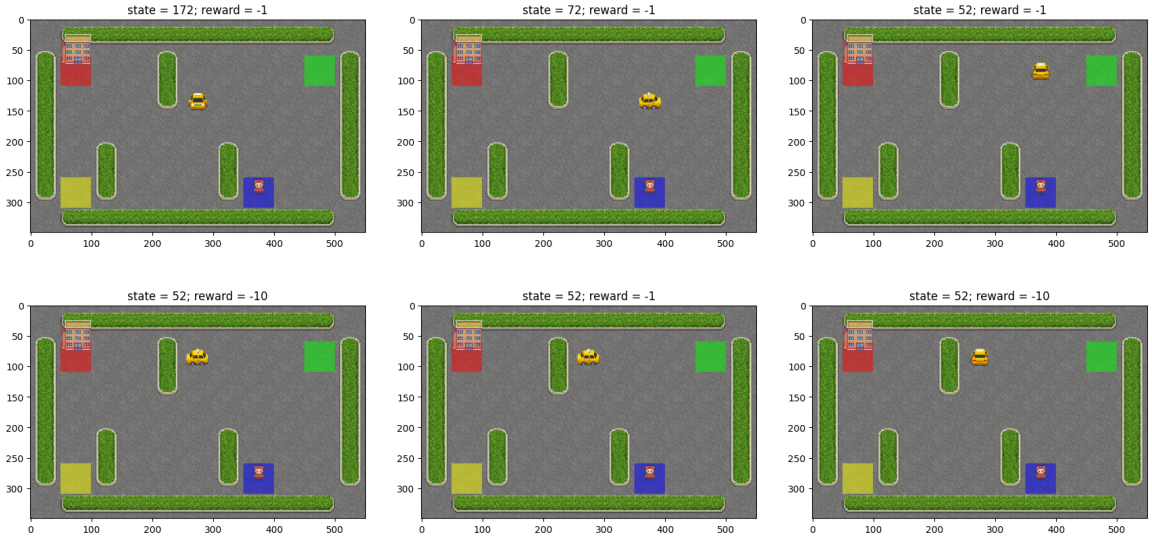

图 4.3: 随机策略过程

5 实验总结

通过本实验，我了解了 Gym 库的基本使用方法。利用 Taxi-v3，构建了符合 MDP 的环境，为后续实验奠定了基础。该环境的信息如下：

- **状态空间：** 共 500 个，其中出租车的可能位置有 25 个，乘客上车位置有 5 个（0: R(ed)，1: G(reen)，2: Y(ellow)，3: B(lue)，4: in taxi），乘客目的地有 4 个（0: R(ed)，1: G(reen)，2: Y(ellow)，3: B(lue)）。
- **动作空间：** 共 6 个，包含了出租车可以执行四个基本动作（0：向南移动，1：向北移动，2：向东移动，3：向西移动），以及乘客的两个动作（4：乘客上车，5：乘客下车）。
- **回报函数：**
  - 移动：-1。
  - 错误运送：-10。
  - 成功送达：20。
- **状态转移概率：** 对于一个状态而言，采取固定动作的下一个状态是确定的。