COMP9414 24T2
Artificial Intelligence

Assignment 2 - Reinforcement Learning

Due: Week 9, Wednesday, 24 July 2024, 11:55 PM.

1 Problem context

Taxi Navigation with Reinforcement Learning: In this assignment, you are asked to implement Q-learning and SARSA methods for a taxi navigation problem. To run your experiments and test your code, you should make use of the Gym library[1], an open-source Python library for developing and comparing reinforcement learning algorithms. You can install Gym on your computer simply by using the following command in your command prompt:

```
pip install gym
```

In the taxi navigation problem, there are four designated locations in the grid world indicated by R(ed), G(reen), Y(ellow), and B(lue). When the episode starts, one taxi starts off at a random square and the passenger is at a random location (one of the four specified locations). The taxi drives to the passenger's location, picks up the passenger, drives to the passenger's destination (another one of the four specified locations), and then drops off the passenger. Once the passenger is dropped off, the episode ends. To show the taxi grid world environment, you can use the following code:

[1] https://www.gymlibrary.dev/environments/toy_text/taxi/

```
env = gym.make("Taxi—v3", render_mode="ansi").env
state = env.reset()
rendered_env = env.render()
print(rendered_env)
```

In order to render the environment, there are three modes known as *"human"*, *"rgb_array*, and *"ansi"*. The *"human"* mode visualizes the environment in a way suitable for human viewing, and the output is a graphical window that displays the current state of the environment (see Fig. 1). The *"rgb_array"* mode provides the environment's state as an RGB image, and the output is a numpy array representing the RGB image of the environment. The *"ansi"* mode provides a text-based representation of the environment's state, and the output is a string that represents the current state of the environment using ASCII characters (see Fig. 2).

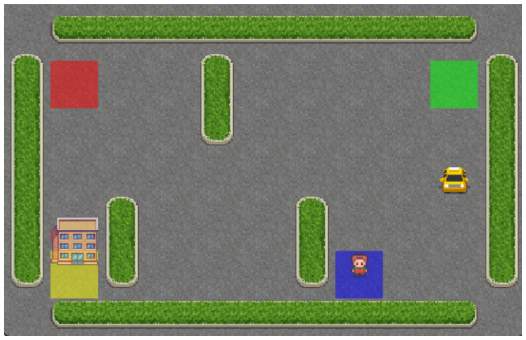
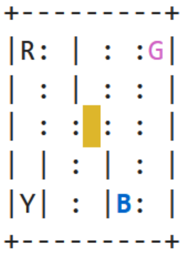


Figure 1: *"human"* mode presentation for the taxi navigation problem in Gym library.

You are free to choose the presentation mode between *"human"* and *"ansi"*, but for simplicity, we recommend *"ansi"* mode. Based on the given description, there are six discrete deterministic actions that are presented in Table 1.

For this assignment, you need to implement the Q-learning and SARSA algorithms for the taxi navigation environment. The main objective for this assignment is for the agent (taxi) to learn how to navigate the gird-world and drive the passenger with the minimum possible steps. To accomplish the learning task, you should empirically determine hyperparameters, e.g., the learning rate α, exploration parameters (such as ϵ or T), and discount factor γ for your algorithm. Your agent should be penalized -1 per step it



```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
```

Figure 2: *“ansi”* mode presentation for the taxi navigation problem in Gym library. Gold represents the taxi location, blue is the pickup location, and purple is the drop-off location.

Table 1: Six possible actions in the taxi navigation environment.

Action	**Number of the action**
Move South	0
Move North	1
Move East	2
Move West	3
Pickup Passenger	4
Drop off Passenger	5

takes, receive a +20 reward for delivering the passenger, and incur a -10 penalty for executing “pickup” and “drop-off” actions illegally. You should try different exploration parameters to find the best value for exploration and exploitation balance.

As an outcome, you should plot the accumulated reward per episode and the number of steps taken by the agent in each episode for **at least 1000** learning episodes for both the Q-learning and SARSA algorithms. Examples of these two plots are shown in Figures 3–6. Please note that the provided plots are just examples and, therefore, your plots will not be exactly like the provided ones, as the learning parameters will differ for your algorithm.

After training your algorithm, you should save your Q-values. Based on your saved Q-table, your algorithms will be tested on at least 100 random grid-world scenarios with the same characteristics as the taxi environment for both the Q-learning and SARSA algorithms using the greedy action selection

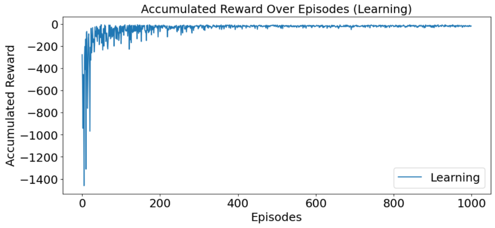


Figure 3: Q-learning reward.

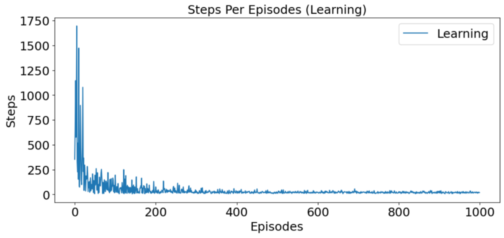


Figure 4: Q-learning steps.

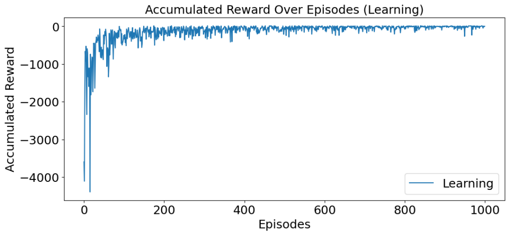


Figure 5: SARSA reward.

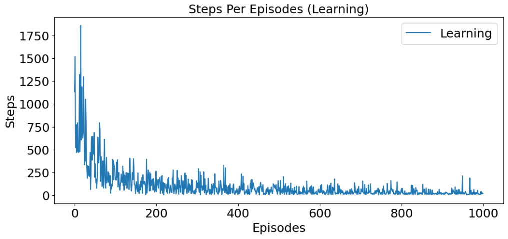


Figure 6: SARSA steps.

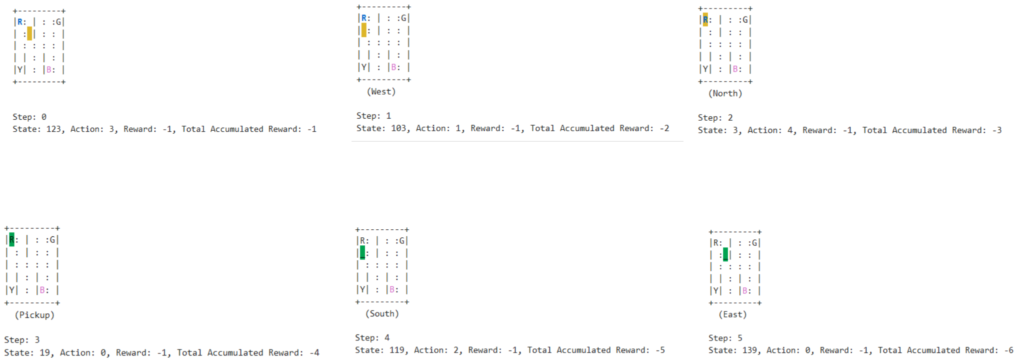
method. Therefore, your Q-table will not be updated during testing for the new steps.

Your code should be able to visualize the trained agent for both the Q-learning and SARSA algorithms. This means you should render the *"Taxi-v3"* environment (you can use the *"ansi"* mode) and run your trained agent from a random position. You should present the steps your agent is taking and how the reward changes from one state to another. An example of the visualized agent is shown in Fig. 7, where only the first six steps of the taxi are displayed.

2 Testing and discussing your code

As part of the assignment evaluation, your code will be tested by tutors along with you in a discussion carried out in the tutorial session in week 10. The assignment has a total of 25 marks. The discussion is mandatory and, therefore, we will not mark any assignment not discussed with tutors.

Before your discussion session, you should prepare the necessary code for this purpose by loading your Q-table and the *"Taxi-v3"* environment. You should be able to calculate the average number of steps per episode and the



```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+

Step: 0
State: 123, Action: 3, Reward: -1, Total Accumulated Reward: -1
```

```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
  (West)

Step: 1
State: 103, Action: 1, Reward: -1, Total Accumulated Reward: -2
```

```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
  (North)

Step: 2
State: 3, Action: 4, Reward: -1, Total Accumulated Reward: -3
```

```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
  (Pickup)

Step: 3
State: 19, Action: 0, Reward: -1, Total Accumulated Reward: -4
```

```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
  (South)

Step: 4
State: 119, Action: 2, Reward: -1, Total Accumulated Reward: -5
```

```
+---------+
|R: | : :G|
| : | : : |
| : : : : |
| | : | : |
|Y| : |B: |
+---------+
  (East)

Step: 5
State: 139, Action: 0, Reward: -1, Total Accumulated Reward: -6
```

Figure 7: The first six steps of a trained agent (taxi) based on Q-learning algorithm.

average accumulated reward (for a maximum of 100 steps for each episode) for the test episodes (using the greedy action selection method).

You are expected to propose and build your algorithms for the taxi navigation task. You will receive marks for each of these subsections as shown in Table 2. Except for what has been mentioned in the previous section, it is fine if you want to include any other outcome to highlight particular aspects when testing and discussing your code with your tutor.

For both Q-learning and SARSA algorithms, your tutor will consider the average accumulated reward and the average taken steps for the test episodes in the environment for a maximum of 100 steps for each episode. For your Q-learning algorithm, the agent should perform at most 14 steps per episode on average and obtain a minimum of 7 average accumulated reward. Numbers worse than that will result in a score of 0 marks for that specific section. For your SARSA algorithm, the agent should perform at most 15 steps per episode on average and obtain a minimum of 5 average accumulated reward. Numbers worse than that will result in a score of 0 marks for that specific section.

Finally, you will receive 1 mark for code readability for each task, and your tutor will also give you a maximum of 5 marks for each task depending on the level of code understanding as follows: **5. Outstanding, 4. Great, 3. Fair, 2. Low, 1. Deficient, 0. No answer**.

Table 2: Marks for each task.

Task	**Marks**
Results obtained from agent learning	
Accumulated rewards and steps per episode plots for Q-learning algorithm.	2 marks
Accumulated rewards and steps per episode plots for SARSA algorithm.	2 marks
Results obtained from testing the trained agent	
Average accumulated rewards and average steps per episode for Q-learning algorithm.	2.5 marks
Average accumulated rewards and average steps per episode for SARSA algorithm.	2.5 marks
Visualizing the trained agent for Q-learning algorithm.	2 marks
Visualizing the trained agent for SARSA algorithm.	2 marks
Code understanding and discussion	
Code readability for Q-learning algorithm	1 mark
Code readability for SARSA algorithm	1 mark
Code understanding and discussion for Q-learning algorithm	5 mark
Code understanding and discussion for SARSA algorithm	5 mark
Total marks	25 marks

3 Submitting your assignment

The assignment must be done individually. You must submit your assignment solution by Moodle. This will consist of a single .zip file, including three files, the .ipynb Jupyter code, and your saved Q-tables for Q-learning and SARSA (you can choose the format for the Q-tables). Remember your files with your Q-tables will be called during your discussion session to run the test episodes. Therefore, you should also provide a script in your Python code at submission to perform these tests. Additionally, your code should include short text descriptions to help markers better understand your code. Please be mindful that providing clean and easy-to-read code is a part of your assignment.

Please indicate your full name and your zID at the top of the file as a comment. You can submit as many times as you like before the deadline – later submissions overwrite earlier ones. After submitting your file a good practice is to take a screenshot of it for future reference.

Late submission penalty: UNSW has a standard late submission penalty of 5% per day from your mark, capped at five days from the assessment deadline, after that students cannot submit the assignment.

4 Deadline and questions

Deadline: Week 9, Wednesday 24 of July 2024, 11:55pm. Please use the forum on Moodle to ask questions related to the project. We will prioritise questions asked in the forum. However, you should not share your code to avoid making it public and possible plagiarism. If that's the case, use the course email `cs9414@cse.unsw.edu.au` as alternative.

Although we try to answer questions as quickly as possible, we might take up to 1 or 2 business days to reply, therefore, last-moment questions might not be answered timely.

For any questions regarding the discussion sessions, please contact directly your tutor. You can have access to your tutor email address through Table 3.

5 Plagiarism policy

Your program must be entirely your own work. Plagiarism detection software might be used to compare submissions pairwise (including submissions for any similar projects from previous years) and serious penalties will be applied, particularly in the case of repeat offences.

Do not copy from others. Do not allow anyone to see your code. Please refer to the UNSW Policy on Academic Honesty and Plagiarism if you require further clarification on this matter.

Table 3: COMP9414 24T2 Tutorials

Number	**Class ID**	**Time**	**Tutor**	**Email**
1	4210	Fri 12:00 - 14:00	Siti Mariyah	s.mariyah@unsw.edu.au
2	4211	Fri 12:00 - 14:00	Malhar Patel	malhar.patel@unsw.edu.au
3	4212	Fri 14:00 - 16:00	Stefano Mezza	s.mezza@unsw.edu.au
4	4213	Fri 14:00 - 16:00	Janhavi Jain	j.jain@student.unsw.edu.au
5	4214	Fri 14:00 - 16:00	Adam Stucci	a.stucci@unsw.edu.au
6	4215	Fri 16:00 - 18:00	Janhavi Jain	j.jain@student.unsw.edu.au
7	4216	Fri 18:00 - 20:00	Jingying Gao	jingying.gao@unsw.edu.au
8	4217	Thu 14:00 - 16:00	Shengyuan Xie	shengyuan.xie@student.unsw.edu.au
9	4218	Thu 14:00 - 16:00	Adam Stucci	a.stucci@unsw.edu.au
10	4219	Thu 14:00 - 16:00	Malhar Patel	malhar.patel@unsw.edu.au
11	4220	Thu 16:00 - 18:00	Siti Mariyah	s.mariyah@unsw.edu.au
12	4221	Thu 18:00 - 20:00	Jingying Gao	jingying.gao@unsw.edu.au
13	4223	Tue 09:00 - 11:00	Zahra Donyavi	z.donyavi@unsw.edu.au
14	4224	Tue 12:00 - 14:00	Maher Mesto	m.mesto@unsw.edu.au
15	4225	Tue 12:00 - 14:00	Raktim Kumar Mondol	r.mondol@unsw.edu.au
16	4226	Tue 12:00 - 14:00	Stefano Mezza	s.mezza@unsw.edu.au
17	4227	Tue 16:00 - 18:00	Shengyuan Xie	shengyuan.xie@student.unsw.edu.au
18	4228	Tue 16:00 - 18:00	Zahra Donyavi	z.donyavi@unsw.edu.au
19	4229	Tue 16:00 - 18:00	Raktim Kumar Mondol	r.mondol@unsw.edu.au
20	4230	Tue 16:00 - 18:00	Aayush Gupta	aayush.gupta@unsw.edu.au
21	4231	Tue 18:00 - 20:00	Aayush Gupta	aayush.gupta@unsw.edu.au
22	4232	Wed 09:00 - 11:00	Kiran Jeet Kaur	kiran_jeet.kaur@student.unsw.edu.au
23	4233	Wed 13:00 - 15:00	Stefano Mezza	s.mezza@unsw.edu.au
24	4234	Wed 13:00 - 15:00	Kiran Jeet Kaur	kiran_jeet.kaur@student.unsw.edu.au
25	12564	Wed 12:00 - 14:00	Lina Phaijit	l.phaijit@unsw.edu.au
26	12565	Tue 16:00 - 18:00	Zhijin Meng	zhijin.meng@student.unsw.edu.au
27	12696	Thu 18:00 - 20:00	Ramya Kumar	ramya.kumar1@student.unsw.edu.au
28	12695	Tue 18:00 - 20:00	Maher Mesto	m.mesto@unsw.edu.au
29	12693	Wed 18:00 - 20:00	Zhijin Meng	zhijin.meng@student.unsw.edu.au
30	12694	Wed 18:00 - 20:00	Ramya Kumar	ramya.kumar1@student.unsw.edu.au