22417.202210 Homework 5 - Reinforcement Learning

Pankaj Kumar Jatav

TOTAL POINTS

22 / 24

QUESTION 1

Value iteration 20 pts

1.1 q1 **6 / 6**

✓ **- 0 pts** *Correct*

1.2 q2 **2 / 2**

✓ **- 0 pts** *Correct*

- 2 pts Incorrect

1.3 q3 **6 / 6**

✓ **- 0 pts** *Correct*

- 0.5 pts Slightly off

1.4 q4 **4 / 4**

✓ **- 0 pts** *Correct*

- 1 pts one or two incorrect direction

- 2 pts three or many incorrect direction

1.5 q5 **2 / 2**

✓ **- 0 pts** *Correct*

- 2 pts Incorrect

QUESTION 2

Extra credits: Q-Learning 2 pts

2.1 q1 **0.25 / 0.25**

✓ **- 0 pts** *Correct*

- 0.25 pts Blank

- 0.05 pts slightly off

2.2 q2 **0.25 / 0.25**

✓ **- 0 pts** *Correct*

- 0.25 pts Blank

- 0.05 pts slightly off

2.3 q3 **0.5 / 0.5**

✓ **- 0 pts** *Correct*

- 0.5 pts Blank

- 0.1 pts slightly off

- 0.4 pts Very wrong but good idea

2.4 q4 **0.5 / 0.5**

✓ **- 0 pts** *Correct*

- 0.5 pts Blank

- 0.3 pts Wrong values

2.5 q5 **0.5 / 0.5**

✓ **- 0 pts** *Correct*

- 0.25 pts Half Correct

- 0.35 pts Wrong numbers

- 0.5 pts Blank

QUESTION 3

3 Extra credits: Q-Learning Implementation **0 / 2**

✓ **- 2 pts** *Blank*

QUESTION 4

4 Collaboration Questions **0 / 0**

✓ **- 0 pts** *Correct*

HOMEWORK 5
REINFORCEMENT LEARNING[1]

CS 688 MACHINE LEARNING (SPRING 2022)
https://nlp.cs.gmu.edu/course/cs688-spring22/

OUT: April 14, 2022
DUE: April 28, 2022

Your name: Pankaj KumarJatav

Your GID: 01338769

[1] Compiled on Sunday 1^{st} May, 2022 at 03:55

1 Written Questions [20 pts]

1.1 Value Iteration

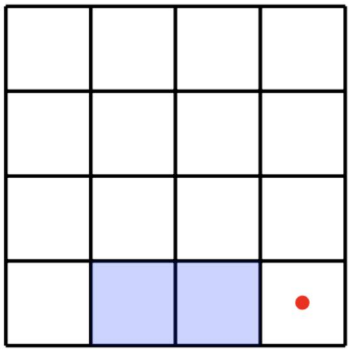


Figure 1.1: The cliff-walking environment.

In class we saw this 4x4 cliff-walking grid environment. The reward for reaaching all cells is 0, except for the cell with the red dot, which has a reward of 1, and the shared cells, which have a reward of -1. The episode ends if the agent lands in either the sahded cells or the red-dot cell. The state space ($\mathcal{S}$) is simply all the cells. The action space (A) is up, down, left, and right. Our transition function is deterministic. If the agent tries to move out of the grid, it simply goes back to its previous state. The discount factor γ is 0.9.

The **Bellman optimality equation** for state value function is:

$$V^*(s) = \max_{\alpha} \sum_{s' \in \mathcal{S}} p(s'|s,\alpha)(R(s,\alpha,s') + \gamma V^*(s'))$$

1. (6 points) We can numerically approximate V^* using synchronous value iteration. That is, we use the recurrence relation defined as follows: $\forall s \in \mathcal{S}$

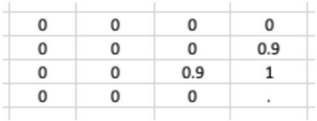
$$\begin{aligned} V_0(s) &= 0 \\ V_{k+1}(s) &= \max_{\alpha} \sum_{s' \in \mathcal{S}} p(s'|s,\alpha)(R(s,\alpha,s') + \gamma V^*(s')) \end{aligned}$$

Using value iteration, find the updated value of each cell for iterations 2 to 4. For example, for $k = 1$ we obtain the following:

0	0	0	0
0	0	0	0
0	0	0	1
0	0	0	0

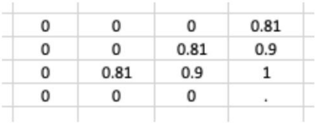
Now fill in the blank cells for each of the ones below:

$k = 2$



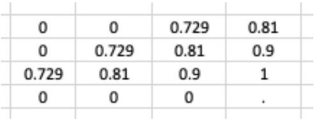
0	0	0	0
0	0	0	0.9
0	0	0.9	1
0	0	0	.

$k = 3$



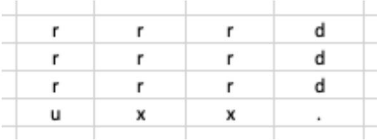
0	0	0	0.81
0	0	0.81	0.9
0	0.81	0.9	1
0	0	0	.

$k = 4$



0	0	0.729	0.81
0	0.729	0.81	0.9
0.729	0.81	0.9	1
0	0	0	.

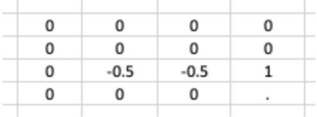
2. (2 points) What is an optimal policy after you run value iteration to convergence? (fill in the blank for each cell using arrows or writing `u, d, l, r` for up, down, left, and right respectively.



r	r	r	d
r	r	r	d
r	r	r	d
u	x	x	.

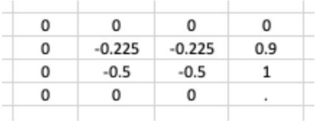
3. (6 points) Now suppose that the environment is sloped downward towards the cliff, with all the other settings unchanged. For every action taken, there is a 0.5 probability that the agent will move as intended and a 0.5 probability that the agent will slip and move 1 cell down instead. Fill in the empty cells for each of the $k = 1, 2, 3$:

$k = 1$



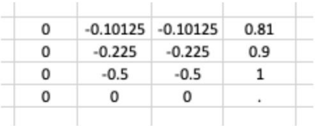
0	0	0	0
0	0	0	0
0	-0.5	-0.5	1
0	0	0	.

$k = 2$



0	0	0	0
0	-0.225	-0.225	0.9
0	-0.5	-0.5	1
0	0	0	.

$k = 3$

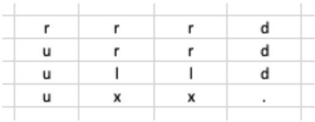


0	-0.10125	-0.10125	0.81
0	-0.225	-0.225	0.9
0	-0.5	-0.5	1
0	0	0	.

4. (4 points) Below is the result after 100 iterations (you can confirm if you have implemented the above).

0.14	0.22	0.54	0.81
0.09	-0.03	0.38	0.9
0.07	-0.47	-0.05	1
0.07	▨	▨	

What is the optimal policy now? (fill in the blanks as above)



r	r	r	d
u	r	r	d
u	l	l	d
u	x	x	.

5. (2 points) How could one change the environment setting to result in the optimal policy below?

→	→	→	→
→	→	→	→
↑	↑	↑	→
←	▨	▨	

Select one:

- ○ The agent receives a reward of -1 when it reaches any cell except the red-dot cell.
- ○ The agent receives a reward of 1 when it reaches any non-shaded cell.
- ○ The discount factor is now 0.1 instead of 0.9.
- ● Instead of slipping downwards, the agent now has a 0.5 probability of slipping to the right side of its intended direction.

2 [Extra Credit: 2 points (overall)] Q-Learning

1. In this question, we will practice using the Q-learning algorithm to play a simple two-player board game called "Connect 3". Each player, either a blue circle 🔵 or a red circle 🔴, takes turns marking a location in a 4x4 grid. Each time, a player has to either start from the bottom of the grid (if it is empty), or place symbols above locations with markers already placed (i.e., you can only stack the circles vertically starting from the bottom). The player who first succeeds in placing three of their marks in a column, a row, or a diagonal wins the game.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Table 2.1: Connect 3 Board Positions

We will model the game as follows: each board location corresponds to an integer between 1 and 16, illustrated in the graph above. Actions are also represented by an integer between 1 and 16. Playing action a results in marking the location a and an action a is only valid if the location a has not been marked by any of the players and there are no empty positions below a. We train the model by playing against an expert. The agent only receives a possibly nonzero reward when the game ends. Note a game ends when a player wins or when every location in the grid has been occupied. The reward is +1 if it wins, -1 if it loses and 0 if the game draws.

		🔴	
		🔵	
🔴	🔵	🔵	🔴

Table 2.2: State 1 (blue circle's turn)

To further simplify the question, let's say we are the blue circle player and it's our turn. Our goal is to try to learn the best end-game strategy given the current state of the game illustrated in table 2.2. The possible actions we can take are: $\{3, 9, 10, 12\}$. If we select action 3, 9, or 12, the expert will select action 10 to end the game and we'll receive a reward of -1; if we select action 10, the expert will respond by selecting action 9, which results in the state of the game in table 2.3. In the scenario in table 2.3, we can select actions $\{3, 5, 6, 12\}$. If we select actions 5, 6, or 12, then we end the game and receive a reward of +1; if we select action 3, then the expert will select action 5 to end the game and we'll receive a reward of -1.

		🔴	
🔴	🔵	🔵	
🔴	🔵	🔵	🔴

Table 2.3: State 2 (blue circle's turn)

Suppose we apply a learning rate $\alpha = 0.01$ and discount factor $\gamma = 1$. The Q-values are initialized as:

$Q(1,3) = -0.4$	$Q(1,9) = -0.3$	$Q(1,10) = 0.5$	$Q(1,12) = -0.3$
$Q(2,3) = -0.4$	$Q(2,5) = 0.5$	$Q(2,6) = 0.3$	$Q(2,12) = 0.3$

Note: Showing your work in these questions is optional, but it is recommended to help us understand where any misconceptions may occur. Only your answer in the left box will be graded.

(a) (1 point) In the first episode, the agent takes action 3, receives -1 reward, and the episode terminates. Derive the updated Q-value after this episode. Remember that given the sampled experience (s, a, r, s') of (state, action, reward, next state), the update of the Q value is:

$$Q(s,a) = Q(s,a) + \alpha \left(r + \gamma \max_{a' \in A} Q(s', a') - Q(s,a) \right) \tag{2.1}$$

Note if s' is the terminal state, $Q(s', a') = 0$ for all a'. **Please round to three decimal places**.

Q(1, 3)

-0.406

Work

(b) (1 point) In the second episode, the agent takes action 10, receives a reward of 0, and arrives at State 2 (2.3). It then takes action 3, receives a reward of -1, and the episode terminates. Derive the updated Q-values after each of the two experiences in this episode. Suppose we update the corresponding Q-value right after every single step. **Please round to three decimal places**.

Q(1, 10)

0.5

Q(2, 3)

-0.406

Work

(c) (2 points) In the third episode, the agent takes action 10, receives a reward of 0, and arrives at State 2 (2.3). It then takes action 5, receives a reward of +1, and the episode terminates. Derive the updated Q-values after each of the two experiences in this episode. Suppose we update the corresponding Q-value right after every single step. **Please round to three decimal places**.

Q(1, 10)	Q(2, 5)
0.5	0.505

Work

(d) (2 points) If we run the three episodes in cycle forever, what will be the final values of the following four Q-values. **Please round to three decimal places**.

Q(1, 3)	Q(1, 10)	Q(2, 3)	Q(2, 5)
-1	+1	-1	+1

Work

(e) (2 points) What will happen if the agent adopts the greedy policy (always pick the action that has the highest current Q-value) during training? Calculate the final four Q-values in this case. **Please round to three decimal places**.

Q(1, 3)	Q(1, 10)	Q(2, 3)	Q(2, 5)
-0.4	1	-0.4	1

Work

3 [Extra Credit: 2 point (overall)] Implement Q-Learning and the Connect-3 environment

Implement the above "Connect-3" environment in Python, along with Q-learning. Create visualizations that show the episodes as they get played.

For even additional credit (tbd), consider making the environment customizable, e.g.:

- the size of the board (e.g. we might want to play on a 6x6 board;
- the required connected circles e.g. we might want to play "Connect-6" on a 8x8 board;
- the number of agents (i.e. we might want to have 3 players play "Connect-4" on a 7x7 board.

Share a github repository (with appropriate documentation and/or python notebooks) with the advisor (Github ID: `antonisa`) to claim this extra credit.

4 Collaboration Questions

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found in the syllabus.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details.
2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details.
3. Did you find or come across code that implements any part of this assignment? If so, include full details.

Your Answer