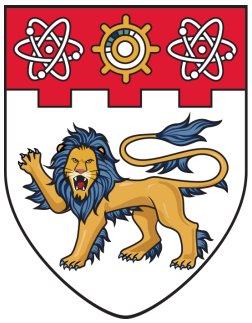

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

ASSIGNMENT REPORT FOR

AI6101: Introduction to AI and AI Ethics

Reinforcement Learning Assignment

ZENG ZHENG

G211941J

zzeng006@e.ntu.edu.sg

School of Computer Science and Engineering

March 10, 2022

1 Q-learning Algorithm

Q-learning algorithm is chosen to train my agent for the BoxPushing game. In this algorithm, after we observe the reward we obtained from taking the action from the previous state, we can then update the Q-value for that state-action pair in the Q-table. And the fomula of Q-learning algorithm is as Equation 1.

$$Q_{new}(S_t, A_t) = (1 - \alpha)Q_{old}(S_t, A_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a)) \tag{1}$$

where $Q_{new}(S_t, A_t)$ is the new estimation, $Q_{old}(S_t, A_t)$ is the old estimation, α is the learning rate, and γ is discount factor for the future reward.

According to this formula, the code snippet of updating the Q-value is as follows:

```
def train(self, state, action, next_state, reward):
action -= 1
self.Q[state][action] = self.Q[state][action] * (1 - self.alpha) + \
        self.alpha * (reward + self.discount_factor * np.max(self.Q[next_state]))
```

2 Learning Progress

In this experiment, I trained 10000 episodes for the agent. And the learning progress is as Figure 1. After all the episodes are finished, I calculated the average reward per hundred episodes from our list that contains the rewards for all episodes so that I can plot it and see how the rewards changed over time(The right subfigure). The original data is also plotted(The left subfigure).

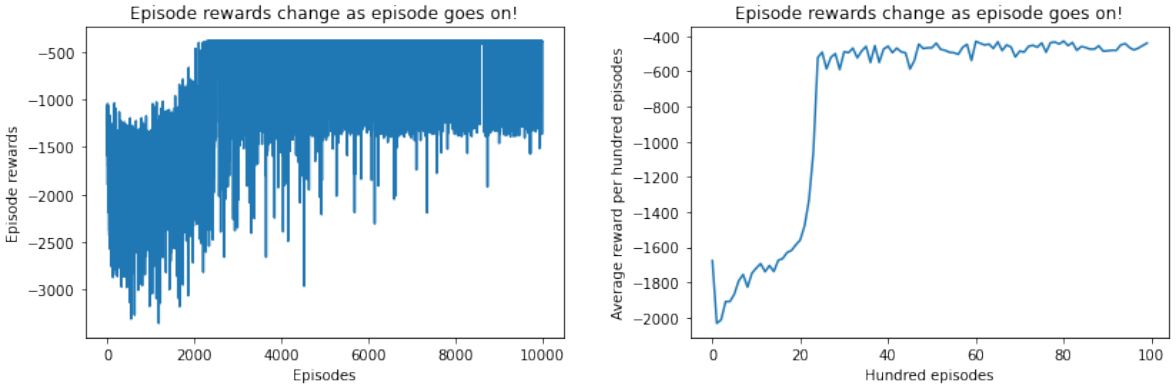

Figure 1: The learning progress. **Left**: Episode rewards vs. episodes. **Right**: Average reward per hundred episodes vs. Hundred episodes.

From the subplot on the right, we can see that our average reward per hundred episodes does increase over time. When the algorithm first starts training, the average reward for the first hundred episodes is only about **-1800**, but after a few hundred episodes (after about 3000 episodes), the reward increases dramatically to nearly **-450**.

3 Experiment Result

The final value table is partially demonstrated as Figure 2 since the complete table is too big. And the full table **V table.csv** has been submitted with the report.

((5, 0), (4, 1))	[-347.58889981 -347.34671179 -347.34771868 -335.70476858]
((4, 0), (4, 1))	[-342.2852387 -341.96992006 -341.44560201 -337.28875241]
((3, 0), (4, 1))	[-339.29083391 -339.10871356 -337.73817179 -340.58681929]
((2, 0), (4, 1))	[-338.09840304 -339.29045633 -335.40990649 -336.528052]
((1, 0), (4, 1))	[-336.09111677 -338.90895021 -336.47070145 -332.74980337]
((0, 0), (4, 1))	[-334.38866325 -336.66093315 -334.38866325 -329.17384991]
((5, 1), (4, 1))	[-324.9543117 -335.70468687 -347.34768248 -347.34761232]
((4, 1), (3, 1))	[-313.08516333 -337.68129061 -337.69912673 -337.40748438]
((3, 1), (2, 1))	[-300.08602357 -326.85470848 -326.94610401 -326.95427761]
((2, 1), (1, 1))	[-300.08602312 -315.0419896 -285.94547835 -314.99467695]
((3, 1), (1, 1))	[-300.08173875 -305.18655875 -300.85481816 -304.70208713]
((2, 0), (1, 1))	[-270.65199833 -302.08601882 -285.94547603 -300.08602235]
((1, 0), (1, 1))	[-285.93501544 -285.94542538 -270.65191203 -256.21413973]

Figure 2: Partial data presentation of the final value table.

Taking the first row as an example, the agent is in the state ((5,0),(4,1)), which means that the agent is in position (5,0) and the box is in position (4,1), according to the table of values we have the expected values of the actions when the agent is in this state. The third action will be chosen because it has the largest value, which means that the agent should go to the left.

In addition, in order to visualize the value table, I used the appearance style code written by Li[1]. I then implemented code to visualize the grid world during testing and labeled each grid with the corresponding strategy (arrows are used to indicate directions). The environment is a 2D grid world with 5 rooms, and the size of the environment is 6×14. **A** indicates the agent, **B** stands for the box, **G** is the goal, and black blocks mean cliff.

Some random screenshots during a test(one episode) are as shown in Figure 3, 4, and 5, which uses the Q-table trained after 10,000 episodes. And a video recording of the entire process of this test can be found here [2].

[1]Visualisation Style from `https://github.com/AccSrd/NTU_MSAI_AI6101_Introdution/blob/main/Assignment_ReinforcementLearning/code/Q_learing.py`

[2]Experiment result: value table visualisation of a test: `https://photos.app.goo.gl/aMx1FkDJSJWTFLod9`

	0	1	2	3	4	5	6	7	8	9	1 0	1 1	1 2	1 3
0	→	↓	→	→	↑	↓	▓▓	→	↑	▓▓	↑	↑	←	↑
1	←	→	←	←	↓	↓	▓▓	↑	↑	▓▓	→	↓	→	↓
2	→	B	↓	▓▓	→	↓	▓▓	→	←	▓▓	→	←	▓▓	→
3	→	A	←	▓▓	↓	↓	▓▓	→	↑	→	↑	↓	▓▓	↓
4	↑	↑	←	▓▓	↓	←	↓	↓	↑	▓▓	↓	↓	▓▓	G
5	↑	↑	←	▓▓	↑	←	→	↓	↑	▓▓	↓	↓	▓▓	↓

Agent Action Now: ↑

Figure 3: The agent and the box are at positions (3,1) and (2,1), respectively.

	0	1	2	3	4	5	6	7	8	9	1 0	1 1	1 2	1 3
0	↓	→	→	↓	→	←	▓▓	↑	←	▓▓	↑	←	→	↓
1	↑	↑	↓	↑	←	↓	▓▓	↑	←	▓▓	↑	↑	←	↓
2	←	→	←	▓▓	↓	↓	▓▓	→	↑	▓▓	↓	↑	▓▓	→
3	↓	→	↑	▓▓	A	←	▓▓	↓	↓	←	↑	↓	▓▓	↓
4	←	←	↑	▓▓	→	B	↓	→	↓	▓▓	↓	↓	▓▓	G
5	↓	↑	↑	▓▓	↑	→	→	↑	←	▓▓	↑	←	▓▓	↓

Agent Action Now: ↓

Figure 4: The agent and the box are at positions (3,4) and (4,5), respectively.

	0	1	2	3	4	5	6	7	8	9	1 0	1 1	1 2	1 3
0						←	▓▓	↑	←	▓▓	↓	→	↓	↓
1					↑	←	▓▓	→	←	▓▓	A	B	←	↑
2				▓▓	↑	↑	▓▓	↓	←	▓▓	↑	←	▓▓	→
3				▓▓	→	←	▓▓	→	↓	←	↑	←	▓▓	→
4				▓▓	→	↓	←	↑	↓	▓▓	↑	←	▓▓	G
5				▓▓	→	↓	→	↓	↑	▓▓	↓	↑	▓▓	→

Agent Action Now: →

Figure 5: The agent and the box are at positions (1,10) and (1,11), respectively.