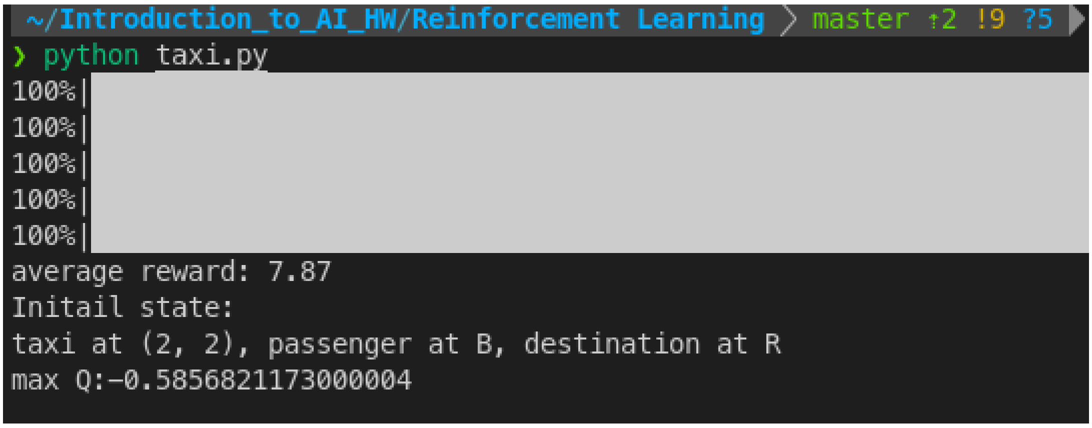
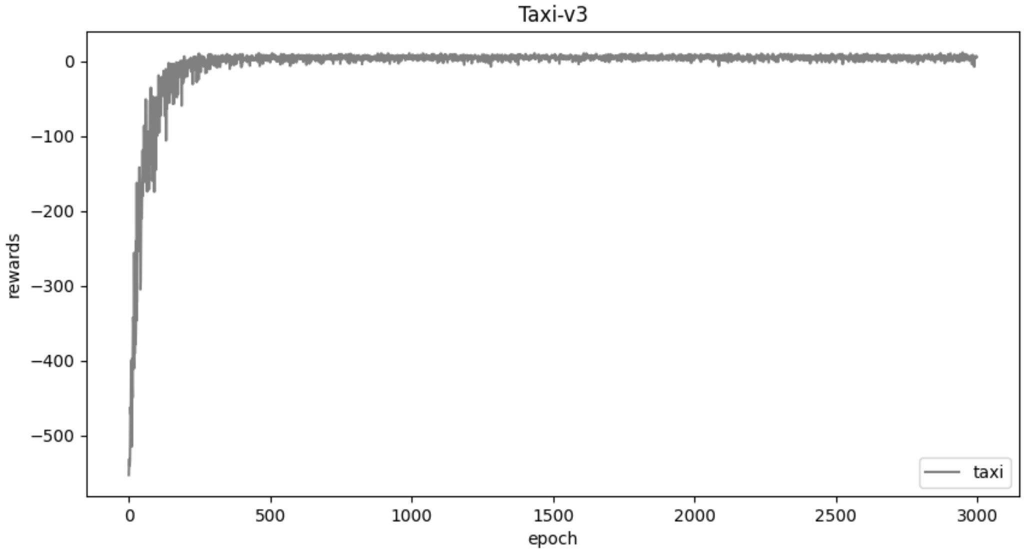
Homework 4: Reinforcement Learning Report Template

Part I. Experiment Results

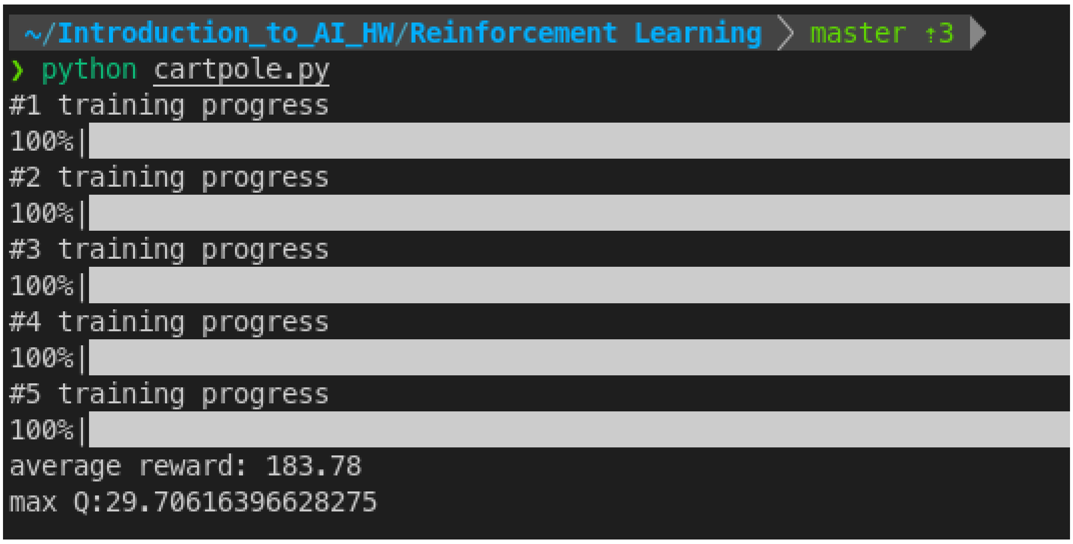
Taxi



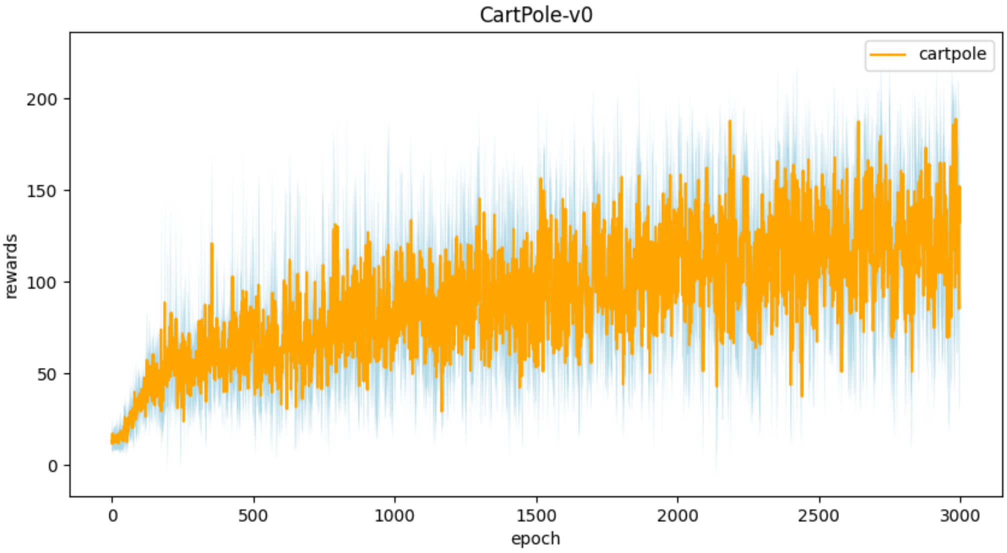
```
~/Introduction_to_AI_HW/Reinforcement Learning  master ↑2 !9 ?5
❯ python taxi.py
100%|
100%|
100%|
100%|
100%|
average reward: 7.87
Initail state:
taxi at (2, 2), passenger at B, destination at R
max Q:-0.5856821173000004
```



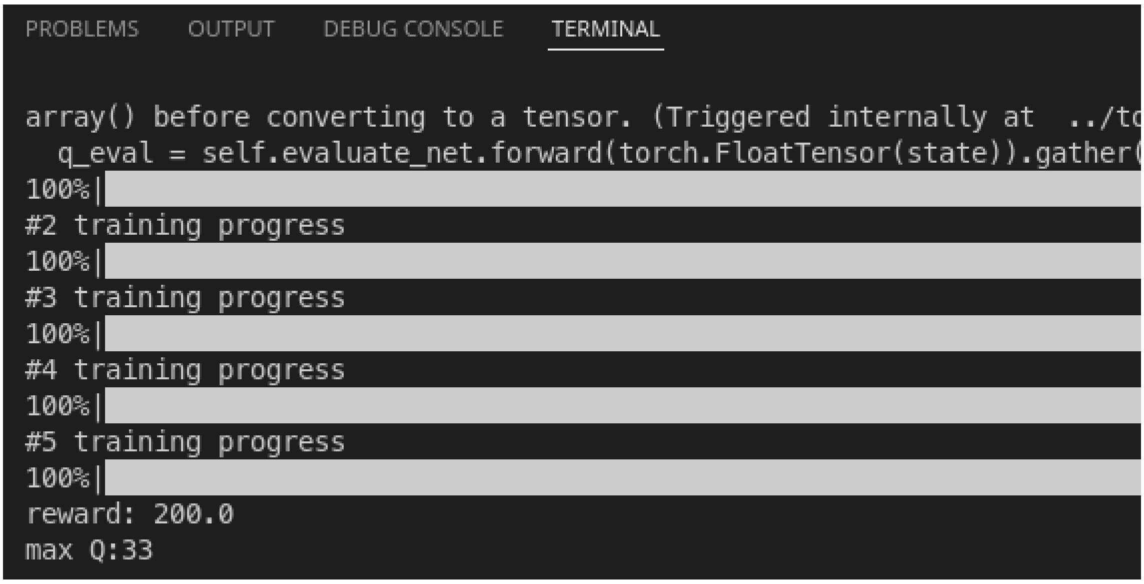
Cartpole



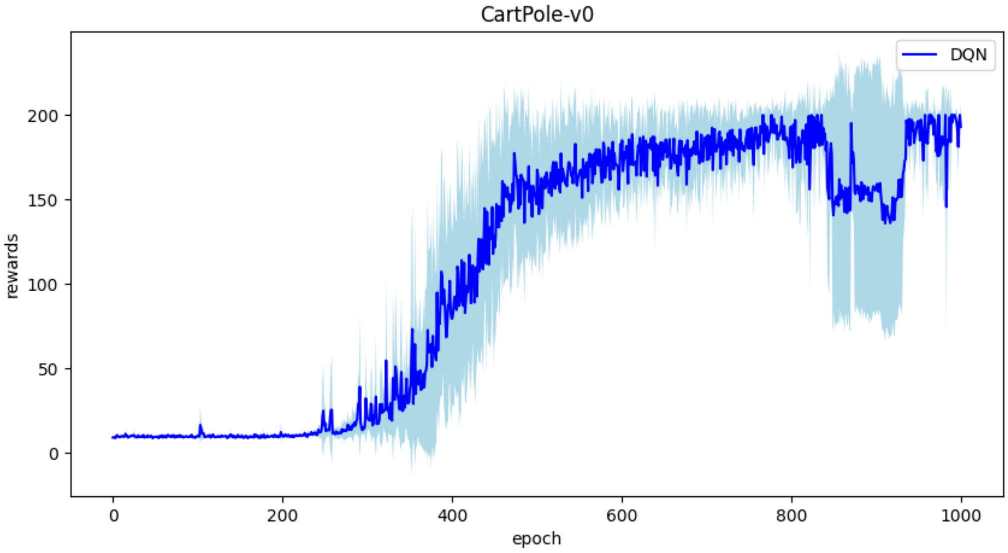
```
~/Introduction_to_AI_HW/Reinforcement Learning  master ↑3
❯ python cartpole.py
#1 training progress
100%|
#2 training progress
100%|
#3 training progress
100%|
#4 training progress
100%|
#5 training progress
100%|
average reward: 183.78
max Q:29.70616396628275
```



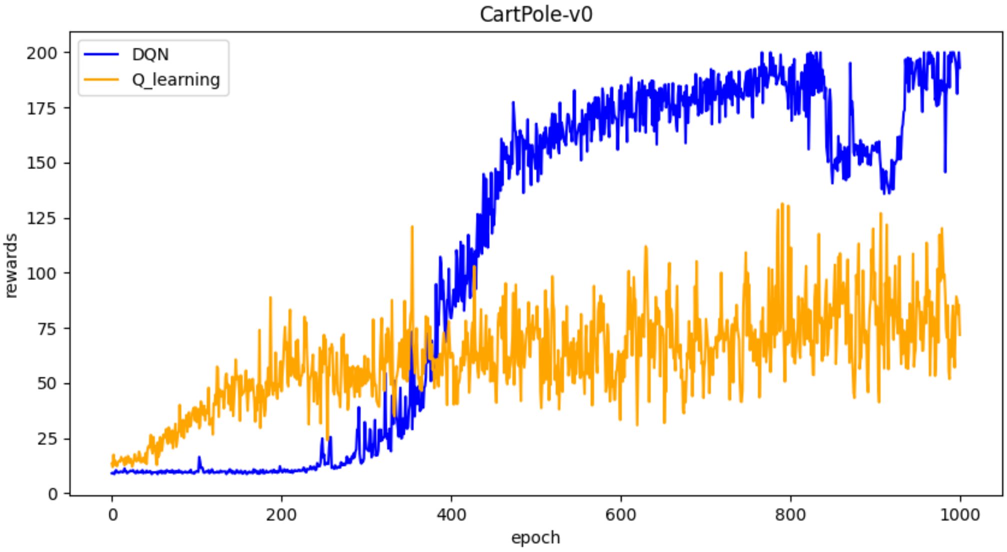
DQN



```
array() before converting to a tensor. (Triggered internally at  ../to
  q_eval = self.evaluate_net.forward(torch.FloatTensor(state)).gather(
100%|
#2 training progress
100%|
#3 training progress
100%|
#4 training progress
100%|
#5 training progress
100%|
reward: 200.0
max Q:33
```

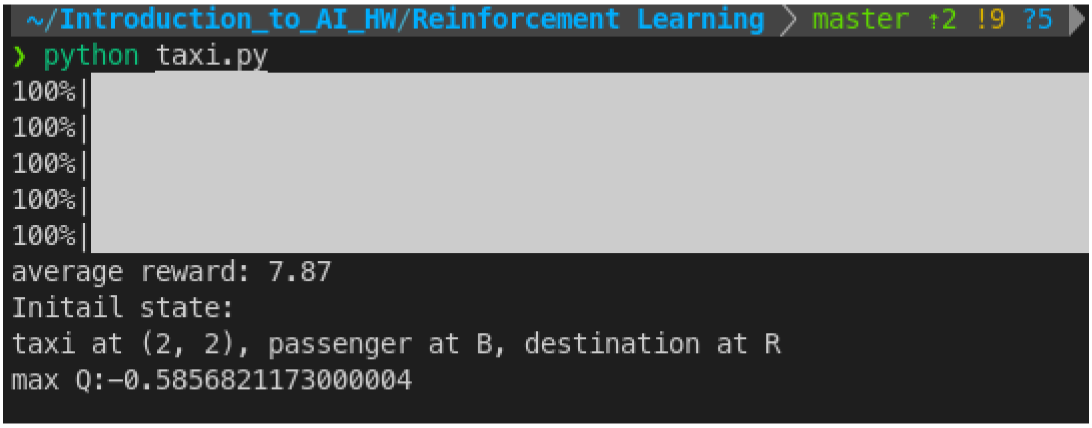


Compare

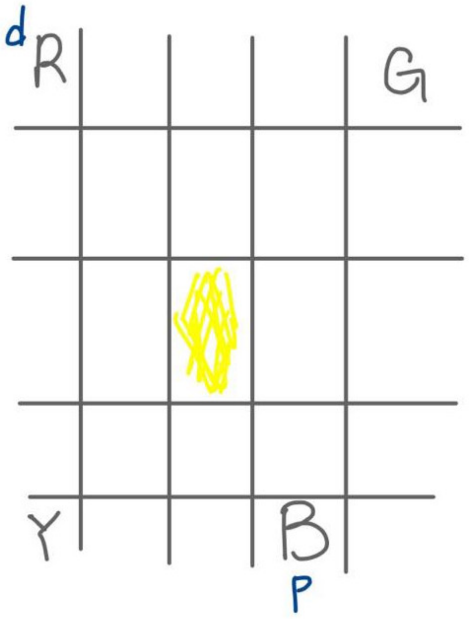


Part II. Question Answering (50%):

1. Calculate the optimal Q-value of a given state in Taxi-v3 (the state is assigned in google sheet), and compare with the Q-value you learned (Please screenshot the result of the "check_max_Q" function to show the Q-value you learned). (4%)



```
~/Introduction_to_AI_HW/Reinforcement Learning  master ↑2 !9 ?5
❯ python taxi.py
100%|
100%|
100%|
100%|
100%|
average reward: 7.87
Initail state:
taxi at (2, 2), passenger at B, destination at R
max Q:-0.5856821173000004
```



optimal: 右下下載上上上上
左左左放

$$-1+(-1)\times(0.9^1+0.9^2+0.9^3+0.9^4+0.9^5+0.9^6+0.9^7+0.9^8+0.9^9+0.9^{10})+20\times0.9^{11}$$

$$= -1+20\times0.9^{11}+(-1)\times\left(\frac{0.9^{11}-0.9}{0.1}\right)\approx -0.586$$

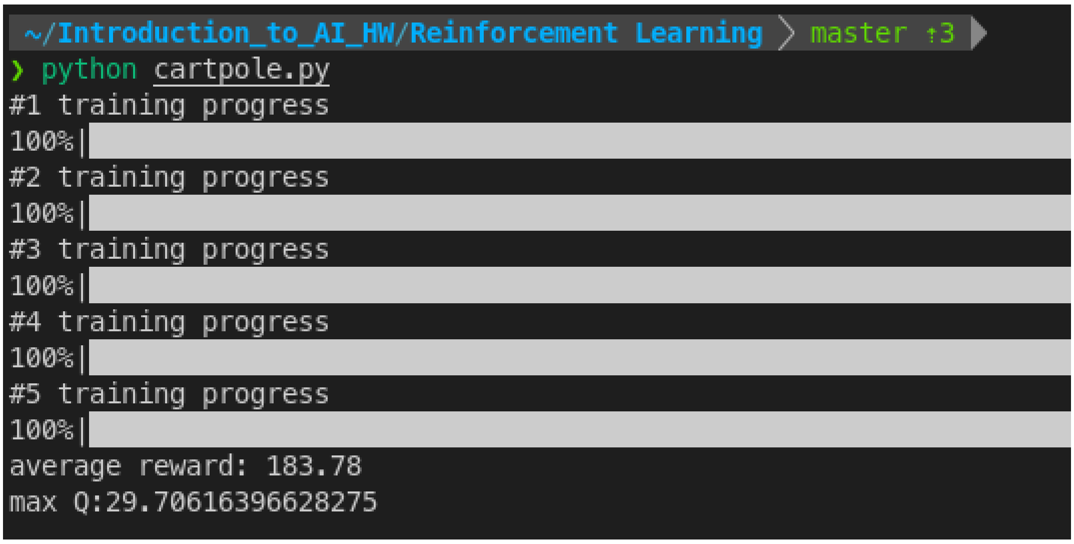
The Q value “check_max_Q” produce is similar to the optimal Q value I calculated.

2. Calculate the max Q-value of the initial state in CartPole-v0, and compare with the Q-value you learned. (Please screenshot the result of the “check_max_Q” function to show the Q-value you learned) (4%)

$$1+0.97+0.97^2+\cdots+0.97^{199}$$ (cartpole-V0最多到200回)

$$=\frac{1-0.97^{200}}{0.03}\approx 33.258$$

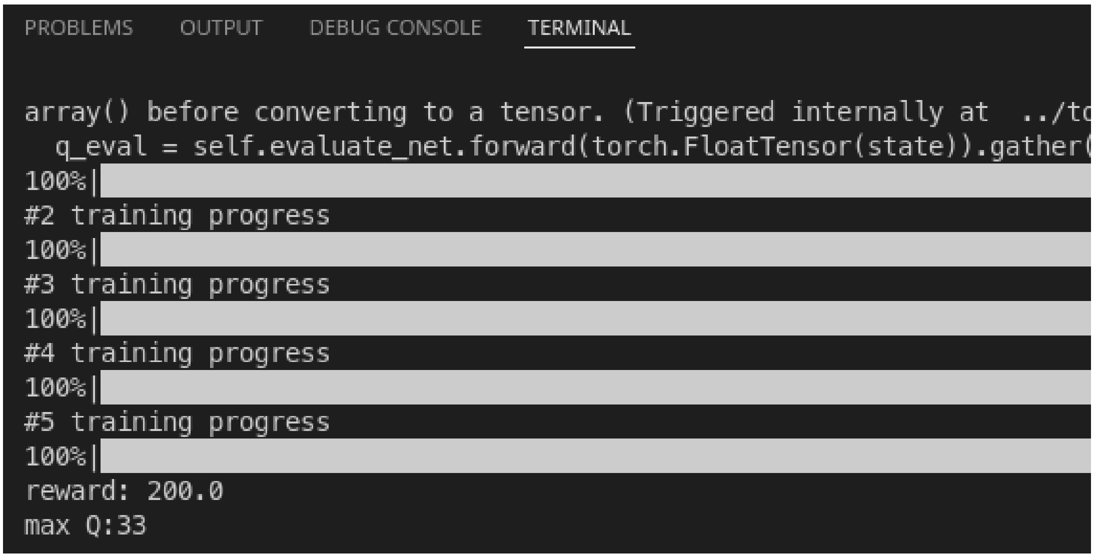
▼ cartpole



```
~/Introduction_to_AI_HW/Reinforcement Learning 〉 master ↑3
❯ python cartpole.py
#1 training progress
100%|
#2 training progress
100%|
#3 training progress
100%|
#4 training progress
100%|
#5 training progress
100%|
average reward: 183.78
max Q:29.70616396628275
```

The Q value “check_max_Q” produce close but not equal to the optimal Q value I calculated.

▼ DQN



```
array() before converting to a tensor. (Triggered internally at  ../to
  q_eval = self.evaluate_net.forward(torch.FloatTensor(state)).gather(
100%|
#2 training progress
100%|
#3 training progress
100%|
#4 training progress
100%|
#5 training progress
100%|
reward: 200.0
max Q:33
```

The Q value “check_max_Q” produce is similar to the optimal Q value I calculated.

DQN estimated better Q value.

3.

a. Why do we need to discretize the observation in Part 2? (2%)

If we don’t discretize the observation, the state might be infinite and we cannot build the Q table.

Therefore, we need to discretize it.

b. How do you expect the performance will be if we increase “num_bins”? (2%)

The performance may become better if we increase “num_bins” a little bit, since we can consider more states.

But if we increase a lot, the performance may become worse. In my opinion, the Q value may not converge, and we will get bad performance.

c. Is there any concern if we increase “num_bins”? (2%)

Yes, I discussed it in question b.

Q value may not converge, if num_bins is large and we train with the same episode.

4. Which model (DQN, discretized Q learning) performs better in Cartpole-v0, and what are the reasons? (3%)

DQN. We can obverse the Q value it calculated. DQN has better Q value estimated than discretized Q learning.

5.

a. What is the purpose of using the epsilon greedy algorithm while choosing an action? (2%)

The algorithm help balance exploration and exploitation, and it is efficient.

b. What will happen, if we don’t use the epsilon greedy algorithm in the CartPole-v0 environment? (3%)

The program may not have a chance to do exploration, and the result may be weird, or the program will only choose random action. Maybe there is other methods to balance exploration and exploitation, but they may not run faster than the epsilon greedy algorithm.

c. Is it possible to achieve the same performance without the epsilon greedy algorithm in the CartPole-v0 environment? Why or Why not? (3%)

Yes. In fact, randomly choose between exploration and exploitation may not be the best solution.
Maybe the program can use some condition and observation to determine when should the program do exploration.

d. Why don’t we need the epsilon greedy algorithm during the testing section? (2%)

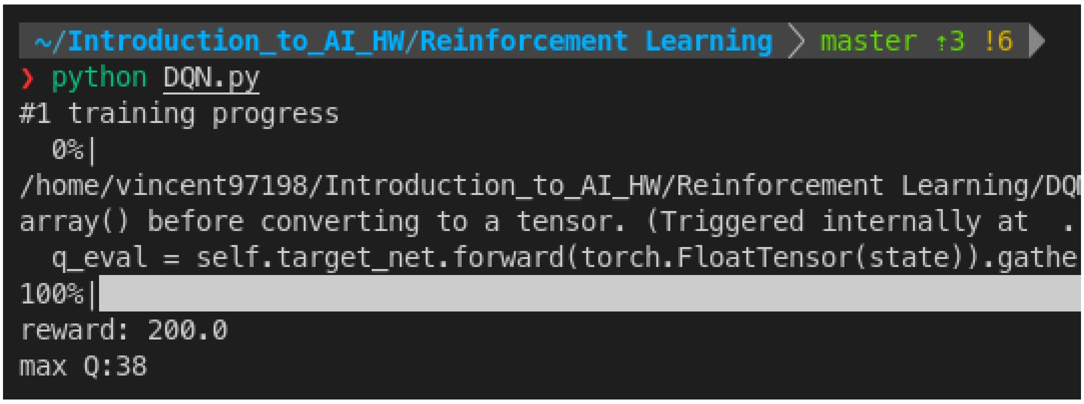
We assume that all Q value in Q table has already converged. Therefore, the agent don’t need exploration to explore other states. The agent just need to choose the best action.

6. Why is there “with torch.no_grad():“ in the “choose_action” function in DQN? (3%)

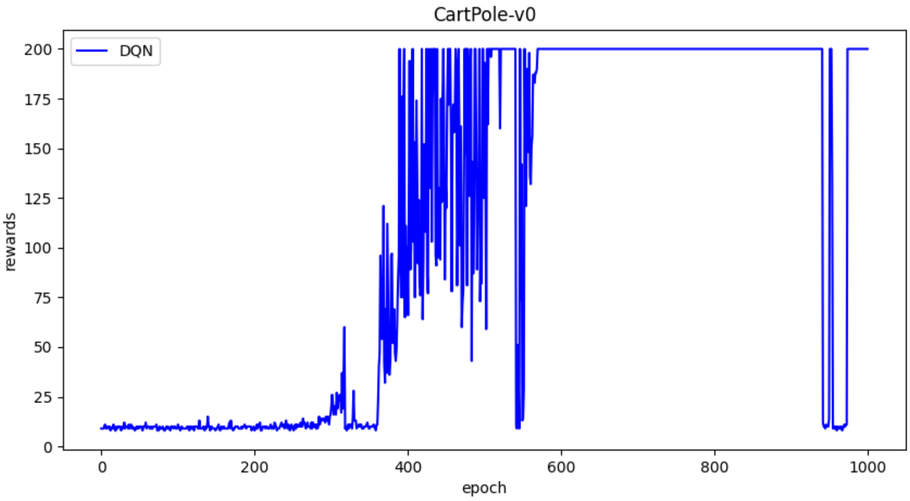
   In choose_action we don’t need to optimize the neural network. Thus, we can use `with torch.no_grad` to speed up.

7.

   a. Is it necessary to have two networks when implementing DQN? (1%)



```
 ~/Introduction_to_AI_HW/Reinforcement Learning 〉 master ↑3 !6
❯ python DQN.py
#1 training progress
  0%|
/home/vincent97198/Introduction_to_AI_HW/Reinforcement Learning/DQ
array() before converting to a tensor. (Triggered internally at  .
  q_eval = self.target_net.forward(torch.FloatTensor(state)).gathe
100%|
reward: 200.0
max Q:38
```



   I try to use only one network. The final reward is 200. Therefore, two networks may not be necessary.

   b. What are the advantages of having two networks? (3%)

   Though one network can gain high score, the Q value is overestimated. In two networks model we can avoid overestimation.

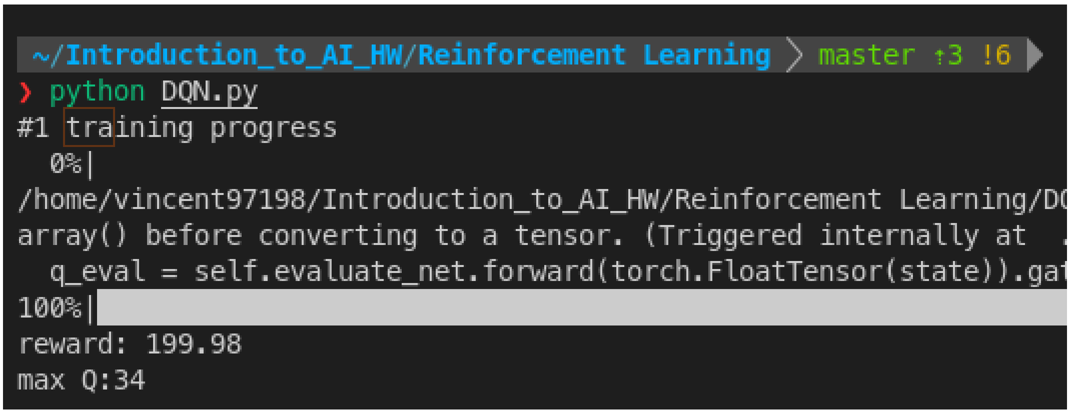
   c. What are the disadvantages? (2%)

   Compare with one network model’s learning curve and two networks model’s, we can discover that one network model converge faster than two networks model.

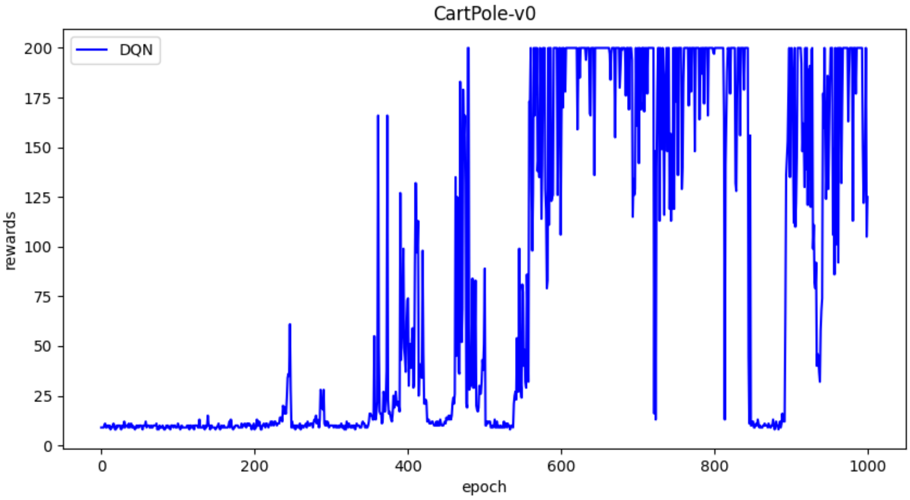
One network model reach 200 in 400 epoch, but two networks model reach 200 in 700 epoch.

8.

a. What is a replay buffer(memory)? Is it necessary to implement a replay buffer? What are the advantages of implementing a replay buffer? (5%)



```
 ~/Introduction_to_AI_HW/Reinforcement Learning  master ⇡3 !6
❯ python DQN.py
#1 training progress
  0%|
/home/vincent97198/Introduction_to_AI_HW/Reinforcement Learning/DQ
array() before converting to a tensor. (Triggered internally at  .
  q_eval = self.evaluate_net.forward(torch.FloatTensor(state)).gat
100%|
reward: 199.98
max Q:34
```

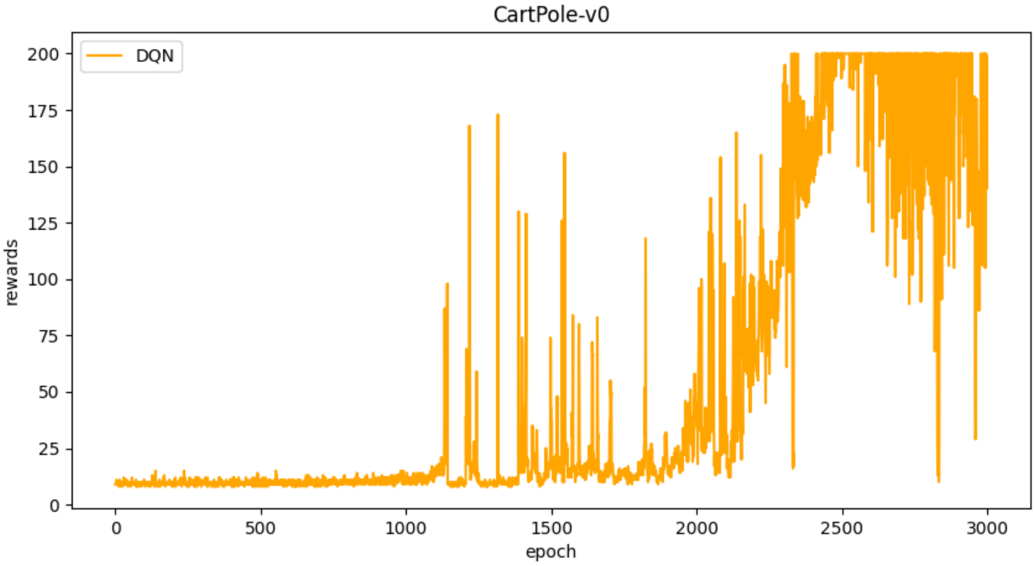


It is a buffer that record the history transition.

Yes, it is necessary to implement it.

Without the replay buffer, the agent cannot review what ii has learned. The picture above is the DQN program with batch size of replay buffer. It shows that the performance may become worse often.

b. Why do we need batch size? (3%)



It let the optimizer look more data in one time, so it can consider the better gradient. Therefore, Batch size makes the program converge faster and more stable.

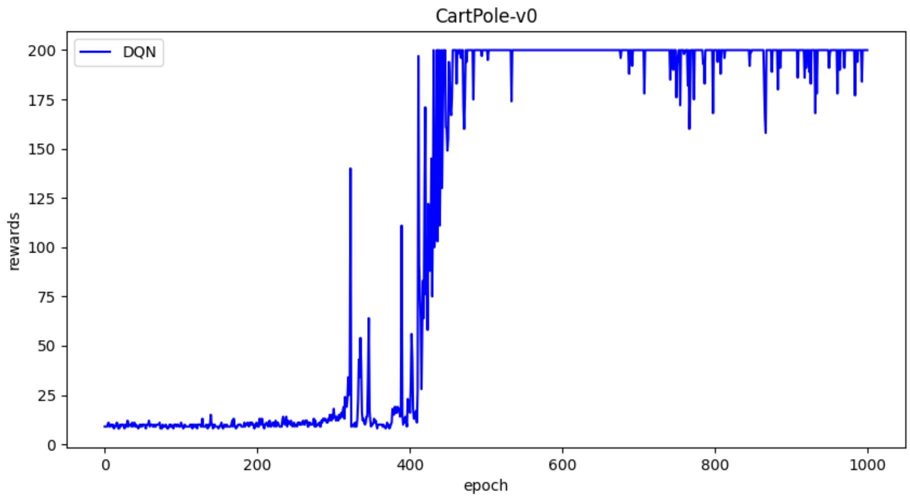
c. Is there any effect if we adjust the size of the replay buffer(memory) or batch size? Please list some advantages and disadvantages. (2%)

▼ Advantage

1. Small batch size: take less time to calculate
2. Small buffer: take less time to calculate
3. Large batch size: stable
4. Large buffer: stable and converge faster

▼ Disadvantage

1. Small batch size: converge slower and unstable
2. Small buffer: unstable
3. Large batch size: take long time to calculate
4. Large buffer: take long time to calculate



9.

a. What is the condition that you save your neural network? (1%)

In taxi, I save it when the episode is done.

In cartpole, I run 3 times test to check it's minimum reward. If it is not less than 150, the program save it.

In DQN, I save it when it every 5000 counts.

b. What are the reasons? (2%)

In taxi, I don't need to worry about it performance will much lower than 8. I just want the program run faster, so I let the program only save when the episode is done.

In cartpole, I want to make sure the Q table I save is great. Thus, I want it do the test before save. The reason of testing 3 times is that I want to make sure it is not just lucky and I want my program run fast.

10. What have you learned in the homework? (2%)

I learned the methods of searching data and reading document. The most difficult parts of this homework is the data type in PyTorch. It takes me lots of time.