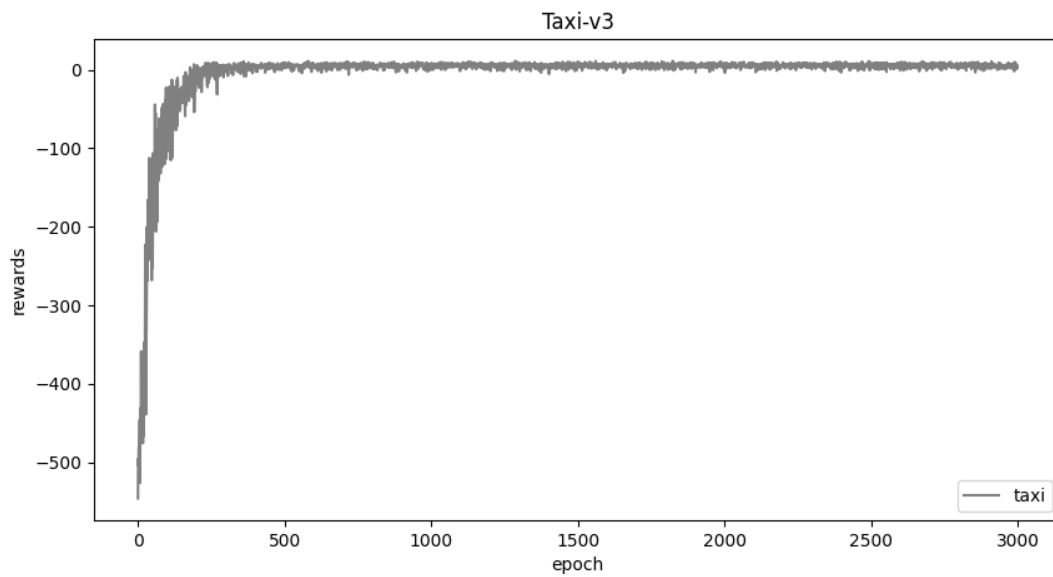


Reinforcement Learning report

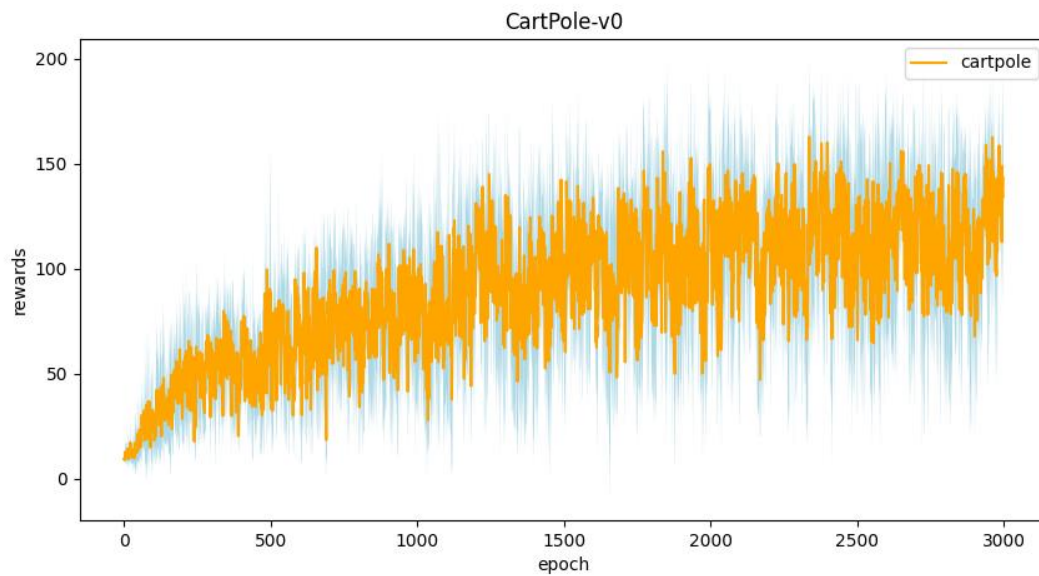
071125 葉長瀚

Part I. Implementation:

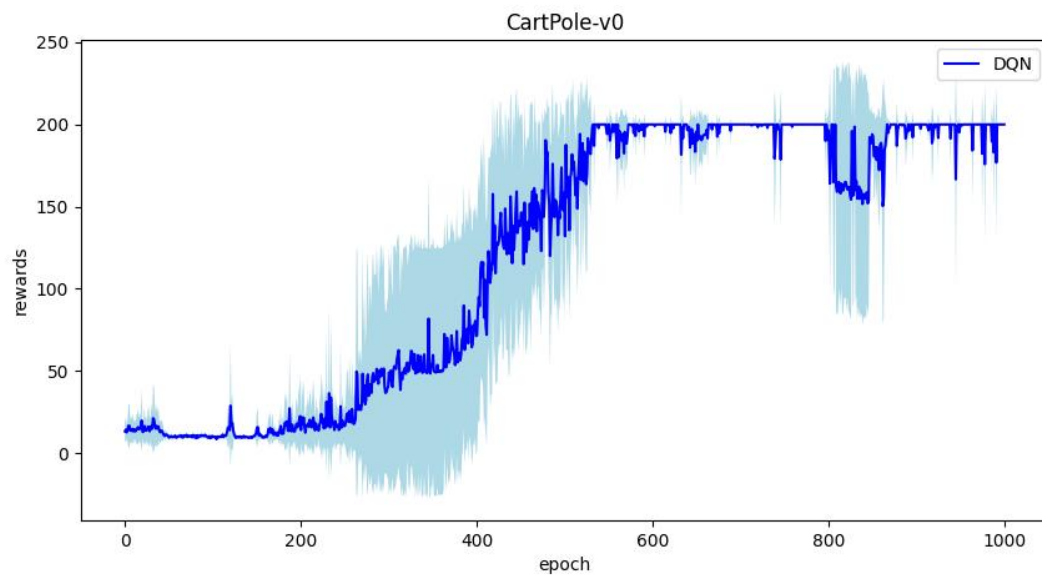
taxi.png:



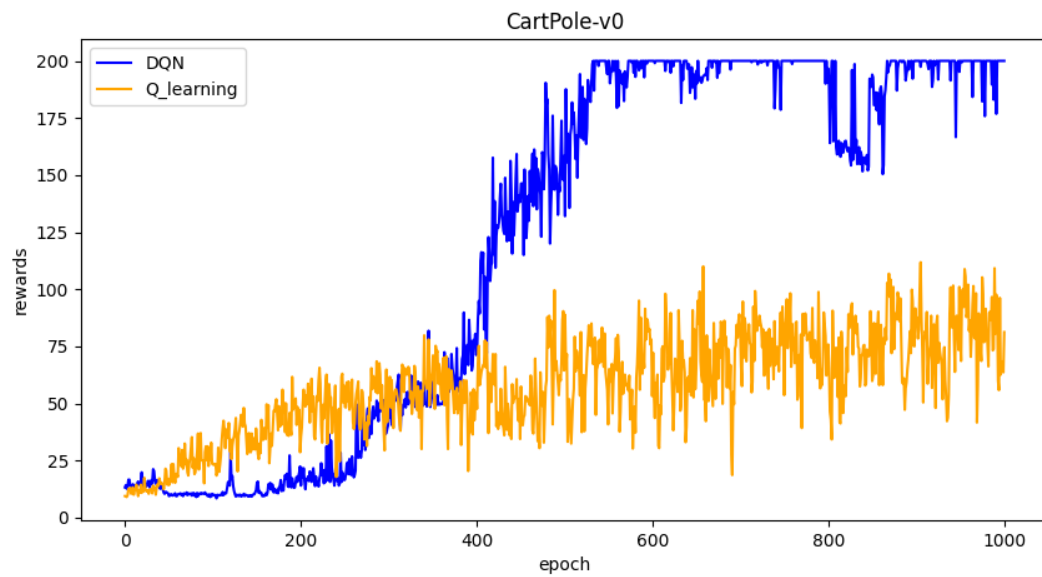
cartpole.png:



DQN.png:



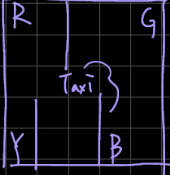
compare.png:



Part II. Question Answering:

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).

Initial state:
taxi at (2, 2), passenger at B, destination at G
max Q: 1.6226146700000021



state = 253 $\gamma = 0.9$
passenger = B destination = G
4 steps to pick up, 6 steps to drop off

$$Q_{opt} = -1 + 0.8(-1) + 0.8^2(-1) + \dots + 0.8^8(-1) + 0.8^9 \cdot 20$$

$$= - \left(\frac{1 - 0.9^9}{0.1} \right) + 0.9^9 \cdot 20$$

$$= -10 + 0.9^9 \cdot 30 = 1.6226$$

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)

max Q: 31.250807556595774
max Q: 0.04073899984359741

$$Q_{opt} = 1 + 0.9\gamma + 0.9\gamma^2 + \dots + 0.9\gamma^{199}$$

$$= \frac{1 - 0.9\gamma^{200}}{0.03} = 33.258$$

3.

- a. Why do we need to discretize the observation in Part 2?
The original observation is floating number, if we don't discretize, we aren't able to construct a valid q table.
- b. How do you expect the performance will be if we increase "num_bins"?
The performance will be greater if we increase num_bins, because we are able to describe the state more accurately.
- c. Is there any concern if we increase "num_bins"?
The required training time will be larger, since the increased size of q table.

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

DQN perform better.

In discretized Q learning, we discretize the state from infinite combinations of state, but in DQN the network learns directly from the raw data of state, so it can decide the action and calculate the Q value more precisely, resulting in better performance.

5.

- a. What is the purpose of using the epsilon greedy algorithm while choosing an action?

Maintain a balance between exploration and exploitation, preventing our training from no exploration of the environment and exploit the q table constructed by always taking the maximum reward action.

- b. What will happen, if we don't use the epsilon greedy algorithm in the CartPole-v0 environment?

If we always choose the action randomly, each episode will be easily ended, since we don't take the best action, and it will also require both tremendous time and luck for agent to get and construct a good q table.

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

No, as answered in b., the pole will fall down easily.

- d. Why don't we need the epsilon greedy algorithm during the testing section?

The purpose of testing section is to test the q table we got after episodes of training, so we can't and won't need to do the exploration part.

6. Why is there "with torch.no_grad():" in the "choose_action" function in DQN?

There's no need of calculating gradient and back propagation in *choose action*, so we can increase the computational time by *with torch.no_grad()*.

7.

- a. Is it necessary to have two networks when implementing DQN?

Yes, if there's no target network, the learning will be unstable causing Q value to diverge.

- b. What are the advantages of having two networks?

The process will be more stable, and Q value can converge

- c. What are the disadvantages?

Extra memory use, increase the time of learning

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?

Replay buffer is a large buffer storing past experience from the agent interacting with the environment and we sample training data from it, instead of using the latest experience.

Yes.

Break the correlation between state, treat each state as an individual, learn from it for many times, and make the most use of the experience.

- b. Why do we need batch size?

We need training data representing the whole data to get the loss function or there will be bias in each update, so we sample a batch size of data from the memory.

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

Increase the replay buffer size means we can store more complete state but will need more memory to implement it.

Increase the batch size require more training time but will kind of result in better performance.

9.

- a. What is the condition that you save your neural network?

First, when the episode is done and the gotten reward is larger or equal to the maximum reward gotten so far. Second, sample a batch of state from memory, forward them to the current neural network and saved neural network to get the Q value, if all of the Q value gotten from the former is larger than gotten from the latter, then save the neural network.

b. What are the reasons?

If gotten reward or Q value is smaller than the one the network was saved, it means that network will not work better than saved one, so I think that will be a reasonable condition to determine whether to update the neural network.

10. What have you learned in the homework?

The basic structure and concept of how Q learning and deep Q network works, and experiencing using the PyTorch library.