**Homework 4:**

**Reinforcement Learning**

**Part I. Implementation (-5 if not explain in detail):**

* **Please screenshot your code snippets of Part 1 ~ Part 3, and explain your implementation**
* **Part 1**
* **Part 2**
* **Part 3**

**Part II. Experiment Results:**

**Please paste taxi.png, cartpole.png, DQN.png and compare.png here.**

**1. taxi.png:**

**2. cartpole.png**

**3. DQN.png**

**4. compare.png**

**Part III. Question Answering (50%):**

1. Calculate the optimal Q-value of a given state in Taxi-v3, 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). **(10%)**
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) **(10%)**
3. Why do we need to discretize the observation in Part 2? **(3%)**
4. How do you expect the performance will be if we increase “num\_bins”? **(3%)**
5. Is there any concern if we increase “num\_bins”? **(3%)**
6. Which model (DQN, discretized Q learning) performs better in Cartpole-v0, and what are the reasons? **(5%)**
7. What is the purpose of using the epsilon greedy algorithm while choosing an action? **(3%)**

1. What will happen, if we don’t use the epsilon greedy algorithm in the CartPole-v0 environment? **(3%)**
2. Is it possible to achieve the same performance without the epsilon greedy algorithm in the CartPole-v0 environment? Why or Why not? **(3%)**
3. Why don’t we need the epsilon greedy algorithm during the testing section? **(3%)**
4. Why does “with torch.no\_grad():“ do inside the “choose\_action” function in DQN? **(4%)**