CSE 474/574
Machine Learning
Project 4 Report

Yun Bao
50159678
2018/12/05

1. What parts have you implemented?

   I have implemented three parts in this project. For the first part, I built a three-layer neural network with two hidden layers and an output layer. The first layer's weights are 128*6, the second layer's weights are 128*128, and the output layer's weights are 4*128. I used "relu" as activation function for first and second layer, and "linear" activation function for the output layer to return real values. Then I set the input shape of first layer to the size of the observation space. Lastly, I added these three layers to my sequential keras model as the Deep Q-network. For the second part, I followed the epsilon formula $\epsilon = \epsilon min + (\epsilon max - \epsilon min) * e^{-\lambda|S|}$ to implement epsilon function for the agent to decrease the chance of random actions. For the third part, I implemented Q-function to help the agent to pick the highest reward of next state. If the current state is the final state, then the agent will keep the reward he already has and terminate, otherwise, the agent will have the current reward plus the maximum reward of next state.

2. What is their role in training the agent?

   In this case, the three -layer neural network acts as Deep Q-network. DQN will pass in a stack of six tuple as an input, then the input will go through two hidden layers and output a vector of q-values for the agent to determine the maximum reward of each steps. Epsilon plays important role in this project. In the beginning, the agent needs to explore the way to the target, then the agent will take random actions at first. Once the epsilon gets smaller and smaller, the agent starts to see the pattern to the target and explore the environment. Lastly, Q-function helps the agent to figure out the maximum q-values of each steps, then the agent can make the best decisions to get the highest reward, if the agent is already at the final state, Q-function will him the reward that he already has.

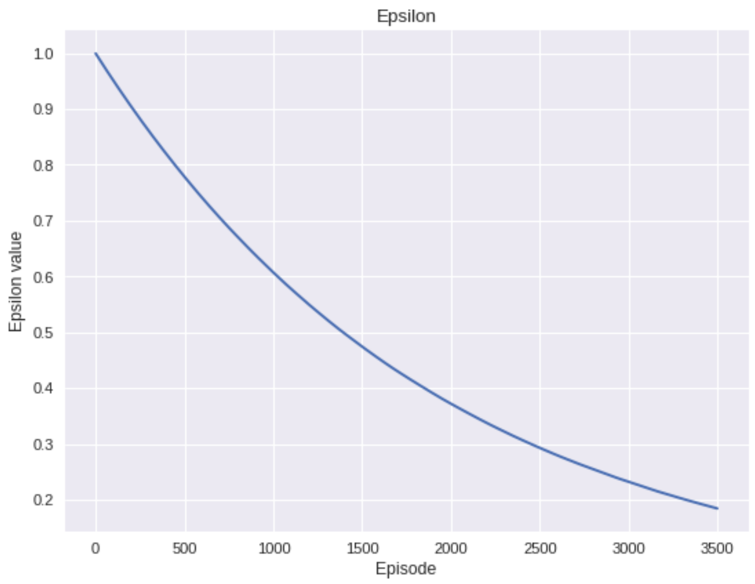
3. Can these snippets be improved and how it will influence the training the agent?

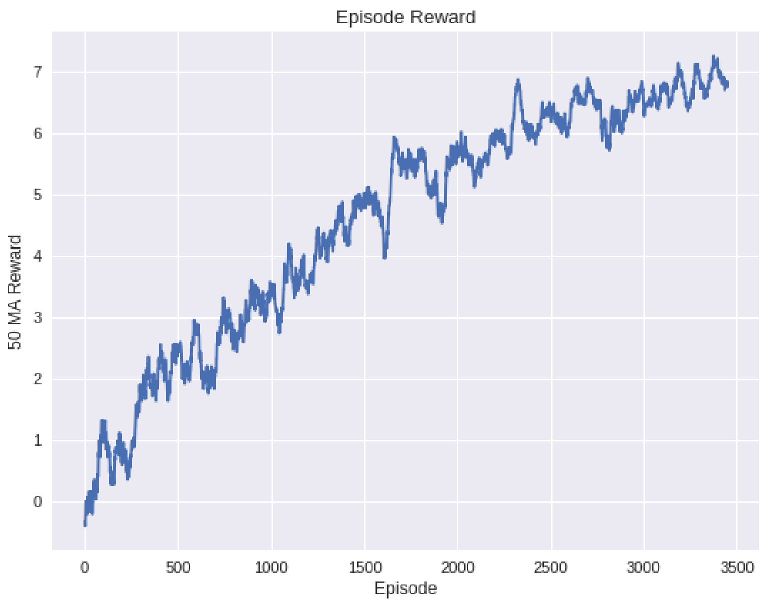
   If we can reduce the epsilon value in the beginning, the chances are higher for the agent to pick the highest reward at first, also it increases the chances that the DQN value will be higher than epsilon in the beginning. Then later on, epsilon will be smaller and smaller, so the agent will learn well very quickly to find the best way to the target. Also, we can improve our Q-function. Instead of looking at the q-values, we can play around with gamma, if gamma is getting bigger, the higher reward the agent will have. If gamma is getting smaller, the lower reward the agent will get.

4. How quickly your agent were able to learn?

   In my case, my agent takes 9-10 seconds to finish 100 episodes, and in 3500 episodes, my agent can only catch the target once. Therefore, I think the agent is pretty slow to learn under 3500 episodes. If you increase the episodes to above 5000 episodes, the agent will start to learn quickly. Epsilon will get closer and closer to zero, so the agent will make the best decision to get the highest reward for each step. Then later on, the agent will always pick the highest reward. In my case, my agent will always get around 7 reward after 4700 episodes. (maximum reward is 8)

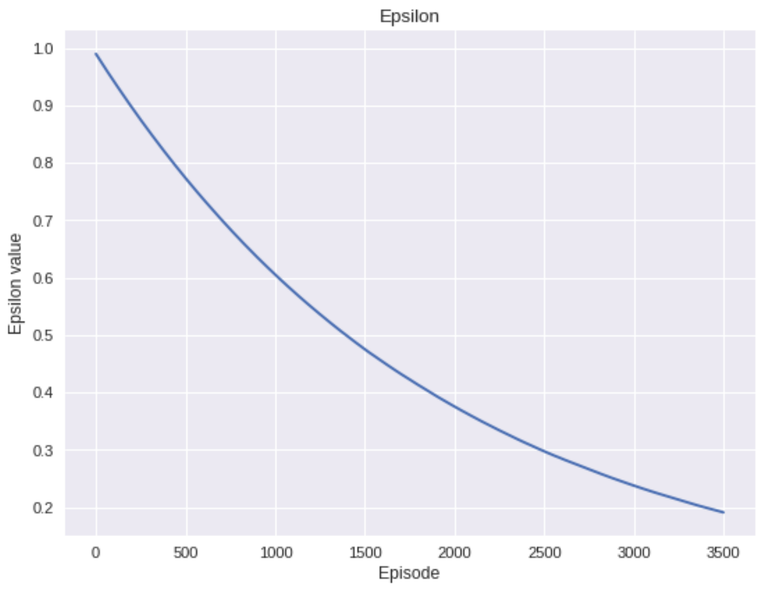
Max epsilon	1	0.99	0.98	0.97
Min epsilon	0	0.01	0.03	0.05

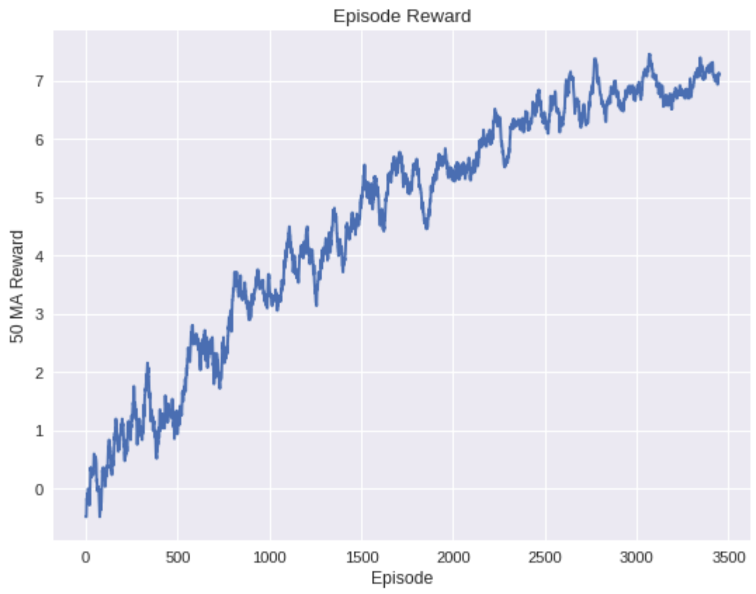




Max: 1

Min: 0



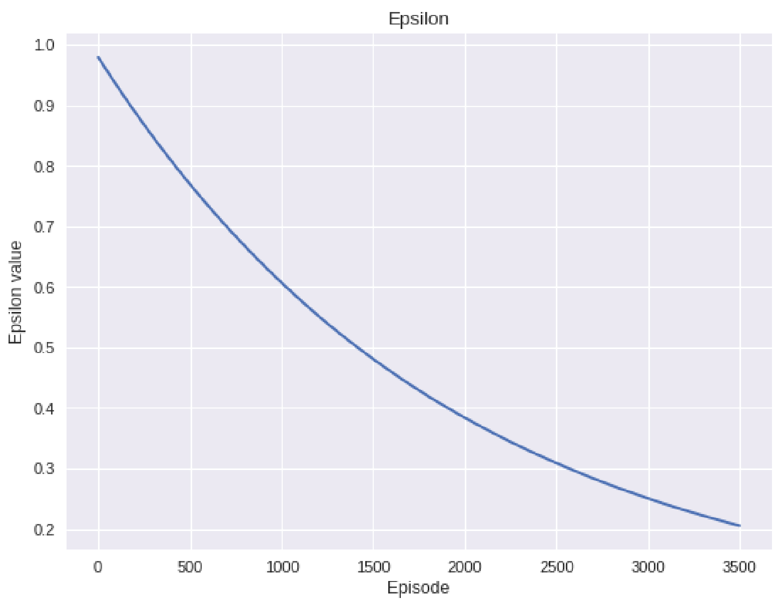


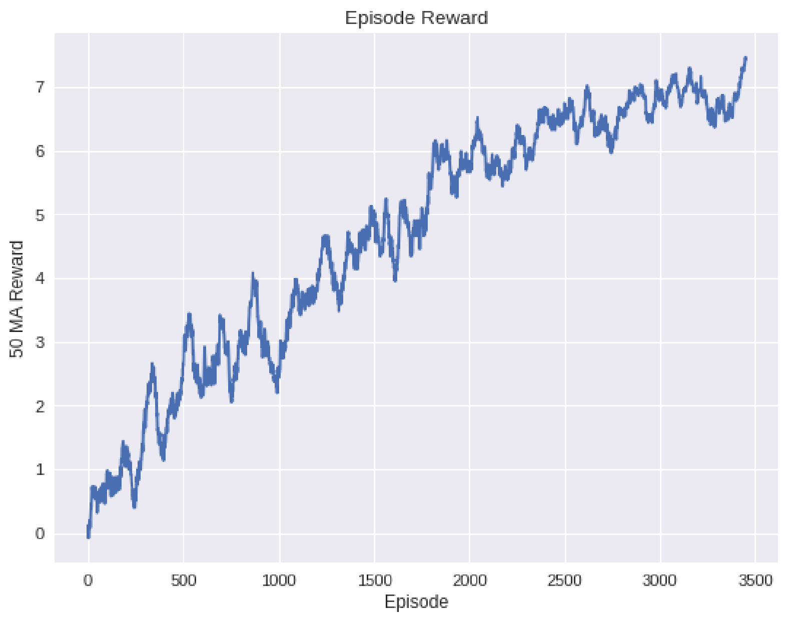
Max:0.99

Min:0.01

Slightly different from above, epsilon reduces a little bit quicker than the previous one.

However, the rewards don't change that much.

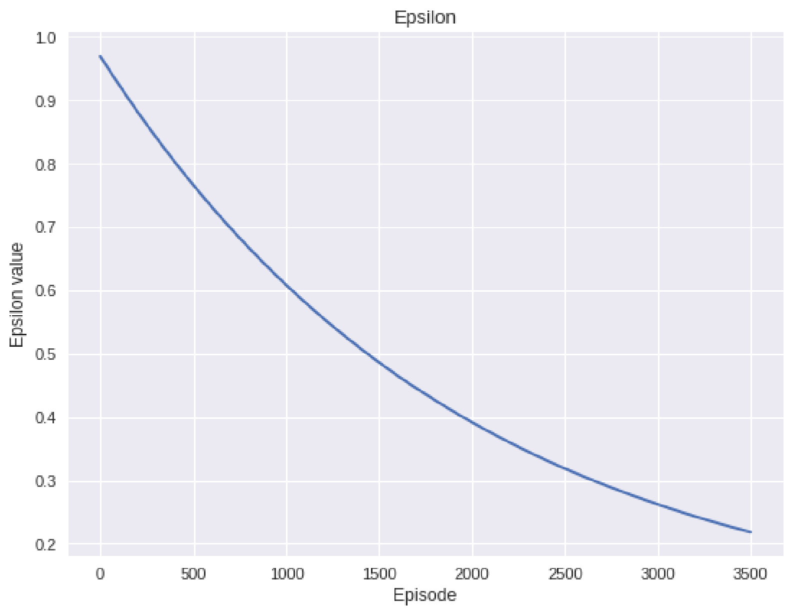


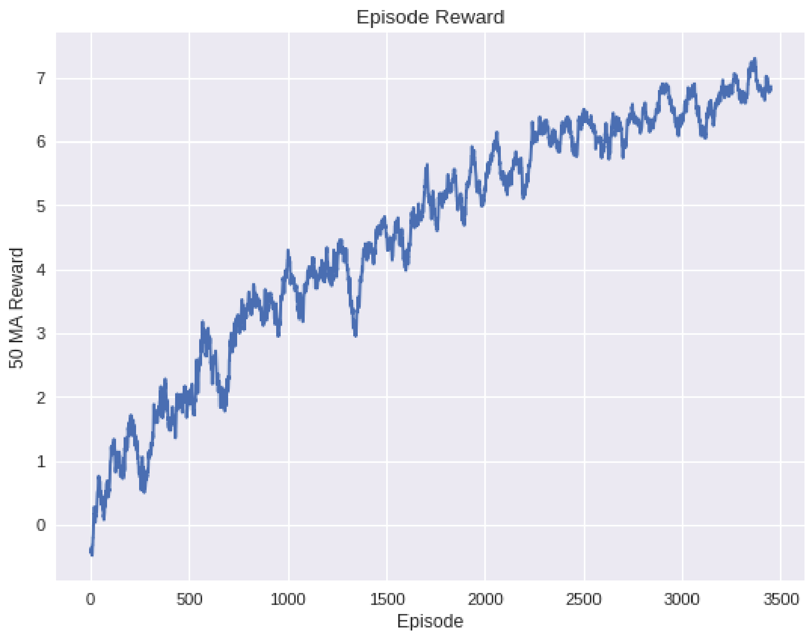


Max: 0.98

Min: 0.03

The rewards were not stable between 500 episodes and 1000 episodes.

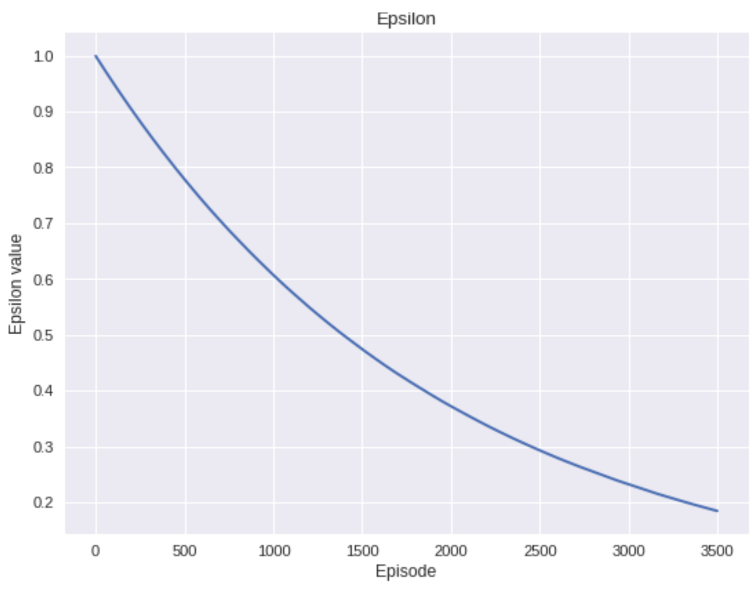


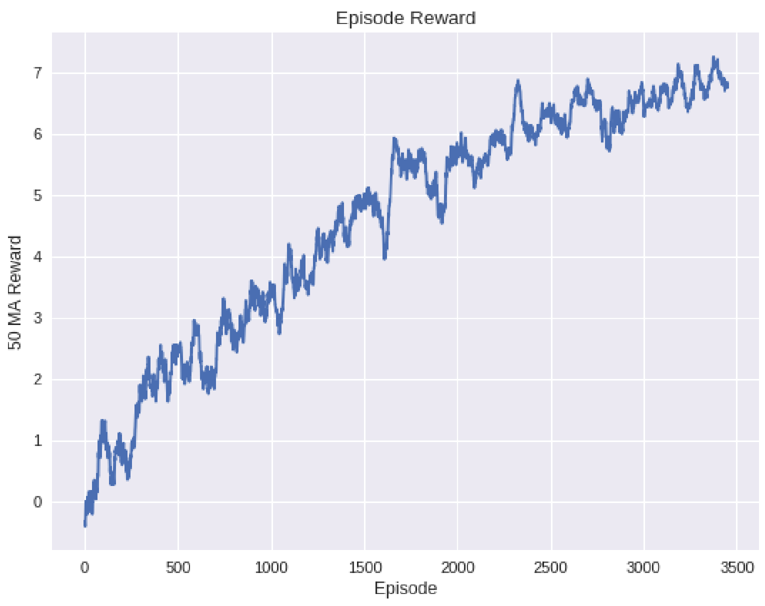


Max: 0.97

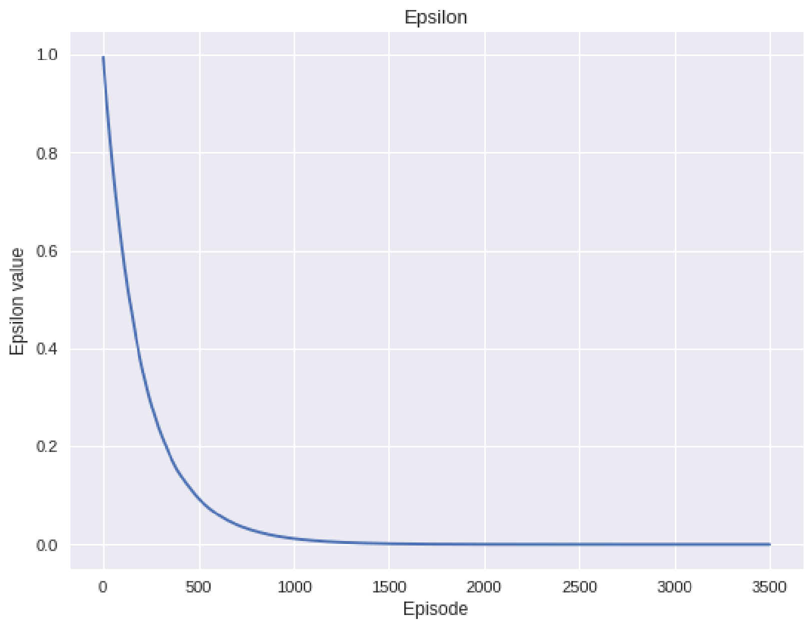
Min:0.05

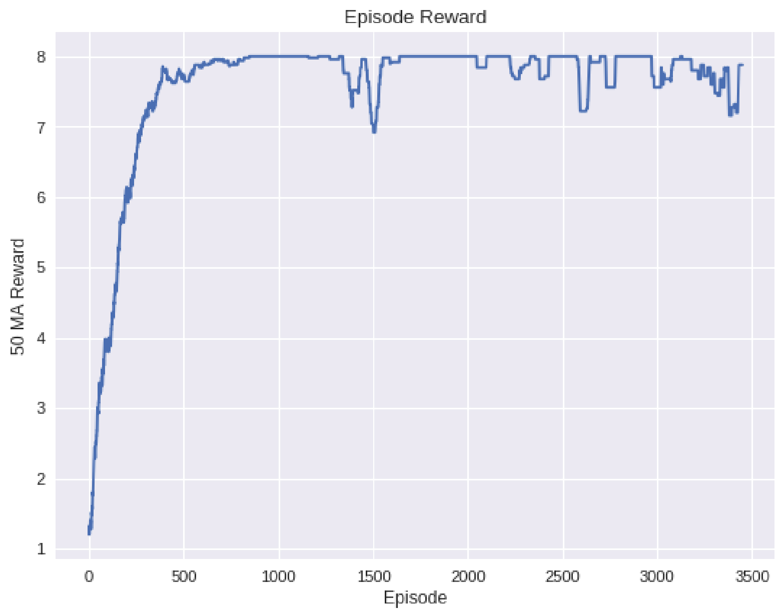
gamma	0.00005	0.0005	0.005





Gamma: 0.00005

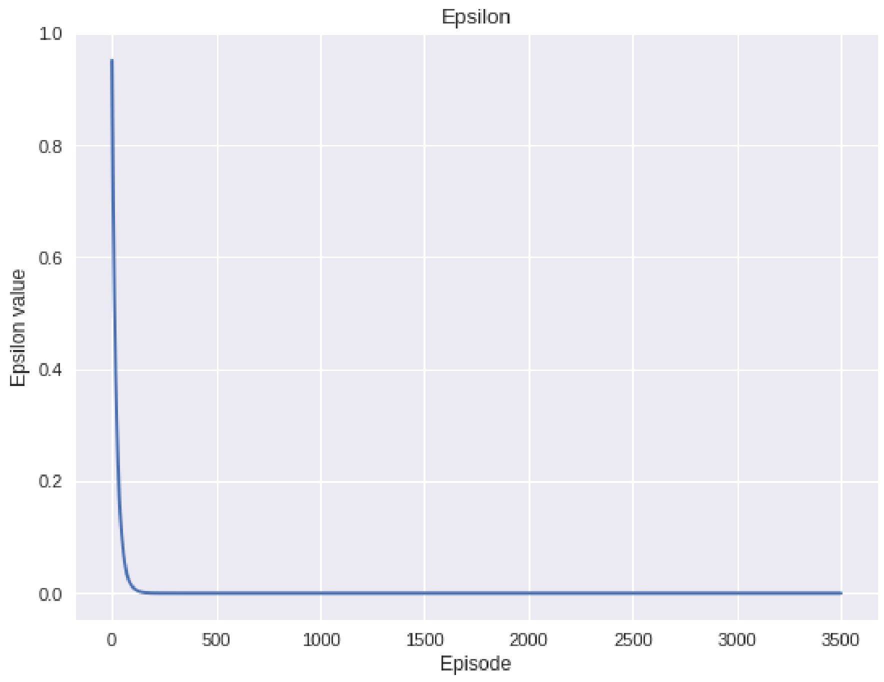


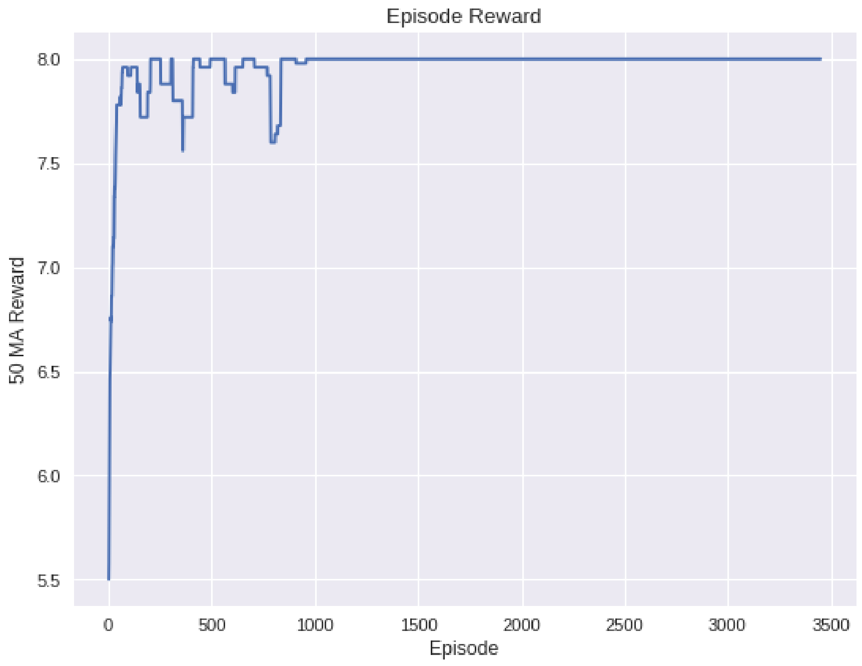


Gamma:0.0005

Epsilon decreases very fast, that makes the agent easier to pick the highest reward. In the second graph, we can see the agent has learned very well, the rewards always stay at the top.

If gamma gets larger, it is easier for the agent to make the best decision and pick the highest reward





Gamma:0.005

If gamma gets larger, it is easier for the agent to make the best decision and pick the highest reward

Writing task:

1. Explain what happens in reinforcement learning if the agent always chooses the action that maximizes the Q-value. Suggest two ways to force the agent to explore.

   The first way is what we use in this project, our DQN will be initialized to a random value when we start the agent, so doesn't matter what epsilon is, the agent will always make random choices in the beginning. If we start the agent on the top left corner, then initialized DQN says going left is the maximum q-value, the agent will stay at the same position if it goes left. Therefore, in this case, we need to force the agent to explore.

   The second way is we can make epsilon to be higher in the beginning. Epsilon is the percentage of the agent will make random actions. Therefore, if we increase epsilon, the agent will have higher chances to make random choices in the beginning, the agent keeps getting lower rewards, then it is necessary for the agent to explore later on.

2. $Q(S_4, down) = Q(S_4, up) = Q(S_4, left) = Q(S_4, right) = 0$

   $Q(S_3, down) = 1 + \gamma * maxaQ(S_4, a) = 1 + 0.99*0 = 1$

   $Q(S_3, right) = 0 + \gamma * maxaQ(S_3, a) = 0 + 0.99*1 = 0.99$

   $Q(S_3, up) = -1 + \gamma * maxaQ(S_n, a) = -1 + 0.99*1 = -0.01$

   $Q(S_3, left) = -1 + \gamma * maxaQ(S_2, a) = -1 + 0.99*1.99 = 0.97$

   $Q(S_2, right) = 1 + \gamma * maxaQ(S_3, a) = 1 + 0.99*1 = 1.99$

   $Q(S_2, down)$ = symmetric above = 1.99

   $Q(S_2, left) = -1 + \gamma * maxaQ(S_m, a) = -1 + 0.99*1 = -0.01$

   $Q(S_2, up) = -1 + \gamma * maxaQ(S_1, a) = -1 + 0.99*2.97 = 1.94$

   $Q(S_1, up) = 0 + \gamma * maxaQ(S_1, a) = 0 + 0.99*2.97 = 2.94$

   $Q(S_1, down) = 1 + \gamma * maxaQ(S_2, a) = 1 + 0.99*1.99 = 2.97$

   $Q(S_1, left) = -1 + \gamma * maxaQ(S_0, a) = 2.9$

   $Q(S_1, right) = 1 + \gamma * maxaQ(S_m, a) = 1 + 0.99*1 = 1.99$

   $Q(S_0, up) = 0 + \gamma * maxaQ(S_0, a) = 3.9$

   $Q(S_0, left) = 0 + \gamma * maxaQ(S_0, a) = 3.9$

   $Q(S_0, down) = 1 + \gamma * maxaQ(S_w, a) = 3.94$

   $Q(S_0, right) = 1 + \gamma * maxaQ(S_1, a) = 3.94$

State	Up	Down	Left	Right
0	3.9	3.94	3.9	3.94
1	2.94	2.97	2.9	1.99
2	1.94	1.99	-0.01	1.99
3	-0.01	1	0.97	0.99
4	0	0	0	0