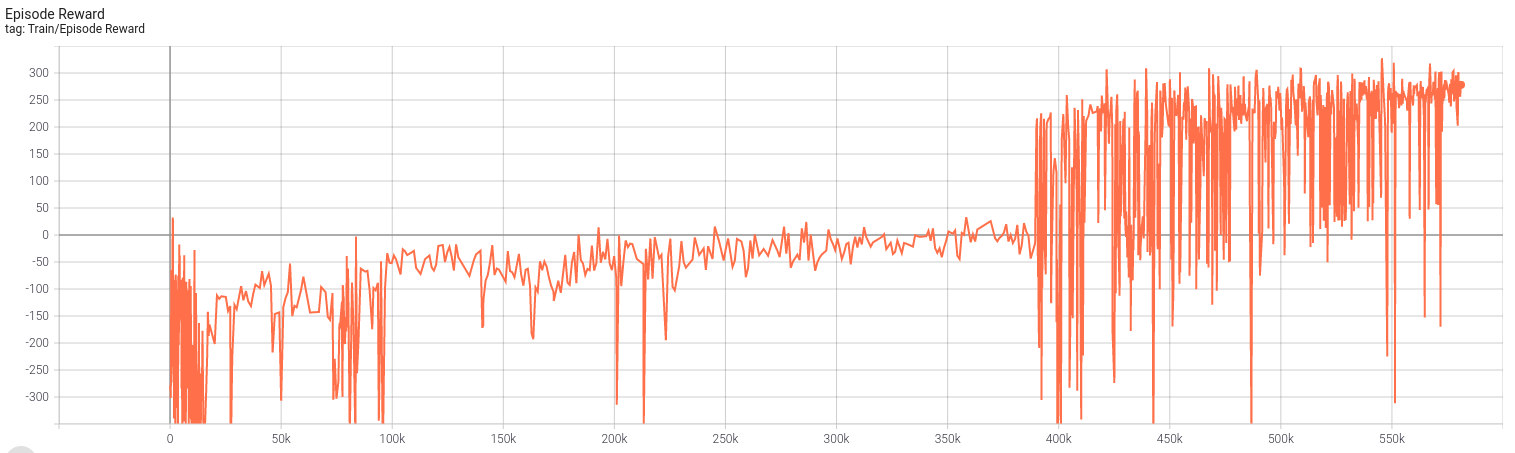
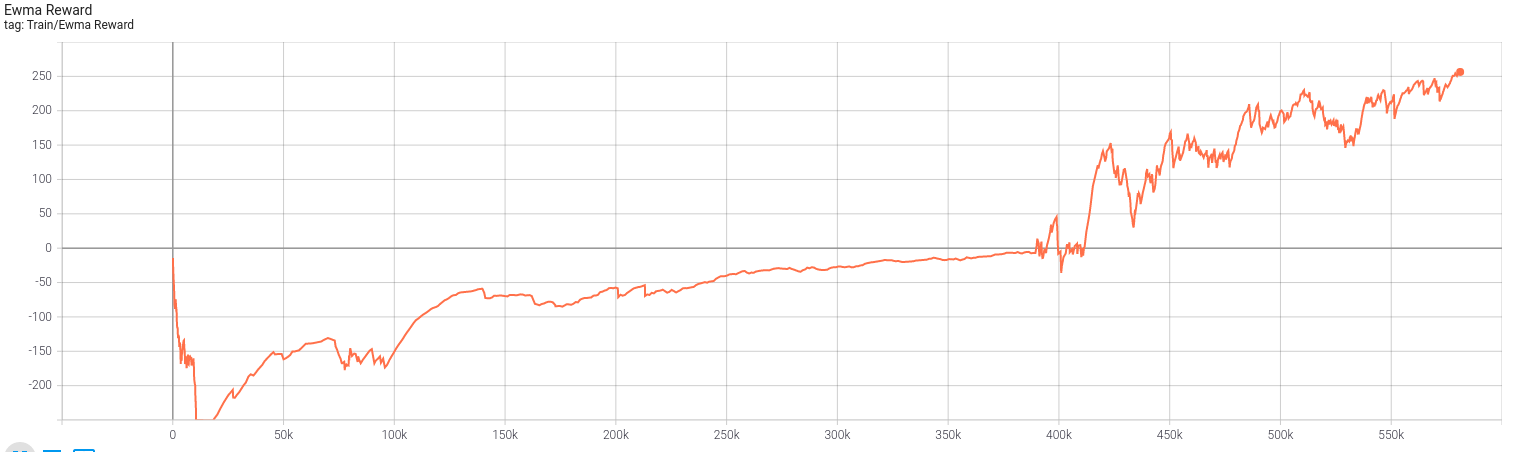
LAB6 Deep Q-Network and Deep Deterministic Policy Gradient

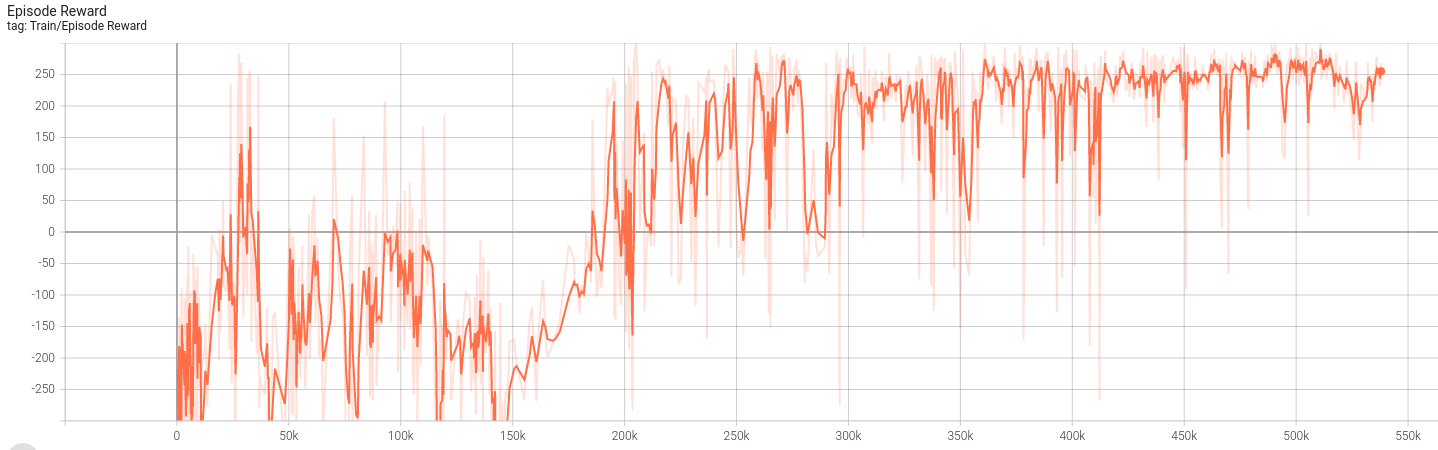
1. Tensorboard

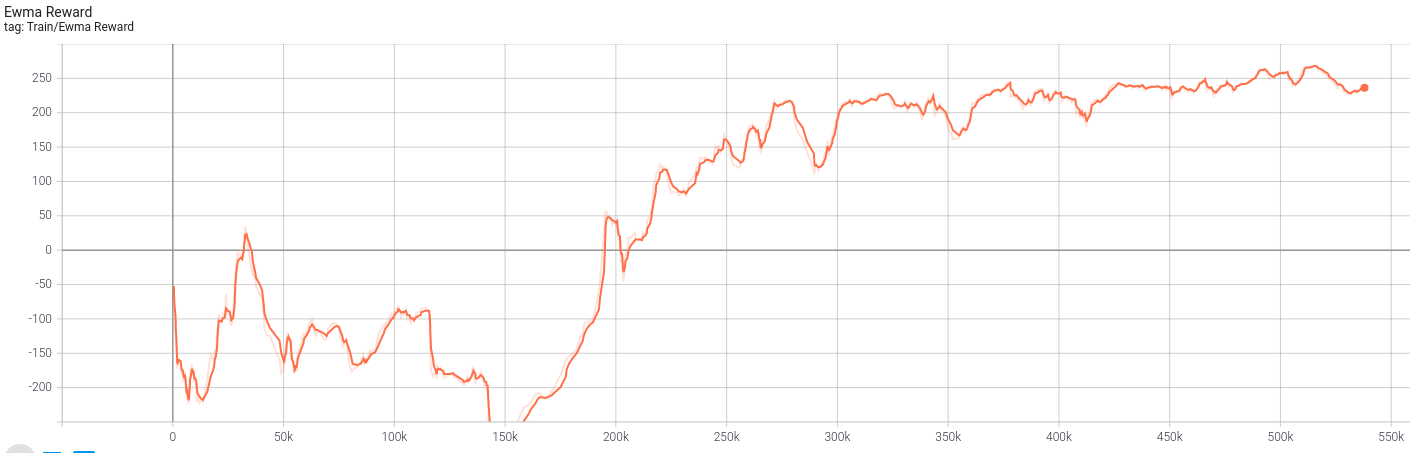
LunarLander-v2 (DQN)





LunarLanderContinuous-v2 (DDPG)

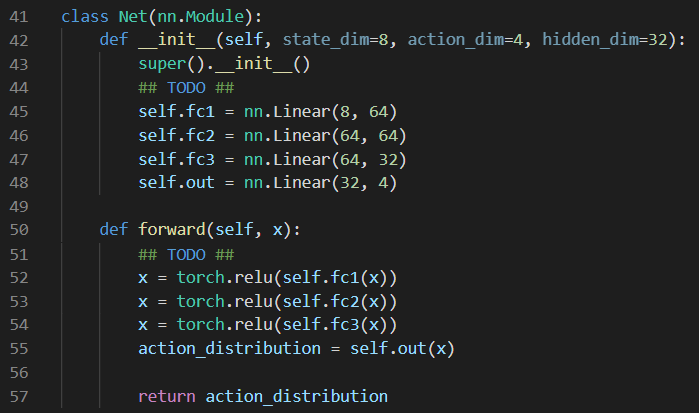




1. Implement detail
2. **Describe your major implementation of both algorithms in detail. (TODO)**

DQN :

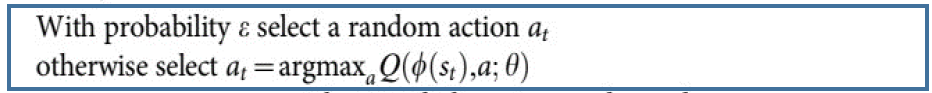
Create a network to generate distribution of four action .

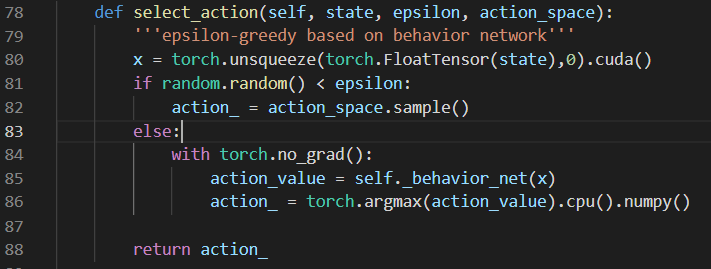


Set the optimizer and loss function .

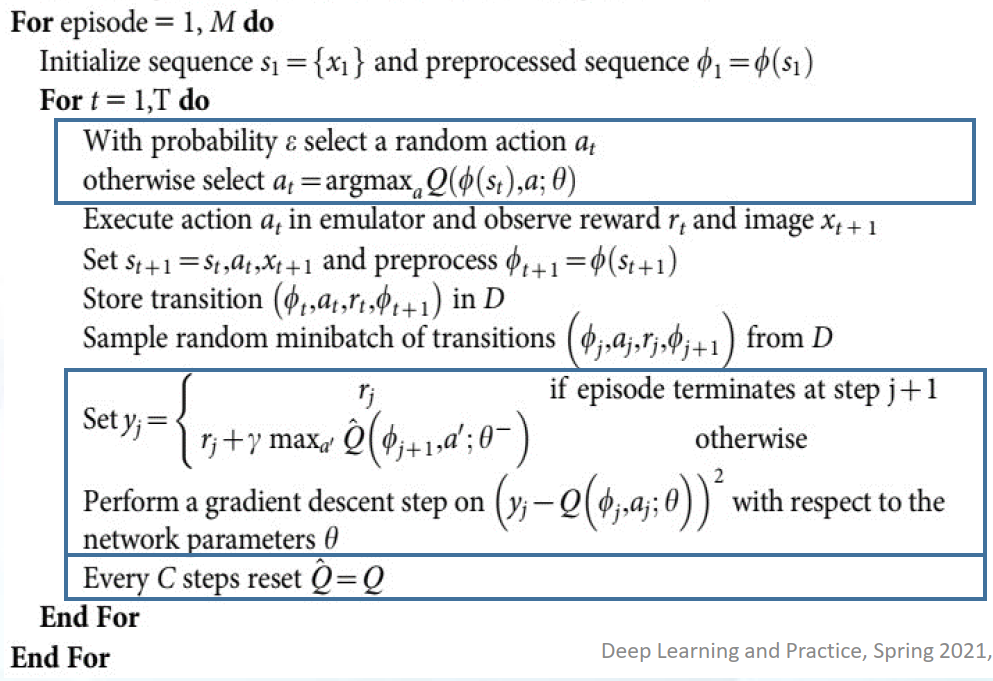


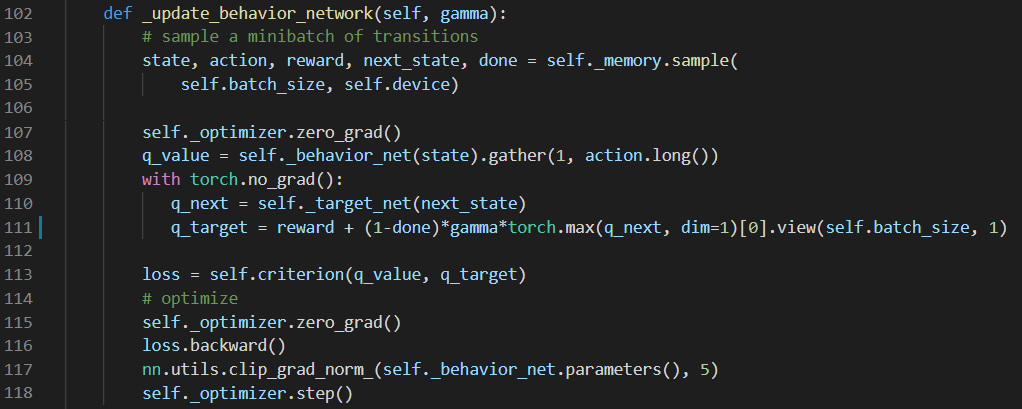
When training , use the action of the highest possibility Q( S, ai ) or randomly choose action with the epsilon .



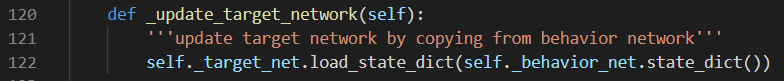


Update the behavior network with the algorithm :





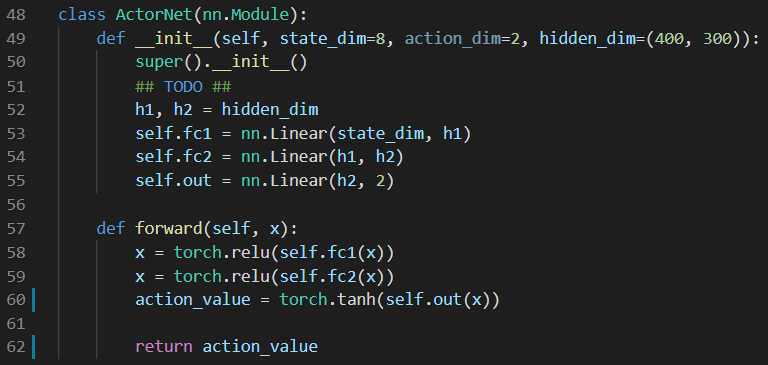
Update the target network with behavior network every 4 steps .



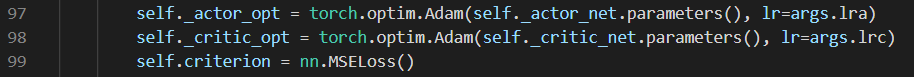
My DQN training parameters are using default .

DDPG :

Create a actor network to generate value of “Main engine” and “Left-Right engine” , so the num of the output is two .

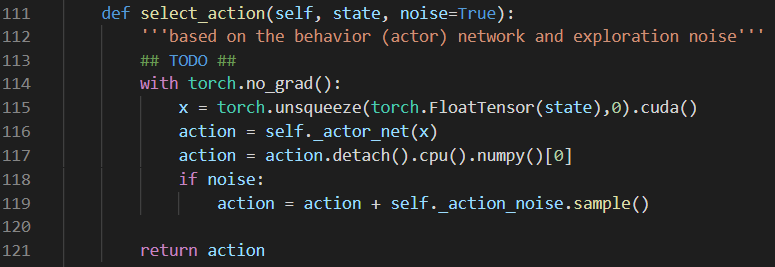


Set the optimizer and loss function .

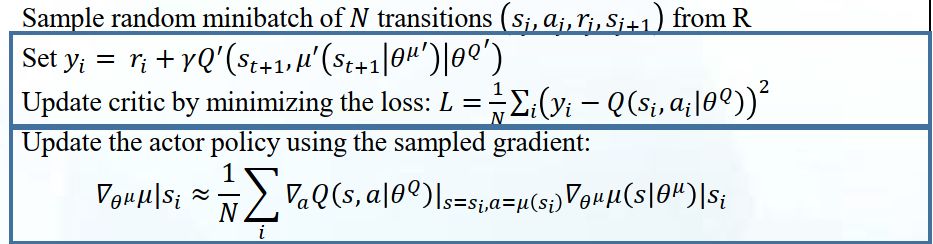


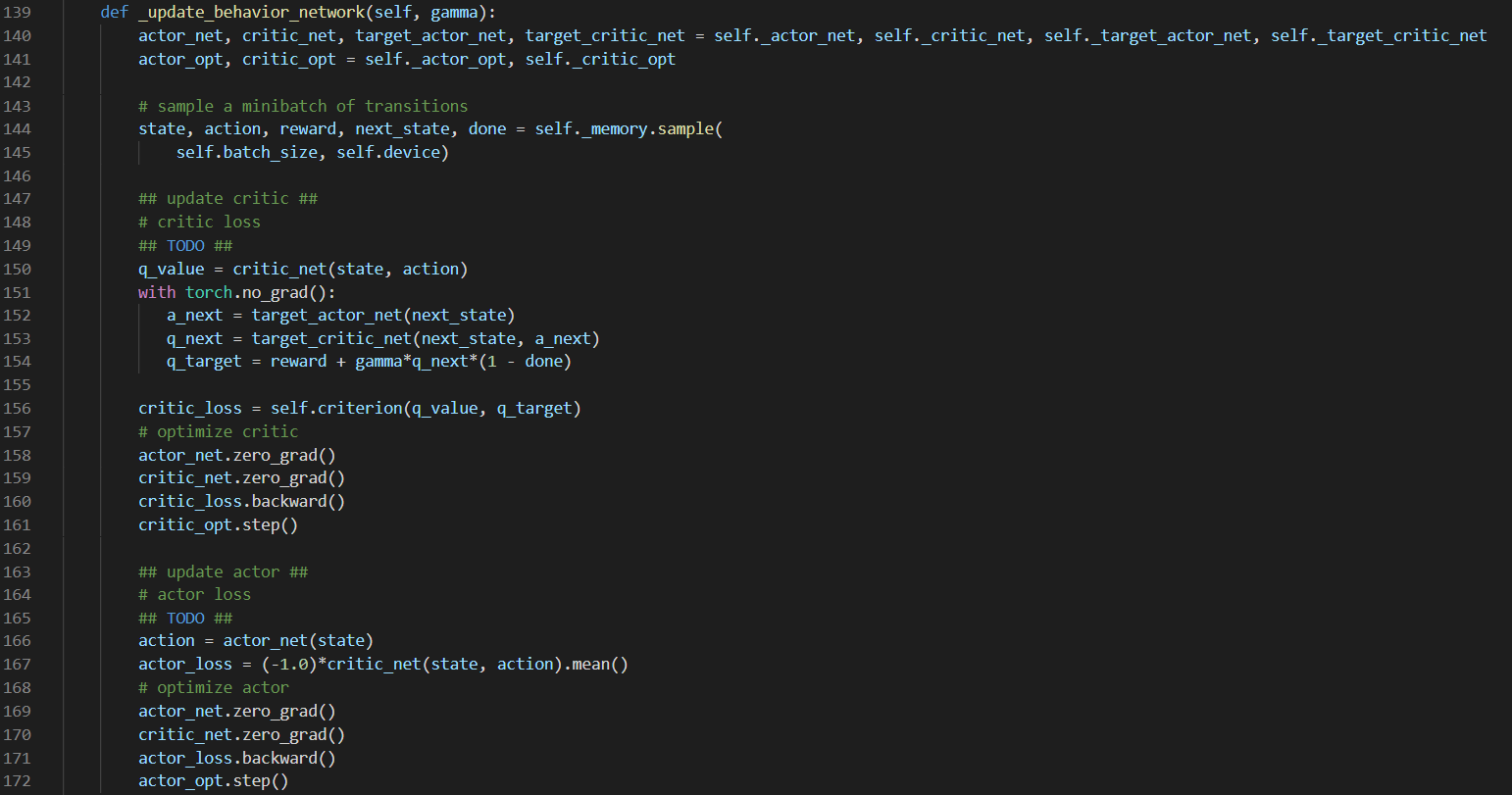
When training , generate action values and add the noise .



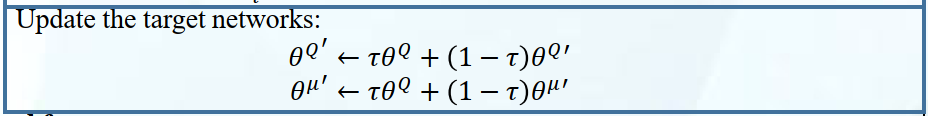


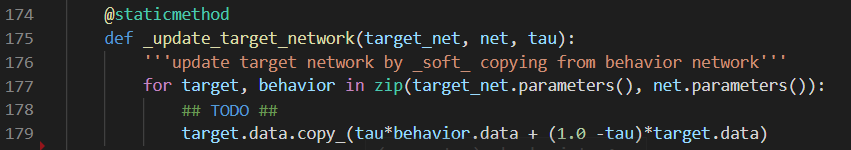
Update the behavior network with the algorithm :





Update the target networks .

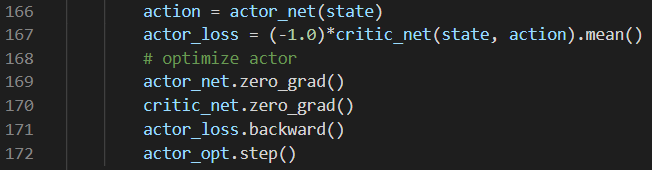


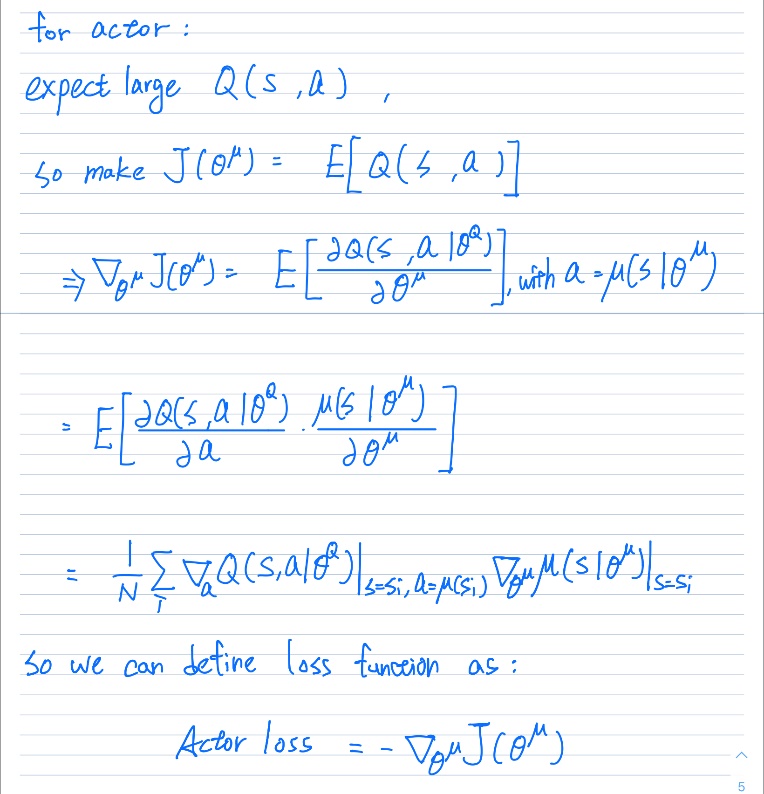


1. **Describe differences between your implementation and algorithms.**

When start training , in order to have enough sample for training , we don’t update the network within warmup step , just randomly choose action to play the game and store the result in the replay memory . And in the DQN , we update the target network every few iteration , it can reduce the correlation between target and behavior network .

1. **Describe your implementation and the gradient of actor updating.**

****

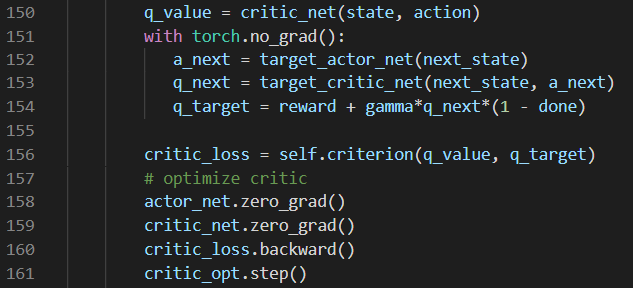


We only update the actor network .

1. **Describe your implementation and the gradient of critic updating.**

Compute the MSELoss with Q\_target from target network and Q\_value from behavior network .





1. **Explain effects of the discount factor.**

As the training step is longer , the sum of the reward will become infinity , so the discount factor can reduce the reward in the later step , in other words , it reduce the correlation of the later step . In this lab , only using the one next step , but we still give the 0.99 as discount factor .

1. **Explain benefits of epsilon-greedy in comparison to greedy action selection.**

The epsilon-greedy make additional opportunity to select the best action , because the action selected by the network may not be the best selection , and with the episode growing , we increase the epsilon . In conclusion , it can improve the exploratory of the network in the early step .

1. **Explain the necessity of the target network.**

We use the periodically update network (target network) to improve the stability of training .

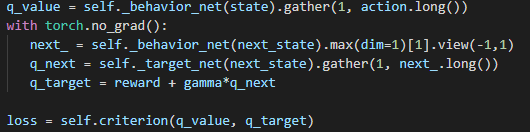
1. **Explain the effect of replay buffer size in case of too large or too small.**

The higher replay buffer size can make training more stable but need more time to train . And if too small , it is easier to occur overfitting .

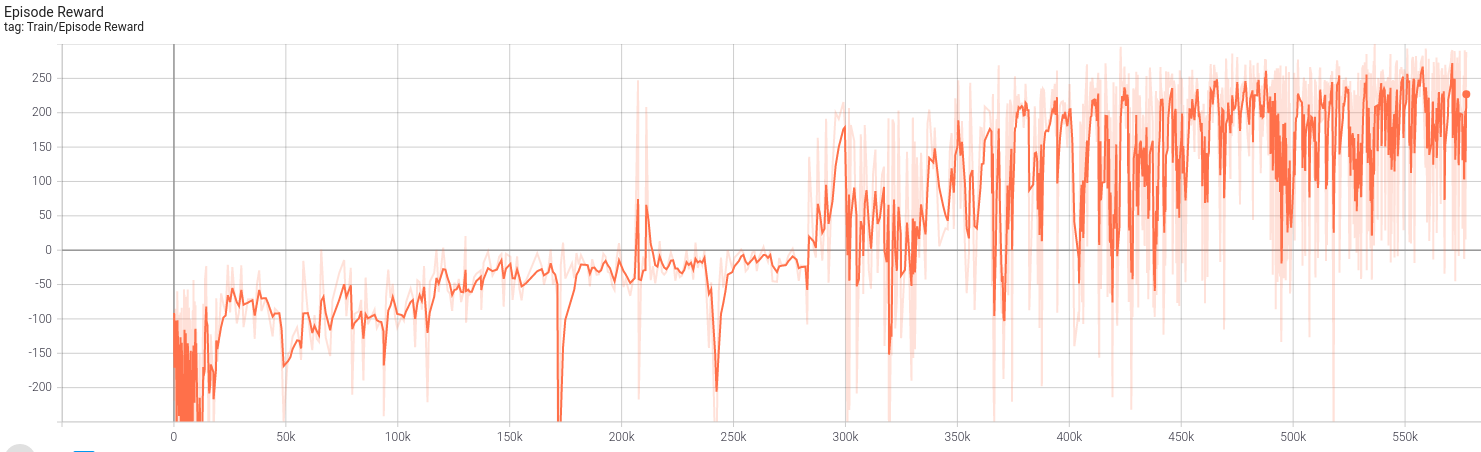
1. Bonus

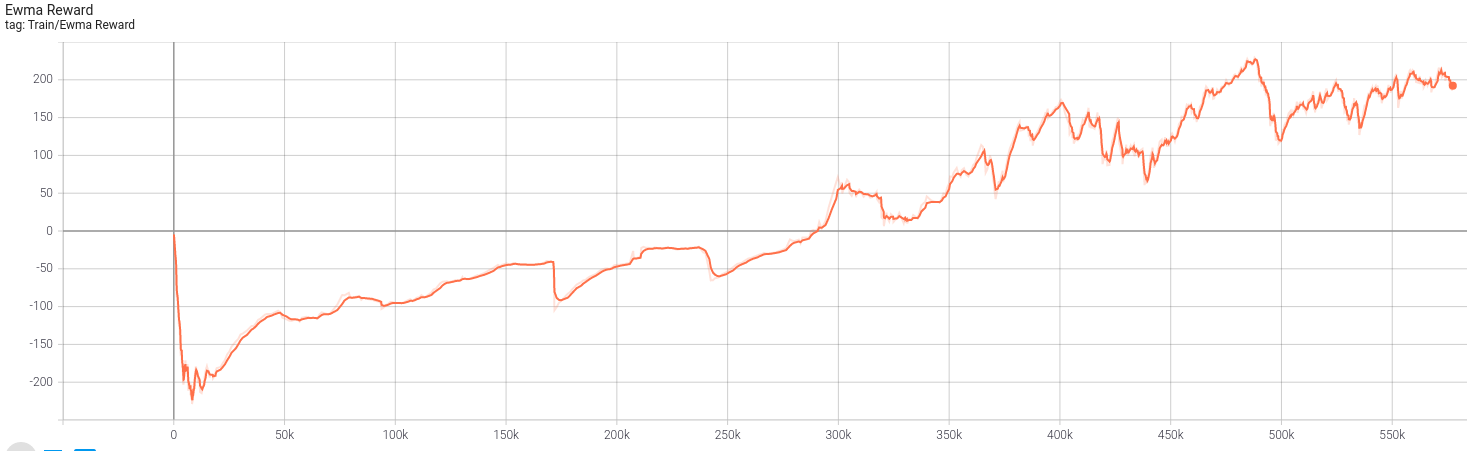
DDQN

The difference between DQN and DDQN is that in DDQN case , when compute the q\_target , it don’t use the max of Q’(si, ai ) , but use the index of the max value in Q(si, ai ) as the index of Q’(si, ai ) , that mean in DDQN , the action selection and evaluation are generate from the different function .



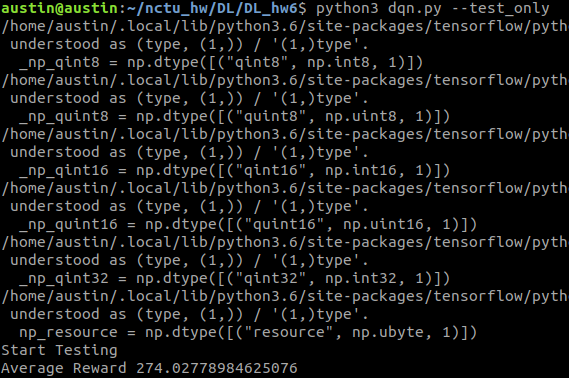
Tensorboard :



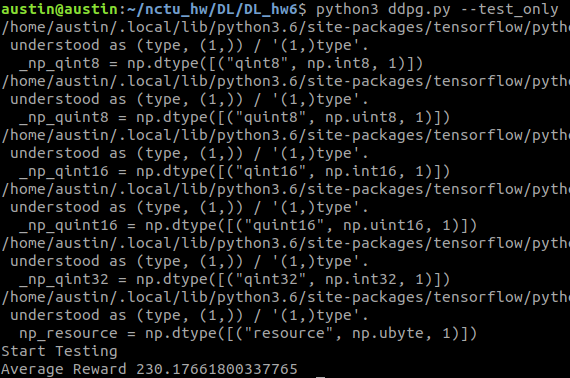


1. Performance ( Average reward of 10 games )

DQN ( train 1200 episode ) :



DDPG ( train 1500 episode ) :



DDQN ( train 1200 episode ) :

