

# Reinforcement Learning Assignment 4

Yifan Zhao (TA), Prof. Zou

2020-5-7

# 1 Introduction

The goal of this assignment is to do experiment with deep Q network (DQN), which combines the advantage of Q-learning and the neural network. In classical Q-learning methods, the action value function  $Q$  is intractable with the increase of state space and action space. DQN introduces the success of deep learning and has achieved a super-human level of play in atari games. Your goal is to implement the DQN algorithm and its improved algorithm and play with them in some classical RL control scenarios.

# 2 Deep Q-learning

**Algorithm 1: deep Q-learning with experience replay.**  
Initialize replay memory  $D$  to capacity  $N$   
Initialize action-value function  $Q$  with random weights  $\theta$   
Initialize target action-value function  $\hat{Q}$  with weights  $\theta^- = \theta$   
**For** episode = 1,  $M$  **do**  
  Initialize sequence  $s_1 = \{x_1\}$  and preprocessed sequence  $\phi_1 = \phi(s_1)$   
  **For**  $t = 1, T$  **do**  
    With probability  $\epsilon$  select a random action  $a_t$   
    otherwise select  $a_t = \operatorname{argmax}_a Q(\phi(s_t), a; \theta)$   
    Execute action  $a_t$  in emulator and observe reward  $r_t$  and image  $x_{t+1}$   
    Set  $s_{t+1} = s_t, a_t, x_{t+1}$  and preprocess  $\phi_{t+1} = \phi(s_{t+1})$   
    Store transition  $(\phi_t, a_t, r_t, \phi_{t+1})$  in  $D$   
    Sample random minibatch of transitions  $(\phi_j, a_j, r_j, \phi_{j+1})$  from  $D$   
    Set  $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$   
    Perform a gradient descent step on  $(y_j - Q(\phi_j, a_j; \theta))^2$  with respect to the network parameters  $\theta$   
    Every  $C$  steps reset  $\hat{Q} = Q$   
  **End For**  
**End For**

Figure 1: Deep Q-learning with experience replay

You can refer to the original paper for the details of the DQN. *“Human-level control through deep reinforcement learning.” Nature 518.7540 (2015): 529.*

# 3 Experiment Description

- Programming language: python3
- You should compare the performance of DQN and one kind of improved DQN and test them in a classical RL control environment–MountainCar. OPENAI gym provides this environment, which is implemented with python (<https://gym.openai.com/envs/MountainCar-v0/>). What’s more, gym also provides other more complex environment like atari games and mujoco.

Since the state is abstracted into car's position, convolutional layer is not necessary in our experiment. You can get started with OPENAI gym refer to this link (<https://gym.openai.com/docs/>). Note that it is suggested to implement your neural network on the Tensorflow or Pytorch.

## 4 Report and Submission

- Your report and source code should be compressed and named after “studentID+name+assignment4”.
- The files should be submitted on Canvas before May 15, 2020.