

Reinforcement Learning Homework Assignment

Handed out: November 8, Due: December 6, 11:59PM (KST)

1. Overview

In this project, you will solve the problem of "Mountain Car Climbing" using reinforcement learning.

2. Problem Description

You can find the environment in the following link: <https://gym.openai.com/envs/MountainCar-v0/>.

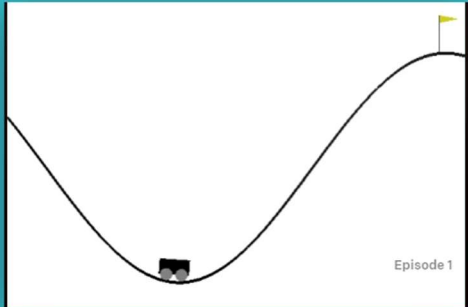
MountainCar-v0

A car is on a one-dimensional track, positioned between two "mountains". The goal is to drive up the mountain on the right; however, the car's engine is not strong enough to scale the mountain in a single pass. Therefore, the only way to succeed is to drive back and forth to build up momentum.

This problem was first described by Andrew Moore in his PhD thesis [Moore90].

[Moore90] A Moore, Efficient Memory-Based Learning for Robot Control, PhD thesis, University of Cambridge, 1990.

[VIEW SOURCE ON GITHUB](#)



The **states** are defined as follows.

Observation

Num	Observation	Min	Max
0	position	-1.2	0.6
1	velocity	-0.07	0.07

```
env = gym.make('MountainCar-v0')
env.observation_space.high # array([0.6 , 0.07], dtype=float32)
env.observation_space.low # array([-1.2 , -0.07], dtype=float32)
```

The **actions** are defined as follows.

Actions

Num	Action
0	push left
1	no push
2	push right

For each timestep, a **reward** of -1 is given. The maximum timestep is 200. The goal is to reach position **0.5** which is a terminating state.

3. Requirements

- (1) You must implement a model-free RL algorithm to train the agent.
- (2) If you train neural networks (such as in DQN or policy gradient), you may use Tensorflow/Keras, but Pytorch is recommended.

4. Submission

- (1) You need to submit your **source code**, and a **report document (hwp, docx, ppt, pdf)**. Combine all your files into a zip file named **20200001.zip**. The red numbers should be changed to your student ID. Submit your files on the cyber campus.

5. Report Document

The report document should contain the following contents.

- (1) A description of the algorithm you used.
- (2) The documented code. You should divide your code into blocks and explain what the block of code implements.
- (3) The result graph. You can draw a graph in which the X-axis is iteration and Y-axis is the reward.
- (4) Lessons learned while implementing and running the code.
- (5) You should explain how to run your code. That includes the command, and the necessary libraries that needs to be installed in your environment.
- (6) If you have used a code from the Internet as a reference, you must state that in your document and write its URL as well.

6. Evaluation Criteria

- (1) Implementation: 50%
- (2) Document: 50%

7. Notes

- You must write your own code. You may look up the websites that contain codes implemented by others, but you must not copy them.
- Our department uses a software that detects duplicate codes. Since the software uses an assembly code level detection, just changing the variable names will not bypass the software. Make sure you write your own code from the scratch. Then, you will have no problem.

8. Late Policy

No late submissions are accepted.