Reinforcement Learning

Assignment #03

SAPIENZA
UNIVERSITÀ DI ROMA

Info

- **Deadline**: Friday December 8th

- Students may discuss assignments, but the solutions must be typed and coded up **individually**

- Students must **indicate the names of colleagues they collaborated with**

Folder organization

- The assignment source code will be available on Classroom. You will find:
  - assignment3.pdf: with all the information
  - assignment3.zip that contains:
    - car_racing/
      - requirements.txt
      - main.py (do not touch!)
      - student.py

Theory submission

The theory solutions must be submitted in a pdf file named “XXXXXXX.pdf”, where XXXXXXX is your **student ID**.

We encourage you to type equations on an editor, rather than uploading scanned files.

Use the pdf file also to communicate the **students you collaborated with** and to **thoroughly explain your solution to the practical exercise**.

Code submission

The code solutions must be submitted in a zip file named “XXXXXXX.zip”, where XXXXXXX is your **student ID**.

The zip file must be **organized exactly as the original assignment.zip file**. Wrongly submitted assignments will be penalized.

Only edit the “students.py” files.

Theory

Suppose you have an environment with 2 possible actions and a 2-d state representation ($x(s) \in R^2$). Consider the 1-step Actor-Critic Algorithm with the following policy and action-state value function approximators:

$$\pi_\theta(a = 1|s) = \sigma(\theta^T x(s)) = \frac{1}{1 + e^{-(\theta^T x(s))}}$$

$$Q_w(s, a = 0) = w_0^T x(s)$$

$$Q_w(s, a = 1) = w_1^T x(s)$$

Given

$$w_0 = (0.8, 1)^T, w_1 = (0.4, 0)^T$$

$$\theta_0 = (1, 0.5)^T$$

$$\alpha_w = \alpha_\theta = \alpha = 0.1$$

$$\gamma = 0.9$$

and the following transition:

1. $x(s_0) = (1, 0)^T, a_0 = 0, r_1 = 0, x(s_1) = (0, 1)^T, a_1 = 1$

Compute new values of w_0, w_1 and θ after the transition.

Coding

Solve the **CarRacing-v2** Gymnasium environment using one of the following algorithms:

- Double DQN with proportional prioritization
- World Models
- Advantage Actor-Critic (A2C)
- TRPO
- PPO

Coding

In the folder “car_racing” you find three files:

- “main.py” that contains the main script to evaluate your solution. Don’t modify this file!
- “student.py” is the file you have to modify, by implementing the algorithm you have chosen.
- “requirements.txt” contains the name of the libraries needed for this part of the assignment.

Coding: additional libraries

You may add some requirements, but they need to be **authorized on Classroom** by writing a private comment under the assignment post with the name and version of the libraries you want to add.

Stable-baselines and any other library that already implements the algorithm are <u>not allowed</u>!

You need to use **PyTorch** as the deep learning framework.

Coding

The **grade will be assigned basing on the correctness** of the code.

3 additional points will be awarded to the 3 best students according to the **following criteria**:

- agent performance

- algorithm/implementation complexity