Individual Project / Exame

Q-learning policy for the cleaning robot

OEOD, Master in Data Science, Iscte - IUL

Diana A. Mendes

December 29, 2024

Deadline: 17 January, 2025

---

1 Project Goals

- For the cleaning robot problem, prove empirically (3-5 trials) that starting from any policy - the optimal policy will be attained (convergence).
- Try to discover some hyper-parameters values (3-5 trials) for which speeds up the Q-learning convergence.
- Please use (and improve, if necessary) the Python scripts (gridworld.py, qlearning.py) attached with the project

1.1 Introduction

Let suppose you notice that the cleaning robot bought last week does not follow an optimal policy while going back to the charging station. The robot is following a suboptimal path which is unsafe.

You want to find an optimal policy and propose an upgrade to the manufacturer. There is a problem, you do not have any access to the robot firmware. The robot is following its internal policy μ and this policy is inaccessible. What to do?

1.2 What to do?

What you can do is to use an off-policy algorithm like Q-learning to estimate an optimal policy.

- Create a discrete version of the room using the GridWorld class (3x4 grid).
- Get a camera and choose some markers to estimate the position of robot in the real world and relocate it in the grid world.
- At each time step you have the position of the robot and the reward.
- The camera is connected to your workstation and in the work-station is running the Q-learning Python script.
- The code is available in the Python script called qlearning.py.
- The script is based on the usual 3x4 gridworld, but can be easily extended to more complex scenarios.

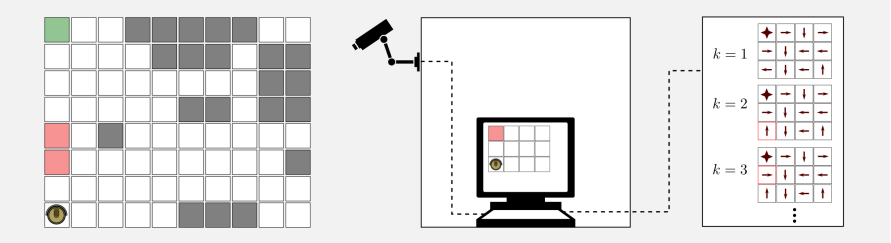

Figure 1: Gridworld for cleaning robot problem

Remember the Q-learning update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

where a_t is the action at step t, s_t is the state at step t, r_t is the reward, α is the learning rate and γ is the discount rate.

Running the script with $\alpha = 0.001$, $\gamma = 0.999$ and $\epsilon = 0.1$ the algorithm converged to the optimal policy in 300000 iterations. So, you got the optimal policy.

However there are two important limitations. First, the algorithm converged after 300000 iterations, meaning that you need 300000 episodes. Probably you have to monitor the robot for months in order to get all these episodes. In a deterministic environment you can estimate the policy through observation and then run the 300000 episodes in simulation.

However in environments which are non-deterministic you need to spend much more energies in order to find the motion model and it is generally very time consuming.

The second limitation is that the optimal policy is valid only for the current room setup. Changing the position of the charging station or the position of the obstacles the policy must be learned again.

Q-learning followed a policy which was sub-optimal and estimated the optimal policy starting from a random policy.

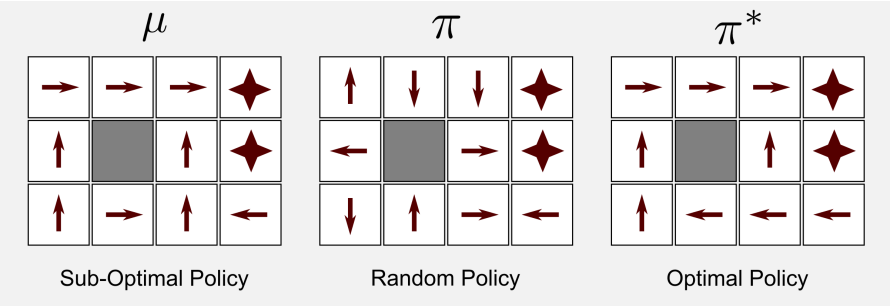

Figure 2: Policy type

So, do at least 3 trials and conclude about the Q-learning algorithm convergence (if attains an optimal solution and how long it takes).

1.3 Report and Scripts

- Write a report (less than 10 pages) where you should have: an introduction, the project problem, methodology, data, results and conclusions.
- Please do not collocate Python outputs in the report (except for figures).
- All secondary information can be written in the Appendix.
- The Python Notebook ou Script without errors, with explication comments, and able to run as-is.
- The report (in **pdf** format) and the Python file should be submitted to Moodle before the deadline (17 January, 2025)

1.4 Python Libraries

- numpy
- matplotlib
- script in the folder

2 Bibliography

[1]. Reinforcement learning: An introduction. Sutton, R. S., Barto, A. G. (1998). Cambridge: MIT press.