UNIVERSITY OF PADUA, DEPARTMENT OF INFORMATION ENGINEERING

- Neural Network and Deep Learning -

Reinforcement Learning

Giulia Bressan, ID 1206752

A.Y. 2019/2020

1 INTRODUCTION

The assignment of the fifth homework was to train an agent to move in a simple grid world (10x10 matrix) using a set of 5 basic actions (stay, move up, down, right, left). The goal was to find a target position in the grid and, in order to do so, some parameters were tuned, such as the ε and α parameters, and two different approaches were analysed: Sarsa and Q-learning.
The work is structured as follows:

1. First I created the environment in which the agent was moving,
2. Then I tuned some parameters to improve the learning,
3. Finally, I tested the results using both Sarsa and Q-learning.

2 THE ENVIRONMENT

The fist task was the creation of the environment in which the agent had to move in order to reach the target position. To the blank 10x10 grid, I added an obstacle to make the learning a bit more challenging: I created a barrier in the 5^{th} and 6^{th} row, such that the only way to overcome it was to pass through the free space in the middle. This was done by adding a penalty in the environment, in a way in which, if the action took the agent in a position occupied by the barrier, the reward assigned was equal to -1, the same as going out of the boundaries of the grid.
The resulting environment is shown in Figure 2.

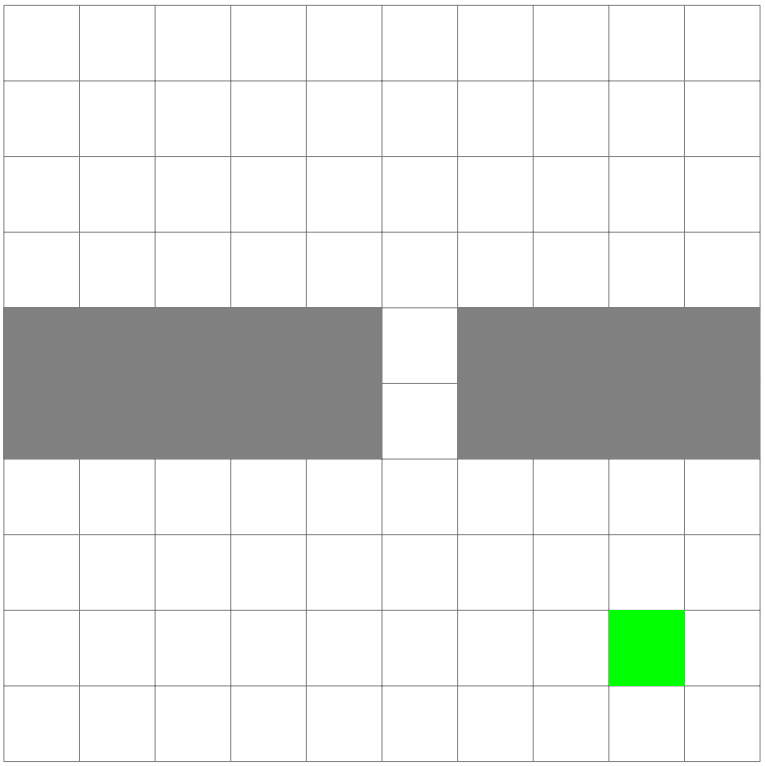

Figure 2.1: Resulting environment with the barrier and the goal highlighted in green.

3 Parameters tuning

I used two different algorithms to train the agent: Q-learning and Sarsa. For Sarsa I also tested two different policies: $\varepsilon - greedy$ and $softmax$. I trained it for 3000 episodes. Each of them had a maximum length of 50 steps: the agent had the possibility of moving in 4 directions (up, down, left, right) and staying in the same position, so once it reached the goal, it remained there until the maximum length was reached.
I fixed the discount factor to 0.9, a quite high value, in order to improve the foresightedness of the agent, even if the convergence was maybe a bit slower.
For the values of α and ε I implemented a very simple grid search, to find values for a better tuning. Then I used them for the training with the different algorithms and policies.
For the ε parameter (the parameter for the stochastic behaviour policies, $\varepsilon-greedy$ and $softmax$, used with Sarsa), I used a decreasing behaviour among the episodes, starting from 0.7 down to 0.001.
For the learning rate α, I tried both a fixed value and a decreasing behaviour. The performances were better with the decreasing behaviour for the Q-learning algorithm, so I used a starting value of 0.8 and a final value of 0.1. For Sarsa with the $softmax$ policy I used a decreasing behaviour too, but starting from 0.9 down to 0.2, while for Sarsa with $\varepsilon - greedy$ the best results were obtained by fixing the value to 0.2.

4 Results

The following figures show the pattern that the agent follows to reach the goal position (green), from four different fixed starting points (red).
We can see from the plots that Q-learning tries to go towards the direction of the goal from the

beginning, even if it means going near the barrier, so it takes more risks. This is due to the fact that Q-learning is an off-policy algorithm with a greedy update policy (ε was set to zero): it takes more risks in the actions, but it assumes that it will not make mistakes in the following ones.
On the other hand, the agent trained with Sarsa using $\varepsilon - greedy$, tries to stay more far away from the obstacle, when that is possible. When using $softmax$ the behaviour is more similar to the one of Q-learning, but not as accentuated. The policies used to update the estimates are stochastic in this case. It may happen, as in the case of the third path of Sarsa with $\varepsilon - greedy$ policy, that the choice of the next action results in a bad decision: the agent needs to go back on its steps to find the right path.
In all cases, even with these differences, the result is similar: the agent learns how to pass through the barrier in the only possible path and manages to reach the final target successfully.

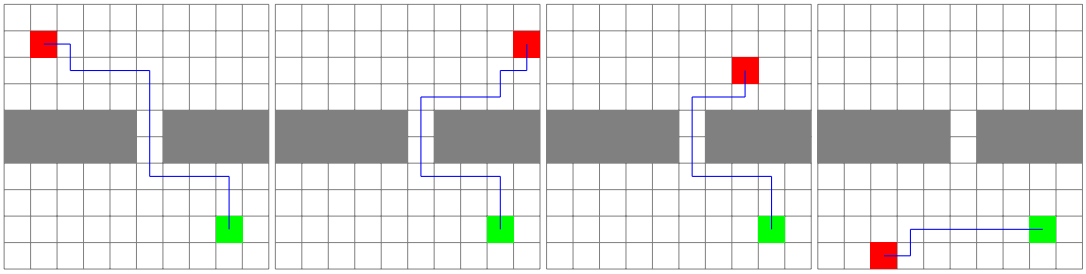

Figure 4.1: Examples of paths learned by the agent with Q-learning.

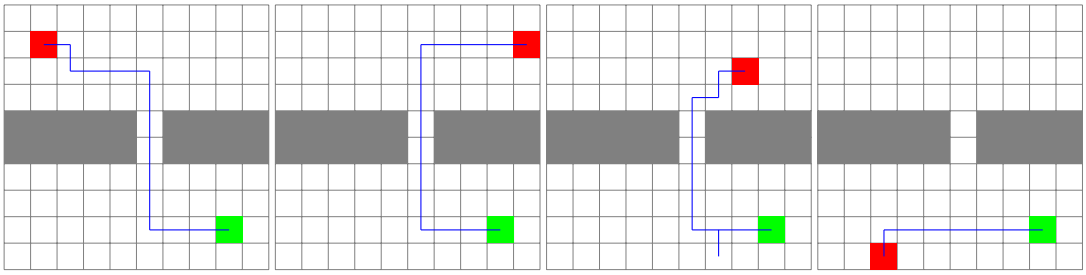

Figure 4.2: Examples of paths learned by the agent with Sarsa ($\varepsilon - greedy$).

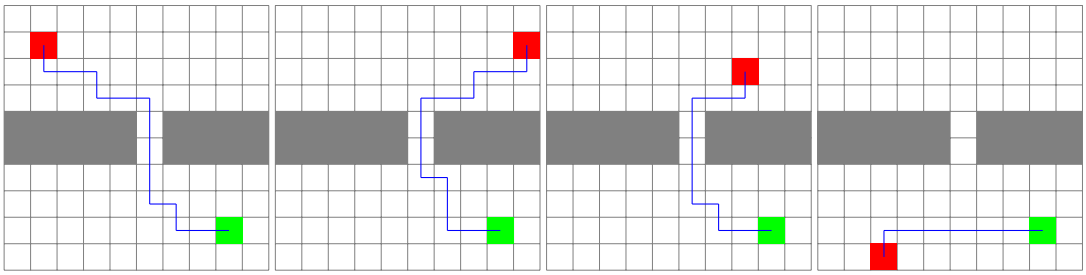

Figure 4.3: Examples of paths learned by the agent with Sarsa ($softmax$).