UNIVERSITA' DEGLI STUDI DI PADOVA
DEPARTMENT OF INFORMATION ENGINEERING

ICT for Internet and Multimedia

Neural Networks and Deep Learning Course

Homework 3: Deep Reinforcement Learning Report

Fatıma Rabia YAPICIOĞLU
2049536

23 December 2021

CONTENTS

Instructions and warnings about running the notebook

*If you run this notebook on the Colab please make sure that you download all the necessary libraries. I've written the code and run the notebook in my local environment, and as you can see from the outputs it compiles with all the necessary libraries successfully. Run the notebook in a vertical manner starting from the first cell to the end. Runtime takes nearly 55-60 mins. Each chapter in this report has been neatly designed as in the following structure: Introduction to the tasks that will be implemented, technical implementation details, and results concerning the experiments. For some very large tables, I only provided a preview on the report, you can see all of them in the *.ipynb. Thank you.*

1. Study Exploration Profiles & Speed-Up Convergence

1.1. Exploration & Exploitation: Epsilon Greedy and Softmax Strategies

The Deep RL algorithm's fundamental ideas are exploration and exploitation. It relates to how the agent chooses his or her activities. What is the difference between exploration and exploitation? Exploration occurs when you want to try new experiences, whereas exploitation occurs when you want to stay in your comfort zone and head straight to your best. The agent is in the same boat. It wants to investigate the surroundings first. It will make decisions based on exploitation rather than exploration as long as it interacts with the environment.

There are two approaches you can take:

- Greedy, in which the agent does a random action with probability, then explores the environment and chooses the greedy action with probability 1-, we have an exploitation situation.

- Soft-max, in which the agent chooses the best course of action based on the artificial neural network's Q-values.

1.2. Hyperparameter Tuning & Speed-up Model Convergence

In this section first I've completed and run the notebook which belongs to 7'th LAB. In the notebook, we implemented an agent at Cart Pole Gym. The system was controlled by applying a force of +1 or -1 to the cart. Observations we used were Cart Position, Cart Velocity, Pole Angle, Pole Angular Velocity; and the actions were pushing cart right or left. As for the network update, the hyperparameters we used were learning rate '1e-2' which is 0.7 nearly and 1000 episodes. At the end of the 1000'th episode, the final score of the agent was 500. Also, in the final test, we observed that the best score it printed was 500.

Therefore, I tried to reach the same accuracy with fewer episodes & a number of iterations. I tried to tune the hyperparameters of the model, for instance, increased the learning rate from 0.7 to 0.85, and changed target network update states from 10 to 5 in order to capture patterns in the original network better. Then I experimented with my agent training with 660 episodes and when we tested the final agent and it reached the score of 500. We may also change the structure of the neural network inside the DQN agent and use some regularization methods such as dropout and batch normalization. Also tried to change the gamma value to tweak the reward, increased it from 0.97 to 0.99 in the second part of the homework. Gamma is the discount factor. It quantifies how much importance we give for future rewards. It's also handy to approximate the noise in future rewards. You can see the convergence after the final test as in figure 1.0.

```
EPISODE 1 - FINAL SCORE: 53.0
EPISODE 2 - FINAL SCORE: 500.0
EPISODE 3 - FINAL SCORE: 30.0
EPISODE 4 - FINAL SCORE: 500.0
EPISODE 5 - FINAL SCORE: 30.0
EPISODE 6 - FINAL SCORE: 59.0
EPISODE 7 - FINAL SCORE: 500.0
EPISODE 8 - FINAL SCORE: 30.0
EPISODE 9 - FINAL SCORE: 500.0
EPISODE 10 - FINAL SCORE: 59.0
```

Figure 1.0

2. Acrobot-v1 Gym RL Agent Implementation

As the second homework in this homework, I tried to train a deep RL agent in a different Gym environment called 'Acrobot-v1'. The acrobot system includes two joints and two links, where the joint between the two links is actuated. Initially, the links are hanging downwards, and the goal is to swing the end of the lower link up to a given height as can be seen in figure 1.1. I used the same structure that we used the train the agent which is in the cart pole gym but it was harder to train and get a good score for the acrobot-v1 agent. The observation state is the current condition of the robotic arm.

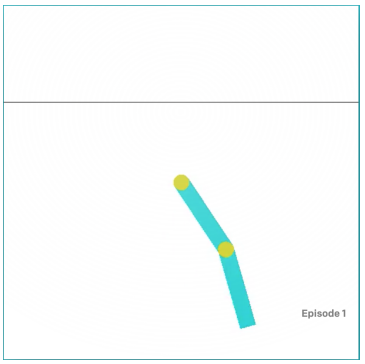

Figure 1.1

The state can be constituted by the screen pixels of the game or by some information about the agent. These lasts are:

- sin and cos of the two rotational joint angles (4 in total)
- the two angular velocities

In this example, I focused on the external information, instead of the current frame pixels. As for the actions, There are three possible actions:

- Apply positive torque (+1)
- Apply negative torque (-1)
- Do nothing (0)

Then, the agent's task is to understand which actions can maximize the cumulative reward. The environment imposes 500-time steps for each episode. Then, the worst cumulative reward is -500. Initially, the random agent was producing the worst cumulative reward which is -500. After episode 161, the agent started to get better scores in further episodes. I used gamma as 0.99, learning rate as 1e-3, target network update steps as 10, and batch size as 256. According to my observation, it requires more episodes than the cart-pole gym as we have more states in the observations. So, after a couple of experiments, I decided to set the number of iterations as 1000 and trained the agent. We can plot the accumulated episodic reward versus training episodes as in figure 1.2. We can easily observe that the accumulated episodic reward is increasing in time.

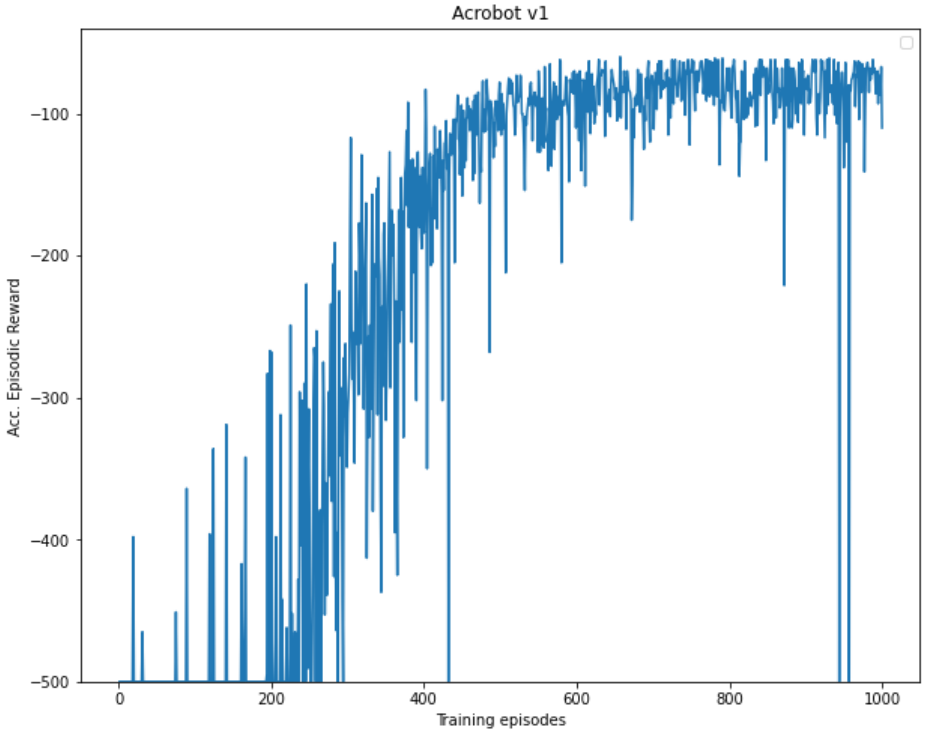

Figure 1.2

The final score may still be improved by further tuning the hyperparameters, but here I plotted some pictures which I extracted from the final videos that my agent has produced the following shots as a result.

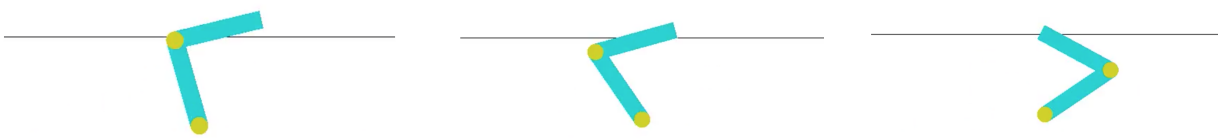

Figure 1.3

You can observe the final test score with 10 episodes as in Figure 1.4.

```
EPISODE 1 - FINAL SCORE: -93.0
EPISODE 2 - FINAL SCORE: -91.0
EPISODE 3 - FINAL SCORE: -68.0
EPISODE 4 - FINAL SCORE: -70.0
EPISODE 5 - FINAL SCORE: -69.0
EPISODE 6 - FINAL SCORE: -125.0
EPISODE 7 - FINAL SCORE: -135.0
EPISODE 8 - FINAL SCORE: -68.0
EPISODE 9 - FINAL SCORE: -70.0
EPISODE 10 - FINAL SCORE: -67.0
```

Figure 1.4.