Reinforcement Learning
Exercise 1

September 8, 2020

1 Introduction

In this first exercise, we will take a first look at a reinforcement learning environment, its components and modify the reward function of a simple agent.

2 States, observations and reward functions

The provided Python script (`cartpole.py`) instantiates a *Cartpole* environment and a RL agent that acts on it. The `agent.py` file contains the implementation of a simple reinforcement learning agent; for the sake of this exercise, you can assume it to be a black box (you don't need to understand how it works, although you are encouraged to study it in more detail). You don't have to edit that file to complete this exercise session - all changes should be done in `cartpole.py`.

2.1 Cartpole

The *Cartpole* environment consists of a cart and a pole mounted on top of it, as shown in Figure 1. The cart can move either to the left or to the right. The goal is to balance the pole in a vertical position in order to prevent it from falling down. The cart should also stay within limited distance from the center (trying to move outside screen boundaries is considered a failure).

The observation is a four element vector

$$o = \begin{pmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{pmatrix}, \tag{1}$$

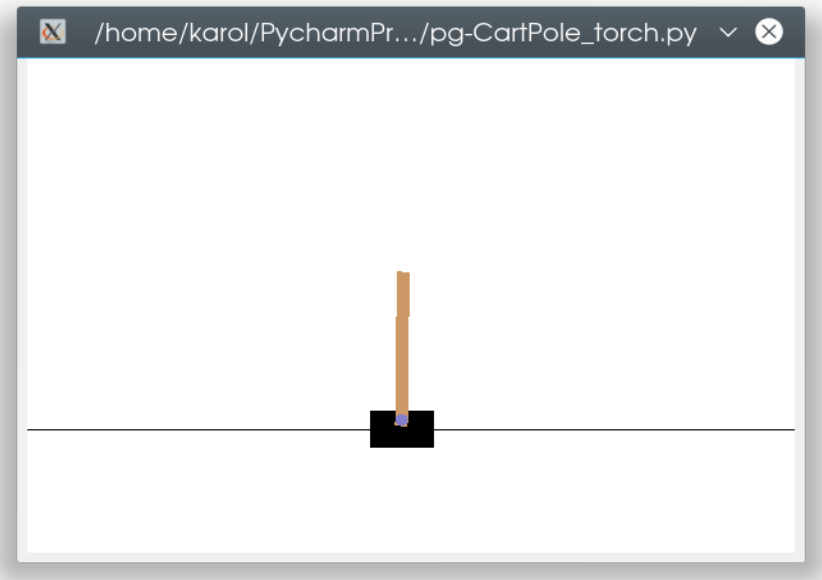

Figure 1: The Cartpole environment

where x is the position of the cart, $\dot{x}$ is its velocity, θ is the angle of the pole w.r.t. the vertical axis, and $\dot{\theta}$ is the angular velocity of the pole.

In the standard formulation, a reward of 1 is given for every timestep the pole remains balanced and the task is considered to be solved if the pole is balanced for 200 timesteps. Upon failing (the pole falls) or completing the task, an episode is finished.

2.2 Reward functions

Run the `cartpole.py` script. See how it learns to balance the pole after training for some amount of episodes. You can use the `--render_training` argument to visualize the training (it will run slower when rendering). When the training is finished, the models are saved to `CartPole-v0_params.ai` and can be tested by passing `--test path/to/model/file.ai`. You can use the `--render_test` argument here.

Task 1 - 5 points Train a model with 200 timesteps per episode. Then test the model for 500 timesteps by changing line 130 in `cartpole.py`. You can test the model with: `python cartpole.py -t MODELFILE --render_test`

Note: the episode length and the number of episodes given by the command line argument `train_episodes` are two different things. If you include plots in your submission, use the train episodes as their x-axis.

Question 1.1 — 15 points Can the same model, trained to balance for 200 timesteps, also balance the pole for 500 timesteps? Briefly justify your answer.

Task 2 - 10 points Repeat the experiment a few times, each time training the model from scratch with 200 timesteps and testing it for 500 timesteps. Evaluate its performance by using the `--render_test` argument

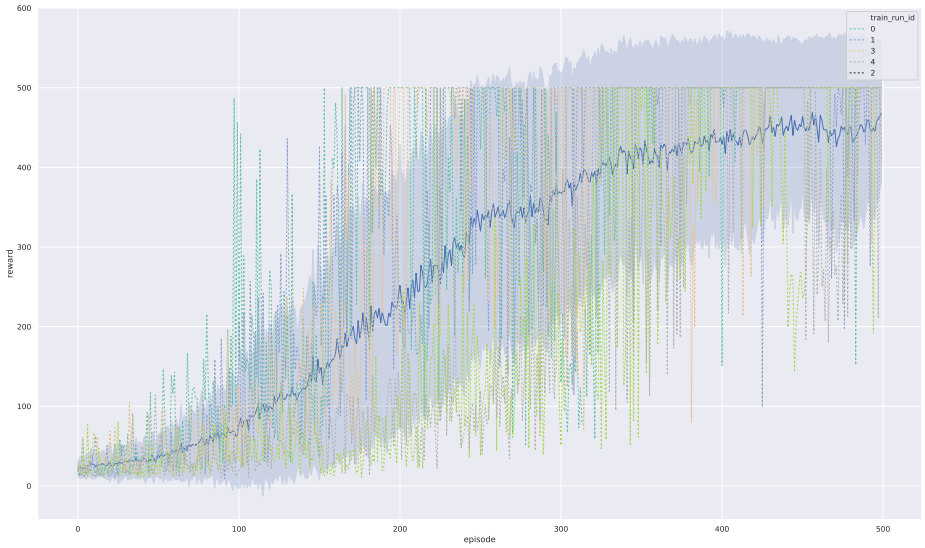

Figure 2: Variance in performance between multiple repetitions of the experiment.

and remember the average test rewards.

Question 1.2 — 15 points: Are the behavior and performance of the trained model the same every time? Analyze the causes briefly.

2.3 Repeatibility

Figure 2 shows the mean and standard deviation throughout 100 independent training procedures. You can notice that there is a large variance between the runs of the script.

Question 2 — 10 points: Why is this the case? What are the implications of this stochasticity, when it comes to comparing reinforcement learning algorithms to each other? Please explain.

You can generate a similar plot by running the `multiple_cartpoles.py` script. If the script is slow, try disabling PyTorch multithreading (by running the script with `OMP_NUM_THREADS=1`)

2.3.1 Reward functions

Let us design a custom reward function and use it instead of the default reward returned by the environment (see the `new_reward` function for an example).

Task 2 — 30 points: Write different reward functions to incentivise the agent to learn the following behaviors and include them in your report:

1. Balance the pole close to the center of the screen (close to $x = 0$),
2. Balance the pole in an arbitrary point of the screen ($x = x_0$), with x_0 passed as a command line argument to the script,

3. Keep the cart moving from the leftmost to rightmost side of the track as fast as possible, while still balancing the pole. The minimum speed of the agent should be high enough that the agent is visibly moving from one side of the other (not just jittering in the center) and changes direction at least once and not just drifting in a random direction.

Train a model with each reward function and include them in your submission.

Hint: Use the observation vector to get the quantities required to compute the new reward (such as the speed and position of the cart). If you feel like your model needs more time to train, you can leave it running for longer with the command line argument `--train_episodes number`, where `number` is the amount of episodes the model will be trained for (default is 500).

Task 3 — 5 points: Let us visualize the third reward function. Train the model at least 10 times using `multiple_cartpoles.py` (adjust the `--num_runs` and `--train_episodes` parameters). and include the performance plots similar to Fig. 2.

Question 3 — 10 points: Observe the agent using `python cartpole.py −t MODELFILE −−render_test` and keep track of the observations. What do you observe? What is the highest velocity the agent reaches?

3 Submitting

The deadline to submit the solutions through MyCourses is on Monday, 21.09 at 12:00. Example solutions will be presented during exercise sessions the following week (30.9 and 2.10).

Your submission should consist of (1) a **PDF report** containing **answers to the questions** asked in these instructions and **reward plots**, (2) **the code** with solutions used for the exercise and (3) the **trained model files** for each reward function (remember to report what reward functions were used). Please remember that not submitting a PDF report following the **Latex template** provided by us will lead to subtraction of points.

For more formatting guidelines and general tips please refer to the submission instructions file on mycourses.

If you need help or clarification solving the exercises, you are welcome to come to the exercise sessions in weeks 37/38.