Reinforcement Deep Learning

Edoardo Antonaci 1234431
edoardo.antonaci@studenti.unipd.it

Abstract

In this work we solve reinforcement deep learning problems using neural networks(NN) models. In particular we apply two different NN architectures in order to face two gym games : Cartpole and Mountain Car. The results are very satisfactory if we update the state using the "step" function but ,instead ,if we train the models using the episode's images the results get much worse.

1 Introduction

We show our attempts to solve two common problems-game: the Cartpole and Mountain Car. A game's episode is considered finished if specific conditions are satisfied: parameters in close ranges,episode length,threshold score and so on. Each game has two main spaces : the observation space and the action one.

In the first one we define the state's variables and in the other we have the available actions. We apply also two different strategies: the first one (the "*getting states*")is strictly forward and it consists by getting the next status using the proper "environment.step" function ;the other("*getting images*") is more complex and consists by obtaining the episode's images trying to extract the useful informations ,such as velocity and position , in order to resolve the game.

2 Methods

2.1 Cartpole

The first game is called Cartpole and it is a mobile platform with a rod that can rotate around a pole. A state is defined as 4 variables : platform's position and speed,rod's angle and rod's angular speed. The possible actions can be : 0,the cart moves to left,1, the cart moves to right. The choice between them is done using the softmax function giving also a temperature τ as input parameter. When τ goes to 0 choosen action is the "best one" that is the network output's $argmax$. The τ parameter follows the "exploration profile" distribution(figure 1) with $initial$ value = 5,$iterations\ numbers$ = 500, and $decay\ parameter$ = 6

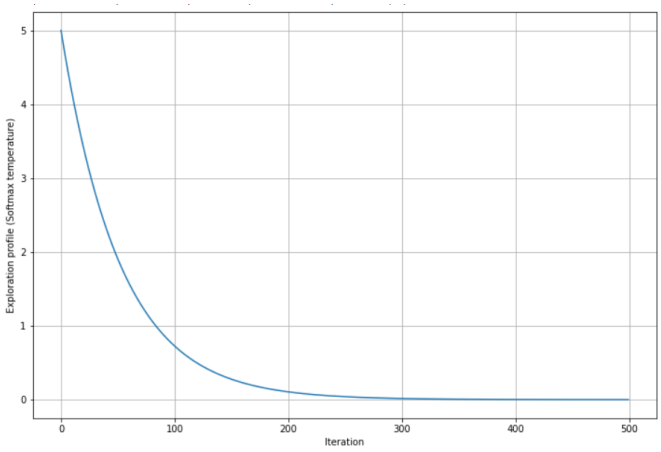

Figure 1: cartpole exploration profile

Max and min values in observation space are:

- cart positions must be between -4.8 and +4.8
- Pole angle must be between -24° and +24°

But the game is consider finished if :

- Pole angle is ± 12°
- Car position is more then ± 2.4(the center reaches screen bounds)
- episode's length is greater or equal to 500 (game resolved)

To help the convergence before than 1000 episodes we little modify the reward equation using a sort of "weight" parameter w :

$$w = 1.2$$

$$reward = reward - w * |x|$$

where x stands for the cart center.

The model used is a simply linear network with three layers interspersed by hyperbolic tangent as activation function and 128 neurons in main body.

As parameter we choose the best found after many attempts:

γ	replay memory capacity	Learning rate	Optimizer	Batch size
0.99	10^4	$7 * 10^{-2}$	SGD	100

Table 1: Optimal parameters to Cartpole architecture .

where γ stands for the parameter used by the long-term reward and the *replay memory capacity* stands for the number of tuples can be stored in memory.

As said in previous section we tried also to solve the game *without* getting status but *only* using episode's images.The first think done was to crop the images taking only 150 x 600 pixels per photos and to reduce the channels from 3 to 1.

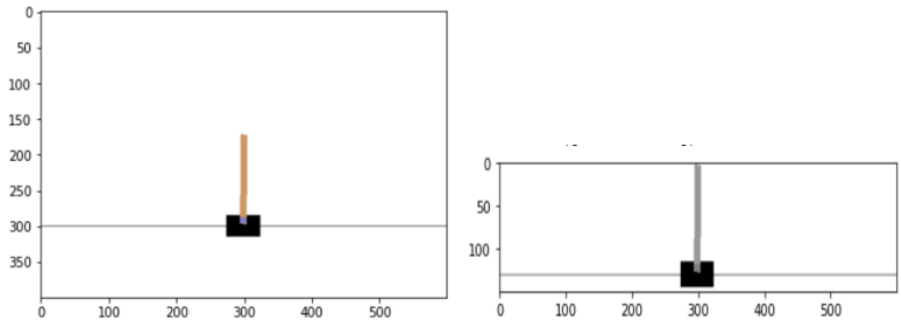

Figure 2: before and after crop

Then we modify the linear architecture used before adding one convolutional layer (with 4 channels in output) at the beginning and then we apply different numbers of hidden neurons in the body network. Parameters and reward function are the same as before.

2.2 Mountain Car

The second game is the "Mountain Car" and it consists by a mobile car that tries to climb the hill starting from the valley. The problem is not trivial because the car hasn't enough strength to mountain up and so it needs to gain momentum to reach the top.

The scenario in composed by two ups and one down where the car is placed. The unique possibility in order to achieve this aim is "to swing" around the minimum till the momentum is strong enough to overcome the right up and reach the top.

As before we have the observation space and the action space. For the first one we have 2 variables : car position and car speed. For the second ,instead,we have 3 variables : 0, means to accelerate to the left,1,doesn't accelerate ,2,accelerate to the right.

As before we choose only one action following a new "exploration profile" for τ parameter where the number of iterations are fixed to 100.

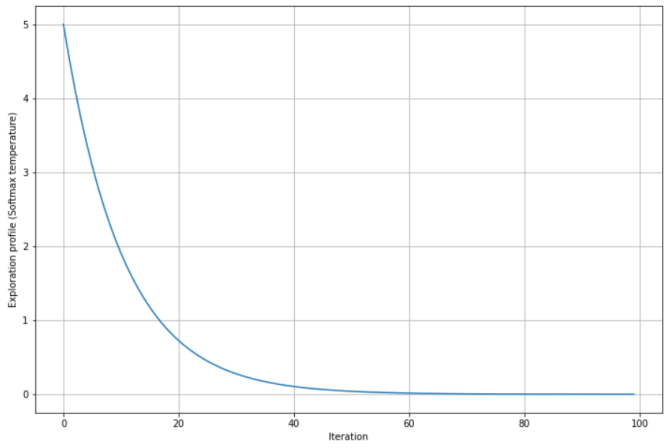

Figure 3: mountain exploration profile

The Max and min values for the game are:

- car positions must be between -1.2 and +0.6
- car velocity must be between $\pm$ 0.07

And the game is considered finished if :

- car position is more then 0.5(game resolved)
- episode's length is greater or equal to 200

As before we apply some changes to the reward function and in particular we have :

```
pos_weight = 4

if (action ==0 and state[1]<0) or (action==2 and state[1]>0):
  reward= reward + pos_weight * np.abs(state[0])
else:reward= reward
```

Figure 4: reward mountain car

where $state[0]$ is the car position and $state[1]$ its speed. The NN model tested for this task is the linear one applied also for the cartpole problem.

3 Results

3.1 Cartpole

In the following we show the results obtained with two strategies.

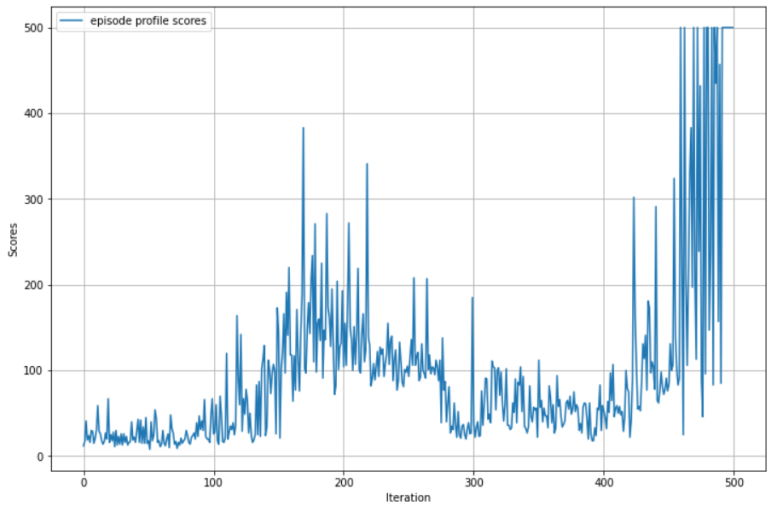

Figure 5: score profile getting states

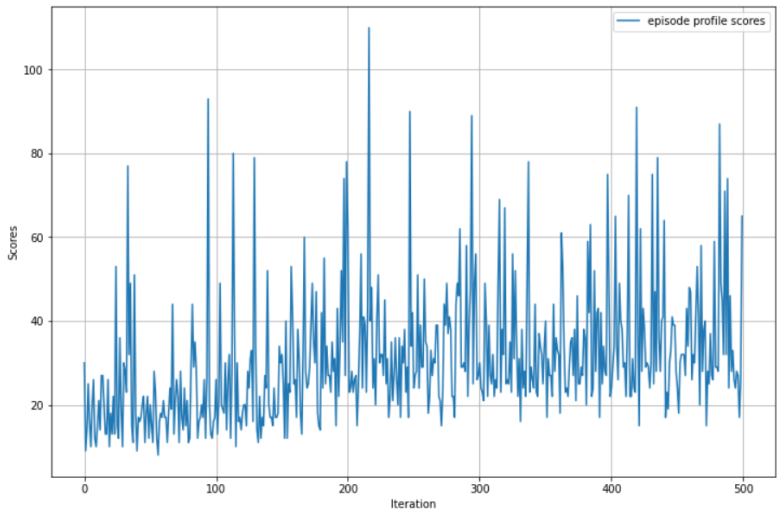

Figure 6: score profile getting still images

The figure 5 is refereed to the "*getting states*" strategy and when we apply linear NN model.Next we can see in figure 6 the application of " *getting images*" strategy with convolutional model. As shown above the results change drastically : for the first case we reaching 500 score and so we successfully resolve the game ,on the contrary,the second case's score swings around 40 .

Even if we tried more convolutional sophisticated model and also different combinations of parameters we get very bad results . So we deduce that ,probably , solving the game using only the images is a very hard task and indeed it is reasonable if we consider that the network should estimate changing quantities such car's speed or angular's speed using still images.

3.2 Mountain Car

Now,as before,we show the results for Mountain Car game:

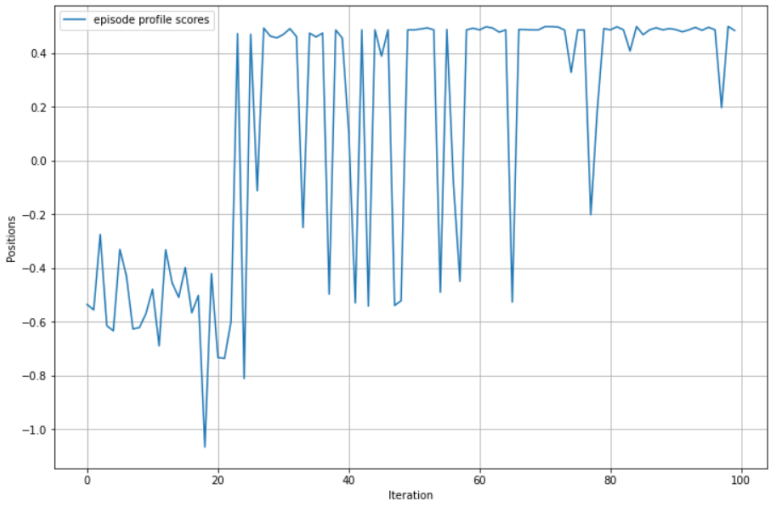

Figure 7: score profile mountain car

The results are vary good and also on 10 episodes *test* we have 0.6 as final positions (figure 8)

```
EPISODE 1 - FINAL SCORE: 0.6
EPISODE 2 - FINAL SCORE: 0.6
EPISODE 3 - FINAL SCORE: 0.6
EPISODE 4 - FINAL SCORE: 0.6
EPISODE 5 - FINAL SCORE: 0.6
EPISODE 6 - FINAL SCORE: 0.6
EPISODE 7 - FINAL SCORE: 0.6
EPISODE 8 - FINAL SCORE: 0.6
EPISODE 9 - FINAL SCORE: 0.6
EPISODE 10 - FINAL SCORE: 0.6
```

Figure 8: test mountain car

4 Appendix

In the folder we upload also animated gif :

- cartpole states : gif for cartpole problem with "*getting state*" strategy,
- cartpole images : gif for cartpole problem with "*getting images*" strategy
- mountain car : gif for mountain car problem