# Udacity - Project Navigation

**Student's name:** Paulo Cotta


### Abstract

In this paper, Deep Q-Learning (DQN) of various formats was tested, to meet the best format, sometimes reaching the level of zeroing the notebook to have no remnants in memory. At 271 epoch DQN reached its goal 13.03, the value was> = 13.0 in the score window generated by DQN. The configuration used:

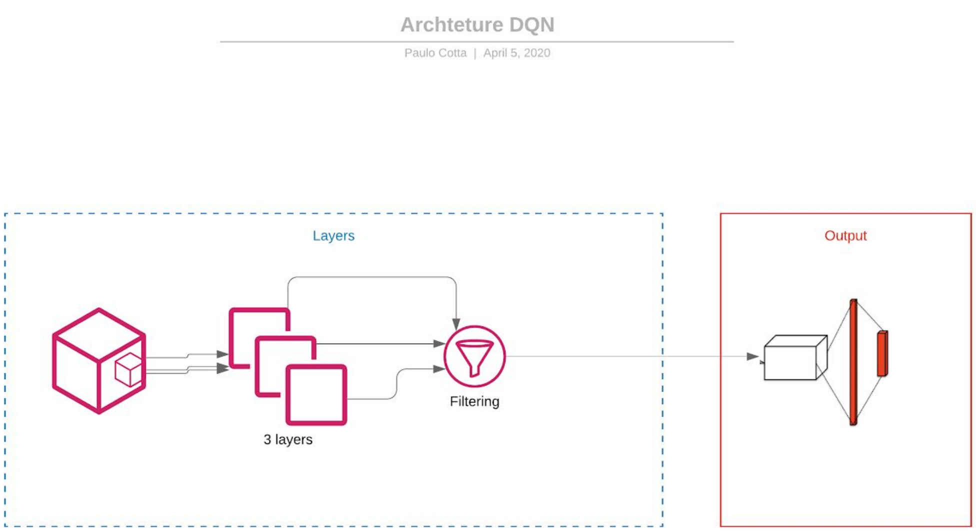

| n_epoch_max | eps_end | eps_decay |
| --- | --- | --- |
| 700 | .008 | .9277 |

### Object Model

The goal is to learn the techniques that were passed in the Deep Reinforcement Learning course and put them into practice in deliverable projects.

### Model Architecture

After experimenting with different numbers of layers (number of layers used [2, 3, 4]), it was found that three layers make more sense because the advance of the activation pattern with ReLu at the time was a good compromise to reach the goal. With dimension 37 of the state space and dimension 4 of the (exit / action) space, the problem is not very dimensional, so a very high number of layers or hidden units in the layers does not seem to be justified and does not make epoch for achieve the goal.




As described in the classes and exercises, the choice of ε (initial value, decay / speed factor, final value) has a great effect on the learning speed. It was decided to opt for a multiplicative "decay" with minimum value in the long run: $\varepsilon = \max(\varepsilon_0 \cdot \varepsilon^{k}_{decay}, \varepsilon_{min})$, where k denotes the epochs. After starting with relatively conservative and previously seen $\varepsilon_{decay}$ values (for example, 0.99), it decreased more and more and I observed a very fast training progress in the case the .9277 value.

My interpretation is that the environment is not very complex (that is, it does not generate much variation in the state space) and, therefore, the agent needs relatively little exploration compared to other environments.

In this work, the agent_model_dqn.py class was used, where it has 3 classes (Agent, QNetwork, ReplayBuffer).

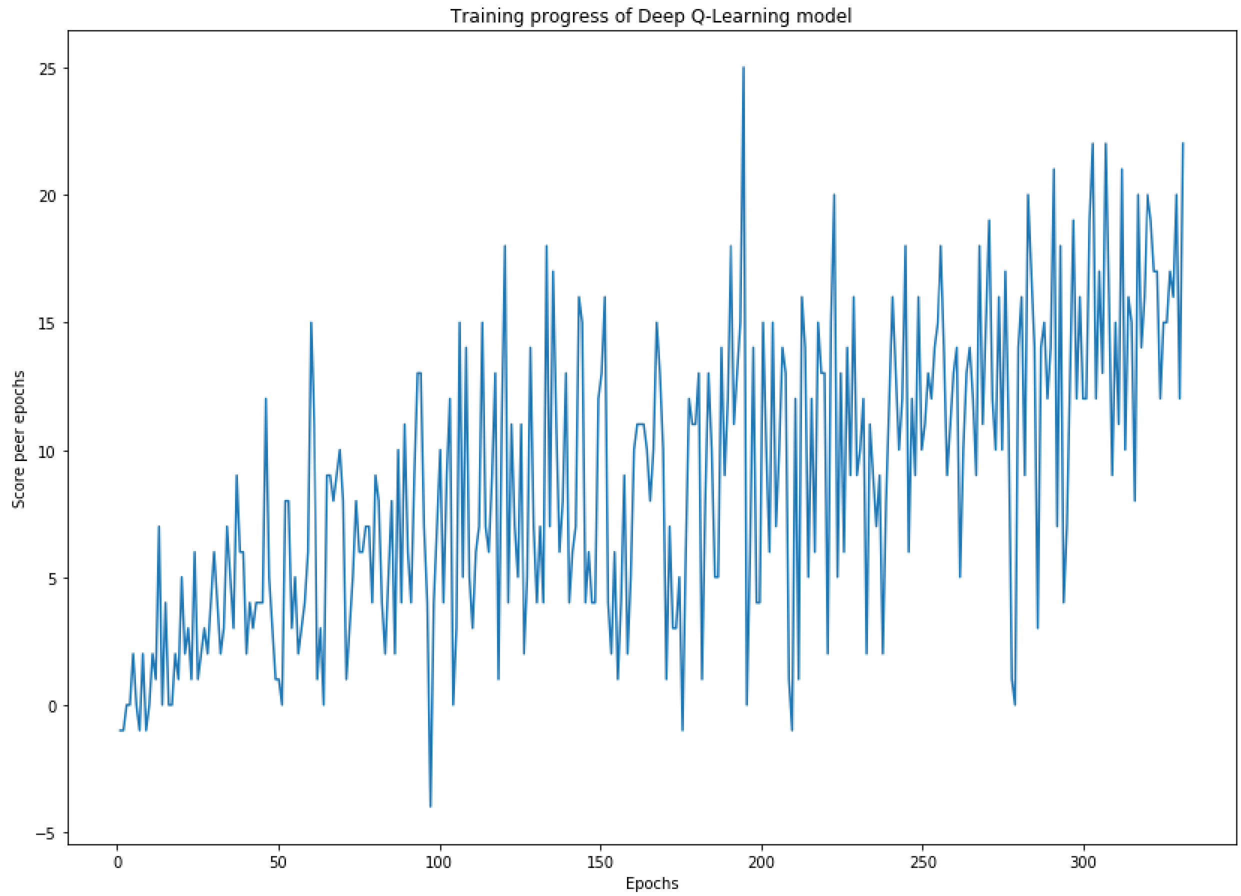
**Final result - Conclusion**

We reached 271 epochs to really get to where goal 13.0 was, 700 epochs were placed, which suggests that < that 700 epochs we managed to reach the goal and demonstrates that the problem is not so complex. In this case, we have the results of training with the final configuration.

| n_epoch_max | eps_end | eps_decay |
|---|---|---|
| 700 | .008 | .9277 |

After an initial phase with little progress (due to the high participation in the exploration), the agent begins to obtain higher scores after approximately 10 epochs. After about 179 epochs == 10.03 and 100 epochs had already parked the episilon in 0.00800, progress seems more flattened, but the average score still increases until an average score of 13.0 is reached. Result the problem is not of high complexity for the agent.

| | | |
|---|---|---|
| Epoch 50 | AVG Score: 3.04 | epsilon: 0.02346 |
| Epoch 100 | AVG Score: 4.88 | epsilon: 0.00800 |
| Epoch 150 | AVG Score: 8.49 | epsilon: 0.00800 |
| Epoch 200 | AVG Score: 11.07 | epsilon: 0.00800 |
| Epoch 250 | AVG Score: 12.49 | epsilon: 0.00800 |
| Epoch 271 | AVG Score: 13.03 | |
| Environment saved in 271 episodes! | AVG Score: 13.03 | |




### References


https://medium.com/@awjuliani/simple-reinforcement-learning-with-tensorflow-part-4-deep-q-networks-and-beyond-8438a3e2b8df

Book - Reinforcement Learning, Sutton