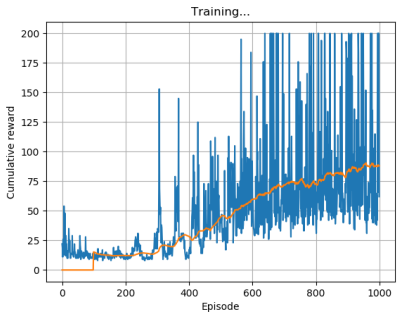
Reinforcement Learning Exercise 4

Franceschini Riccardo 769697

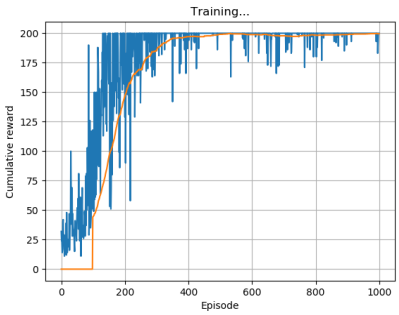
October 2019

1 T-1

The two different features representations were implemented. In order to implement the handcrafted version, the state and his absolute value were concatenated and used as a feature. As it is possible to see using handcrafted features the agent is learning but very slowly and is not able to reach good performances in 1000 episodes. On the other hand, using the RBF features after only 500 episodes the agent is able to solve the task and balance the cart pole for 200 timesteps.



(a) Single update with handcrafted features



(b) Single update with RBF

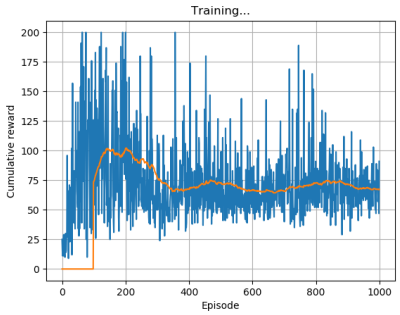
2 Q-1

a)Passing directly the state to a linear regressor, it wouldn't be possible to learn the Cart Pole because it is not a linear problem. For this reason, it is not possible to learn it simply using the sate as a feature representation.

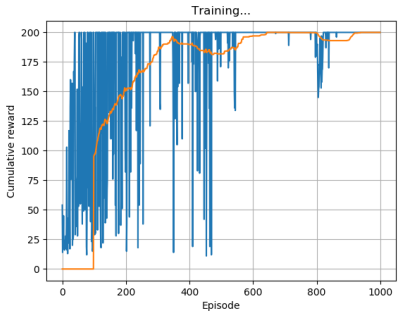
3 T-2

The experience replay was implemented storing the values of the states in the memory and sampling from there during the updating part. As it is possible

to see from the figures below, using experience replay the agent is able to reach immediately the maximum in each feature representation compared to the plots from the task 1.



(a) Replay handcrafted features



(b) Replay RBF features

4 Q-2

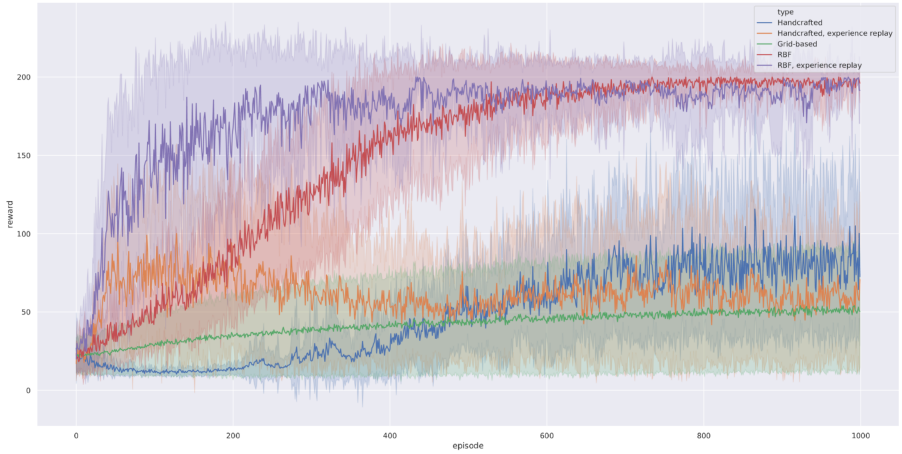


Figure 3: training process with different methods

4.1 Q-2.1

As it is possible to see from the plots, the RBF features and the handcrafted features with experience replay are the most samples efficient because they are able to learn and increase the performance of the agent in a shorter amount of time. This happens because they are trained using experience replay, so, the agent has trained with 32 different samples each time from the memory and this allows the agent to learn faster.

4.2 Q-2.2

The efficiency of the handcrafted features could be improved adding a non-linearity factor, for example, a square. This happens because the system is not linear.

4.3 Q-2.3

In this case, the grid-based system is more sample efficient than the handcrafted feature representation without replay. This happens because despite the fact that the grid-based system is still limited in learning the Q function due to the high dimensionality space, is still able to learn his maximum faster than the handcrafted. However, in the long run, the handcrafted version outperforms the grid-based system.

4.4 Q-2.4

As we can see from fig.3 the feature representation is a relevant parameter, in fact, if a problem is non-linear is not possible to learn a policy using linear features for this reason the feature representation is relevant. Another parameter that impact the sample efficiency is how we train the network, for example, if the network is trained simply using the immediate action instead for example of using the experience the performances are affected significantly and also the dimension of the batch is a relevant parameter for increasing the efficiency.

5 T3

As it is possible to see from the fig.4, the plot of the policy was generated picking the best action for each state after training the policy using RBF features and experience replay. In order to obtain the following plot, the x and θ are discretized in 100 different steps and $\dot{x}$ and $\dot{\theta}$ are setted to zero. Thus, the state is :

$$s = [x, 0, \theta, 0] \tag{1}$$

Iterating through the 100 different values and collecting the best action we are able to generate the following plot.

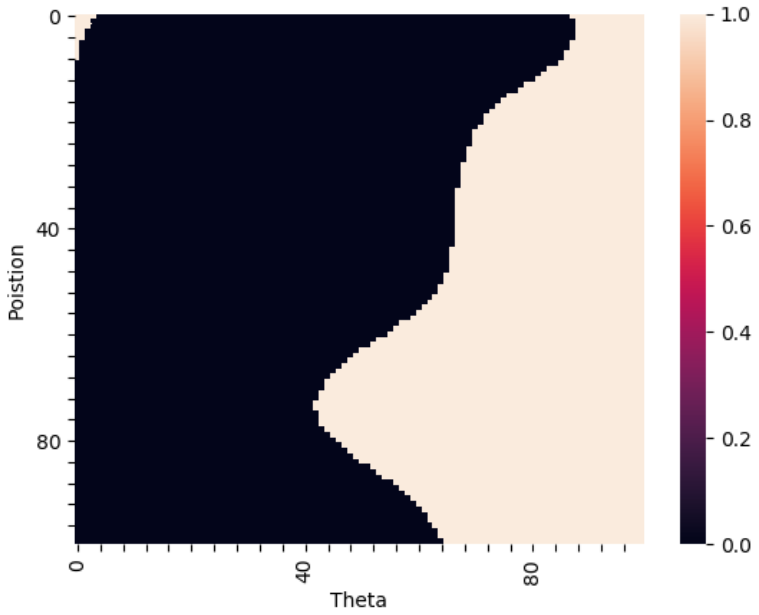
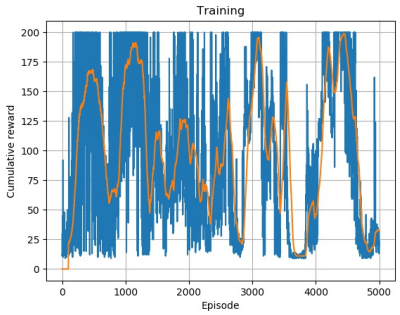


Figure 4: action plot

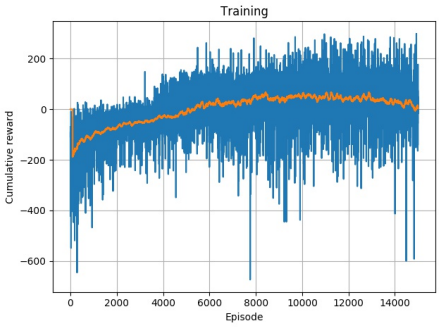
It is interesting to notice that the agent tries to move the Cart Pole to the right (right =pink) in order to balance the bar when the angle theta is greater than a certain threshold.

6 T4

The DQN agent was implemented and after that, the agent was trained for 5000 episodes for the Cart Pole and for 15000 for the Lunar Lander. As it is possible to see from fig.5a the network is able to learn the policy and reach the 200 timesteps in some cases, however, the performances are not stable. The Lunar Lander agent is able to improve his performance and reach a positive reward, however also in this case as we can see from fig.5b there is a lot of variance on each episode and this is due to the high complexity of the problem.



(a) Training Cart Pole



(b) Training Lunar Lander

7 Q3.1

Q-Learning cannot be used directly in a continuous action space because in Q-Learning the value of each pair action state (Q(s, a)) has to be computed and stored in the grid, using continuous action space obviously this approach is not feasible because the dimension of the grid will become infinite. For this reason, Q-Learning cannot be directly used in continuous action space.

8 Q3.2

In case of a continuous action space picking the maximum between the different actions could be a problem because the possibility of action is infinite. However, it could be possible to use regression with a neural to network in order to obtain an approximation in the continuous space.