RL Lab 2

Kildo Alkas Alias 971106-7430
Andrej Wilczek 880707-7477

November 2020

1 Problem 1

1.1 a)

We are familiar with the code.

1.2 b)

Why do we use a replay buffer and target network in DQN? In order to use SGD optimization we require that the data samples are i.i.d. which is not true for our case since samples are highly correlated and not distributed identically to the optimal policy. To address this we implement a replay buffer which gives us pseudo independent data. The buffer also allows us to learn from experiences multiple times.

The reason behind the target network is to make training more stable. In order to make the target more stationary the target network is only updated every C-steps therefore avoiding the "chasing our own tail" problem.

1.3 c)

The problem was solved and verified using DQN_check_solution.py
The reward was over 50 points on average as can be seen in Figure 1

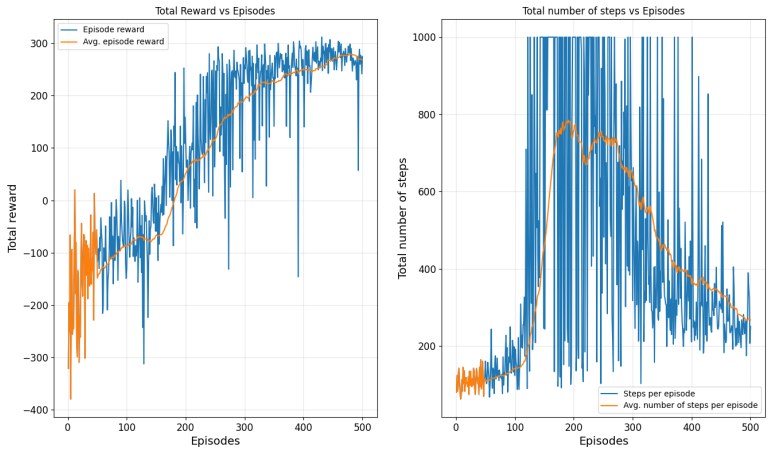

Figure 1: Reward and number of steps for the solved problem with parameters as stated in section d).

1.4 d)

Network: A fully connected network with one hidden layer with 128 nodes using the ReLu activation function. The chosen optimizer was Adam as suggested in the lab instructions.

Parameters:
$\gamma = 0.99$
$L = 15000$
$T_E = 500$
$C = L/N$
$N = 32$

$$\epsilon_k = \max(\epsilon_{min}, \epsilon_{max}(\frac{\epsilon_{min}}{\epsilon_{max}})^{(\frac{k-1}{Z-1})}), \quad k = 1, 2, ...$$

where
$e_{max} = 0.99$
$e_{min} = 0.05$
$Z = 0.95T_E$

Explain the layout of the network that you used; the choice of the optimizer; the parameters that you used (, L, T E , C, N,) and which modification did you implement (if any). Motivate why you made those choices.

1.5 e)

The different γ values tested were the following
$\gamma_0 = 0.99$
$\gamma_1 = 1$
$\gamma_2 = 0.1$
where γ_0 is the value used in our solution.

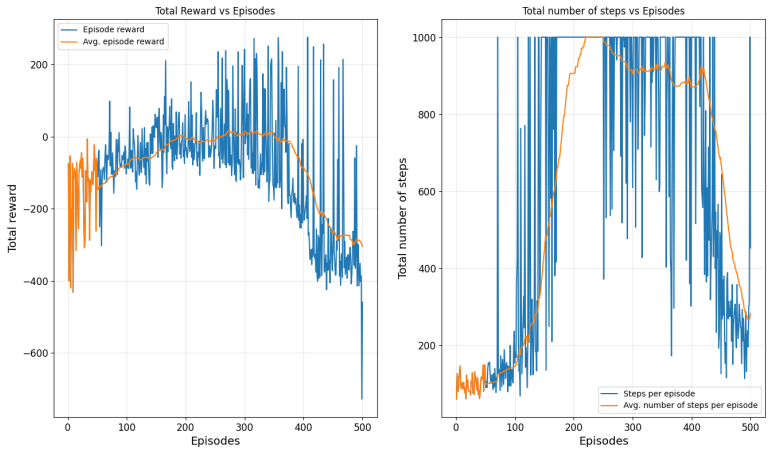

Figure 2: Reward and number of steps for $\gamma_1 = 1$.

When using $\gamma = 1$ the optimization of the policy for infinite time horizon would try to optimize a infinite sum for all polices that that have obtained on average a positive reward at each time step which is a bad optimization. As seen in Figure 2 the lander cannot find a policy that yields a good reward.

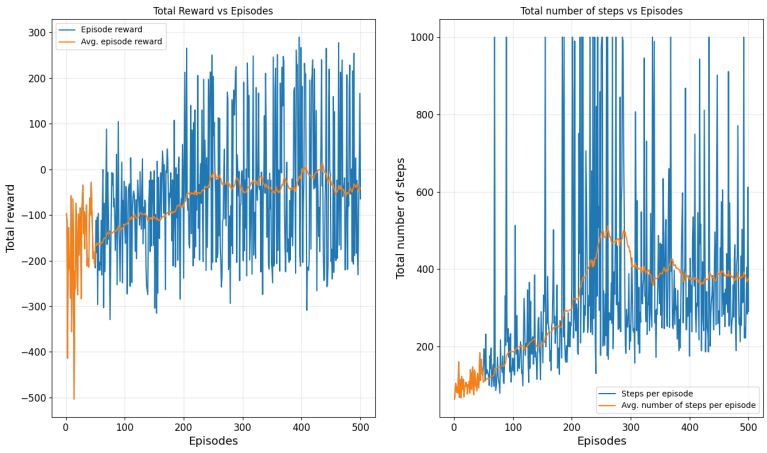

Figure 3: Reward and number of steps for $\gamma_2 = 0.1$.

With a small discount factor the agent only regards rewards very close in time, which makes it unable to predict actions several time steps away that will result in a successful landing. This results in an average reward close to zero and episodic reward oscillating between large positive and large negative values as it is quite random and luck-based if it will successfully land or not.

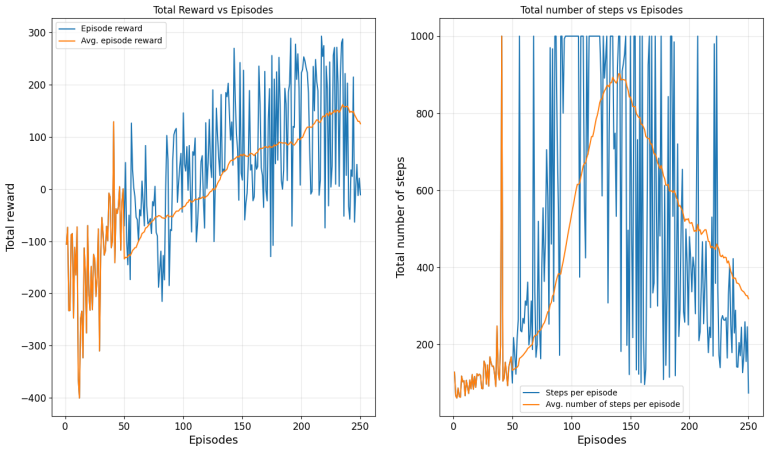

Figure 4: Reward and number of steps for $N_1 = \frac{1}{2}N = 250$.

Decreasing the number of episodes to 250 results in the plots shown in Figure

4. As can be seen at the end of the episodes, the variance is high for Figure 4 compared to Figure 1 as the network has not had enough training to find the optimal policy.

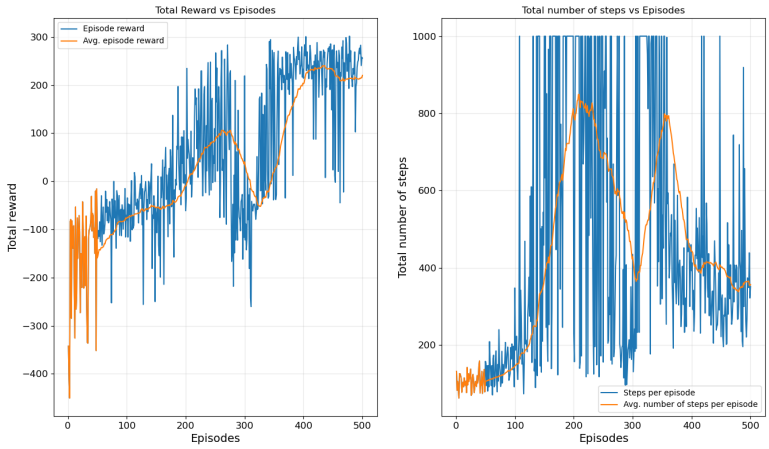

Figure 5: Reward and number of steps for $L_1 = \frac{1}{2}L = 7500$.

Decreasing the buffersize, L, does not appear to have an effect on the final policy but seems to make the training process less stable as can be seen in Figure 5 where the average reward decreases in the middle of the process rather than the constant linear increase seen in Figure 1. This might due to the increased update frequency of the target network due to the decrease in C which implies a less stationary target.

1.6 f)

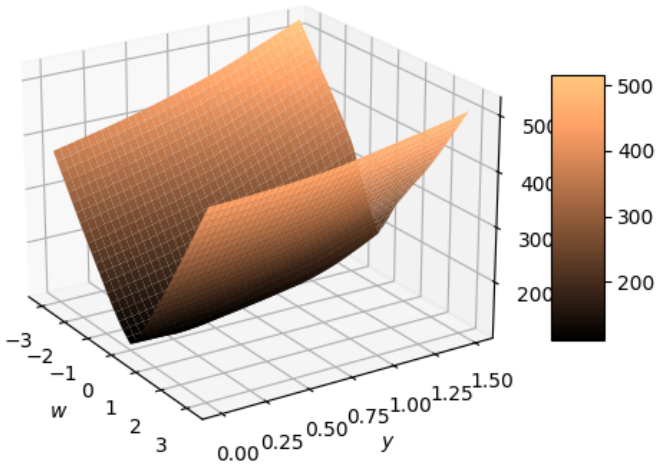

Figure 6: $\max_a(Q_\theta(s(y, w), a))$

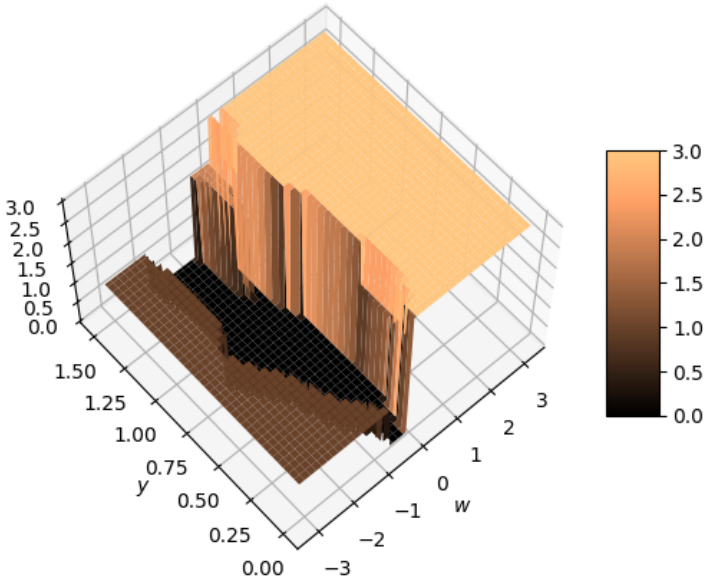

Figure 7: $arg\max_a(Q_\theta(s(y,w),a))$

Figure 6 shows the maximum Q values for y $\in$ [0, 1.5] and w $\in [\pi, \pi]$. The figure shows the maximum total discounted reward. This means that it is all the remaining reward, therefore the shape and values makes sense. When you are at (x,y) = (0,0) with w = 0 you will be done and receive the reward +100 for a successful landing which corresponds to the approximation that the Q-values gives which is approximately 100. When you are at higher angels or further away from zero there is a higher reward to collect in the episodes by doing actions.

The policy shown in Figure 7 also makes sense since depending on the angle, w, we should fire either the left or the right orientation engine in order to straighten the lander and if the angle is close to zero do nothing since we want to land. The exception is if y is large then we should fire main engine in order to lower the decent speed and aviod a crash.

1.7 g)

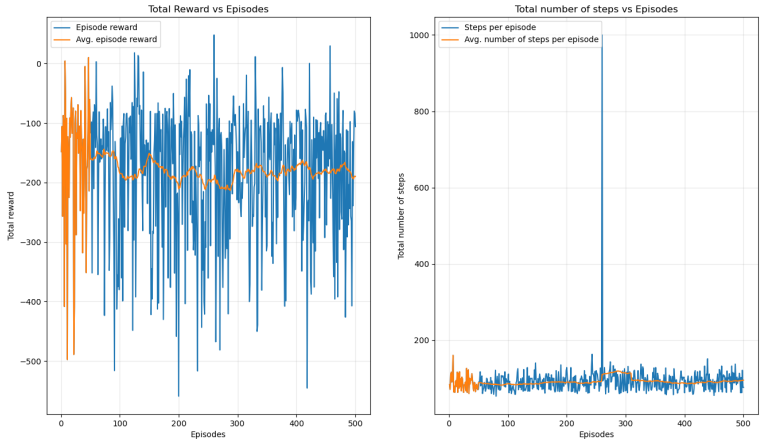

Figure 8: Reward and number of steps for a random agent

As we can see in Figure 8 the random agent only does random actions and does not improve its reward as the agent in Figure 1.

1.8 h)

The policy was validated using "DQN_check_solution.py"