Salma Chaaban – 301216551

COMP257 - Assignment 6 – Analysis Report

1. A simple DQN epsilon policy network with 4 output neurons, corresponding to the 4 possible actions the agent can make in the LunarLander-v2 Environment was created.
2. There is an input layer, and 3 hidden layers containing 128 neurons each and using the ReLU activation function as it is commonly used and works well in many applications. Other activation functions were attempted such as tanh and leaky ReLU. The output layer uses a linear activation function as Q-values can any real value with no bounded range required. When the model is compiled, the MSE is used for the loss function as it is also commonly used for its good performance.
3. The hyperparameters were experimented with multiple times. In the end the ones kept were:
   1. **Learning Rate:** 0.001.
   2. **Discount Factor:** 0.99 heavily prioritizing future rewards**.**
   3. **Batch Size:** 128.
   4. **Max Episodes:** 300.
   5. **Max Steps:** 500.
   6. **Epsilon Decay:** 0.995, with a minimum of 0.01.
   7. **Replay Buffer Size:** 20,000, 30,000 was attempted to try to increase the agent’s ability to generalize.
   8. **Target Network Update Frequency:** 5 episodes as it is s standard.
   9. **Exploration Rate:** 1.0, starts with full exploration.

Additionally, different numbers of hidden layers and neurons were experimented with, such as 1 hidden layer with 64 neurons, 2 with 64 neurons, 3 with 256, 128, and 64 neurons, and so on.

1. The agent was trained on 300 episodes. For a better performance, there should be a larger number of episodes; however, given the time and computational power needed, 300 was settled with as other values were also experimented with such as 200, 400, 500, and more which did not significantly improve the performance and took a lot longer, so not many different hyperparameters were tested on larger episode values.
2. Some performance metrics were plotted. The rewards per episode were increasing overall as the episodes went on. The cumulative rewards over time decreased over the episodes and then at about episode 190, they started increasing, which might mean that the agent needs more episodes. The episode lengths over time were increasing towards the end, showing that the agent is taking more time and surviving for longer using more steps per episode or not landing at all (as shown in the episode at the end), unlike at the beginning where it drops and crashes and does not take much time.
3. The main challenge faced during training was the time taken, given that it takes a lot of time to train the agent, not all combinations of hyperparameters that would seem to perform well were tried. The results, with an episode number on the lower side, did not yield the most satisfactory results even with a lot of hyperparameter tuning; therefore, being the biggest obstacle. A higher number of episodes usually allows the agent more experiences, a better chance to learn, and a lower exploration rate at the end of training to allow for exploitation. A good approach might be to train on a low number of episodes multiple times in an incremental learning fashion.

Another implementation was attempted, with these hyperparameters:

1. **Learning Rate**: 0.01, smaller than the first value, as the agent seemed to have difficulties learning.
2. **Discount Factor**: 0.95 also heavily prioritizing future rewards, but a little less than the first one
3. **Batch Size**: 32 less than the first.
4. **Max Episodes**: 1000 a lot more than the first.
5. **Max Steps**: 200 gives less time for the agent to explore in an episode than the first.
6. **Epsilon Decay**: Starting with 1.0 and decaying to 0.01.
7. **Replay Buffer Size**: 2000 same as the first.
8. **Exploration Rate**: 1.0 starts with full exploration, like the first.

A plot showing the sum of rewards over the episodes was plotted, and it shows an overall increase over 500 episodes with a slight decrease towards the end. Then an animation of one episode was shown, where the agent does not land and just floats. This might mean that there should be a low epsilon decay.

In conclusion, an episode number between 500 and 1000, maybe about 800, might be the best with a lower epsilon decay, and some more hyperparameter tuning.