Navigation project Report :

For this project, I have briefly summarized the learnings and yper parameters used for building the model. I have observed that it was possible to solve the environment in around ~ 500 episodes with a number of layers in DQN and different choices of ε.

Model:

A number of experiments are performed with different numbers of layers]in DQN and finally a   3 layer feed-forward layers with ReLu activation was identified for better performance of DQN. With state space dimension of 37 and output/action space dimension of the problem is not too high-dimensional, so too high numbers of hidden layers or units within the layers do not seem to be justified. A different choices number of hidden layers was ideal to perform the experiment and showed significant results. All these architectures allow to solve the problem in well below 1000 episodes. A final model of 128*64*32 sizes of the three hidden layers was chosed for the experiment.

Experiments:

Also, the model was used with both Deep Q-learning, as well as Double Deep Q-learning. During experimentation Double DQN seemed to outperform plain DQN, so I used it in my final version.
Choice of ε. As outlined in the lectures and as observed in some of the previous coding exercises, the choice of ε (starting value, decay factor/speed, final value) has a large effect on the speed of learning. I decided to go with a multiplicative decay with minimum value in the long run: ε = max(ε 0 · ε k decay , ε min), where k denotes the episode. After starting with relatively conservative, and previously seen values of ε decay (e.g., 0.995) I decreased it further and further and observed quite fast training progress. My interpretation is that the environment is not too complex (i.e. it does not generate too much variation in the state space) and thus the agent needs relatively little exploration compared with other environments. The final choice of parameters was as follows: ε 0 = 1, ε decay = 0.97, ε min = 0.005.

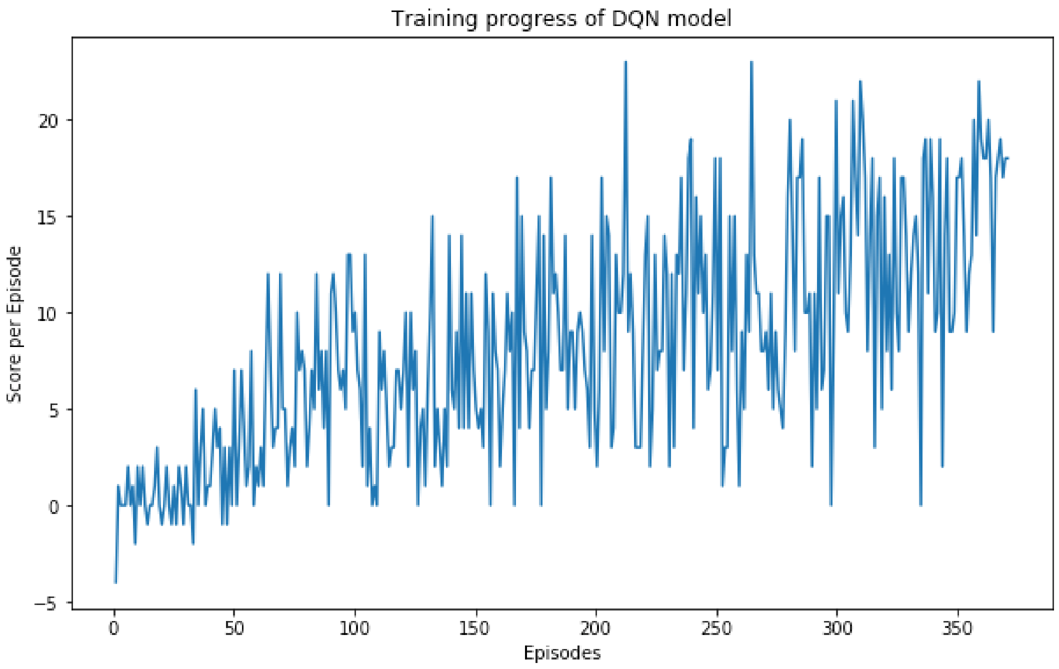

Figure 1: Example training run of the agent.

Results:

As mentioned earlier, the environment can be solved in < 500 episodes with different layers of architectures and choices of ε. We have experimented and shown the results of a training run with the final setup. At the initial phase the agent showed little progress due to exploration and starts to receive higher scores after $\sim$ 50 episodes. The progress seems to flatten around 200 episodes, but the average scores still rises until average score of 13 over 100 episodes is reached.

Further work:

Instead of random based steps from the memory buffer, prioritized experience replay might show a better performance. Also, applying policy based, actor-critic method would be ideal for this experiment. My future work would be adopting the same experiment using the above approaches.