Project 1: Navigation

This report includes a summary of the learning algorithm for the Navigation project, its outcome and ideas for follow-up work.

1 – Learning Algorithm

In this project, the agent was modelled as a DQN agent. In DQN, a deep neural network was used to approximate the action value function. The neural network took as input the 37 dimensional state of the environment, and outputted the action values for the 4 discrete actions. An epsilon-greedy policy was used to select the action in a state. The starting value for the epsilon was 1.0, and after each episode it was decayed by the factor of 0.995. The value for epsilon was not allowed to go lower than 0.01.

In DQN we used a target and a local network. Both networks had the same architecture. At every 4 episodes, the weights of the target network were calculated using the following soft update:

$$\theta_{target} = \tau\theta_{local} + (1-\tau)\theta_{target}$$

where $\tau = 0.001$ was used. This suggests that the weights of the target network were slowly changed compared to those of the local network. By fixing the weights of the target network for multiple steps, we obtained more stationary targets which improved the convergence of the optimizer. We used ADAM as optimizer with a learning rate of 5e-4 and a batch size of 64. The size of the replay buffer was set to 1e5. The discount factor of 0.99 was used.

As mentioned above both the local and target networks have the same architecture. In this architecture, we used 2 hidden layers each having 64 nodes. We used ReLU activation function after the hidden layers.

2 – Results

Figure 1 shows the score (cumulative reward) per episode and the average score over the last 100 episodes. It can be seen that after 449 episode, the average score of 13.18 was obtained. At that episode, the training was stopped as the requirement for the agent's success was achieved.

3 – Ideas for follow-up work

The following ideas come to mind for improving the agent's performance:

- The lowest hanging fruit is to do a hyperparameter tuning

- Embed the improvements from this implementation of DQN towards Rainbow. Prioritized Experience Replay, Double DQN, Dueling DQN, etc. could be the improvements we can consider.

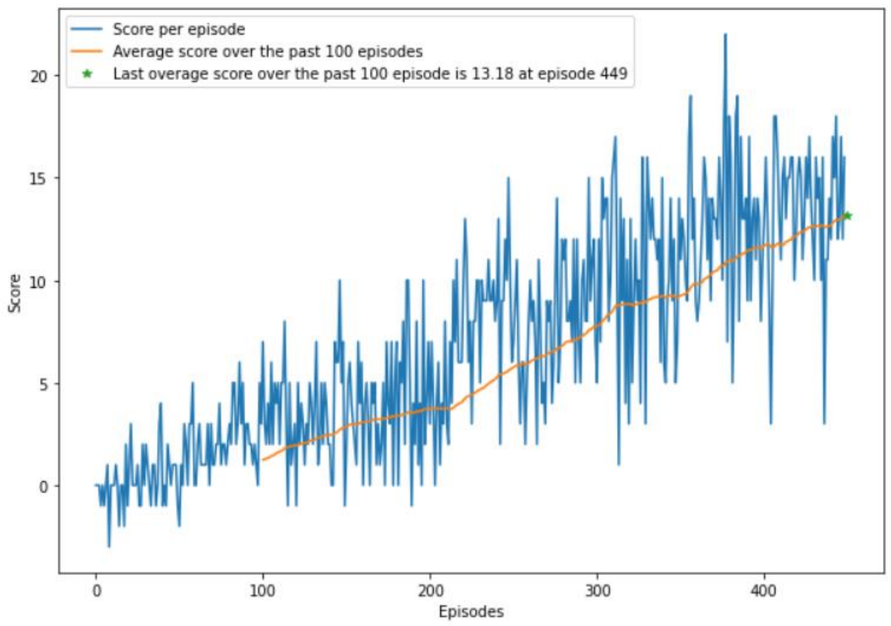

Figure 1 Performance of the agent over the episodes.