Report

This project uses Deep Deterministic Policy Gradient (DDPG) with the following hyperparameters to solve the environment:

BUFFER_SIZE = int(1e5) # replay buffer size

BATCH_SIZE = 1000 # minibatch size

GAMMA = 0.99 # discount factor

TAU = 1e-3 # for soft update of target parameters

LR_ACTOR = 1e-4 # learning rate of the actor

LR_CRITIC = 1e-3 # learning rate of the critic

WEIGHT_DECAY = 0 # L2 weight decay

DDPG uses actor-critic method, where the critic model learns the value function and uses it to determine how the actor should change. Since the policy is deterministic, to explore the environment, it uses the Ornstein–Uhlenbeck process to introduce noise. DDPG also uses replay buffer to store the experience of each step, then sample mini-batches of experience to update the actor and critic. The actor and critic neural network architecture have 2 hidden layers with 200 nodes for the first hidden layer and 100 nodes for the second hidden layer. The actor and critic target network are updated using soft updates strategy to provide faster convergence and better stability.

During initial testing, the result has big fluctuations and does not seem to be learning even after 100 episodes. Changing from learning from 1 agent to 20 agents produced the same result, in one case, the agents' score decreased with each episode. After making changes to the batch size and actor and critic architecture, the agents started to perform better. To see what failed and what worked, see Continuous control test log.pdf. At the end, two methods produced desirable result. The first one is, it learns once every 50 steps and the second one learns three times every 50 steps. The first method produced less noisy result at a faster rate compared to the second method. Initially, the intuition I had was that, learning more would produce a better result, but this is not the case. Further research must be done to better understand the relationship between learning rate and result stability.

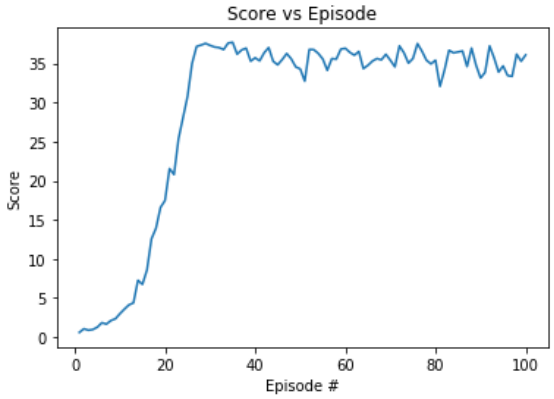

Figure 1: Method 1

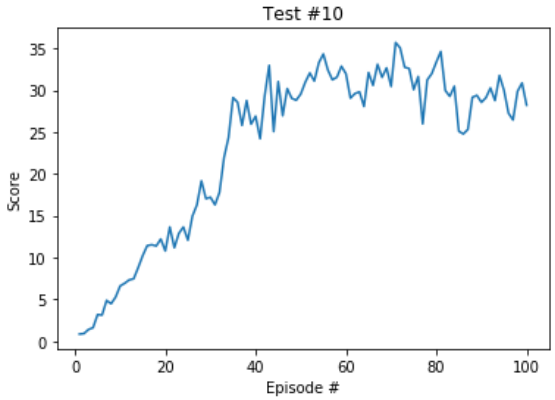

Figure 2: Method 2

As you can see from the figures above, Method 1 was able to reach the score of 30 at around episode 25 and plateaued at around episode 30 with a score of 37. Although it took as much as 3 times longer to run compared to Method 1, Method 2 reach the score of 30 at around episode 37, and it produced results with higher variance compared to method 1.

The result produced by method 1 with 20 agents received an average score of 31.09 at the 30^{th} episode and maintained an average score above 30 until the last episode with an average of 34.96 over 100 episodes across 20 agents.

In the future I have 2 plans for this project. The first one is to implement different algorithms (PPO, TRPO, TNPG...) and compared its' training time, variance and general performance. The second one is to find the optimal neural network architecture for actor and critic and also the optimal learning rate, to further reduce the time needed to reach an average score of 30.