Udacity Deep Reinforcement Learning Nanodegree
Continuous Control Project Report

Deborah Fish

Introduction

This report provides a brief description of my project to train an agent to control a double-jointed robot arm in the Unity ML Agents Reacher environment. The environment has a discrete observation space with a state size of 33 (comprising positions, angles, velocities and angular velocities), while the arm(s) have a continuous action space with four actions. A rewards of +0.1 is earned for every timestep in which the robot arm's hand is in a target location. There are two versions of the environment: one containing a single robot arm; another containing 20 robot arms. Success is obtained by achieving an average score (averaged over all 20 robot arms in the latter environment) exceeding 30 over 100 consecutive episodes.

Learning Algorithm

I adapted the DDQN algorithm provided in the course (Lillicrap *et al.*, 2015), to work in the second environment, containing 20 robot arms. The main changes required were:

- Using the actor and critic networks to generate 20 trajectories (each up to 1000 timesteps long), and adding all 20 trajectories to the replay buffer at each timestep.
- Modifying the Ornstein-Uhlenbeck noise function to add Gaussian random noise to the actions generated by each of the 20 agents.

Following the guidance from Udacity, I only took one batch of samples from the experience replay buffer each timestep, despite adding 20 trajectories to the buffer. I also clipped the gradients for the critic network, as suggested by Udacity.

Network architecture. My Actor and Critic networks took the state space as input, added two fully connected layers of sizes [256, 128], each with ReLU activation, then finally a fully connected output layer of size equal to the action space.

Optimiser. I used the Adam optimizer to iterate the network weights. I started with the default value of the learning rate for the critic, and found that decreasing this learning rate significantly increased the rate at which the agents trained.

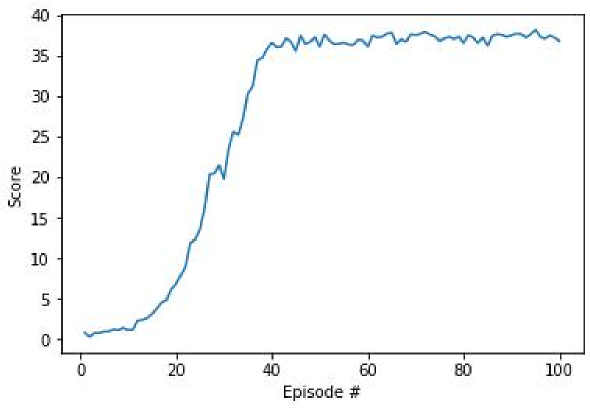
Hyperparameters. I started with the default hyperparameter values provided in the Udacity DDPG implementation, and sought to experiment with key hyperparameters. The key parameter that I changed was decreasing the critic's learning rate from the value in the Udacity DDPG algorithm. The hyperparameters used, and the other values tried, are summarised below:

Hyperparameter	Value used	Other values tried
Batch_size	128	None
Buffer_size (experience replay)	1e5	None
Gamma	0.99	None (standard value)
Tau (for soft updates)	1e-3	None
Actor - Hidden_layers	[256,128]	[400,300], [256,128]
Critic - Hidden_layers	[256,128]	[400,300], [128,64]. 128 and 64 hidden layers worked almost as well as 256 and 128, but didn't train any

		quicker.
Actor - Learning rate (Adam optimizer)	1e-4	None
Critic - Learning rate (Adam optimizer)	2e-4	1e-3, 1e-4. Decreasing this learning rate from the original value of 1e-3 provided in the DDPG algorithm greatly accelerated the agent's learning.
Weight_decay	0	5e-4 (this value hardly learnt at all).

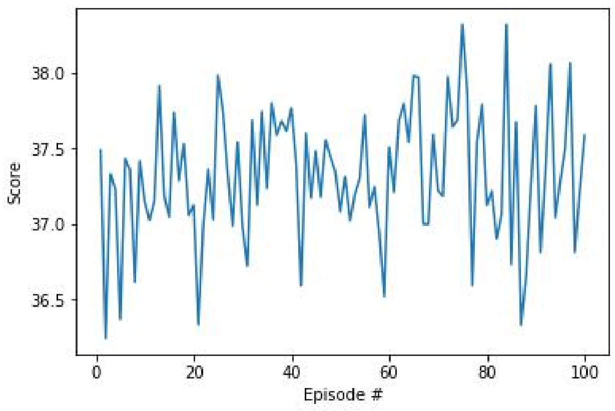
Plot of Rewards

A plot of the rewards achieved during training is shown below. Rewards are plotted as a function of the episode number, showing that after around 38 episodes (for trajectories with a maximum of 1000 timesteps) the agent is obtaining a score over 30.



The model details are saved to file 'checkpoint.pth', and were used to evaluate the model in evaluation mode achieving an average score of 37.32 over 100 episodes.

```
Episode 100     Average Score: 37.32
```



Ideas for Future Work

To be completely honest, I am struggling to see how to improve the performance of this agent, as it trains fairly quickly and achieves a performance clearly above the required threshold. However, it would be interesting to compare the agent's performance with that of a PPO algorithm, noting that TRPO has been shown to perform well on similar tasks.

References

Lillicrap, Timothy P. et al., 2015, Continuous control with deep reinforcement learning, available at https://arxiv.org/abs/1509.02971