Syed Bilal Mehdi
Project: Continuous Control
Deep Reinforcement Learning

## Learning Algorithm (DDPG)

An implementation of Deep Deterministic Policy Gradient (DDPG) algorithm with some slight variations was used. The details of the implementation are as follows:

**DDPG parameters:**

A few little variations were made to the algorithm based on the information provided with the project and after discussion with other course members:

1. The training wasn't started until 20e4 samples were already collected.
2. Since data from 20 different agents was received after every step of the environment, the local actor and critic networks were updated 10 times (instead of just one) after every step of the environment.
3. Soft update for the target networks was still done only once after every step of the environment.
4. The training was halted after a score of 35 was observed since this was a good indicator that an average score of 30 was easy to achieve for the agent and no more training is required.

The hyperparameters used are as follows:

Adam optimizer with a learning rate of 1e-4 and gradient clipping was used for both the actor and critic network. Batch size of 128 and a total buffer size of 1e6 was used.

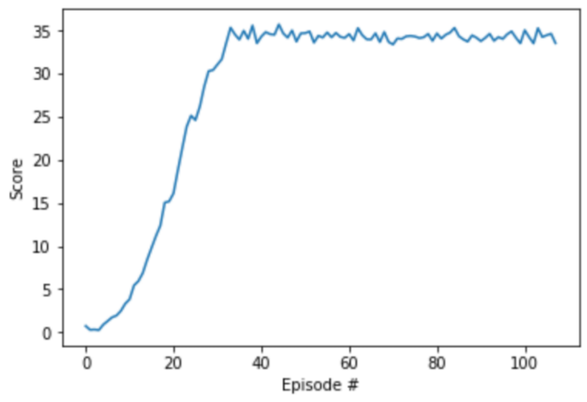
**Actor Network:**

Two hidden layers with 400 and 300 units each with ReLU are used. The first layer also had batch normalization. The output layer used tanh non-linearity.

**Critic Network:**

Two hidden layers with 400 and 300 units each with ReLU are used. The states fed the first layer, whereas, the first hidden layer and the actions fed the second layer. The output layer also used ReLU non-linearity.

## Results



As can be seen in the figure, the agent trained quickly and reliably.

## Future Work:

Although this version of the environment could be solved quickly with decent ranges of parameters, a more difficult variation such as crawler could be tried. A more difficult environment would also justify use of more sophisticated methods such as D4PG.