Udacity Deep Reinforcement Learning Nanodegree
Project 2: Continuous Control

Submitted by: Tamoghna Das

1. Network architecture and Learning Algorithm

A Deep Reinforcement Learning Actor-Critic Agent has been trained using DDPG algorithm to solve the Reacher Environment with Twenty agents, each with its own copy of the environment. The implementation is in Continuous_Control_Reacher.ipynb notebook. The DDPG Agent is coded in deep_rl\agent\DDPG_agent.py. The Actor and Critic Networks are coded inside deep_rl\network\network.py.
The codes are mainly taken from udacity ddpg-pendulam project and adapted to suit the current project.

Deep Deterministic Policy Gradients (DDPG) algorithm can be seen as a Actor-Critic method. It could also be seen as approximate DQN algorithm with Continuous Action spaces. The Critic in DDPG is used to approximate the maximizer over the Q Values of the next state, and not as a learned baseline.
DDPG uses four Deep Neural Networks , two for Actor (local Actor, target Actor) and two for Critic (local Critic, target critic). Two interesting aspects of DDPG are: a) Replay buffer b) soft updates of the target Network. Local Actor and Critic Networks are actually trained. A soft update strategy for slowly blending the local network weights with target network weights is used.
The Actor is used to approximate the optimal policy deterministically. That means, the Actor outputs best believed Action every single time upon query. The Critic learns to evaluate the optimal Action Value function by using the Actor's best believed Action.

Actor network Architecture:
- Batch normalization
- Inputs: 33 units (state_size)
- hidden layers
    - fc1: fully connected 33 x 128. (Leaky ReLU)
    - fc2: fully connected 128 x 128. (Leaky ReLU)
- output layer. Fully connected layer.4 tanh output units for 4 actions(action_size)

Critic network Architecture:
- Batch normalization
- Inputs: 33 units (state_size)
- hidden layers
    - fcs1: fully connected 33 x 128. (Leaky ReLU)
    - fc2: fully connected 132 (4 actions from Actor Network added) x 128. (Leaky ReLU)

- output layer. Fully connected layer.1 linear output unit.

Optimizer : Used for local Actor and local Critic. Adam Optimizer (torch.optim.Adam). LR_Actor = 1e-4, LR_Critic = 1e-3

Hyperparameters:

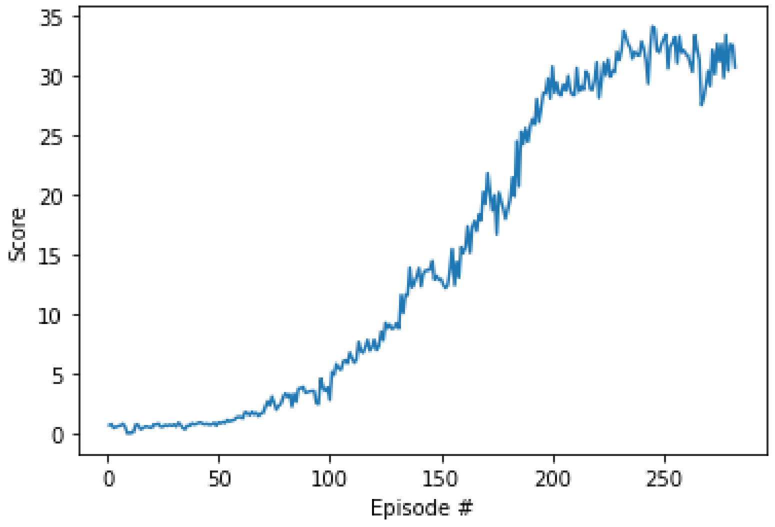
- Replay buffer size: 1000000
- Batch size : 256
- Discount fator (Gamma) : 0.99
- Tau (used for soft blending of target parameters): 0.001
- Learn Every 2 steps

2. Results

The environment is solved after 182 episodes:

```
Environment solved in 182 episodes!  Average Score: 30.05
```

Plot of Scores vs Episodes#:



3. Ideas of future work

- To implement A3C, D4PG on the same environment, compare their performances.
- To implement more advanced versions of DQN e.g. Double DQN, Rainbow DQN for performance improvement.