Deep Reinforcement Learning Nanodegree

Project 2 : Continuous Control

Chenbo Gu

1. Problem Introduction

In this project, a continuous control problem is solved. The physical model of the problem is a double-jointed arm which is designed to move to target locations. The states are the 33 physical states of the system, and the control inputs are torque applicable to two joints.

Reward:
Agent's hand in goal location: reward +0.1/each step
State:
The state space has 33 dimensions including position, rotation, velocity, and angular velocities of the two arm Rigidbodies.
Action:
4 continuous actions available:
torques applicable to two joints of the agent, ranging from -1 to 1.
Target:
An average score of +30 over 100 consecutive episodes.

More detailed environment information can be found on the github website provided by Unity:
https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Learning-Environment-Examples.md

2. Training Algorithms

DDPG is mainly adopted for solving this problem. Later on, DDPG with batch normalization in the neural network is also tested. Many changes are made to the sample code provided by Udacity.

2.1 **Vanilla DDPG**

Personally speaking, a large replay buffer size and a large batch size play a great role in my trials.

I. Hyper-parameters

```
BUFFER_SIZE = int(1e6)    # replay buffer size
BATCH_SIZE = 2**10            # minibatch size
GAMMA = 0.99                  # discount factor
TAU = 1e-3                    # for soft update of target parameters
LR_ACTOR = 1e-3               # learning rate of the actor
LR_CRITIC = 1e-3              # learning rate of the critic
WEIGHT_DECAY = 0.000    # L2 weight decay

# the network weights in both actor and critic cases are updated every 20 time steps
net_update_every = 20
n_episodes = 1000
max_t = 1000
```

II. Model Structure

Actor:

state_size-->fc1-->ReLU-->fc2-->ReLU-->fc3-->tanh-->action_size

fc1_units=400

fc2_units=300

Critic:

state_size-->fc1-->ReLU-->fc2-->ReLU-->fc3--> state-value function (dim = 1)

action_size-->

2.2 DDPG with batch normalization in the neural network

I. Hyper-parameters

```
BUFFER_SIZE = int(1e6)    # replay buffer size
BATCH_SIZE = 2**10            # minibatch size
GAMMA = 0.99                  # discount factor
TAU = 1e-3                    # for soft update of target parameters
LR_ACTOR = 1e-3               # learning rate of the actor
LR_CRITIC = 1e-3              # learning rate of the critic
WEIGHT_DECAY = 0.000    # L2 weight decay

# the network weights in both actor and critic cases are updated every 20 time steps
net_update_every = 20
n_episodes = 1000
max_t = 1000
```

II.Model Structure

Actor:

state_size-->BN0-->fc1-->BN1-->ReLU-->fc2-->BN2-->ReLU-->fc3-->tanh-->action_size

fc1_units=400

fc2_units=300

Critic:

state_size-->BN0-->fc1-->BN1-->ReLU-->fc2-->BN2-->ReLU-->fc3-->state-value function

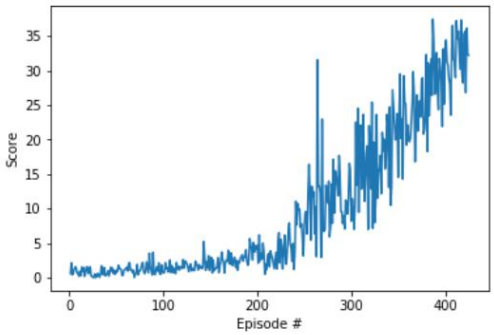
action_size-->

3. Results

3.1 Vanilla DDPG (from the Linux version)

The PC/windows version takes around 1500 episodes to achieve satisfying results.

```
Episode 50      Average Score: 0.76
Episode 100     Average Score: 1.19
Episode 150     Average Score: 1.63
Episode 200     Average Score: 2.65
Episode 250     Average Score: 4.36
Episode 300     Average Score: 11.42
Episode 350     Average Score: 16.94
Episode 400     Average Score: 25.67
Episode 425     Average Score: 30.01
```



Validation:

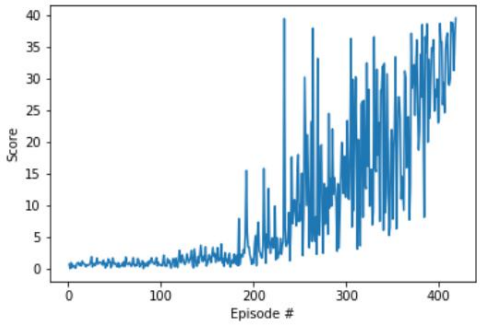
Total score (averaged over agents) this episode: 39.35999912023544

Total score (averaged over agents) this episode: 70.08999843336642

Total score (averaged over agents) this episode: 101.1599977388978

3.2 DDPG with batch normalization in the neural network (from the Linux version)

```
Episode 50      Average Score: 0.76
Episode 100     Average Score: 0.80
Episode 150     Average Score: 1.33
Episode 200     Average Score: 2.28
Episode 250     Average Score: 6.46
Episode 300     Average Score: 12.63
Episode 350     Average Score: 18.21
Episode 400     Average Score: 25.31
Episode 418     Average Score: 30.37
```



Validation:

Total score (averaged over agents) this episode: 35.46999920718372

Total score (averaged over agents) this episode: 70.23999843001366

Total score (averaged over agents) this episode: 109.62999754957855

3.3 Discussion

Both versions succeeded to achieve average score +30.0 in 500 episodes with the Linux environment provided. I also tested Vanilla DDPG on my local computer and it takes less than 1500 episodes but I lost the plots during validation.

Batch normalization, mainly designed for solving vanishing/exploding gradient problems, seems to improve the performance a little bit but it's effect is not obvious in this case. The rewards corresponding to the network with batch normalization rise slowly at the very beginning and start rising faster than those of the vanilla version.

4. Future Work

There is still much to do. I am still working on PPO. Maybe it's also a good idea to combine PPO and actor- critic together. A3C or D4PG may also be applied if time permits. I will keep updating the repository.