Udacity Deep Reinforcement Learning Nanodegree
Collaboration and Competition Project

1. Learning algorithm
    - Use the algorithm below

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

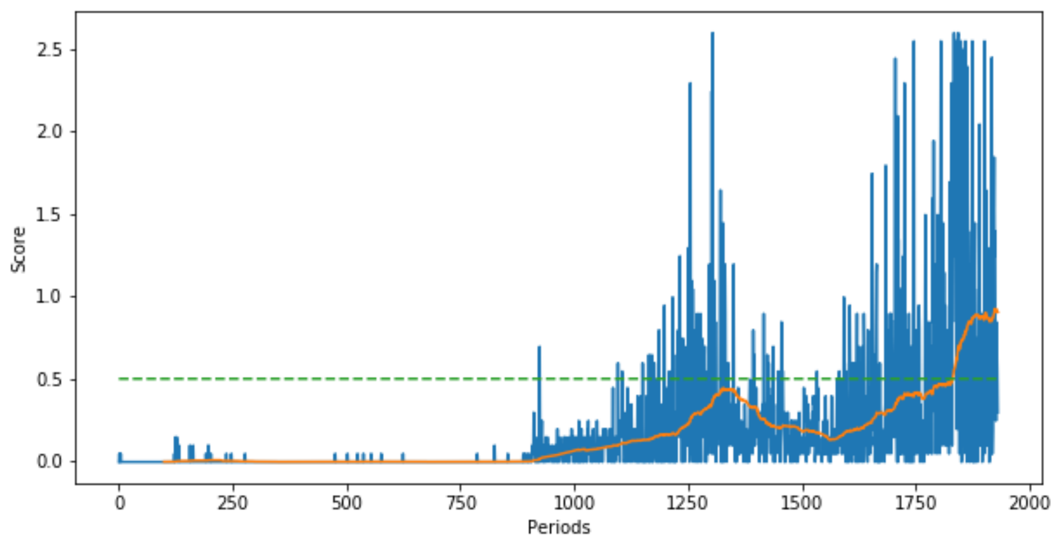
  end for
end for

- Neural network:
    - Actor
        - Two hidden layers
        - First layer with 128 nodes; relu activation
        - Second layer with 64 nodes; relu activation
        - Tanh activation for last layer
    - Critic
        - Two hidden layers
        - First layer with 128 nodes; relu activation
        - Second layer with 128 nodes; concatenated with actions; relu activation
        - Linear for last layer

2. Hyperparameters:
    - BUFFER_SIZE = int(1e5) # replay buffer size
    - BATCH_SIZE = 128 # 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 # L2 weight decay

3. Plot of scores



The environment is solved in 1931 periods. The criteria is that the average scores (with each score equal to average over the 2 agents in one period) over the previous 100 periods are greater than 0.5 for a consecutive 100 periods.

4. Future work:
   - I tried a few versions of MADDPG, but the result is not as good as DDPG reported here. I plan to further experiment with network structures in MADDPG and see whether MADDPG can beat DDPG for this case.
   - Use batch normalization to improve performance
   - Adding prioritized experience replay