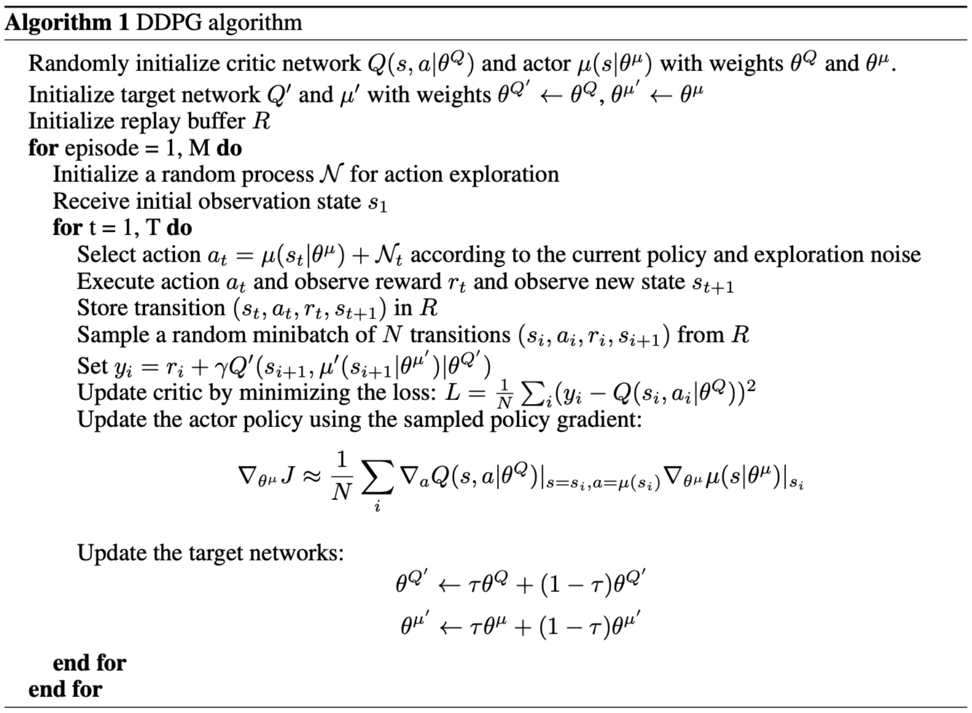
Udacity Deep Reinforcement Learning Nanodegree
Continuous Control Project

1. Learning algorithm
    - Use the algorithm below

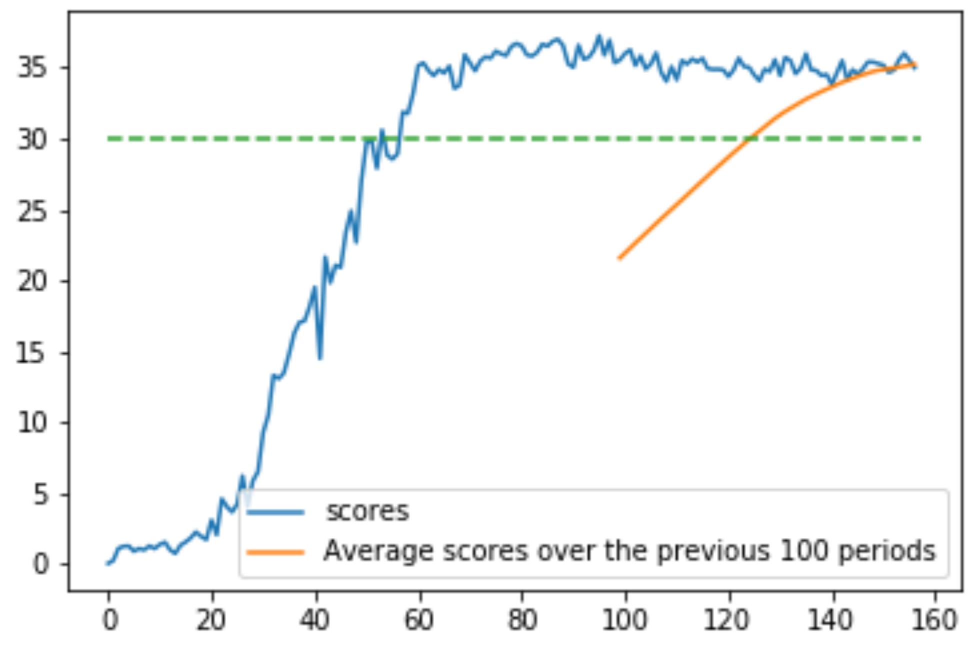


Algorithm 1 DDPG algorithm

```
Randomly initialize critic network $Q(s,a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights $\theta^Q$ and $\theta^\mu$.
Initialize target network $Q'$ and $\mu'$ 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 (4 actions)
    - Critic
        - Two hidden layers
        - First layer with 128 nodes; relu activation
        - Second layer with 128 nodes; concatenated with 4 actions; relu activation
        - Linear for last layer (one value)

2. Plot of scores
   For the criteria that the average scores (with each score equal to average over the 20 agents in one period) over 100 periods are greater than 30 for 100 consecutive periods, the environment is solved in 126 periods.



3. Future work:
   - Use batch normalization to improve performance
   - Adding prioritized experience replay
   - Using TD-lambda instead of TD(0) for updating Q value
   - Try other methods: PPO and A3C