Introduction

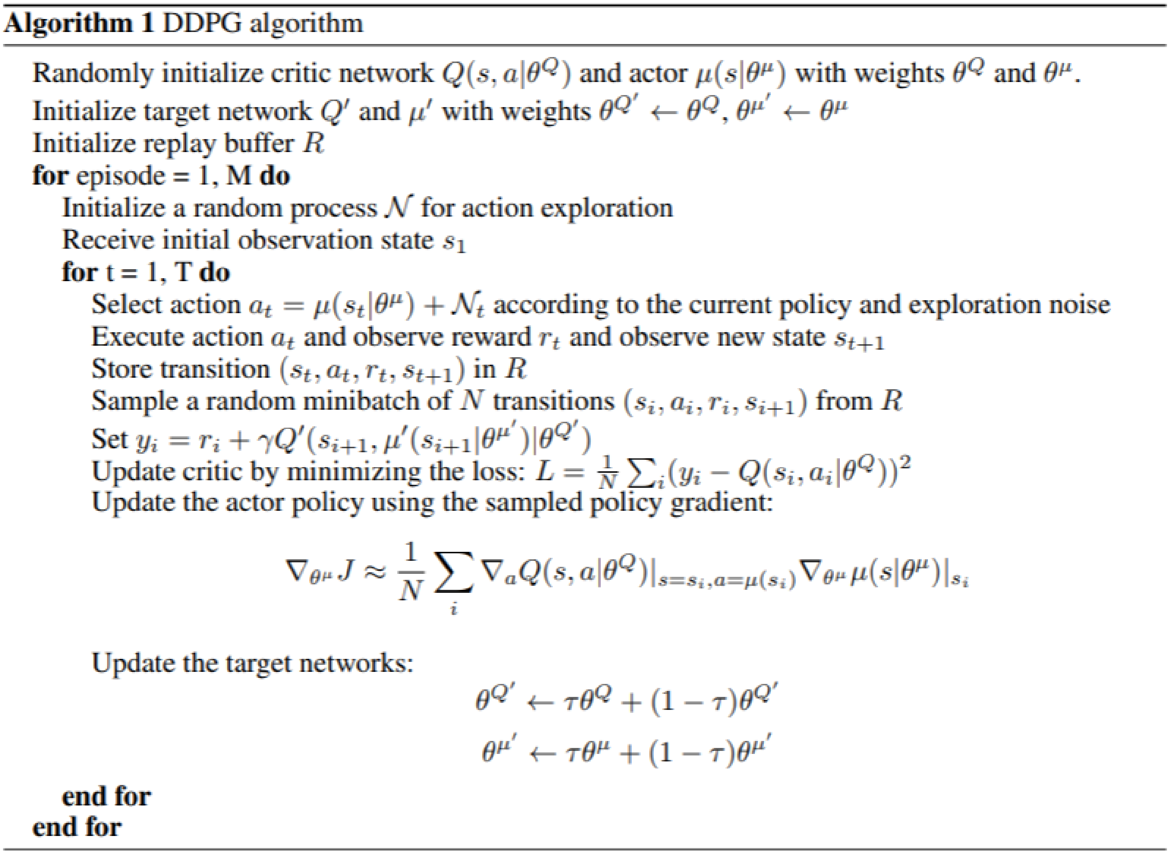
This is my report of project 2: Continuous Control. In this project, I trained a double-jointed arm to move towards target locations and maintain its position for as many as time steps as possible. The observation space consists of 33 variables corresponding to position, rotation, velocity, and angular velocities of the arm. Each action is a vector with 4 numbers, corresponding to torque applicable to two joints. Every entry in the action vector should be a number between -1 and 1.

I chose the second version of the environment which contains 20 identical agents. The problem is considered solved when the average (across 100 consecutive episodes) of those average scores from 20 agents is at least +30.

Algorithm

I solved the problem using Deep Deterministic Policy Gradient (DDPG).

Pseudocode of DDPG:



```
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
```

Hyperparameters:

buffer size: 1000000

batch size: 128

γ (discounting rate): 0.99

τ (soft update): 0.001

learning rate for actor: 0.001

learning rate for critic: 0.001

number of time steps that model learns: 20

number of times that model updates: 10

noise parameters: 0.2 (σ), 0.15 (θ), 1.0 (ϵ-init), 1e-6 (ϵ -decay)

Architecture of the actor network:

input layer (#33) -> hidden layer (#400) -> batch normalization -> hidden layer (#300) -> output layer (#4)

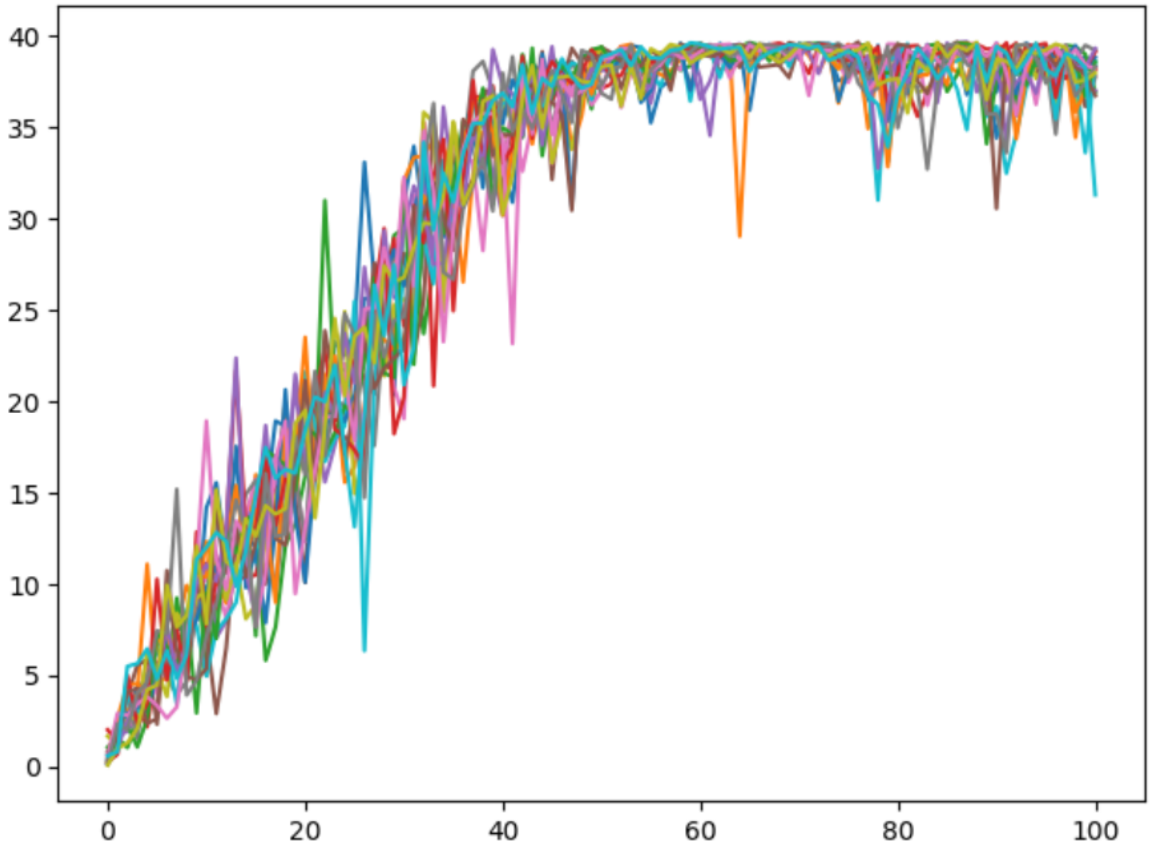
All fully connected layers use ReLU for activation except for the output layer which uses tanh for activation.

Architecture of the critic network:

input layer (#33) -> hidden layer (#400) -> batch normalization -> concatenate actions -> hidden layer (#300) -> output layer (#1)

All fully connected layers use ReLU for activation except for the output layer which doesn't use an activation.

Result



As shown above, the score achieved the target (30) in less than 40 episodes. And the agents stop improving after around 50th episode.

Future work

Implement PPO, A3C and D4PG, and then compare the results.