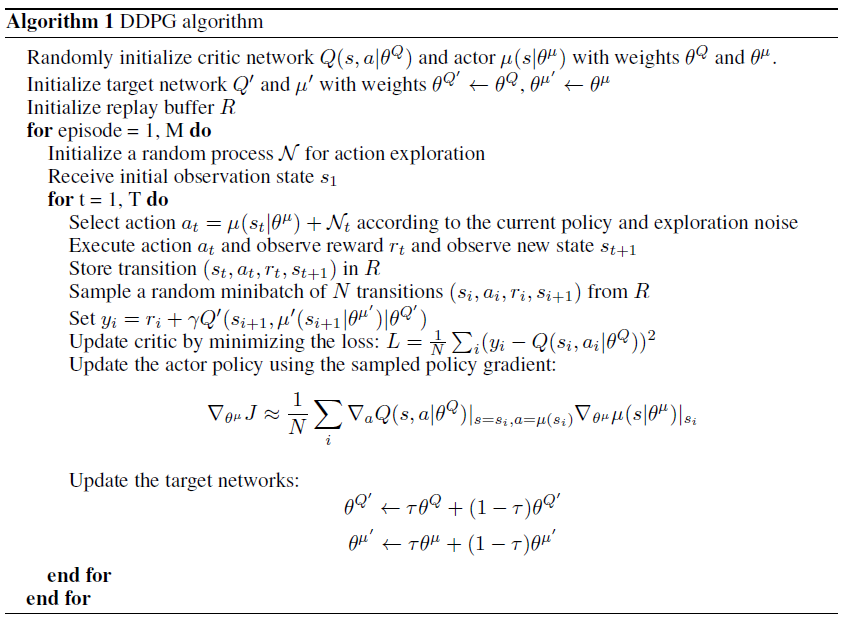
Report

# Algorithm

Selected Algorithm: DDPG (*ddpg\_agent.py*)



Parameters chosen for the DDPG Agent:

BUFFER\_SIZE = int(1e5) # replay buffer size  
BATCH\_SIZE = 1024 # minibatch size  
GAMMA = 0.9 # discount factor  
TAU = 1e-3 # for soft update of target parameters   
LR\_ACTOR = 1e-4 # learning rate   
LR\_CRITIC = 1e-3 # learning rate   
WEIGHT\_DECAY = 0 # L2 weight decay   
UPDATE\_EVERY = 20 # how often to update the network   
UPDATE\_EVERY = 10 # how many times to train the agent in a row

The agent consists of two different NN architectures:

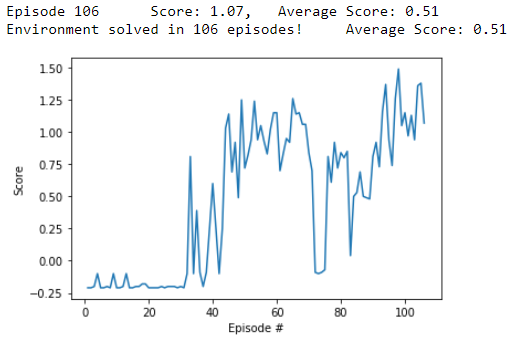
Actor

* A three-layer network with following number of units: Input (33) -> Hidden (256) -> Output (4)

Critic

* A five-layer network with following number of units: Input (33) -> Hidden1 (256) -> Hidden2 (260) -> Hidden3 (128) -> Output (1)

Environment was solved in 106 episodes (as can be seen in the following chart as well as in the *Tennis.ipynb*).



# Modifications compared to the lecture

The selected method, approach and parameters are exactly same as in case of Continuous Control project. No modification has been done in order to achieve the goal.

Batch normalization added according to the DDPG paper to all the input and layers in actor and the input and all layers before the action input in the critic – here in both cases it means only once

Bacth size increased to 1024

Agent is trained always after 10 steps, but 5 times in a row

Sigma for adding noise decreased to 0.1

# Improvements

Compare with other algorithms: PPO, A3C

Experiment with deeper actor network

Experiment with wider networks