Project Report

Learning Algorithm:

This project is similar to previous reacher project with some modifications for two agents. I have earlier tried to use MADDPG but could not train agent properly so i switched back to DDPG to train these agents to play tennis.

Here there are two agents learning simultaneously with shared replay buffer, seperate actors and a shared critic network. DDPG works good here because each agent learns seperately, having their own state and actions. Each agent is rewarded for hitting the ball correctly over the net in bounds, it is able to learn this properly using DDPG algoirthm and acheive average score of 0.5 over 100 episodes. If we want to improve performance then we should look for algorithm like MADDPG which works good for multi agent systems.

ACTOR:

Linear(state_size, fc1_units), relu #state_size 24, fc1_units = 256
BatchNorm1d(fc1_units)
Linear(fc1_units, fc2_units), relu #fc2_units = 128
BatchNorm1d(fc2_units)
Linear(fc2_units, action_size), tanh #action_size = 2

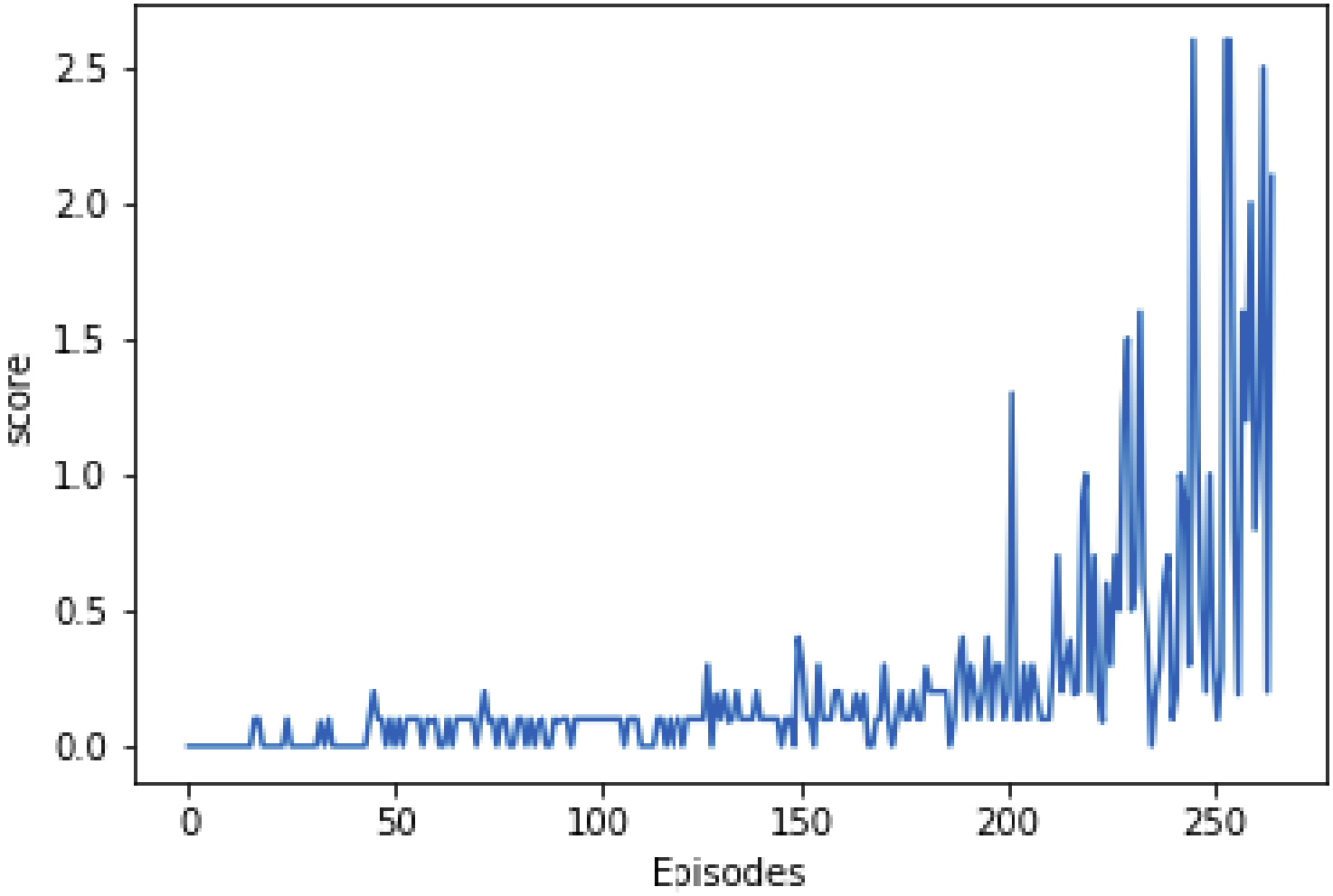
CRITIC:

Linear(state_size, fc1_units) #state_size 24, fc1_units = 256
BatchNorm1d(fc1_units)
Linear(fc1_units+action_size, fc2_units) #fc2_units = 128 action_size = 2
Linear(fc2_units, 1)

AGENT:

BUFFER_SIZE = int(1e5) #replay buffer size
BATCH_SIZE = 256 #mini batch size
GAMMA = 0.99 #discount factor
TAU = 0.02 #for soft update of target parameters
LR_ACTOR = 1e-3 #learning rate for actor
LR_CRITIC = 1e-3 #learning rate for critic
NUM_UPDATES : 4 #how many times network is updated
TIME_STEPS : 2 #update network every 2nd timestep
Nosie EPSILON : 0.2
EPSILON_DECAY Noise : 0.9999
SIGMA Noise : 0.2

Rewards plot per episode:



Idea for future work: Implement this using MADDPG Algorithm.