Project 3

Multi-Agent Actor-Critic Deep Reinforcement Learning

1 Overview

The goal of this project is to train two agents that play tennis. The agents receive a reward of +0.1 if they play the ball over the net. If they drop the ball they receive a reward of -0.01. The score is calculated based on the total cumulative reward both agents get. Thus, the agents are trained to work cooperatively rather than competitive.

For training I used the MADDPG (Multi Agent Deep Deterministic Policy Gradient). I used two separate actor-critic agents that share a replay buffer, and their actor network.

The state space consists of eight variables corresponding to the position and velocity of the ball and racket. Each agent revives its own, local observation. The action space consists of two continuous actions corresponding to movement in direction of the net and vertical direction of the racket.

2 Results

Initially I had a hard time training the two agents because one agent was always way better than the other agent. This also had a negative performance on the better agent, since they are trying to optimize a collaborative score. After using the same actor network I got way better results and was able to solve the environment in 1638 episodes.

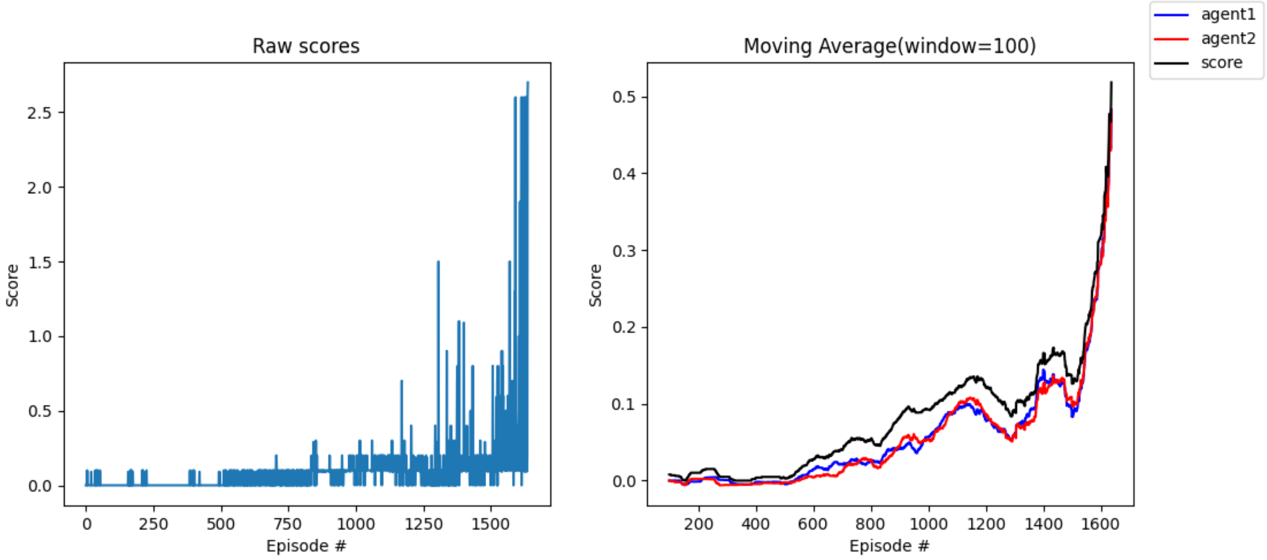

Figure 1: Scores that the agents achieved per episode

I achieved these scores using the following **hyper parameters**:

Parameter	Value	Description
gamma	0.994	discount factor
lr_actor	0.001	learning rate of the actor network
lr_critic	0.0005	learning rate of the critic network
tau	0.001	update factor for soft update
buffer_size	$1 * 10^6$	Size of the experience replay buffer
batch_size	150	minibatch size (how many samples are drawn from the buffer when learning)

The actor networks of both agents had one hidden layer with 64 input features and 64 output features. The critic networks of both agents had one hidden layer with 102 input features and 100 output features

3 Ideas for future improvements

I could improve the learning of the individual DDPG Agents by using algorithms like REINFORCE or Rainbow.