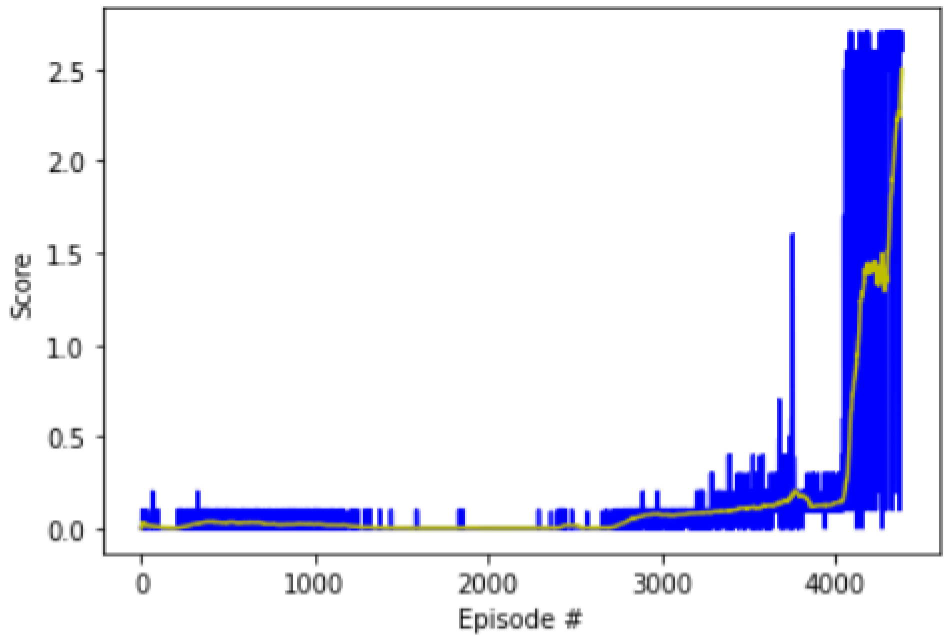
Report

The problem was solved in 4277 episodes. The highest average score achieved over 100 episodes was 2.52 in episode 4679, at which point the program terminates early.



The major parameters are:

BUFFER_SIZE = int(1e6) # replay buffer size

BATCH_SIZE = 256   # minibatch size

GAMMA = 0.99   # discount factor

TAU = 1e-3   # for soft update of target parameters

LR_ACTOR = 1e-4   # learning rate of the actor

LR_CRITIC = 3e-4   # learning rate of the critic

WEIGHT_DECAY = 0   # L2 weight decay for Adam Optimizer

UPDATE_INTERVAL = 25   # Number of episodes elapsed between updates

UPDATE_FREQUENCY = 4   # Number of times parameters and policy will be updated from a sample of experiences in each update

UPDATE_STEP = 0   # Step counter for update (update begins when it hits 0 again after UPDATE_INTERVAL steps)

NOISE_REDUCTION_FACTOR =0.99999 #reduce noise magnitude over time

```
max_t = 2000 #maximum number of training steps per episode
```

It uses a Deep Deterministic Policy Gradient (DDPG) approach, which involves an actor network and a critic network.

The “Critic” estimates the value function. This could be the action-value (the Q value) or state-value (the V value). The “Actor” updates the policy distribution in the direction suggested by the Critic (such as with policy gradients). Both the Critic and Actor functions are parameterized with neural networks.

To foster collaboration between agents, a few modifications are made to the regular DDPG approach:

1.) Shared replay buffer - both agents contribute to the experiences of the buffer and can sample from it.

2.) Shared Critic - both agents are initialized with the same Critic which provides a common baseline.

3.) Shared information of states - each agent has access to states of the ball and the opponent, but from it's own point of view.

I experimented with update interval and frequency, which is the number of replay examples used in each update. The current configuration seems to generate the best results. Basically, I want to do a number of updates each time after a certain number of episodes. These variables also seem to affect the results the most in the second project.

Both actor and critic neural networks consist of multiple linear layers. Their structures mirror those in the Bipedal and Pendulum examples in the class's Github repository.

For future improvement, I will try adding noise to the parameters as well, which can lead to more consistent exploration and a richer set of behaviors (arXiv:1706.01905 [cs.LG]). I will also try priority experience replay: replaying important transitions more frequently can help agent learn more efficiently (arXiv:1511.05952v4 [cs.LG]).