# Project 3 Tennis Report

Alvin (Cheuk Hin) Li

April 2020

## 1 Introduction

The selected tennis environment involves two agents bouncing a ball over a net on two sides of a platform. If the ball is hit over the net without having it touching the ground, +0.1 is rewarded. Otherwise, it loses 0.01 points. The state space has 24 variables per agent, such as the position of the ball and racket. The action space consists of two continuous variables, toward/away from the net, or jumping. The environment is considered solved if there is an average score of +0.5 over 100 episodes (max over both agents).

## 2 Learning Algorithm

Deep Deterministic Policy Gradient is used to combat the problem of continuous spaces as seen in this project [1]. Key Features:

1. Actor-Critic

   The Actor (policy) is defined by $\mu(s|\theta^{\mu})$. The Critic is defined by Q(s,a), similar to Q-learning.
   For this specific case, the Actor network has 3 Dense layers with size 24, 400, 300 respectively. The last layer is put through a tanh activation for continuous output space.
   The Critic network has 3 Dense layers with size 24, 400, 300 respectively.

2. Soft Update

   Since directly implementing Q learning with neural networks is unstable, soft update is used to slowly merge the learned networks back into the target networks. This is achieved by updating the weights with the following equation:

   $$\theta' \leftarrow \tau\theta + (1-\tau)\theta'$$

   where $\tau$ is a very small number.

3. Replay Buffer

   The buffer is used for sampling. Once the cache is full, the old samples are discarded.

4. Noise

   Noise is added to facilitate the exploration of the continuous space. The Ornstein-Uhlenbeck process is used to generate noise and add to the actor policy.

Batch normalization was recommended to improve training speed but was found to decrease scores. Thus it was not used.

The following hyperparameters were used:

- BUFFER SIZE: $1e5$
- BATCH SIZE: 128
- GAMMA: 0.99
- TAU: $1e-3$
- LR ACTOR: $8.5e-5$
- LR CRITIC: $2e-4$
- WEIGHT DECAY: 0
- UPDATE EVERY: 5
- of UPDATES per UPDATE = 1

Attempted to merge both agent's experiences in the same buffer (add and sample in the same buffer), but it takes too long to train the episodes that way, with not much significant improvements.

## 3 Plot of Rewards

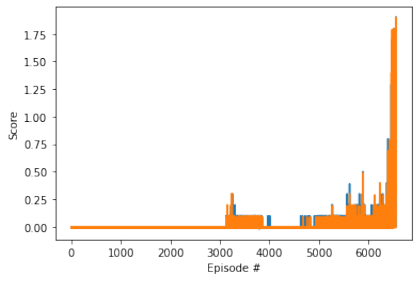

Figure 1: Rewards per Episode

The environment is solved in 6561 episodes.

## 4 Ideas for Future Work

Consider using a more complex model for Actor and Critic. Proximal Policy Optimization (PPO) Algorithm can also be utilized [2]. While prioritized experience replay was implemented, it was not used in the training process as it takes up more time per episode. The use of that can be further investigated [3].

## 5 References

[1] https://arxiv.org/abs/1509.02971
[2] https://arxiv.org/abs/1707.06347
[3] https://arxiv.org/abs/1511.05952