Report for Tennis Project

1. Introduction

This report introduces the algorithm for this Tennis project. Detailed algorithm is shown in section 2. Results are discussed at section 3. Future ideas are shown at the last section.

2. Approach

The algorithm used in this project is modified from project 2. Deep Deterministic Policy Gradient (DDPG)[1] is used for this multi-agent system. In this environment, although there are two agents, it can be figured out that to achieve a high score, the policy for both of them should be the highly similar or the same. Consequently, one shared actor network is used for both of the agents as well as the critic network. The update of the actor network is mainly guided by the results from [2]. The update of the critic network follows [3]. Experience replay and target networks are used for both of the two updates to increase the training stability.

Detailed algorithm is shown below.

Initialize the critic network $Q(s,a|\theta^Q)$ with random parameters θ^Q

Initialize the target critic network $Q'(s,a|\theta^{Q'})$ with $\theta^{Q'} = \theta^Q$

Initialize the actor network $\mu(s|\theta^\mu)$ with random parameters θ^μ

Initialize the target actor network $\mu'(s|\theta^{\mu'})$ with $\theta^{\mu'} = \theta^\mu$

Initialize the replay memory *D*

For episode = 1, M **do**

Get initial state $s_{1,1}, s_{1,2}$ from environment

For t = 1, T **do**

Select action $a_{t,1} = \mu(s_{t,1}|\theta^\mu)$ and $a_{t,2} = \mu(s_{t,2}|\theta^\mu)$ according to the current policy plus noise

Execute action $a_{t,1}$and $a_{t,2}$ on the environment

Observe next state $s_{t+1,1}, s_{t+1,2}$ and reward $r_{t,1}, r_{t,2}$ from environment

Store the transition $(s_{t,1}, a_{t,1}, r_{t,1}, s_{t+1,1})$ and $(s_{t,2}, a_{t,2}, r_{t,2}, s_{t+1,2},)$ in *D*

Sample random minibatch (size *N*) of transitions (s_i, a_i, r_i, s_{i+1}) from *D*

Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$

Update critic network $Q(s,a|\theta^Q)$ by minimizing the loss: $L = \frac{1}{N}\sum_{i=1}^{N}(y_i - Q(s_i,a_i|\theta^Q))^2$

Update the actor network $\mu(s|\theta^\mu)$ using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_a Q(s,a|\theta^Q)|_{s=s_i,a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

Update target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

End For

End For

---

3. Results

For simplicity, the actor network and the critic network have very similar structure. They both have 2 hidden layers with 128 and 96 hidden units. Leaky ReLU is used as activation function in the hidden layers. For the critic network, there is no activation function in the output unit. For the actor network, tanh is used as activation function to provide a bounded action (-1,1). According to my parameter tuning, the most important part for this environment are the learning rates. After long times of parameter tuning, the learning rate for the actor is 0.00015 and 0.001 for the critic. The results are shown in Fig. 1. The average score from episodes 5030 to 5130 is **0.68**. The performance can be further improved by continuing tuning the parameters.

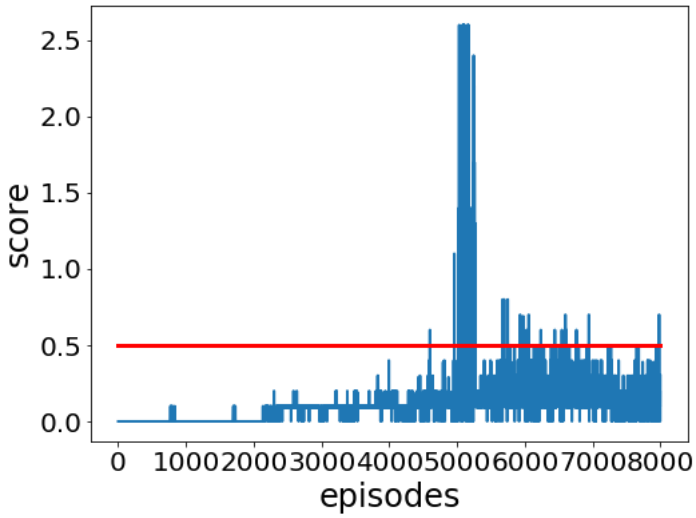

Figure 1 Learning curve

4. Future work

First, the parameters can be further tuned. Second, recent development related to DDPG like D4PG[4] can be tried.

Reference

[1]. Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." *arXiv preprint arXiv:1509.02971*(2015).
[2]. Silver, David, et al. "Deterministic policy gradient algorithms." *ICML*. 2014.
[3]. Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." *Nature* 518, no. 7540 (2015): 529.
[4]. Barth-Maron, Gabriel, et al. "Distributed Distributional Deterministic Policy Gradients." *arXiv preprint arXiv:1804.08617* (2018).