Project3: Collaboration and Competition

Mengheng Xue

July 14, 2019

1 Introduction

In this report, I will discuss the learning algorithm, along with the chosen hyper-parameters, my model architectures for neural network, and algorithm performance in my collaboration and competition project.

2 Learning Algorithm

To solve this ccollaboration and competition problem, I used Deep Deterministic Policy Gradients (DDPG) algorithm with modification to suit for multiagent environment.

DDPG is an approximate Actor-Critic Method, which extends DQN to work in continuous spaces. So the agent is composed of two neural networks, one as *actor* and the other as *critic*. Similar network architectures are used for both actors and critic which will be discussed in the following section.

2.1 Hyper-parameter Settings

- $BUFFER\_SIZE = 1000000$. It defines the replay buffer size. Buffer is an object that contains tuples called experiences composed by *state, actions, rewards, next states* and *dones*, which is necessary information for learning.
- $BATCH\_SIZE = 1024$. It defines the mini-batch size. When the number of experiences in the replay buffer exceeds the batch size, the learning method is called.
- $TAU = 0.001$. It controls the model soft updates which is used for slowly changing the target networks parameters slowly, improving stability.
- $LR\_ACTOR = 0.0001$ and $LR\_CRITIC = 0.001$. They are optimizer learning rates which control the gradient ascent step. Thus, it will control the speed of learning process.
- $WEIGHT\_DECAY = 0.0001$. It is the L_2 regularization parameter of the optimizer.
- $GAMMA = 0.99$. It is the discount factor, which controls the importance of the future rewards versus the immediate ones.

2.2 Neural Network Architecture

2.2.1 Actor

Neural network for actor is consisted of input layer with 24 nodes representing state inputs and two hidden layers with nodes $[128, 64]$. The activation function for each hidden layer is ReLU, which will accelerate the training process. Also batch normalization is also applied for each hidden layers to avoid overfitting problem. The activation function for output layer with 2 nodes representing action outputs is Tanh, which will make possible tackling continuous action spaces. This network takes the states as inputs and outputs the best calculated action for the input states.

2.2.2 Critic

For Critic, neural net structure is consisted of $24 * 2 = 48$ nodes as input layer which represent input for local states of both agents and two hidden layers with nodes $[128, 64]$. Similar as Actor, each hidden layer using ReLU activation function. This network takes the states as inputs and outputs the action value function for the best action outputted by the Action network. Therefore, the output layer contains only 1 node to represent Q-value output.

3 Performance

We can see that our model can solve the this problem in 1110 episodes and achieve average score (over 100 episodes) of +0.51. The average scores of last 100 episodes and each episode scores are shown in the following plot.

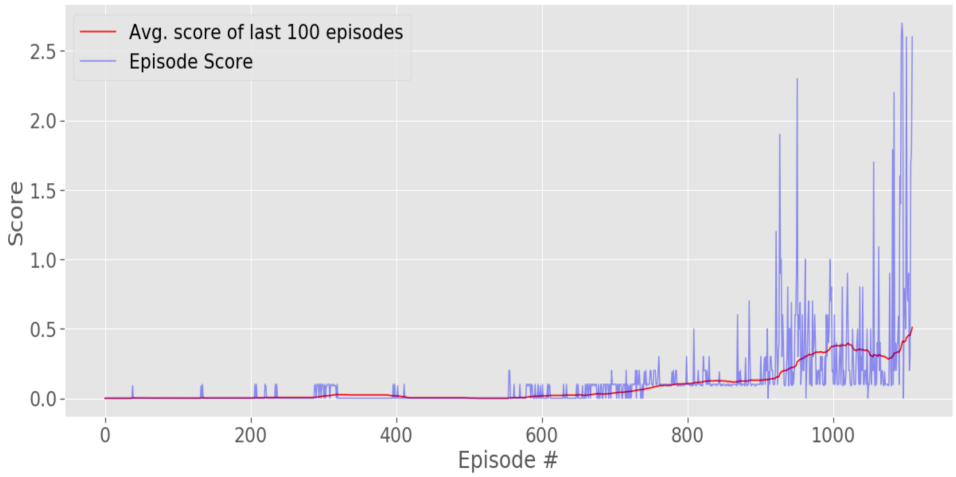

Fig. 1: moving average scores and each episode scores for our final model

4 Future Work

Different models such as Proximal Policy Optimization (PPO) can be tested to compare the performance with our DDPG approach. Also we could try to modify our current model using Multi-Agent DDPG (MADDPG) algorithm [1] considering the collaborative self-play agents. The main difference is that in MADDPG, actor-critic methods considering action policies of other agents, which is expected to improve learning performance under complex multi-agent coordination. Lastly, our DDPG algorithm is apt for environments like Soccer. We could implement the algorithm in different environment to evaluate the performance.

References

[1] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.