Report

Learning Algorithm (using DDPG)

Initialize Actor Network with weight θ and Critic Network ω

Initialize target Actor Network and Critic Network with weights $\theta\_$ and $\omega\_$

Initial replay buffer R

For each episode do

Initialize a random process for action exploration

Initial observation state

While

Choose action a_t from s_t using policy derived from current θ also plus exploration noisy

Execute action a_t and observe new state s_{t+1}

Store transition (s_t, a_t, r_t, s_{t+1}) in R

Sample a random minibatch transitions (s_i, a_i, r_i, s_{i+1}) from R

Evaluate target value using $\omega\_$ where choose action a_{i+1} from s_{i+1} using $\theta\_$

Update critic by minimizing MSE loss function

Update actor policy using sampled policy gradient

Soft update the $\theta\_$ and $\omega\_$

Until done

End For

We use the replay buffer that contain both agent experiences in it. They share experience for each other.

We use the epsilon-greedy method, and the decay is 0.9999. Every episode end, we update the value of epsilon.

Hyperparameters

epsilon = 1.0

decay = 0.9999

actor_alpha = 1e-4

critic_alpha = 1e-4

tau = 1e-3

gamma = 0.99

batch_size = 64

max_memory_size = 50000

update_every=4

Model Architecture

Actor network:

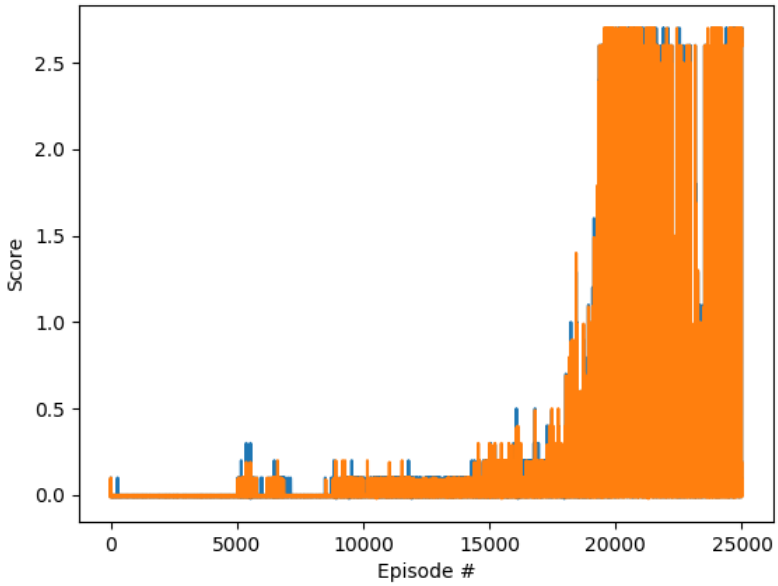
Input: state_size→fc1:64 →ReLU→fc2:128→ReLU→fc3: action_size→tanh

Critic network:

Input: state_size→fc1:64 →ReLU→fc2:128→ReLU→256→fc4: 128→ ReLU→fc5: 1

Input: action_size→fc3:128→ReLU

Plot of Rewards




```
No.100 score this episode: -0.0020,
......
No.7000 score this episode: 0.0359,
No.7100 score this episode: 0.0245,
No.7200 score this episode: 0.0165,
No.7300 score this episode: 0.0130,
No.7400 score this episode: 0.0325,
No.7500 score this episode: 0.0240,
No.7600 score this episode: 0.0240,
No.7700 score this episode: 0.0280,
No.7800 score this episode: 0.0210,
No.7900 score this episode: 0.0270,
No.8000 score this episode: 0.0325,
No.8100 score this episode: 0.0410,
No.8200 score this episode: 0.0440,
No.8300 score this episode: 0.0385,
No.8400 score this episode: 0.0360,
No.8500 score this episode: 0.0345,
No.8600 score this episode: 0.0405,
No.8700 score this episode: 0.0460,
No.8800 score this episode: 0.0355,
No.8900 score this episode: 0.0420,
No.9000 score this episode: 0.0475,
No.9100 score this episode: 0.0435,
No.9200 score this episode: 0.0380,
No.9300 score this episode: 0.0485,
No.9400 score this episode: 0.0530,
```

```
No.9500 score this episode: 0.0565,
No.9600 score this episode: 0.0610,
No.9700 score this episode: 0.0505,
No.9800 score this episode: 0.0465,
No.9900 score this episode: 0.0560,
No.10000 score this episode: 0.0490,
No.10100 score this episode: 0.0575,
No.10200 score this episode: 0.0550,
No.10300 score this episode: 0.0480,
No.10400 score this episode: 0.0565,
No.10500 score this episode: 0.0620,
No.10600 score this episode: 0.0590,
No.10700 score this episode: 0.0645,
No.10800 score this episode: 0.0870,
No.10900 score this episode: 0.0965,
No.11000 score this episode: 0.0980,
No.11100 score this episode: 0.1010,
No.11200 score this episode: 0.1290,
No.11300 score this episode: 0.1270,
No.11400 score this episode: 0.0975,
No.11500 score this episode: 0.1490,
No.11600 score this episode: 0.1250,
No.11700 score this episode: 0.0985,
No.11800 score this episode: 0.1450,
No.11900 score this episode: 0.2020,
No.12000 score this episode: 0.1435,
No.12100 score this episode: 0.1630,
No.12200 score this episode: 0.1795,
No.12300 score this episode: 0.2590,
No.12400 score this episode: 0.2485,
No.12500 score this episode: 0.2015,
No.12600 score this episode: 0.2541,
No.12700 score this episode: 0.3976,
No.12800 score this episode: 1.1521,
```

Future Work

In the Future, we consider to reduce the convergent time in the training process. So, we want to try the multiple-agent DDPG algorithm to play this game. We can also use prioritized experienced replay buffer to learn more

effectively instead of sampling experience transitions uniformly from a replay memory. The intuition behind that is the more important transitions should be sampled with higher probability we consider to use and try to reduce the convergent time.

References

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.