# Learning algorithm

## Overview of the technique

The agent is implemented using the Multi-agent Deep Deterministic Policy Gradient (MADDPG for short) method. The paper for this algorithm can be found here:   
[[1706.02275v4] Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments (arxiv.org)](https://arxiv.org/abs/1706.02275v4)  
This algorithm is a multi-agent version of the Deep deterministic policy gradient (DDPG). A quick recap, DDPG is an algorithm which concurrently learns a Q-function and a policy. It uses off-policy data and the Bellman equation to learn the Q-function, and uses the Q-function to learn the policy. DDPG is an off-policy algorithm and can only be used in continuous action space.  
MADDPG, or Multi-agent DDPG, extends DDPG into a multi-agent policy gradient algorithm where decentralized agents learn a centralized critic based on the observations and actions of all agents. It leads to learned policies that only use local information (i.e. their own observations) at execution time, does not assume a differentiable model of the environment dynamics or any particular structure on the communication method between agents, and is applicable not only to cooperative interaction but to competitive or mixed interaction involving both physical and communicative behavior. The critic is augmented with extra information about the policies of other agents, while the actor only has access to local information. After training is completed, only the local actors are used at execution phase, acting in a decentralized manner.

The algorithm in detail:

Text

Description automatically generated

## Network architecture

The network architecture for actor and critic are described below (the number is # neurons at each hidden layers)

Chart, waterfall chart

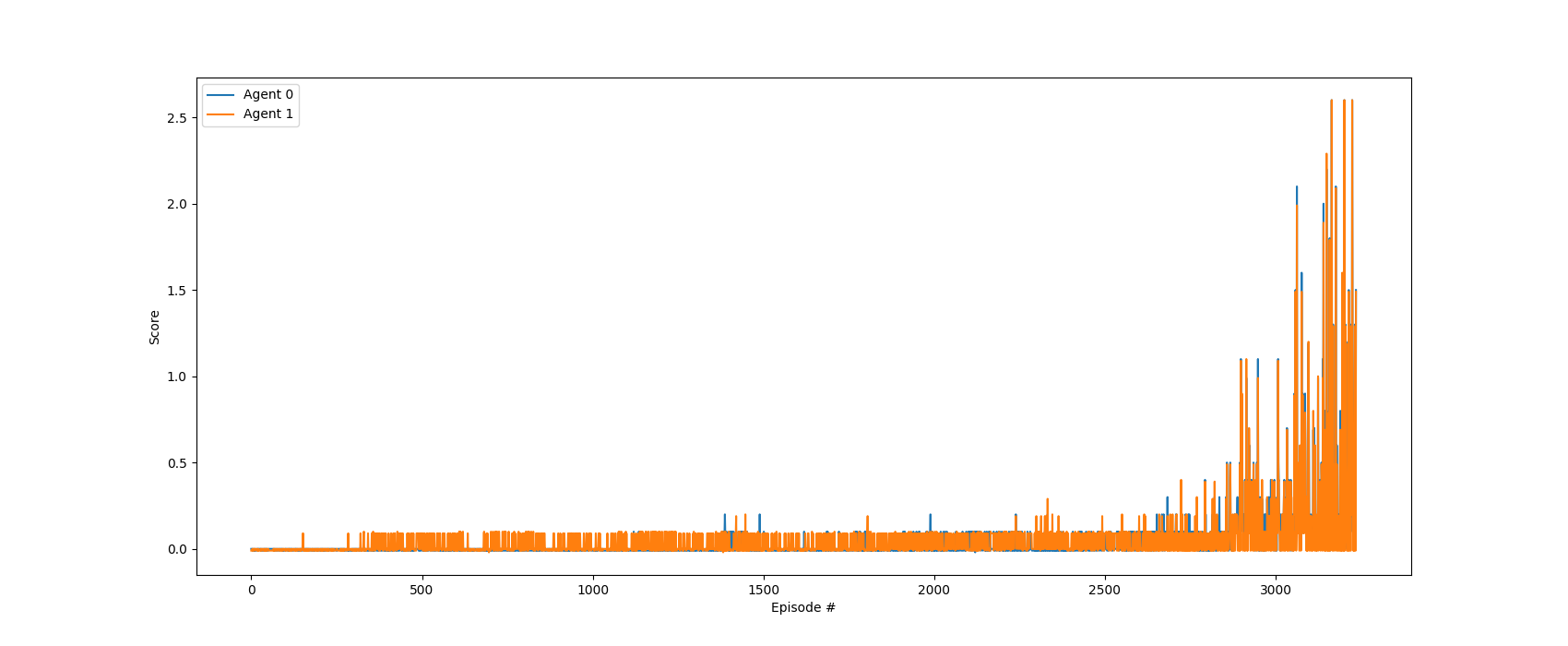
Description automatically generatedChart

Description automatically generated

## Hyperparameters select

|  |  |  |
| --- | --- | --- |
| Name | Value | Meaning |
| seed | 37 | Random seed for stable result |
| actor\_lr | 0.00001 | Learning rate for actor’s Adam optimizer |
| critic\_lr | 0.0001 | Learning rate for critic’s Adam optimizer |
| gamma | 0.95 | Discount factor |
| tau | 0.01 | Soft update target network hyperparameter |
| buffer\_size | 1000000 | Number of transitions stored in replay buffer |
| batch\_size | 128 | Batch size for learning phase of the agent |

# Plot of rewards



Plot of rewards in training. Blue line indicates accumulate reward of the first agent while the orange is for the other.

# Ideas for future works

The result I have shown is acceptable, which solve the problem in around 3000 episodes. However, the result is a bit unstable during training. So with that said, I think some modifications will make the result even better:

* Tuning hyperparameters such as learning rate and the architecture of the actor and critic network probably enhance the learning ability of the agent
* Maybe I can do a multi-agent with this centralized-critic and decentralized-actor framework for other better single algorithms than DDPG such as TD3, PPO or SAC.