Learning Algorithm

The learning algorithm is (Multi Agent Deep Deterministic Policy Gradient) MADDPG, which extends DDPG algorithm to Multi Agent scenario.

The algorithm builds upon a actor-critic framework, where the actor outputs best actions and the and the critic estimates the Q-value function. In MADDPG each agent has a unique actor, but shares a centralised critic which takes into account observations and actions of all agents(centralised training)

Exploration is added through noise to the deterministic action, implemented through the class OUNoise (hyperparams standard_deviation=0.2)

Experience replay allows breaking correlations of sequential actions.We use a replay buffer of size 1e6

DDPG uses target networks both for actor and critic, this stabilises learning by not updating the target during a few iterations.

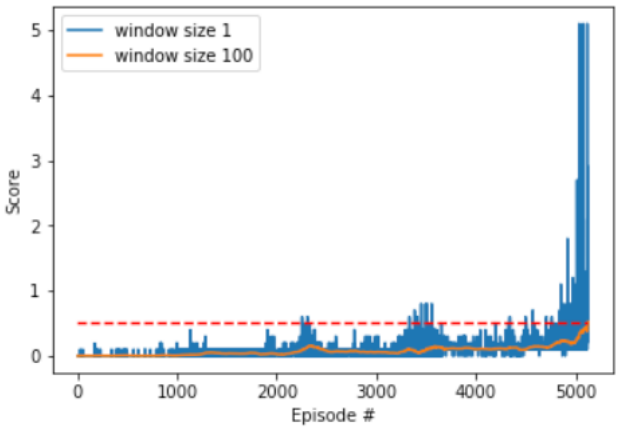
One of the main features of DDPG is the soft updates, which brings stabilisation by gradually updating the target network. This is done through the TAU = 1e-3

Other hyperparams used are:

BATCH_SIZE = 256 # minibatch size
GAMMA = 0.99 # discount factor
LR_ACTOR = 1e-4 # learning rate of the actor
LR_CRITIC = 1e-3 # learning rate of the critic
WEIGHT_DECAY = 0 # L2 weight decay

Actor Architecture Layers Size: [400, 300, 4]
Critic Architecture Layers: [128, 260(|256+4), 1]

Plot of Rewards



Ideas for Future Work

- 'Hyperparameter Tuning' - Automated strategy to find the best hyperparameters, I have just modified slightly what we learn in the lessons
- Comparison Across other models… PPO, A3C…