Deep Reinforcement Learning Nanodegree

Project #3: Navigation

Implementation

Brief description: This coding project implements the Multi-Agent Deep Deterministic Policy Gradient architecture to allow 2 agents to maintain a rally in a Tennis environment.

Model architecture: The base Actor and Critic networks are defined in *model.py*. The Actor network consists of 3 fully connected layers, with node sizes of 400, 300, and 2 respectively. A rectified linear unit activation (**ReLU**) function is used between each layer. Batch normalization is applied after the first layer. The final layer has a tanh function applied, enabling it to work with continuous action spaces. The Critic network consists of an identical number of layers, with node sizes of 400, 300 and 1 instead. Batch normalization is applied after the first layer, but there is no ending tanh function.

Deep Deterministic Policy Gradient: The method used for this project is known as DDPG, and it arises from a combination of the Deep Q-Network and actor-critic approaches to use non-linear function approximators for the optimal policy. The critic is meant to estimate the appropriate action-value function for the environment, while the actor updates the distribution of probabilities in the policy used to interact in the environment based on the critic's function. Both networks have local and target versions, to alleviate the issue of chasing a moving target. Both use the Adam optimizer, a tweaked version of stochastic gradient descent, retaining a similar goal to the DQN - minimize mean squared error between the target and local networks. The target network is updated using a soft update so it is not fully overwritten, the amount of the target network which is preserved is determined by the hyperparameter tau. The implementation can be seen in *model.py* and ma*ddpg_agent.py*.

Hyperparameters:
BUFFER_SIZE = int(1e5) BATCH_SIZE = 128
GAMMA = 0.99 TAU = 2e-3
LR_ACTOR = 1.5e-3
LR_CRITIC = 1.5e-3
WEIGHT_DECAY = 0

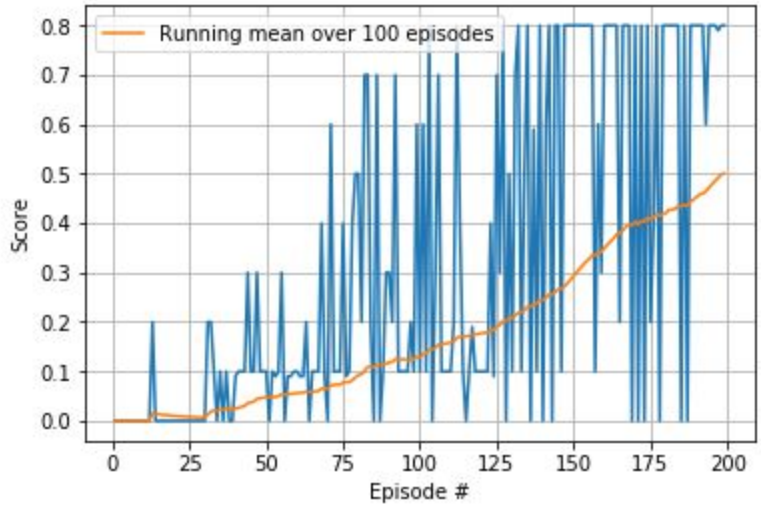
LEARN_EVERY = 1

LEARN_NUM = 10
GRAD_CLIPPING = 1.0

OU_SIGMA = 0.1
OU_THETA = 0.15
EPSILON = 1.0
EPSILON_DECAY = 1e-6

Scores While Training

```
Environment solved in 99 episodes!      Average Score: 0.5006
```



Future improvements:

- Prioritized experience replay
- Hyperparameter tuning and architecture tweaking
- Implementation of framework described in Multi-Agent Actor-Critic for Mixed
- Cooperative-Competitive Environments paper