It is better to apply Deep Deterministic Policy Gradients (DDPG) over Deep Q Network (DQN). Because DDPG works on continuous state and action space whereas DQN is meant to solve discrete action space problems.

Critic Value Model computes the Q-values for any given (state, action) pair. After that, the gradient of this Q-value is computed with respect to the corresponding action vector which is then fed in as input for the training of the Actor Policy Model.

Since there are two networks, that's why the model is sample efficient.

Hyper-parameters;

BUFFER_SIZE = int(1e6) # replay buffer size

BATCH_SIZE = 1024   # minibatch size

GAMMA = 0.99   # discount factor

TAU = 1e-3   # for soft update of target parameters

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's layer units: 128, 64

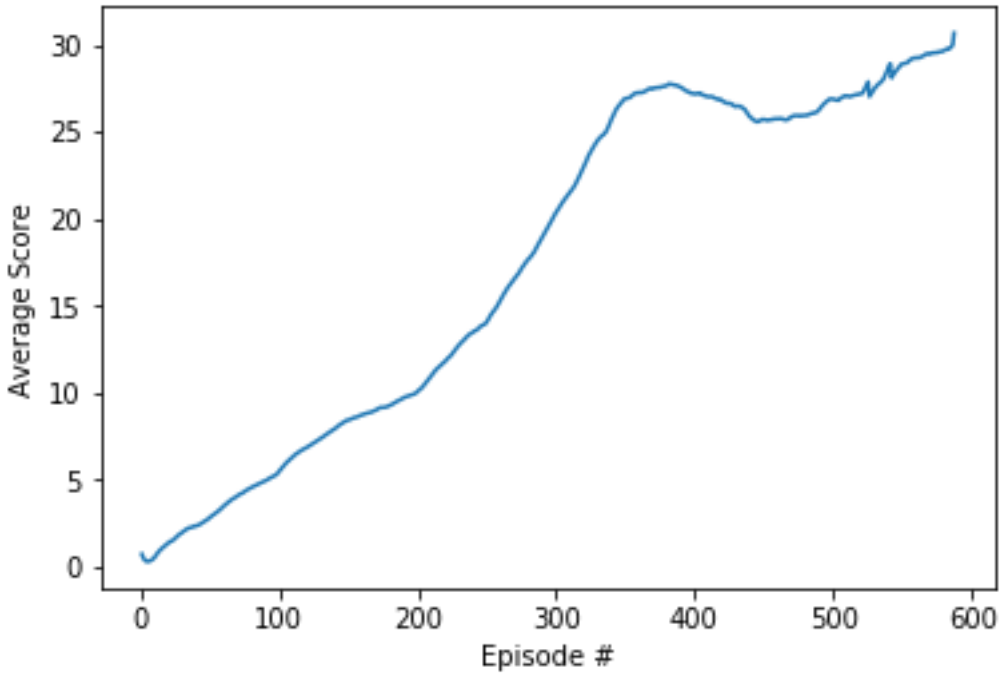
Critic's layer units: 128, 64

Xavier weight initialization used for the hidden layers of both networks

ReLU activation used

Batch normalization applied on the Critics input layer.

Environment solved in 589 episodes with the average Score: 30.66 over 100 episodes.



The graph shows average score over the last 100 episodes in the y-axis. The total number of episodes in the x-axis.

Future Work

I'd like to try out algorithms like PPO, A3C, and D4PG that use multiple (non-interacting, parallel) copies of the same agent to distribute the task of gathering experience.

I'd like to try out dropouts, various weight initialization and further hyper-parameters like cost functions that may yield better results.