Report

Learning Algorithm

The underlying algorithm chosen to solve this problem was a deep q network. Very similar to Q-learning, this algorithm takes advantage of a neural network to determine the necessary action given the continuous state it is in.

Functioning as a Q-learning algorithm at heart, this algorithm steps through different time steps of a given episode and makes decisions based off the max value of the networks outputs. The algorithm decides it’s time to learn when enough of a particular state samples are recorded in memory.

The states/next states that are in memory are then used to calculate a reward value, given the set of circumstances the state is in – then fed into a MSE loss algorithm to train against the current state of the network.

The network has two hidden layers that both have 64 nodes (with being input 37, output 4). All layers are fully connected and uses Adam as a gradient decent algorithm.

Hyperparameters:

BUFFER_SIZE = int(1e5) # replay buffer size

BATCH_SIZE = 64   # minibatch size

GAMMA = 0.99   # discount factor

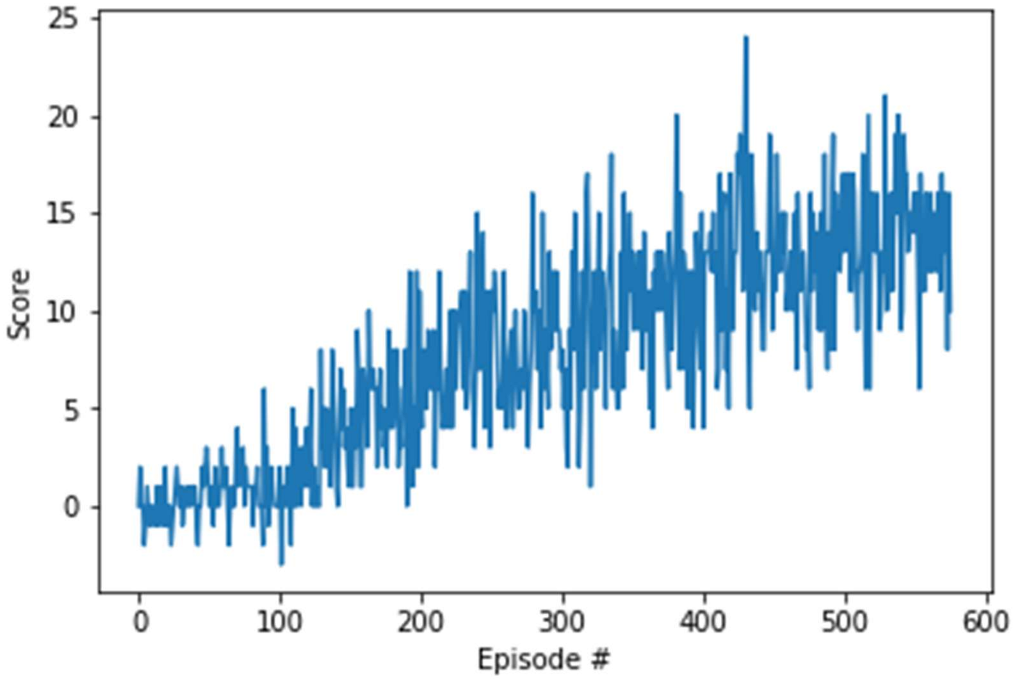
TAU = 1e-3   # for soft update of target parameters

LR = 5e-4   # learning rate

UPDATE_EVERY = 4   # how often to update the network

Plot of Rewards

Solves in under 600 episodes



Future Work

- New algorithm implementations i.e. double dqn, a dueling dqn, and/or a prioritized experience replay
- Different hyper parameters for the network
- The raw pixel challenge implementation would also be a fun challenge