1. Learning Algorithm:

The source code and weights of a trained agent are placed in the main directory. The algorithm is implemented following the next guidelines:

- The reinforcement learning algorithm is the Double DQN (DDQN)

- Neural Network configuration:
  The configuration refers to both networks.

  - Hidden layers are composed of State -> 64 -> LeakyReLU -> 64 -> LinearReLU -> Action
  - The optimizer is ADAM and
  - The loss metric to be optimized is mean squared error (MSE)
  - The learning rate = 5e-4

- Other algorithm configuration:
  The UPDATE_EVERY parameter determines how many steps take place until the next update. The TAU is the rate at which this update takes place so that the network weights do not change suddenly but smoothly.

  - BATCH_SIZE = 64 # minibatch size to be processed by the neural network
  - GAMMA = 0.99 # discount factor for future expected rewards
  - UPDATE_EVERY = 4 # how often, measured in steps, to update the network
  - TAU = 1e-3 # for soft update of target parameters TAU is the rate at which this update takes place so that the network weights do not change suddenly but smoothly.

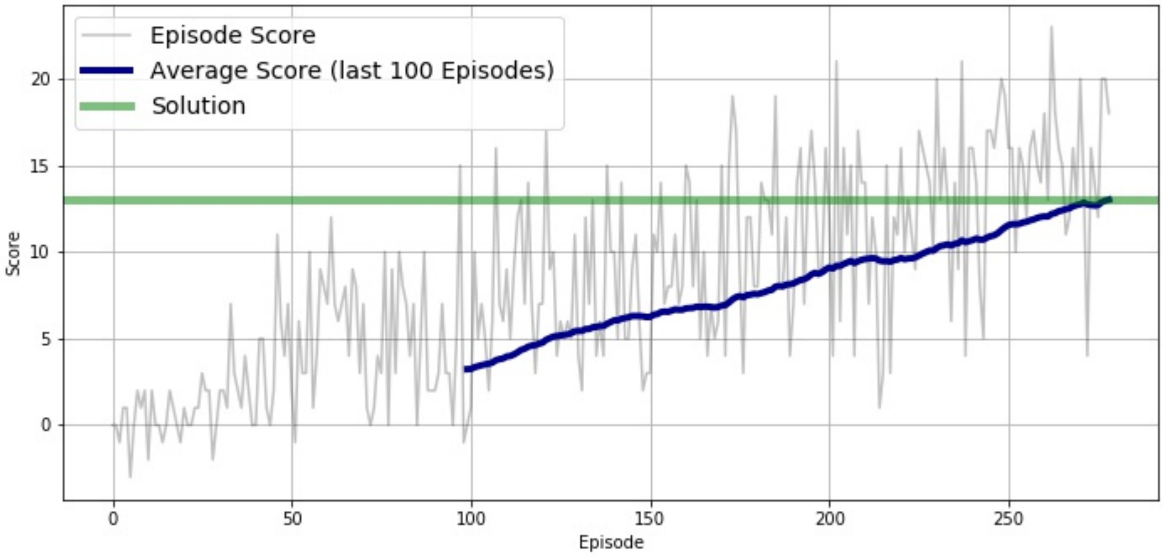
- Experience Replay is also implemented. In this technique, DDQN model is trained by mini-batch from a replay buffer with a size 1e5

- Agent selects next action based on Epsilon Greedy. The value of epsilon is set initially to 1, and decreases at a rate of epsilon_decay = 0.95 with time until 0.000001.

The parameters just mentioned can be of course changed. These have been empirically found to work well for a rapid convergence.

2. Agent Results:

The following plot of rewards shows the training results. At Episode 278, agent performance met the criteria and stopped training. (mean scores of last 100 episodes is above +13).



3. Future Work:

An improvement to be done is to implement a prioritized experience replay. Those states and actions that produced the largest errors are being prioritized in order to get the most benefit in learning from the experience replay.

The “rainbow” example from the lecture includes a learning algorithm with additional features such as like “Dueling DQN”, “multi-step bootstrap targets”, Distributional DQN and Noisy DQN.