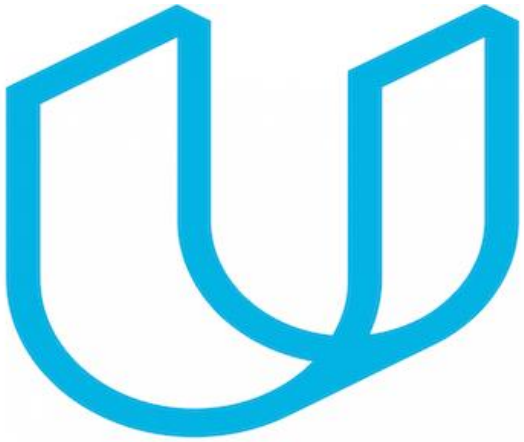


UDACITY

DEEP REINFORCEMENT LEARNING NANODEGREE UDACITY

Navigation Project

1. Algorithm Details

    1. Initialize the environment and the Q-network.

    2. At each timestep:

        2.1. Select an action using an epsilon-greedy policy.

        2.2. Execute the action in the environment.

        2.3. Store the experience (state, action, reward, next_state, done) in the replay buffer.

        2.4. Sample a random batch of experiences from the replay buffer.

        2.5. Compute the target Q-values using the target network.

        2.6. Update the Q-network by minimizing the loss between the predicted Q-values and the target Q-values.

    3. Periodically update the target network to match the Q-network.

    4. Repeat until the environment is solved.

2. Network Architecture

    Q-Network: A fully connected neural network with:

        Input layer: 37 units

        Hidden layers: Two hidden layers with 128 units each, ReLU activation

        Output layer: 4 units

3. hyperparameters

    The following hyperparameters were used:

    BUFFER_SIZE = 1e5 # replay buffer size

    BATCH_SIZE = 64 # minibatch size

    GAMMA = 0.99 # discount factor

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

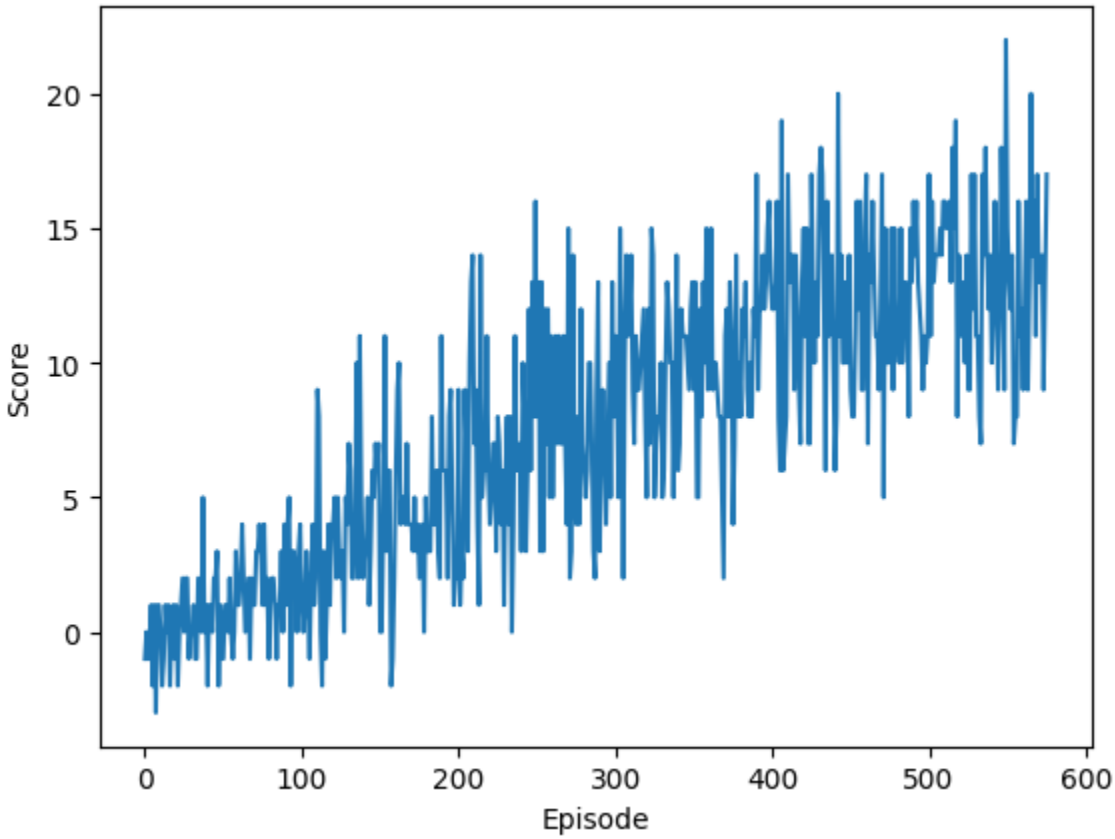
    LR = 5e-4 # learning rate

    UPDATE_EVERY = 4 # how often to update the network

4. Plot of Rewards

I needed 476 episodes to solve the environment:

```
Episode 100     Average Score: 0.90
Episode 200     Average Score: 3.85
Episode 300     Average Score: 7.21
Episode 400     Average Score: 10.05
Episode 500     Average Score: 11.94
Episode 576     Average Score: 13.04
Environment solved in 476 episodes!     Average Score: 13.04
```



5. Ideas for Future Work

To improve convergence speed, the developments covered in the dual DQN course can be used to help reduce overestimation of action values