Udacity Deep Reinforcement Learning Nanodegree

Project 1: Navigation

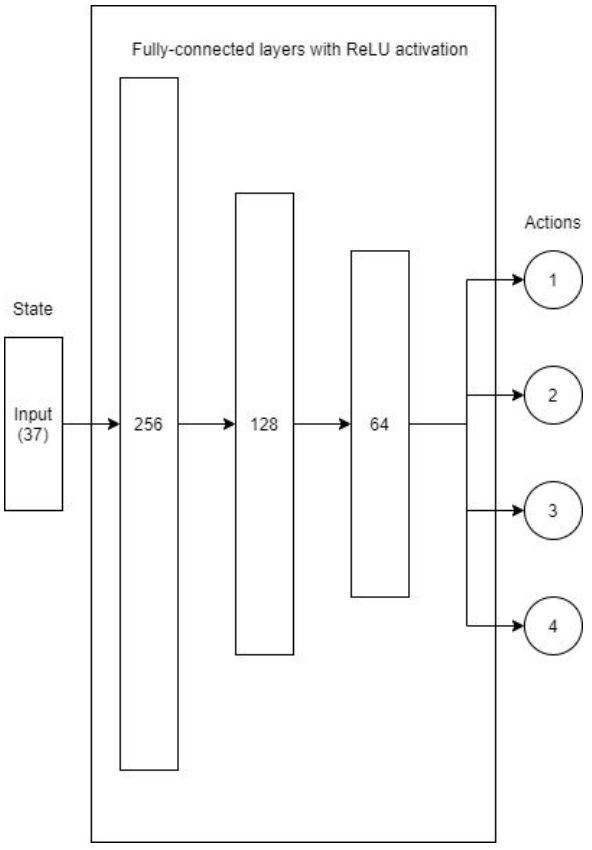
Khanh Nguyen Vu

I. Approaches

For this project, I decided to go for the plain vanilla deep Q learning initially to see where is the baseline, or how good the simplest deep learning approach can achieve. After that, I implemented **double DQN** and **prioritized experience replay**, and both worked like charms.

Q-Network architecture

1. Inputs: 37 units (state_size)
2. Fully-connected layers
    a. Fc1: 256 units (ReLU)
    b. Fc2: 128 units (ReLU)
    c. Fc3: 64 units (ReLU)
3. Outputs: 4 units (linear, action_size)



Optimizer: torch.optim.Adam (LR=5e-4)

Training hyperparameters:

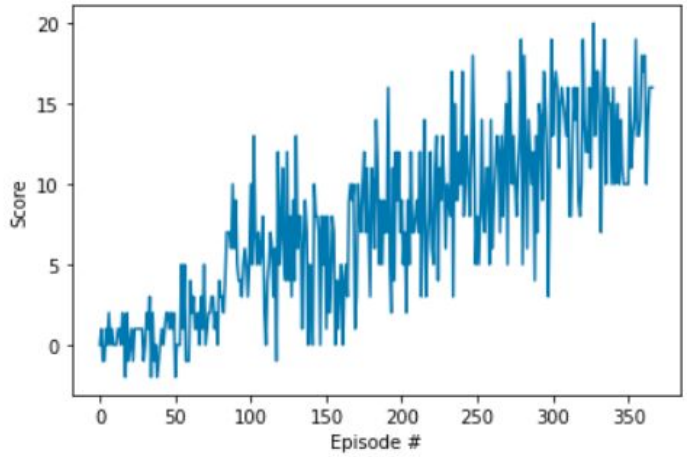
1. Max episode length: 500
2. Epsilon-greedy decay: 0.95 (start=1.0, end=0.01)

Prioritized Experience Replay: I used this framework.

II. Results

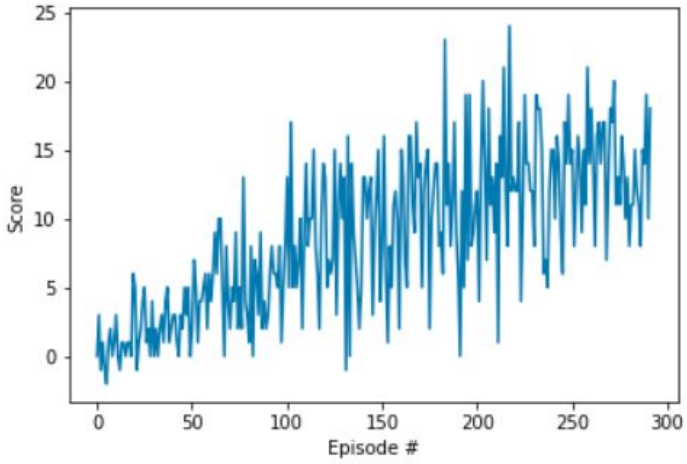
Plotted reward of Double DQN

```
Episode: 367, Eps: 0.02501, Average score: 13.00
```



Plotted reward of Double DQN+Prioritized Experience Replay

```
Episode: 292, Eps: 0.01000, Average score: 13.01
```



There is a clear improvement when using Prioritized Experience Replay on top of Double DQN. It took my agent 292 episodes to solve the task.

III. Ideas for improvements

- Try to implement Duel DQN (or even Rainbow DQN)
- Find a better framework for Prioritized Experience Replay (the one I used was a bit old and buggy)