Udacity Deep Reinforcement Learning Nanodegree

Project 1: Navigation

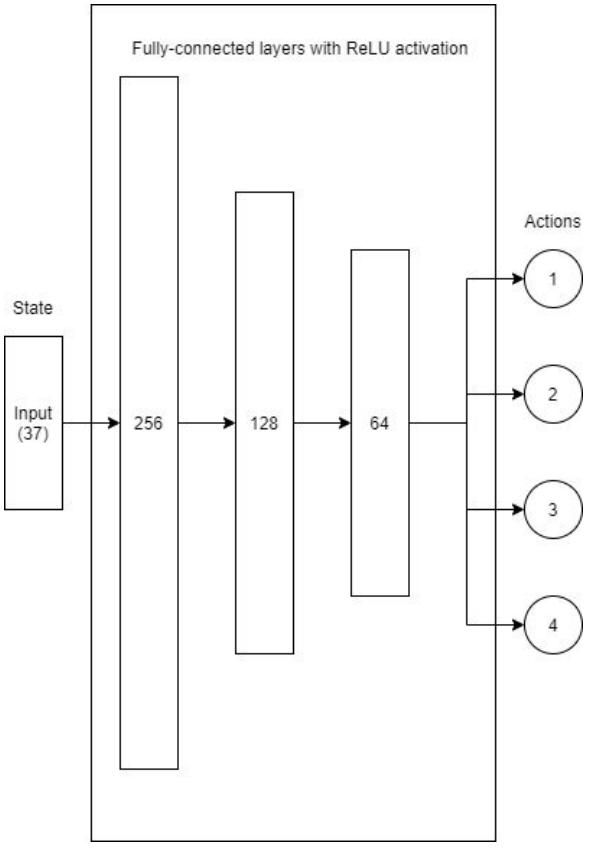
Khanh Nguyen Vu

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 it 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)

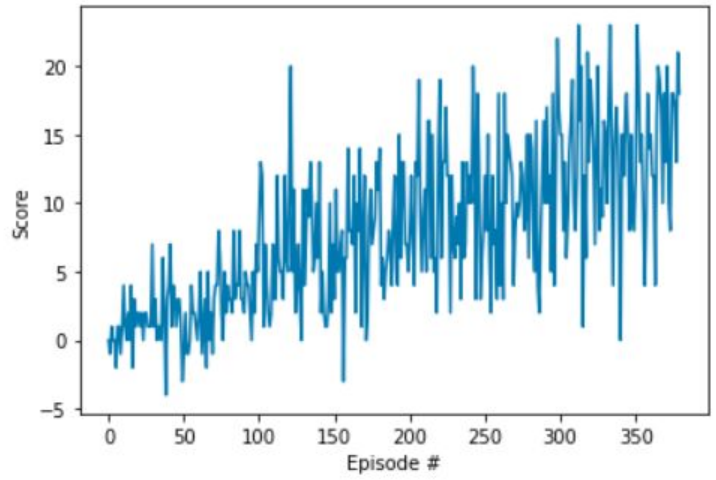
Training hyperparameters:

1. Max episode length: 500
2. Epsilon-greedy decay: 0.95 (start=1.0, end=0.01)
3. Memory buffer size = 100000
4. Batch size = 32
5. Update every = 4

II. Results

Plotted reward of Double DQN

```
Episode: 380, Eps: 0.01000, Average score: 13.11
```



II. Ideas for improvements

- Try to implement Prioritized Experience Replay and Duel DQN (or even Rainbow DQN)
- Develop a general DQN framework that could be used for other environments.