**Report**

**Learning Algorithm** (using DQN with experience replay)

initialize replay buffer D size to N

initialize qnetwork\_local with weights

initialize qnetwork\_target with weights =

For episode =1, n\_episodes do

Initialize state

For t=1, max\_t do

With probability select a random action

Given , get

Store transition (,) in D

Sample random minibatch of transitions (,) from D

Set =

Perform a gradient descent step on with respect to the

Every C steps do soft update =

EndFor

Decrease

EndFor

**Hyperparameters**

batch\_size = 64

eps = 1.0

eps\_end = 0.01

decay = 0.999

max\_memory\_size = 100000

gamma = 0.99

alpha = 5e-4

tau = 1e-3

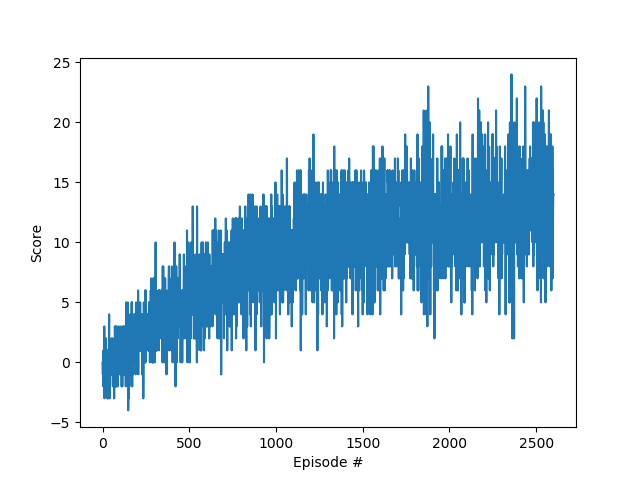
update\_every=4

max\_t=1000

**Model Architecture**

Input: state\_size🡪fc1:128 🡪ReLU🡪fc2:64🡪ReLU🡪fc3:action\_size

**Plot of Rewards**

****

Episode: 100, Average Score: -0.11

Episode: 200, Average Score: 1.27

Episode: 300, Average Score: 2.63

Episode: 400, Average Score: 3.83

Episode: 500, Average Score: 4.58

Episode: 600, Average Score: 5.54

Episode: 700, Average Score: 6.29

Episode: 800, Average Score: 7.78

Episode: 900, Average Score: 8.59

Episode: 1000, Average Score: 8.85

Episode: 1100, Average Score: 9.32

Episode: 1200, Average Score: 10.42

Episode: 1300, Average Score: 9.91

Episode: 1400, Average Score: 10.41

Episode: 1500, Average Score: 10.46

Episode: 1600, Average Score: 10.92

Episode: 1700, Average Score: 11.60

Episode: 1800, Average Score: 11.83

Episode: 1900, Average Score: 12.00

Episode: 2000, Average Score: 11.85

Episode: 2100, Average Score: 12.03

Episode: 2200, Average Score: 12.60

Episode: 2300, Average Score: 12.42

Episode: 2400, Average Score: 12.23

Episode: 2500, Average Score: 12.96

Episode: 2600, Average Score: 13.39

**Future Work**

In order to solve the DQN’s problem about overestimating action values, using double Q-learning can be a better choice.

It can also use prioritized experienced replay buffer to learn more effectively instead of sampling experience transitions uniformly from a replay memory. The intuition behind that is the more important transitions should be sampled with higher probability.

By replacing the traditional Deep Q-Network (DQN) architecture with a [dueling architecture](https://arxiv.org/abs/1511.06581), we can assess the value of each state, without having to learn the effect of each action. The value of most states don’t vary a lot across actions, thus it makes sense to try and directly estimate them. Meanwhile it still need to capture the difference actions make in each state where the advantage function comes in.

**References**

[1] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529.

[2] Van Hasselt, H., Guez, A., & Silver, D. (2016, February). Deep Reinforcement Learning with Double Q-Learning. In AAAI (Vol. 2, p. 5).

[3] Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. arXiv preprint arXiv:1511.05952.

[4] Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2015). Dueling network architectures for deep reinforcement learning. arXiv preprint arXiv:1511.06581.