Learning Algorithm

To solve this project I use and agent with a Deep Q-Network (DQN). DQN uses a neural network to create an optimal action-value function, using the state as an the input and the actions as outputs. This alone may lead to instabilities. We use the original DQN algorithm, where they alleviate this in two ways:

- Experience replay: we create a buffer of S,A,R,S' tuples from the episodes and when we train the network, we select some at random. Doing this we break the correlation between consequential actions.
- Fixed Q-target: To train the network we use a loss function based on the update rule of Q-Learning where we have a target, R + ɣ Q(a,S'), and our current value Q(A,S). If we calculate both with the same network, making the target move as the model learns, we introduce a source of instability. To fix this, we create a copy of the network and freeze its parameters, used to calculate the output, only updating it after the learning step, that is, after every finished batch.

Hyperparameters

```
BUFFER_SIZE = int(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
seed=33
```

I used the hyperparameters from the coding exercise *OpenAI Gym's LunarLander* as reference. Tried a learning rate of 5e-3 and seed=42, but the training failed to converge.

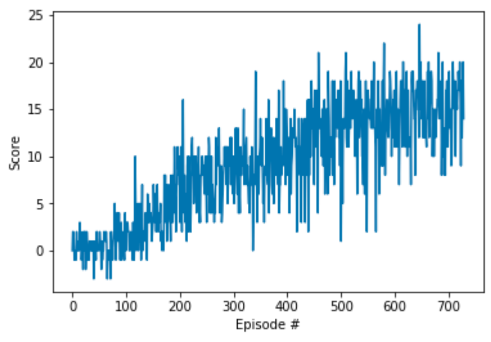
Model architectures

Simple feed-forward network with two hidden layers with 128 and 64 units and ReLU as non-linearity.

Plot of rewards

The environment is solved at an average score of 13. I use 15 as stopping condition to make sure the enviroment is really solved.

```
Episode 100	Average Score: 0.57
Episode 200	Average Score: 3.49
Episode 300	Average Score: 7.51
Episode 400	Average Score: 9.14
Episode 500	Average Score: 11.91
Episode 600	Average Score: 13.56
Episode 700	Average Score: 14.71
Episode 729	Average Score: 15.02
Environment solved in 629 episodes!	Average Score: 15.02
```



Ideas for future work

- Use the improvements to the original algorithm explained in the lessons: double DQN, dueling DQN, and prioritized experience replay.
- More training.
- More complex NN.
- Hyperparameters optimization.