Project

This is the first project of the Deep Reinforcement Learning nanodegree. The goal of the agent is to gather yellow bananas and avoid the blue ones. The environment has 4 discrete actions (the agent's movement, forward, backward turn left and right) and a continuous observation space with 37 values representing the speed and the ray-based perception of objects around the agent's forward direction.

```
Unity brain name: BananaBrain
 Number of Visual Observations (per agent): 0
 Vector Observation space type: continuous
 Vector Observation space size (per agent): 37
 Number of stacked Vector Observation: 1
 Vector Action space type: discrete
 Vector Action space size (per agent): 4
 Vector Action descriptions: , , ,
```

For each yellow banana that is collected, the agent is given a reward of +1. The blue ones give -1 reward. We consider that the problem is solved if the agent receives an average reward (over 100 episodes) of at least +13

Implementation

We have implemented a Deep Q Network with the below architecture:

37 neurons INPUT
128 neurons HIDDEN LAYER
64 neurons HIDDEN LAYER
4 neurons OUTPUT

We use ELU as an activation function. The optimizer used is Adam.

The hyperparameters chosen to reach the goal after some trials are:

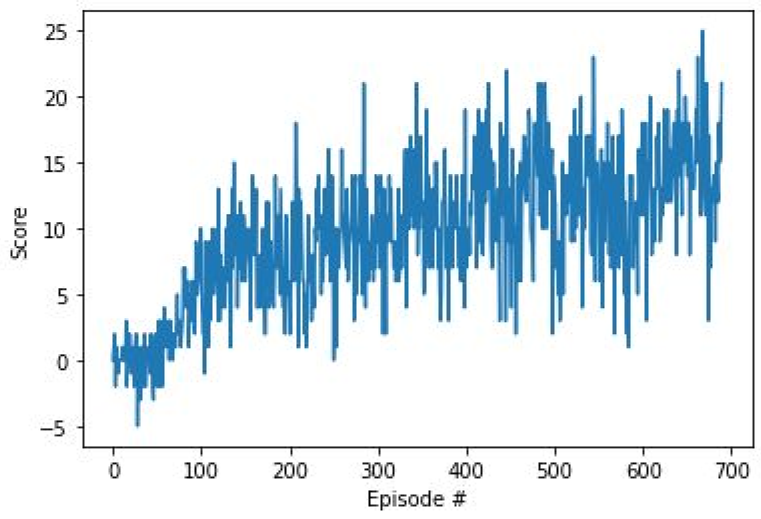
```
BUFFER_SIZE = 10000     # replay buffer size
BATCH_SIZE = 64         # minibatch size
GAMMA = 0.99            # discount factor
TAU = 0.05              # for soft update of target parameters
LR = 5e-4               # learning rate
UPDATE_EVERY = 4        # how often to update the network
EPSILON_START = 1       # epsilon start value
EPSILON_MIN = 0.001     # epsilon min value
```

```
EPSILON_DECAY = 0.99    # epsilon decayment value
N_EPISODES = 2000       # number of episodes
MAX_TIMESTEPS = 1000    # max number of timesteps per episode
GOAL = 14               # goal to consider the problem solved
```

We have used the following techniques to improve the behaviour of our agent:

- Soft update parameters
- Epsilon-greedy
- Gamma decayment
- Fixed Learning rate
- Experience Replay
- Fixed Q Target

Plot of rewards *(averaged over 100 episodes)*



We reached our goal in about 800 episodes.

Future ideas

In order to improve our model, it would be nice to perform a grid search to find the best hyperparameters. Other improvements related to the DQN can be implemented such as Double DQN, Dueling DQN, and/or prioritized experience replay. In fact, we can implement the Rainbow paper to include a set of demonstrated improvements that can make our model better.

Another idea would be to replace the information provided by Unity (observation space) and learn from the raw pixel instead. That would imply the use of a Convolutional Neural Network to apply filters and find patterns directly from the images.