Navigation Problem Report

In this exercise, I used a Deep Q-Network Agent to control a unity environment brain, that in turns controls a first-person agent that must collect a specific object in a square continues environment. In this report, I documented the characteristics of the environment, the method used to solve the environment control problem, the result of the training process, and what can be done to improve it.

The Environment

I used a modified version of the Unity Banana Collector environment. The objective of the environment is conducting a first-person agent through a square space where are two types of bananas objects on the floor, yellow bananas, and blue bananas. A reward of +1 is provided for collecting a yellow banana, and a reward of -1 is provided for collecting a blue banana. Thus, the goal of the agent is to collect as many yellow bananas as possible while avoiding blue bananas.

The state space has 37 dimensions and contains the agent's velocity, along with the ray-based perception of objects around the agent's forward direction. Given this information, the agent has to learn how to best select actions. Four discrete actions are available, corresponding to:

1. Forward

2. Backward

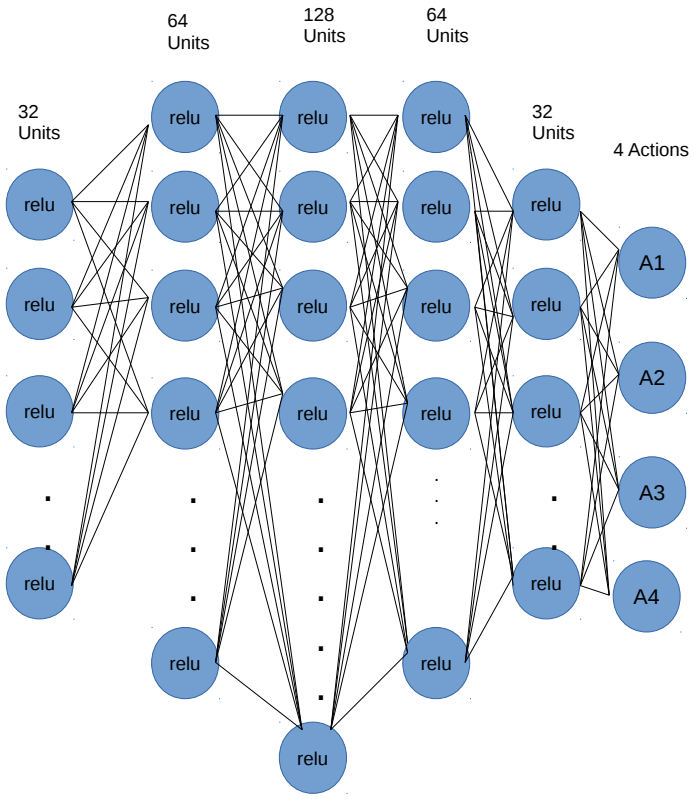
3. Left

4. Right

The environment is described by a continues space and discrete action space, the control problem at hands is to guide an agent so it collects only the yellow bananas and avoids the blue ones.

The Agent

The agent is a version of a Deep Q-Network Agent (https://deepmind.com/research/dqn/) that uses a deep neural network to approximate the optimal q* value function. Figure 1 contains the deep neural network architecture I used to create the agent.



Deep Neural Network: Figure1

A Q-learning agent or Sarsamax is an off-policy reinforcement method to solve discrete MDP (Markov Decision Problems). In a discrete problem that Q-learning algorithm approximates the true action value updating the current Q(s, a) every time step value by selecting the current state action using the e-greedy policy and as a target the action that maximizes the next state action-value function. The limitation of the Q-learning algorithm is that it does not work with continues spaces.

An adaptation of the Q-learning method to make it suitable for continues spaces is the Deep Q-network agent, which replaces the local and target values “q” by two neural networks.

The architecture depicted in Figure 1 uses Batch Normalization, dropout to stabilize the learning and avoid the networks to over-fit (memorize wrong actions); these two futures plus the used of a replay buffer and fix q-targets, additional mechanisms make this agent a suitable solution to the presented control problem.

The parameters I used in the agent calculations are listed below.

The replay buffer size was 1000; the learning batch consisted of 32 samples. I used a discount factor of 0.99, to perform a soft update of the target weights (see the fix q-targets weight documentation in the dqn URL) I used a TAU factor of 0,001, my learning rate was of 0.0001, and I performed weighs update every other fourth time step, once I had at least 32 samples in the replay buffer.

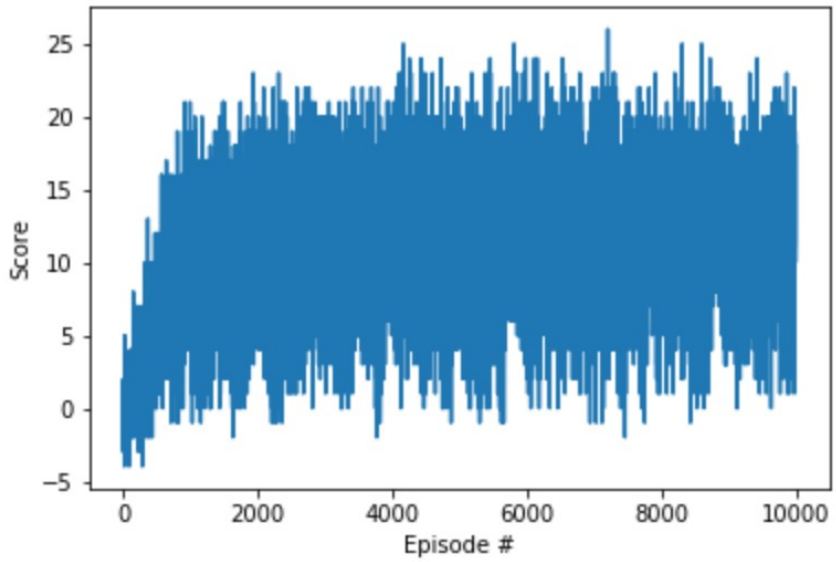
The network was created using Pytorch.

The result of the training

The agent was trained for 10000 episodes each with 1000 time steps, using the parameters described in the previous section. In table 1 is the average reward of each of the 10000 episodes, and in Figure 2 is the plot of such values.

The illustrations below are a clear indication that the agent is learning, although more than 10000 episodes may be necessary for the agent to be fully training and the control problem solve (solution score 22 or above.)

In the last section of the report, some proposed solutions are documented that may help the algorithm to converge to the answer faster than its simplified version.



Episode Score Chart: Figure 2

```
Episode 100     Average Score so far: 0.28
Episode 200     Average Score so far: 0.74
Episode 300     Average Score so far: 0.97
Episode 400     Average Score so far: 1.30
Episode 500     Average Score so far: 1.77
Episode 600     Average Score so far: 2.48
Episode 700     Average Score so far: 3.35
Episode 800     Average Score so far: 3.84
Episode 900     Average Score so far: 4.32
Episode 1000    Average Score so far: 5.04
Episode 1100    Average Score so far: 5.48
Episode 1200    Average Score so far: 5.79
Episode 1300    Average Score so far: 6.00
Episode 1400    Average Score so far: 6.36
Episode 1500    Average Score so far: 6.74
Episode 1600    Average Score so far: 6.95
Episode 1700    Average Score so far: 7.04
Episode 1800    Average Score so far: 7.20
Episode 1900    Average Score so far: 7.35
Episode 2000    Average Score so far: 7.59
Episode 2100    Average Score so far: 7.85
Episode 2200    Average Score so far: 8.01
Episode 2300    Average Score so far: 7.98
Episode 2400    Average Score so far: 8.04
Episode 2500    Average Score so far: 8.16
Episode 2600    Average Score so far: 8.28
Episode 2700    Average Score so far: 8.36
Episode 2800    Average Score so far: 8.50
Episode 2900    Average Score so far: 8.61
Episode 3000    Average Score so far: 8.74
```

```
Episode 3100    Average Score so far: 8.85
Episode 3200    Average Score so far: 8.97
Episode 3300    Average Score so far: 8.99
Episode 3400    Average Score so far: 9.09
Episode 3500    Average Score so far: 9.19
Episode 3600    Average Score so far: 9.29
Episode 3700    Average Score so far: 9.37
Episode 3800    Average Score so far: 9.39
Episode 3900    Average Score so far: 9.42
Episode 4000    Average Score so far: 9.52
Episode 4100    Average Score so far: 9.63
Episode 4200    Average Score so far: 9.70
Episode 4300    Average Score so far: 9.78
Episode 4400    Average Score so far: 9.84
Episode 4500    Average Score so far: 9.90
Episode 4600    Average Score so far: 9.95
Episode 4700    Average Score so far: 10.02
Episode 4800    Average Score so far: 10.05
Episode 4900    Average Score so far: 10.09
Episode 5000    Average Score so far: 10.14
Episode 5100    Average Score so far: 10.17
Episode 5200    Average Score so far: 10.18
Episode 5300    Average Score so far: 10.22
Episode 5400    Average Score so far: 10.23
Episode 5500    Average Score so far: 10.26
Episode 5600    Average Score so far: 10.30
Episode 5700    Average Score so far: 10.32
Episode 5800    Average Score so far: 10.38
Episode 5900    Average Score so far: 10.44
Episode 6000    Average Score so far: 10.49
Episode 6100    Average Score so far: 10.52
Episode 6200    Average Score so far: 10.54
Episode 6300    Average Score so far: 10.56
Episode 6400    Average Score so far: 10.60
Episode 6500    Average Score so far: 10.63
Episode 6600    Average Score so far: 10.69
Episode 6700    Average Score so far: 10.75
Episode 6800    Average Score so far: 10.77
Episode 6900    Average Score so far: 10.78
Episode 7000    Average Score so far: 10.81
Episode 7100    Average Score so far: 10.85
Episode 7200    Average Score so far: 10.88
Episode 7300    Average Score so far: 10.91
Episode 7400    Average Score so far: 10.93
Episode 7500    Average Score so far: 10.93
Episode 7600    Average Score so far: 10.96
Episode 7700    Average Score so far: 10.99
Episode 7800    Average Score so far: 11.02
Episode 7900    Average Score so far: 11.05
Episode 8000    Average Score so far: 11.07
Episode 8100    Average Score so far: 11.08
Episode 8200    Average Score so far: 11.09
Episode 8300    Average Score so far: 11.12
Episode 8400    Average Score so far: 11.13
Episode 8500    Average Score so far: 11.14
Episode 8600    Average Score so far: 11.15
Episode 8700    Average Score so far: 11.16
Episode 8800    Average Score so far: 11.18
Episode 8900    Average Score so far: 11.22
Episode 9000    Average Score so far: 11.21
Episode 9100    Average Score so far: 11.23
Episode 9200    Average Score so far: 11.22
Episode 9300    Average Score so far: 11.22
Episode 9400    Average Score so far: 11.24
Episode 9500    Average Score so far: 11.25
Episode 9600    Average Score so far: 11.25
Episode 9700    Average Score so far: 11.27
Episode 9800    Average Score so far: 11.29
Episode 9900    Average Score so far: 11.30
Episode 10000   Average Score so far: 11.31
```

Improvements

Even if the agent was training to an acceptable level, the following improvement could be made to optimize the agent behavior and make it converge to the option q* value in less training episodes.

Double Deep Q-Network

The problem with the dqn agent is that it tends to overestimate the q values, this is because it uses the q value if the actions that maximize the q estimate function to determine the real q value, this produces a tendency toward given values to actions higher than the real ones. The solution to this problem is the use of two sets of Q-target networks weighs, one is used to select the action and the other set of weighs is used to select the value of that action.

Prioritize Experience Replay

The main idea behind this improvement is that during the selection of the experiences to be used in the learning process, we select these using a random process, with this we run the risk of selecting the same set of samples or even worst not selecting valuable experiences. The improvement consists of using a parameter to weigh the importance of the experience base on its value. The value we can use to prioritize the selection of important experiences can be the error term obtained when we are performing the weigh optimization, using this value we can calculate the probability of the sample selection, the higher the experience error term the highest its probability is of been selected.