# Project Navigation

## Introduction

The aim of the project is to train an agent to navigate (and collect bananas!) in a large, square world.

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 ray-based perception of objects around agent's forward direction. Given this information, the agent has to learn how to best select actions. Four discrete actions are available, corresponding to:

- `0` - move forward.
- `1` - move backward.
- `2` - turn left.
- `3` - turn right.

The task is episodic, and in order to solve the environment, the agent must get an average score of +13 over 100 consecutive episodes.

## Solution

The algorithmic approach used to solve the problem is to use a DQN (Deep Q-Network) that takes in the state vector and outputs the predicted action values for each possible action that the agent can take. The training runs over many episodes of game playing where the network learns to approximate the underlying action-value function. We use experience replay and fixed Q-targets to improve the learning of the network.

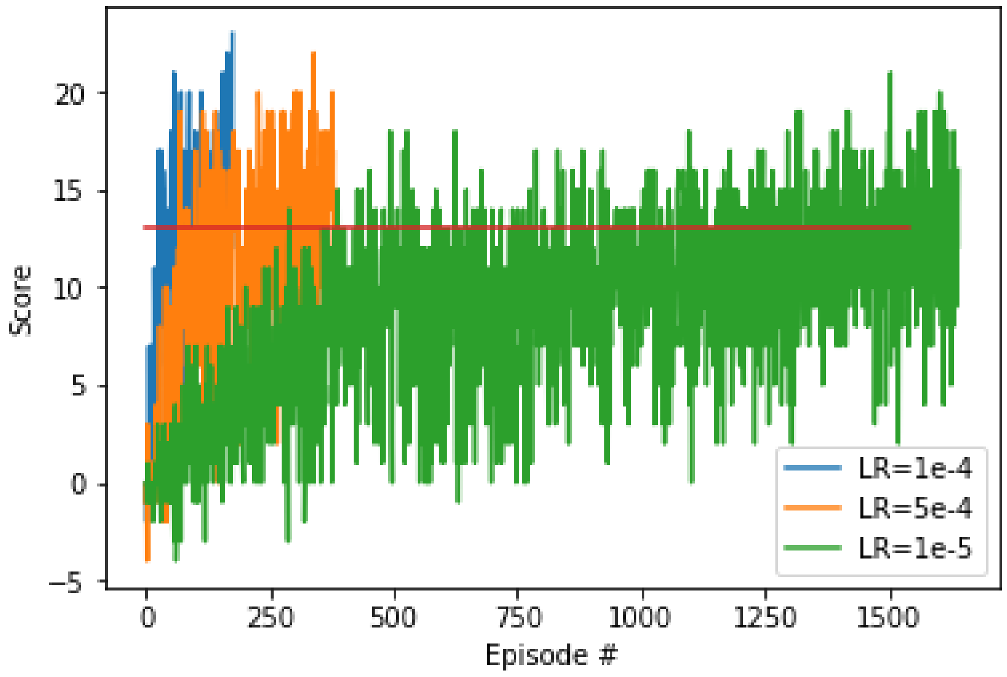
The optimizer used to learn the parameter is Adam optimizer with a specified learning rate. Experiments on the learning rate to find the optimal value for the problem would be described in a later section

## Network Architecture

We use a simple fully connected network with 2 hidden layers. The input vector size is 37. The first layer has 64 neurons, the second layer has 64 neurons and the output size is 4 (corresponding to the 4 actions)

### Determining the Learning Rate

The learning rate was determined based on various training runs with only the learning rate varied between the runs. The best results (in terms of how fast the network converged) were obtained for a learning rate of  1e-4. The plot below shows the rewards per episode obtained with the different learning rates. The red line sows the target +13 that needs to be achieved for the problem to be considered solved. As can be seen the learning rate of 1e-4 achieved the target with least number of iterations. Over 3 different training runs, the average number of iterations to reach the target was ~122 iterations.



**Future Work**

The performance of the algorithm could be improved by exploring other ideas such as Double DQN, prioritized replay as well as dueling DQN. These approaches might especially be useful if we are trying to solve the problem from the images directly.