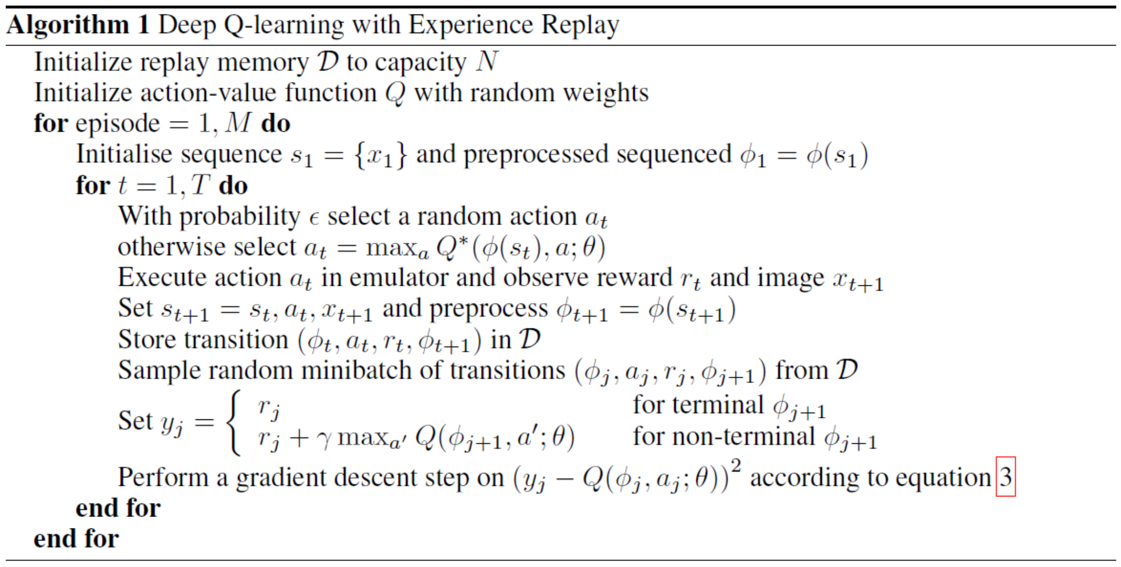
Report – Deep Q-Learning applied for Project 1-Navigation

Objective : The main aim of the agent is to navigate the environment and collect maximum possible number of yellow bananas and the blue bananas should be avoided

Algorithm: The idea behind Deep-Q-Learning is to approximate the optimal action-value function using neural networks



```
Algorithm 1 Deep Q-learning with Experience Replay
  Initialize replay memory $\mathcal{D}$ to capacity $N$
  Initialize action-value function $Q$ with random weights
  for episode $= 1, M$ do
      Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
      for $t = 1, T$ do
          With probability $\epsilon$ select a random action $a_t$
          otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
          Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
          Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
          Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
          Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
          Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
          Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
      end for
  end for
```

Algorithm [1]

The process involves in sampling the Environment in which actions are taken in state using greedy policy and observe the reward and move to the next state. The experience is stored in the experience tuple in the replay memory D.

Then follows the learning process, in which a random minibatch of tuples from memory D, are taken and learning is done using a gradient descent update step. Here, Adam is used in the project.

Model architecture : It consists of fully connected layers with 64 nodes in the first hidden layer and also 64 nodes in the second hidden layer and the final layer trickles down to with the number of neurons same as that of the 'action_size'. This neural network maps the states to the action values.

Hyperparameters : All the hyperparameters except (eps_decay = 0.992) are set to the similar values from the lesson on Deep Q-learning provided in Udacity's course module Deep Reinforcement Learning. These were considered best for environment to be solved in 302 episodes with an average score of 13.05.

References

[1] Mnih, Volodymyr & Kavukcuoglu, Koray & Silver, David & Graves, Alex & Antonoglou, Ioannis & Wierstra, Daan & Riedmiller, Martin. (2013). Playing Atari with Deep Reinforcement Learning.

[2] Udacity lectures on Deep Q - Learning