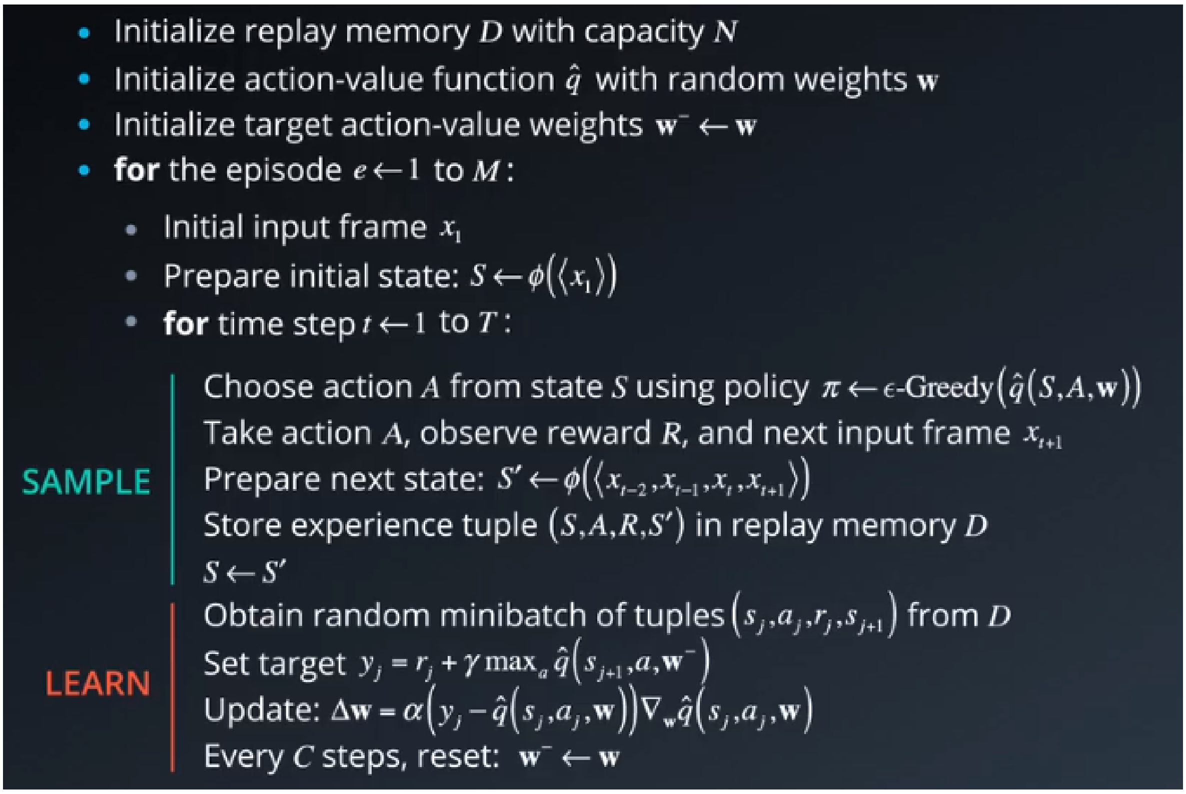
Introduction

This is my report of project 1: Navigation. In this project, I trained an agent to navigate in a large, square world. The goal of the agent is to collect as many yellow bananas as possible while avoiding 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. The episodic task is considered solved when the agent gets an average score of +13 over 100 consecutive episodes.

Algorithm

I solved the problem using Deep Q-Learning algorithm (DQN) .

Pseudocode of DQN:



```
- Initialize replay memory $D$ with capacity $N$
- Initialize action-value function $\hat{q}$ with random weights $\mathbf{w}$
- Initialize target action-value weights $\mathbf{w}^- \leftarrow \mathbf{w}$
- for the episode $e \leftarrow 1$ to $M$:
    - Initial input frame $x_1$
    - Prepare initial state: $S \leftarrow \phi(\langle x_1 \rangle)$
    - for time step $t \leftarrow 1$ to $T$:
                 | Choose action $A$ from state $S$ using policy $\pi \leftarrow \epsilon\text{-Greedy}(\hat{q}(S,A,\mathbf{w}))$
                 | Take action $A$, observe reward $R$, and next input frame $x_{t+1}$
        SAMPLE   | Prepare next state: $S' \leftarrow \phi(\langle x_{t-2}, x_{t-1}, x_t, x_{t+1} \rangle)$
                 | Store experience tuple $(S,A,R,S')$ in replay memory $D$
                 | $S \leftarrow S'$
                 | Obtain random minibatch of tuples $(s_j, a_j, r_j, s_{j+1})$ from $D$
        LEARN    | Set target $y_j = r_j + \gamma \max_a \hat{q}(s_{j+1}, a, \mathbf{w}^-)$
                 | Update: $\Delta\mathbf{w} = \alpha(y_j - \hat{q}(s_j, a_j, \mathbf{w}))\nabla_{\mathbf{w}}\hat{q}(s_j, a_j, \mathbf{w})$
                 | Every $C$ steps, reset: $\mathbf{w}^- \leftarrow \mathbf{w}$
```

Hyperparameters:

buffer size: 100000

batch size: 64

learning rate: 0.0005

γ (discounting rate): 0.99

τ (soft update coefficient): 0.001

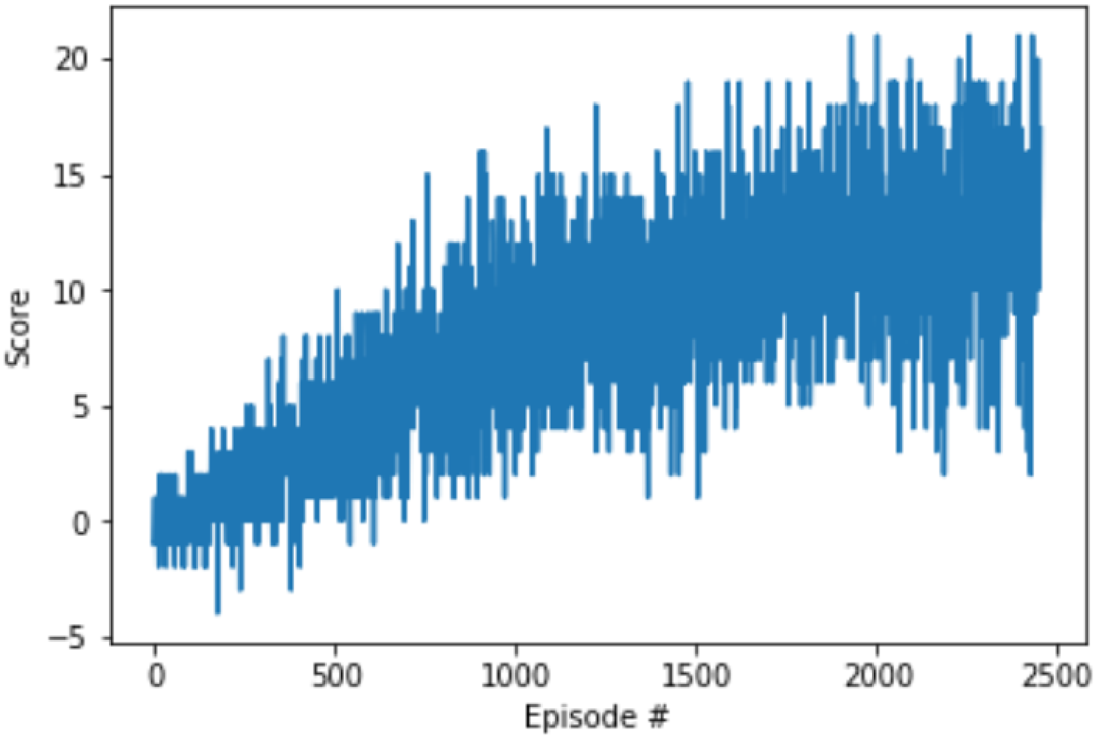
number of time steps that model learns: 4

ϵ-greedy parameters: 1.0 (start), 0.01 (end), 0.999 (decay)

Architecture of the neural network:

input layer (# 37) -> hidden layer 1 (# 128) -> hidden layer 2 (# 64) -> output layer (# 4)

Result



The task was solved after 2354 episodes with average score of 13.04. The weights of trained model were saved in checkpoint.pth.

Future work

Implement Double DQN, Dueling DQN, and Prioritized Experience Replay.