Building Reinforcement Learning Agent to navigate in a 4x4 grid world using Q learning

Shwetasree Chowdhury

Person Id: 50296995

Department of Computer Science

State University of New York at Buffalo

shwetasr@buffalo.edu

Abstract

The task at hand for this project is to build a reinforcement learning agent to navigate the classic 4x4 grid-world environment to start from one position- random or otherwise and reach its stipulated goal. The agent will learn an optimal policy to reach its goal through Q-Learning (using the formulation of a q table which will allow it to take actions to reach a goal while avoiding obstacles and maximizing rewards.

Reinforcement learning is a machine learning paradigm which focuses on how automated agents can learn to take actions in response to the current state of an environment so as to maximize some reward. This is typically modeled as a Markov decision process (MDP), which we will learn about later.

Problem Statement / Challenge

The Problem Statement for this project is a fairly simple one. We are implementing Reinforcement Learning to solve a grid world environment. The Grid world environment is described in details in Section 1.2.

The Grid World problem is a classic reinforcement learning problem. The agent located in one position (top left corner in our case) aims to reach a goal which is located at the rightmost bottom corner of our grid. But the environment may/ may not be replete with obstacles. The game ends when the agent reaches the goal or reaches any other stopping condition as may be mentioned. In this project, in order to incentivize the agent for making a progressive choice of movement (up, down, left, right) towards the goal, the rewards associated with each step are:

• -1, if the agent stays at the same place or it moves away from the goal

• +1, if it moves towards the goal

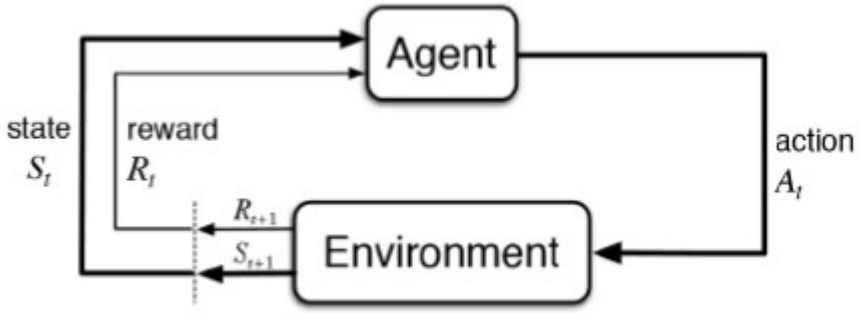
Some other parameters used to train/hypertune our agent :

- state space: n x n matrix with real values on the interval [0; 1]
- action space: up, down, left, right.
- Learning rate
- No of episodes: 1000
- Epsilon

1.1 What is Reinforcement Learning?

As previously discussed, Reinforcement learning is a machine learning paradigm which focuses on how automated agents can learn to take actions in response to the current state of an environment so as to maximize some reward. The learner and decision-maker is called the agent. The thing it interacts with, comprising everything outside the agent, is called the environment. These interact continually, the agent selecting actions and the environment responding to those actions and presenting new situations to the agent. The environment also gives rise to rewards, special numerical values that the agent tries to maximize over time.

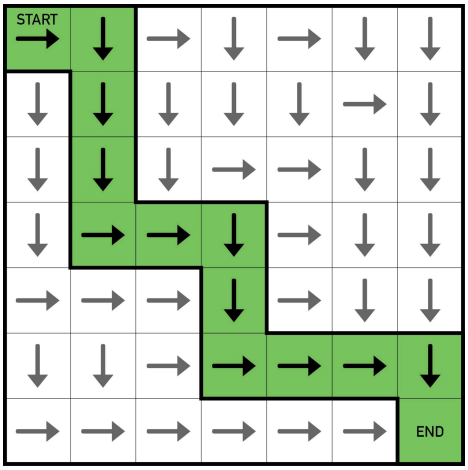
- Agent receives state S(t) from the Environment.
- Based on that state St, agent takes an action A(t) (our agent will move right/down)
- Environment transitions to a new state S(t+1) (new frame)
- Environment gives some reward R(t) to the agent (one step closer to goal: +1)
- This RL loop outputs a sequence of state, action and reward.
- The goal of the agent is to maximize the expected cumulative reward.
- At each time step, the agent implements a mapping from states to probabilities of selecting each possible action. This mapping is called the agent's policy and is denoted by π.



The Markov property expresses that environment's next state, s(t+1) at time (t+1) only depends on the representation of the current state and action. Markov Decision Process (MDP) describes a Reinforcement Learning task that states can be specified by Markov properties.

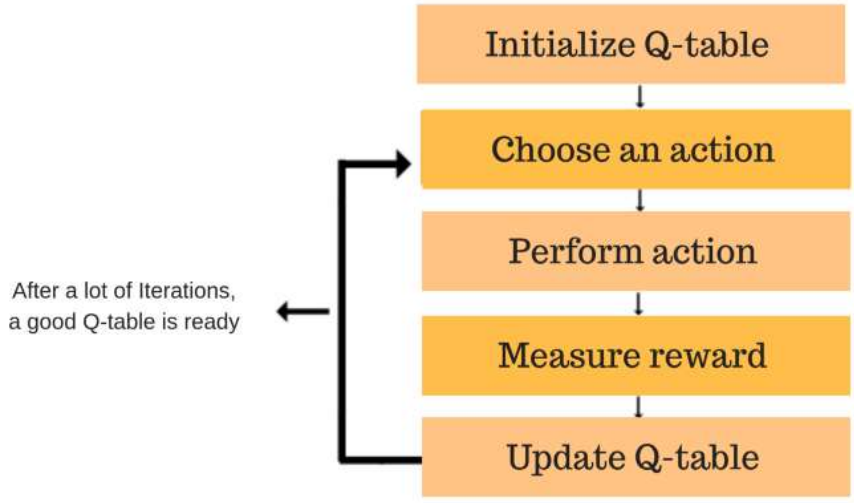
1.2 Environment Used

Grid World is a 2D rectangular grid of size (n x n) with an agent starting off at one grid square and trying to move to another grid square located elsewhere. Such an environment is a natural one for applying reinforcement learning algorithms to discover optimal paths and policies for agents on the grid to get to their desired goal grid squares in the least number of moves



1.3 What is Q learning

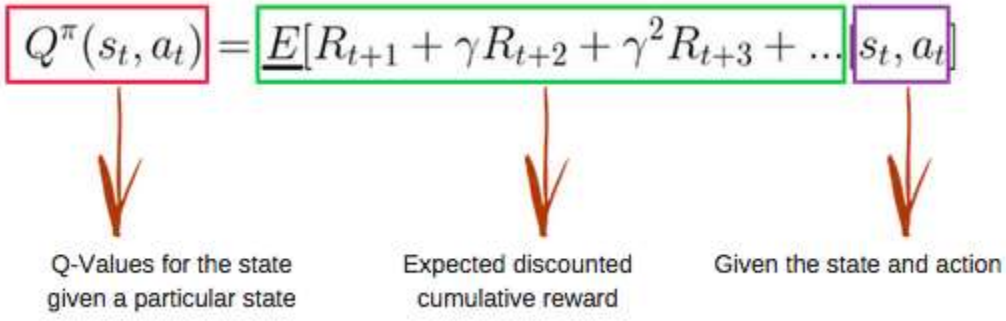
Q-learning is a model-free reinforcement learning algorithm. The goal of Q-learning is to learn a policy, which tells an agent what action to take under what circumstances. It does not require a model (hence the connotation "model-free") of the environment, and it can handle problems with stochastic transitions and rewards, without requiring adaptations.



This process is done to maximize reward for each step or episode to find the optimal solution and it runs until the episode ends after which it may start again or replayed for the next episode.

When we try to train our agent using Q learning, we use the Q learning algorithm which uses a Q table to tabulate the rewards at each step, which helps the agent choose between exploration and exploitation. Q(state, action) returns the expected future reward of that action at that state.

This function can be estimated using Q-Learning, which iteratively updates Q(s,a) using the Bellman equation.



Initially we explore the environment and update the Q-Table. When the Q-Table is ready, the agent will start to exploit the environment and start taking better actions.

Q- Table

Q-Table is just a fancy name for a simple lookup table where we calculate the maximum expected future rewards for action at each state. This table will guide us to the best action at each state.
For the purpose of our project, there will be four numbers of actions at each non-edge tile. When an agent is at a state it can either move up or down or right or left. In the Q-Table, the columns are the actions and the rows are the states.

Each Q-table score will be the maximum expected future reward that the robot will get if it takes that action at that state. This is an iterative process, as we need to improve the Q-Table at each iteration. Initially, all the values in the Q-table are zeros.

As we start to explore the environment, the Q-function gives us better and better approximations by continuously updating the Q-values in the table.

2. Implementation

1. Importing libraries
2. Setting a random seed
3. Defining the grid world environment
    - **init** - initialises the environment parameters - namely observation space and action space.

Observation space is the number of observable states that our environment provides. Action space defines the number of actions that can be taken on the environment - namely up, down, left, right movements.

- **reset** - Resets the environment
- **step** - function that defines the essence of the environment. It contains the decision making unit of the environment. A step function takes in an 'action' and decides what to do next or where to go.
- Step returns four values. These are:
    a. observation (object): an environment-specific object representing your observation of the environment.
    b. reward (float): amount of reward achieved by the previous action.
    c. done (boolean): whether it's time to reset the environment again.
    d. info (dict): diagnostic information useful for debugging. It can sometimes be useful for learning.
- In step function based on the action received, we get the possible next positions from the transition table we created while initializing the environment.
- Based on this next state we calculate the rewards associated with moving to this next state or position based on the values defined above
- **render** - Renders one frame of the environment
- **_get_distance**- calculate distance between two points
- **Transition_func** - creates a probability transition matrix to store probabilities associated with each movement.

4. Defining a Random/ Heuristic agent to render in the environment defined above to see if our model is working as expected.
5. Defining Q learning Agent:
    - Central to Q learning is the concept of exploitation and exploration. Exploitation make the best decision given current information while exploration is gathering more information. The best long–term strategy may sometimes involve short–term sacrifices. While the overall aim of the agent would be to gather enough information to make the best overall decisions.
    - The reward calculation at each step is tabulated and considered for calculation for next step with the help of a Q table which is a lookup table where we calculate the maximum expected future rewards for action at each state. In the Q-Table, the columns are the actions and the rows are the states. The values in the Q table are calculated by the Q-Learning algorithm explained in above section
    - The RL model here is loosely based on an epsilon-decreasing strategy. With this strategy we explore with probability epsilon, and exploit with probability 1 — epsilon. Epsilon decreases over time.
    - **Init**: initialises Q learning agent
    - **Policy**: defines the policy for our Q learning agent (as described in our project requirement)

- **Update**: updates the Q table

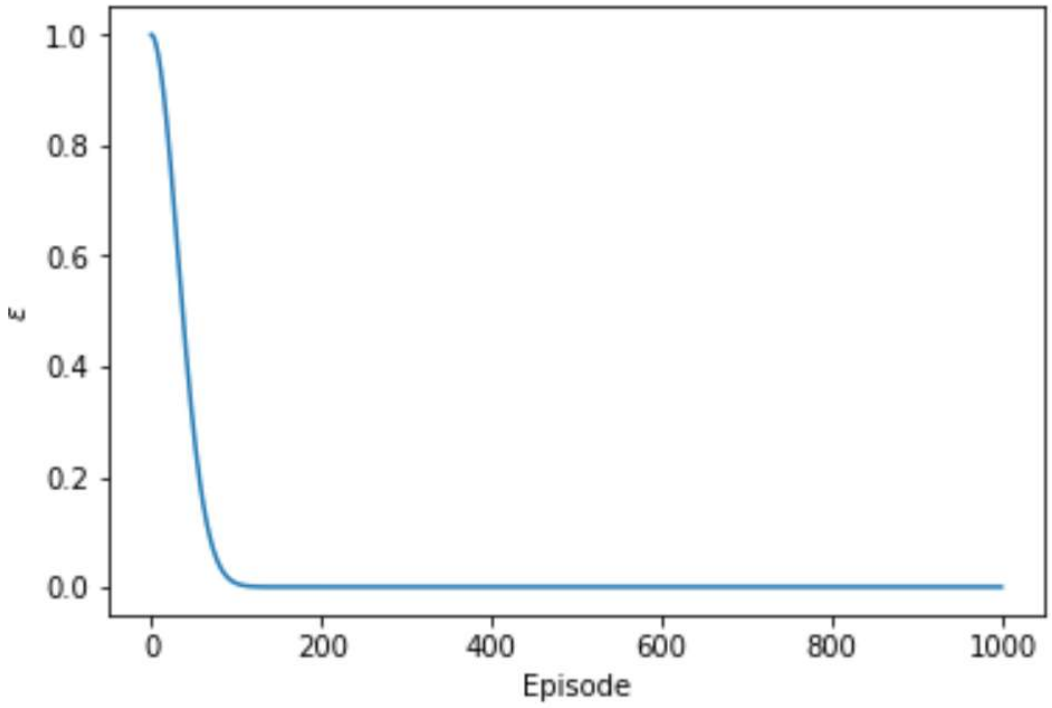
6. Training the agent:
    - The agent is run for total number of episodes.
    - For each episode, the environment is first reset and the total rewards are set to 0.
    - Then, until agent reaches goal or no of maxsteps are exceeded, the step function is called and the new state and rewards are calculated.
    - after calculation of rewards associated with current state and action, step function returns the next_state and we append the total_reward for that episode. We then update the q table with the current state and the next state.
    - Once done, the epsilon is updated and the total reward of each episode is appended for visualisation
    - Also, convergence (the first episode at which the agent reaches the goal) is also calculated.

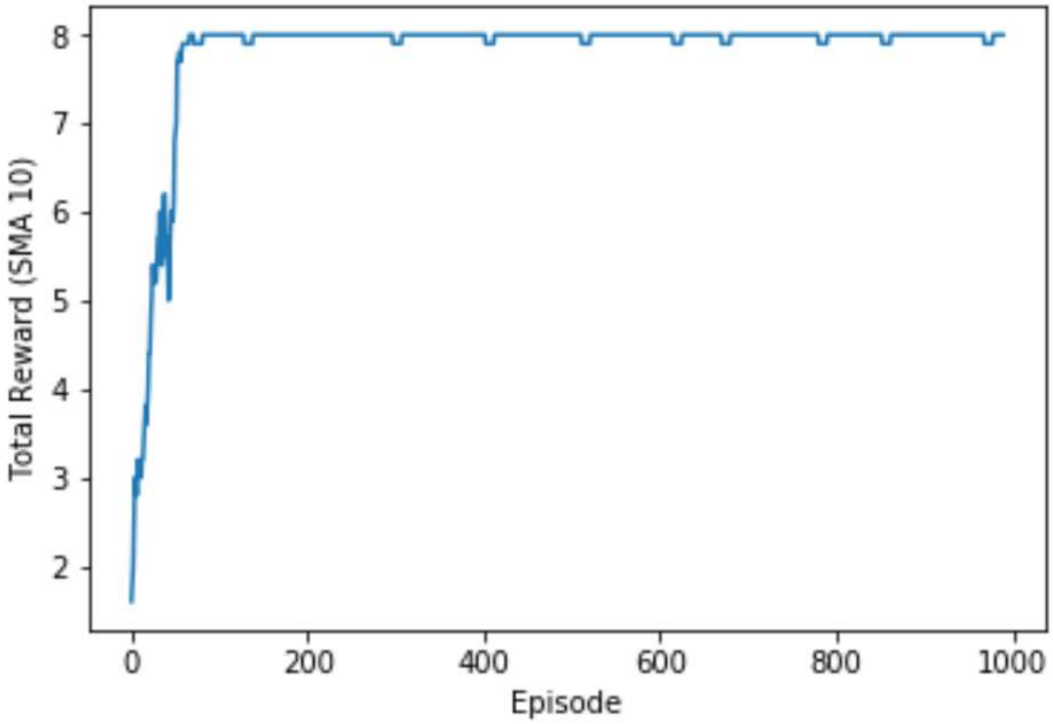
3. Results

Case 1

epsilon=1.0
lr=0.05
gamma=0.9
total_episodes = 1000
decay = 0.001

```
Final Q table
[[[ 5.69532790e+00  6.73866914e-02  8.64357793e-01 -1.99652955e-03]
  [ 1.10128795e+00 -2.16119508e-01  4.61171256e-01 -5.64325628e-02]
  [ 1.06823706e-01 -1.97278886e-01  3.73804023e-01 -9.06768187e-02]
  [ 5.23512500e-02 -5.00000000e-02  2.87073998e-01 -8.65389000e-02]
  [ 2.64908109e-01  0.00000000e+00 -5.00000000e-02 -4.77500000e-02]]

 [[ 3.60481642e-01 -7.01122112e-02  5.21703100e+00 -1.70843158e-01]
  [ 4.68559000e+00 -2.09502579e-01  4.00469203e-01 -1.12608979e-01]
  [ 5.88740490e-01 -9.08737500e-02  5.22500000e-02 -2.09361765e-02]
  [ 5.22500000e-02 -4.77500000e-02  9.75000000e-02 -4.29099813e-02]
  [ 5.00000000e-02 -8.70152812e-02 -5.00000000e-02 -1.74816122e-01]]

 [[ 3.03912704e-01 -8.85187947e-02  1.36308369e-01 -8.49846562e-02]
  [ 2.99295255e-01  1.33614453e-01  4.09510000e+00 -2.28626318e-01]
  [ 3.43900000e+00 -1.76841608e-01  2.06216813e-01 -6.02248875e-02]
  [ 1.92018750e-01 -5.00000000e-02  5.00000000e-02 -1.79974379e-02]
  [ 0.00000000e+00 -5.00000000e-02  0.00000000e+00  0.00000000e+00]]

 [[ 9.99563222e-02 -1.20555566e-01  0.00000000e+00  0.00000000e+00]
  [ 9.97500000e-02 -2.60864964e-02  5.24043693e-01 -5.00000000e-02]
  [ 2.71000000e+00 -1.09935884e-01  1.89881250e-01 -1.66661337e-01]
  [ 0.00000000e+00 -1.31937500e-01  1.89881250e-01 -4.77500000e-02]
  [ 5.00000000e-02  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[ 0.00000000e+00  0.00000000e+00  1.12967402e-01  0.00000000e+00]
  [-5.00000000e-02 -9.07775625e-02  3.70080570e-01  0.00000000e+00]
  [-9.69383525e-02 -7.26003566e-02  1.90000000e+00 -1.78106148e-01]
  [-9.52500000e-02 -8.47872188e-02  1.00000000e+00 -4.77500000e-02]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]]

Convergence reached at episode:  26
```

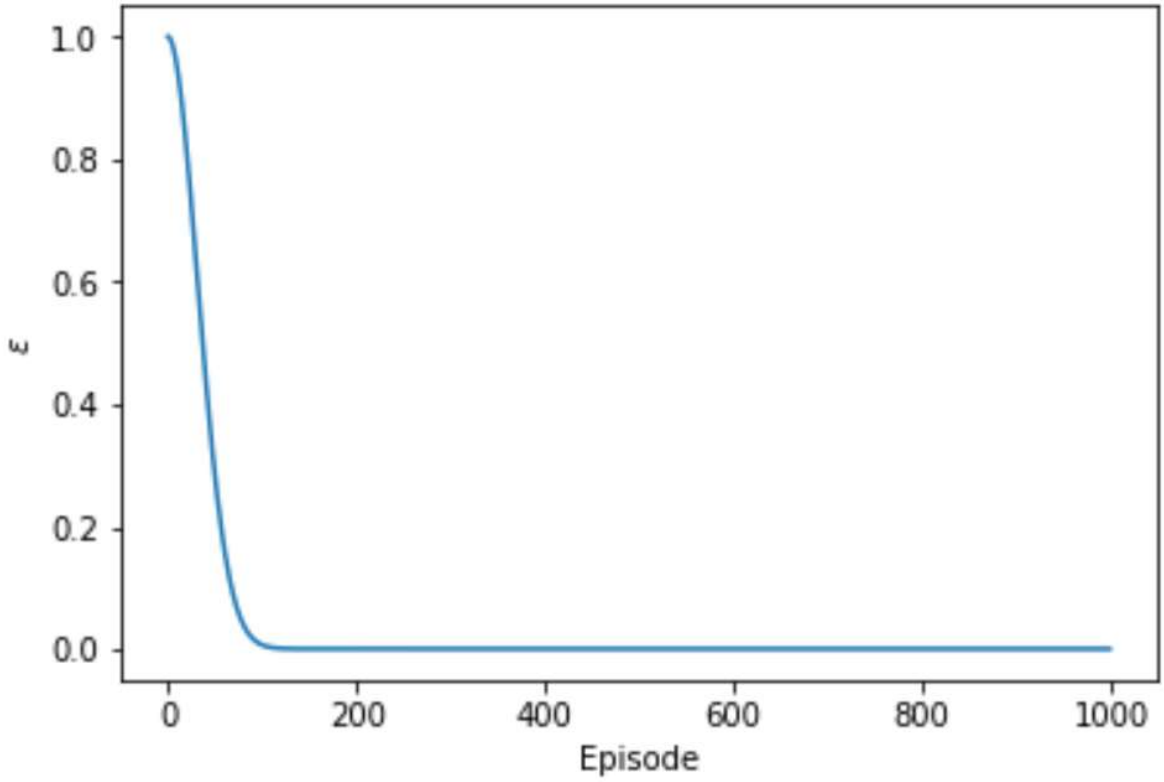

1.0
0.8
0.6
0.4
0.2
0.0
ε
0
200
400
600
800
1000
Episode

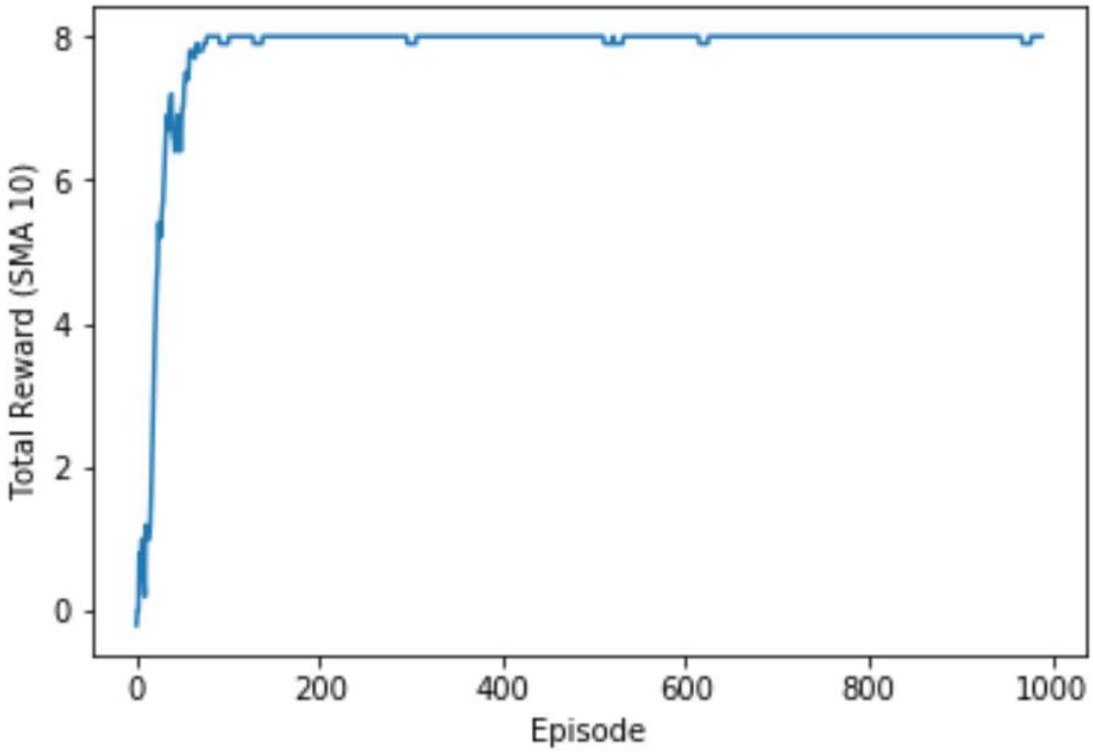

8
7
6
5
4
3
2
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 2

epsilon=1.0
lr=0.1
gamma=0.9
total_episodes = 1000
decay = 0.001

```
Final Q table
[[[ 5.69532790e+00  4.18753094e-01  1.58390894e+00  1.77025015e-01]
  [ 1.28870403e+00 -2.46694410e-01  7.00543561e-01  5.13916130e-03]
  [ 3.76390000e-01  0.00000000e+00  1.00000000e-01 -1.63249183e-01]
  [ 0.00000000e+00  0.00000000e+00  1.00000000e-01 -9.10000000e-02]
  [ 1.90000000e-01 -1.00000000e-01  0.00000000e+00  0.00000000e+00]]

 [[ 5.21703100e+00 -9.83124314e-03  1.06650910e+00 -1.16596492e-01]
  [ 1.29229247e+00 -2.82582990e-01  5.50702000e-01 -1.16745483e-01]
  [ 1.90000000e-01 -7.40710000e-02  4.42000000e-01 -2.89459000e-01]
  [ 3.61000000e-01 -1.00000000e-01  1.90000000e-01  0.00000000e+00]
  [ 0.00000000e+00 -9.10000000e-02 -1.00000000e-01 -8.29000000e-02]]

 [[ 4.68559000e+00  1.03641669e-01  1.08419441e+00  8.25928284e-02]
  [ 2.27569278e+00 -2.80674422e-01  2.82523879e-01 -2.34881759e-01]
  [ 2.52401680e-01 -1.57510000e-01  0.00000000e+00  0.00000000e+00]
  [ 1.00000000e-01 -7.48000000e-02  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[ 3.68290000e-01  5.21865334e-01  4.09510000e+00 -1.56974081e-01]
  [ 3.43900000e+00 -1.21439042e-02  5.42697267e-01  3.77416059e-01]
  [ 1.07268482e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 1.00000000e-01  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[-9.10000000e-02 -1.19530739e-01  5.02127132e-01 -2.53900000e-01]
  [-4.37608000e-01 -3.46591731e-02  2.71000000e+00 -3.68263900e-01]
  [-2.12181558e-02 -1.62108236e-01  1.90000000e+00  1.72467163e-01]
  [-1.81000000e-01  0.00000000e+00  1.00000000e+00 -1.00000000e-01]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]]

Convergence reached at episode:  38
```

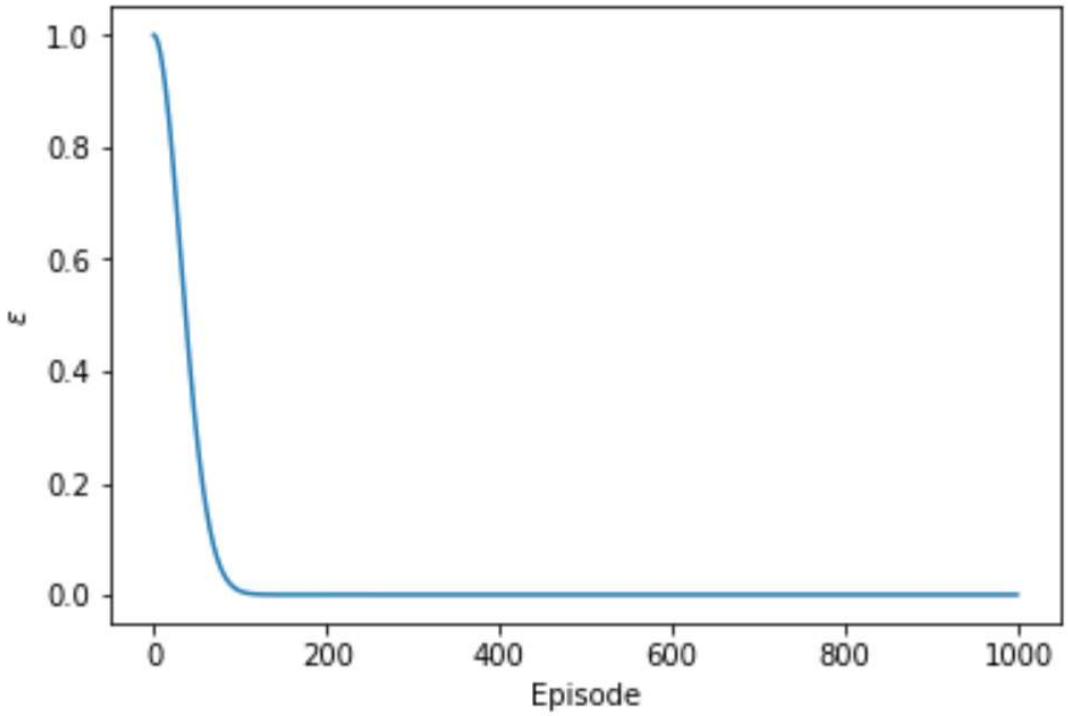

1.0
0.8
0.6
0.4
0.2
0.0
ε
0
200
400
600
800
1000
Episode

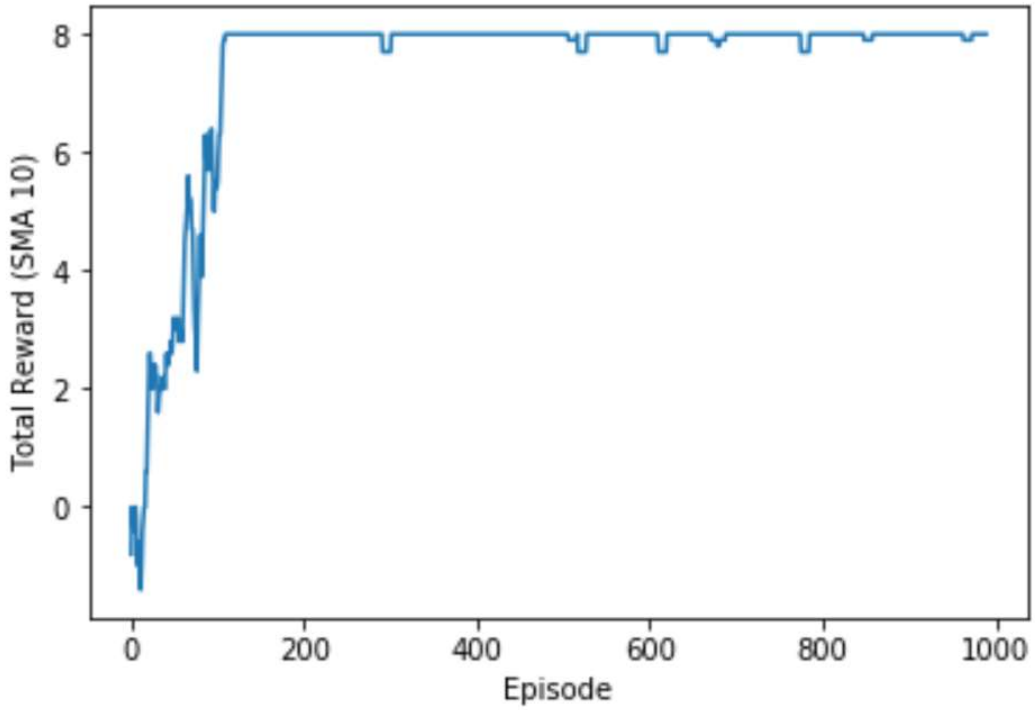

8
6
4
2
0
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 3

epsilon=1.0
lr=1.5
gamma=0.9
total_episodes = 1000
decay = 0.001

```
Final Q table
[[[-7.73586515  4.75514306  5.6953279  -1.8171929 ]
  [ 1.21948034  1.63819067  5.217031   -0.64171551]
  [ 4.68559     4.37372999  1.94422825  1.80526707]
  [ 0.94756394  0.75774433  1.5         1.04048224]
  [ 3.525       0.          0.          0.        ]]

 [[-4.05096437 -1.15834894 12.40326857 -1.99721163]
  [ 3.71011127  1.25916606  2.24511328  2.04467926]
  [ 1.83463047  1.40127275  4.0951      0.13941951]
  [ 3.439       0.          1.089375    1.3668098 ]
  [ 2.45625     0.          0.          0.        ]]

 [[-3.44159625  0.88813575 -0.11470999  1.22026186]
  [-0.25774157  1.2589477   3.31717727  1.61701055]
  [ 2.04224559  1.90255657  3.64830273  0.525     ]
  [ 2.71        0.525       0.          0.64014409]
  [ 2.9625      0.          0.          0.        ]]

 [[ 1.43189393  0.75588695  1.14084751  0.16672588]
  [-2.28054004  0.32699059 -0.34307337 -0.38729961]
  [ 0.          1.7118322   0.676875   -0.4875    ]
  [ 1.9         1.38331687  1.7625      0.12166073]
  [ 0.9375      0.          0.          0.        ]]

 [[-1.5         0.02001463  0.          0.436359  ]
  [-0.75       -0.87467874 -1.12300945 -1.5       ]
  [-1.5        -1.54353146  2.85263672  0.        ]
  [-1.5        -0.4875      1.          0.        ]
  [ 0.          0.          0.          0.        ]]]

Convergence reached at episode:  79
```

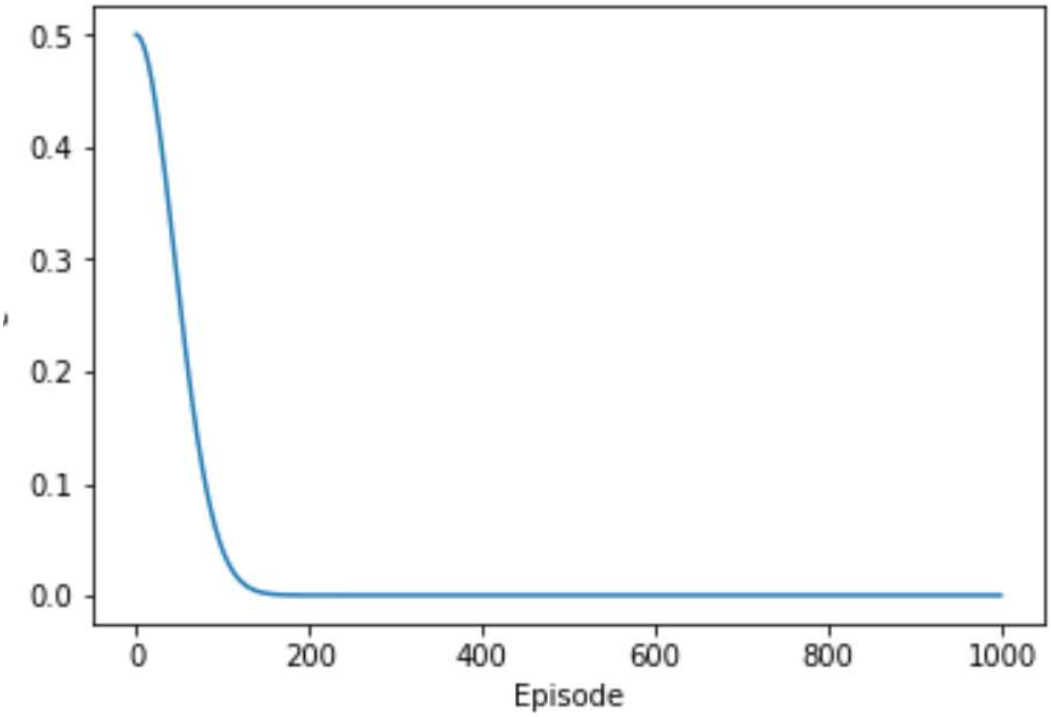

1.0
0.8
0.6
0.4
0.2
0.0
ε
0
200
400
600
800
1000
Episode

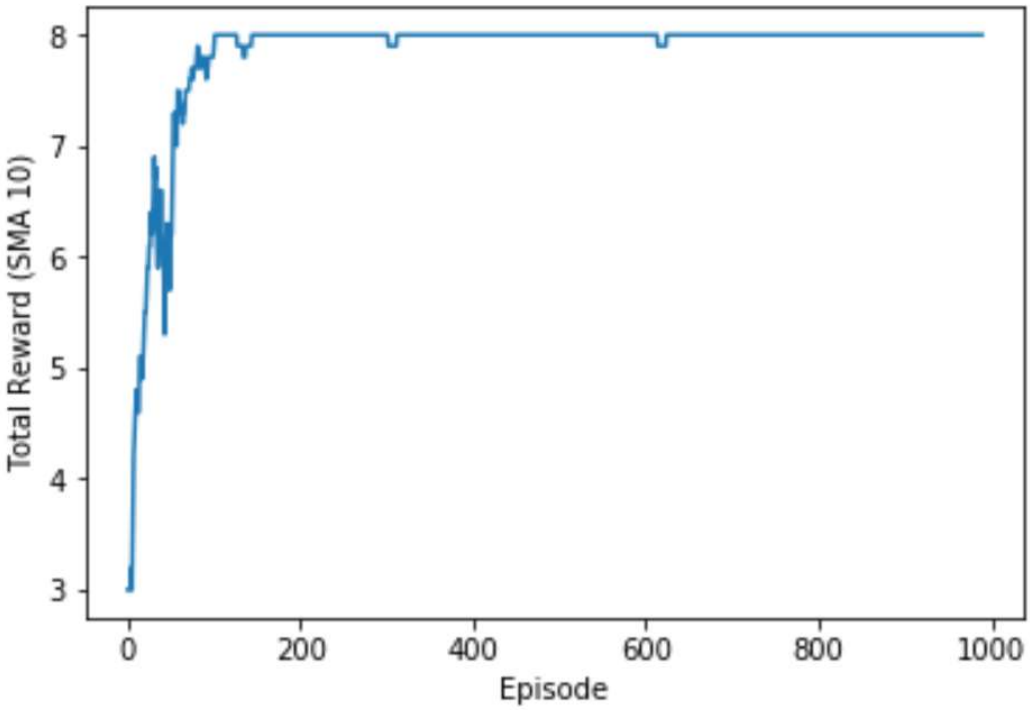

8
6
4
2
0
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 4

epsilon=0.5
lr=0.1
gamma=0.9
total_episodes = 1000
decay = 0.001

```
Final Q table
[[[ 5.69532790e+00  2.37924058e-01  8.07210213e-01  1.23989149e+00]
  [ 8.54276020e-01 -3.01268622e-01  2.00677510e-01  2.34502178e-01]
  [ 1.18639000e-01  0.00000000e+00  1.00000000e-01  0.00000000e+00]
  [ 1.00000000e-01  0.00000000e+00  0.00000000e+00 -8.93224900e-02]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[ 5.21703100e+00  6.50860302e-01  1.03698002e+00  9.62955022e-01]
  [ 1.91287907e-01 -1.15938296e-01  1.51302034e+00 -6.74953674e-02]
  [ 1.39148666e+00 -8.93224900e-02  0.00000000e+00  3.15116766e-03]
  [ 0.00000000e+00 -9.10000000e-02  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[ 7.62442968e-01  6.66751085e-01  4.68559000e+00  1.66237789e-01]
  [ 4.09510000e+00 -6.81304879e-02  7.37328664e-01  1.09648042e-01]
  [ 1.36145332e+00 -9.01900000e-02  0.00000000e+00  9.65894592e-02]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[ 1.05175519e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]
  [ 3.43900000e+00  4.07236690e-01  3.64584939e-01 -8.84263710e-02]
  [ 1.30951000e-01  0.00000000e+00  1.50194290e+00 -6.22175994e-02]
  [ 1.39631122e+00 -1.00000000e-01  1.00000000e-01 -1.27022410e-01]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]

 [[-1.00000000e-01 -1.83973165e-01  1.40841278e+00  0.00000000e+00]
  [ 1.10682508e-01  1.10775239e-01  2.71000000e+00 -2.36651825e-01]
  [-2.21031408e-03 -3.32065420e-03  1.90000000e+00  1.67891172e-01]
  [-1.88486792e-01 -2.52536939e-01  1.00000000e+00  5.47667672e-02]
  [ 0.00000000e+00  0.00000000e+00  0.00000000e+00  0.00000000e+00]]]

Convergence reached at episode:  10
```

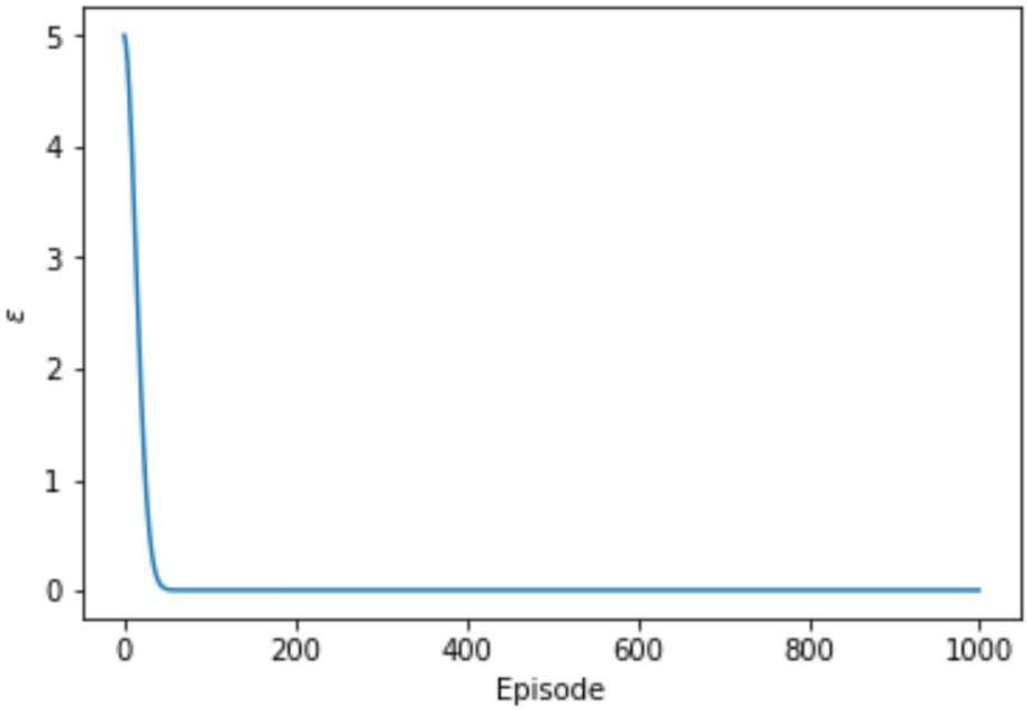

0.5
0.4
0.3
0.2
0.1
0.0
0
200
400
600
800
1000
Episode

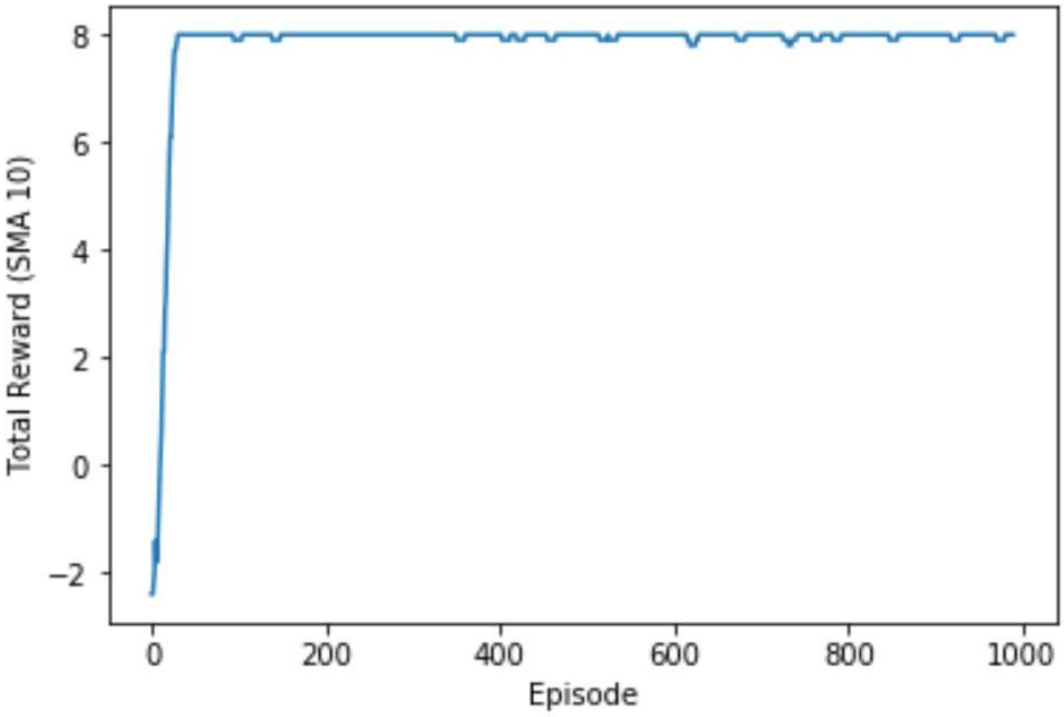

8
7
6
5
4
3
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 5

epsilon=5.0
lr=0.1
gamma=0.9
total_episodes = 1000
decay = 0.001

```
Final Q table
[[[ 1.39977552  0.47624516  5.6953279   0.56529006]
 [ 0.66625579  0.12609171  5.217031    0.12667037]
 [ 0.62267732  0.13435782  4.68559     0.54409641]
 [ 4.0951     -0.3353131   0.30177292  0.09333234]
 [ 0.36985846  0.          0.          0.        ]]

[[ 0.82259975  0.08648798  0.4821679  -0.41241261]
 [ 0.         -0.18164026  0.59931161 -0.16299864]
 [ 0.74528932 -0.143659    0.1        -0.290269  ]
 [ 3.439       0.0802169   0.50597758 -0.18126181]
 [ 0.7628538   0.          0.         -0.04808798]]

[[ 0.199      -0.22861593  0.4501     -0.1729    ]
 [ 0.31078    -0.091       0.         -0.074071  ]
 [ 0.91193234  0.          0.          0.        ]
 [ 2.71        0.35423911  0.50550342 -0.1705591 ]
 [ 0.95856821  0.          0.          0.        ]]

[[ 0.199       0.          0.          0.        ]
 [ 0.3529      0.          0.          0.        ]
 [ 1.1520911   0.          0.          0.        ]
 [ 1.9         0.27341     0.         -0.16130114]
 [ 0.6861894   0.          0.          0.        ]]

[[ 0.         -0.17209     0.1        -0.091     ]
 [-0.1        -0.091       0.2749041  -0.181     ]
 [-0.1        -0.1         1.33512479  0.        ]
 [-0.19       -0.091       1.         -0.07480411]
 [ 0.          0.          0.          0.        ]]]

Convergence reached at episode:  31
```

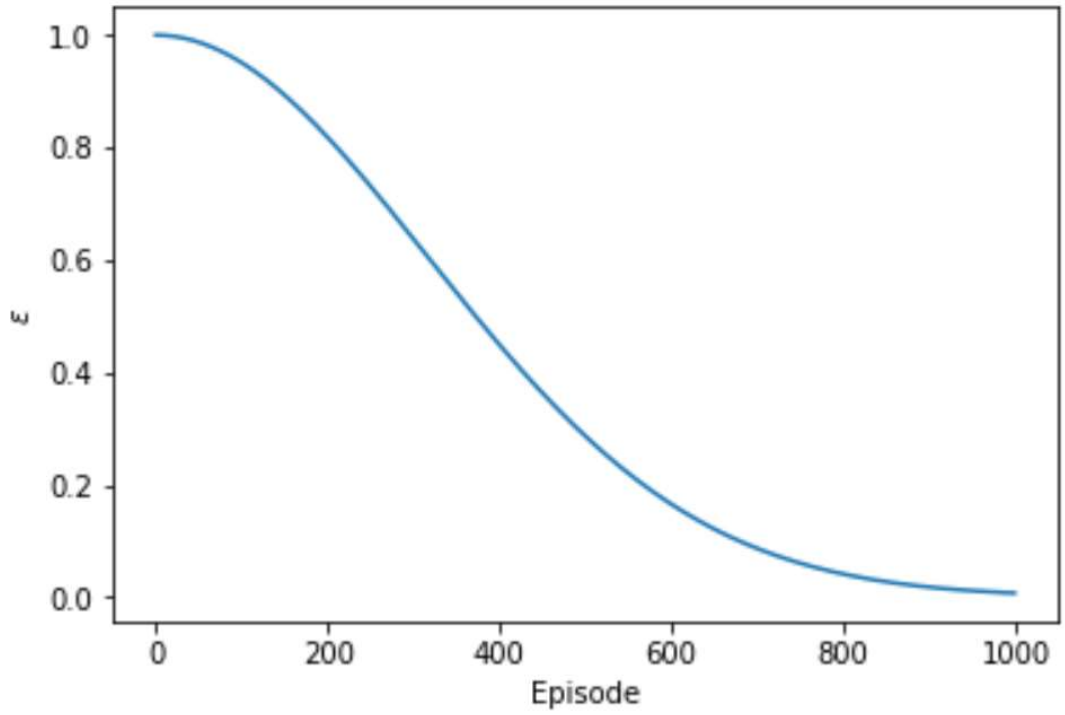

5
4
3
2
1
0
ε
0
200
400
600
800
1000
Episode

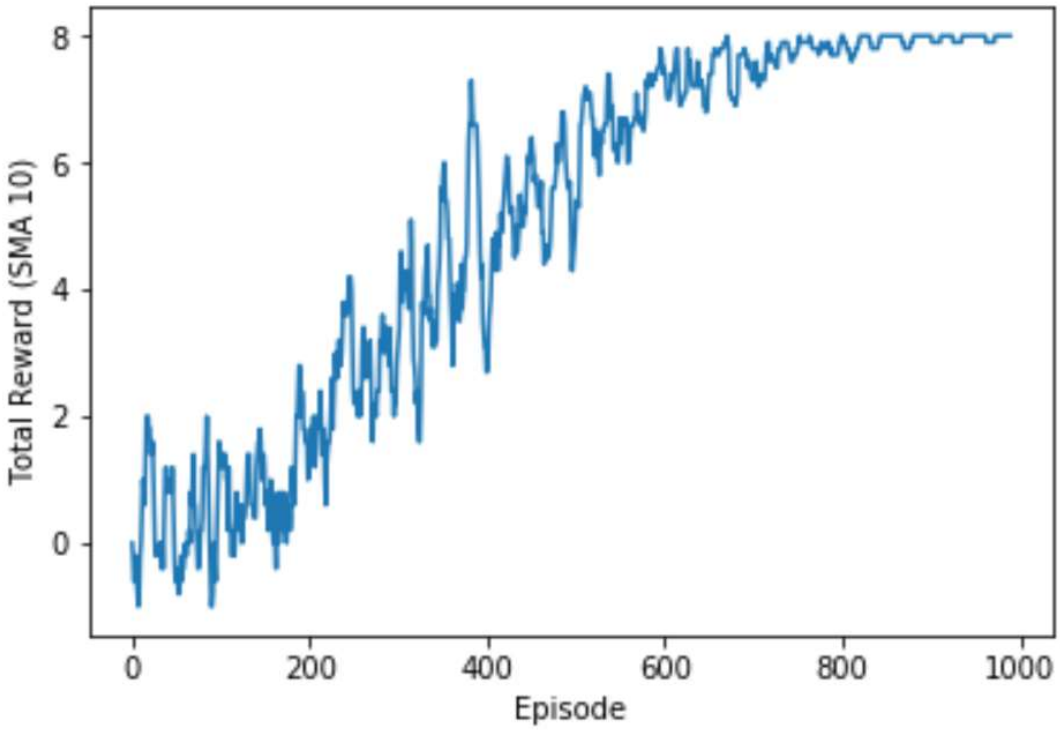

8
6
4
2
0
−2
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 6

epsilon=1.0
lr=0.1
gamma=0.9
total_episodes = 1000
decay = 0.00001

```
Final Q table
[[[ 5.33397587  3.83828464  5.6953279   3.95376383]
  [ 5.217031    3.5047705   4.6525556   3.90563897]
  [ 4.59643224  1.56255185  2.81116988  2.47049853]
  [ 3.67132768  0.27165012  1.09005723  0.80670361]
  [ 1.15544733 -0.14972884 -0.17096861 -0.15336069]]

 [[ 2.95834362  3.1766276   5.14713112  2.44386377]
  [ 3.70793027  3.51834992  4.68559     3.30977541]
  [ 3.90425397  2.8674253   4.0951      2.99658831]
  [ 3.439       1.7309689   2.5062322   2.38648373]
  [ 2.36381887 -0.23840667 -0.16224899  0.02651436]]

 [[ 1.63510961  1.40920573  2.67055808  0.91131983]
  [ 1.74339717  0.77952434  3.55901399  0.801879  ]
  [ 1.41404271  1.37082854  3.41572462  0.91056141]
  [ 2.71        1.70158921  2.45157541  1.64052038]
  [ 1.88106236 -0.0232838  -0.18834924  0.13671486]]

 [[ 0.90701954  0.43092806  0.81075491 -0.17698823]
  [ 1.28129742  0.04205775  0.59414311 -0.17453307]
  [ 1.48559636  0.04281972  0.65456677  0.01084757]
  [ 1.40131092  1.07036638  1.9         0.05529033]
  [ 1.          0.14223127 -0.18733376  0.33329735]]

 [[-0.37007911 -0.27722747  0.61493144 -0.60204051]
  [-0.11758113 -0.17447674  0.91006175 -0.1685908 ]
  [-0.2252341  -0.13153424  1.10583011 -0.25597175]
  [-0.22523401 -0.0176599   0.74581342  0.        ]
  [ 0.          0.          0.          0.        ]]]

Convergence reached at episode:  279
```

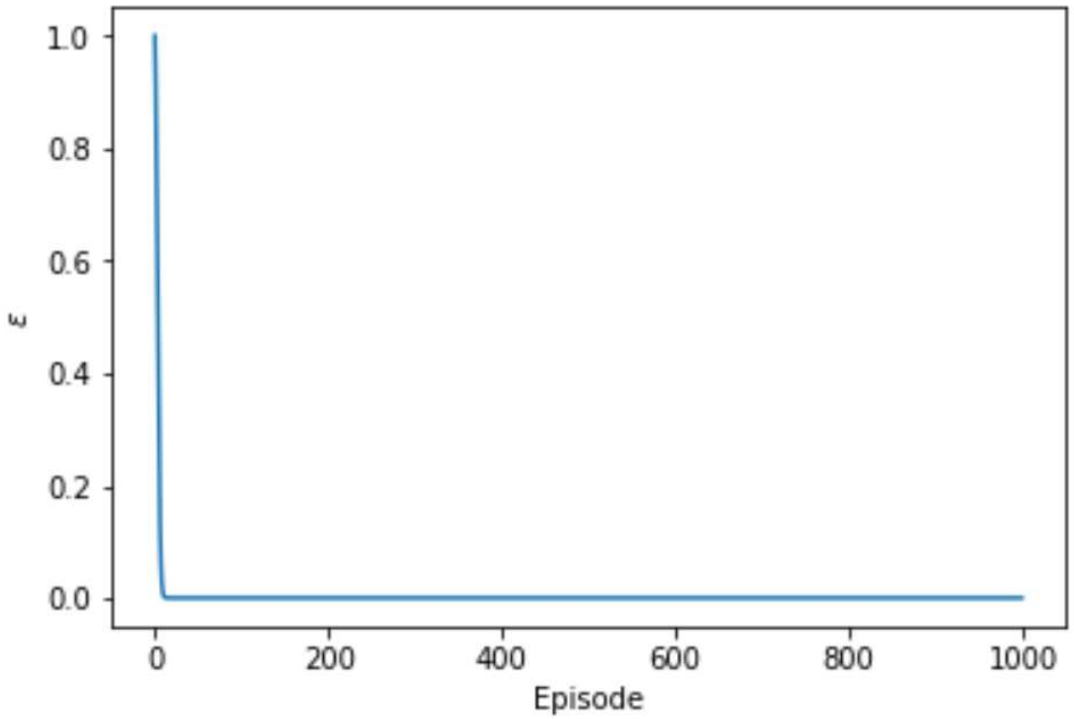

1.0
0.8
0.6
0.4
0.2
0.0
ε
0
200
400
600
800
1000
Episode

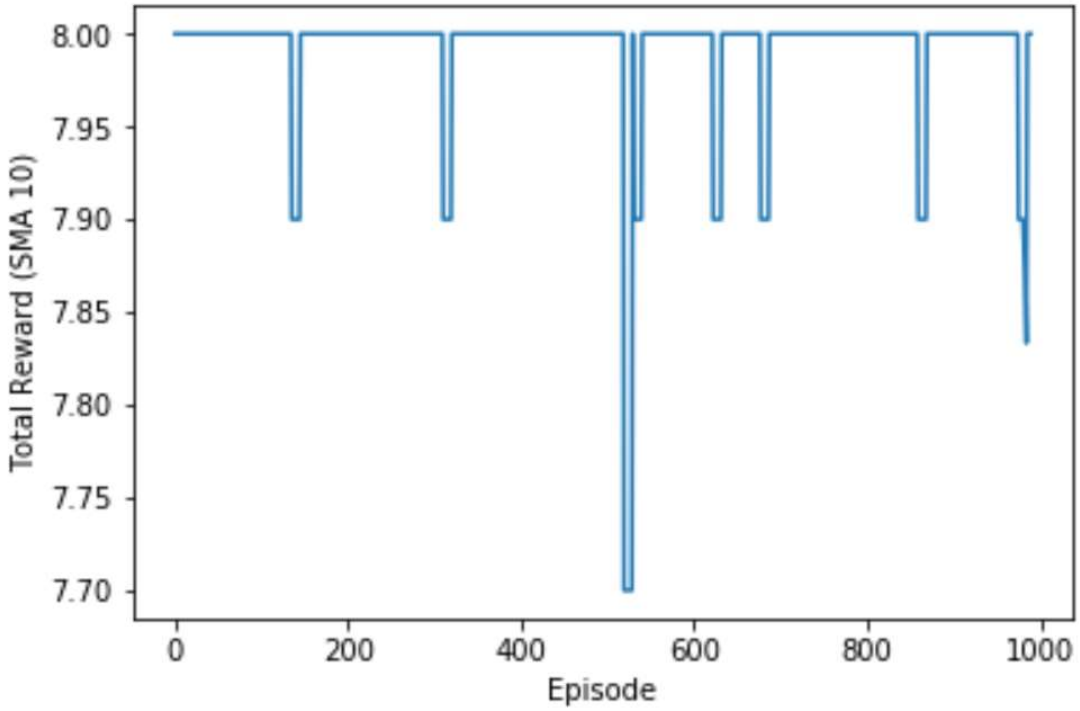

8
6
4
2
0
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

Case 7

epsilon=1.0
lr=0.1
gamma=0.9
total_episodes = 1000
decay = 0.1

```
Final Q table
[[[ 5.6953279   0.          0.30349    -0.32450633]
  [ 0.1         0.          0.42661     0.        ]
  [ 0.         -0.181       0.19       -0.269776  ]
  [ 0.1         0.          0.         -0.091     ]
  [ 0.          0.          0.          0.        ]]

 [[ 0.1         0.26564461  5.217031    0.0539    ]
  [ 4.68559    -0.068239    0.         -0.23122   ]
  [ 0.          0.          0.          0.        ]
  [ 0.1         0.          0.          0.        ]
  [ 0.          0.          0.          0.        ]]

 [[ 0.1         0.          0.          0.        ]
  [ 4.0951      0.          0.          0.        ]
  [ 0.          0.          0.          0.        ]
  [ 0.1         0.          0.          0.        ]
  [ 0.          0.          0.          0.        ]]

 [[ 0.19        0.          0.          0.        ]
  [ 3.439       0.          0.1         0.        ]
  [ 0.271       0.          0.          0.        ]
  [ 0.361       0.          0.          0.        ]
  [ 0.          0.          0.          0.        ]]

 [[-0.1        -0.091       0.3439      0.        ]
  [-0.1         0.20951     2.71        0.        ]
  [-0.1        -0.1         1.9         0.        ]
  [-0.10000002 -0.1729      1.          0.071     ]
  [ 0.          0.          0.          0.        ]]]

Convergence reached at episode:  6
```


1.0
0.8
0.6
0.4
0.2
0.0
ε
0
200
400
600
800
1000
Episode


8.00
7.95
7.90
7.85
7.80
7.75
7.70
Total Reward (SMA 10)
0
200
400
600
800
1000
Episode

4. Evaluation

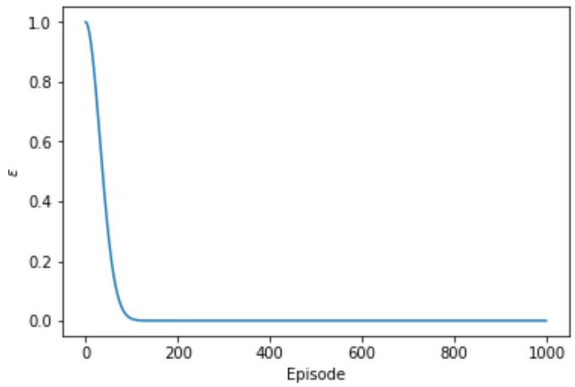
1. While tuning hyperparameters, as seen in Case6, if the decay rate is reduced too much, the agent is a slow and more no of episodes are required to reach the goal

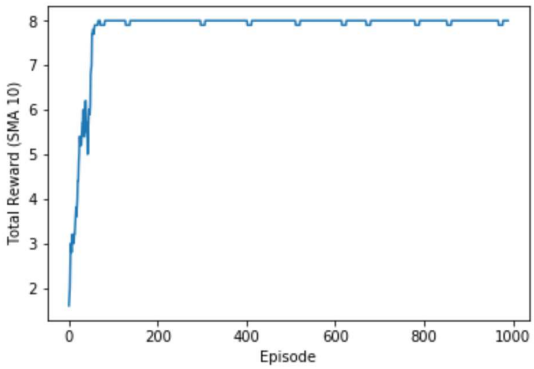
2. If the decay is increased to a large value, then the convergence is reached very fast, but the behavior is erratic, as can be seen in rewards graph of case 7

3. In case 3, if the learning rate is increased to 1.5, the agent takes more number of episodes to converge to goal (larger LR could mean overfitting) but as can be seen from case 1 and 2, for smaller value of LR, like 0.1, agent converges faster and behavior is smooth

4. In case 4, a low value of epsilon, lets agent converge to goal faster.

5. The optimum case is Case1, and the ideal epsilon and rewards curve are shown as below





Hence as can be seen, the epsilon decreases to a very small value and the rewards increase as the agent learns to reach the goal.

5. Assumption:

We have considered a decay rate of 0.001 for exponential decay of epsilon

Acknowledgments

Thankful to Professor Srihari and the group of Teaching Assistants who have helped us ace the projects.

References

1. http://gym.openai.com/docs/

2. https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/
3. https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56
4. https://towardsdatascience.com/reinforcement-learning-generalisation-in-continuous-state-space-df943b04ebfa