CSE 574 - Introduction to Machine Learning
Project 4 Report

Ajith Kumar Natarajan
Fall 2019
ajithkum@buffalo.edu

Abstract

In this project I have made developed a reinforcement learning agent that learns to navigate a 4x4 grid. The algorithm used to develop the reinforcement learning agent is Q-learning. The agent, without being provided with any model, learns the policy due to an action-based reward system. Throughout the process, the agent's goal is to maximize the total amount of rewards that it receives from taking actions in given states. Hence, the agent wants to maximize not just the immediate reward, but the *cumulative* rewards it receives over time.
The following observations were made:

- On an average, the reward collected by the agent increases along with episodes.
- After training for 1000 episodes, the agent consistently reaches the target in 8 steps, the minimum required with the possible directions (up, down, left, right) at each state.

1 Introduction

The working environment and framework for the learning agent is built with OpenAI's Gym library and Q-learning is implemented. At each step, the tabular Q-Learning approach utilizes a table of Q-values as the agent's policy to decide on the action for that state.

2 Markov Decision Process

Finite Markov Decision Process (MDP) is the framework used to depict the problem mathematically. It helps us in choosing sequential decisions.

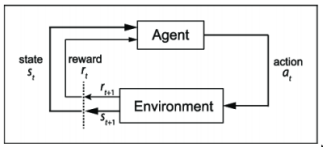

Figure 1: The canonical MDP diagram[1]

3 Environment

The 4x4 grid is the chosen environment. The environment design is implemented using OpenAI's Gym. In this environment, the agent (green square) has to reach the goal (yellow square) in the least

amount of time steps possible.
The environment's state space is described as n n matrix with real values on the interval [0, 1] to designate different features and their positions. The agent works within an action space consisting of four actions: up, down, left, right. At each time step, the agent takes one action and moves in the direction described by the action. The agent will receive a reward of +1 for moving closer to the goal and 1 for moving away or remaining the same distance from the goal.

4 Policy

A policy, is better than or the same as the policy, if the expected return of is greater than or equal to the expected return of for all states. A policy better than or the same as other policies is an optimal policy. A random number is generated between 0 and 1. If the number is greater than epsilon, agent chooses exploitation (action with highest Q-value). Otherwise, it chooses exploration. (exploring random actions and their effects).

5 Q-learning

Q-learning solves for the optimal policy in an MDP. The goal of Q-learning is to find the optimal policy by learning the optimal Q-values for each state-action pair.

5.1 Q-table

Q-table, to store the Q-values for each state-action pair. The horizontal axis of the table represents the actions, and the vertical axis represents the states. So, the dimensions of the table are the number of actions by the number of states. The learning process takes place at each episode. The Q-table is initialized with all values equal to zero.

6 Exploration and Exploitation

During training, the agent explores the environment in order to learn which actions will lead to maximal future discounted rewards. Agent's begin exploring the environment by taking random actions at each state. Doing so, however, posed a problem: the agent may not reach states which lead to optimal rewards since they may not take the optimal sequence of actions to get there. In order to account for this, the agent is slowly encouraged to follow it's policy in order to take actions it believes will lead to maximal rewards. This is called exploitation, and striking a balance between exploration and exploitation is key to properly training an agent to learn to navigate an environment by itself.

Epsilon Greedy Strategy gives the agent a way to choose between Exploration and Exploitation. Exploration Rate, is initially set to 1. With 1, it has the 100% certainty that the agent would start exploring. As the agent starts to learn about the environment, a decay occurs in the where the likelihood of exploration reduces. The agent becomes greedy with exploiting the environment as it gets the opportunity to explore and learn more.

7 Training process

Each episode is built of steps. For each episode, the environment state is changed to the starting state. If the number is greater than epsilon, agent chooses exploitation (action with highest Qvalue). Otherwise, it chooses exploration (exploring random actions and their effects)

7.1 Parameters

The parameters like , (max and min), and the number of episodes is used in the project.
Episodes: For episodes the initial value provided was 10000. However, the agent could reach the target after completing around 7000 episodes, by taking approximately 0.24 sec. for each episode.

```
[[[ 2.87908543 -9.56091812  3.0389184  -9.61854945]
  [ 3.18422293 -7.6000752   2.26695695 -9.60930174]
  [ 2.96455323 -3.8888276   1.42035914 -8.63412192]
  [ 2.81936754 -0.95617925  0.49233802 -4.27334191]
  [ 0.91359069 -0.3940399  -0.4900995  -0.20951862]]

 [[ 2.0837258  -9.38759692  3.18890838 -7.89401554]
  [ 2.42474324 -7.68680351  2.42457848 -8.07943457]
  [ 2.15677837 -4.35614268  1.57481801 -8.02352292]
  [ 2.25513632 -1.53041395  0.6054582  -5.35681126]
  [ 1.36620589 -0.73011839 -0.67934652 -0.55743032]]

 [[ 1.19233975 -8.78076209  3.00388588 -4.07033554]
  [ 1.57751552 -7.35669283  2.16732604 -4.62177376]
  [ 1.27850546 -4.68514667  1.29087564 -5.11910462]
  [ 1.3602192  -2.12394763  0.32296677 -5.41666967]
  [ 1.53418948 -0.90225324 -0.86482753 -2.26044405]]

 [[ 0.18445228 -5.60706101  2.89353989 -2.52827906]
  [ 0.62178486 -6.15243351  2.34381752 -2.71480386]
  [ 0.285703   -4.26265099  1.34543494 -2.81934028]
  [ 0.39247518 -2.54709273  0.23930911 -2.99502559]
  [ 0.96184796 -0.90180354 -0.95617925 -1.955088  ]]

 [[-1.04661746 -1.87303947  0.92449315 -0.3940399 ]
  [-0.58519851 -2.28229409  1.31584464 -0.49951   ]
  [-1.22478977 -0.3956043   1.44539161 -0.71680815]
  [-0.67934652 -1.42841954  0.99868998 -0.75150106]
  [ 0.          0.          0.          0.        ]]]
```

Figure 2: Obtained Q-table

Exploration factor : This parameter allows the environment to explore a lot in the agent. Irrespective of the min and max values, as the agent starts with '0' steps, the S = 0, which leads to maximum = 1 in the initial state. As the agent keeps learning the value exponentially reduces. The speed of decay of the is determined by parameter.

8 Challenges

The epsilon should be decreasing for every episode, so that the exploration is done, and exploitation is increased. As the agent learns more and more, it can get to the target faster.

The different values were given to the min_epsilon and max_epsilon and the variations were observed. It was a challenge to get a non-increasing episode vs epsilon graph and to get a highest point for reward in the last episode. Thus, the values were changed and observed to get the optimal path.

9 Result

The result is obtained by changing the hyperparameters, max_epsilon and min_epsilon.

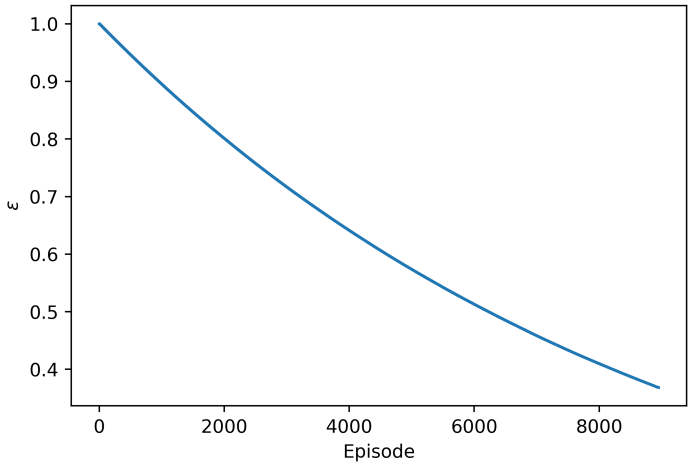

Figure 3: Episodes vs epsilon (min_epsilon = 0, max_epsilon = 1)

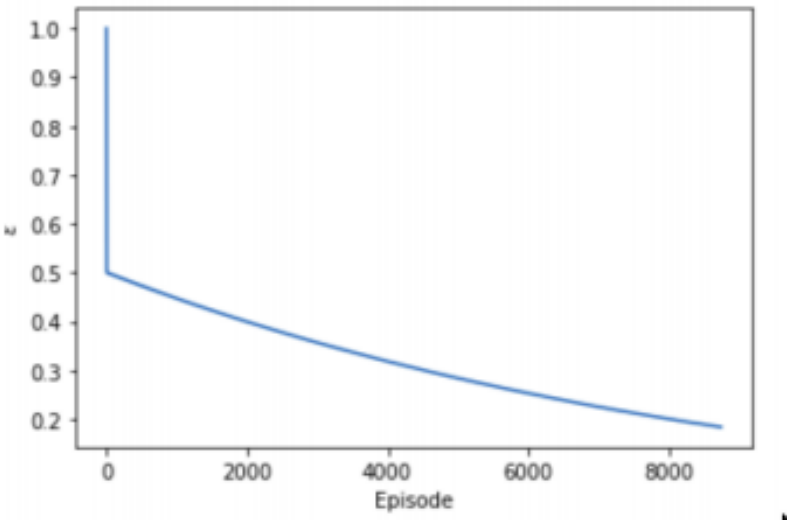

Figure 4: Episodes vs epsilon (min_epsilon = 0.5, max_epsilon = 1)

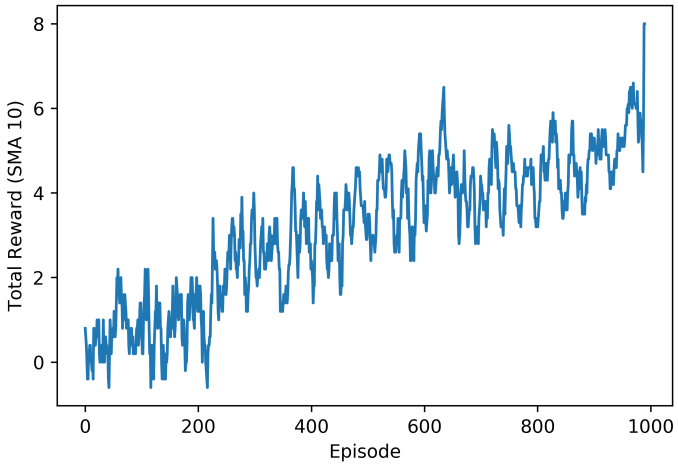

Figure 5: Episodes vs cumulative reward

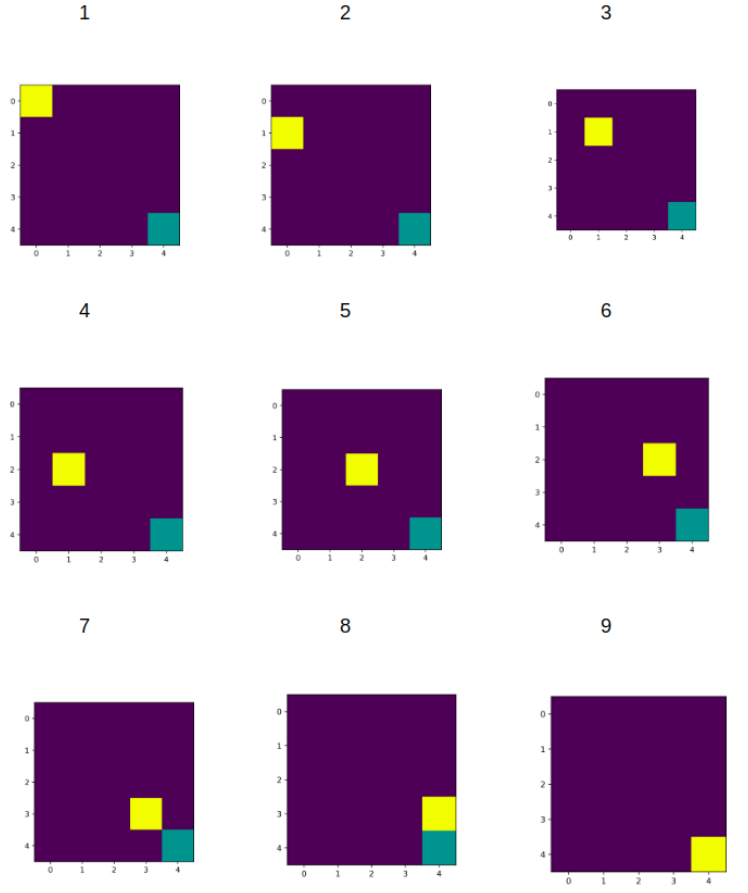

Figure 6: Steps taken by the agent to reach the goal

10 Conclusion

Reinforcement learning was performed on the agent in the 4x4 environment to make it reach the target. The agent was trained for 1000 episodes and on an average, its cumulative reward was observed to increase. The agent was able to meet the target at the end within the lease number of steps possible (8) with the given movement possibilities.

References

[1] Srihari, Sargur N. Lecture Slides for Machine Learning

[2] https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56

[3] https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/