Reinforcement Learning

Dhayanidhi Gunasekaran
University at Buffalo
Buffalo, NY 14214
dhayanid@buffalo.edu

Abstract

To teach the agent (Tom) to reach the target (Jerry) in a grid environment using Reinforcement Learning and Deep Learning.

1 Problem Statement

Given a grid environment of size 5 * 5 in which the agent (Tom) and the Target (Jerry) are initialized at random positions. The task is to teach the agent to reach the target changing state after state towards the target attaining reward points at every successful move. This could be done with the combined process of Reinforcement Learning and Deep Q Networks. And to answer the given questions related to Reinforcement learning and Fill the Q-table.

2 Solution

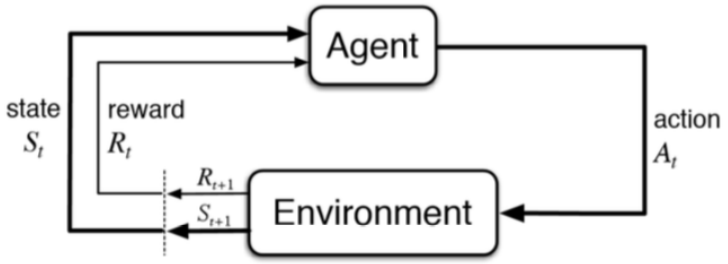
The given task can be completed by the combination of Reinforcement Learning, Deep Q Network and Neural network each performing specific task which contributes in reaching the target.

2.1 Environment

The environment is a 5 * 5 gird with states from (0,0), (0,1) … (4,4). There are 4 actions- left, right, up and down. Initially the agent and the target are initialized at random states and the objective of the agent is it reach the target attaining maximum reward as possible. Whenever the agent choses the correct action it gets rewarded by positive value while on taking a wrong step leads to negative reward.

2.2 Reinforcement Learning

Reinforcement learning is an area in machine learning in which the agent tends to take an action in order to maximize the cumulative reward of the task which is being performed. In this task we use Markov's decision process to calculate the reward for every action.



Markov's decision process is modeled as a 5 tuple process (S ,A, P, A, γ).

Where S – Finite set of states
A – Finite set of actions
P – Probability that an action 'a' in state 's' at time t will lead to state s' at t+1
R – Immediate Reward received after transition.
γ - Discounting factor.

In this process, whenever an agent wants to make a decision on which action it has to make, it will run the Markov's decision process and obtain the probabilities for all the possible actions that can take and will perform the action with larger probabilities.

Markov's process can be written as

$$\begin{aligned} Q_\pi(s,a) &\doteq \mathbb{E}_\pi[G_t|S_t = s, A_t = a] \\ &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s, A_t = a\right] \end{aligned}$$

Where Q(s,a) is an Action value function. The Reward value will be smaller while it gets larger value upon learning from experiences.

2.2 Deep Q Network

So far, we have computed the action reward based on the probabilities and no learning has been done. We need to store the past experiences so that the agent can learn based on its past experiences.

Deep Q Network is a kind of memory in which all the past state, action is stored so that is can be used to perform the action learning.

It also helps increasing the learning speed of the algorithm. The experiences are stored as a tuple <state, action, reward, next state>.

2.3 Neural Network

A 2-layer Neural network with 128 nodes each has been used to train the agent in getting the reward value for each action for the particular state. It uses the memory obtained from Deep Q network to train the model.

The input is the dimension of possible states and the output is the number of actions with their probabilities.

The activation layer used is relu,relu and Linear respectively for first, second and the final layer.

2.4 Exploration vs Exploitation

The agent initially will randomly select the action based on the exploration rate or epsilon. It is better for the agent to initially try all the possible actions before it learns from the experiences. The epsilon decay formula can be given as

$$\epsilon = \epsilon_{min} + (\epsilon_{max} - \epsilon_{min}) * e^{-\lambda|S|},$$

The epsilon holds larger value initially and decrease eventually.

λ - Hyperparameter for epsilon

|s| - total number of steps.

2.5 Steps

All the states and actions are initialized and the number of episodes set.
For each episode,
Initialize the states to s0,
Calculate the Q value for all the states, actions until q value = r
All the scenarios are stored in the memory.
Neural Network is trained on the sample batches from the memory

3 Tuning Hyperparameters

The hyper parameters that can be tuned in λ, and the number of episodes. The graphs of epsilon and the reward for various λ are shown below.

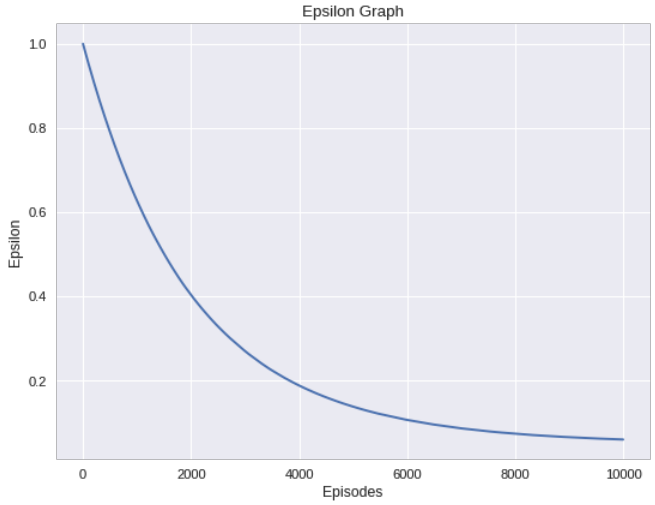


Figure 1: Epsilon vs Number of Episodes for λ = 0.00005

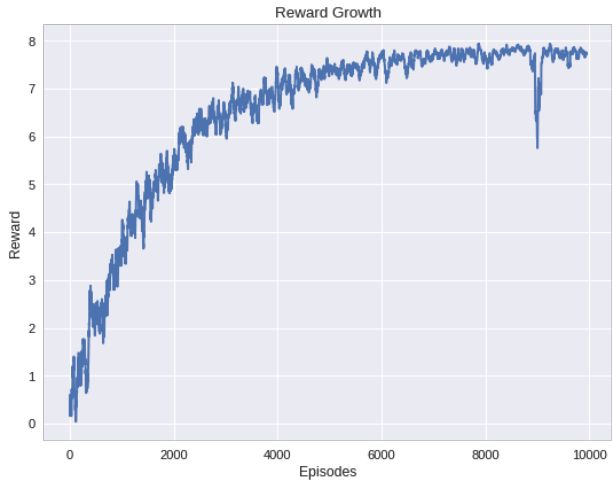


Figure 2: Rewards vs Episodes for λ = 0.00005

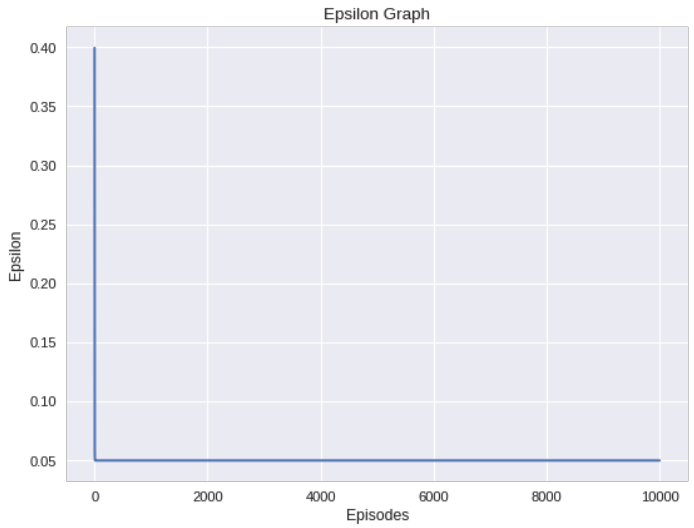


Figure 3: Epsilon vs Number of Episodes for λ = 0.1

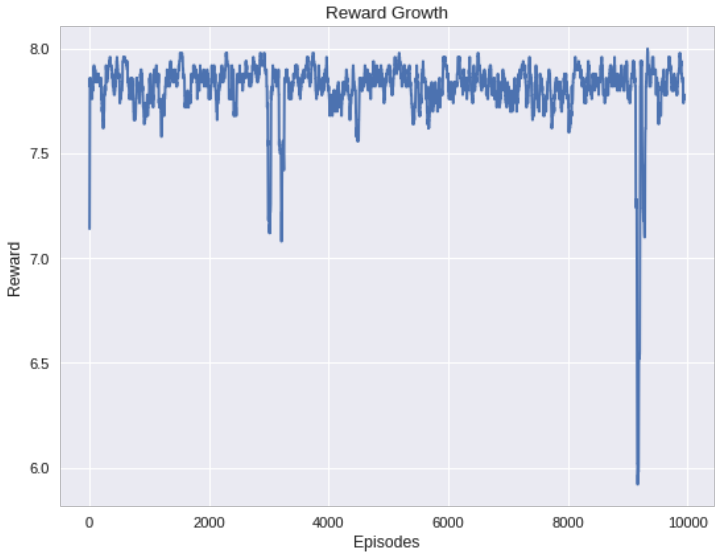


Figure 4: Reward vs Number of Episodes for λ = 0.1

4 Questions

1. if the agent always chooses the action that maximizes the Q-value ?

   If the agent didn't learn anything from the past experiences or the optimal policy has not been found out, the agent might not reach the target while getting stuck up at some non-optimal actions.

   To force the agent to explore,

   1. It should decide the actions based on the exploration rate instead of Q value for initial learning.
   2. To set a function of q- value in which the agent explores all the possible states.

2.Q-table for the given environment

State	Up	Down	Left	Right
0	3.9	3.94	3.9	3.94
1	2.94	2.94	2.9	2.97
2	1.94	1.99	1.94	1.99
3	0.97	1	0.97	0.99
4	0	0	0	0

References

[1]https://medium.freecodecamp.org/diving-deeper-into-reinforcement-learning-with-q-learning-c18d0db58efe

[2] Keras Documentation - https://keras.io/utils/

[3]https://medium.freecodecamp.org/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8