

Seminar/Case Study

FinalReport

Q - Learning

By

Nimesh Choudhary

190135

Supervisor

Dr. Soharab Hossain Shaikh

Associate Professor SOET - CSE



**BML MUNJAL
UNIVERSITY™**

**Department of Computer Science and Engineering
School of Engineering and Technology**

December, 2021

Acknowledgement

I would like to express my sincere gratitude to my supervisor Dr. Soharab Hossain Shaikh for guiding me throughout the duration of this course and for providing me with the necessary resources that enabled me to complete the project.

I would like to express my sincere gratitude to Prof. Manoj K. Arora, Vice Chancellor and Dr. Anirban Chakraborti, Dean SOET for the facilities provided to accomplish this project.

I would also like to express my gratitude to my peers who provided me with honest and valuable feedback & review of my application during its development.

Last but not the least, many thanks go to the faculty at BML Munjal University for giving me right guidance and teaching me the right skills that enabled me to meet the requirements of this course.

Thanking You.

Nimesh Choudhary, 190135

BTech 2019, CSE

12th December, 2021



BML Munjal University, Gurgaon, Haryana

CANDIDATE'S DECLARATION

I, **Nimesh Choudhary**, hereby declare that the work done in my seminar/case study/project entitled **Q - learning** in fulfillment of completion of 5th semester of Bachelor of Technology (B.Tech) program in the Department of Computer Science and Engineering, BML Munjal University is an authentic record of our original work carried out under the guidance of **Dr. Soharab Hossain Shaikh, Associate Professor SOET - CSE.**

Due acknowledgements have been made in the text of the project to all other materials used.

This seminar/case study/project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Nimesh Choudhary

190135

Place: Gurugram, Haryana

Date: 12-12-2021

CERTIFICATE

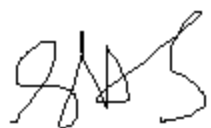
This is to certify that the Seminar/Case Study/Project entitled **Q-learning** to the best of my knowledge is a record of the bonafide work carried out by Mr. **Nimesh Choudhary** under my guidance and/or supervision. The contents embodied in this report, to the best of my knowledge, have not been submitted anywhere else in any form for the award of any other degree or diploma. Indebtedness to other works/publications has been duly acknowledged at relevant places. The work was carried out during July -December 2021 as part of their 5th semester coursework for Bachelor of Technology (B.Tech) program in the Department of Computer Science and Engineering, BML Munjal University.

Name and Designation of the Supervisor:

Dr. Soharab Hossain Shaikh

Associate Professor SOET - CSE

Signature:



Date: 12- December - 2021

Place: Gurugram, Haryana

Contents

| S.No. | Title | Page No. |
|--------------|---|-----------------|
| 1. | Abstract | 6 |
| 2. | Problem Definition | 7 |
| 3. | Objectives | 7 |
| 4. | Challenges in Q - learning | 7 |
| 5. | Literature Review | 8 |
| 6. | Methodology | 11 |
| 6. | Dataset | 13 |
| | i. Automobile Factory analogy | 13 |
| | ii. OpenAI - Gym library's interface | 15 |
| 7. | Experimental Results | 17 |
| | i. Implementing the Automobile Factory analogy | 17 |
| | ii. Implementing the OpenAI - Gym library's interface | 17 |
| | iii. Implementing the Tic - tac toe game | 19 |
| 8. | Conclusion | 18 |
| 9. | Reference | 18 |
| 10. | Plagiarism Report | 19 |

Abstract

This report is based on the Q - learning algorithm. Q learning is a type of reinforcement learning which helps us to learn an action policy that agents need to choose to maximize its reward.

In this report we are going to know the methods used in Q - learning like implementation of Bellman Equation used to calculate the optimal policy, Data structure use by Q - learning and little bit use of Markov Decision process.

In this report we will also see what will be the impact of Q - learning on the training data, how Q - learning improve the data and make them optimize.

Problem Definition

To explore how an autonomous agent senses and acts in its environment and learns to choose optimal actions to achieve its goal.

Objectives

The main objective of q-learning is

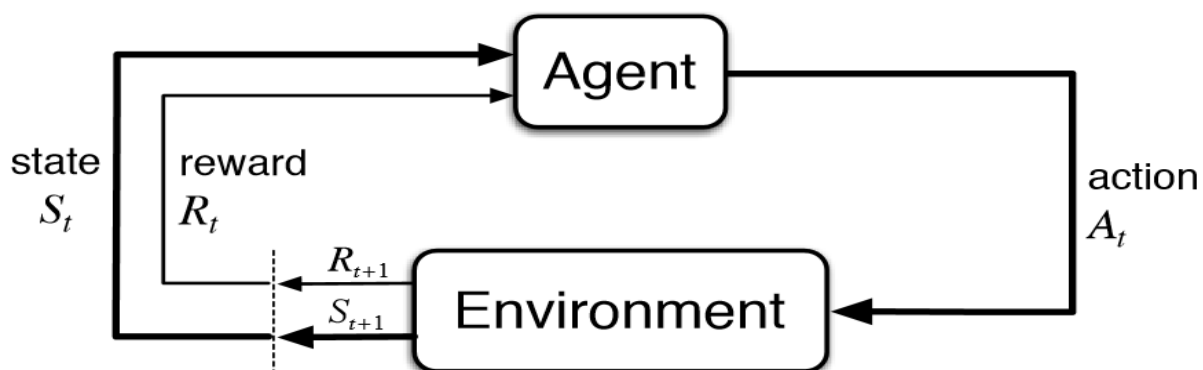
- to learn an optimal policy for an arbitrary environment
- to learn a numerical evaluation function defined over state and actions
- What evaluation function should agents attempt to learn?

Challenges

- Q learning works only in an environment having discrete and finite states and action space.
- So, generally Q - learning uses Q - table data structure to optimize the solution. When the number of states are very high then it is very difficult to use the Q - table so for that we need to use the Deep neural network instead of Q - table.

Literature Review

Q-learning is a model-free reinforcement learning algorithm and value-based method which is used to learn the value of an action using Q-values which computes the expected reward from an action taken in a given state and also used to inform the agent which action he should take.



The common terms used in q - learning are:

- **Environment** - surrounding of the agent in which it perform different actions
- **State** - particular situation happen in the environment
- **Reward** - getting feedback from the environment
- **Value** - future reward that an agent get while performing an action on particular state
- **Policy- $f: S \rightarrow A$** selecting the appropriate action a_t for particular state s_t {where S represents the set of state and A represents the set of action} i.e. $\pi(s_t) = a_t$.

So agent a understands a set S of different states in its environment and selects the action from set A that agent needs to perform.

At finite time t , the agent observe the state s_t and choose the action a_t where a_t represents the particular action at time t and performing some action on the environment so in the result we receive some rewards $r_t = r(s_t, a_t)$ and produce the next state $s_{t+1} = \delta(s_t, a_t)$ where r and δ functions not known to the agent.

So as we know that Q-learning algorithm is a value based method. In this method our main objective is to optimize $V(s)$ which is known as a value function. So the value function helps to find what will be the future reward we get after performing some actions at a particular state.

Now question comes which policy π Would the agent learn? So, our main approach is to get the appropriate policy (need to be optimal) such that we get the maximum possible reward(cumulative). So cumulative value $V^\pi(s_t)$ defined by

$$\begin{aligned} V^\pi(s_t) &\equiv r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots \\ &\equiv \sum_{i=0}^{\infty} \gamma^i r_{t+i} \end{aligned}$$

where r_t represents the reward at state s_t by repeatedly using the policy π to select the appropriate actions and where γ represents the constant which determines the relative values of intermediate reward and range between $0 \leq \gamma < 1$

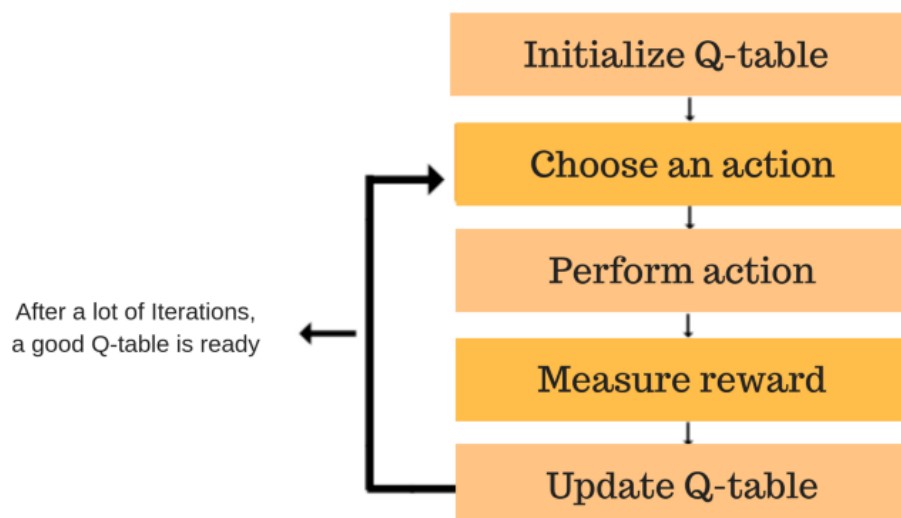
So, our main task of using Q - learning is used to optimize the policy π that maximize the $V^\pi(s)$ for all states and which is denoted by π^* and defined as

$$\pi^* \equiv \operatorname{argmax}_{\pi} V^\pi(s), (\forall s)$$

So the optimal action that apply on the state s is the action a that maximizes the sum of intermediate reward $r(s, a)$ plus the value V^* of the next state and discounted by factor γ

$$\pi^*(s) = \operatorname{argmax}_a [r(s, a) + \gamma V^*(\delta(s, a))]$$

So, the data structure used in Q- learning is called Q-table which is used to calculate the maximum future reward for a particular action at a particular state. The approach used by the Q - table is an iterative approach. The steps we have to followed for calculating the entries of Q - table are:



Initially all values we need to put are 0s. So, at every iterative approach we need to update the Q-table which is of the order $[n \times m]$ where n and m represent the number of possible states and actions respectively.

Methodology

Algorithm behind Q - learning -

Before learning the algorithm of Q - learning we need to know what is evaluation function $Q(s, a)$ which is used to calculate the maximum cumulative future reward we get from a particular state s after applying action a on it. So, Q - function is defined by

$$Q(s, a) \equiv r(s, a) + \gamma V^*(\delta(s, a))$$

so, the above equation is also known as the bellman Equation.

The key problem is to find a appropriate way to estimating training values for Q, which gives the intermediate reward r spread over time which uses the iterative approach not recursively So the relation between Q and V^* is.

$$V^*(s) = \max_{a'} Q(s, a')$$

Then from above relationship we can write as

$$Q(s, a) = r(s, a) + \gamma \max_{a'} Q(\delta(s, a), a')$$

So the recursive definition of Q provides the basis for an algorithm that iteratively approximates Q which is denoted by Q' which refers to the learner's hypothesis. So the agent observe the state s and

choose some action a_t and get some reward $r = r(s, a)$ and the next state will be $s' = \delta(s, a)$ and then we will update the entry for Q' according to the rule

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

So in short the conclusions shown below

Q learning algorithm

For each s, a initialize the table entry $\hat{Q}(s, a)$ to zero.

Observe the current state s

Do forever:

- Select an action a and execute it
- Receive immediate reward r
- Observe the new state s'
- Update the table entry for $\hat{Q}(s, a)$ as follows:

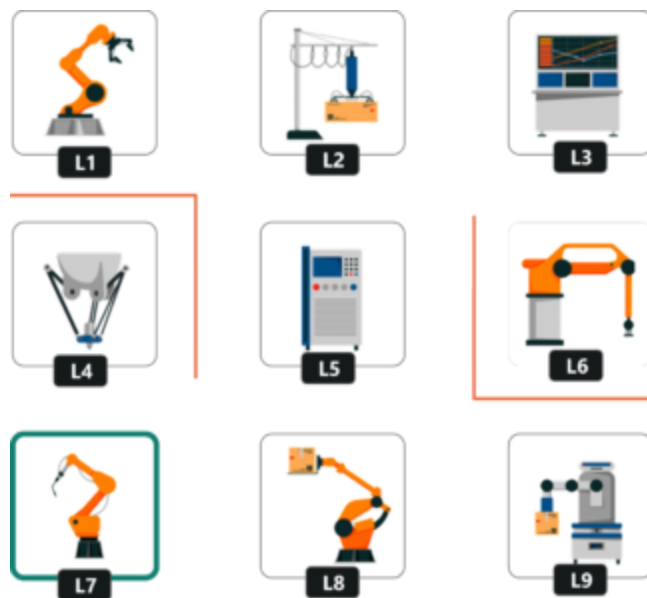
$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

- $s \leftarrow s'$

Description of the Dataset (if applicable/available)

1. Automobile Factory analogy

In this factory there are different parts of a vehicle located at different 9 locations. The parts are wheels, dashboard, engine and so on. So there is one factory master which prioritized the location where different part are installed at the highest priority and there are some blockage(hurdles) between the station as shown below



so from this you can't go directly from L4 to L1 so for that you need to go through L7, L8, L5 and L2.

So from the above table, we have all the possible rewards that a robot gets from one to another state by changing its location. Let assume L6 will be our topmost priority location so, it will give the highest reward value as compared to others. So Let's put the reward value **999** in the cell (L6,L6). The reward table will be

| | L1 | L2 | L3 | L4 | L5 | L6 | L7 | L8 | L9 |
|----|----|----|----|----|----|-----|----|----|----|
| L1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| L2 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| L3 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| L4 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| L5 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| L6 | 0 | 0 | 1 | 0 | 0 | 999 | 0 | 0 | 0 |
| L7 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| L8 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| L9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

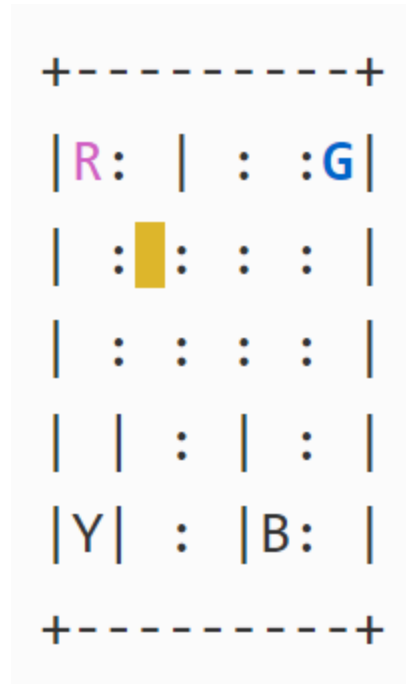
Table of rewards with a higher reward for the topmost location

So from this dataset with help of Q-learning we need to find out the optimal path that the robot needs to choose from starting location to goal location.

2. OpenAI - Gym library's interface

So Gym provides different types of environments that we can plug into our code and test it. This library takes care of API which provides us all the information about the environment, the number of possible states and actions present in it.

So in this tutorial we are going to use the Taxi - v2 Gym environment.



So the above diagram tells us the particular state at any time t . where R, Y, G and B tells the pickup and drop off location. Where the yellow color indicates the taxi.

So, there are 4 different locations of passenger where passenger can be picked up and maybe passenger's destination location and one taxi. So, the taxi can move in forward(north), backward(south), right(east) and left(west) directions. So, there are a total 6 actions (south, north, east, west, pickup and drop off). There are 500 possible states.

when taxi moves left, right, forward and backward at once then the reward will be -1, and when pickup/dropoff action performs then we will get -20 as reward. So suppose if taxi has a passenger and it need to perform dropoff (where passenger want to dropoff) action then we will get reward as 20.

Experimental Result

1. Implementing Q - learning algorithm of Automobile Factory Analogy in python using numpy

Suppose a robot wants to go from Location L1 to L9 so by using python 's numpy library and implementing the Q - learning algorithm on it then Q - table above dataset and the optimal path chosen by robot shown below.

```
Optimal Path -> ['L1', 'L2', 'L5', 'L8', 'L9']
Q-Table ->
[[ 0.          1267.08330176  0.          0.          0.
  0.          0.          0.          0.          ]
 [ 951.31231932  0.          951.31215236  0.          1688.11106913
  0.          0.          0.          0.          ]
 [ 0.          1267.08329304  0.          0.          0.
  714.47822447  0.          0.          0.          ]
 [ 0.          1267.08330157  0.          0.          0.
  0.          1688.11678477  0.          0.          ]
 [ 0.          1267.08330184  0.          0.          0.
  0.          0.          2249.48143351  0.          ]
 [ 0.          0.          951.31239925  0.          0.
  0.          0.          0.          0.          ]
 [ 0.          0.          0.          1267.08511677  0.
  0.          0.          2249.48905087  0.          ]
 [ 0.          0.          0.          0.          1688.11107484
  0.          1688.11642553  0.          2997.9854017 ]
 [ 0.          0.          0.          0.          0.
  0.          0.          2249.48905124 3995.99299882]]
```

2. Implement the Q - learning algorithm in python using OpenAI - Gym

So if we want to know what the possible states will be, which action we need to perform and what reward we will get at every state. So by using the OpenAI gym, IPython.display and time library we can, which is shown below.

```

+-----+
|R:  |  :  :G|
|  :  :  :  |
|  :  :  :  |
|  |  :  :  |
|Y|  :  |B:  |
+-----+
      (Dropoff)

Timestep: 1
State: 328
Action: 5
Reward: -10

```

As we clearly see, it takes thousands of timesteps and performs wrong drop offs to deliver just one passenger to the right destination. So with the help of Q - learning we can train it and reduce the time steps taken by it agent

So after applying Q - learning we get the output

```

Results after 100 episodes:
Average timesteps per episode: 12.3
Average penalties per episode: 0.0

```

before applying Q - learning the output will be

Episode: 100000

Training finished.

Time steps taken: 1117

Penalties incurred: 363

Wall time: 30.6 s

So from these two we clearly see that earlier there were 363 penalties done by a taxi agent and now there are 0.0 penalties/100 episodes done by taxi agent. So as we clearly see from the evaluation, the agent's performance improved significantly which means it performed the correct pickup and drop off actions.

3. Implement the Q - learning algorithm on Tic Tac Toe game

So, there are three possible cases in this.

1. So in this blue color represents the current move and red colors represents the next possible move.



Block

2. so the starting state will be shown below, where all the cells are of blue color because we can mark on any cell.



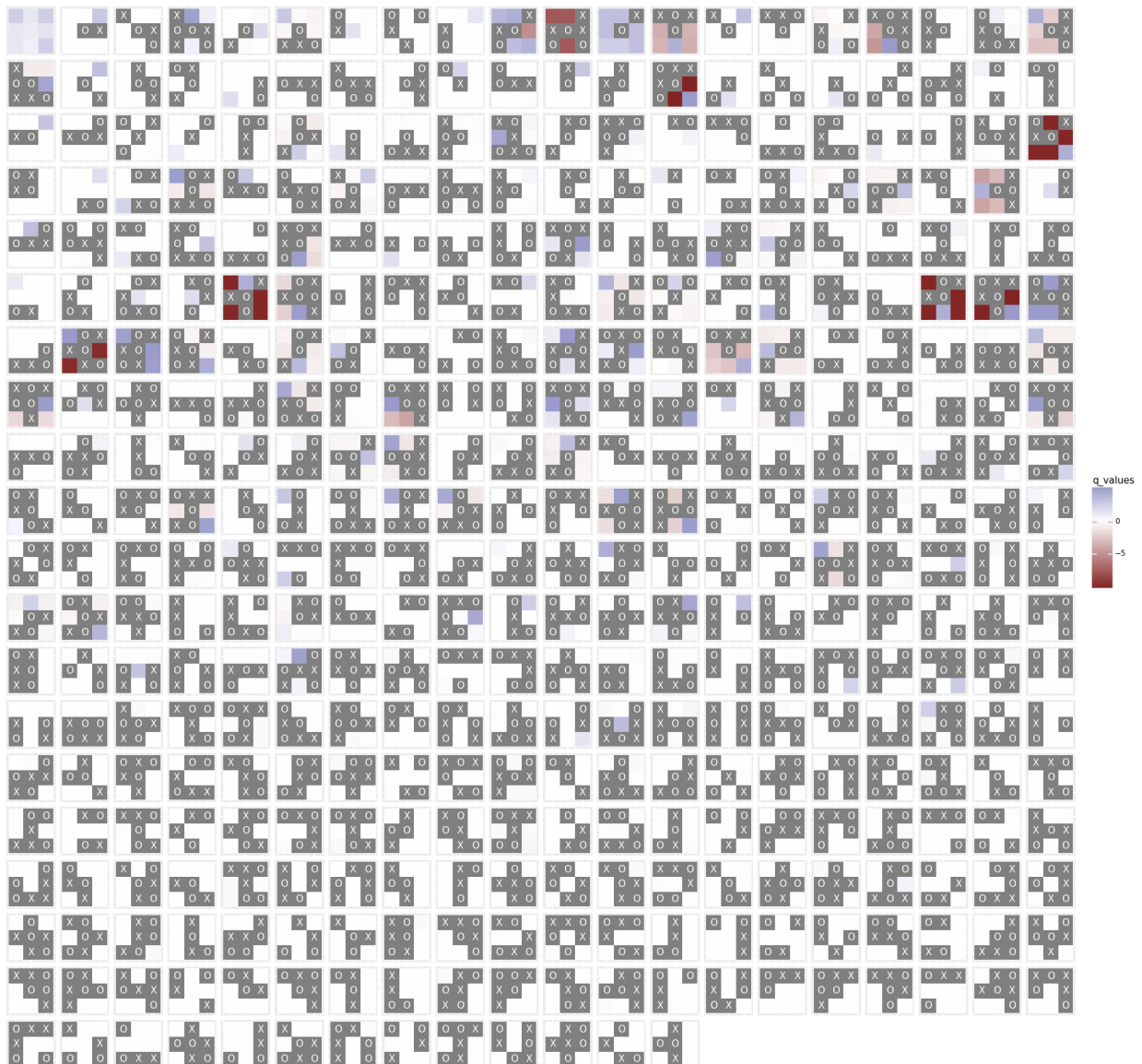
Start

3. so, in the third which is shown below. Where the blue mark represent the current move where the game will be over and skin color will represents that there will be no next possible move and game is over.



Block

So the Q - table of the Tic Tac Toe game is shown below which tells the total number of possible moves or states.



So for making the move you need to input/ enter the row number and column number where you want to mark. So after marking, the computer will automatically make the move which is shown below.

```

-----
|   |   |   |
-----
|   |  x |   |
-----
|   |   |   |
-----

```

Input your action row:2

Input your action col:2

```

-----
|   |   |   |
-----
|   |  x |   |
-----
|   |   |  o |
-----
-----
|   |   |   |
-----
|   |  x |   |
-----
|   |  x |  o |
-----

```

Input your action row:0

Input your action col:1

Conclusions and Future Scope

Q - learning is a type of reinforcement learning which uses action - values (also known as Q - values). It is used to optimize the behavior of learning agents. Q learning uses off-policy reinforcement learning to find out the best action for the current state. So, Q - learning is a value based method and model free reinforcement learning in which we get the reward after performing some action on the state.

Nowadays, Q - learning plays a vital role in Artificial Intelligence. Q - learning also helps us to build/ create the tools for building the intelligent agents. Q - learning is also used to build AI games. So, Q - learning is not limited to games, sometimes it is also used for managing stock portfolios and finances, for making human robots, etc.

References

1. An introduction to Q-Learning: Reinforcement Learning by Sayal Paul-

<https://blog.floydhub.com/an-introduction-to-q-learning-reinforcement-learning/>

2. Reinforcement Learning -

https://en.wikipedia.org/wiki/Reinforcement_learning

3. An Introduction to Q - learning : Reinforcement Learning by Daniel -Cheung

<https://www.freecodecamp.org/news/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc/>

4. Q - learning in python by geeksforgeeks -

<https://www.geeksforgeeks.org/q-learning-in-python/>

5. Book of Machine Learning written by Tom Michell - <https://www.cin.ufpe.br/~cavmj/Machine%20-%20Learning%20-%20Tom%20Mitchell.pdf>

6. Reinforcement Learning - Implementing TicTacToe by Jeremy Zhang -

<https://towardsdatascience.com/reinforcement-learning-implementation-tictactoe-189582bea542>

Plagiarism Report

| Class Homepage | | | | |
|--|-------------------|---|----------------|--|
| <p>This is your class homepage. To submit to an assignment click on the "Submit" button to the right of the assignment name. If the Submit button is grayed out, no submissions can be made to the assignment. If resubmissions are allowed the submit button will read "Resubmit" after you make your first submission to the assignment. To view the paper you have submitted, click the "View" button. Once the assignment's post date has passed, you will also be able to view the feedback left on your paper by clicking the "View" button.</p> | | | | |
| Assignment Inbox: B.Tech. 2019 (Dr. Kiran) | | | | |
| Assignment Title | Info | Dates | Similarity | Actions |
| B.tech 2019 Batch | ? | Start 12-Dec-2021 9:24PM Due 19-Dec-2021 11:59PM Post 20-Dec-2021 12:00AM | 5% <div></div> | Resubmit View Download |

Seminar/Case Study

FinalReport

Q - Learning

By

Nimesh Choudhary

190135

Supervisor

Dr. Soharab Hossain

<Designation>

Match Overview

5%

1

Zhen Ni, Haibo He, Jiny...

Publication

2%

>

2

www.lta.disco.unimib.it

Internet Source

1%

>

3

Tan Szi Hui, Mohamad ...

Publication

1%

>

4

www.tj-source.com

Internet Source

1%

>