

**Artificial Intelligence**

**CZ 3005 2020S1**

**Assignment 2**

**Submitted By: Yeo Kok Leong Desmond**

**Matriculation Number: U1821023C**

School of Computer Science and Engineering

1. **Introduction**

In this assignment, I will be implementing Q-Learning reinforcement learning algorithm for the one grid-world-based environment: Treasure Hunting. The Q-Learning agent will be defined as a class called *Q\_learnAgent.*

1. **Implementation of Q-Learning agent**

The Q-Learning Agents has the follow method:

1. \_\_init\_\_
2. take\_action (state)
3. train (state, action, next\_state, reward)
4. print\_Qtable
   1. Initialization of Parameter according to the assignment requirement
   2. Discount factor γ = 0.99
   3. Learning rate α = 0.5
   4. Exploration rate ε = 0.01
   5. Cube dimension = 4
   6. Action = (forward, backward, left, right, up, down)

Next, We will initialize Q(s,a) for all s **∈** S+ ,a **∈** A+ = 0 for our Q table. The Q table is declared as *self.Q* using nested dictionary with the state ‘zxy’ as outer key and *action\_space* (E.g ‘up’, ‘down’)as inner key and set all values as 0. Q table will be a matrix where we have a row for every state (64) and a column for every action (6) (Fig. 2).

Figure 1 shows the Initialization function of the Q\_learnAgent.

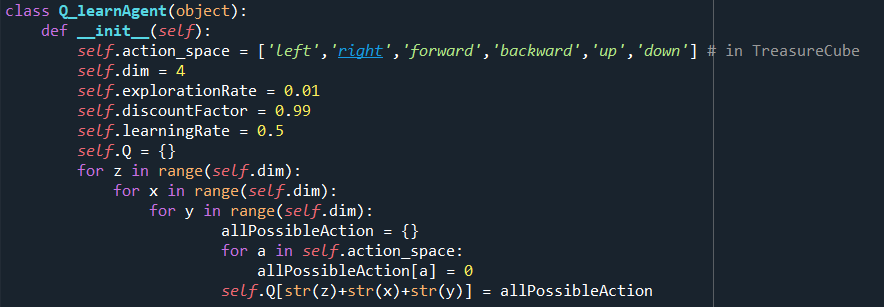


Figure 1 \_init\_ method in Q\_learnAgent

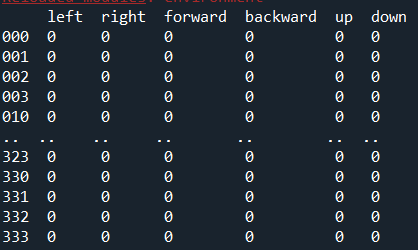


Figure 2 Initialised Q table

* 1. Choosing action to take

In the *take\_action* method the agent will choose an action to take given its current state. The agent will select an action on a ε-greedily approached this is to balance between exploration and exploitation. Where the probability of ε = 0.01 and greedily = 0.99 as stated in the assignment. The goal is to not be greedy by looking for the quick immediate rewards, but instead to optimize for maximum rewards over the whole training.

For **exploitation**(greedy), the agent take the max of Q(s,a). The agent will choose the most optimal action by selecting the highest value in the Q table. However, if there are multiple optimal action the agent will randomly select one of them.

For **exploration**(ε) the agent will randomly select an action from the action space.

Figure 3 shows the implementation of *take\_action* method

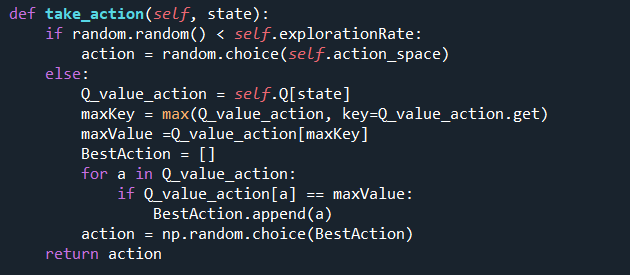


Figure 3 Implemenation of take\_action

* 1. Training the Agent

After the agent has made an action, it will obtain the next state and reward. The agent can use this information to update the Q table.

In the train method, the agent will update the Q table after every step was made. The agent will update the Q table using the following calculation (Fig. 4).

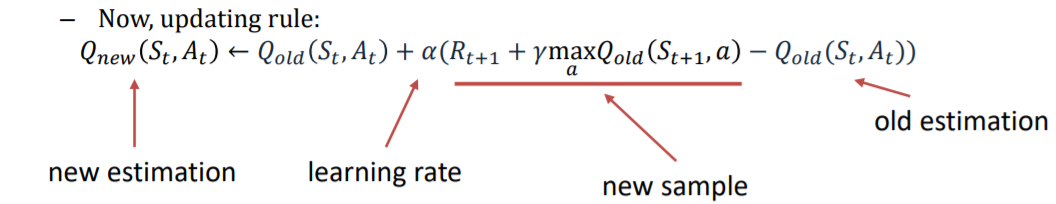


Figure 4 Q table updating rule

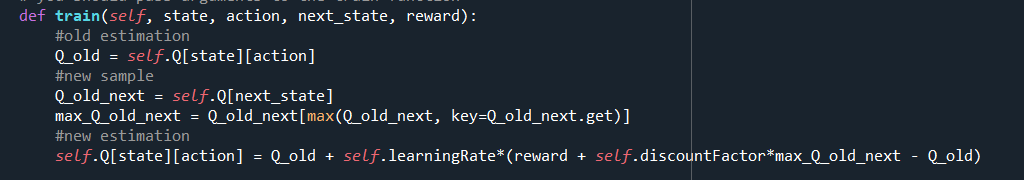


Figure 5 Implementation of Train

1. **Learning progress of the agent**

Figure 6 represent learning progress of the agent where x-axis is the number of episodes and y-axis is the reward gained per episode. The episode rewards gain per episodes increased sharply with just less than 50 < episode. This shows that the agent has quicky learned the most optimal action to take to maximize its reward.

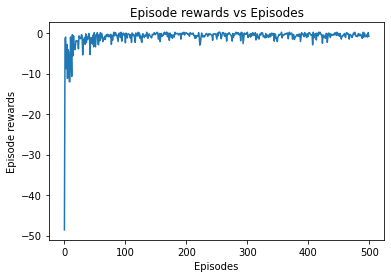


Figure 6 Learning progress graph

1. **Final value table**

Table Final Q Table Value

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| State | Left | Right | Forward | Backward | Up | Down |
| 000 | -0.72106 | -0.40815 | -0.61133 | -0.60297 | -0.71265 | -0.72327 |
| 001 | -0.57880 | -0.40066 | -0.60231 | -0.59414 | -0.59092 | -0.57355 |
| 002 | -0.47516 | -0.30852 | -0.45738 | -0.44914 | -0.48064 | -0.45105 |
| 003 | -0.36588 | -0.42251 | -0.22411 | -0.39307 | -0.36624 | -0.42596 |
| 010 | -0.57968 | -0.20295 | -0.54147 | -0.50278 | -0.53566 | -0.54192 |
| 011 | -0.55084 | -0.31027 | -0.50450 | -0.48890 | -0.54494 | -0.48812 |
| 012 | -0.42879 | -0.43875 | -0.18071 | -0.40142 | -0.39071 | -0.39232 |
| 013 | -0.32724 | -0.04055 | -0.32702 | -0.33285 | -0.36833 | -0.30444 |
| 020 | -0.40941 | -0.00206 | -0.46634 | -0.35639 | -0.40924 | -0.39963 |
| 021 | -0.37394 | -0.36870 | -0.40615 | -0.31609 | -0.04081 | -0.38611 |
| 021 | -0.28679 | -0.29717 | 0.13617 | -0.29627 | -0.26515 | -0.27730 |
| 023 | -0.31021 | -0.27390 | 0.27633 | -0.25898 | -0.27485 | -0.27140 |
| 030 | -0.36239 | -0.35686 | 0.18735 | -0.06491 | -0.37713 | -0.34479 |
| 031 | -0.31306 | -0.29627 | 0.06838 | -0.37090 | -0.31150 | -0.31088 |
| 032 | -0.29015 | -0.29006 | -0.11012 | -0.26555 | -0.28177 | -0.29882 |
| 033 | -0.27113 | -0.28070 | -0.11227 | -0.24751 | -0.24894 | -0.27463 |
| 100 | -0.61024 | -0.49983 | -0.22919 | -0.57425 | -0.60417 | -0.60743 |
| 101 | -0.47101 | -0.46964 | -0.21655 | -0.42732 | -0.47890 | -0.48554 |
| 102 | -0.36598 | -0.40254 | -0.15667 | -0.37837 | -0.36229 | -0.40747 |
| 103 | -0.31741 | -0.19040 | -0.28356 | -0.34725 | -0.34479 | -0.35205 |
| 110 | -0.52863 | -0.07783 | -0.48826 | -0.48716 | -0.49895 | -0.48794 |
| 111 | -0.41812 | -0.02106 | -0.40090 | -0.40725 | -0.37691 | -0.38691 |
| 112 | -0.28814 | 0.09529 | -0.25259 | -0.25994 | -0.24517 | -0.24566 |
| 113 | -0.22759 | 0.10883 | -0.19320 | -0.24280 | -0.26956 | -0.15783 |
| 120 | -0.37086 | 0.10003 | -0.32360 | -0.32076 | -0.33591 | -0.40347 |
| 121 | -0.20151 | 0.18750 | -0.16325 | -0.25127 | -0.14384 | -0.22451 |
| 122 | -0.18137 | -0.14872 | 0.32759 | -0.17375 | -0.18439 | -0.13760 |
| 123 | -0.21194 | -0.18598 | 0.39191 | -0.25112 | -0.22724 | -0.21540 |
| 130 | -0.24238 | -0.22289 | 0.34839 | -0.23355 | -0.15048 | -0.22313 |
| 131 | -0.13481 | -0.16081 | 0.20311 | -0.14925 | -0.14181 | -0.12450 |
| 132 | 0.16699 | -0.09975 | -0.14165 | -0.13675 | -0.09975 | -0.13063 |
| 133 | -0.18446 | -0.14925 | -0.15095 | -0.21195 | -0.17375 | 0.08984 |
| 200 | -0.54231 | -0.54064 | -0.34693 | -0.53237 | -0.02927 | -0.58869 |
| 201 | -0.39458 | -0.42589 | -0.41268 | -0.38622 | 0.05592 | -0.39654 |
| 202 | -0.34479 | -0.09323 | -0.34979 | -0.39124 | -0.35478 | -0.37409 |
| 203 | -0.34784 | -0.33166 | -0.33854 | -0.34030 | -0.36881 | 0.02332 |
| 210 | -0.32345 | -0.33730 | -0.38667 | -0.35755 | 0.05716 | -0.32063 |
| 211 | -0.29003 | -0.27417 | -0.28130 | -0.31236 | 0.23350 | -0.33025 |
| 212 | -0.20491 | -0.25597 | 0.32804 | -0.20968 | -0.25999 | -0.25876 |
| 213 | -0.21063 | 0.50228 | -0.20133 | -0.19437 | -0.09753 | -0.19232 |
| 220 | -0.26601 | -0.24494 | -0.25218 | -0.27867 | 0.11619 | -0.23880 |
| 221 | -0.17513 | 0.33342 | -0.17979 | -0.14282 | -0.15956 | -0.14925 |
| 222 | -0.05000 | -0.09975 | 0.35278 | -0.05000 | -0.05706 | -0.00846 |
| 223 | 0.15025 | -0.07500 | 0.42888 | -0.07475 | 0.01684 | -0.07475 |
| 230 | -0.16769 | -0.19850 | 0.48012 | -0.16098 | 0.07320 | -0.19850 |
| 231 | -0.13341 | 0.32786 | 0.45945 | 0.09074 | -0.11200 | -0.12450 |
| 232 | -0.09975 | -0.09975 | -0.05000 | -0.07500 | 0.79025 | -0.05000 |
| 233 | 0.32874 | -0.05000 | 0.93111 | -0.07475 | 0.00000 | 0.00000 |
| 300 | -0.48569 | -0.50713 | -0.48555 | -0.54307 | 0.01116 | -0.48410 |
| 301 | -0.39205 | 0.28768 | -0.35822 | -0.36822 | -0.15839 | -0.34224 |
| 302 | -0.33279 | 0.36339 | -0.34175 | -0.31295 | -0.30085 | -0.31443 |
| 303 | -0.33862 | -0.30779 | -0.29681 | -0.33894 | -0.32066 | -0.31485 |
| 310 | -0.24943 | 0.10431 | -0.24751 | -0.28267 | -0.22916 | -0.24745 |
| 311 | -0.28936 | 0.07869 | -0.22289 | -0.21987 | -0.23197 | -0.21676 |
| 312 | -0.22751 | 0.39164 | -0.19850 | -0.21848 | -0.01973 | -0.22616 |
| 313 | -0.22612 | 0.66178 | -0.22289 | 0.23983 | -0.24751 | -0.23362 |
| 320 | -0.21682 | 0.34694 | -0.19850 | -0.18922 | -0.19847 | -0.19850 |
| 321 | -0.17962 | -0.12438 | -0.14925 | -0.04204 | 0.49944 | -0.14726 |
| 322 | -0.05000 | -0.05000 | 0.24255 | -0.05000 | 0.67725 | -0.05000 |
| 323 | -0.05000 | 0.94401 | -0.05000 | 0.09706 | -0.07500 | -0.05000 |
| 330 | 0.04926 | -0.09975 | -0.00754 | -0.09975 | 0.58470 | -0.09975 |
| 331 | -0.05000 | 0.06564 | 0.15447 | 0.18630 | 0.80037 | 0.24124 |
| 332 | -0.05000 | 0.38193 | 0.47500 | -0.05000 | 0.98613 | -0.05000 |
| 333 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |