109403034 賴昀廷 Colab 連結

最佳結果

最好的一次結果是花了 253 步走到終點，並且也拿到了 5 個寶藏。

```
[13]: if __name__ == "__main__":
          q_table = rl()

          print("\n")
          print("best result:")
          best = "Episode %s: total_steps=%s score=5 " % (BEST_STEP_EP, BEST_STEP) # 因為在定義BEST_STEP時就有篩選過score要等於五了
          print(best)
          print("\n")

          print("\r\nQ-table:\n")
          print(q_table)
```

```
['Episode 537: total_steps=2357 score=5 ']
['Episode 538: total_steps=527 score=3 ']
['Episode 539: total_steps=1511 score=5 ']
['Episode 540: total_steps=862 score=2 ']
['Episode 541: total_steps=1055 score=5 ']
['Episode 542: total_steps=253 score=5 ']
['Episode 543: total_steps=861 score=4 ']
['Episode 544: total_steps=690 score=4 ']
['Episode 545: total_steps=2769 score=5 ']
['Episode 546: total_steps=1395 score=4 ']
['Episode 547: total_steps=818 score=2 ']
```

Reward 設定

其實一開始只有設定上下左右四個動作的 Reward 而已，撞到牆回饋-5 分是依照原本助教的參考程式碼，就沒有再多做改動。最初設定拿到寶藏的回饋是 10，但不知道是獎勵不夠多的關係，還是那時候的一些超參數還沒設定好，最終跑出來的結果都要幾千幾萬步才能走出迷宮，後來在陸陸續續改良後，發現拿到寶藏的回饋 100 的結果好像不錯，因此後續就沒有再往這方面調整了。

後來，在調整完其他參數並得到不錯的成果後，我又加了下圖的幾行設定，主要是想讓機器走到寶藏附近時也可以有獎勵，而走到寶藏右邊時會得到 5 的回饋，是因為我看地圖中有兩個寶藏需要左轉才可以拿到，因此把他們的 rewards 設高一點。

加了這幾行後，最佳步數從在 280~400 間徘徊下降到 253，因此似乎也算有一點成效。

```
[10]: def get_env_feedback(S, A, path):
          global SCORE
          global STEP_COUNTER
          STEP_COUNTER += 1

          if (S + 1) in treasure_positions: # 走到寶藏左邊
              R = 1
          if (S - N_STATES_x) in treasure_positions: # 走到寶藏下面
              R = 1
          if (S + N_STATES_x) in treasure_positions: # 走到寶藏上面
              R = 1
          if (S - 1) in treasure_positions: # 走到寶藏右邊
              R = 5
```

```
    if A == "right":
        # step_counter += 1
        if S % N_STATES_x == N_STATES_x - 1: # 撞到牆
            S_ = S
            R = -5
        else:
            S_ = S + 1
            if S_ in treasure_positions:
                R = 100  # 踩到寶藏
                SCORE += 1  # 更新寶藏分數

                treasure_positions.remove(S_)  # 移除已經踩過的寶藏位置
            else:
                R = 0 #3  # 未踩到寶藏
    elif A == "left":
        # step_counter += 1
        if S % N_STATES_x == 0: # 撞到牆
            S_ = S
            R = -5
        else:
            S_ = S - 1
            if S_ in treasure_positions:
                R = 100 # 踩到寶藏
                SCORE += 1
                treasure_positions.remove(S_)
            else:
                R = 0
```

```
    elif A == "up":
        # step_counter += 1
        if S < N_STATES_x: # 撞到牆
            S_ = S
            R = -5
        else:
            S_ = S - N_STATES_x
            if S_ in treasure_positions:
                R = 100 # 踩到寶藏
                SCORE += 1
                treasure_positions.remove(S_)
            else:
                R = 0
    elif A == "down":
        # step_counter += 1
        if S == GOAL - 21: #走到終點
            S_ = "terminal"
            R = 10
        elif S >= (N_STATES_x * (N_STATES_y - 1)): # 撞到牆
            S_ = S
            R = -5
        else:
            S_ = S + N_STATES_x
            if S_ in treasure_positions:
                R = 100 # 踩到寶藏
                SCORE += 1
                treasure_positions.remove(S_)
            else:
                R = 0
    return S_, R
```

Q-table

只截了一小部分的圖

(<built-in function all>,	left	right	up	down
0	-4.999588	0.000615	-4.999467	0.000589
1	0.000302	0.001843	-4.999284	0.001983
2	0.000912	0.004065	-4.997045	0.002528
3	0.002262	0.009323	-4.995633	0.006822
4	0.004081	0.055721	-4.944134	0.016348
5	0.014590	30.468184	-4.955014	0.063884
6	0.076158	0.093236	-4.558727	0.995165
7	0.470290	0.017415	-4.976956	0.018064
8	0.093906	0.018745	-4.986747	0.009516
9	0.028856	0.003563	-4.968953	0.014919

心得

寫這次的作業心情起伏很大，第一次跑的時候程式執行了三個小時還看不到結果，放著跑了一整夜才發現它走了幾千萬步才出迷宮，那時真的很崩潰，想說這次作業不知道要做多久。

還好後來有幾次幸運調到不錯的參數，才能漸漸地有修改的方向。

一開始，我 Gama 都設 0.5~0.9 之間，Epsilon 也都是 0.5 以上，但執行出來的步數都幾千幾萬，慢慢調整後才找到比較適合的參數。

[3]:
```
# todo: 修改迷宮大小
N_STATES_x = 21
N_STATES_y = 11
ACTIONS = ["left", "right", "up", "down"]

# 目標位置
GOAL = 230

# 參數設定
EPSILON = 0.3
ALPHA = 0.3
GAMMA = 0.3
MAX_EPISODES = 1000
FRESH_TIME = 0
```

印象最深的就是中間我本來以為我完成作業了，每個執行步數都超級少，甚至出現了 43 步拿到 5 個寶箱的成果，讓我很雀躍，但好險有助教提醒迷宮的最低步數至少會有六七十，回頭一看才發現迷宮牆壁等等的位置被我打錯了，甚至 score 也忘記每次要重置，導致不管怎麼跑都會是 5 分的情況發生。

```
[27]: if __name__ == "__main__":
          q_table = rl()
          print("\r\nQ-table:\n")

          # print("best result: \n")
          # print("Episode %s: total_steps=%s score=%s \n", min_total_steps_episode , min_total_steps, min_total_steps_score)


          print(q_table)
```

```
['Episode 75: total_steps=95 score=5 ']
['Episode 76: total_steps=100 score=5 ']
['Episode 77: total_steps=68 score=5 ']
['Episode 78: total_steps=69 score=5 ']
['Episode 79: total_steps=87 score=5 ']
['Episode 80: total_steps=73 score=5 ']
['Episode 81: total_steps=214 score=5 ']
['Episode 82: total_steps=131 score=5 ']
['Episode 83: total_steps=84 score=5 ']
['Episode 84: total_steps=115 score=5 ']
['Episode 85: total_steps=118 score=5 ']
['Episode 86: total_steps=134 score=5 ']
```

好險後來重新修改後沒多久就得到正確成果了，而且這次執行時間都很快，比起前幾次的作業，這次可以很快地就看到成果然後再微調，因此寫的過程都蠻快樂的。

其他

因為我的 episode 設 1000，有點大，因此我有再加一行 best result 來儲存最好的紀錄，就可以省去從第一筆一直找到最後一筆資料的時間。
會直接把 score=5 寫死是因為在儲存 BEST_STEP 時就有判斷過 score 是否等於 5 了，因此這部分確認過沒問題。

```
[13]: if __name__ == "__main__":
          q_table = rl()

          print("\n")
          print("best result:")
          best = "Episode %s: total_steps=%s score=5 " % (BEST_STEP_EP, BEST_STEP) # 因為在定義BEST_STEP時就有篩選過score要等於五了
          print(best)
          print("\n")

          print("\r\nQ-table:\n")
          print(q_table)
```

```
['Episode 996: total_steps=591 score=2 ']
['Episode 997: total_steps=2046 score=5 ']
['Episode 998: total_steps=1000 score=4 ']
['Episode 999: total_steps=629 score=4 ']
['Episode 1000: total_steps=304 score=2 ']


best result:
Episode 542: total_steps=253 score=5
```