Ouassim Fari – Ouassim.fari@gmail.com

Project 4 Train a Smartcab to Drive

QUESTION: *Observe what you see with the agent's behavior as it takes random actions. Does the* ***smartcab*** *eventually make it to the destination? Are there any other interesting observations to note?*

After changing the action to "random", the smartcab moves. But the motion is erratic and with no particular pattern. The smartcab did reach destination a couple of times but hit the "hard time limit" more often.

```
Environment.step(): Primary agent hit hard time limit (-100)! Trial aborted.
```

QUESTION: *What states have you identified that are appropriate for modeling the* ***smartcab*** *and environment? Why do you believe each of these states to be appropriate for this problem?*

I think the best stats are the inputs (traffic light, incoming car on three direction) and the next waypoints. I initially added the deadline but ended up removing it as it did not prove to be really beneficial. I think the input are very important to describe the state as it also defines the potential moves and thus the potential reward. Without knowing that the light is red, the smartcab wouldn't be able to link the negative or positive reward to a state. I added the next waypoint as it gives also the information of where is located the next point, which also influences the action the smartcab should take.

OPTIONAL: *How many states in total exist for the* ***smartcab*** *in this environment? Does this number seem reasonable given that the goal of Q-Learning is to learn and make informed decisions about each state? Why or why not?*

We have 5 different parameters:
next_waypoint : forward / right / left
light : reg / green
oncoming : right / left / forward / none
right : right / left / forward / none
left : right / left / forward / none

Total number of state is then equal to N=3*2*4*4*4 = 384 states

This number seems to be reasonable as we have 100 trials and for each trial we have between 20 and 55 steps (deadlines). This leads to potentially visit between 15360 and 42240 states. Despites some states probably overlapping each other, we can assume that most of the 768 original states will be visited.

QUESTION: *What changes do you notice in the agent's behavior when compared to the basic driving agent when random actions were always taken? Why is this behavior occurring?*

We can now notice that the agent is way more likely to reach destination after a few trial/iterations. The path followed make more sense. This behavior is occurring because we are now taking the action (forward left right) which tends to maximize the reward.

QUESTION: *Report the different values for the parameters tuned in your basic implementation of Q-Learning. For which set of parameters does the agent perform best? How well does the final driving agent perform?*

I wrote an optimization code to find the question. I iterate through many potential values of alpha and gamma and extract the ones which gives the best success rate.

Values tested for alpha and gamma goes from 0 to 1 with increment of 0.05. (I suggest changing the increment in the code if you would like to test it out. A step of 0.25 with n_trial = 10, runs in less than a minute).
Multiple combination of alpha and gamma can give the best rate, so I then filter one more time on the one that gives the best rewards (which means less penalty and the fastest).

The code runs roughly in 100 minutes on a i5 laptop.

Top 10 results where success rate is 100% sorted from best:

Alpha	Gamma	Reward
45	50	2446
90	70	2398
20	40	2384
15	55	2377
5	90	2350.5
90	60	2325
90	65	2313.5
30	30	2304
5	10	2302.5

The best solution I found is for alpha : 0.45 and gamma: 0.50

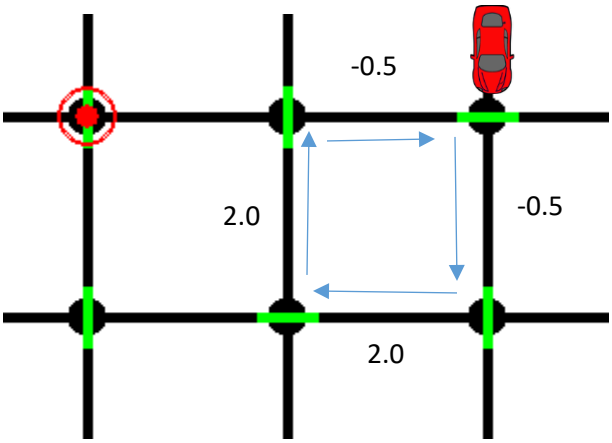
With that configuration, I reached destination more than 99% of the time.

QUESTION: *Does your agent get close to finding an optimal policy, i.e. reach the destination in the minimum possible time, and not incur any penalties? How would you describe an optimal policy for this problem?*

An optimal policy can be defined as the policy which brings the smartcab in the less moves as possible and by respecting traffic rules. As explained in the question above, my agent maximizes the success rate (reaching destination) and maximize the reward. The reward itself is the sum of the penalties and positive rewards which incurs during the travel.

The current state of the reward (+ 2 for legal move in proper direction, -0.5 in legal move in wrong direction, -1 for illegal move, +10 for reaching destination) can create some problems.
Firstly, it doesn't prohibit explicitly illegal moves. If I would have included the deadline in my state, the agent would learn that he wouldn't get the bonus reward (+10) on expiry, and thus would be likely to cross a red traffic light. But since I did not implement deadline in the state, I did not see such behavior.

Another issue, is the low penalty for non-proper moves (moves which are not toward the target). The current penalty is -0.5 while the proper move bonus is +2.0. This can lead to situation where circling can be beneficial for the agent but not respectful toward the optimal policy.



For example, here, the car could go forward instead of right, incurring -0.5, and then right (+2.0 because it's the good direction), then up (+2.0), then right -0.5 and thus finishing a circle with a reward of +3.0.

In order to avoid situations like this, I would suggest to change the reward system and include a new penalty which multiplies the elapsed time by a negative factor. This should prevent such behavior as the smart agent would then rush to the destination. The illegal move penalty should also be increased to avoid "rushing" illegally in that case.