Tasks
Project Report

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?

The agent never makes it to the destination except for rare random casualties and does not consider any traffic rule.

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?

The smartcab could potentially have lots of different states because we might think that every position the smartcab could possibly be might me, multiplied by every possible side the cab could be heading towards, multiplied by every possible traffic condition and traffic light status, multiplied by every action the planner could suggest as possible action. This calculation could easily result on hundreds, maybe thousands of different states. However, I noticed that every position of the grid are essentially equivalent because being in a different position does not affect the behaviour of the agent. Furthermore, the agent doesn't really care about the status of traffic lights and traffic: all it really needs to know is the list of actions allowed by the law.

Another way to confirm this analysis would be to read the reward function and see what it is strictly affecting the reward calculation. The reward have three different values: -1 if the action is not allowed by the law, -0.5 if the action is not the same as the action chosen by the planner, -0 if the action is to remain in the same position, and 12 if the new state is the target. The cab doesn't really need to know where the target is as far as the planner is driving him step by step to the target. However, it does need to know the list of actions allowed by the law and the next action suggested by the planner. These results confirm the previous analysis.

The state must include the minimum amount of data necessary to enable the cab to perform any possible action without getting any penalty. This will limit the number of the states, speed up the learning process, and make the cab behaviour precise enough to provide a very good performance. The cab need to know the oncoming traffic to be able to turn left and the left traffic to be able to turn right while the traffic light is red without getting any penalty. Therefore the status will include the status of the traffic light, the next action from the planner, the status of the oncoming traffic and the traffic from the left.

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?

If the traffic light is red the actions allowed might be none or turn right only. If the light is green the cab can always turn right and go forward, but only when there is not oncoming traffic he will be able to turn left. This brings up to 4 different combinations of values the array might have. The planner can suggest 3 possible actions. The planner is unlikely to suggest to not move. The oncoming and left traffic can have 4 statuses. The total number of statuses is 96. This number, multiplied by all the possible actions, produces up to 384 records in our Q Learning table. This number seems more than reasonable. Training the cab with only 300 trials with an average of 30 moves per trial means that we are training those 384 Q values with approximately 9000 moves in total which is approximately 23 moves per Q value. This means that we will probably need to use a fairly high learning rate in order to train the cab because we must consider that those executions will not be equally distributed among the states.

In the practice I tested few combinations of inputs to consider in the state and my final best result was to consider only the light status, the next move from the planner, the oncoming and the left traffic. The oncoming traffic is needed to be able to turn left without making any penalties, while the traffic from the left is needed to be able to turn right when the light is red.

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?

In my implementation I decided to not use the epsilon and to set the initial q value of every state to a very large number so that the machine learning is forced to visit all of them. I also implemented the training to be in a separate 300 trials with 100 cars, increasing the traffic and the chances to visit all the possible statuses. The agent most of the times it makes it to the target in time and it almost always take the fastest route. However the driving still have a 35% of penalties and only 85% of success rate. I started the machine with an initial learning rate of 0.001 and discount of 0.8. The poor performance is probably due to a discount parameter being too high and a learning rate being too low. In the next phase I am going to tune those parameters to achieve the best performance.

The agent is now very oriented and it makes it to the target chosing the best path suggested by the planner. Penalties are very low because the qTable stores the old rewards and reuse them to choose the best action the second time the cab is in the same status.

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?

We are looking into training the smart cab with a relatively low amount of iterations, which means we should probably increase the learning rate. I imagine a discount rate to be very low, or maybe 0: the cab gets rewards and penalties according to the move it makes at a precise state, but if the final q value will be determined by the reward and the best q value of the next state, q values of penalties will be affected by the best q values of the next state, and in most of the cases there will be an action in the next state which gives a good reward, as there is probably on the other state which is reached without collecting a penalty. The decision of our case really depends on the single state reward and not so much from the next one. On the other hand, we really want the reward of hitting a target to spread out among the other positive moves, so that the cab will tend to make the right choice.

My smartcab is training in the first 300 trials and the last 100 trials are used to test the final results. The statistics reported below have been calculated just on the last 100 trials. The success rate indicates the percentage of trials where the cab was able to hit the target, while the penalties rate indicates the percentage of moves the cab got a reward lower than 0.

alpha	discount	success rate	penalties
0.1	0	100	0.7
0.2	0	100	0.45
0.3	0	100	0.39
0.4	0	100	0.084
0.5	0	100	0
0.6	0	100	0.078
0.7	0	100	0
0.8	0	100	0
0.9	0	100	0
1	0	100	0
0.9	0.2	100	0.15
0.9	0.4	100	0
0.9	0.6	98.98	0.24
0.9	0.8	91.92	27.53

The values that I chose are alpha:0.9 and discount:0.

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?

The best policy, given the input information collected in the status, would be to turn right only when the light is green or there is no traffic coming from the left and going forward. Going forward is also only allowed then the light is green. Turning left is allowed only according to status of the oncoming traffic, and the status of the light of course. Our policy should suggest either the best action suggested by the planner in case this is allowed in the current state, if this is not allowed, the cab should just stop and wait for that action to be allowed by the traffic conditions and the traffic light.

My agent is performing very well, reaching levels of 100% of success rate and 0% of penalties. The policy learnt by the agent is the same of what I described because there couldn’t be another policy that would provide a 0% penalty rate.