P4 - Training a smartcab with Reinforcement Learning.

Julio Rodrigues (juliocezar.rodrigues@gmail.com)

September 2016

1 Implement a Basic Driving Agent

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 sometimes does reach the destination by chance, but most of the time it drives around aimlessly and exceeds the deadline.

2 Inform the Driving Agent

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?

Since the primary task here is to learn how to reach the destination without causing any accidents, the only sensor information used for this task are the light status (green or red), left/right/oncoming traffic and the next waypoint to be reached. At first, additionally to those, I also tried to use the current location, the final destination, and the deadline. This set of states increased the state space considerably and made learning very slow and inefficient. I also tried removing the deadline information from the status, but it did not improve the learning rate.

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?

In the setting described above, there are 384 possible states. In practice, however, even after running the simulation hundreds of times, only a handful different states are seen and learning still converges.

3 Implement a Q-Learning Driving Agent

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?

Now, only after a few iterations, the agent learns to reach the destination consistently.

4 Improve the Q-Learning Driving Agent

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?

The initial alpha and epsilon values that performed the best in my grid search were an alpha value of 0.4 and an epsilon value of 0.4. Using this set of parameters, the success rate after 1000 trials is 0.99.

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?

Yes, it does. The success rate of the agent consistently reaches 98% or more.

An optimal policy would be one which accurately follows the traffic rules and doesn't cause the agent to remain inactive when it doesn't have to.