Machine Learning: Assignment 4: Reinforcement Learning

Vivek Khetan
UT-EID: vkk287

1. Introduction:

The goal of this assignment is to use reinforcement learning to allow a clumsy car to how to navigate on the road with obstacle in it. The interesting (tough) part is to make it learn to behave in presence of some parked cars, moving cars, pedestrian as well as to follow traffic lights. The other side of the road is not the terminating condition, i.e. once the car will reach its end it will restart from the starting point, as if it's a circular road.
Another interesting part was to make the environment so that the driving environment consists of a four-lane road with two lanes in each direction. Our environment is supposed to have:

1. It should be circular so that the cars going over the end of the road appear at the beginning of the other end.
2. Traffic lights with a yellow-red-green pattern
3. Cars traveling in both directions at different speeds and lane changing ability
4. Parking (Parked Cars)
5. Pedestrians who cross the road at random points and times

Q-learning, like virtually all RL methods, is one type of algorithm used to calculate state-action values. It falls under the class of temporal difference (TD) algorithms, which suggests that time differences between actions taken and rewards received are involved.

For SARSA (on-policy, sampling, non-optimal, and behavioral policy), given old Q value Q (st, at), learning rate α, discount factor γ, reward rt, greedy algorithm random probability ε, Q value of next state action pair Q (st+1, at+1), updated Q value of current state action pair can be calculated as :

Q (st, at) := Q (st, at) + α (rt + γ . (Q (st+1, at+1)) - Q (st, at)) with probability ε
Q (st, at) := Q (st, at) + α (rt + γ . maximum (Q (st+1, at+1)) - Q (st, at)) with probability 1 – ε

I used Q (Qmax) learning for this assignment problem. The Q learning algorithm can be understood as follow:

```
initialize Q[numstates,numactions] arbitrarily
observe initial state s
repeat
   select and carry out an action a
   observe reward r and new state s'
   Q[s,a] = Q[s,a] + α(r + γmaxa' Q[s',a'] - Q[s,a])
   s = s'
until terminated
```

α in the algorithm is a learning rate that controls how much of the difference between previous Q-value and newly proposed Q-value is taken into account

2. Method:

I decided to break the problem in two different parts: first, to make the environment working with predefined random actions for the agent and second, then implement the q-learning algorithm and make the agent (smart-car) take actions according to Q learning. Obviously, at start all the Q values will be zero and smart cars actions will be almost random. But, with each step the algorithm will change the Q value according to the given reward and will slowly start to take better actions.

I started to make the world in jupyter notebook using numpy but that was unsuccessful attempt. Then, I learned about 'tkinker' library in python and used it to make the world. It was a very tedious but hectic world. I started with just the road and then I added parked cars, moving cars as well as pedestrian. I had more difficulty with making pedestrian appears randomly, make the smart car restart with the end of the road and then the toughest work was to make traffic light work. But, after a lot of reading about use of tkinter library and a lot of tweaking, I ended up with a very beautiful GUI with all the assignment requirements.

For the Q learning implementation, I started with an empty q table (all zero), eplison = 0.1, alph = 0.2 and gamma = 0.9. Although the change in these values has affect in training time for my car but I didn't want to get much in these complications. I also decided reward just based on my believes; I gave +150 for reaching the goal, i.e., for reacing the other side, -3- for hitting the pedestrian, -2- for hitting the dumb car (the other moving cars), and -20 for hitting the parked cars. Then, deciding the reward function was a little confusing for traffic lights but in the end, I gave +1 for green light, -5 for crossing the yellow light and -20 for crossing the red light. I also gave +1 reward to taking any step without hitting anything. After implementing my environment and q learning, I changed the action of my smart car form random to the decision taking by the Q learning module.

Results and Discussions

Training the car was the most interesting part of the assignment. Earlier the car was taking a lot of time to make the next move and having a lot of collision, but slowly its speed increased and the collision decreased with time. I trained the car for several hours with all the other modules in my environment and after around ten hours of training, it was working surprising well.

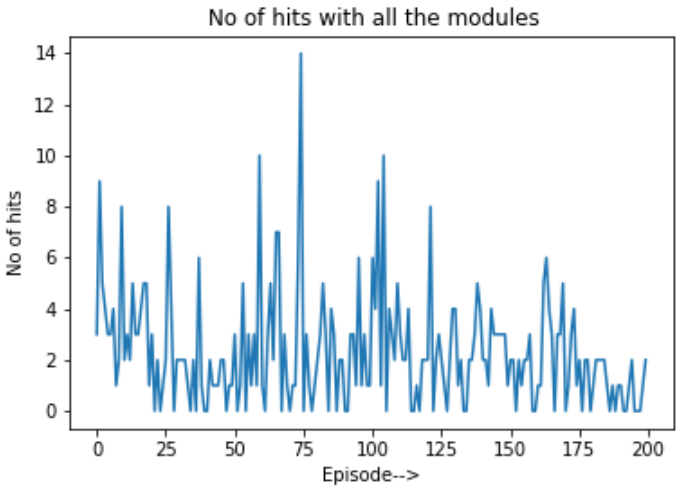
Below is the link of the park, with just two random pedestrians in the environment:
Car and Pedestrians

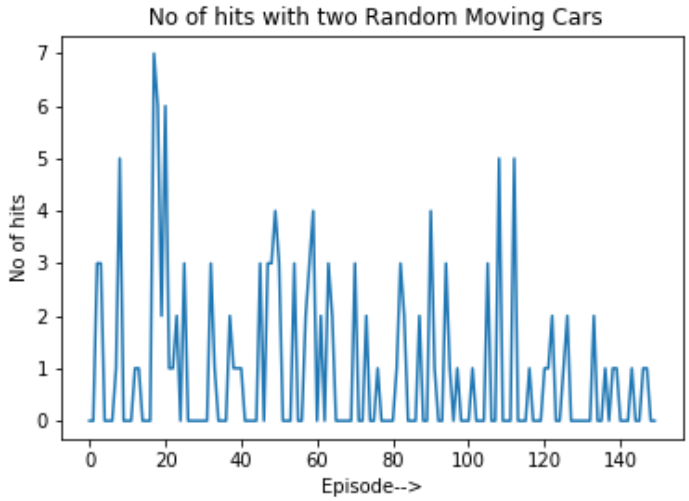
Following is the link to the video of the car after ten hours of learning with every module active in the environment:
Smart Car

I also tested the car on individual modules separately, for different modules the car was behaving differently. Or, better to say, the car learnt differently about the differently modules. When I had only the parked cars in my environment, my smart car learnt very fast and was having a collision seldom but with the moving car, it took more time (episode) to learn.

It was a wonderful experience to see the car stopping for pedestrians or moving cars to pass even without explicitly being programmed for it.

I have plotted a line diagram for no of collision with each passing episode for various modules as well as all the modules in the environment:





Conclusion and Future Work:

This project has given me an exposure to GUI modeling as well as implementation of Q learning for it. It was a wonderful experience how I worked on building the environment from scratch and then implemented Q learning in it. I believe, this assignment has given me enough exposure to the area of reinforcement learning that now I can go and explore the other advance implementation of it. Be it infinite state Q learning or Deep Reinforcement learning. I look forward to take more courses related to reinforcement learning and to do more implementation like this.

Although, it is said that the Artificial Intelligence is the area of machine learning that we haven't figured out yet, but I think Reinforcement learning is the way in that direction. All of Google's DeepMind Project are much sophisticated use of Reinforcement learning and a machine trained on Reinforcement learning have just snatched the alpha-go world championship from the human. I think it is just the start in this area.