Chapter – 2 (implementation – Part 4 – Testing)

**Artificial Intelligence**

**AI-app (Deep-Q): Self Driving Car** (part 4)

4 level AI app test

**Application/test**

**2.20 App-Test (level-1)**

We're going to test our AI-car on *four different levels*. i.e. we're going to play a game, the game will have *four levels of difficulty* and the AI will have to pass these four levels.

* Level 1: First level is going to be to reach the airport and then do some round trips between the ***airport*** and the ***downtown***. So as soon as we see the car do these round trips it'll pass level 1.
* Level 2: Level 2 will be to *still do these round trips*, but on the specific road that we draw ourselves. It'll going to be an easy road because it's level 2 difficulty.
* Of course the car will have to self-drive by staying on that road. The car will go from the *airport* to *downtown* and *back again*. The car will have to do these *round trips* by *sitting* on that road. If it does, we will pass level 2.
* Level 3: In this level, we'll draw some obstacles on the map to see if the car manages to ***avoid the obstacles*** and still reaching its goal.
* Level 4: It is the most challenging level for the car. We'll draw a *very difficult road* to reach the downtown. It will be a road like some zigzag.

|  |  |
| --- | --- |
| * Map only: At first we're going to look at the map. We're going to look at the self-driving car without the AI-Brain. * To deactivate the AI, we need to set the 'Temperature = 0' in our **ai.py** file. It will just be a car having those random actions at the beginning of this model. * Right now 'Temperature = 7'. That's a low temperature. We will increase that afterward. * But if we don't want the car to have a brain *(i.e. to deactivate the AI)* we simply need to set 'Temperature, T = 0'. * Then we save the **ai.py** file to update the change. * Next we execute the **map.ai**. You will see the car is having totally random actions: it goes left, straight, right. It is not reaching the airport- the upper left corner of the map (sometimes it does by random). |  |

Also it is not reaching the other goal which is downtown at the bottom right corner of the map. We can clearly see that the actions are totally random. It is going here and there is definitely no artificial intelligence.

|  |  |
| --- | --- |
| * Activate AI-brain: We will activate our AI-car's-Brain now. First close the ***map.py*** and then restart the Kernel of Spider. * To activate the AI we need to raise the temperature. We set 'Temperature, T = 7' as we had before. We save the ai.py to update the change. Then we execute the map.py gain. * The car gets punished if it goes further from the goal. Notice that the car tries to move diagonally from upper left corner (airport) to lower right corner (downtown). * Increase the Temperature: Notice the car moves like a bug (i.e. it has no plain movement). |  |

* Recall, the Temperature is the parameter in the softmax function that we can increase, so that the ***action*** is returned with more ***certainty***.
* Increased Temperature makes the AI to be more sure of *which action it should play*. AI will remember that *action* because: the action will be played with a ***higher probability***.
* Issue: The only ***problem*** with this ***increasing the temperature*** is that the AI is less exploring the other actions because: by increasing the ***temperature***, other actions we'll have low probabilities.
* T = 100: So we close the map, restart the kernel, set 'Temperature, T = 100' in ai.py and save the change. Restart the map.py.
* Notice the car moves very smoothly.
* You can see that it is moving more/less on a straight-line diagonally. It is not doing these *quick left and right movements* (like an insect).
* That's because now the car is more sure of which direction to take at each time. It wants to take the ***best direction*** going to the ***airport*** and then to ***downtown***.

So clearly we can now say that we passed level 1. The car is doing these round trips between the airport and the downtown.

|  |  |
| --- | --- |
| * To save that brain: just click on this save button. |  |

|  |  |
| --- | --- |
| * Curve of the reward: Notice in the spider kernel, we have the "curve of the reward". * At the beginning we can observe some mistakes that it made, that's why the reward is negative. * But then it *learned* from its *mistakes* and the *reward increased* little by little until reaching a ***constant positive reward*** equals to **0.1** (that's the maximum reward we set). And that's because the **car** *ended up exploring*. * That's the expiration phase. And then it just knew what it had to do. * That's where it was doing the round-trips between the airport and the downtown without any mistake. |  |

**2.21 App-Test (level-2)**

Let's try to pass level-2 where the car will perform some round trips on a specific road. We're going to draw the road ourselves.

* Since we've saved the brain, it won't be a problem if we quit the app.
* To restart everything, we quit the application and start again. We then go back to the app and load the last brain (it is the last weights of the neural networks).
* It's more like the last synapses of the brain that was saved. So it will already know how to do the round trips between the airport and the downtown.
* In the level 2, the car will still do the round trips. But the challenge is: it need to stay on a specific road that we will draw ourselves.

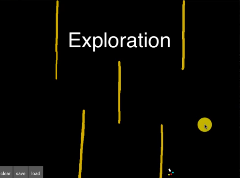
|  |  |  |
| --- | --- | --- |
| * Let's make this road simple because the most complicated road will be in level 4. * First we'll execute the **map.py**. * We'll see some ***random moves*** of the car until we ***load the brain***. * Once the brain is loaded, the car will start to take the round trips. * Next we draw the road, we'll observe some mistakes done by the car when it cross the *sand-barrier*. It'll of course learn from those mistakes. | |  |
| * To get the learning-curve, we *save* the *brain*. Then we can see the *mistakes* done by the car while it *crossing* the *sand*. * At the *left* of the *curve* we can see a *big fluctuation*, that’s the *–ve reward* from the **Level-1**. Now, at the *right* we can see *another fluctuation*, it's from current level, **Level-2**. * Notice, this time it won't take much time for the training. * However, if it does any mistake, we can observe it from the learning curve. | Fig 1 : level 2, learning curve Fig 2 : level 2, observing mistakes | |

It's doing a great job right now. So we will take it to level 3. In the Level-3 we'll draw some difficult obstacles; the car will have to find its way to the downtown and the airport by avoiding the obstacles.

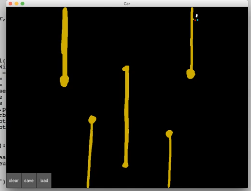
**2.22 App-Test (level-3)**

In this Level-3 the car is still do the round trips between the airport and the downtown. The new challenge is it has to avoid obstacles on the road.

* We'll draw the obstacles but we'll not try to make it too complicated. Because we'll add maximum difficulty at level 4 but let's keep this challenging.
* Load: So as usual we *execute* ***map.py*** and then we load the car's brain. Notice the car is doing the trips properly.
* Obstacles: We draw some vertical lines as obstacles. Sometimes the car won't recognize the vertical lines.
* To force the car to recognize the lines we make the line thicker. This will increase the penalty.
* Also we add circles at the tip of each vertical line to recognize the ends.
* Exploration: The car will understand the obstacles more because for thicker lines and round tips it gets more penalties (i.e. hard punishment with a worse reword).



* We can make the middle line much longer if we want.



* How it find best path: It'll go around the obstacle, but is not trying to find the best path. But anyway, the challenge is to reach the two goals the downtown and the airports. Later we can add some code to find the best path.
* But for now we definitely have a self driving car.

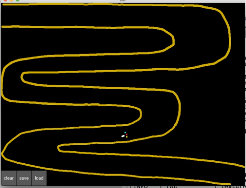
|  |  |
| --- | --- |
| * Score function at level 3: Now let's see the score function. To get the score-function plot, we save the brain. * At this score function, we can see the car gets the punishment (there is some *oscillations* also) and in this Level 3 the score is decreasing with time. |  |
| * Now give it a more time to explore, and we can see the car is trying to ***avoid*** the ***tips*** and the ***obstacles***. If we see the score-graph again, we can observe the improvements. * The car is actually getting better with time.   So we can say that it passed Level 3, because anyway it manages to do the round trips between the airports and downtown by *going* *around obstacles*. And that's what we wanted. |  |

* Let's move on to Level 4, it'll be very challenging. Maybe our car can't pass the Level-4. That's we have to figure out next.
* It'll be a homework challenge. We have to change something in the code to pass Level 4.
* It could be the reword that need to change
* Or something that in the strategy of the game or
* Or even something with the architecture of the neural network
* Or something with the DQN algorithm.
* Look in the internet, do some research and try to get a better AI-Car that pass Level-4 challenge.
* That avoids any kind of obstacle that will
* Never go into sand (border or obstacles)
* Find the best path & shortest path.

**2.23 App-Test (level-4)**

In Level-4, the goal will be to beat the self-driving car because the *Homework challenge* is to modify the code/brain/strategy to improve the AI-Car.

* So in this level we're going to make a ***highly complex road*** that will still go from the airport to the downtown but too complicated for the car to find its way.
* Execute ***map.py*** and then load the brain. You'll see the car will do the round trips easily after loading the brain.
* New road: You can see the car seems to have a lot of trouble after drawing the following road.



* It doesn't find its way to the downtown.
* How to improve the AI-car: We have to change something in the code to pass Level 4.
* Do need to change the strategy? Or
* Do we need to change the parameters of the ***Neural Networks*** to change the *synapses (weights)* in the *brain* of the *car*?
* Do we need to do something with the Rewards? We can give a worse reward when it's not finding the goal (stuck somewhere on the road).
* Change the reward strategy: We can see the ***reward*** is *not very severe right now* when it's not finding the goal.
* It's not getting closer to the goal because the reward is just **-0.2**. That's *not a severe or bad reward*.
* You could try to decrease the reward by setting a reward equal to **-0.5**. Maybe that will work. So it's just a suggestion to help you.
* Neural Network modification: We also can make some changes to our Neural Network.
* Currently we have the architecture of **30** hidden neurons and one hidden layer. We can change the architecture by adding some **more layers** or some **more hidden neurons**.
* Modify Deep Learning Algorithm: We can also try to change something in our Deep Learning Algorithm or we can try out other algorithms.

So try it yourself and try to draw that same road because it's a pretty exciting enigma.

* Hints: I give you a hint or if you want me to explain quickly what the problem is:
* Notice, when the car reaches ***some point*** in the ***middle horizontal road***, it moves back and forth. *It's doing it because: when it's reaching that point it's getting too far away from the goal (downtown) than it comes back.* But the car need to pass that point to reach the goal.
* So you have to change something in the code: In the strategy to (maybe) punish it less when it's getting further away from the goal. Maybe that's the solution.

Before solving this Homework, try to implement the exact same Ai. That would be excellent practice. There is a huge difference between reading to something and try to do the thing yourself.