International Journal of Information Sciences and Application (IJISA). ISSN 0974-2255, Vol.11, No.1, 2019, (Special Issue)

Self-driving Car Using Soft Computing

Praval Kumar, Shivam Shandilya, Tushar Sachan and Mr. Nizam Uddin Khan

Department of Information Technology, IMS Engineering College,
Adhyatmik Nagar, Ghaziabad, Uttar Pradesh 201009, India.

Abstract

AI technologies power self-driving car systems. This Virtual self-driving car uses vast amounts of data from systems, along with machine learning and neural networks, to build systems that can drive autonomously. Autonomous vehicles are being viewed in their ability to improve safety and the driving experience. This paper focus on autonomous car, this car intelligence incorporating, among others, an algorithmic "brain" and powerful sensors "senses" using Deep Q-learning. The paper will review the car virtual assistants underlining their Actions and knowledge in the connected and automated driving ecosystem.

Keywords: *Reinforcement learning, self-driving car,*
Deep Q-Learning, Temporal Difference Learning, Actions.

I. INTRODUCTION

In this, the Learning Model will learn by experimentation. The agent in an environment with specific states has a set of actions that it can perform and after performing those actions, it receives a reward which tells it how good that action was and stores these correct action in "Brain". We have to find when this car receives positive reward or negative reward using graph.

II. BACKGROUND

A. Setting up the Environment

Initially had to create the virtual environment for this car to perform moves in. In this python package called Kivy is used to create UI. Kivy is an accelerated framework for the creation of new user interfaces.

B. Reinforcement Learning

Reinforcement Learning (RL), specifically Deep Q-Learning to make this self-driving car. RL is about performing corrective actions to maximize the reward in a particular situation and all Reinforcement Learning algorithms have three key elements, State, Actions, and Rewards.

C. Deep Q-Learning

Deep-Q Learning we can program AI agents which can operate in environments with discrete actions spaces. A discrete action space refers to actions that are well-defined, e.g. moving left, right, up or down.

D. States

The state the car is in consists of five variables:

Sensor Red, Sensor Yellow, Sensor Blue, Orientation, -Orientation. The first 3 sensors on the front of the car.

These sensors are used to detect if there was a wall of sand in three directions left, right and front of the car. This allows for the car to determine where sand is, and therefore which direction to travel and last two variables are used to represent the orientations of the car. The (-ve) of the orientation is also added to improve the performance while tuning.

E. Actions

There are three possible actions: Turn to 20 degrees clockwise, Turn to 20 degrees Anti-clockwise, and do not turn.

F. Temporal Difference Learning

Deep Q-Learning is used to solve the action-value function Q(s, a). If the AI agent knows Q(s, a) can be quality actions. The reason is knowledge of Q(s, a) would enable the agent to determine the quality of any possible action in any given state. Thus, the agent could behave accordingly. Temporal Difference (TD) learning algorithm used to solve Q(s, a) iteratively.

III. CHOOSING THE MODEL

A. Deep Q- Learning

We finally make usage of deep learning. Look on the update rule for Q(s, a) whereas 's' is a state and 'a' is an action. It can recognize that this will not get any updates if the TD-Target and Q(s, a) have the same values. In this case, Q(s, a), converged the true action-values and the goal is achieved.

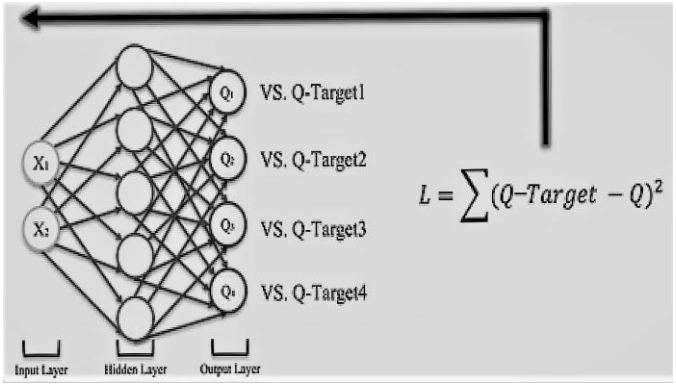

Fig. 1. Deep Q-Network.

This means the objective is to minimize the distance between the TD-Target and Q(s, a). This is the architecture of a neural network, this neural network will take Input Layer, Hidden Layer, Output Layer, Activation Functions, Optimizer.

Input Layer: Five Nodes (One for each state input)

1) Hidden Layer: 30 Nodes
2) Output Layer: 3 Nodes (One for each action)
   - Activation Functions: ReLU
   - Optimizer: Adam Optimizer

One hidden layer is enough for simpler problems. Other would take longer to train and would not result in significant performance improvements.

B. Experience replay

- Experience replay is one of the key technology behind many advances in deep reinforcement learning. Which allows the agent to learn from earlier memories then speed up learning and break undesirable temporal correlations.

- To perform experience replay this will store the agent's experiences. That means instead of using Q-learning on state/action pairs as they occur during simulation, the system stores the data discovered for [state, action, reward, next_state].
- More efficient use of previous experiences, by learning with multiple times. This is key when gaining real-world experience is costly, and the Q-learning updates are incremental and don't converge quickly.

IV. METHODOLOGY

In this, firstly we have to create an environment and Brain of the car using Deep Q-Networks. This Brain will be Trained each time the program is executed. This training will occur while the car is performing some actions in the environment.

We have added buttons to save, and load previous models into the car brain. This car will move from source (Top left corner) to the destination (Bottom right corner).

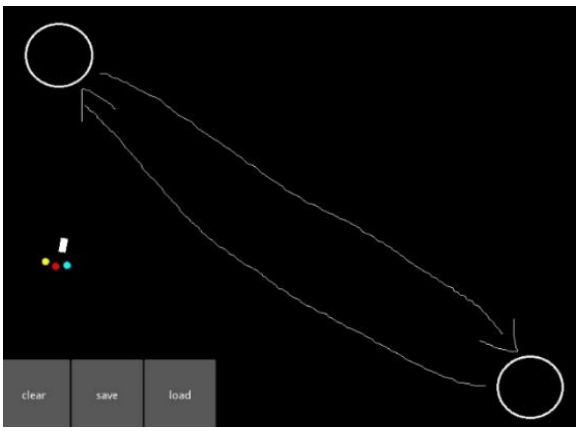

Fig. 2. Source and destination points.

Learning Model will learn with the help of sensors, car sensors will sense any obstacles on the road. While a car is moving on the road it will get +0.1 reward when a car got closer to the objective, -0.1 when a car got farther away from the objective and -5 if the car drives into the sand. Whenever it senses something on the road, the autonomous car will take another way to reach its destination.

Correct moves can be decided with the help of temporal difference and Markov decision process (MDP) which is used to provide a mathematical framework for modeling decision-making.

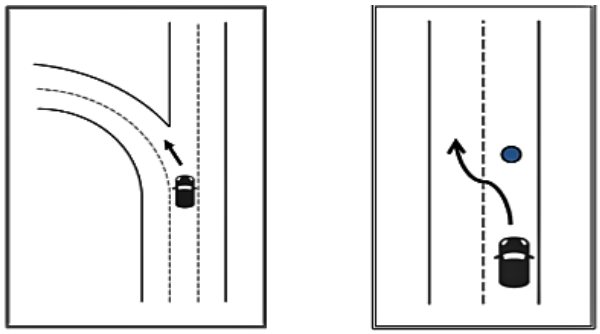

Fig. 3. Obstacle detection using sensors.

If there is an obstacle in between source and destination then the car will take another way to reach its goal and also reward will store into the brain of a car.

$$\text{Temporal difference} = \Sigma(Q\_target - Q)^2$$

This Equation will determine the quality of actions and output of this equation will be treated as input to the neural network in order to improve the quality of the action. Also, these actions will be stored in the brain of the car so that it makes easy to take a correct decision in the future.

V. RESULT ANALYSIS

Table. 1

Rewards	No. of Iteration
-0.10	0-100
0.00	100-1500
+0.10	1500-1600
+0.05	1600-1800
+0.03	1800-2000
+0.07	3000
+0.06	3000-3500
-0.10	3500-4000
0.06	5000-5500

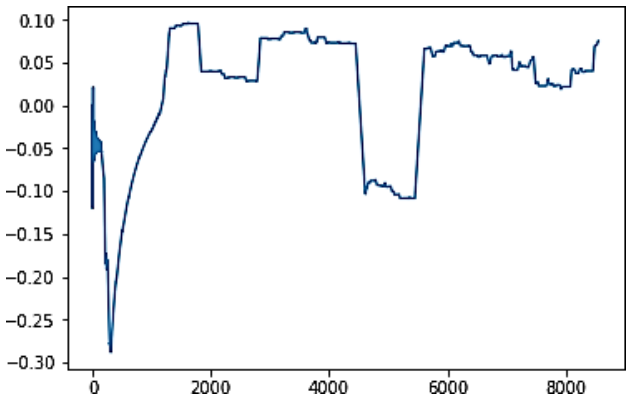

Fig.4.Reward showing graph between +0.1 and -0.3.

The graph was generated earlier. When the agent was given a +0.1 and -0.1 reward for getting closer

and farther away from the goal respectively. The training plateaued after 6000-time steps.

Table. 2

Rewards	No. of Iteration
0.00	0
-1.00	0-300
+0.02	300-500
0.00	500-600
+0.01	600-650
+0.75	650-1100

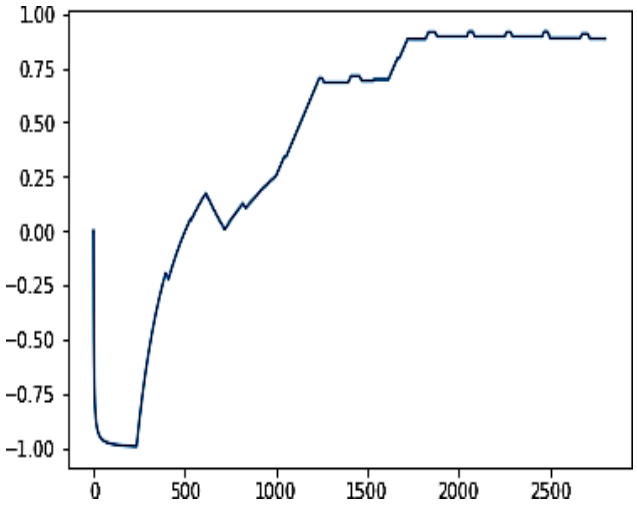

Fig. 5. Reward showing graph between +1 and -1.

Graph of reward received by Agent "Car" by time step. In this graph, the agent was given a +1 reward every time it got closer to its goal, and a -1 reward for the opposite one. The training plateaued after 1500-time steps.

VI. CONCLUSION

As we have observed that it made some mistakes at the beginning but then it learned from its mistakes and the reward increased little by little until reaching a constant positive reward equals to open one but that's the maximum reward we set and that's because it ended up exploring after that it just knew what it had to do also the results shows that the +1 and -1 rewards lead to an agent that trained faster than +0.1 and -0.1 rewards.

VII. FUTURE SCOPE

If we create one more neural network and a number of hidden layers then the accuracy of this car will increase. This work presents the methodology behind the Self Driving Car which aims to define and inspire the next generation of Autonomous driving. We leverage the power of Neural Network and Reinforcement learning to automatically Learn from various learning techniques with exploration and exploitation tradeoff.

REFERENCES

[1] M. M. Rahman, S. M. H. Rashid, and M. M. Hossain, "Implementation of Q learning and deep Q network for controlling a self balancing robot model," 2018.

[2] M. J. A. M. N. Saquib and C. (Dr, "O.P Malik " Self Driving Car System Using AI (Artificial Intelligence)"," *Asian Journal of Applied Science and Technology (AJAST), Volume*, vol. 1, no. 6, pp. 85–89, 7 2017.

[3] V. V. Dixit, S. Chand, and D. J. Nair, "Autonomous vehicles: disengagements, accidents and reaction times,," *PLoS one*, vol. 11, no. 12, p. 0168054, 2016.

[4] G. M.C, "Liability rules and self-driving cars: The evolution of tort law in the light of new technologies." NAPOLI: ES, 2019, pp. 978–88.

[5] K. Iyengar, J. Tarakhovsky, and A. Kosinski, "Developing Autonomous Street-Legal Vehicles: Analysis of 2017 Intelligent Ground Vehicle Competition (IGVC) Self-Drive/Auto-Nav Competition Vehicle Design and Implementation," in *of 2017 IGVC*, August 2018.

[6] A. Wilson, A. Fern, S. Ray, and P. Tadepalli, "Multi-Task Reinforcement Learning:A Hierarchical Bayesian Approach," 2007.

[7] B. Behr, "Learning from Renningen: On the Perceptual Architectures of Self-driving Cars. In: Research Fellowship Lecture, 26," 2018.