SELF-DRIVING CAR SIMULATION

Ashutosh Dubey  
Computer Science & VIT  
ashuvit1998@gmail.com

Prabakaran N  
Computer Science & VIT  
dhoni.praba@gmail.com

Manuscript History  
Number: IRJCS/RS/Vol.07/Issue05/MYCS10083  
Received: 06, May 2020  
Final Correction: 12, May 2020  
Final Accepted: 20, May 2020  
Published: May 2020  
DOI: https://doi.org/10.26562/irjcs.2020.v0705.002  
Citation:

Editor: Dr.A.Arun L.S, Chief Editor, IRJCS, AM Publications, India
Copyright: © 2020 This is an open access article distributed under the terms of the Creative Commons Attribution License, Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Abstract: For the recent years, there has been a flood of enthusiasm for self-driving vehicles. This is because of forward leaps in the field of deep learning where deep neural networks are trained to perform tasks that usually require human intervention. CNN apply models to distinguish examples and highlights in pictures, making them helpful in the field of Computer Vision. Instances of these are object detection, image classification, image captioning, etc. In this project, we have prepared a CNN utilizing pictures captured by a simulated vehicle so as to drive the vehicle self-sufficiently. The CNN learns unique and distinct features from the images and generates steering predictions permitting the vehicle to drive without a human. For testing purposes and preparing the dataset the Unity based simulator provided by Udacity was used.

Keywords: CNN, Deep Learning, Computer Vision, autonomous driving. Behavioural cloning

I. INTRODUCTION

As of late, autonomous driving algorithms utilizing minimal cost vehicle-mounted cameras have attracted increasing research endeavours from academia as well as industry. Different levels of automation have been characterized in self-governing driving. There’s no computerization in level 0. A human driver controls the vehicle. Level 1 and 2 are advanced driver assistance systems. Level 3 vehicles are autonomous, however, a human driver is still needed to monitor and intervene whenever necessary. Level 4 vehicles are completely autonomous yet the mechanization is restricted to the operational structure space of the vehicle for example it doesn’t cover each driving situation. Level 5 vehicles are required to be completely self-ruling and their presentation ought to be human driver. We are very far from achieving level 5 autonomous vehicles in the near future. However, level 3/4 autonomous vehicles are potentially becoming a reality in the near future. Primary reasons for drastic technical achievements in these fields are technical breakthroughs and excellent research being done in the field of computer vision and machine learning and also the low-cost vehicle mounted cameras which can either independently provide actionable information or complement other sensors. Many vision-based drivers assist features have been widely supported in modern vehicles. Some of these features include pedestrian/bicycle detection, collision avoidance by estimating the front car distance, lane departure warning, etc. However, in this project, we target autonomous steering, which is a relatively unexplored task in the field of computer vision and machine learning

II. RELATED WORK

The DAVE framework was made by DARPA [1] and utilized pictures from two cameras just as left and right steering commands to train a model to drive. It exhibits that the procedure of end-to-end learning can be applied to self-sufficient driving. This implies the middle of the road highlights, for example, the stop signs and path markings don’t need to be explained or named for the framework to learn. The engineering of this model was a CNN composed of fully connected layers that originated from systems recently utilized in object recognition. The ALVINN framework [2] is a 3-layer back-propagation system worked by a group at CMU to finish the undertaking of path following.
It trains on pictures from a camera and a distance measure from a laser range discoverer to yield the direction the vehicle should move. ALVINN’s model uses a single hidden layer back-propagation network. We recreated an examination by NVIDIA [3]. The framework utilizes an end-to-end approach where the data is first gathered in numerous distinctive environmental conditions. The information is then augmented to make the framework robust to driving off center and to various potential situations. The system design is a total of 9 layers beginning with the input number of filters and sizes and a dropout after that to handle overfitting. In the end, 3 dense layers were added followed by the output layer. Adam optimizer was used for parameter optimization with a fixed learning rate of 0.04. My model has batch size of 100 with number of epochs as 10 having steps per epoch as 300. In addition to that, batches are generated on the fly so as to optimize the memory utilization with the help of batch generator. For performance evaluation during training, mean squared error was used as a loss function to keep a track of the performance of the model.

\[ \text{MSE} = \frac{1}{n} \sum (y - y')^2 \]

A blend of standard vector-based Long Short-Term Memory (LSTM) [4] and convolutional LSTM at various layers of the proposed deep network. Sequential layers for the most part have a comparative visual appearance, but subtle per pixel movements can be seen when the optical stream is figured. Traditional picture convolutions, as those embraced by cutting edge picture order models, can move along both spatial measurements in a picture, which suggests that they are basically 2-D.

A Convolutional Long Short-Term Memory Recurrent Neural Networks (C-LSTM) [5] can fundamentally improve end-to-end learning execution in self-sufficient vehicle steering dependent on camera pictures. Inspired by the amplitudes of CNN in visual component extraction and the proficiency of Long Short-Term Memory (LSTM) Recurrent Neural Networks in managing long-go transient conditions our methodology permits to display dynamic temporal dependencies with regards to steering angle estimation dependent on camera input.

III. DATASET COLLECTION

We have utilized Udacity’s self-driving vehicle simulator system for gathering the data. This simulator system is built in Unity and was utilized by Udacity for the Self-Driving Nanodegree Program But was recently Open Source. It repeats what NVIDIA did in the simulation. We can gather every one of our information from the simulator system. Utilizing our keyboard to drive the vehicle, we had the option to train the simulated vehicle to turn left, right, accelerate and delay down. Another significant angle is that this test system can be utilized for preparing just as testing the model. Henceforth, it has two modes: (I) Training mode, and (ii) Autonomous mode.

The training mode is utilized to gather the information and the self-sufficient mode is utilized to test the model. Additionally, there are two kinds of tracks in the test system - the lake track and the jungle track. The lake track is generally littler and simpler to deal with the vehicle when contrasted and the jungle track as demonstrated. The simulator system captures information when the vehicle is driven around the track left and right keys to control the steering angles and up and down arrows to control the speed. From this, the simulator system produces a folder containing pictures and one CSV record. The picture folder contains three pictures for every frame captured by the left, center and right camera and each row in the CSV document contains measurements like steering angle, for each left, focus and right picture, for one frame.

IV. DATA PREPROCESSING

The information that we accumulate for example the captured pictures are pre-processed before training the model. While pre-processing, the pictures are edited to expel the sky and forward portion of the car. The pictures are then changed over from RGB to YUV and resized to the info shape used by the model. This is done on the grounds that RGB isn’t the best mapping for visual observation. YUV shading spaces is a significantly progressively powerful coding and decreases the data bandwidth more than RGB capture can. In the wake of choosing the final arrangement of frames, the information is increased by adding artificial movements and rotations to show the framework how to recover from a poor position or course.

While augmenting, we self-assertively pick right, left or center pictures, haphazardly flip the photos left/right and change the steering angle. The steering angle is balanced by +0.2 for the left picture and - 0.2 for the right picture. Using the left/right flipped pictures is useful to prepare the recovery from driving circumstances. We likewise haphazardly translate the picture horizontally.

V. MODEL ARCHITECTURE

There are 5 convolutional layers with varying number of filters and sizes and a dropout after that to handle overfitting. In the end, 3 dense layers were added followed by the output layer. Adam optimizer was used for parameter optimization with a fixed learning rate of 0.04. My model has batch size of 100 with number of epochs as 10 having steps per epoch as 300. In addition to that, batches are generated on the fly so as to optimize the memory utilization with the help of batch generator. For performance evaluation during training, mean squared error was used as a loss function to keep a track of the performance of the model.
It shows a high-level architecture of the system. After performing data augmentation on the input images, batches are created from them and fed to the CNN model for training. After the training is completed, the model is used to perform prediction on the steering angle and send the predictions to the Udacity Simulator to drive the car in real time.

**VI. FUTURE WORK**

Our simulator does not include any pedestrians, objects on the road as well as the road is considered to be single lane. Therefore it does not deploy object detection. In addition to that, it does not alter the speed. The car drives with a constant speed. Our track does not cover 90 degrees turns. Another possible improvement would be to consider each of the cameras separately and create CNN models using each stream of images to create a distinct steering command coming from the left, center, and right model. Then averaging the three of these to get a more accurate prediction. We would expect that the majority of the time, this model would have accurate predictions but if one model predicts a steering angle that is very unlike the other two, then it would skew the steering angle in an unexpected direction.
VII. CONCLUSIONS

In this project, I was able to successfully predict the steering angles using convolutional neural networks as well as was able to understand the inner details of convolutional neural networks along with the way they can be tuned. We also demonstrated that CNN’s are able to learn the entire task of lane and road following without manual decomposition into road. A model is considered a good model if it does not suffer from losses like underfitting and overfitting. Overfitting is a modelling error, it occurs when a function is too closely fit to a set of limited data points. If the training loss and validation loss is getting converged to a lower value, it is a sign of a good model.

REFERENCES

6. Visualizing and Understanding Convolutional Networks by Matthew D. Zeiler, Rob Fergus.