Deep Neural Network approach for navigation of Autonomous Vehicles

Mayank Raj
Dept. of Computer Science and Engineering
Manipal Institute of Technology
Manipal , India
email : rmayank331@gmail.com

Narendra V G
Dept. of Computer Science and Engineering
Manipal Institute of Technology
Manipal , India
email : narendra.vg@manipal.edu

***Abstract*—Ever since the DARPA challenge on autonomous vehicles in 2005, there has been a lot of buzz about 'Autonomous Vehicles' amongst the major tech giants such as Google, Uber and Tesla. Numerous approaches have been adopted to solve this problem which can have a long-lasting impact on mankind. In this paper, we have used Deep Learning techniques and TensorFlow framework with the goal of building a neural network model to predict (speed, acceleration, steering angle and brake) features needed for navigation of autonomous vehicles. The Deep Neural Network has been trained on images and sensor data obtained from comma.ai dataset. A heat map was used to check for correlation among the features and finally four important features were selected. This was a multivariate regression problem. The final model had five convolutional layers followed by five dense layers. Finally, the calculated values were tested against the labeled data where mean squared error was used as a performance metric.**

Keywords—Convolutional Neural Network, Autonomous vehicles, Deep Learning

I. INTRODUCTION

In recent times, several industries have witnessed radical transformations due the rapid advancement made in the field of Artificial Intelligence. Automobile industry has seen enormous disruptions in the form of several new features such as Lane Keep Assist, Adaptive cruise control etc. which have already demonstrated a high level of autonomy [1]. The most disruptive impact will happen with the introduction of autonomous vehicles. Major tech. giants have already invested enormous amount of resources in this research. We have tried to explore and improve one of possible approaches of this domain. Most of the traditional autonomous driving approaches have been divided into two major categories [2][6] :

A. *Traditional autonomous driving system*

This approach is quite analogues to the way a human eye functions. It consists of separate components for perception, localization and mapping ,path planning and decision making and system components. In this approach there are dedicated functions for specific task. The major issue with this approach is it's over reliance on pre-built maps and it's inability to navigate on unmarked roads .This reduces their ability to learn and adapt .

B. *End-to-end autonomous driving approach*

An end-to-end approach is one of the most popular approaches for developing an autonomous vehicle which predicts the decisions by using a deep neural network to mimic the driving behaviors from observed images and information. It consists of a single centralized system that performs all the processes ranging from mapping inputs from sensors such as camera and LiDAR to issuing commands such as velocity, steering-angle, acceleration, and brake. Even though this approach simplifies self-driving systems, it is still extremely difficult to build a model that can fully control an autonomous vehicle.

In this paper we have taken the latter approach . In order to work on our idea we decided to select an existing open-sourced labeled dataset . After careful consideration , it was decided that we would use the comma.ai dataset [3] which consists of video and sensor data of 7 and half hours of highway driving under various lightnings and weather conditions.

II. EXPLORING THE DATASET

A. Image processing

The comma.ai data-set had images stored in the hierarchical format. H5py package of python was used to explore the data .The images stored were of dimensions 3 x 160 x 320 at 20Hz . Given the enormity of the data-set . It was decided to resize the images into numpy arrays of dimensions 3 x 80 x 80 . It has been established that features denoting vehicle dynamics can be used to mimic the decisions taken by a human driver .For the sake of this project only a part of the entire data-set was randomly selected . So the final data-set had 134732 images for training, 33684 images for validation and over 18000 images for the testing the performance of the trained model obtained under various conditions.

B. Pre-processing sensor data

There were several features in the log files which had an impact on the movement of the vehicles. The features which were eventually picked for making the model were speed, acceleration, steering angle and brake as these features have a major impact on the vehicle dynamics [4]. In order to make the model efficient, it was important to determine the correlation between the features. Correlation is a term which is used to measure the extent of linearity between two variables. Features with high correlation are more linearly dependent. So, when two features have high correlation, one of those features can be dropped.

1. Correlation between features

A heat map was built to show the correlation between the variables using the seaborn package. A value close to 1 indicates strong positive correlation and a value close to zero indicate a strong negative correlation . The heat indicated that there was strong positive correlation between acceleration and steering angle, whereas there was no correlation between other features.

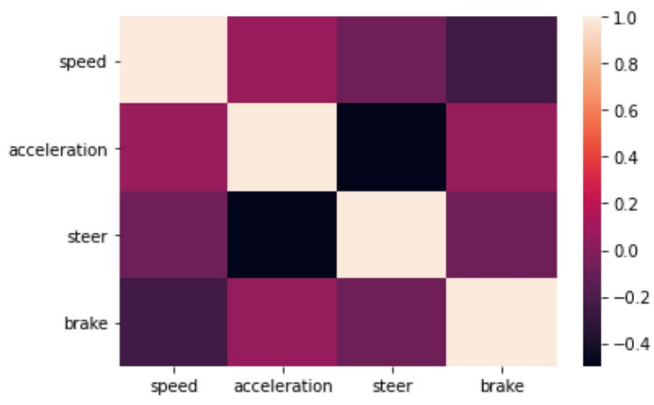

Fig. 1 Heat map of selected features

2. NORMALIZATION OF FEATURES

The data of the log files was recorded at a much higher rate than the images. The sensor data was recorded at 20Hz. A feature ‘cam_ptr’ was used to align the images with the respective log data. All the features were normalized with the following equation:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Eq. 1 Normalization function

Where ‘X’ is value of the respective feature and ‘Xmin’ and ‘Xmax’ represent the minimum and maximum values, respectively. This ensured that the values for all the features were between 0 and 1. Finally four necessary features were selected for building the model and were placed in a NumPy array ‘y_train’.

III. NETWORK ARCHITECTURE

The deep convolutional neural network has been trained on the images as well as the sensor data. Convolution layers are the foundation of any convolutional neural networks. A convolution is the simple application of a filter which helps with feature detection. Here a series of convolution layers were used to identify features in the image which helped in accurately predicting the target variables. Its architecture consists of five 2D-convolutional layers and five dense layers. Each convolution layer is followed by a pooling layer. The neural network recognizes features in the images and predicts all the four parameters. A regression output layer

predicts a steering angle, acceleration, speed and brake value. The activation function used for all the hidden layers was Relu (rectified linear unit), whereas the linear activation function was used for output layer. The loss function used was mean squared error[5]:

$$\textbf{MSE} = \frac{1}{n} * \Sigma_{i=1}^{i=n}$$

Eq. 2 Mean squared error

Here (eq. 2) 'n' is the length of training data, 'y_i' is one of the selected features and ' y_i ' is the predicted value.

Table I – Network Architecture

Layer (type)	**Output Shape**	**Param #**
Conv2d_11 (Conv2D)	(None,128,78,78)	3584
Max_pooling2d_9 (MaxPool 2D)	(None , 64,39,78)	0
Conv2d_12 (Conv2D)	(None,256,37,76)	147712
Max_pooling2d_10 (Maxpool2D)	(None,128,18,76)	0
Conv2d_13 (Conv2D)	(None,256,16,74)	295168
Max_pooling2d_11 (Maxpool2D)	(None, 128,8,74)	0
Conv2d_14 (Conv2D)	(None, 128,6,72)	147584
Max_pooling2d_12 (Maxpool2D)	(None, 64,3,72)	0
Conv2d_15 (Conv2D)	(None, 128,1,70)	73856
global_max_pooling_3 (GlobalMaxPooling)	(None. 70)	0
Dropout_3 (Dropout)	(None, 70)	0
dense_7 (Dense)	(None, 256)	18176
dense_8 (Dense)	(None, 256)	65792
dense_9 (Dense)	(None, 128)	32896
dropout_4 (Dropout)	(None, 128)	0
dense_10 (Dense)	(None, 64)	8256
dense_11 (Dense)	(None, 4)	260
Total params :793,284 Trainable params: 793,284		
Training samples: 134732		
Validation samples: 33684		

Dropout function was used for regularization. The dropout function plays a very important role, it ensures that the model is less sensitive to specific weights of neurons. This use of dropout function helps in building a network that can make better predictions and is less prone to over-fitting on the training data-set.

Gradient descent is used to find the optimum value of parameters involved in the cost function. The significance of gradient descent cannot be overlooked in any deep learning model.

The Adam optimizer was used for gradient descent. To estimate the moments, Adam optimizer utilizes exponentially moving averages, computed on the gradient evaluated on a current mini batch:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Eq. 3 Adam Optimization function

Here m and v are moving averages, g is gradient on current mini-batch, and betas — new introduced hyper-parameters of the algorithm. The default values of the hyper parameters gave good results here and hence these values were selected.

The final model architecture is shown in table I.

IV PRELIMINARY ANALYSIS

The model was built after fifteen epochs. The model was trained on 134732 images and was validated on 34682 images. It was clear that the model was able to perform multi-variate regression and predict four features with a great accuracy. The loss values and mean squared error have been plotted below.

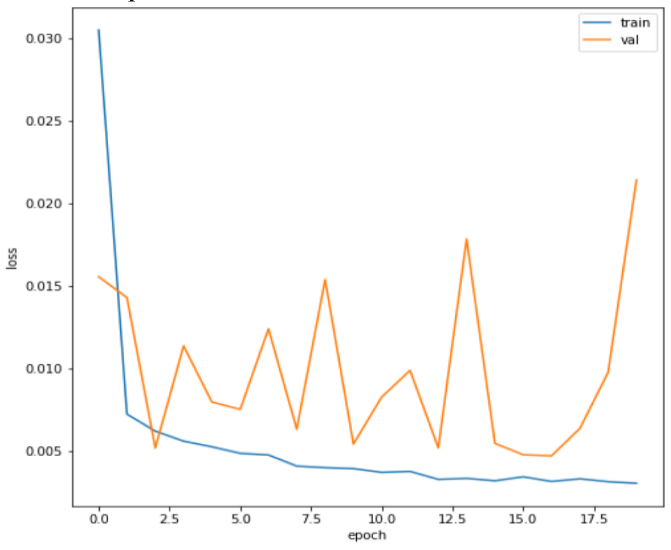

Fig. 2 Loss value over epochs

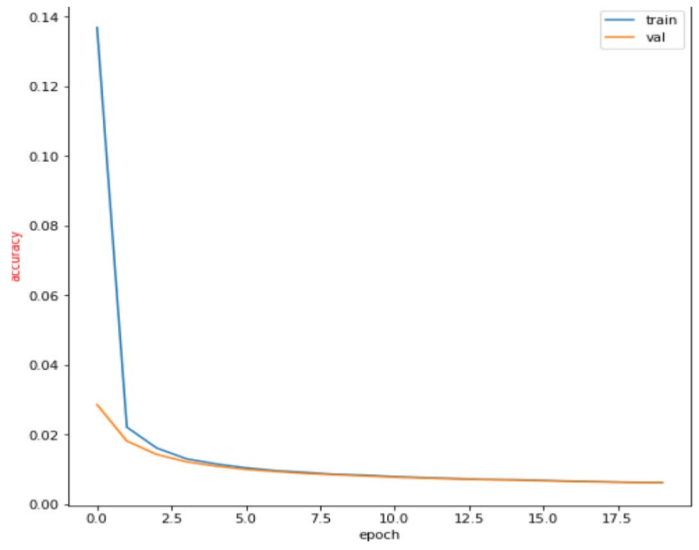

Fig.3 Mean squared error over epochs

As shown in the graphs (Fig. 2 and Fig. 3) it has been shown that for the training as well as the validation data the mean squared error continues to fall till the last epoch .Whereas the loss value for training data continues to fall till the last epoch, but the loss for the validation data reaches its minima at the fifteenth epoch and then starts increasing.

The performance of the model on test data is shown below.

1. Speed

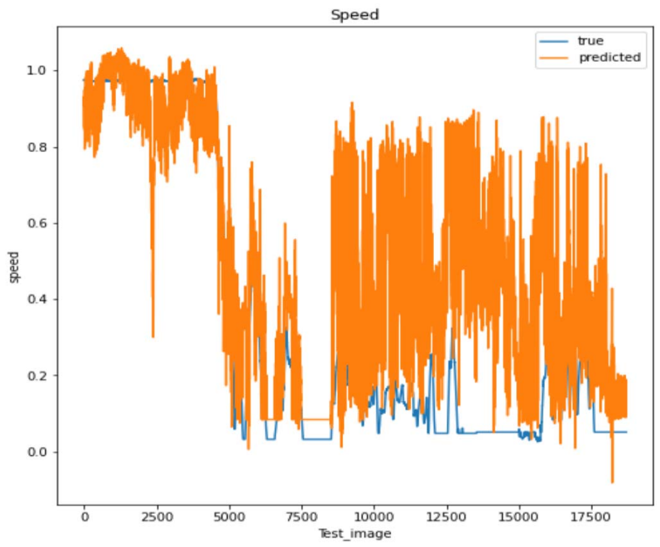

Fig. 4 performance of speed

The vehicle saw minimal variation in speed in the first 5000 images , beyond which there was a steep fall in the speed, which is also validated by the changes observed in acceleration .The performance of this model on unseen data for 'speed' where the mean squared error for this feature is **0.0851** .

2. Acceleration

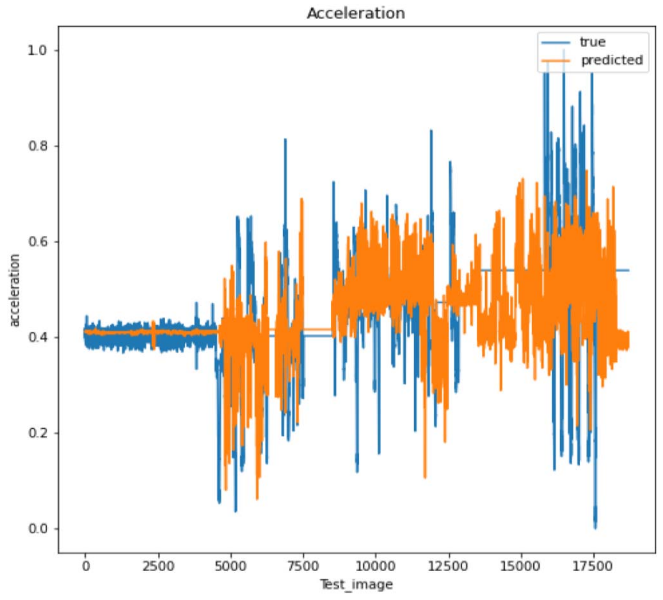

Fig. 5 performance of acceleration

This feature has a negative correlation with steering-angle and this has been established by the two graphs. The accuracy for this metric is very high, where the mean squared error is **0.012**.

After careful comparison of speed and acceleration parameters it became clear that initially when acceleration had minimal variation the speed was following the similar trend whereas when speed dropped off the cliff around the five thousandth test image , there were lots of spikes in acceleration .

3. Steering Angle

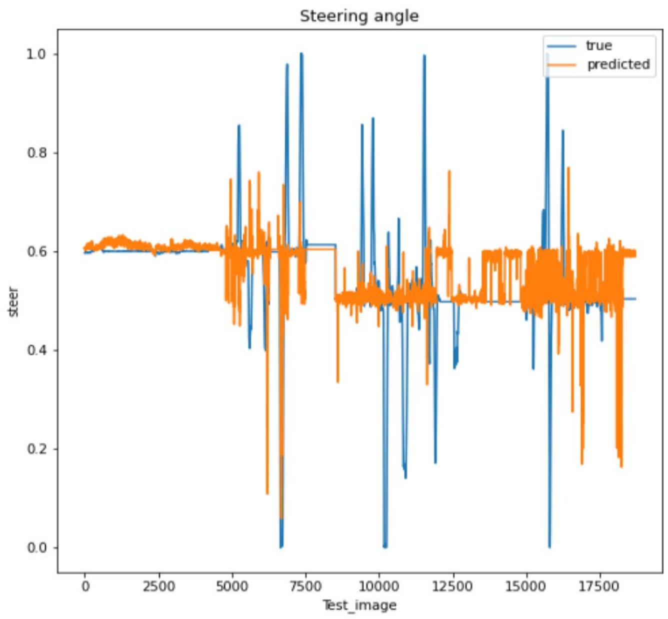

Fig. 6 performance of steering angle

The performance of the steering angle follows the similar pattern of acceleration and Speed. The mean squared error for this feature is **0.0109**. As shown in the previous two graphs, a sharp change in acceleration and speed leads to a sharp change in the steering angle.

4. Brake

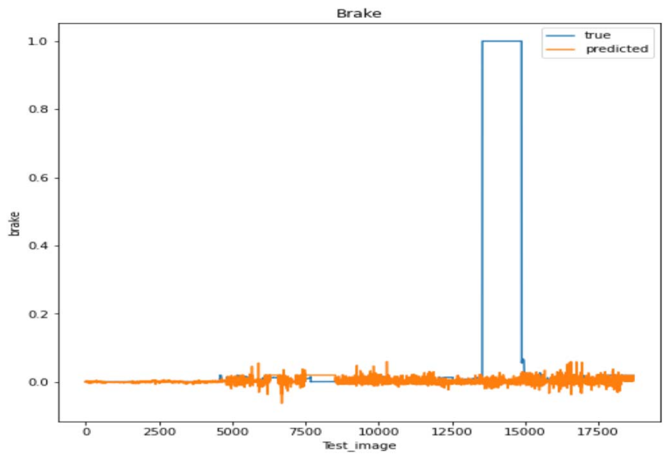

Fig. 7 performance of Brake

The mean squared error for 'Brake' feature was **0.0714.** So the graph of test data shows that brake value is close to zero for most part of the journey, except for a short while around 12000th image. At this point the steering angle experiences very sharp turn indication a major change in the vehicle's trajectory, however the model fails to detect this, indicating that the prediction for break value is highly sensitive to the outliers.

V CONCLUSION

In this paper, we investigated a real-world challenge pertaining to fully autonomous motion of self-driving vehicle, the aim was to improve the accuracy by combining the vehicle dynamic features for an autonomous vehicle in computer vision from its images taken by cameras and sensor data. Our major technical contributions was on a deep neural network which was trained on images and various sensor data obtained from the comma.ai dataset which can accurately predict the features that are important for vehicle dynamics of host vehicle to improve the predictions of speed, acceleration, steering angle and brake .

The work began with exploring the data set which consisted of images as well as log files. The images had to be resized and the log files were normalized. The data was visualized and stored in a compressed file. This was followed by a check for correlation among the features and the feature showing high p-value was dropped.
Eventually the model was formed consisting of initially five convolutional layers, which were connected to five dense layers and that performed regression to produce the final outputs. The performance was evaluated by using mean squared error as a performance metric and the test values were plotted against the predicted values.

5.2 Future Scope

This project can be taken forward by including more and more features which would make the car more aware of its surrounding and would get closer towards a fully autonomous vehicle capable of navigating in any environment

REFERENCES

1. Shobhit Sharma , Girma Tewolde , Jaerock K won , " Lateral and Longitudinal motion control of Autonomous vehicle using Deep Learning" . 2019 IEEE International Conference on Electro Information Technology (EIT)

2. Jiakai Zhang , " End-to-end decision making for Autonomous driving" Department of Computer Science , New York University ,May 2019

3. Santana Eder and George Hotz. "Learning a Driving Simulator." - comma.ai-2016, *ArXiv* abs/1608.01230 (2016): n. pag.

4. Tsung-Ming Hsu , Cheng-Hsien Wang ,Yu-Rui Chen , " End-to-End Deep Learning for Autonomous Longitudinal and Lateral Control based on Vehicle Dynamics" , AIVR 2018: Proceedings of the 2018 International Conference on Artificial Intelligence and Virtual Reality

5 Johann Haselberger, Dr. Jian Chen , Professor Bernhard Schick , "Deep Learning for lateral vehicle control – an end-to-end trained multi-fusion steering model" .

6. Deheng Qian , Dongchun Ren , Yingying Meng , Yanliang Zhu,Shuguang Ding, Sheng Fu, Zhichao Wang1, Huaxia Xia , " End to End Driver Policy using Moments Deep Learning " 2018 IEEE International Conference on Robotics and Biomimetics .