Self Driving Car Simulation

Pruthvish E
PES1201701629
PES University
umapruthvish@gmail.com
Bangalore

Ashrith Sridhar
PES1201700649
PES University
ashrith.sridhar@gmail.com
Bangalore

R Ananth
PES1201700286
PES University
rananth108@gmail.com
Bangalore

***Abstract*— Autonomous navigation and self-driving cars are extensively researched and deep learning plays a crucial role in this area. A self-driving car, also known as an autonomous vehicle, is a vehicle that is capable of sensing its environment and moving safely with little or no human input. Controlling the steering wheel is one of the most important tasks while driving. our model uses CNN to predict the angle of the steering wheel while driving. Based on only the image of the surrounding environment our model will predict the steering angle of wheel and simulates the steering wheel action in a real time test dataset as frames of images or videos.**

I. Introduction

Positioning the steering wheel at exact angles in real time environment needs a lot of practice and should be completely accurate with very less margin of error as a lot of life's depend on it, so we chose this problem to accurately predict the steering angle in real time environment.

Our CNN model is able to learn the entire task of lane and road following without manual decomposition into road or lane marking detection, semantic abstraction, path planning, and control. The system learns for example to detect the outline of a road without the need of explicit labels during training.

Our CNN is able to learn meaningful road features from a very sparse training signal from steering angle alone.

Our model learns to predict the steering angles with very little margin of error in just 30 epochs compared to other methods like reinforcement learning and sequential models like RNNs and LSTMs.

Our model compared to other CNN models attains convergence quicker as our model works on reduced dimensions of the variables and features and is more robust and predicts the steering angle accurately even when tested on other terrain and conditions of the environment (road).

II. Solution Architecture

Images are first preprocessed to irrelevant features are removed and then resized to a smaller image using INTER_AREA interpolation of cv2 module.

Images are segregated into train and validation sets.

Train set is passed in batches to the CNN model which is initialized with random normal weights , then after the feedforward of data through the network the weights are validated with the validation set then loss is calculated for which we use MSE with L2 normalization.

We use Adam optimizer in backpropagation and weights are updated, and the process is run till we reach a suitable accuracy.

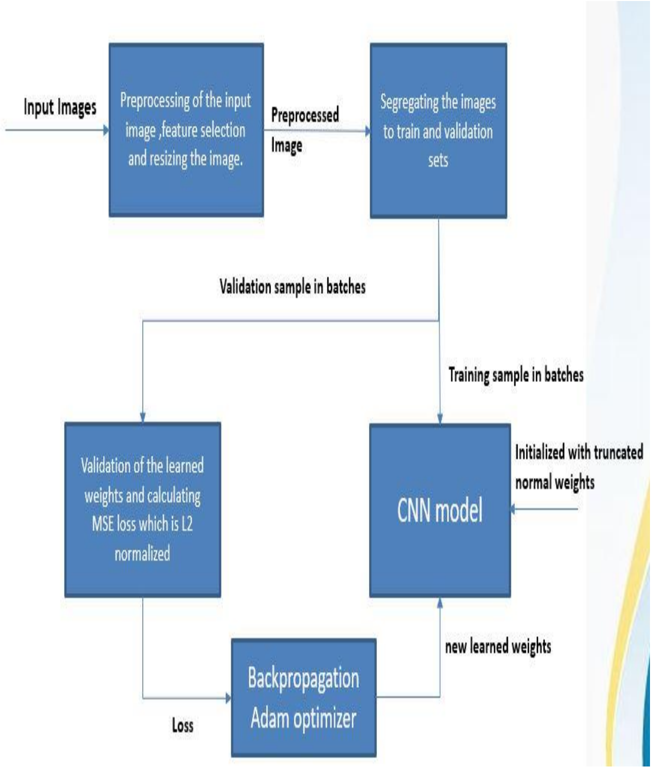


Fig 1. Solution Architecture

III. Solution

A. Algorithm

a) preprocessing of the image is done by cropping the top portion of the image, the sky as it does not contribute much to our steering angle and the processed image is resized to just 120*40*3.

b)Splitting the dataset to training and validation sets in the ratio 80:20.

c)CNN model

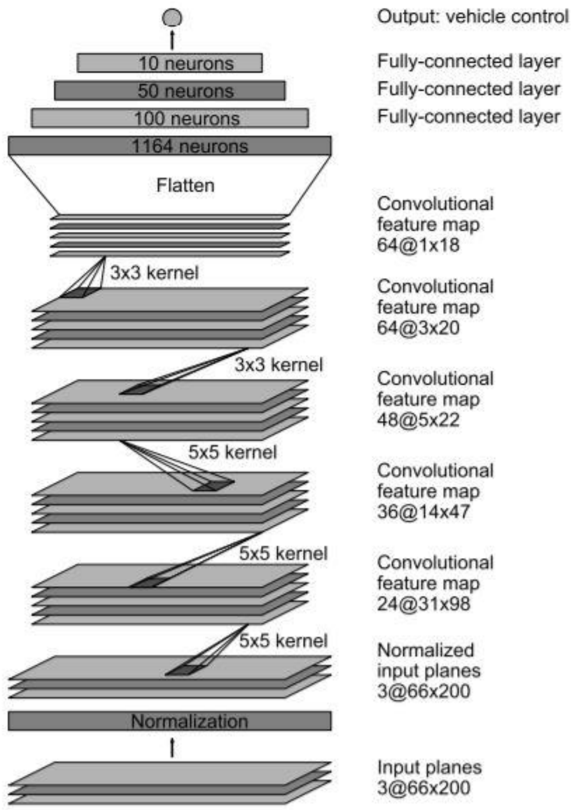
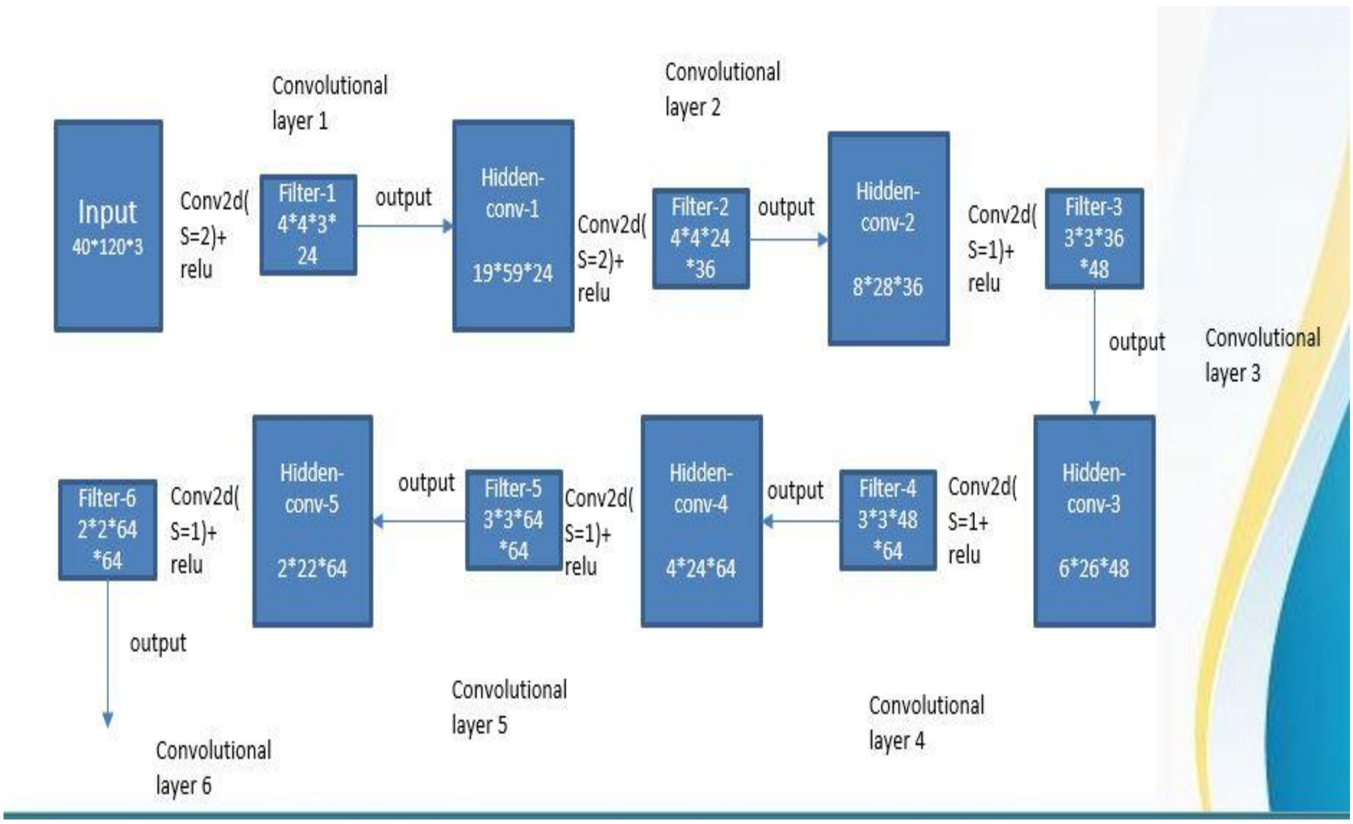


Fig 2.Nvidia's model

*Our model is slightly different from the Nvidia's model, We have an extra convolution area and our filter sizes are different, we have reduced the input size to just 40*120.*



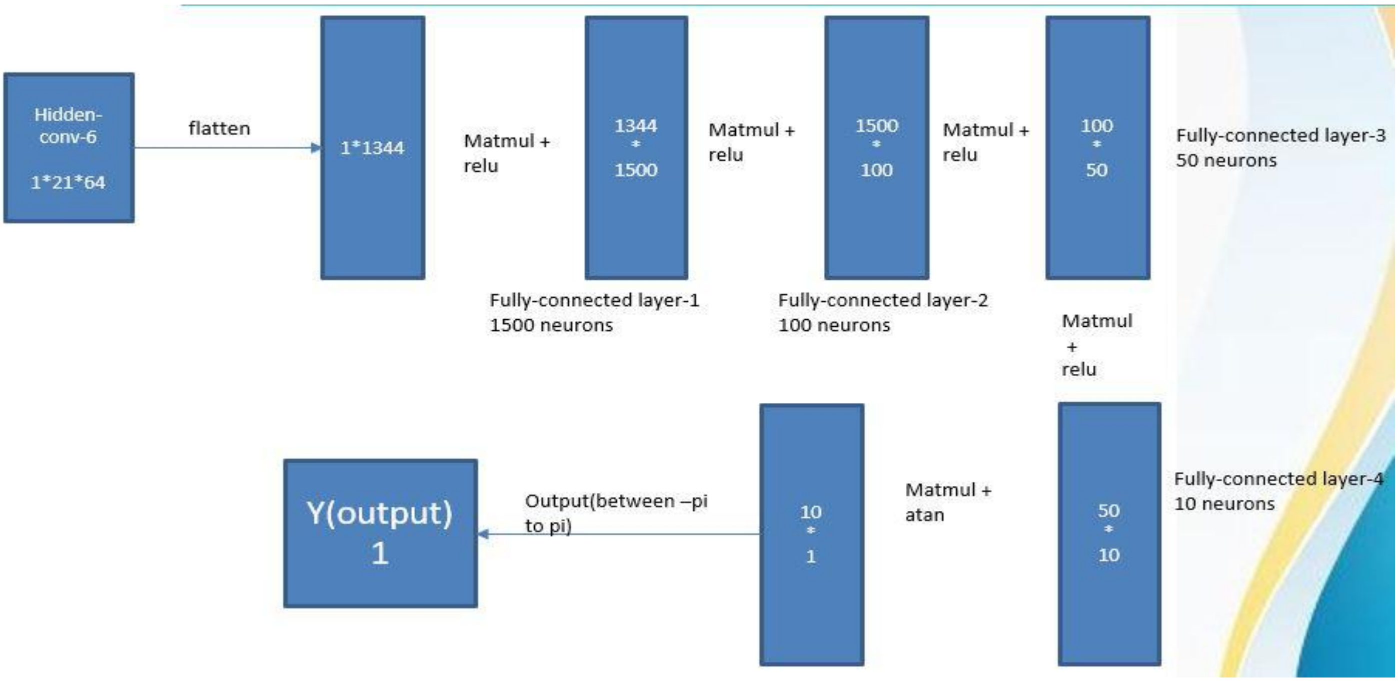


Fig 3. Our CNN Model

d) backpropagation is done using Adam optimizer and loss function as MSE with L2 normalization.

B. Math

Input = [100, 40, 120 ,3]
(batch, height, width, channel)

Padding = 0 for all convolution layers.

Filter-1 = [4, 4, 3, 24]
(filter height, filter width, input channel, output channel]

Convolution layer 1

Stride = 2

Width of output = (Input width – Filter width)/stride + 1

= (40 – 4)/2 +1 = 19

Similarly, height of output = (120-4)/2 + 1 = 59

Therefore hidden_conv_1 = [100,19,59,24]

Convolution layer 2

Strides = 2
Filter-2 = [4, 4, 24, 36]
hidden_conv_2 = [100,8,28,36]

Convolution layer 3

Strides = 1
Filter-3 = [3, 3, 36, 48]
hidden_conv_3 = [100,6,26,48]

Convolution layer 4

Strides = 1
Filter-2 = [3, 3, 48, 64]
hidden_conv_4 = [100,4,24,64]

Convolution layer 5

Strides = 1
Filter-2 = [3, 3, 64, 64]
hidden_conv_5 = [100,2,22,64]

Convolution layer 6

Strides = 1
Filter-2 = [2, 2, 64, 64]
hidden_conv_6 = [100,1,21,64]

then hidden_conv_6 is flattened to 1344 one dimensional tensor.

Then It passes through 1500,100,50,10 neurons forming fully connected layers.

MSE loss with L2 normalization is calculated. And Adam optimizer is used in backpropagation and the weights are updated.

C. Testing Carried out

Our model was trained with a dataset of the roads of California and was tested in the roads of India under various circumstances like sudden obstacles in the way, the model does a good job predicting the steering angles.

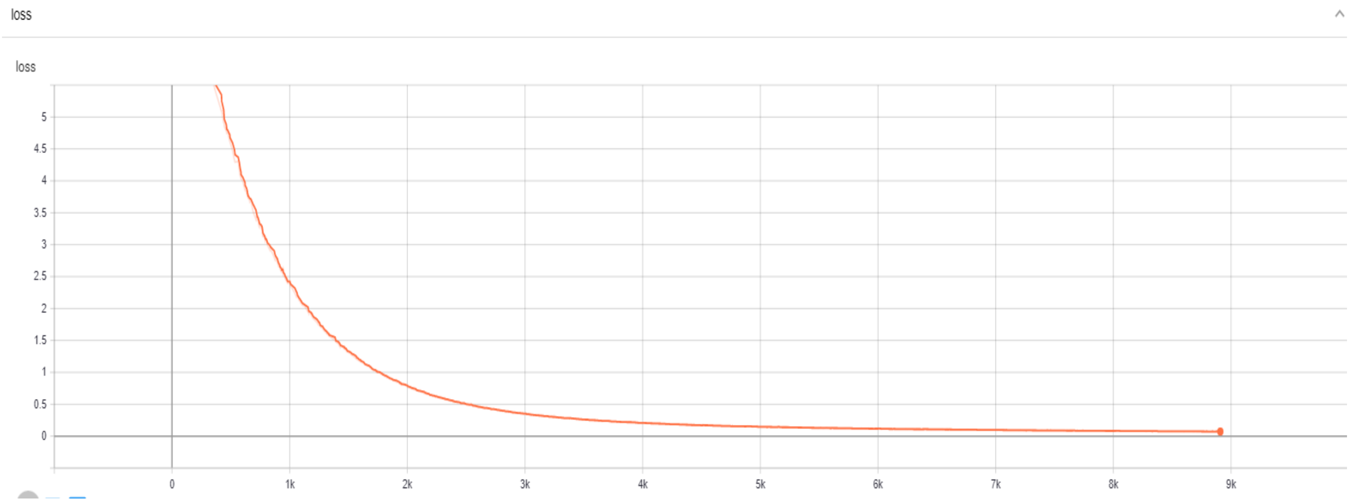


Fig 4. Loss vs Epoch

D. Criteria for declaring our solution good

Our has a high accuracy ~95% , we trained our model on the images of California roads and tested it on the roads of Indian.

Our model performs better and attains convergence quicker than the one on https://arxiv.org/pdf/1604.07316.pdf

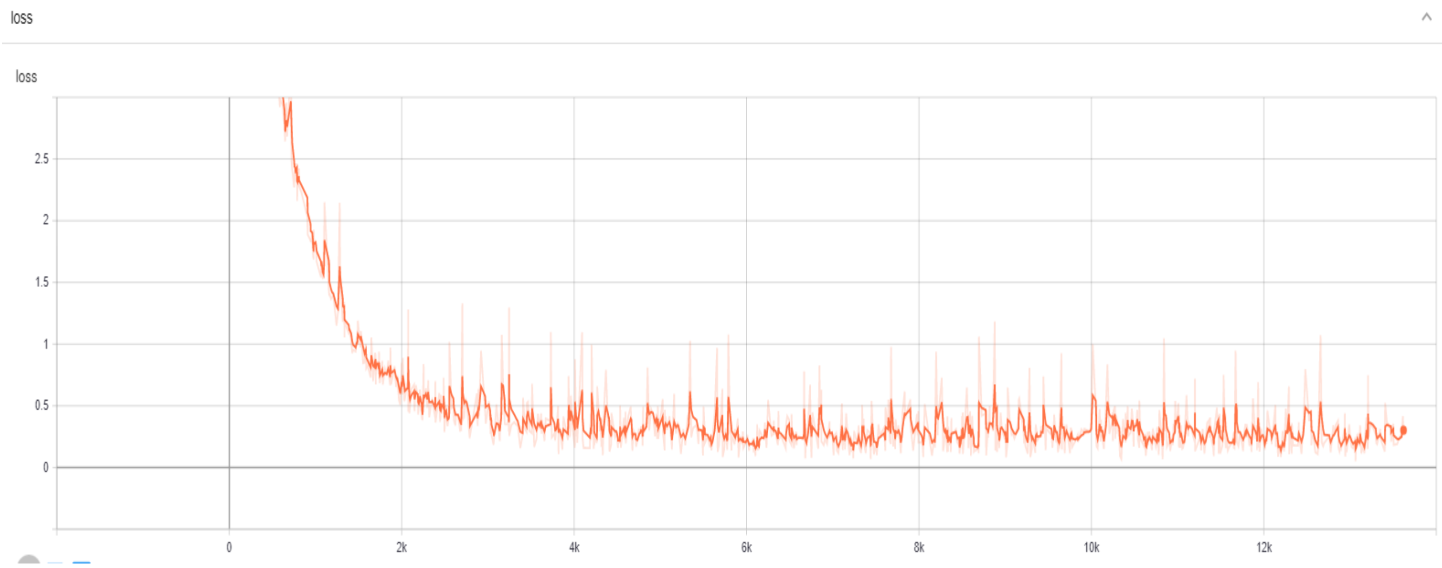


Fig 5. Nvidia model's Loss vs Epoch

Time for Convergence on the same system:
Our model: 1 hr. 37 min
Nvidia's model: 2 hrs. 42 min

IV. Constraints, Assumptions & Dependencies

A. Constraints

Steering angle should be predicted from only a single image of the present surrounding, irrespective of the car's history.

B. Assumptions

Image feed to the network should be taken from the center of the car's present location. (camera attached to the center of the car).

C. Dependencies

TensorFlow, cv2, scipy.

V. Future Work plan

To train our model to predict with high accuracy the acceleration and braking required to autonomously drive the car with complete safety.
To use history of the previous decisions made to make the present decision, which will be useful when integrating steering angle with acceleration and braking.

VI. References

[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, Karol Zieba. **End to End Learning for Self-Driving Cars,** *25 Apr 2016.*
URL: https://arxiv.org/pdf/1604.07316.pdf.