Predicting Steering Angle for Self Driving Vehicles

Mohith G G

Mohith GG, Student, Bangalore Institute of Technology, Bangalore - 560004

---***---

Abstract - Self-driving cars have a significant future since they save time that can be utilised productively. Recently, several accidents were caused by self-driving automobiles. In this suggested study, a CNN model is created to forecast the steering angle for self-driving cars, since steering angle prediction is their core notion. The simulator is used to capture the CNN model's steering angle and road visuals. The trained model ran autonomously in the simulator, completing full laps with training loss of 10.74% and validation loss of 11.89%. This task is between level 1 and level 2 vehicle automation. The system can help build self-driving cars. The system's technique may be deployed in actual cars and prototypes, enhancing self-driving vehicle safety.

Key Words: Self-driving, machine learning, deep learning, object detection, classification

1. INTRODUCTION

This project's objective is to build a convolutional neural network (CNN) model that, if deployed, will make it possible for a simulated car to operate independently. This will be achieved by making precise predictions of the steering angle needed to safely operate the vehicle while driving. Specifically, neural networks and other AI-related concepts are used. The dataset of human drivers will be used to train the machine at first. Through this, the model may acquire the knowledge necessary to correctly predict the steering angle of the car or vehicle at any given time. Artificial intelligence might potentially enhance its prognostic capabilities by training and experience gained from both automated and manual driving. Someday this concept may be refined to the point where it may be used to preexisting highways. Connecting the model to the simulator and then pushing it to work in autonomous mode constitutes the system's functionality. When determining the optimal steering angle, the CNN model takes the path into account. CNNs have powerful skills in the fields of learning and classifying a wide range of characteristics. When given a training dataset, it may develop or extract features that are exclusive to the data on its own. The photos of roadways and the associated steering angles are used to train the CNN model. Once the training phase concludes, the model is re-established with the programme and the simulator for further use. The model will make projections about the angle of the vehicle's steering wheel. The purpose of the work that is going to be done is to enhance the performance of autonomous cars that are already on the market in terms of their level of safety. In today's world all are automated, so there is big demand for automation in every field, even in the field of automobiles. There are several research going on automated vehicles Autonomous cars. Self-Driving Vehicles can move safely with little or no human input. 6 tiers of self-driving cars (0- 5 levels).

With current CNNs, it is possible to outperform humans on benchmark datasets, which has profoundly altered the landscape of image and pattern recognition. Most pattern recognition tasks were formerly accomplished using a combination of hand-crafted feature extraction and a classifier before the widespread use of CNNs. CNNs' main strength lies in the fact that they can learn features automatically from training instances, eliminating the need for humans to manually choose only sensible characteristics. CNNs excel in accuracy compared to a regular flattened neural network because they use the 2D structure of pictures to their advantage. Although CNNs with learnt features have been in use in the business sector for over twenty years, their popularity has skyrocketed in recent years as a result of two breakthroughs. The first is the availability of huge, labelled data sets for training and validation, such as the Large Scale Visual Recognition Challenge (ILSVRC). Second, convolutional neural network (CNN) learning techniques have been adapted for use on massively parallel graphics processing units (GPUs), which significantly quickens the processes of both learning and inference. A CNN that can do more than just recognise patterns is described here. An whole processing pipeline necessary to control a vehicle is learned by the system. One sort of deep neural network architecture, the Convolutional Neural Network (CNN) is optimised for tasks like picture categorization. The first part of a CNN is called the input layer. Also included is an output layer, which is normally a one-dimensional group of neurons. In order to process images, CNN employs a series of convolution layers that are only loosely coupled to one another. Additionally, they have down sampling layers, also known as pooling layers, which help to lower the required number of neurons in the following layers of the network. Last but not least, in order to link the pooling layer to the output layer, CNNs often use one or more completely linked layers. We can take little bites out of an image's visual elements using a method called convolution. In a convolution layer, each neuron is in charge of a tiny group of neurons in the layer above it.CNN works well for a variety of tasks including image recognition, image processing, image segmentation, video analysis, and natural language processing. Like a classic neural network, convolution is a linear process that multiplies an input by a set of weights. Since this method takes in data in a two-dimensional array, the multiplication operation is carried out between the filters or weights in that array. Networks for visual perception (CNNs) use a wide variety of layered structures.

1.1 Contributions

- To develop a model which can be used in self driving vehicles.
- Focused on basic level of implementation, i.e., intermediate between level 1 and 2 automation.
- Main aspect of the project is to maintain vehicle on the road(steering).
- A CNN model will be trained using gathered road visuals and steering angle data to enable autonomously controlled vehicle navigation.
- To improve the security of autonomous cars while on the road.

2. Literature survey

A literature review, also known as a narrative review, is a sub-genre of review article as well as a type of academic paper that summarises the most recent findings, theories, and methods related to a specific subject area. This type of review may also be referred to by its alternative name: narrative review. Literature reviews are considered secondary sources since they do not report on new or previously unpublished material. Research in practically every academic discipline begins with a study of the relevant body of prior work. Evaluation, exploration, and application are the three primary approaches to writing a review of the relevant literature.

Features are extracted from raw data and transformed into a set. The Feature Extraction process in machine learning takes the initial consistent data and creates the borrowed values, also called features, which are meant to be descriptive and non-redundant, thereby streamlining the subsequent learning and observed phases.

Reducing the resources required to define a massive data collection is a primary goal of feature extraction. The fundamental issue that arises from the complex sum of variables during analytical inquiry of intricate data. Large-scale analyses are notoriously memory and processing-intensive since they use the classification algorithm to "overfill" the training pattern with the new pattern's data.

The process of devising a series of variables to solve these challenges while effectively defining the data is known as "Feature Extraction."

Here, images are input into one of three different models or setups:

1. CNN
2. MLP (MULTILAYER PERCEPTRON)
3. MLP along with manual feature extraction

The three methods are summarised in Figure 1. For the survey, we logged all outcomes and tried to guess which method would turn out to be the most effective. The artificial neural network known as a multilayer perceptron (MLP) is a kind of feed forward network (ANN). Nodes are organised into at least three distinct levels in an MLP, with the input layer serving as the foundation for the hidden and output layers. Each node is a neuron with a nonlinear activation function, except for the input nodes. By way of training, MLP employs the supervised learning method of back propagation. In contrast to a linear perceptron, MLP has both several layers and non-linear activation. [1]

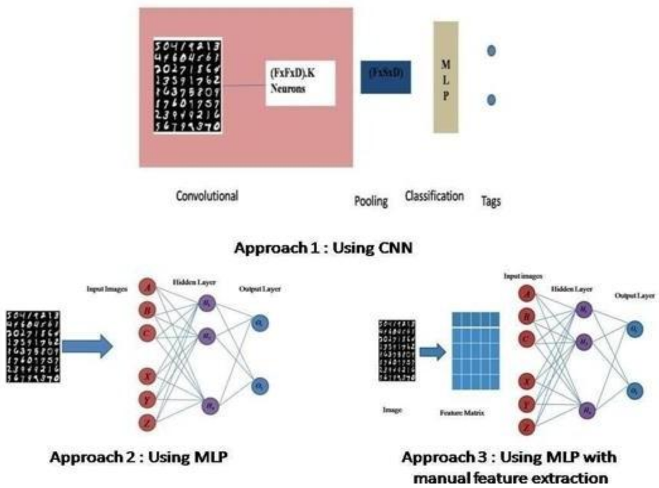

Figure 1: Survey carried on three methods.

To improve the efficacy of machine learning algorithms, feature extraction involves transforming training data and establishing it with additional features. Using tools from Deep Learning, including Convolutional Neural Networks, this method can determine the effects of fully automated feature extraction and distribution (CNN). [2]

We compared the performance of a Convolutional Neural Network equipped with automated feature extraction to that of a conventional Multi-Layer Perceptron using the whole picture and a typical Feature Extraction method. It is intended to provide a review of the modern make-up of Feature Extraction methods that has arisen over the last several years. As the raising of application demand increases, a large study and analysis in the Feature Extraction platform became very active.[3]

Data is collected by the unmanned vehicle's cameras, radars, and other sensors, and the vehicle's computer system powers the intelligent driving instrument necessary for the vehicle to operate autonomously. The ability of an autonomous vehicle to correctly identify lanes is a crucial component of its overall safety performance.

Numerous studies are being conducted on various lane line recognition techniques at present, with most falling into one of two categories: feature-based or model-based. For lane detection, you may employ techniques like boundary tracking, etc. It is still difficult to precisely recognise the lane line in the gathered lane line picture due to the effects of light, wear, vehicle shade, and tree shadow.

The approach consists of 3 stages:

- Pre-Treatment
- Image Processing
- Restore to the original perspective

Camera calibration and identifying the region of interest are two examples of the preparatory work performed before actual treatment begins. Edge detection, colour thresholding, obtaining lane line pixels by combining edge and colour filtering, sliding window, and polynomial fitting are all examples of procedures used in image processing. Tests conducted on the Open CV platform confirm the algorithm's superior real-time and anti-interference performance, demonstrating its ability to accurately identify both the dotted and solid lanes and so enable real-time line marking inside video footage. This technique may be used in the safety-assistant driving system or the autonomous vehicle system to further improve the amount of computation and the resilience of the algorithm. [4]

To do this, we trained a convolutional neural network (CNN) to convert the raw data from a single front-facing camera into directional inputs. This comprehensive strategy yielded impressive results. Using simply the human steering angle as a training input, the system automatically learns internal representations of the required processing processes, such as recognising valuable road elements. Because the internal components self-optimize to maximise overall system performance, rather than targeting human-selected intermediate criteria like lane detection, the system's performance will improve. Figure 2 and 3 shows the block diagram and CNN network used.[5]

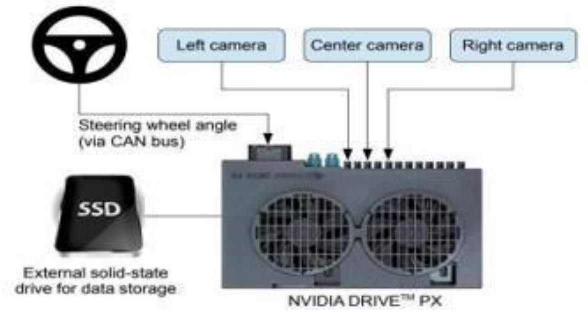

Figure 2: Employed System architecture in paper.

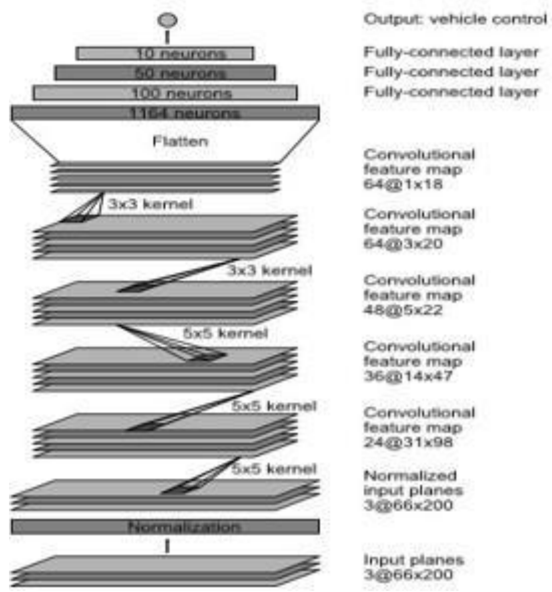

Figure 3: Architecture employed in paper.

In order to function safely, self-driving vehicles need to be able to stick within their designated lanes. Even though autonomous vehicles have a plethora of sensors fitted, including radar, LiDAR, ultrasonic sensors, and infrared cameras, regular colour cameras are still crucial due to their inexpensive cost and ability to glean substantial data.

One of the most critical responsibilities for a self-driving automobile is determining the appropriate vehicle control input based on a camera's collected picture. In the conventional method, several aspects of the problem are treated independently, such as lane recognition, course planning, and control logic. [6]

Image processing methods like colour enhancement, the Hough transform, edge detection, etc. are often used to identify the lanes. After detecting the lanes, the next step is to use that information to inform the path planning and control logic. The success of this method is dependent on being able to properly extract and analyse visual features.

Here we describe an end-to-end learning solution to lane maintaining for autonomous vehicles, which uses frames from the front camera to calculate the appropriate steering angles.

The comma.ai dataset is used for both training and assessing the CNN model. This dataset includes both still images and data on the driver's steering angle while on the road. Based on the results of the tests, it can be concluded that the model can provide reasonably precise vehicle steering. [7]

3. Proposed methodology

The neural networks seen in animals' brains provide as inspiration for ANN. Without task-specific rules, these systems learn from examples. By analysing manually annotated samples and using the results to identify cats in other photos, they may learn to recognise photographs including cats during image recognition. They are unaware of the fact that felines have fur, tails, whiskers, and other feline characteristics. Characteristics used for identification are generated from processed examples.

Artificial neurons, or "neurons," are the building blocks of an ANN and are designed to function like their biological counterparts in the brain. Like biological synapses, each connection has the potential to relay information to nearby neurons. An artificial neuron with the ability to send and receive signals.

The output of a neuron is a non-linear function of its inputs in ANN implementations, and the signal of a connection is a real integer. Connecting edges. As a system learns, the weights of its neurons and edges adapt. The quality of a connection might be diminished by adding more weight to a device. It's possible that neurons have a signalling threshold. Anatomically, neurons have several layers. Each layer may adjust its inputs in its own unique way. This process of sending and receiving signals from the input layer to the output layer may be repeated several times.

Deep learning neurons are layered. One layer's neurons only link to the two layers above and below. Input layer receives outside data. Output is the last layer. Hidden layers exist between them. There are also single-layer and unlayered networks. Multiple connections are available between two levels. Each neuron in one layer may link to the next. A group of neurons in one layer may link to a single neuron in the next layer, lowering the layer's neuron count. Feedforward networks are formed by neurons with solely these connections. Recurrent networks link same- or previous-layer neurons. Fig. 4 demonstrates ANN layering.

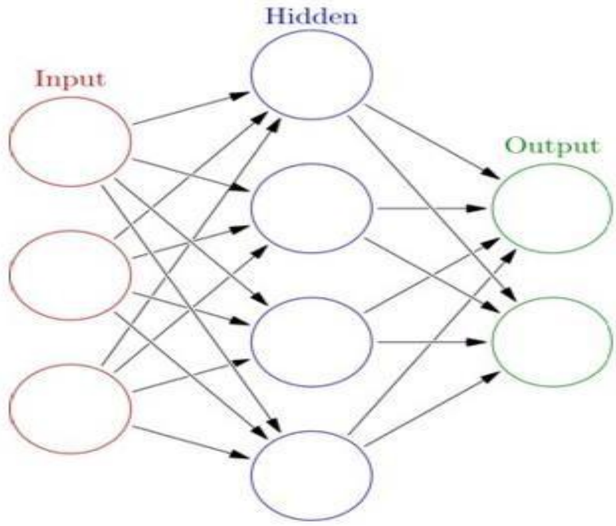

Figure 4: Architecture for neural Network.

3.1 Convolutional neural network

In the field of machine learning, pictures are analysed using deep, feed-forward convolutional neural networks. Convolutional networks, like the visual cortex of animals, use a tree-like pattern of connections. Cortical neurons only respond to stimuli that fall inside their own receptive field. It is possible to fill the visual area because the receptive fields of many neurons overlap. By contrast, CNNs need less work before they can be used. This means the network picks up filters that were specifically designed by humans. It's a significant plus because you don't need to rely on human expertise and work from the past to use this. CNNs are used in NLP, recommender systems, and image/video recognition.

Input, output, and hidden layers are all components of convolutional neural networks. Convolutional layers convolved by multiplication or dot product are often used as the hidden layers in CNNs. Due to the activation function and final convolution, the inputs and outputs of the pooling, fully connected, and normalising layers (together referred to as "hidden layers") are hidden from view. Conventionally, layers are called convolutions. It's a sliding dot product or cross-correlation. This influences how weight is computed at a matrix index point.

3.2 Employed dataset

- The dataset is collected through manual driving the car in the simulator. The images ofthe road and it's respective steering anglesare recorded.
- Then these are stored in the form of driving log, in the format of csv file.

Figure 5: Dataset employed

	A	B	C	D	E
1	C:\Users\	C:\Users\	C:\Users\	0	0
2	C:\Users\	C:\Users\	C:\Users\	0	0
3	C:\Users\	C:\Users\	C:\Users\	0	0
4	C:\Users\	C:\Users\	C:\Users\	0	0
5	C:\Users\	C:\Users\	C:\Users\	0	0
6	C:\Users\	C:\Users\	C:\Users\	0	0
7	C:\Users\	C:\Users\	C:\Users\	0	0.324622
8	C:\Users\	C:\Users\	C:\Users\	0	0.60342
9	C:\Users\	C:\Users\	C:\Users\	0	0.951201
10	C:\Users\	C:\Users\	C:\Users\	0	1
11	C:\Users\	C:\Users\	C:\Users\	0	1
12	C:\Users\	C:\Users\	C:\Users\	0	1
13	C:\Users\	C:\Users\	C:\Users\	0	1
14	C:\Users\	C:\Users\	C:\Users\	0	1
15	C:\Users\	C:\Users\	C:\Users\	-0.12568	1
16	C:\Users\	C:\Users\	C:\Users\	-0.10074	1
17	C:\Users\	C:\Users\	C:\Users\	0	1
18	C:\Users\	C:\Users\	C:\Users\	0	1
19	C:\Users\	C:\Users\	C:\Users\	0	1
20	C:\Users\	C:\Users\	C:\Users\	-0.33841	1
21	C:\Users\	C:\Users\	C:\Users\	-0.37264	1

Figure 6: Driving log

The below figure shows the system architecture of the proposed system,

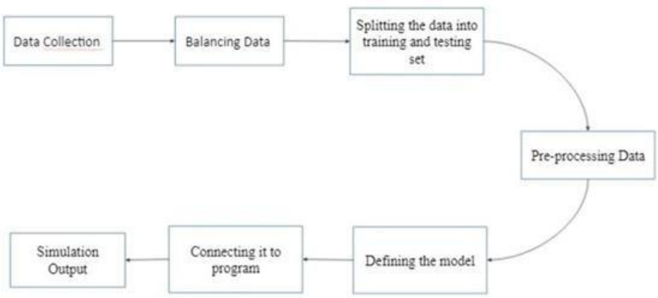

Figure 7: Flowchart of the proposed system

The system design includes the general framework and system architecture. The input goes through a series of stages before its class is predicted by the neural network model. These include image filtering and contour detection. The video capture input is later predicted based on its features. System architecture provides a high-level view of the core component in the application that is, the layers in the Convolutional Neural Network (CNN) model.

3.3 General Framework

The procedure used, which is shown in Fig. 7, to extract thepicture and, by extension, to categorise it, is as follows.

3.3.1. Data Collection

Images of the track and respective steering angle is recorded through manual simulation.

Figure 8: Database selection

3.3.2. Balancing Data

There should be balance in the data, i.e. there should be almost same number of left turns and right turns ,else the model gets biased towards one side of turning.

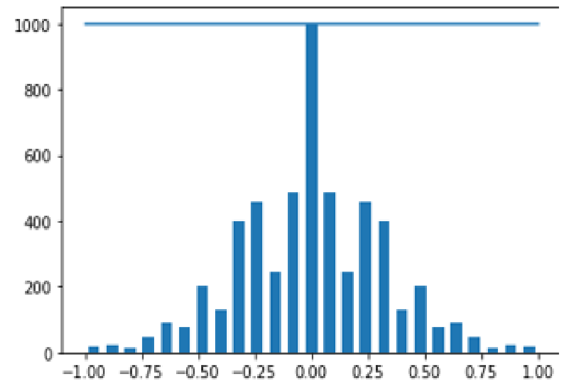

Figure 9: Balancing the model.

3.3.3 Splitting Data

Since model is to defined and trained, that has to be tested to know about errors in the model during training. Hence the entire data is split into training and testing data.

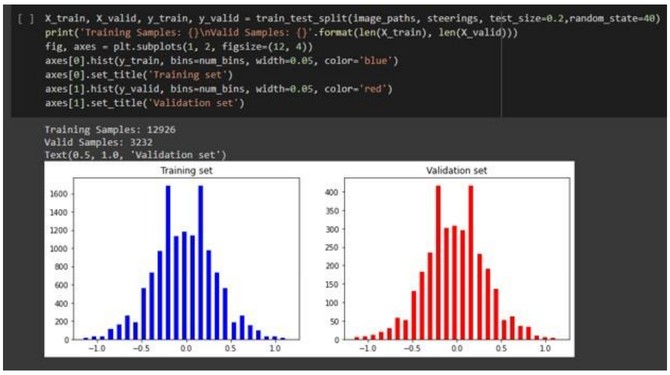

Figure 10: Data splitting

3.3.4 Preprocessing Image

Preprocessing of images are required hence to make training of model efficient and model should have higher accuracy.

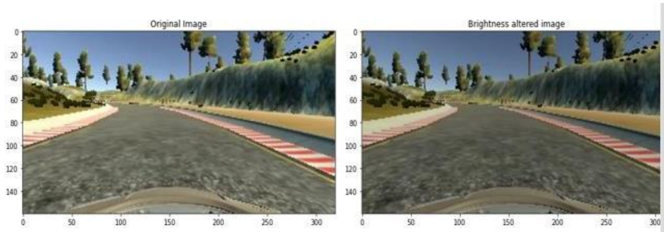

Figure 11: Pre-processing of the image.

3.3.5 Defining a model

A Convolution neural network based sequential model is defined and is trained.

3.3.6 Connecting to the program

The trained model should loaded and connected to the simulation program.

3.3.7 Simulation Output

Model is then made to run in simulator in the autonomous mode.

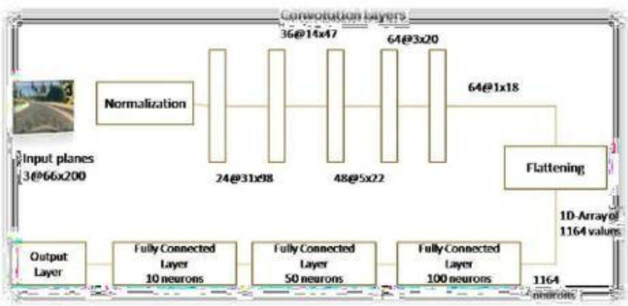

Figure 12: System architecture for the proposed system

Images of the road and the steering angle at which they were taken, captured during manual driving in a simulator, will be included in the dataset. A CNN model is constructed after randomly preprocessing these photos. The constructed CNN model then has to be trained to provide accurate steering angle predictions. It is important to validate the model to see how well it predicts. This is achieved by splitting the dataset into a training set and a validation set. Validation of the model requires only 20% of the data. When the model is complete, it is tested in a simulator to see whether it consistently provides accurate predictions of the steering angle.

```
def our_model():
    model = Sequential()
    model.add(Conv2D(24,(5,5), strides=(2, 2), input_shape=(66, 200, 3), activation='elu'))
    model.add(Conv2D(36, (5,5), strides=(2, 2), activation='elu'))
    model.add(Conv2D(48, (5,5), strides=(2, 2), activation='elu'))
    model.add(Conv2D(64, (3,3), activation='elu'))

    model.add(Conv2D(64, (3,3), activation='elu'))

    model.add(Flatten())

    model.add(Dense(100, activation = 'elu'))

    model.add(Dense(50, activation = 'elu'))

    model.add(Dense(10, activation = 'elu'))

    model.add(Dense(1))
```

Figure 13: Layers designed in the model.

The model is trained by employing the following parameters is as shown in the below figure,

```
history = model.fit(batch_generator(X_train, y_train, 100, 1),
                    steps_per_epoch=500,
                    epochs=10,
                    validation_data=batch_generator(X_valid, y_valid, 100, 0),
                    validation_steps=200,
                    verbose=1,
                    shuffle = 1)
```

Figure 14: Model parameters.

The trained model is loaded by using the h5 file format and it is as shown in the below figure,

```
if __name__ == '__main__':
    model = load_model('model.h5')
    app = socketio.Middleware(sio, app)
    eventlet.wsgi.server(eventlet.listen(('', 4567)), app)

(17524) wsgi starting up on http://0.0.0.0:4567
(17524) accepted ('127.0.0.1', 51514)

Connected
-1.099096417427063 1.0 0.0
-1.099096417427063 1.0 0.0
-1.099096417427063 0.9562 0.438
-0.9653634428977966 0.68109 3.1891
-0.8461366295814514 0.83616 1.6384
-0.9318703413009644 0.87732 1.2268
-0.9249778389930725 0.83287 1.6713
-0.8875439763069153 0.78756 2.1244
-0.7754414677619934 0.731 2.69
-0.6189704537391663 0.63908 3.6092
-0.29554980993270874 0.54893 4.5107
0.0287947840988636 0.45125000000000004 5.4875
0.23017069697380066 0.36968 6.3032
0.265430748462677 0.27122999999999997 7.2877
```

Figure 15: Loading the model.

The below chart shows the activity planned for the implementation of the proposed model.

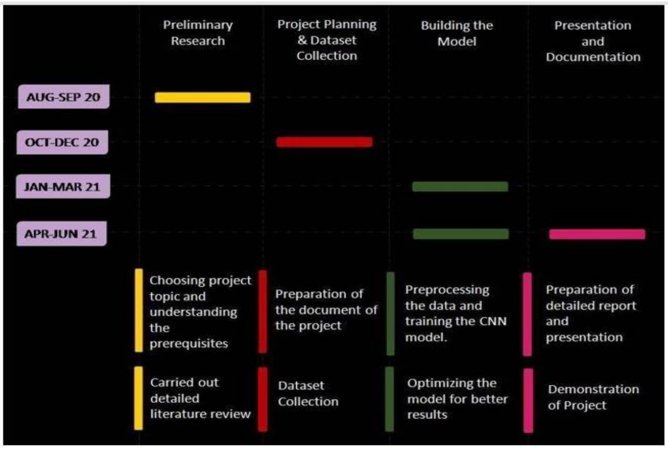

Figure 16: Planned chart.

4. Result and discussion

This section provides the details of the screenshot captured after the implementation and shows the performance analysis.

The employed dataset is an important part of any model is the dataset. The accuracy and the precision of a model can be determined by the robustness and the variety of data in the dataset. Data were recorded using manual driving of the car in simulator. Around 8000-14000 images were collected and its respective steering angle is recorded. The steering angle for the respective images were already stored in the csv file automatically as a driving log. Then randomly few images were preprocessed for bringing in the variety for the images and for the better accuracy for the model trained.

The summary of the model is as shown in the below figure which gives the information of the hidden layer.

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_10 (Conv2D)           (None, 31, 98, 24)        1824
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 14, 47, 36)        21636
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 5, 22, 48)         43248
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 3, 20, 64)         27712
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 1, 18, 64)         36928
_________________________________________________________________
flatten_2 (Flatten)          (None, 1152)              0
_________________________________________________________________
dense_8 (Dense)              (None, 100)               115300
_________________________________________________________________
dense_9 (Dense)              (None, 50)                5050
_________________________________________________________________
dense_10 (Dense)             (None, 10)                510
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 11
=================================================================
Total params: 252,219
Trainable params: 252,219
Non-trainable params: 0
```

Figure 17: Model summary.

4.1 Training and testing losses.

The trained final model is made to connect with the simulator and is made to run on autonomous mode. There are two different tracks, out of which one track is used for data collection and the other were not used for data collection, i.e. manual driving is done for first track and notfor second track during data collection. The model will perform well in the first track as it was the used for data collection. An efficient model should run properly in second track also, which the proposed model done effectively with least errors. The model proposed completes full round of both the tracks.

Losses of the training and validation is provided below, Loss is calculated per epoch while training the CNN model itself. There are two types of losses, one is training loss and another one is validation loss. There are various problems when it comes to training the model. The model should not be overfitted or underfitted, this can be evaluated by the loss obtained while training the model. If validation loss is more than the training loss thenthe model is said to be overfitted and vice versa is underfitted. In these both the conditions modelcannot predict accurately.

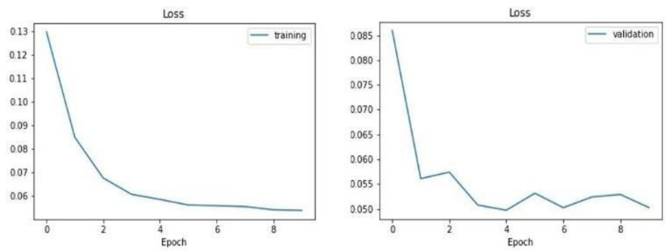

Figure 18: Validation and training losses.

If both the losses are almost equal or if the graph lines are converging then the model does not have overfitting or underfitting problem and it is perfect for predicting the required output. The comparison of the losses per epoch is shown in Figure.

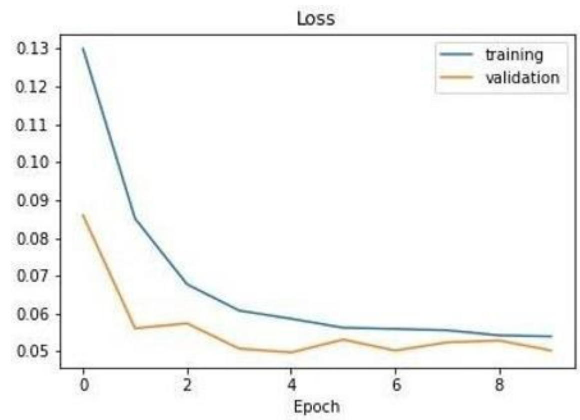

Figure 19: Losses of the model.

5. CONCLUSIONS

The proposed work proposed a CNN model that can predict the steering angle for the vehicle in the simulator to make it run safely on the road and in different tracks as well. The proposed model has a training loss of around 10.74% and validation loss of around 11.89%. And the model could drive the vehicle in the simulator effectively on both the tracks, even on the track on which it was not trained. This shows the model is accurate enough and it did not make any wrong predictions as well. Hence can say the model is accurate and efficient enough. The proposed work is of intermediate level between level 1 and level 2 of autonomous vehicle. Use of CNN made it still more efficient, since CNN is best for analyzing visual data. This proposed work can be implemented in real life scenarios for self-driving vehicles. This work will increase the efficiency of self-driving vehicles in terms of safety.

The future work includes, Since the proposed work is of intermediate level between level 1 and level 2 of autonomous vehicle, there is a huge room for future development and self-driving vehicles has more future scope as well. The advancement of the proposed work can be increased by adding other required features of self-driving vehicles like braking system, detecting the traffic signals, object detection in all sides, automatic speed control and other controls as well making it a complete autonomous to implement in real life self-driving vehicles.

REFERENCES

[1] S. Dara and P. Tumma, "Feature Extraction By Using Deep Learning: A Survey," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, 2018, pp. 1795-1801,doi: 10.1109/ICECA.2018.8474912.

[2] C E Nwankpa, W Ijomah, A Gachagan, and S Marshall, “Activation Functions: Comparison of Trends in Practice and Research for Deep Learning,”2018, doi: arXiv:1811.03378v1.

[3] Z. Wang, Y. Fan and H. Zhang, "Lane-line Detection Algorithm for Complex Road Based on OpenCV," 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 2019, pp.14041407,doi:10.1109/IMCEC46724.2019.8983919.

[4] Bojarski, Mariusz & Del Testa, Davide & Dworakowski, Daniel & Firner, Bernhard & Flepp, Beat & Goyal, Prasoon & D. Jackel, Lawrence & Monfort, Mathew & Muller, Urs& Zhang, Jiakai & Zhang, Xin & Zhao, Jake & Zieba, Karol. (2016). End to End Learning for Self-Driving Cars.

[5] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, 2017, pp. 1856-1860, doi: 10.1109/IVS.2017.7995975.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson,R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4):541–551, Winter 1989.

[7] LIU C,WANG Z Q.The Research on Advertising Model of Self-Driving Car Platform[C]//2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference(ITOEC2017),2017.