**STEERING ANGLE PREDICTION**

**IN**

**SELF DRIVING CARS**

**A project report submitted to the in partial fulfilment of the requirements for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted By**

J. Sai Vishnu Priya (15071A05D9)

P. Sai Tejaswi (15071A05I0)

V. Rashmika Reddy (15071A05H7)

K. Jaya Sri(15071A05E7)

**Under the Guidance Of**

Mrs. D. N. Vasundhara

**(**Assistant Professor, VNRVJIET)

****

**COMPUTER SCIENCE AND ENGINEERING DEPARTMENT**

**VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

(An Autonomous Institute, NAAC Accredited With ‘A++’ Grade, NBA Accredited, Approved by AICTE, New Delhi, Affiliated to JNTUH)

**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

(An Autonomous Institute)

**Hyderabad-500090**

****

**CERTIFICATE**

This is to certify that **J.Sai Vishnu Priya (15071A05D9), P.Sai Tejaswi (15071A05I0),V.Rashmika Reddy (15071A05H7), K. Jaya Sri (15071A05E7)** have successfully completed their project work at CSE Department of VNR VJIET, Hyderabad entitled **“STEERING ANGLE PREDICTION IN SELF DRIVING CARS”**in partial fulfilment of the requirements for the award of B. Tech degree during the academic year 2018-2019.

Project GuideHead of the Department

**Mrs. D. N. Vasundhara Mrs.B. V. Kiranmayee**

Associate Professor Associate Professor &

Department of CSE Head of the Department CSE

VNRVJIET VNRVJIET

**DECLARATION**

This is to certify that the project work entitled **“STERRING ANGLE PREDICTION IN SELF DRIVING CARS”** submitted in VNR Vignana Jyothi Institute of Engineering & Technologyin partial fulfilment of requirement for the award of Bachelor of Technology in ComputerScienceand Engineeringis a bonafide report of the work carried out by us under the guidance and supervision of Mrs. D. N.Vasundhara (Assistant Professor), Department of CSE, VNRVJIET. To the best of our knowledge, this report has not been submitted in any form to any university or institution for the award of any degree or diploma.

**J. Sai Vishnu Priya P. Sai Tejaswi V. Rashmika Reddy K. JayaSri**

(15071A05D9) (15071A05I0) (15071A05H7) (15071A05E7)

IV B.Tech-CSE, IV B.Tech-CSE, IV B.Tech-CSE, IV B.Tech-CSE,

VNRVJIET VNRVJIET VNRVJIET VNRVJIET

**ACKNOWLEDGEMENT**

Behind every achievement lies an unfathomable sea of gratitude to those who activated it, without it would ever never have come into existence. To them we lay the words of gratitude imprinting within us.

We are indebted to our venerable principal **Dr. C. D. Naidu** for this unflinching devotion, which lead us to complete this project. The support, encouragement given by him and his motivation lead us to complete this project.

We express our thanks to internal guide **Mrs. D. N. Vasundhara sq**and also Head of the department **Mrs. B. V. Kiranmayee** for having provided us a lot of facilities to undertake the project work and guide us to complete the project.

We express our sincere thanks to our faculty of the department of **ComputerScience and Engineering** and the remaining members of our college **VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY** who extendedtheir valuable support in helping us to complete the project in time w

J.Sai Vishnu Priya(15071A05D9)

P. Sai Tejaswi (15071A05I0)

V. Rashmika Reddy (15071A05H7)

K. Jaya Sri (15071A05E7)

**ABSTRACT**

“The recent trend of proliferation of road accidents and fatalities, calls for a methodology that can appease the burden on the drivers during certain crucial moments, requiring fast response and decision making. In recent years the development of self-driving cars has seen some great progress. More and more companies and organizations have vehicles out in traffic that partly steer themselves. However, the process of reaching a level where a car can drive satisfactory by its own is a long one and involves a lot of training. Until today, the most common approach is to expose the system to as many different situations as possible as it may encounter in reality. All in order to make the car as versatile as possible. The mathematical system responsible for the steering is called an artificial neural network. A much-wanted feature in such a system is the ability to generalize and use prior knowledge in new situations.”

“In our system, steering angle predictions were made using Convolutional neural network including drop out feature. The car takes images from a single front camera as input and produces the steering angle as output. The system automatically learns the internal representations of the important pre-processing steps such as features of the road like outline of the roads. Our system learns itself by minimizing the mean-squared error between steering angle output by network and steering angle of human driver.”

**INDEX**

Contents Page. No

**CHAPTER 1: INTRODUCTION** 1

1.1 Introduction to Neural Networks 1

1.2 Introduction to Convolution Neural Network 2

1.2.1 Drop-Out 3

1.3 Existing system 4

1.4 Proposed system 4

**CHAPTER 2: LITERATURE SURVEY** 5

2.1 Hough and Euclidean Transform 5

2.2 Convolutional Neural Network 6

2.3 Support Vector Machine

**CHAPTER 3: ALGORITHM DESCRIPTION** 8

3.1 t-distributed stochastic neighbour embedding 8

3.1.1 2D Example of t-SNE 9

3.1.2 Reusable t-distributed stochastic neighbour embedding 9

3.2 Support Vector Machine 9

3.2.1 Linear SVM Classifier 10

3.2.2 Non-Linear SVM Classifier 11

3.2.3 SVM Applications 12

3.2.4The Shortcoming of Train-Test Split 13

3.2.5The Cross-Validation Procedure 13

3.2.6Applications of Cross Validation 14

3.3 Logistic Regression 14

**CHAPTER 4: SYSTEM ANALYSIS** 17

4.1 System Requirements 17

4.1.1 Software Requirements 17

4.1.2 Hardware Requirements 18

4.2 System Architecture 18

**CHAPTER 5: SYSTEM DESIGN** 20

5.1 UML Diagrams 20

5.1.1 Usecase Diagram 21

5.1.2 Class Diagram 22

5.1.3 Sequence Diagram 24

5.1.4 Activity Diagram 25

**CHAPTER 6: IMPLEMENTATION** 28

6.1 Code for loading dataset and Importing modules 28

6.1.1 Import modules code 28

6.2 Code for performing basic statistics 29

6.3 Basic feature extraction (before cleaning) 30

6.3.1 Violin plot and distplot on word\_share 31

6.4 Text pre-processing 31

6.4.1 Importing modules required for text pre-processing 31

6.4.2 Pre-processing question1 and question2 32

6.5 Advanced feature extraction 32

6.5.1 get\_token\_features( ) function 33

6.5.2 get\_longest\_substr\_ratio() function 33

6.5.3 Pair plot on advanced features 34

6.5.4 Violin plot and distplot on token\_sort\_ratio 34

6.5.5 Performing t-SNE on advanced features and reducing them into 2 dimensions 35

6.6 Converting text data into tf-idf weighted word2vector 36

6.7 Logistic Regression 37

6.7.1 Importing modules required to perform logistic regression 37

6.7.2 Splitting the data into train and test data sets 37

6.7.3 Logistic Regression algorithm 38

6.8 Linear SVM 39

**CHAPTER 7:TESTING** 40

7.1 Testing plan 40

7.1.1 Test Data 40

7.1.2 Unit testing 40

7.1.3 Test Report 40

7.1.4 Error Report 40

**CHAPTER 8:CONCLUSION** 41

8.1 Conclusion 41

8.2 Future scope 41

**CHAPTER 9: BIBILOGRAPHY** 42

8.1 References 42

**LIST OF FIGURES**

Contents Page.No

**List of[\ Figures**

Fig 3.1: t-SNE dimensionality reduction 9

Fig 3.2: Linear SVM Classifier 10

Fig 3.3: Non-linear SVM 11

Fig 3.4: Cross Validation split 13

Fig 3.5: Logistic Regression 15

Fig 4.1 Python 17

Fig 4.2 System Architecture 18

Fig 5.1: Use Case Diagram for Developed Model 21

Fig 5.2: Class Diagram for Developed Model 23

Fig 5.3 Sequence Diagram for Developed Model 25

Fig 5.4 Activity Diagram 27

**CHAPTER 1**

**INTRODUCTION**

* 1. **Introduction to Neural Networks**

“Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. The neural network itself is not an algorithm, but rather a framework for many different machine learning algorithms to work together and process complex data inputs. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge about cats, for example, that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the learning material that they process.”

An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it.

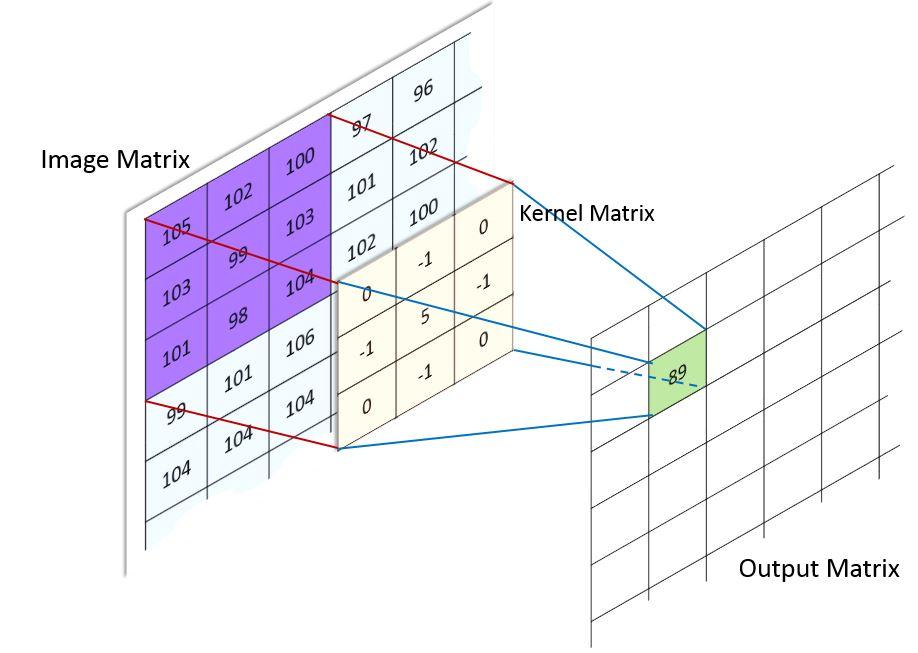
An Artificial Neural Network (ANN) is a data handling worldview that is motivated by the way organic sensory systems, for example, the mind, process data. The key component of this worldview is the novel structure of the data handling framework. It is made out of a substantial number of exceptionally interconnected preparing components (neurones) working as one to tackle explicit issues. ANNs, similar to individuals, learn by precedent. An ANN is designed for a particular application, for example, design acknowledgment or information characterization, through a learning procedure. Learning in natural frameworks includes changes in accordance with the synaptic associations that exist between the neurons. This is valid for ANNs too.

**1.2 Introduction to Convolution Neural Networks**

“A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of convolutional layers, RELU layer i.e. activation function, pooling layers, fully connected layers and normalization layers.”

**1.2.1 Convolution**

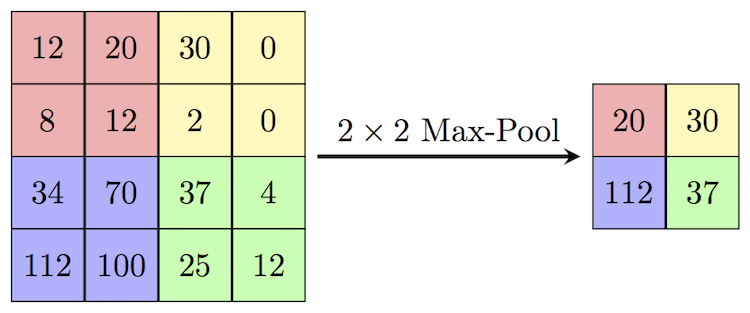
“Each convolutional neuron processes data only for its receptive field. Although fully connected feedforward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in a shallow architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a image of size 100 x 100 has 10000 weights for each neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. In this way, it resolves the vanishing or exploding gradients problem in training traditional multi-layer neural networks with many layers by using back propagation. In mathematics (and, in particular, functional analysis) convolution is a mathematical operation on two functions (f and g) to produce a third function that expresses how the shape of one is modified by the other. The term convolution alludes to both the outcome work and to the way toward figuring it. Convolution is like cross-relationship. Convolutional Neural Networks are fundamentally the same as normal Neural Networks. They are comprised of neurons that have learnable loads and predispositions. Every neuron gets a few sources of info, plays out a spot item and alternatively tails it with a non-linearity. The entire system still communicates a solitary differentiable score work: from the crude picture pixels toward one side to class scores at the other. Convolutional Neural Network designs make the unequivocal presumption that the sources of info are pictures, which enables us to encode certain properties into the engineering. These then make the forward capacity increasingly proficient to actualize and tremendously decrease the quantity of parameters in the system.



**Fig:1.1 convolution**

Convolutional layers apply a convolution operation to the input, passing the result to the next layer. The convolution emulates the response of an individual neuron to visual stimuli. Each convolutional neuron processes data only for its receptive field. Although fully connected feedforward neural networks can be used to learn features as well as classify data, it is not practical to apply this architecture to images. A very high number of neurons would be necessary, even in a shallow (opposite of deep) architecture, due to the very large input sizes associated with images, where each pixel is a relevant variable. For instance, a fully connected layer for a (small) image of size 100 x 100 has 10000 weights for each neuron in the second layer. The convolution operation brings a solution to this problem as it reduces the number of free parameters, allowing the network to be deeper with fewer parameters. For instance, regardless of image size, tiling regions of size 5 x 5, each with the same shared weights, requires only 25 learnable parameters. In this way, it resolves the vanishing or exploding gradients problem in training traditional multi-layer neural networks with many layers by using back propagation.

#### 1.2.2 Max Pooling

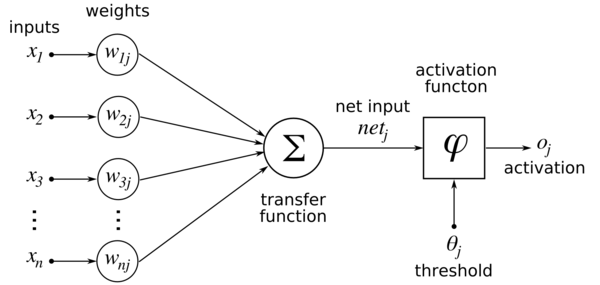


**Fig:1.2 Max Pooling**

Max pooling is an example based discretization process. The goal is to down-example an information portrayal (picture, shrouded layer yield lattice, and so on.), decreasing its dimensionality and taking into consideration suppositions to be made about highlights contained in the sub-locales binned. This is done to partially to help over-fitting by giving a preoccupied type of the portrayal. Also, it lessens the computational expense by decreasing the quantity of parameters to learn and gives fundamental interpretation invariance to the inward portrayal. Max pooling is finished by applying a maximum channel to (typically) non-covering sub areas of the underlying portrayal.

“Max pooling is done by applying a max filter to (usually) non-overlapping sub regions of the initial representation.”

#### 1.2.3 Activation Function

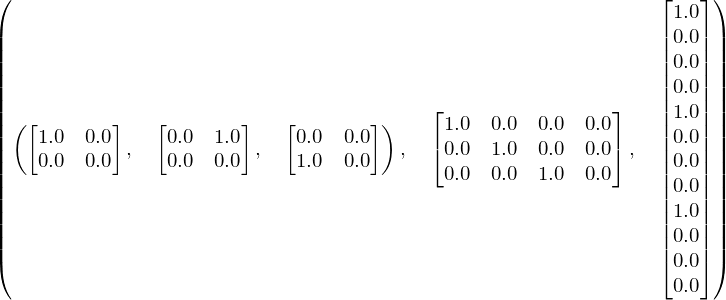


**Fig. 1.3 Activation**

Activation functions are really important for an Artificial Neural Network to learn and understand something truly convoluted and Non-straight complex practical mappings between the sources of info and reaction variable. They acquaint non-direct properties with our Network. Their fundamental intention is to change over an info flag of a hub in an A-NN to a yield flag. That yield flag currently is utilized as a contribution to the following layer in the stack.

#### 1.2.4 Flatten

It flattens the input by reducing it to lower dimensions.



**Fig:1.4 flatten**

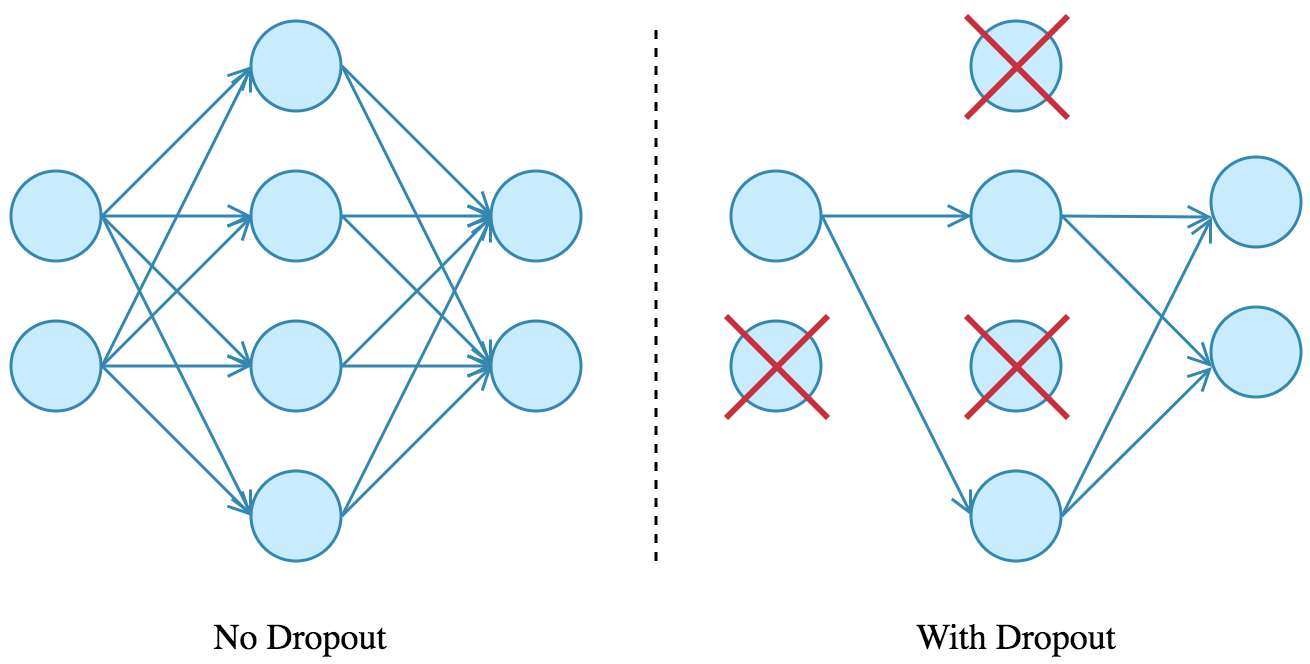
#### 

#### 1.2.5 Dense

A dense layer represents a matrix vector multiplication. “The values in the matrix are the trainable parameters which get updated during backpropagation. So, you get a m dimensional vector as output. A dense layer thus is used to change the dimensions of your vector. Mathematically speaking, it applies a rotation, scaling, translation transform to your vector.”

**1.2.2 Drop-Out**

**“**Dropout is a regularization technique for neural network models. It is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass. A dropout layer is used for regularization where you randomly set some of the dimensions of your input vector to be zero with probability given. A dropout layer does not have any trainable parameters i.e. nothing gets updated during backward pass of back propagation. Dropout consists in randomly setting a fraction rate of input units to 0 at each update during training time, which helps prevent over fitting.



**Fig:1.5 dropout**

”

**1.3 Existing System**

“However, the procedure impregnates the void of economic viability and efficiency in extracting the navigational features from the roads that can assist the vehicle automation. To tackle situations created by gradient technique, the algorithm uses Hough transform following a reduced correlation in the steer angle values. This is not an efficient technique.”

“Another algorithm detects cars on the road, apply a bounding box to the cars, and track their movement. In order to accomplish this task, the front-facing camera of a car is used to capture a video of the road which is then processed frame by frame by the algorithm to achieve the goal.”

**1.4Proposed System**

* “Our project uses **convolutional neural network (CNN, or ConvNet)** which is a class of deep learning neural networks, most commonly applied to analysing visual imagery. drop-out feature.”
* “In this drop-out feature is used.Dropout is a regularization technique for neural network models.It is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.”
* Finally, we get a steering angle with reduced overfitting and gives the output efficiently with reduced loss.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Hough and Euclidean Transform**

“The existing system uses an effective procedure for determining the steering angle required to keep the vehicle in the middle of the lane. The procedure used is divided in multiple stages. The first stage includes determination of the road’s midpoint using Robert edge detection and morphological operations. This midpoint is then used to determine the angle by which a vehicle must steer to maintain its position at the center of the road. It is done using Euclidean distance transform and trigonometric equations. The input data suffers non-linearity, irregular intensity, non-uniform contrast difference which makes edge extraction ineffective. To tackle such situations, Hough transform and virtual road edge was used to determine the road and lane edges.””

“To tackle situations created by gradient technique as discussed earlier, the algorithm uses Hough transform following a reduced correlation in the steer angle values. Moreover, the problem of invisibility of edges was solved by predicting the position of the second edge from the inclination angle and position parameters of the Standard Hough Transform (SHT) [4]. The derived ‘virtual edge’ (VE) provides a near to accurate estimation of the lane and has been made robust by training on several road conditions. The vehicle can be guided along the desired path even with partially visible lanes by the application of Euclidean distance transform. First, the video is captured using a conventional CCD digital camera mounted on the hood of the vehicle. The real time dynamic image is then processed in the computer placed in the vehicle, storing the direction parameters in a variable and indicating the deviation from midpoint via two LED indicators which resemble left and right. By default, the edge detection and Euclidean Distance Transform (EDT) are used to for steering angle estimation as the FPGA implementation of the algorithm renders a low time delay than Hough Transform. The angle variable that stores the steering angle data is monitored in a loop. Any discrepancies, with the variable that leads to a difference of more than 30o (determined experimentally) in steering angle, which can be considered unviable, channels the algorithm to Hough transform and ‘virtual edge’ concept. The algorithm being robust to low lighting and degraded weather conditions, provides a near to accurate prediction of the steering angle.

**2.2 Fully Connected Convolutional Neural Network**

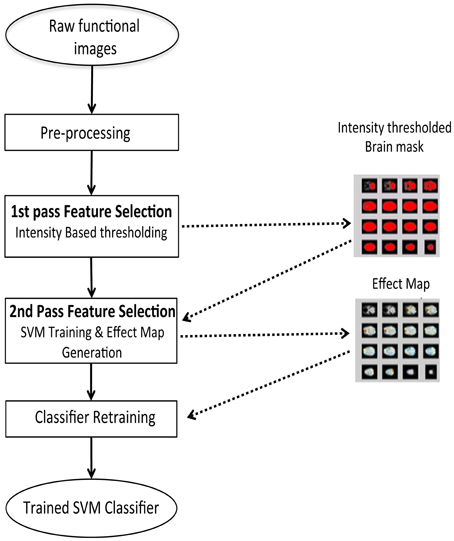
“In this system, pattern recognition tasks on visual input, e.g. images or video, are simplified by using convolutional neural networks (CNN), which are trained with back propagation. Hereby features are automatically learned based on the raw data. CNNs take spatial form of images into consideration and are usually used to recognize multiple features. Progressive steering algorithms nowadays are more often based on pattern recognition methods. For the project a model, consisting of several convolutional and fully connected layers, is used. The training was based on supervised learning. The objective is to solve a regression task. The architecture consists of several convolutional and subsampling layers, followed by ReLU activation functions after each convolutional layer. On the tail of the CNN are several fully connected layers. The last fully connected layer is activated with the tangens hyperbolicus function, which splashes the input data to a predicted steering value.



**2.3 Support Vector Machine**

An autonomous vehicle needs to know what it sees on the road (pedestrian, car, truck, animals etc) and classify it correctly to determine the appropriate action. The ability to classify what an autonomous vehicle sees on the road is achieved when it is trained to a point where it can accurately distinguish between cars, pedestrians, trucks, animals etc and classify them as such. This project focuses on training an autonomous vehicle to give it the ability to detect a car on the road or not. The goal for this project is to detect cars on the road, apply a bounding box to the cars, and track their movement. In order to accomplish this task, the front-facing camera of a car is used to capture a video of the road which is then processed frame by frame by the algorithm to achieve the goal. Perform a Histogram of Oriented Gradients (HOG) feature extraction on a labeled training set of images and train a Linear Support Vector Machine (SVM) classifier. Apply a color transform and combine binned color features, as well as histograms of color, to the HOG feature vector. Implement a sliding-window technique and use the trained classifier to search for vehicles in images. Run the pipeline on a video stream and create a heat map of recurring detection frame by frame to reject outliers and follow detected vehicles. Estimate a bounding box for vehicles detected.

Although the SVM classifier method works for this test case and is a great way to understand the concepts on what it takes to detect and track vehicles, it is also quite slow and computationally expensive. As a result, this may not be applicable in real world situations where there are many different types of vehicles, pedestrians, bicycles, and oncoming vehicles etc. Another issue is that a lot more data is needed to train the SVM to avoid false positives and that would make the current approach even slower.



**CHAPTER 3**

**ALGORITHM DESCRIPTION**

**CONVOLUTION NEURAL NETWORKS**

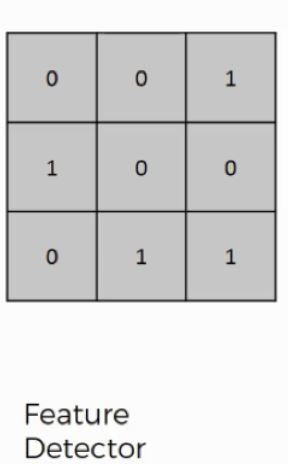
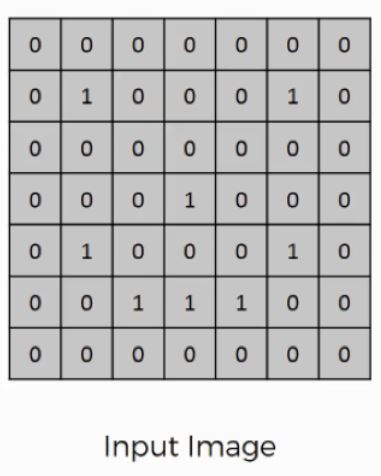
In general, the working principle involves Convolution neural networks. It consists of five basic functionalities they are:

1. Convolution
2. Subsampling
3. Activation(ReLu)
4. Partially connected using drop-out
5. Loss

**3.1.1 CONVOLUTION**

“In purely mathematical terms, convolution is a function derived from two given functions by integration which expresses how the shape of one is modified by the other.”

**3.1.2 THE CONVOLUTION OPERATION**



Here are the three elements that enter into the convolution operation:

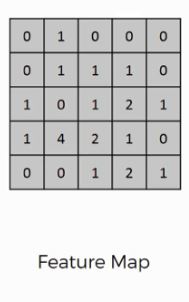
* Input image
* Feature detector
* Feature map

“The input image is pattern of the 1's and 0's.Sometimes a 5×5 or a 7×7 matrix is used as a feature detector, but the more conventional one, and that is the one that we will be working with, is a 3×3 matrix. The feature detector is often referred to as a “kernel” or a “filter”.”

**3.1.3 WORKING OF CONVOLUTION OPERATION**

You can think of the feature detector as a window consisting of 9 (3×3) cells. Here is what you do with it:

* “Place it over the input image beginning from the top-left corner within the borders you see demarcated above, and then you count the number of cells in which the feature detector matches the input image.”
* “The number of matching cells is then inserted in the top-left cell of the feature map.”
* “You then move the feature detector one cell to the right and do the same thing. This movement is called a and since we are moving the feature detector one cell at time, that would be called a stride of one pixel.”
* “In this example the feature detector's middle-left cell with the number 1 inside it matches the cell that it is standing over inside the input image. That's the only matching cell, and so you write “1” in the next cell in the feature map, and so on and so forth.”
* “After you have gone through the whole first row, you can then move it over to the next row and go through the same process.”
* “The cells of the feature map can contain any digit, not only 1's and 0's.”
* “After going over every pixel in the input image in the example above, we would end up with these results:”



**3.1.4 WHY DO WE USE CONVOLUTION OPERATION**

“There are several uses that we gain from deriving a feature map. These are the most important of them: Reducing the size of the input image, and you should know that the larger your strides (the movements across pixels), the smaller your feature map.”

“When dealing with proper images, you will find it necessary to widen your strides. Here we were dealing with a 7×7 input image after all, but real images tend to be substantially larger and more complex.”

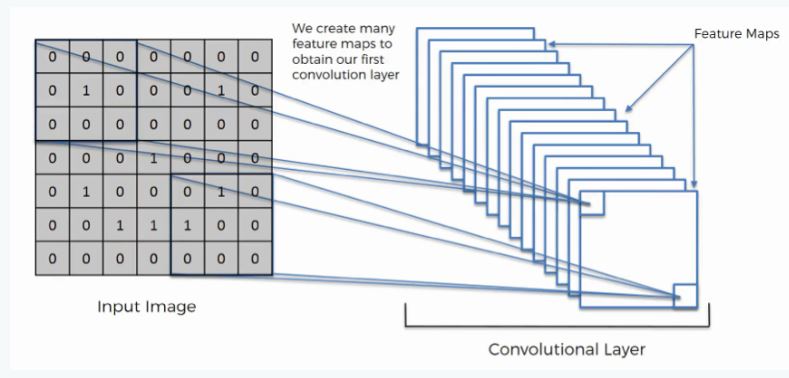
**3.1.5 DO WE LOSE INFORMATION ON USING FEATURE DETECTOR**

“Yes, the feature map that we end up with has fewer cells and therefore less information than the original input image. However, the very purpose of the feature detector is to sift through the information in the input image and filter the parts that are integral to it and exclude the rest.”

**3.1.6 OPERATION BY CNN**

“Through training, the network determines what features it finds important in order for it to be able to scan images and categorize them more accurately.”

“Based on that, it develops its feature detectors. In many cases, the features considered by the network will be unnoticeable to the human eye, which is exactly why convolutional neural networks are so amazingly useful. With enough training, they can go light years ahead of us in terms of image processing.”



**3.2. SUBSAMPLING**

“A pooling or subsampling is a convolution layer in CNN. The primary function of a pooling layer is to reduce the number of parameters which are to be learned by the network. It also has the additional effect of reducing overfitting and thereby increasing the overall performance and accuracy of the network.”

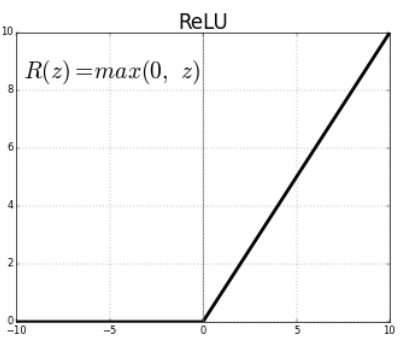
But for our problem statement, we are not imparting any pooling process.

**3.3 ACTIVATION UNIT (ReLU)**

“Activation functions are really important for a Convolution Neural Network in order to learn and make sense of something really complicated mappings between the inputs and response variable. The main purpose is to convert an input signal of a node in a CNN to an output signal. Mathematically, it is defined as y = max(0, x).”

“ReLU stands Rectified linear unit.This is the most used activation function in Convolution neural networks.ReLU is linear for all positive values, and zero for all negative values.This step is done after generating a feature map from the earlier convolution step.”

“The Rectified Linear Unit, or ReLU, is not a separate component of the convolutional neural networks' process.The ReLU is the most used activation function in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning.”



“ReLU is half rectified (from bottom). f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.”

**Range:**[ 0 to infinity)

The function and its derivative **both are** **monotonic**.

“But the issue is that all the negative values become zero immediately which decreases the ability of the model to fit or train from the data properly. That means any negative input given to the ReLU activation function turns the value into zero immediately in the graph, which in turns affects the resulting graph by not mapping the negative values appropriately.”

**3.4 PARTIALLY CONNECTED USING DROP-OUT**

“In general, CNNs are fully connected, but here we are using a special feature called drop-out feature. Drop-out is a kind of mechanism i.e. while choosing the output of the preceding neurons instead of selecting all neurons, it drops certain neurons and select remaining based on the probability given.

So in our problem statement, we are giving certain probability i.e., 0.8 which means it considers 80% of the neurons in choosing the output of the preceding neurons and take them for further process.”

## 3.4.1 DROP OUT FOR NEURAL NETWORK

“Dropout is a regularization technique for neural network models proposed by Srivastava, et al. in their 2014 paper.”

“Dropout is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.”

“As a neural network learns, neuron weights settle into their context within the network. Weights of neurons are tuned for specific features providing some specialization. Neighbouring neurons become to rely on this specialization, which if taken too far can result in a fragile model too specialized to the training data. This reliant on context for a neuron during training is referred to complex co-adaptations.”

“We can imagine that if neurons are randomly dropped out of the network during training, that other neurons will have to step in and handle the representation required to make predictions for the missing neurons. This is believed to result in multiple independent internal representations being learned by the network.”

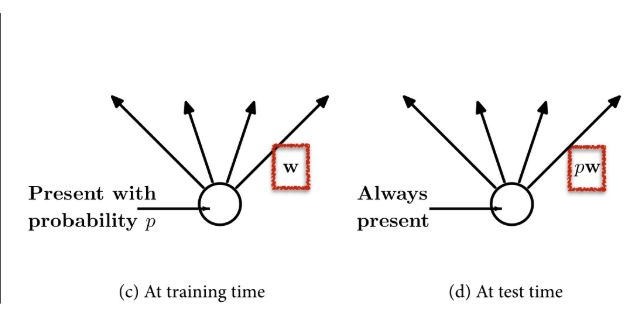
“The effect is that the network becomes less sensitive to the specific weights of neurons. This in turn results in a network that is capable of better generalization and is less likely to overfit the training data.”

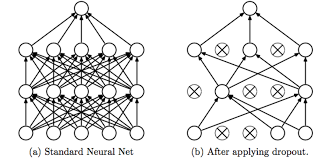
#### Training Phase:

“Training Phase: For each hidden layer, for each training sample, for each iteration, ignore (zero out) a random fraction, p, of nodes (and corresponding activations).”

#### ****Testing Phase:****

Use all activations, but reduce them by a factor p (to account for the missing activations during training).





**3.5 LOSS**

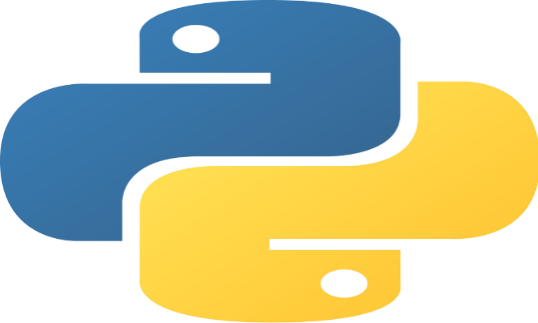
“A loss is a mathematical step which tells how accurate is our algorithm. It is taking the difference between predicted and actual values. It must be approximately equal to zero. The more it is closer to zero, the greater is the accuracy. So according to our problem statement, we initially checked the accuracy which was 0.5 when we ran, with 15 epochs, but later the epochs are increased to 30 which gave the accuracy of 0.1. Hence, we consider the algorithm best which obtained greater accuracy.”

**4.1 System Requirements**

**4.1.1 Software Requirements**

**a) Python**

Python is a deciphered language by Guido van Rossum, Python has a diagram hypothesis that complements code decipherability, and a sentence structure that allows programming architects to exhibit their thoughts in fewer lines of code noticeably using imperative whitespace. It has inbuilt libraries and is readily available to use just by importing. Incorporates a kind modified organization. It is a simple language that doesn’t take much effort to write the code.



**Fig 4.1**Python

1. **Sublime editor**

        The Sublime Text editor is a proprietary cross-platform source code editor with a Python application programming interface. It usually supports many programming languages and markup languages, and functions can be added by users with plugins, maintained under free-software licenses.

            The following is a list of features of the sublime editor

* "Goto Anything," it is quick navigation to files, symbols, or lines
* A "Command palette" where it used for adaptive matching for quick keyboard invocation of arbitrary commands
* Simultaneous editing: We can simultaneously make the same interactive changes to multiple selected areas
* Python-based plugin API
* We can also set the project-specific preferences
* Extensive customizability via JSON settings files, including project-specific and platform-specific settings
* Cross-platform (Windows, macOS, and Linux) and Supportive Plugins for cross-platform
* Compatible with many language grammars from Text Mate

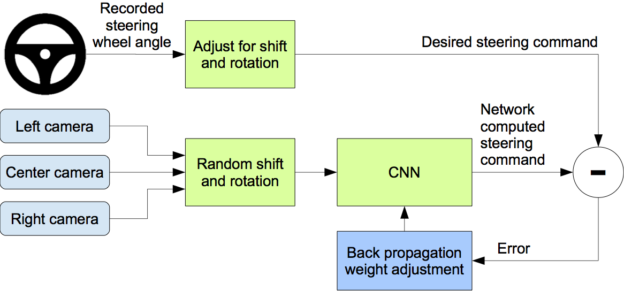
**4.1.2 Hardware Requirements**

RAM: 4GB and Higher

Processor: Intel i3 and above

Hard Disk: 10GB: Minimum

* 1. **System Architecture**



**Fig 4.2** System Architecture

**a) Importing Dataset**

In this, we need to Load the dataset (images). which contains the sequence of road images for predicting the output.

**b) Loading the Dataset**

Loading the Dataset include the following

* Defining the function to load the sequence of images.
* Defining the functions to divide the images into batches
* Dividing the dataset into train and test dataset.

**c) Building the CNN model**

Here we build the model by initializing the weights and steering angle using the rectified linear activation function. In this, the function is defined to build the model where the model contains five convolution layers.

**d) Train the dataset**

After defining the functions to divide the images into train and test dataset, into batches we train the model through which we build the model earlier.

**f) Test dataset**

We test the data over the trained model it predicts whether the steering angle and also displays the actual steering angle

**g) Display results**

The results are displayed by a video showing how the steering angle is turned by using the predicted value and simultaneously an output is shown on the console about the predicted and actual steering angle.

**CHAPTER 5**

**SOFTWARE DESIGN**

**5.1 UML Diagrams**

Unified Modeling Language is a tool that helps a designer to show his ideas about the project to his client and developer. Modeling plays a crucial role in designing software. A poorly designed model can lead to poorly developed software.

A UML system has using five different views that help in describing systems from different perspectives. Each view has a set of diagrams and components that represent real time objects.

* 1. **User Model View:**

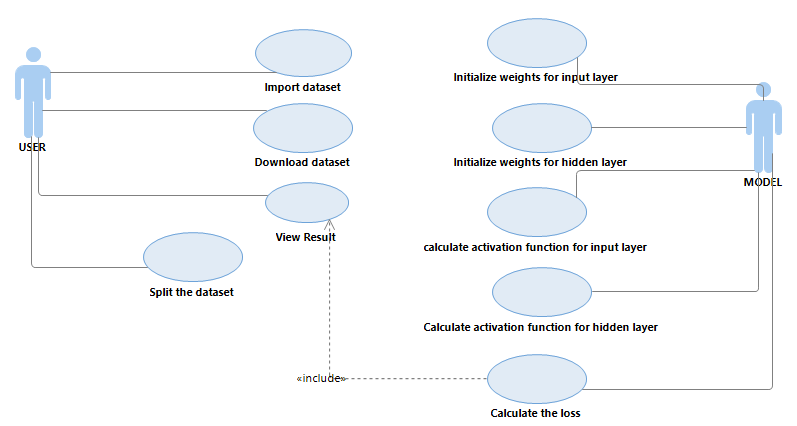
1. It models user behavior in a system context.
2. All the diagrams are drawn keeping in mind the user’s response and reaction towards a system.
   1. **Structural Model View**
3. This view consists of a class diagram and object diagram which is used to model the static structures.
4. It uses objects, attributes, operations, and relationships.
   1. **Behavioural Model View**
5. It mainly consists of the sequence diagram, collaboration diagram, statechart diagram, and activity diagram. They mainly represent the flow of actions between different objects involved in the system
6. They are used to visualize various dynamic aspects of the system architecture.
   1. **Implementation Model View**
7. This view consists of component diagrams and deployment diagrams. This view models the static software modules for an organization.
8. This usually contains the data files, documentation, the executables and source code.
9. These are the physically replaceable components of the system. They are modeled using component diagrams.

**5.1.1 Use Case Diagram**

The basic representation for the interaction of the user with the system is represented using the use case diagram. It involves the relationship between the user and various use cases with the actors being involved. There are different kinds of relationships that are involved between the use cases and the actors. They include:

1. Association relationship
2. Generalization
3. Dependency
4. Realizations
5. Transitions

The following represents the use case diagram of the proposed system:



**Fig 5.1:** Use Case Diagram for Developed Model

**5.2 Class Diagram**

They are a static representation of an application. Only the class diagrams have the capability to be directly mapped with the OOP Languages because in OOPs everything is model in the form of classes and objects. Because of this reason these diagrams are used widely at the time of construction. This is one of the most popularly used UML diagrams in the designer community. A class diagram plays an essential role in forward and reverse engineering.

1. It acts as a base for the component and deployment diagrams.
2. It mainly describes and defines the basic responsibilities of a system’s application.
3. It implements the analysis and design view for a static application.

In a class diagram, each object is modeled as a class. Each class consists of a section or compartments.

1. Class name
2. Attributes of a class or operations
3. Methods or functions
4. Documentation (optional section)

The following points ought to be recollected while drawing a class diagram:

1. The name of the class diagram must be meaningful to portray the aspect of the framework.
2. Each component and its connections must be distinguished ahead of time.
3. Each class has a responsibility (attributes and methods) that must be identified clearly.
4. A number of properties for each class must be minimum. Since pointless properties will make the diagram convoluted.
5. At whatever point required to depict some part of the diagram use notes Since toward the finish of the diagram it must be justifiable to the designer/coder.
6. Before finalizing the last version, the diagram must be drawn on plain paper and revise whatever number circumstances as would be prudent to make it redress.

1.**Scopes:**

The UML diagrams have two different types of scopes for class members:

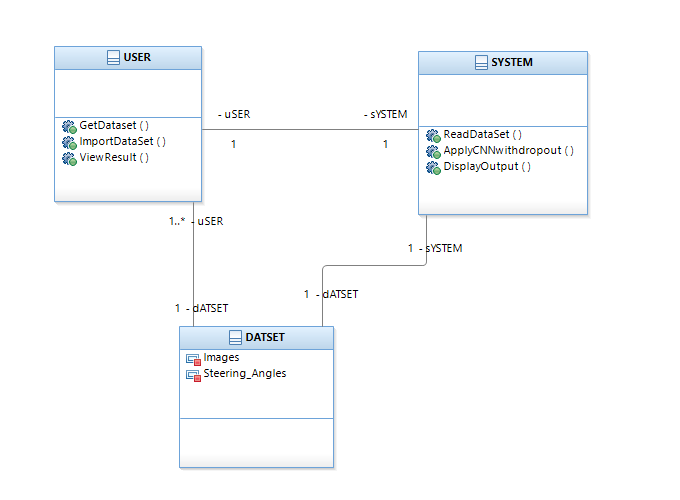
1. instance members scope and
2. classifier members scope

2. **Classifier members** are “static” members of a class in many programming languages. The scope is the class itself.

1. Static attributes are common to all other objects that invoke the class.
2. Static methods are not instantiated.

3. **Instance members** are nothing but the members that are local to an object.

1. The main purpose of instance members is to allow the objects to store their states.
2. Declarations outside the methods are usually known as instance members.



**Fig 5.2:** Class Diagram for Developed Model

**5.3 Sequence Diagram**

The Sequence Diagram depicts the time sequence among various objects in an application. It depicts the sequence of messages with which objects communicate with each other so that they carry out the required functionality.

It consists of the lifelines which are usually parallel vertical lines. It consists of horizontal arrows which indicate the direction of the messages that are exchanged in a proper order which makes the user easy to understand.

The lifeline for a given object represents a role. The synchronous calls are represented with the help of a solid arrow head whereas the asynchronous messages are represented with the help of open arrow heads.

All objects are represented according to their time ordering. The timing of messages plays a major role in sequence diagrams. An object is killed immediately after its use in sequence diagrams.

**I). Common Properties**:

An arrangement graph is much the same as a unique sort of diagram and offers some indistinguishable properties from other diagrams. In any case, it varies from every single other diagram in its content.

**II)**. **Contents**

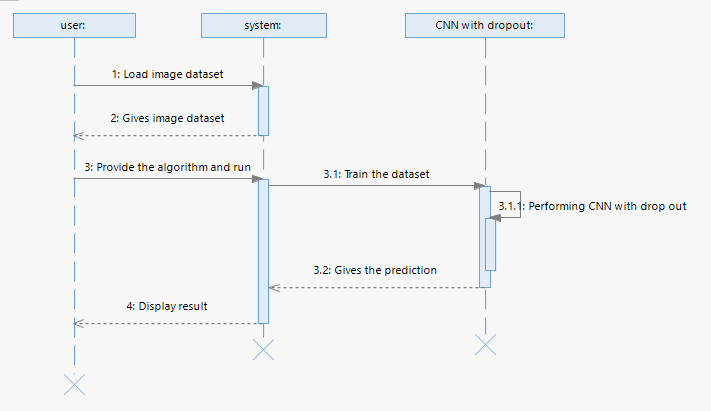
Objects are normally named or unknown instances of the class, however, may likewise speak to occurrences of different things, for example, components, collaboration, and nodes. Graphically, the object is represented as a rectangle by underlying its name.

**III). Links**

A link is a semantic association among objects i.e., an object of affiliation is called a connection. It is represented as a line.

**IV). Messages**

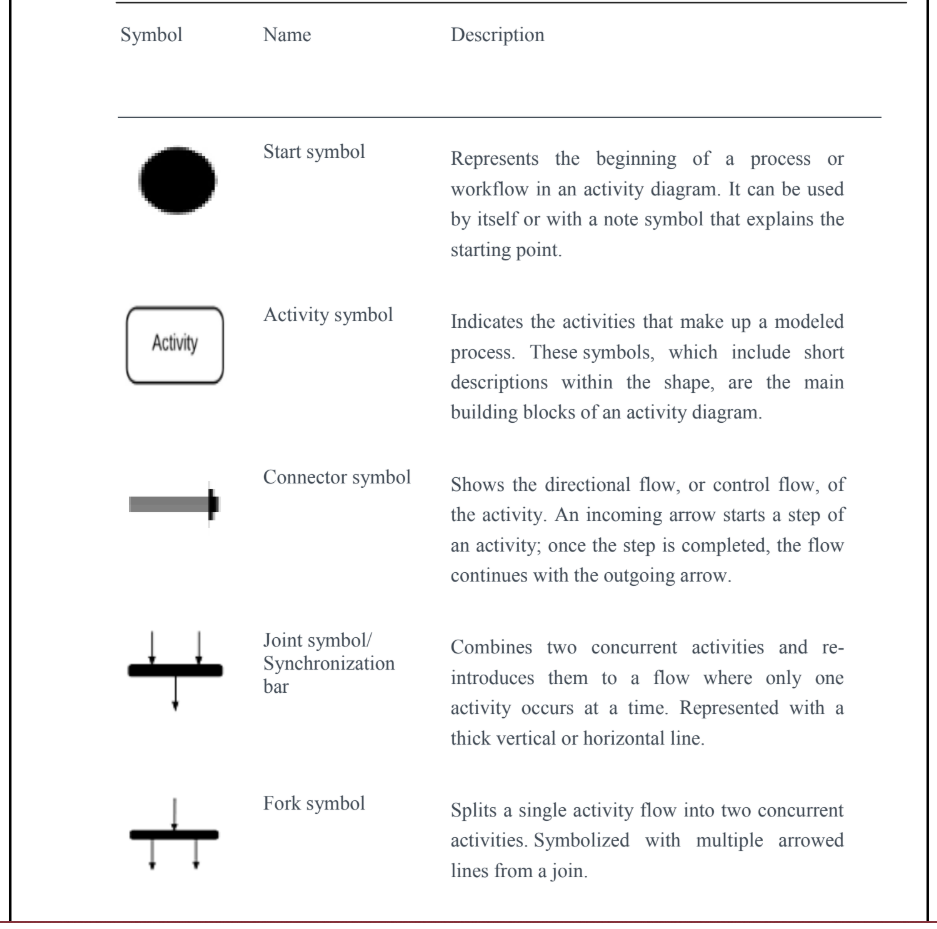
A message is a determination of a correspondence between objects that passes on the data with the desire that the action will follow.



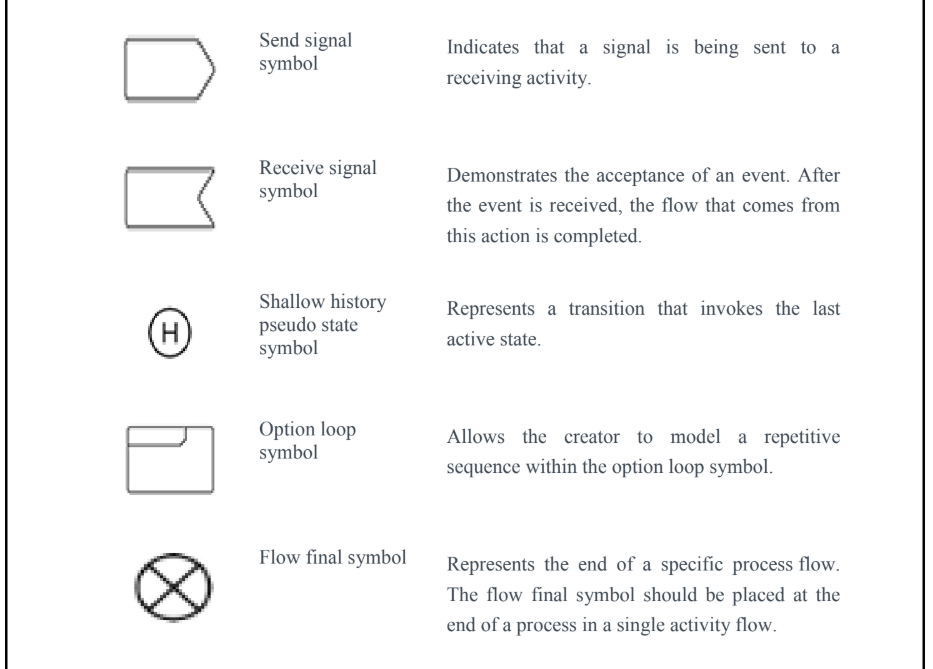
**Fig 5.3**Sequence Diagram for Developed Model

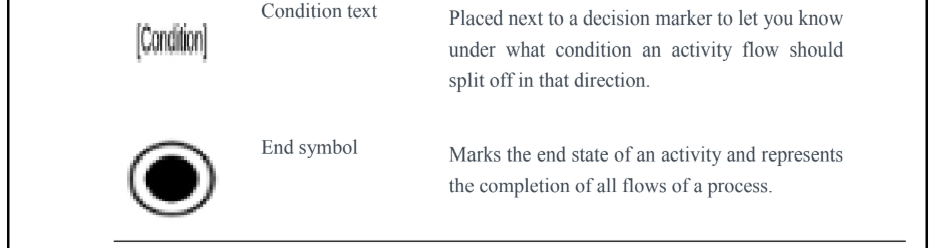
* 1. **Activity Diagram**

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another, but activity diagram is used to show message flow from one activity to another.









The basic purpose of an activity diagram is the same as that of other UML diagrams. The dynamic behavior of the system is viewed by the activity diagram. They are used to construct a system using the backward and forward engineering mechanisms.

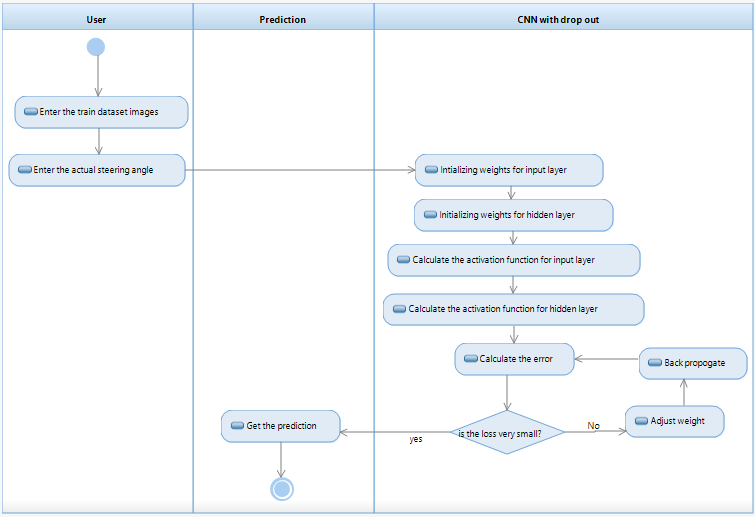
The purpose of an activity diagram is as follows:

1. For drawing the flow (i.e. activity) ina system.
2. For showing the flow of sequence from one activity to another activity.
3. For showing the concurrent and parallel flow of actions in the system. The elements that are used in an activity diagram are as follows:

i)Association relationship

ii)Activities

iii)Conditions and Constraints.



**Fig 5.4 Activity Diagram**

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 Driving dataset**

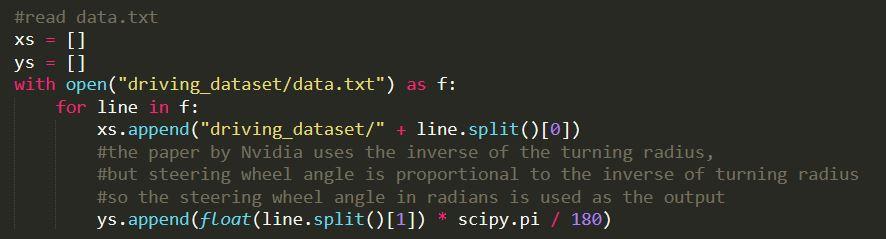
**6.1.1 Code for importing modules**

Syntax for importing modules are

**https://lh3.googleusercontent.com/Idt_4spWs0iwKMWd1ERoTY-GmAM6sh1sQmjOEvLCyhOSt_cSEVcaTfZB7sFU09lorVBERrVIrvxtKliy2oxF01a6Vzc5T1xUHpNruScen8D56PNYu7DbanuigeG_C239hMLaFgF3THW66n-S8A**

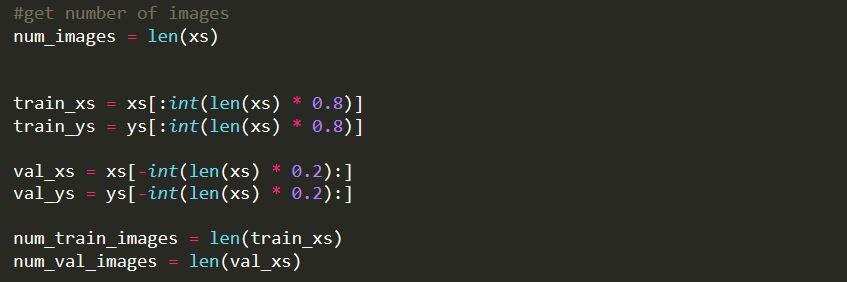
**6.1.2 Code for reading and loading dataset**

We are trying to load the data.txt file which contains our images and steering angle corresponding to that into the editor

****

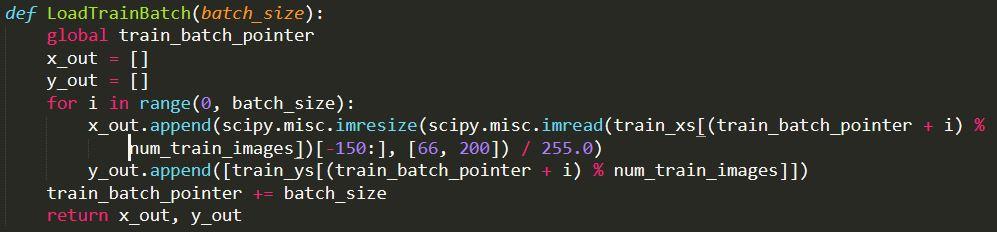
**6.1.3 Splitting the data into train and test**

The next step after loading dataset is to divide that into train and test. In our algorithm we are dividing 80% of dataset into train data and remaining 20% for testing.

****

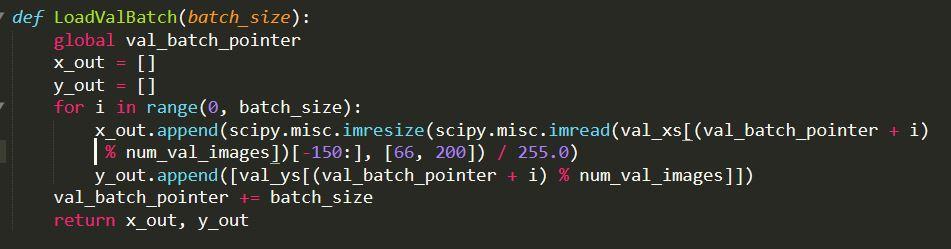
**6.1.4 Function for loading the training batch data**

In CNN we use batches to train i.e. the number of examples we are using in one iteration. In our algorithm there is a function called LoadTrainBatch() which is used to return the mini-batch train data which is greater than 1 but less than total dataset size by taking the batch size as input.

****

**6.1.5 Function for loading the test batch data**

In CNN we use batches to train i.e. the number of examples we are using in one iteration. In our algorithm there is a function called LoadTestBatch() which is used to return the mini-batch test data which is greater than 1 but less than total dataset size by taking the batch size as input.

****

**6.2 Building the Model**

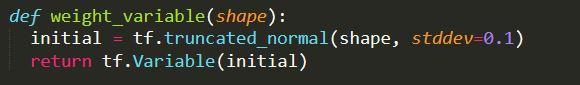
**6.2.1 Importing modules**

Syntax for importing the required modules that are required in building the model.

**https://lh5.googleusercontent.com/dZCSJcpvFVBQEJsHcdavVNgdTJOKaREtBRNYxJGFFHeyuAJDygsYO0w2zAoDPV_lI_nVdCwa0WCycv9bVvKYrnxyr4-zlMO-PmLpiCqExr-X_GGSGx_ZlxLvqy4usOMEbQtRHEFCiyC7lBJsdg**

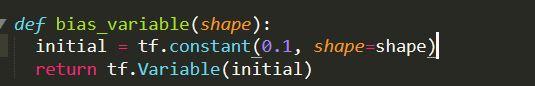
**6.2.1 Function for initializing weight variables**

In building the model we used a function call weight\_variable() which is used to assign weights randomly from the normal distribution given the shape as input.

****

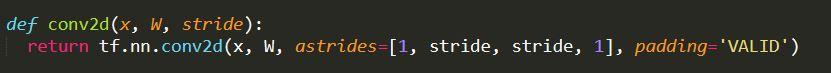
**6.2.2 Function for initializing bias variables**

In building the model we used a function call bias\_variable() which is used to assign a constant value of 0.1 as bias given the shape as input.

****

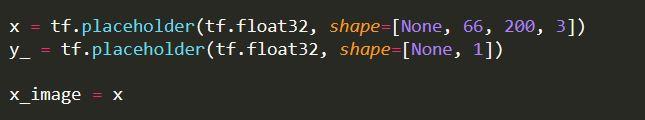
**6.2.3 conv2d Function**

In building the model we used another function call conv2d() which is responsible for performing 2D convolution

****

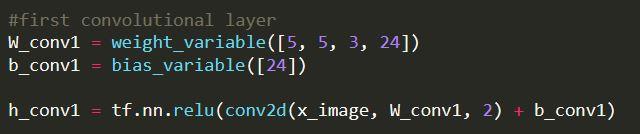
**6.2.4 Declaring placeholders for holding input images and output steering angle**

We used placeholder for holding input image and for holding steering angle

****

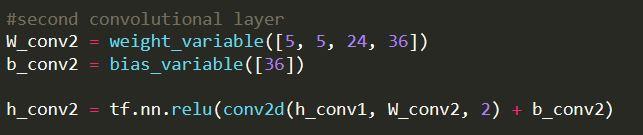
**6.2.5 Code for first convolution layer**

In the first convolution layer the size of kernel is 5\*5 and 3 is the parameter which matches with 3rd parameter in input image and 24 is the number of kernels used.

****

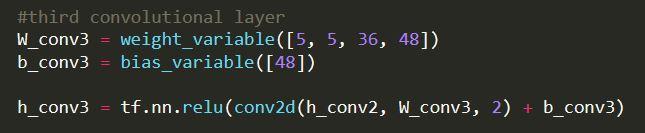
**6.2.6 Code for second convolution layer**

In the second convolution layer the size of kernel is 5\*5 and 24 is the parameter which matches with no of kernels in first convolution layer and 36 is the number of kernels used.

****

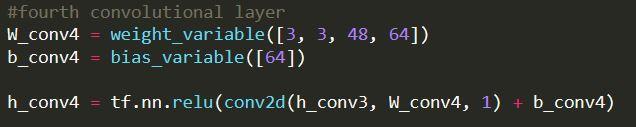
**6.2.7 Code for third convolution layer**

In the third convolution layer the size of kernel is 5\*5 and 36 is the parameter which matches with no of kernels in second convolution layer and 48 is the number of kernels used.

****

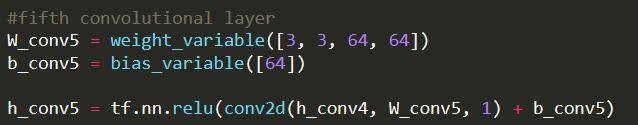
**6.2.8 Code for fourth convolution layer**

In the fourth convolution layer the size of kernel is 3\*3 and 48 is the parameter which matches with no of kernels in third convolution layer and 64 is the number of kernels used.

****

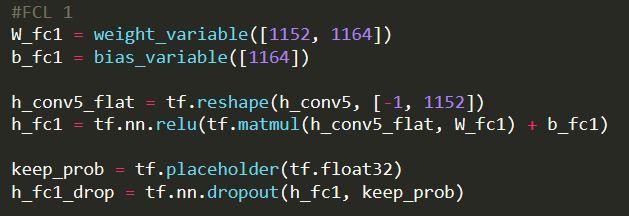
**6.2.9 Code for fifth convolution layer**

In the fifth convolution layer the size of kernel is 3\*3 and 64 is the parameter which matches with no of kernels in third convolution layer and 64 is the number of kernels used.

****

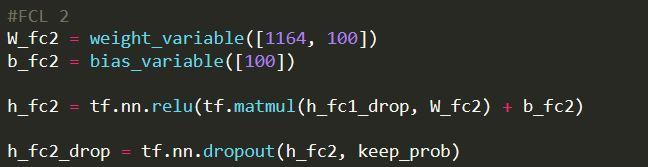
**6.2.10 Code for fully connected layer – 1**

Before going to fully connected layer – 1 we are trying to flatten the output of previous convolution layer then we get size of 1152 as input. In the first fully connected layer we are trying to have 1164 neurons. Then we are performing matmul() function on this, followed by activation function ReLU. In this layer we are performing drop out in order to reduce overfitting with keep probability of 0.8 which means we are retaining 80% of neurons and removing 20% of neurons.

****

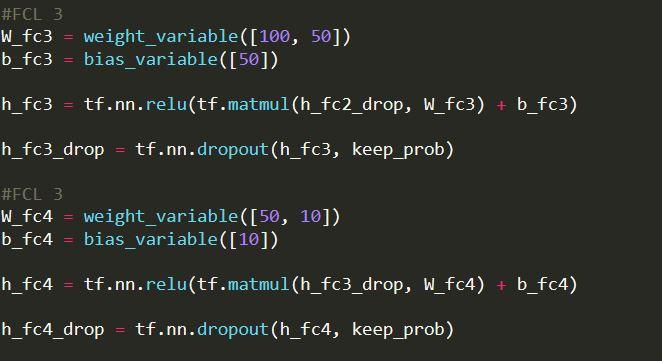
**6.2.11 Code for fully connected layer – 2**

The output of first convolution layer is of size 1164. In the second fully connected layer we are trying to have 100 neurons. Then we are performing matmul() function on this, followed by activation function ReLU. In this layer we are performing drop out in order to reduce overfitting with keep probability of 0.8 which means we are retaining 80% of neurons and removing 20% of neurons.

****

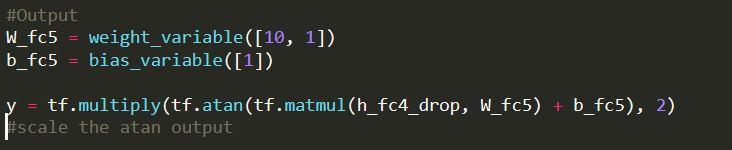
**6.2.12 Code for fully connected layer – 3**

The output of second convolution layer is of size 100. In the third fully connected layer we are trying to have 50 neurons. Then we are performing matmul() function on this, followed by activation function ReLU. In this layer we are performing drop out in order to reduce overfitting with keep probability of 0.8 which means we are retaining 80% of neurons and removing 20% of neurons. We are again trying to have one more convolution layer which has 10 neurons, we has input of size 50.This is again going to perform again matmul() followed by activation function to reduce overfitting. We also perform dropout with same keep probability 0.8 that means to retain 80% of neurons and drop 20% of neurons.

****

**6.2.13 Code for output layer**

Finally, in the output layer we are trying to have 1 neuron which predicts the steering angle.

****

**6.3 Training the model**

**6.3.1 Defining variables**

We are defining variables required to train the model. The first important variable is loss. In our project we are trying to define loss and sum of mean square error between predicted, actual value and L2 normalization with normalization constant as 0.001. We are defining a variable LOGDIR which tells the directory in which the output of model has to be saved and also log\_path which tells where logs have to be saved. We are then creating a summary to monitor the cost tensor and we are merging all summaries into a single operation. We are also using InteractiveSession from tensor flow  that installs as default session on construction. Interactive session supports less typing in a way that we can run variables without constantly referring to session object.

****

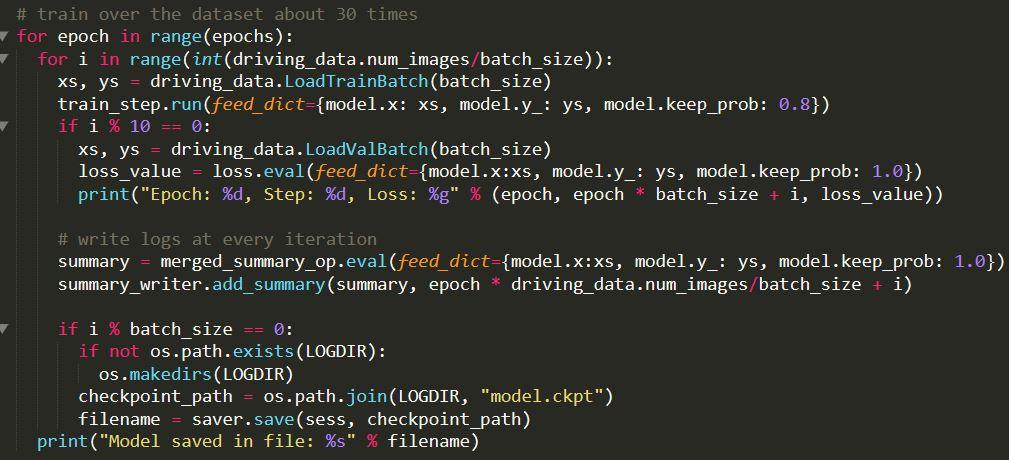
**6.3.2 Defining number of epochs and batch size**

We are giving number of epochs as 30 which means we are passing entire dataset forward and backward through the neural network 30 times. We are using batch size 100 which means we are not passing the entire dataset into CNN we are diving the dataset into batches of 100 and are passing through CNN.

**https://lh6.googleusercontent.com/eFWVhs4EfcDy4W-2bT76STMUvuggq23hI1aj1zVCiW8MTN_6ZwwHIBBBAH-of2SfNhiJWeLCUz_P5KgggHU8vh-WHN2KdsoXOqzWJ_778L218korqxICcnTDwopafnba8CUWe91A2wMXX9s7Zg**

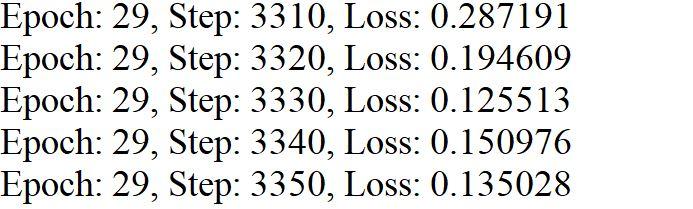
**6.3.3 Code for training the model**

In the below code we are having 2 for loops, one for running the model for 30 epochs and another for sending batch size of 100 each time through the CNN. After 10 iterations we are trying to calculate the loss on test data and save the details the epoch number, step number and loss in a file.

****

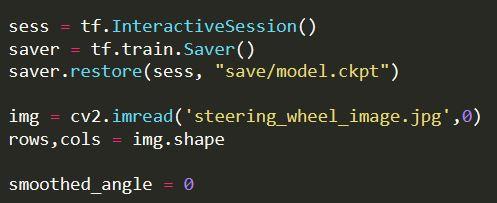
**Output:**

Finally at the end of training we are getting a loss of approximately 0.13

****

**6.4 Running the dataset**

**6.4.1 Defining the variables**

****

**6.4.2 Code for displaying sequence of images as video and steering wheel movement**

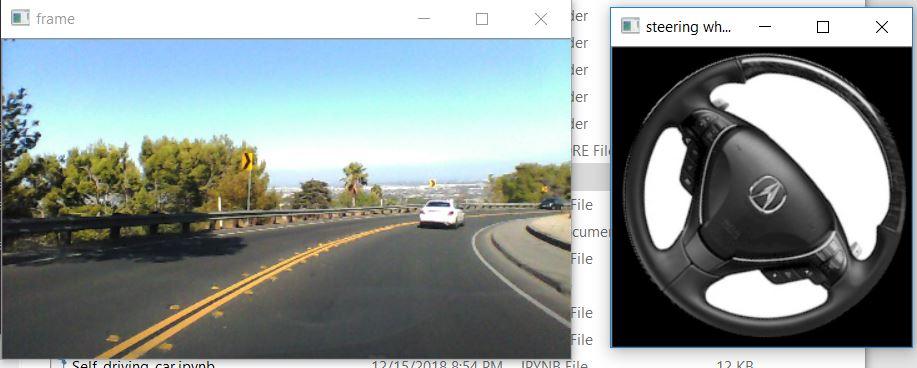
****

**6.4.3 Code for displaying the actual steering angle and steering angle predicted by model**

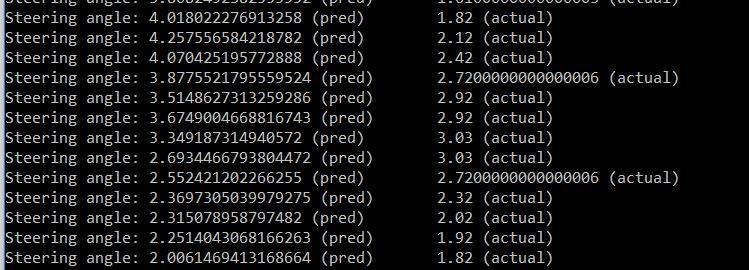
****

**Output:**

Image shows the image and steering wheel rotation predicted by model for that corresponding image.

****

Below image shows actual steering angle and predicted steering angle given by model.

****

**CHAPTER 7**

**TESTING**

**7.1 TESTING PLAN**

Software testing is one of the main stages of project development life cycle to provide our cessation utilizer with information about the quality of the application and ours, in our Project we have under gone some stages of testing like unit testing where it’s done in development stage of the project when we are in implementation of the application after the Project is yare we have done manual testing with different Case of all the different modules in the application we have even done browser compatibility testing in different web browsers in market, even we have done Client side validation testing on our application

1. **Unit testing**

The unit testing is done in the stage of implementation of the project only the error are solved in development stage some of the error we come across in development are given below

1. **Manual Testing**

As our Project is academic Leave, we can do any automatic testing so we follow manual testing by endeavour and error methods

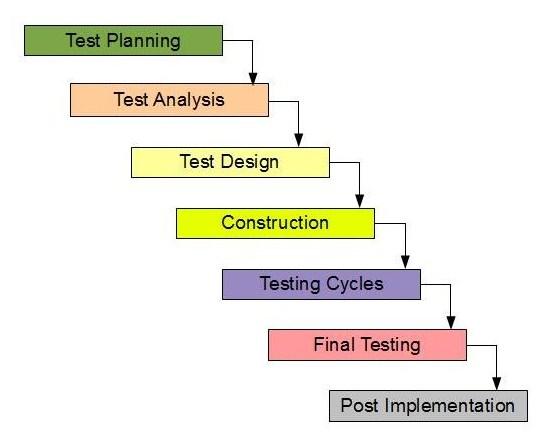
1. **Deployment of System**

Once the project is total yare we will come to deployment of client system in genuinely world as its academic leave we did deployment i our college lab only with all need Software’s with having Windows OS.

1. **Maintenance**

The Maintenance of our Project is one-time process only

|  |  |
| --- | --- |
| Client Needs | Acceptance Testing |
| Requirements | System Testing |
| Design                   Code | Integration Testing    Unit Testing |

****

**Fig7.1:** Testing phases

Testing is the way toward discovering contrast between the normal conduct indicated by framework models and the watched conduct of the actualized system.

**7.2 TYPES OF TESTING**

There are 2 sorts of testing. They are given beneath:

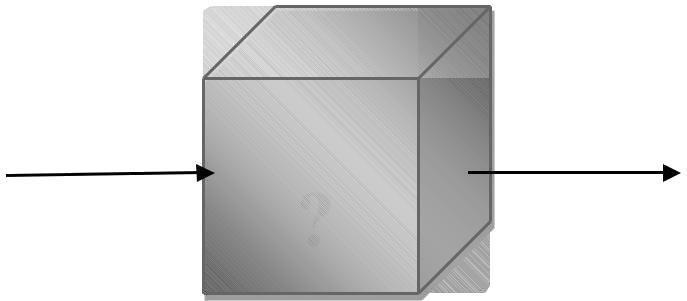
1. Black box or functional testing
2. White box testing or structural testing

**7.2.1Black Box Testing**

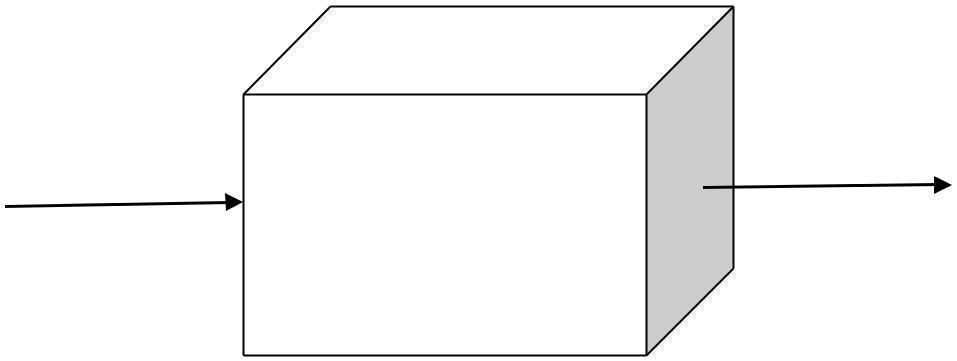
This strategy is used when learning of the predetermined capacity that an item has been intended to perform testing is known. The idea of black box is used for a system whose inside workings are not accessible to examination. In a black box the test thing is a "black", since its rationale is obscure, all that is known in the thing that goes in and what turns out, or the information and the yield.

Black box testing attempts to find errors in the following categories:

1. Wrong or missing functionalities
2. Errors in data structure
3. Initialization and termination errors
4. Interface errors
5. Execution errors

  
 f**ig 7.2:**  Black-box-testing

**7.2.2 White Box Testing**



**Fig 7.3:** White Box-testing

**INTERNALWORKING**

White box testing is about the testing that is related to the execution of the program. The plan of basic isn't to practice every one of the sources of info to practice the diverse programming and information structure used as a part of the program. Along these lines, Acceptance testing plans to accomplish test cases that will constrain the scope of various structures.

**7.3 TESTING PLAN**

Testing process starts with a test plan. This plan identifies all the testing related activities that must be performed and specifies the schedules, allocates the resources, and specified guidelines for testing. During the testing of the unit the specified test cases are executed and the actual result compared with expected output. The final output of the testing phase is the test report and the error report.

**7.3.1 Test Data**

Testing process begins with a test design. This arrangement recognizes all the testing related exercises that must be performed like the timetables, assigning the assets, and determining rules for testing. This testing of the unit of the predetermined experiments are executed and the genuine outcome is expected. The last part of the testing stage is the test report and the error report.

**7.3.2 Unit Testing**

Every individual module has been tried against the necessity with some test information.

**7.3.3 Test Report**

The module is working appropriately given the client must enter data. All information section frames have tested with indicated test cases and all information passage shapes are working properly.

**7.3.4 Error Report**

On the off chance that the client does not enter information in determined request, at that point the client will be incited with error messages. Error reduction is done to deal with the normal and sudden mistakes.

**7.4 Test Cases:**

It is a collection of circumstances or variables over which a tester will decide whether a scheme under examination fulfils necessities or works properly. The procedure of increasing test cases can as well aid to discover problems in the necessities or scheme of an appliance. A Test situation is any functionality that can be tested. It is also called Test Condition. As a tester, one may put themselves in the end user's shoes and find out the real-world scenarios of the Application under Test.

Below are the different test cases used to perform testing upon the functionality of the project?

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case 1: Loading Data Set | | Priority(M): Medium | |
| Description: Loading data set onto the environment. | | | |
| Requirement Verified:  Wrong dataset or empty field | | | |
| Pre-Requisites: dataset(SELF\_DRIVING) | | | |
| Actions | | Expected Results | |
| 1. Access the IDE  2. Run the python file | | **1.**IDE Should be Accessible  **2.**python file should run successfully.  3. It will upload and then we can view the data. | |
| Pass: No | Condition Pass: No | | Fail: Yes |
| Problems/Issues: Cannot upload file. | | | |
| Notes: Successfully Tested and Executed | | | |

**Table 7.1:** Test case-1

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case 2: Loading Data Set | | Priority(M): Medium | |
| Description: Loading data set onto the environment. | | | |
| Requirement Verified:  Yes | | | |
| Pre-Requisites: dataset(SELF\_DRIVING) | | | |
| Actions | | Expected Results | |
| 1. Access the IDE  2. Run the python file | | **1.**IDE Should be Accessible  **2.**python file should run successfully.  3. It will upload and then we can view the data. | |
| Pass: Yes | Condition Pass: Yes | | Fail: No |
| Problems/Issues: None. | | | |
| Notes: Successfully Tested and Executed | | | |

**Table 7.2:** Test case-2

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case 3: loss | | Priority(M): High | |
| Description: Choosing the epochs appropriately in order to reduce the loss | | | |
| Requirement Verified: Yes | | | |
| Pre-Requisites: After building the model successfully | | | |
| Actions | | Expected Results | |
| 1. Load the dataset successfully  2.Splitting the dataset  3.Building the model  4. Training the model | | 1.Model has been built successfully.  **2.** It has been trained successfully.  3. Model should give the minimum loss. | |
| Pass: Partially yes | Condition Pass: Partially yes | | Fail: Partially no |
| Problems/Issues: Number of epochs are deficient to get minimum loss. | | | |
| Notes: Successfully Tested and Executed | | | |

**Table 7.2:** Test case-2

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case 4: loss | | Priority(M): High | |
| Description: Choosing the epochs appropriately in order to reduce the loss. | | | |
| Requirement Verified: Yes | | | |
| Pre-Requisites: After building the model successfully | | | |
| Actions | | Expected Results | |
| 1. Load the dataset successfully  2.Splitting the dataset  3.Building the model  4. Training the model | | 1.Model has been built successfully.  **2.** It has been trained successfully.  3. Model should give the minimum loss. | |
| Pass: Yes | Condition Pass: Yes | | Fail: No |
| Problems/Issues: None | | | |
| Notes: Successfully Tested and Executed | | | |

**Table 7.4:** Test case-4

**CHAPTER 8**

**CONCLUSION**

**8.1 Conclusion**

Based on the results, it is evident that input data plays an important role in prediction along with Drop-out feature. As discussed in the working section, after supplying the input images, the filters are applied to get feautre maps and then next step is drop out feature.This Drop-out is a kind of mechanism i.e. while choosing the output of the preceding neurons instead of selecting all neurons, it drops certain neurons and selects remaining based on the probability given.So in our problem statement, we are giving certain probability i.e., 0.8 which means it considers 80% of the neurons in choosing the output of the preceding neurons and take them for further process. we initially checked the accuracy which was 0.5 when we runned, with 15 epochs, but later we then increased the epochs to 30 which gave the accuracy of 0.1. Hence, we consider the algorithm best which obtained greater accuracy.This is how our steering is predicted with less loss.

**8.2Future Scope**

Based on the results, it is evident that input data plays an important role in prediction along with Drop-out feature. As discussed in the working section, after supplying the input images, the filters are applied to get feautre maps and then next step is drop out feature.This Drop-out is a kind of mechanism i.e. while choosing the output of the preceding neurons instead of selecting all neurons, it drops certain neurons and selects remaining based on the probability given.So in our problem statement, we are giving certain probability i.e., 0.8 which means it considers 80% of the neurons in choosing the output of the preceding neurons and take them for further process. we initially checked the accuracy which was 0.5 when we runned, with 15 epochs, but later we then increased the epochs to 30 which gave the accuracy of 0.1. Hence, we consider the algorithm best which obtained greater accuracy.This is how our steering is predicted with less loss.

**CHAPTER 9**

**BIBILOGRAPHY**

**References**

[1] https://www.tensorflow.org/tutorials/images/deep\_cnn

[2] https://edition.cnn.com/2019/02/25/tech/self-driving-cars singapore/index.html

[3] https://ieeexplore.ieee.org/document/8253388/

[4]https://developer.nvidia.com/discover/convolutional-neural-network

[5]https://www.nvidia.com/content/dam/enzz/Solutions/deeplearning/documents/

DLI\_Catalog\_Published\_June2018.pdf

[6]https://blogs.nvidia.com/blog/2018/09/05/whats-the-difference-between-cnn-and-an-rnn/

[7] https://devblogs.nvidia.com/deep-learning-self-driving-cars/

[8]https://images.nvidia.com/content/tegra/automotive/images/2016/solutions

/pdf/end-to-end-using-px.pdf