National College of Ireland

Prediction of Steering Angle of vehicle using Deep Learning Models

MSc Research Project
MSc in Data Analytics

Javed Mohammed
Student ID: x19219458

School of Computing
National College of Ireland

Supervisor: Dr Catherine Mulwa

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Javed Mohammed
Student ID:	X19219458
Programme:	MSc in Data Analytics **Year:** 2021-22
Module:	Research Project
Supervisor:	Dr Catherine Mulwa
Submission Due Date:	31-Jan-2022
Project Title:	Prediction of Steering Angle of Vehicle Using Deep Learning Models
Word Count:8335	**Page Count: 25**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Javed Mohammed

Date: 31-Jan-22

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Prediction of Steering Angle of Vehicle Using Deep Learning Models

Javed Mohammed
X19219458

Abstract

In today's advent of technology an exponential rise and enhancement has been observed in the field of computer vision, image processing and deep learning. This renaissance has recently dominated the automation field. The concepts of image processing can be easily applied in driving vehicles wherein the drivers can be replaced with the rising technology. However, the most important factor that comes into picture in automation is the steering angle of a car by which the vehicle is responsible to take the curve. Automotive vehicles (AV's) have started considering the steering angle prediction and many automotive companies have also invested in it such as Tesla and Udacity. This filed has however attracted multiple researchers and insurance companies to invest in them. Deep learning architectures have been considered to be the apt fundamentals that can be applied in such a scenario. Hence, this project proposes a steering angle prediction in AV's using DL. The implementation is carried out in two modules namely: image processing and CNN. For every point along the trajectory of the vehicle, a steering angle is calculated, considering the speed of the car and the applied brakes as significant parameters and a set of images are captured by the cameras installed. The first phase of execution involves image processing that utilizes images for the training purpose and data augmentation for resizing the images. In the next phase, CNN concepts are applied and the processed image obtained from first phase is fed as the input to this phased and a predicted steering angle is generated as the output. For this type of automated model, the leading car companies use the architecture called 'Alex Net' and 'Pilot Net' is the other architecture which was created by Tesla, these types of architectures basically can store the bigger size images or highly resulted images which leads to the complexity and high cost. So, for reducing the price and complexity of model in this project we are going to create a light architecture by embedding big frame architectures into it and named as 'J-Net'.

1 Introduction

1.1 Background

An automated system that allows vehicles to drive safely without the involvement or interference of humans is generally termed as autonomous driving. This involves a computer brain to mimic the human brain and function accordingly. A machine is capable to do so with the help of sensors and can drive a vehicle under any circumstances, just like humans. Apart from vehicles, the concept of no or minimal human involvement in driving vehicles can also be applied in aeronautics wherein; the pilot works in a similar fashion of driving in auto-pilot. One of the common ways through which automated driving can be achieved through technology is via deep learning. Deep learning makes this automated driving possible by

establishing interesting properties of being able to automatically learn complex operations just like humans. The machine then further maps these complex functions and scales them up in terms of efficiency. Such properties become important and play a vital role in real life applications. One of the most widely used domains to exhibit automated driving is image classification and recognition. There are 6 autonomy levels are ranging between 0 and 5. Level 0 is not self-sufficient. Level 1 is "hands-on", Level 2 is "hands-off", Level 3 is "eyes off", Level 4 is "mind off", Level 5 is "steering wheel optional". Now getting into detail Level 1, known to be a hands-on system,has little choice except driver intervention to control your car the best example for this is Adaptive Cruise Control. Level 2 is known as hands-off which deals with the controlling of the steering and braking and acceleration system of the vehicle. Level 3 is eyes off where thedriver can close eyes in the conditions like empty roads but the driver should be alert when in traffic. Level 4 mind off in this level the driver can completely get relaxed sleep, listen to music or watch movies and this can be only possible in geo-locations and selected spatial areas. Level 5 steering wheel optional in this there is no need of driver involvement and it is similar to level 4 but not limited to areas or locations where the automated vehicle can function on any type of road and any climatic conditions (Perkin Coie', 2019). The primary goal of the project is to derive a model that could predict the steering angles of an automated driving vehicle. The motivation behind developing such a handy technology is to enhance automated driving without the intervention of humans. For this to take place, a model is given an image that is captured while driving. These images should be further responsible to minimize the loss function so evolved between predicted values and the actual steering angle produced by humans. This thesis focuses on application of such techniques that are accompanied through supervised learning. Such techniques assist to predict vector values of steering angles that are required to navigate a car through a trajectory. The implementation of this model is done using a simulator that is generated with the help of a dataset. The primary goal of the thesis is to design a model that could encapsulate real life driving scenarios and generate results with minimal errors. Each file in the database describes a trajectory curve that is recorded over duration of time and further results to evaluation derived from the training data. Later, this training data is fed to the processing stage which is responsible to extract features of relevant data that can be further used for steering angle prediction in Automatic Vehicles.

1.2 Problem Statement

Deep learning has always had a positive impact in order to control AV's. The primary reason behind this was its ability to process and filter unlabeled data. Hence, it is said that deep learning possesses the capability to understand the world as humans do and analyses it through real life applications. DL does so by focusing on objects that are presented at hierarchical level of resolutions. As the concepts of deep learning are sensitive to the environmental conditions they are surrounded with, it requires large amount of data to process and train the machine in order to generate accurate results. In such a scenario, a neural network plays a beneficial role as it tends to understand complex items and possess the ability to simplify it. In an AV steering a car is the most crucial part in terms of automated driving. This process constitutes a complex task making it difficult for algorithms to predict them.

Research Question: Can deep learning prediction steering angle prediction models improve the estimation of lanes, and braking systems of the automated cars when it is in motion?
Sub RQ: Can combination of deep learning and machine learning improve the performance of J-Net model?

Table 1: Tasks that are done in the Implementation

Tasks	Description
Task 1	Implementation and evaluation of steering angle prediction of a vehicles using deep learning algorithms.
1-1	Implementation and evaluation of the Convolutional Neural Networks Layers Model.
1-2	Implementation and evaluation of the existing complex Pilot Net model to predict the steering angle of the vehicles.
1-3	Implementation and evaluation of the J-Net model to predict the steering angle of the vehicles correctly.
Task 2	Comparison of the Models that are evaluated.
2-1	Outputs taken from the Pilot Net and J-Net are compared.
2-2	Concluding with the best model which can accurately predict steering angle of vehicles.

The overall abstract and working implementation is discussed in chapter 1. Chapter 2 presents a detailed comparison and brief about the existing work in the field of automated driving using the concept of deep learning and CCN. The entire processing workflow along with respective diagrams and features are detailed in chapter 3. Chapter 4 focuses on design implementations of the project followed by the implementation details in chapter 5. The thesis comes to an end with conducted experiments along with their analysis proceeded by conclusions. Lastly, a summary of all the references used in the thesis is presented after conclusions.

2 Literature Review

2.1 Automated Vehicles Introduction

Completely automatic cars are projected to play a central role in the growth of transportation systems in the next years. There are several advantages to self-driving vehicles. For example, automated vehicles can result in considerably safer operation and improved traffic management performance. People who utilize such cars can travel more safely and conveniently. As a result, persons who have been unable to drive due to age or handicap may now have the same amount of flexibility like any other vehicle driver.

2.1.1 History of Automated Vehicles

It wasn't long before people thought about making cars that could drive themselves. In 1932, innovator Frank (Cranswick, 2013) showed off a transmitter car that could drive through the streets of Manhattan without anyone in the driver's seat. The "Demon Auto" car has been managed by broadcast, that was ready to initiate the engine, change gears, and sound the horns, among other things. It was a Casey Vehicle from 1936. It had antennas in the back

seat, and the driver was able to control a second car that was emitting radio waves while he was driving it. Signals from these antennas were kept by the antennas that received them. The transmitters forwarded transmissions to system breakers, who used small electric engines to move the vehicle. It became one of the first types of self-driving cars.

2.2 Direction Change

This Model was developed by the group of engineers in a Mechanical laboratory in Japan (Bimbraw, 2015) they have used a photographic model with embedding a computerized program which can transfer data to read the photos of the road. Again after 10 years the engineers of Germany have developed a fully camera loaded car which can move with a speed of 60kmph and at that time of period technology has also increased and the strength of sensing the cars also increased.
After again by the German engineers in the year of 1980-90 development in artificial intelligence has been increased in transportation. This research was sponsored by the leading car company Benz. With the help of pioneer car company fully loaded car has been made and software was developed to record the road images and sent it to the braking system and after the vast research this model was launched after 90's into market (Nidhi *et.* al 2013).

2.3 Development of Automated Vehicles

All the leading car companies all over the world have developed the automated technology and they have invested billions of dollars for this prestigious development. For these vehicles they have incorporated many sensors and cameras and installed highly developed architecture or software to put this development safe on the road. The sensor that is widely used is LIDAR which is used to detect the margins on the road and helps the car without crossing the lanes (Szikora, 2017).

2.4 Computer Vision

Computer vision is a part of a much broader family belonging to Artificial Intelligence. It enables computers and machines to visualize objects and derive meaningful information from it that could help the model in reaching closer to generating accurate results. It takes inputs from digital images as well as visuals. It is that capability of a machine that makes it able enough to observe, think and act just as what a human would do (Shuyang Du *et*.al 2018). The concept of computer visions works in a similar fashion to that of humans, but however they do not have a head start to it. Computer vision generally, achieves its performance through functions that are derived using cameras installed on the models. However, they need to interact with the camera and process those images much faster along with specific algorithms of deep learning. It also requires a much larger amount of data to process and recognize images. It functions by analyzing a dataset for a period of time and then predict its output. This concept is further, accompanied by two techniques namely: deep learning and CNN's.

2.5 Image Processing

The concept of image processing is to process an image using techniques and algorithms which is clearly shown in the Fig 1 below that is carried out by a computer. It belongs to the

field of digital images wherein images are captured and further processed by undergoing algorithms and deep learning methods. The computers are generally formed in such a way that they take the input images from the camera and process them through layers of computerized algorithms. During this process of applying algorithms, the images undergo a series of resampling and resizing with respect to the output so required. All grey scale images are converted to their respective dimensions and worked upon in the later stages. Generally, the concept of image processing involves processing an image by applying certain transformations to the images in the form of smoothing or sharpening. Such transformations are done by processing algorithms. On the other hand, computer vision as mentioned above makes use of these image processing techniques and algorithms in order to identify certain parts or specific attributes of a digital images.

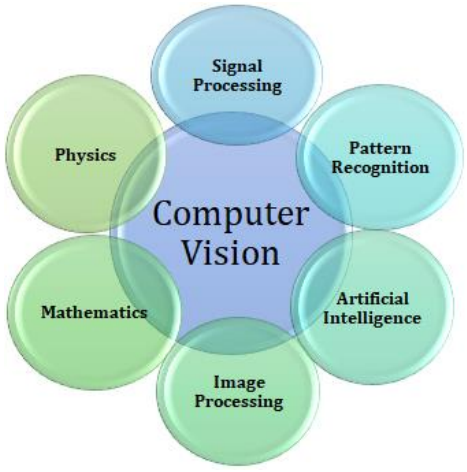

Figure 1: Image processing as a part of Computer Vision

2.6 Neural Networks

Neural networks form as a much larger part of networks that works on the concept of neurons and tries to implement works on the basis of mimicking a human brain. Since the human brain is made of nerves, a neural network is made of neurons that tries to learn recognizing objects and perform computational operations om them. This process is carried out in a similar way so as to how a human brain shall work and operate. The neurons so present in the network are artificially generated by mathematical calculations that adapts o the changing nature of the neurons. The network generates a maximized output with the best possible results using this concept. Although it has roots leading back to artificial intelligence, it has gained wide popularity in the working of many real-life applications. The working is followed by taking images as inputs by the input layer and generating respective outputs as required. In middle to input and the output layer, a neural network comprises of multiple hidden layers. It is in these layers that the actual working of algorithms takes place using activation function and respected weights. A typical structure of a neural network is depicted below:

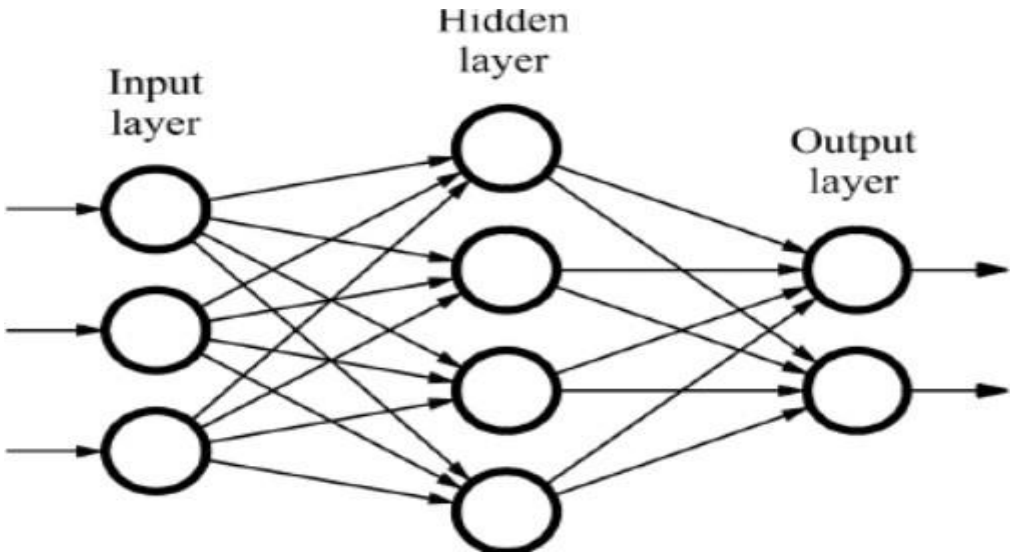

Figure 2: Structure of a neural network

2.7 Deep Learning

Deep learning forms as a sub category of machine learning which further forms a category of artificial intelligence. However, deep learning is a part of neural networks that work on the concept of mimicking a human brain using neurons. A much deeper level of understanding goes into deep learning unlike machine learning that focuses only on basic algorithms such as random forest and KNN. The top three hierarchical classifications of deep learning are CNN, RNN and ANN. All the algorithms that make use of the concept of deep earning works through execution of multiple layers involves within it. Under the mentioned three categories, there are multiple variants that exists and are used in accordance to the application of the model that is to be used. The most commonly used deep learning algorithm is a CNN structure. It involves multiple layers within it and is depicted in the diagram below. However, its working is similar to that of a neural network involving associated weights and activation functions.

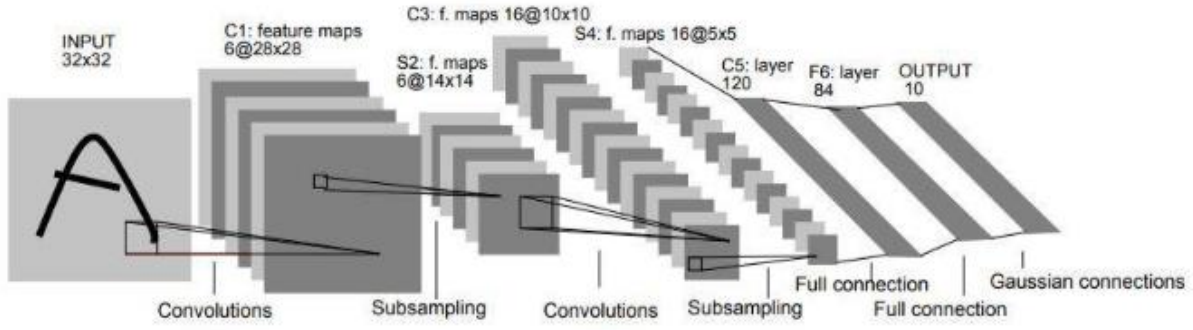

Figure 3: Layers of a CNN

2.8 Data Augmentation

During the implementation of a machine learning based work, at certain times, a data that is trained from a dataset may not function properly. This result in an overfitting issue and it can be prevented only by applying regularization on the trained data. Further, this regularization is obtained from data augmentation. Data augmentation is that techniques of neural network that possess the ability to artificially expand a dataset and create new ones from the existing dataset. This process serves its advantage, when a model undergoes the issue of overfitting. In further stages, this concept can also be applied to various other implementations, such as increasing the performance of a network by augmenting the existing data. Hence, it is believed that this concept is widely used when it comes to deep learning. However, the process of augmentation can be applied to the following sets of data:

- Audio
- Text
- Images

Xiao's research in deep driving makes a significant contribution to the field of guided vision. Affordance was introduced and applied to the detection of writers' articles on highways and byways. Distance between neighboring lane markers, distance between preceding automobiles from current/left/right lanes, and angle of departure of the coach from the road tangent are all factors that contribute to the provision of this sort of additional accessory. Superfluous computations are no longer required as a result of this. In order to better foresee this eventuality, the authors design a deep end-to-end network from scratch (J.XIO et.al. 2015).

Improved learning techniques have resulted in the creation of recent neural networks (with over one million weights) (J.Koutnik et. al. 2013). Collection of data and model testing were carried out using the TORCS Racing Simulator, which was also utilized in Extreme Driven (Bedue C et al. 2020). Automated DARPA Vehicle (Y.LeCun et al.2005) is a DARPA project that intends to develop an off-road robot that can travel through unknown open terrain with visual inputs, similar to the DARPA project DAVE (Automated DARPA Vehicle). A human driver utilized hours of information over a long period of time to train the DAVE in a wide range of situations and environments. In (Y.LeCun et al.2005), a 6 convolution network is used to process an initial pair of low-resolution pictures that are fed into the network. In difficult scenarios, according to one research, the actual distance between DAVE crashes was around 20 meters on average.

ALVINN and DAVE were the inspiration for DAVE-2 and PilotNet (M.Bojarski et al. 2017), which were both developed. Each level has nine layers, with the first being a standardizing stratum, followed by five coevolutionary strata, and the last three being completely linked. Two routes (one with lane markings and one without), automobile-parked local streets, tunnels, and roadways will all be used to acquire trained data from the participants. We have a suspicion that not all of these efforts have made use of real-time information. Pilot Net, for example, was only taught to drive automobiles by viewing the video frame that corresponded to the steering input. Deep Driving, on either hand, makes use of a somewhat different approach. The authors compute the appropriate speed and vehicle angle based on a variety of physical criteria in order to provide a comfortable driving experience. When operating on sloped roads, Deep Driving, on the other hand, has a tendency to predict incorrect wheel angles.

According to the authors, the LSTM units are located in the penultimate layer. To sum up all previous states, the LSTM's internal status is intended to summarize. The authors utilized 64

hidden neurons in their LSTM model. We believe that this statistic is too low to accurately represent the amount of time spent in an autonomous vehicle. Time modeling in the network transfer approach is improved using a variety of strategies, including residual buildup and conventional LSTM (Bedue C, et.al. 2017). This was done by evaluating how well the suggested network was able to acquire space-time information and accurately forecast the steering wheel angle.

(D.A. Pomerleau, 1989, pgs. 305-316) was one of the earliest attempts to train a steering system from beginning to end using a very simple, hidden state neural network. The network employed a very small image features and light end information as inputs and expected classification scoring using the picture and data to reach the middle of the road. However, only the most basic driving maneuvers were used in the initial tests. Situations with modest obstacles, the promise of learning from the start to the finish (H.XU et al. 2017) has demonstrated that the human hand necessary for media perception techniques has reduced the need for rigorous and time-consuming fine tuning. A training program achieved in generating a simulation that continued the route after detecting a roadway producer in very little than half an hour. The device, which looked to work effectively with cruising velocities, was then put itself through the tests on a sprint woodland road that resembled the duration of a conventional transport system confined to such conditions. These preliminary experiments were confined to fairly simple driving situations with little obstructions, but they exhibited the potential of training from the start so the human hand necessary for media perception approaches avoided the need for intensive and time-consuming fine-tuning. A training session was effective in developing a model based on the revealed road generator in very little about half a minute of discovering it.
Following the revolutionary CNNs pattern recognition there is a significant amount of study on self-driving cars utilizing deep learning approaches. Numerous research efforts have lately been made in applying deep learning approaches to the low-level/mid-level computer vision problems of robot navigation trends, where the cheap computing cost of applied CNN allows for real-time processing (Huval et al 2018) offer a technique for predicting 2 points of a single road section using CNN and combining it with RNN to identify lanes borders. Similarly, (Hadsell et al. 2017) suggest using a deep network network to extract relevant and meaningful properties from an input picture, and then using the data for training a genuine classification algorithm openness. They included a self-supervised learning system capable of successfully categorizing complicated terrains at distances ranging from 5 to more than 100m from the platform, much outperforming path-planning.

2.9 Conclusion

In this chapter various topics have been covered like introduction to the Automated vehicles, its evolution and how the artificial intelligence works in these types of models and neural network use in the automated cars. In the next chapter we will discuss about the Design specifications.

3 Design Specifications and Methodology Approach

In this chapter discussion is made regarding the Design of the models such as (Alex Net, J-Net, NVIDIA Architecture) and furtherly we have discussed about the Methodology approach such as (Convolution Neural Network and Combination of Recurrent Neural Network and Convolution Neural Network).

3.1 Alex Net

According to the Table 1 given below original architecture, Alex Net contains 5 levels: one layer that flattens, two completely connected layers, and one output layer. Alex Net was developed in order to do a categorization task. Since it was created for the ImageNet challenge, it has categorized more than 900 photo classes. Because Relu had not yet been formed, activation units such as the Sigmoid or the Tanh were used. It was possible to have three different max-pooling levels in this design. A little amount of padding was provided to each convolutional layer. Because the padding is the same, the picture size is also the same. The input dimensions of 224224X3 were used to create a model to preview the 1000 class image.

Alex Net's initial file size was 500MB in size. The rehabilitation version has a file size of 320MB. The most significant modifications to the reimplemented version were the increased number of pool levels. Because the input picture utilized in this inquiry was modest, just one maximum pool layer was used in the original Alex Net, as opposed to three maximum pool levels in the original Alex Net. In this reimplemented version of Alex Net, another significant modification is the addition of activation units. Units with the ability to determine the output of a node. The Tanh, Sigmoid, ReLu, and so on are all distinct types of activation units. As a result, the original AlexNet classification job was employed for the output layer in conjunction with the SoftMax layer.

Count of Layers	Input	Layer Type	Output	Layer Activated
1st layer	3*65*100	Normal Layer	3*65*100	Null
2nd layer	3*65*100	Normal Layer	3*65*100	Null
3rd layer	15*50*94	Convolutional	10*45*512	ReLu Activation
4th layer	10*45*512	Convolutional	8*10*380	ReLu Activation
5th layer	8*10*380	Convolutional	6*40*380	ReLu Activation
6th layer	6*40*384	Convolutional	1*20*512	ReLu Activation
7th layer	4600	Fully Connected Pooling Layer	4090	ReLu Activation
8th layer	4090	Pooling Layer	4090	ReLu Activation
9th layer	4090	Fully Connected Layer	900	ReLu Activation
10th layer	10	Output	01	Identical

Table2: Layers of Alex Net

3.2 CNN J-Net

In Table 2 given below we have given the layers required in CNN J-Net, comparison to LeNet, AlexNet was the first network to obtain more accurate ImageNet competition (Russakosky et al. 2015). AlexNet is also one of Neural's "heavyweight" networks, which means it is extremely powerful (Russakosky et al. 2015). The bigger the complexity of the network, the better the model's accuracy. It was AlexNet, a deep neural network, that was utilized to categorize a picture. It features a large number of convolutions and is a time-consuming method. It has deeper filters, which allow it to form a denser and more sophisticated network as a result. PilotNet was developed exclusively for autonomous cars

(Bojarski et al. 2016). PilotNet contains five coil layers in a similar fashion, but it also has a smaller number of filters and set filter sizes. Consider both the advantages and disadvantages of neural networks. The novel neural network design is being used in this study, which is called J-Net. J-net has three layers with 16, 32, and 64 filters, 5X5 and 5X5, one flattened and two entirely connected layers, and it has three layers with 16, 32, and 64 filters. M-net features three filters, each of which contains 16 or 32 filters. PilotNet did not have any max pool layers because there were enough fully connected layers and filters to extract the features that were needed. J-Nets use fewer convolutions and maximum pooling in order to prevent losing important attributes from an image. The number of fully connected layers is increased in order to extract additional features such as edges, geometrical qualities, and so on. Because it was indicated in the preceding section, the goal of J-Net is to create a lightweight model of the world. As a result, increase in number of fully connected layers would be prohibitively expensive. The goal of this project is to remove lines and lines that do not require multiple turns. It would be sufficient to use three convolutional layers to extract basic features in this case. The number of completely linked layers in a model reduces the density of the model as well. The ReLu Activation model was based on this one. The last output layer is represented by arg tan or the identity function.

Count of Layers	Input	Layer Type	Output	LayerActivated
1st layer	3*65*100	Normalized Layer	3*65*100	Not Activated
2nd layer	3*65*100	Convolutional Layer	62*195*17	ReLu Activation
3rd layer	60*194*18	Convolutional Layer	28*95*30	ReLu Activation
4th layer	28*95*30	Convolutional Layer	10*45*66	ReLu Activation
5th layer	10*45*66	Flattened Layer	27,500	ReLu Activation
6th layer	27,500	FC Layer	5	ReLu Activation
7th layer	5	FC Layer	01	Identical

Table 3: Layer structure of J-Net

3.3 NVIDIA's Architecture

The Fig 4 given below shows the architecture used to implement the purpose of AV's is accomplished using the NVIDIA's architecture. A modelled network that was trained for 70 hours was used to implement the goal of the project. A CNN architecture was finally employed using multiple hyper parameters. One specific variant that is used in NVIDIA's architecture for implementation of AV's is the PilotNet. This variant is used to detect the steering wheel of the car and predict its angle. This variant works on the concept of neural networks. PilotNet is observed to have 9 layers in all comprising of five convolutional layers, three FC layers and one pooling layer. The input given to the first layer is built over a period of time axis. The normalization layer of this architecture is responsible to convert pixel values into binary values that can be further detected by the computer. Apart from these layers, a dropout layer is utilized to keep the overfitting issue under control. A max-pooling layer is also involved that calculates the associated weights and respected biases.

The diagram below depicts a typical PilotNet model of a neural network:

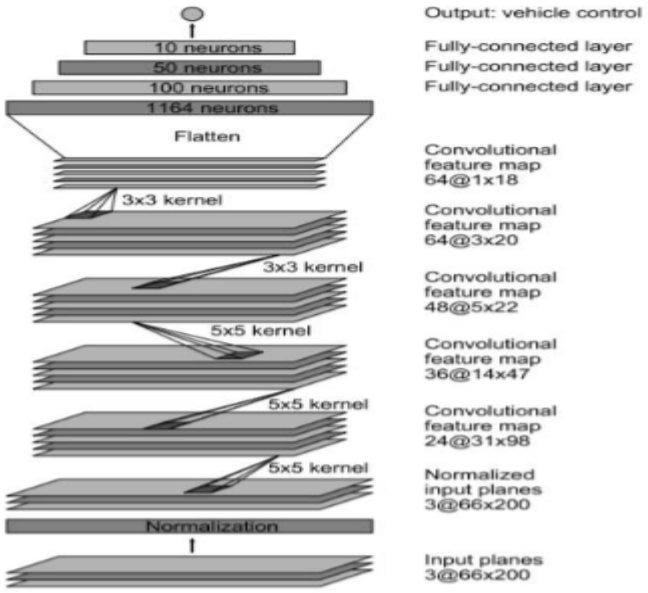

Figure 4: PilotNet model of the NVIDIA Architecture

3.4 Approach 1 –CNN

In this approach the Fig 5 given below shows the workflow of the CNN. The primary aim of the project is to build an inexpensive model that could detect the angle of a steering wheel in automated vehicles. The fundamentals of CNN are best known for their ability to handle images and their concerned data. The data gathered from the cameras so installed is fed to the CNN model. The camera contains a series of videos that are further broken down into images and are captured by the camera which is later processed by the machine. Image processing in this approach takes place through CNN model wherein; all the images are passed through the layers of the model and processed between them. The entire implementation takes place on purely CNN's. The below workflow presents the working of a CNN on images:

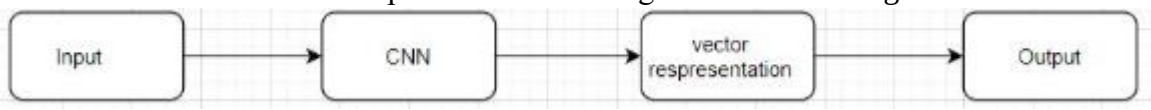

Figure 5: Workflow of CNN

All the digital images so obtained from the camera are fed as the input to the initial layers of a CNN. A process of vector conversion takes place on these images. The vector matrices are formed through mathematical conversions that occur on the hidden layers of a CNN model. Associated weights and biases are calculated and the required output is generated. The entire implementation occurs on the fundamentals of neural networks that involves the working of neurons.

3.5 Approach 2 –CNN + RNN

This approach Fig 6 showcases us about the combination of RNN and CNN. In the second d approach, a combination of two neural networks is used mainly: CNN and RNN. CNNs are widely known for their ease-of-use data and the way they handle large amounts of data, on

the other hand RNN's perform better along with time series model/. An RNN would need historical data to be used as inputs that can be given to its neural networking layers. These layers are further responsible of store sequential information of data that is related with the AV's. This sequence of data can also be in the video format. One of the most widely used variant of RNN is LSTM. The below workflow presents the working of a RNN on images:

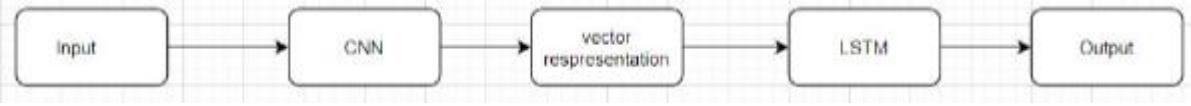

Figure 6: Workflow of RNN

After considering two approaches due to computational limitations and project requirements, the first approach was chosen. The main goal was to build a low-cost model or a lightweight model but using RNN's will be an expensive approach. Using the combination of two neural networks will lead to a complex model where the size of the model will, and training time can take more than 2 to 4 days using CPU. Though the main task was to predict the steering angle which can be accomplished by using simple CNN's where it generates output for every single input.

4 Implementation and Data Pre-Processing

4.1 Dataset Description

Collection and getting access to the dataset was the most challenging part of the implementation work, as the thesis proposes to build an inexpensive model for AV's. however, there were a few available repositories that had an open access to dataset of self-driving cars. The dataset consisted data obtained from sensors of the cameras so installed. These sensors captured images of steering angle of cars and consisted a video of 20 minutes. Other datasets which were large enough contained images from front angle view that were captured using parameters of acceleration and brake. This file was slightly larger than the previous file and hence contained a video of 70 minutes. However, there are many existing companies such as Udacity and Commai that provides data of self-driving cars.
The dataset from Commai had 75 GB of data. This dataset was little difficult to study. On the other hand, the Udacity dataset contained 6 hours of driving data including climatic conditions as a factor that contribute in the pattern changes of the angles of a steering wheel.
The datsaset in Udacity also contained information of brakes and acceleration so applied apart from the angular data of steering wheels.
However, it was observed that the largest dataset was provided by China based company called as Baidu. They possess an open dataset that provides all the details related to self-driving cars and vehicles.
The below image is the screenshot of the angle associated with every image:

Figure 7: Road images captured by camera sensors

4.2 Data Preparation

The most prominent step of any deep learning-based model is the step of data preparation. The data which is obtained from the dataset consists of a large number of image and text-based files. Below Fig 8 will show us about code which is executing the splitting data. These files contain the information of different angles of a steering wheel of a car. However, all the files are present in different folders. During the implementation phase, rather than editing the path and linking it to the files, it is very easy to make use of the python library that links the folders with their respective files automatically. The main repository containing the dataset comprises of 45406 image files in one folder. And the other file comprises data containing images of angles of steering wheels so captured. However, a text file remains common that consists of both: images and angles of the dataset so obtained. An iteration of test and training phase is carried out and every file stores its associated paths and angles. The angles in the dataset are stored in the form of degrees. Further, these angles are converted into radians so that processing the image data becomes easy on the neural network. However, activation functions such as ReLu and Tanh can be used to go deeper into the network. This process makes computational frequency easier to implement on any model.
The below snippet depicts the code of splitting the dataset:

```
images_path = "/driving_dataset/"
Steer_angles = "/driving_dataset/data.txt"

images = []
Angles = []
```

Figure 8: Snippet of code executing the splitting of dataset

4.3 Data Processing

This is the most crucial part of the entire working flow. The dataset so obtained from the repository is meant to undergo the procedure of data processing. This procedure is responsible to filter out irrelevant and redundant data that is observed in the dataset so

acquired *(S. Marra et.*al 2014*)*. The presence of this redundant data results into decreasing the implementation time and increases the computational complexity of the system. Thereby reducing the overall efficiency of the model. As feeding the entire dataset to the neural network might be a time-consuming task, this process proves to be an advantage to the working model of the system. In our dataset, unnecessary features such as sky and mountains are eliminated. Hence, out of total image, only the last 150 pixels contain the images of a road are selected.

4.4 Training and Testing

Fig 9 which is given below shows the training and testing phase, in which the machine is initially trained with the dataset so obtained from the repository. The testing part is generally done in the later stages wherein the model is checked for accuracy. However, this splitting of data in to training and testing is done on temporal data. Every time a split takes place, the model uses different records present from the dataset. This dataset could be series of videos or images which are further broken down in the processing stage. Since the thesis makes use of time stamp model, the input which is fed is historical data containing past values. These past values are fed to the model on a time axis. Hence, Fig 10 shows the process takes place through a time stamp model. However, this splitting of data occurs in a sequence and is split in an 80:20 ratio wherein 80 percent of the data is used for training the model and 20 percent is used to test the model. Once the model is tested, further procedure of evaluation takes place. The snippet below depicts the train and test split of the dataset used for training and testing purpose:

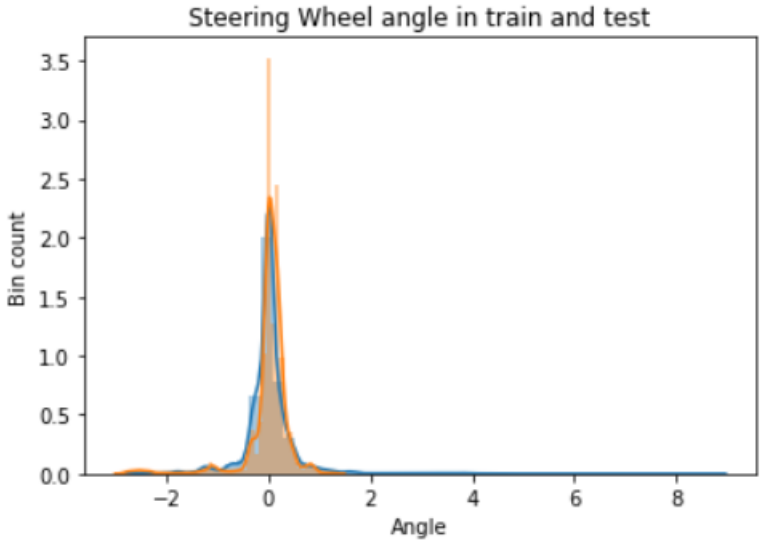

Figure 9: Test Comparision

```
Total Input Images in Train set : 36324
Total Output Values in Train set : 36324
Total Input Images in Test set : 9082
Total Output Values in Train set : 9082
```

Figure 10: Dimensions of train and test set

5. Evaluation and Analysis of Result

In this chapter all the Tasks which are mentioned above are evaluated and results are attached below.

The test set is used to evaluate the model. The testing data is prepared in an unbiased way when compared with the training set. Images were split based on the time axis hence there is no bias for train and test data. Performance metrics such as root mean square error and accuracy are computed for all the epochs. The accuracy at each epoch and average accuracy for all the epochs are calculated. This section is divided into subsections that compare the three different models concerning their training time, train and test loss, accuracy and model size.

5.1 Model 1: Learning rate of Pilot Net

Table 4: Learning rate of Pilot Net

CNN	Convolution Layers	Kernels	Learning Rate	Epochs
PilotNet	5	24, 36, 48, 64, 64	Adam =0.0001	30

The first model that was used to build an autonomous model is 5 PilotNet which we can clearly see in the Table 3 above. The model had 5 convolutional layers, one flattens layer, 4 fully connected dense layers and 1 output layer. The first convolution contained 24 filters, the second contained 36, third contained 48, fourth contained 64 and fifth contained 64 filters. The learning rate used in this model is 0.0001, this was decided by performing hyperparameter tuning. The default learning rate for Adam optimizer is 0.001.
The figure below shows the average train and test loss of the model at each epoch:

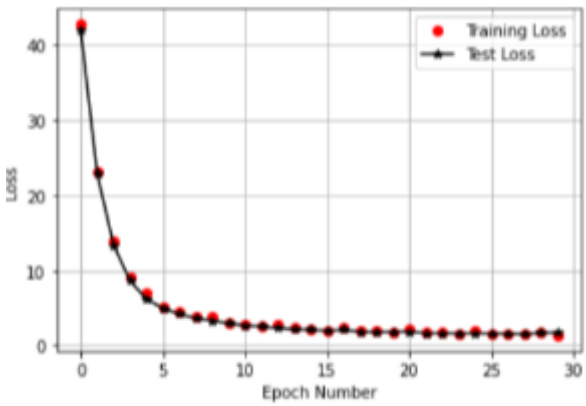

Figure 11: Train and Test set of Pilot Net 1

In the below Fig 11, training loss and test loss is almost the same which means there is not much difference in the actual and predicted values. At the first epoch, the model made 40%

```
Avergae Training loss =  5.186817431891959
Avergae Training loss =  4.995085770224532
```

Figure 11: Average training and test loss

error as the epochs increased the model started performing well. The loss started decreasing. The average loss of 30th epoch was 1.2 but at the end of 30th epoch, the model made an error of 0.12.
Training the model to a greater number of epochs would have overfitted the model. The size of data was not so large to train over 100 epochs, still, the model was achieving 99% accuracy with 30 epochs.

5.2 PilotNet-2: Learning rate = 0.001

Table 5: Learning rate of Pilot Net

CNN	Convolution Layers	Kernels	Learning Rate	Epochs
PilotNet-2	5	24, 36, 48, 64, 64	Adam =0.001	10

The below Fig 12 shows the error the model is making. When the learning rate was increased the model started too overfit. At first epoch the average test loss was around 12, by the end of 10th epoch, the average test loss was around 1. If the number of epochs was increased the model will overfit further and will not give accurate results. When pilotNet-1 was able to achieve results in 30 epochs with good converging rate.

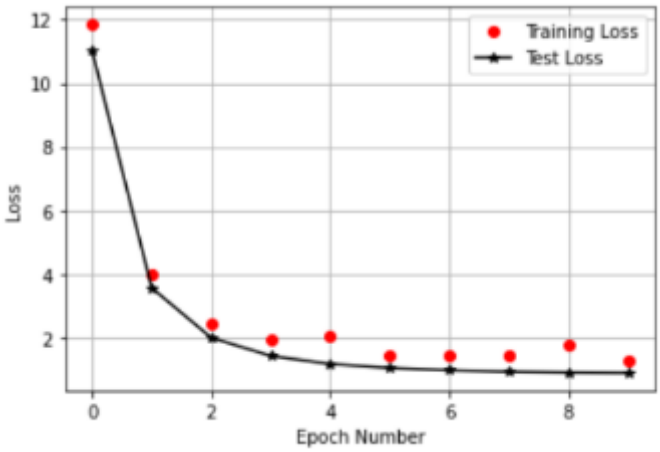

Figure 12: Train and Test loss of PilotNet 2

The below graphs Fig 13, show the comparison of pilotNet with different learning rates. The more the learning rate is the model gets overfitted. And Fig 14 show us the Average training and test loss of Pilot Net1 and 2.

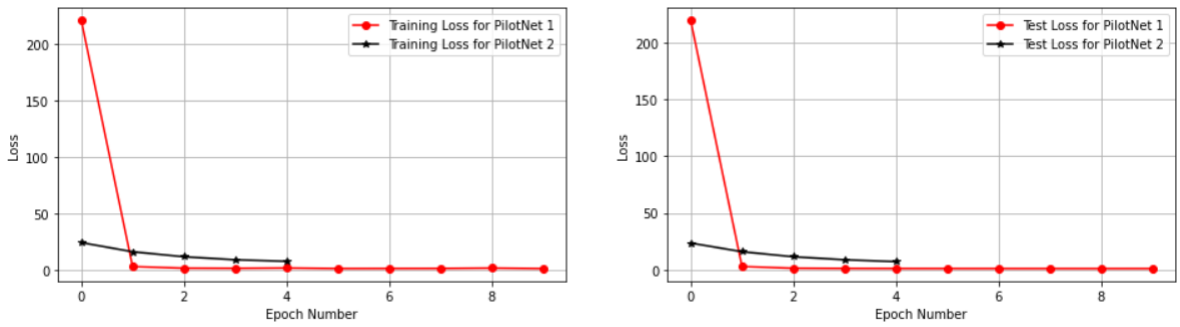

Figure 13: Comparison of train and test loss between Pilot Net 1 and Pilot Net 2

```
Avergae Training loss for 10 Epochs=  2.968265936355105
Avergae Training loss for 10 Epochs =  2.389635551210493
```

Figure 14: Average training and test loss

5.3 PilotNet-2: Learning rate = 0.00001

Table 6: Learning rate of Pilot Net 3

CNN	Convolution Layers	Kernels	Learning Rate	Epochs
PilotNet-3	5	24, 36, 48, 64, 64	Adam =0.0001	5

The below Fig 15 shows the error the model is making. When the learning rate was decreased the model was converging slower to the solution. At first epoch the average test loss was around 24, by the end of 5 th epoch, the average test loss was around 23. This means the model was still trying to reach local optima but at a slower rate. If the number of epochs was increased the model might work but the dataset would not be less to train on more than 30 epochs.

The below graphs show the comparison of test errors of all the three pilotNet with different learning rates.

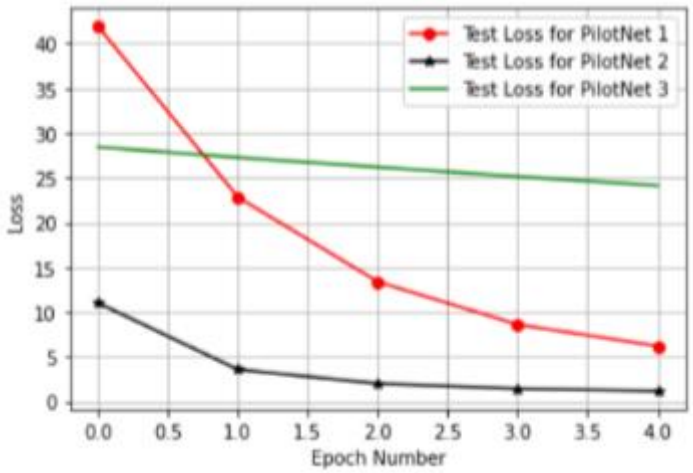

Figure 15: Comparison of Train and Test Loss of PilotNet-1, PilotNet-2 and 3

5.4 Model 2: J-Net

The main goal of this study was to design J-net i.e an inexpensive model. The model is called inexpensive because of its architecture, size and complexity. The model had only three convolutional layers. Convolutional layers were supposed to be the most expensive operation because it is the matrix multiplication of pixels with filters/kernels. The dataset consists of 30k plus images which means performing a greater number of convolutional operations on every image is time and memory consuming. Hence the number of convolutional layers was decided to be 3. Convolutional layer extracts the features from the images. Neural Network required only edges of the lanes but not objects in the images. If the number of convolutional layers is increased, a greater number of features are extracted which are not required. The total number of trainable parameters is 3, 40,533 which is less than AlexNet and PilotNet. The size of J-Net is 3.4MB which is 5 times less than PilotNet and 15 times less than AlexNet. The below tables outline the architecture of the M-Net.
The goal of the lightweight model was achieved but the performance was not as good as PilotNet. J-Net initially made an error of 4% at the first epoch. It was believed that the model was going to overfit, but it eventually converged to the solution. The model could have overfitted if it was trained on a greater number of epochs. Though the performance was good enough to embed them into actual autonomous vehicles they can be embedded into small automotive gaming vehicles. The below Fig 16 shows the error made by the J-Net at each epoch and Fig 17 shows the test loss comparison for J-Net and Pilot Net.

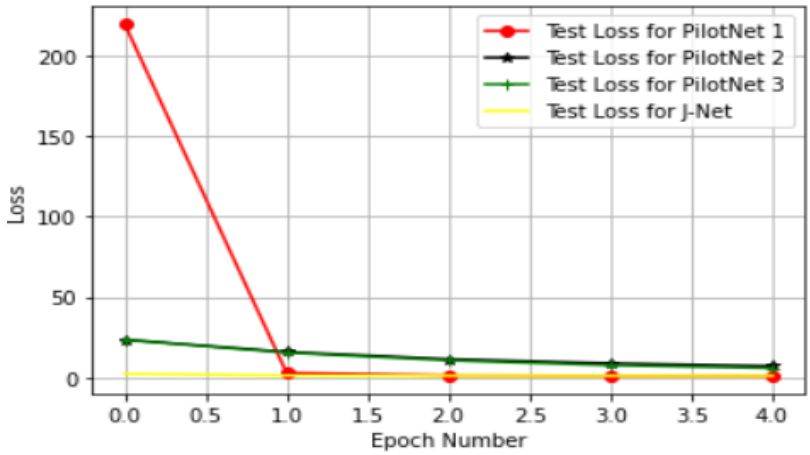

Figure 16: Train and Tess loss comparison for J-Net

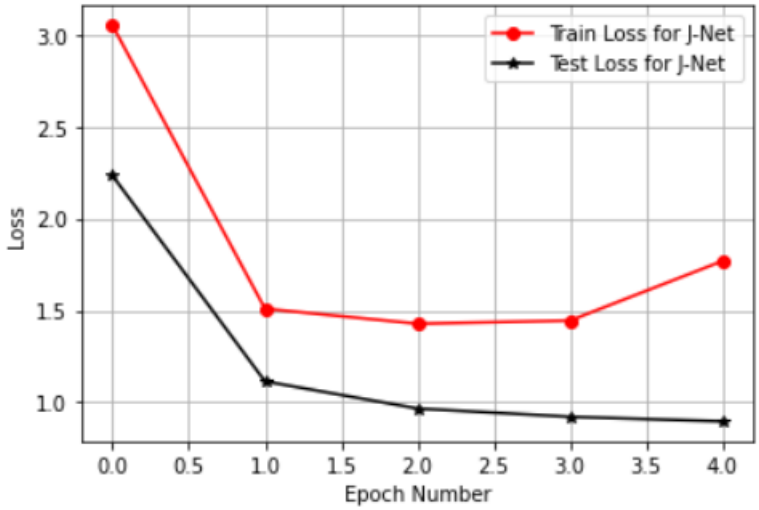

Figure 17: Tess loss comparison for J-Net and Pilot Net

The table 6 below gives a performance comparison of all the models:

Model Name	Size	Steering angle Prediction	Straight Road Prediction	Predicted Turns	Trained Time
Pilot Net	15MB	Perfect	Perfect	Perfect	8 hours
Alex Net	430MB	Bad	Bad	Bad	2 days
J-Net	3.5MB	Good	Good	Medium	3 hours

Table 7: Performance Comparison of models

Hence J-Net can be called as a Lightweight model/Inexpensive Model. The performance was still not as good as PilotNet, but its performance can be improved in future. The next section concentrates more on how the performance can be improved and what's the present use case with this inexpensive model, J-Net.
The below images show the predictions of PilotNet. There is not much difference in the predicted and the actual values. The error is around 0.5%

```
Predicted steering angle: 2.0680818160500727 (pred)     0.0 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     0.81 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     0.71 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     0.71 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -0.1 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     0.0 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     0.0 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -1.41 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -2.22 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -2.02 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -1.41 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -1.2100000000000002 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -0.91 (actual)
Predicted steering angle: 2.0680818160500727 (pred)     -0.81 (actual)
```

Figure 18: Predicted and True Values Pilot-Net

Layers and Parameters	PilotNet	AlexNet	J-Net
Convolutional Layers	5	5	3
Flatten layers	1	1	1
Max Pools	0	3	2
Dense (fully connected) Layers	4	3	2

Table 8: Comparison for CNN

Alex Net	Pilot Net	J-Net
34848 (5, 5, 96, 256) 4 5 5 96 256 614400 (3, 3, 256, 384) 4 3 3 256 384 884736 (3, 3, 384, 384) 4 3 3 384 384 1327104 (3, 3, 384, 256) 4 3 3 384 256 884736 (4608, 4096) 2 4608 4096 18874368 (4096, 4096) 2 4096 4096 16777216 (4096, 1000) 2 4096 1000 4096000 (1000, 1) 2 1000 1 1000 (96,) 1 96	4 5 5 36 48 43200 (3, 3, 48, 64) 4 3 3 48 64 27648 (3, 3, 64, 64) 4 3 3 64 64 36864 (1152, 1164) 2 1152 1164 1340928 (1164, 100) 2 1164 100 116400 (100, 50) 2 100 50 5000 (50, 10) 2 50 10 500 (10, 1) 2 10 1 10 (24,) 1 24 24 (36,) 1	(5, 5, 3, 16) 4 5 5 3 16 1200 (5, 5, 16, 32) 4 5 5 16 32 12800 (5, 5, 32, 64) 4 5 5 32 64 51200 (27520, 10) 2 27520 10 275200 (10, 1) 2 10 1 10

Table 9: Compared outputs of Models

By taking all the comparisons given above in Fig 20, into the consideration we can clearly tell that J-net is the light model that can be combined into the small-scale robots the following attributes are done for the J-net

1. Firstly, we have built a low-cost model to predict the steering angle.
2. Implemented 3 major architectures.

3. Reimplemented high-cost architectures used by the leading car companies now a days.
4. Compared all the outputs and results are presented.

6 Conclusion and Future Work

However, the light model has got success in the prediction of the steering angle as the Pilot Net where the main drawback is computation resources. When there are thousands of images it is taking more time to implement the code so GPU has been used to store and compile the data.

Autonomous vehicles rely heavily on neural networks and computer vision. Object detection, traffic signal prediction, and steering angle prediction are all built into self-driving cars. CNNs were created to process images and videos. CNNs are most commonly used for image classification tasks like detecting animals, humans, and other objects on the road. This study of predicting steering angle is a regression problem rather than a classification problem. The study focused on developing a deep learning model that predicts the vehicle's steering angle. Google Waymo, Tesla, and Uber have all put in a lot of effort in this area (Agarwal et al. 2019). The goal of this research was to create a low-cost, end-to-end deep learning model that can predict the steering angle of a vehicle. The automobile industry can benefit from the development of a low-cost system.

Multiple deep learning models were deployed and tested to accomplish this. Initially, a baseline model was created in order to ensure that the error made by CNN was less than the baseline model. Four different models were created based on NVIDIA's model. PilotNet was the first model, and it was an exact implementation of NVIDIA's model. The model predicted very accurately, with an error rate of less than 0.18 percent. An initial implementation included five convolutional layers, one flattened layer, and three fully connected layers. The learning rate was set to 0.0001 and worked flawlessly. The hyperparameters were later tweaked with the same model.

After training with 0.0001, the model was trained and tested with the Adam optimizer's default learning rate of 0.001. (pilotNet-2). This increased the training time, but the model was unable to learn any features and performed poorly. The learning rate was reduced because increasing the learning rate did not improve the model's performance. The learning was set to 0.00001 for the third trail (pilotNet- 3), so the model took longer to train and learned the features at a slower rate. It was trained for 5 epochs and had a 20% error rate. It would have taken longer to reach an optimal solution if it had been trained for a larger number of epochs.

Acknowledgment

First, I’m grateful for my supervisor Dr Catherine Mulwa for the valuable time and advice and support. By giving feedbacks she had supported and encouraged me for the whole project time period. The immense knowledge which shared by her have boosted me in completing my project within time.

Bibliography

Anderson, J.M., Kalra, N., Stanley, K.D., Sorensen, P., Samaras, C., Oluwatola, T.A. (2016) *Autonomous Vehicle Technology: A Guide for Policymakers*, RAND Corporation, available: https://www.rand.org/pubs/research_reports/RR443-2.html

Layne, J.R., Passino, K.M. (1993) 'Fuzzy model reference learning control for cargo ship steering', *IEEE Control Systems Magazine*, 13(6), 23–34.

Cao, C.-T., Becker, R., Belzner, U., Moeller, T.-W., Lieberoth-Leden, B. (1997) 'System for controlling brake pressure based on fuzzy logic using steering angle and yaw speed', available: https://patents.google.com/patent/US5634698/en

Lee, A.Y. (1989) 'A PREVIEW STEERING AUTOPILOT CONTROL ALGORITHM FOR FOUR-WHEEL-STEERING PASSENGER VEHICLES', *Advanced automotive technologies, 1989*, available: https://trid.trb.org/view/640502

Gidado, U.M., Chiroma, H., Aljojo, N., Abubakar, S., Popoola, S.I., Al-Garadi, M.A. (2020) 'A Survey on Deep Learning for Steering Angle Prediction in Autonomous Vehicles', *IEEE Access*, 8, 163797–163817.

End-to-End Deep Learning for Self-Driving Cars [online] (2016) *NVIDIA Developer Blog*, available: https://developer.nvidia.com/blog/deep-learning-self-driving-cars/

Du, S., Guo, H., Simpson, A. (2019) 'Self-Driving Car Steering Angle Prediction Based on Image Recognition', *arXiv:1912.05440 [cs, stat]*, available: http://arxiv.org/abs/1912.05440 [accessed 15 Dec 2021].

Hu, W., Lv, J., Liu, D., Chen, Y. (2018) 'Unsupervised Feature Learning for Heart Sounds Classification Using Autoencoder', *Journal of Physics: Conference Series*, 1004, 012002.

Kubo, Y., Tucker, G., Wiesler, S. (2017) 'Compacting Neural Network Classifiers via Dropout Training', *arXiv:1611.06148 [cs, stat]*, available: http://arxiv.org/abs/1611.06148

Wang, J., Mall, S., Perez, L. (n.d.) 'The Effectiveness of Data Augmentation in Image Classification using Deep Learning', 8.

Clevert, D.-A., Unterthiner, T., Hochreiter, S. (2016) 'Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)', *arXiv:1511.07289 [cs]*, available: http://arxiv.org/abs/1511.07289

MR-Contrast-Aware Image-to-Image Translations with Generative Adversarial Networks | SpringerLink [online] (2021) available: https://link.springer.com/article/10.1007/s11548-021-02433-x

Sokipriala, J. (2021) 'Prediction of Steering Angle for Autonomous Vehicles Using Pre-Trained Neural Network', *European Journal of Engineering and Technology Research*, 6(5), 171–176.

Autonomous Delivery Robot - PDF Free Download [online] (2021) available: https://docplayer.net/196819325-Autonomous-delivery-robot.html

Badue, C., Guidolini, R., Carneiro, R.V., Azevedo, P., Cardoso, V.B., Forechi, A., Jesus, L., Berriel, R., Paixão, T., Mutz, F., Veronese, L., Oliveira-Santos, T., De Souza, A.F. (2019)

‘Self-Driving Cars: A Survey’, *arXiv:1901.04407 [cs]*, available: http://arxiv.org/abs/1901.04407

Brougham, D., Haar, J. (2018) ‘Smart Technology, Artificial Intelligence, Robotics, and Algorithms (STARA): Employees’ perceptions of our future workplace’, *Journal of Management & Organization*, 24(2), 239–257.

Du, S., Guo, H., Simpson, A. (2019) ‘Self-Driving Car Steering Angle Prediction Based on Image Recognition’, *arXiv:1912.05440 [cs, stat]*, available: http://arxiv.org/abs/1912.05440

Daily, M., Medasani, S., Behringer, R., Trivedi, M. (2017) ‘Self-Driving Cars’, *Computer*.
Lee, A.Y. (1989) ‘A PREVIEW STEERING AUTOPILOT CONTROL ALGORITHM FOR FOUR-WHEEL-STEERING PASSENGER VEHICLES’, *Advanced automotive technologies, 1989*, available: https://trid.trb.org/view/640502

Pomerleau, D.A. (1989) ‘ALVINN: An Autonomous Land Vehicle in a Neural Network’, in *Advances in Neural Information Processing Systems*, Morgan-Kaufmann, available: https://proceedings.neurips.cc/paper/1988/hash/812b4ba287f5ee0bc9d43bbf5bbe87fb-Abstract.html

Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., Pazhayampallil, J., Andriluka, M., Rajpurkar, P., Migimatsu, T., Cheng-Yue, R., Mujica, F., Coates, A., Ng, A.Y. (2015) ‘An Empirical Evaluation of Deep Learning on Highway Driving’, *arXiv:1504.01716 [cs]*, available: http://arxiv.org/abs/1504.01716

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., Zieba, K. (2016) ‘End to End Learning for Self-Driving Cars’, *arXiv:1604.07316 [cs]*, available: http://arxiv.org/abs/1604.07316

Bimbraw, K. (2015) ‘Autonomous Cars: Past, Present and Future - A Review of the Developments in the Last Century, the Present Scenario and the Expected Future of Autonomous Vehicle Technology’:, in *Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics*, Presented at the 12th International Conference on Informatics in Control, Automation and Robotics, SCITEPRESS - Science and and Technology Publications: Colmar, Alsace, France, 191–198, available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0005540501910198

Williams, B.C., Sullivan, G. (2003) ‘16.410 / 16.413 Principles of Autonomy and Decision Making, Fall2003’, available: https://dspace.mit.edu/handle/1721.1/36896
Brougham, D., Haar, J. (2018) ‘Smart Technology, Artificial Intelligence, Robotics, and Algorithms (STARA): Employees’ perceptions of our future workplace’, *Journal of Management & Organization*, 24(2), 239–257.

H. Kato and T. Harada, "Image Reconstruction from Bag-of-Visual-Words," 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 955-962, doi: 10.1109/CVPR.2014.127.