"Recognition-Based Automobile Specification Application "

Submitted in partial fulfillment of the requirements of the degree of

Bachelor of Engineering

(Computer Engineering)

By

Rohith Kunnumakara

Sushmita Kushwaha

Omkar Lohar

Under the guidance of:

Prof. Jignesh Patel

AET'S

Atharva College of Engineering

Atharva Educational Complex,Malad Marve Road,

Charkop Naka,Malad(W),Mumbai-400095

(Affiliated to University of Mumbai)

April 2022

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Rohith

Rohith Kunnumakara

Sushmita

Sushmita Kushwaha

Lohar

Omkar Lohar

Date: 30/4/22

(Name and Signature of Students)

Project Report Approval for B.E.

This project report entitled "***Recognition-Based Automobile Specification Application***" by ***Rohith Kunnumakara , Sushmita Kushwaha , and Omkar Lohar*** is approved for the degree of ***Bachelor of Engineering in Computer Engineering.***

Internal Examiner

Bhau

Prof. Bhavna Arora

External Examiner

30/4/22

Dr. M. T. Patil

Date: 30/4/22

Place: Malad

ATHARVA COLLEGE OF ENGINEERING MUMBAI

College

Seal

ATHARVA
COLLEGE OF ENGINEERING

AET'S

ATHARVA COLLEGE OF ENGINEERING

CERTIFICATE

This is to certify that

Rohith Kunnumakara

Sushmita Kushwaha

Omkar Lohar

Have satisfactorily completed the requirements of the Synopsis

On

"Recognition-Based Automobile Specification Application"

As prescribed by the** University of Mumbai for academic year **2021-22

Under the guidance of

Prof. Jignesh Patel

Prof. Shweta Sharma	**Dr. Suvarna Pansambal**	**Dr. S.P.Kallurkar**
Project Coordinator	**HOD (Computer Engineering)**	**Principal**
(April 2022)	**College Seal**	**Date** 30/4/22

Acknowledgement

It is a matter of plensure of acknowledgment by indebtedness to our guide **Prof. Jignesh Patel** for his great and needy half in the completion of the project. He helped us and provided his valuable guidance at each and every step. We would also like to thank **Principal Dr. Shrikant Kallurkar, Prof. Shweta Sharma** and **Dr. Suvarna Pansambal**, Head of Computer Engineering Department for their deep interest, encouragement and facility provided to us during the course of our project.

Abstract

In modern day times, managing your personal and professional lives can be really hectic. If you don't have your own personal mode of transportation, life can become even more hectic but every functionality is conveniently available via smartphones.To make ones life easier,you should always find an easier and reliable mode of transportation. Purchasing a vehicle is perhaps one of the largest investments you will ever make, second to the purchase of your home. Therefore, your vehicle isn't just a vehicle, it's an investment. For such times where everything is just a click away, the Recognition-based Automobile Specification Application functions as an application on mobile phones that helps its users to identify the vehicles, such as cars, by capturing their image itself. We created an android application to help the user to determine which car is suitable.

This project renders information about the cars just by taking a clear image from any angle as an input. A person can get detailed information about the car whose photo he captures via the application. This information can include various details like the characteristics of the car, its features, its cost, its weight etc. The application would include processing on Cloud or on Device depending on the range of the application. It also allows the users to share their reviews regarding the car models. This application saves the time that we spend on unimportant trivialities and aims to detect and identify cars accordingly.

Index

List Of Figure

1. INTRODUCTION

1.1 Introduction Description

In this fast-paced world, managing your personal and professional lives can be really hectic. If you don't have your own personal mode of transportation, life can become even more hectic. To make your life easier, you should always find an easier and reliable mode of transportation. Purchasing a vehicle is perhaps one of the largest investments you will ever make, second to the purchase of your home. Therefore, your vehicle isn't just a vehicle, it's like property. In times of need, you can sell it for a less expensive model, and use the money for something useful. In times of success, you can trade it in for something that comes with even better features, benefits and looks. So, we create an android application to helps user determine which car is suitable.

1.2 Basic Concept

In now days, every functionality is conveniently available via smartphones. For such times where everything is just a click away, Recognition-based Automobile Specification Application the functions as an application on mobile phones that helps its users to identify the vehicles, such as cars, by capturing their image itself. This project renders information about the cars just by taking a clear image from any angle as an input. A person can get detailed information about the car whose photo he captures via the application. This information can include various details like the characteristics of the car, its features, its cost, its weight etc.

1.3 Application

The application would include processing on Cloud or on Device depending on the range of the application. It also allows the users to share their reviews regarding the car models. This application saves the time that we spend on unimportant trivialities and aims to detect and identify cars accordingly. So it helps to save users time to determine which car is suitable for him/her.

2. REVIEW OF LITERATURE

[1] Payam Goodarzi, Martin Stellmacher, Michael Paetzolad and Elmar Mathes, "Optimization of a CNN based Object Detector for fisheye Cameras"

Fisheye cameras are widely used in the automobile industry, due to their wide field of view for the environment. There are a lot of algorithms that researchers use for object detection, among them Convolutional Neural Networks (CNN) detectors have been widespread during the course of the last decade, however, they are mostly trained and applied on conventional pinhole camera images. In this paper, an attempt to optimize a CNN-based detector for fisheye cameras was made, taking into consideration the barrel distortion, which complicates the object detection. Several experiments were carried out to optimize one state-of-the-art detector, using synthetic fisheye effect on training dataset. The obtained result proves that fisheye augmentation can considerably advance a CNN-based detector's performance on fisheye images in spite of the distortion. Along the way, a new object detection framework based on the Tensorflow Estomator API was constructed to perform the experiments as convenient as possible.

[2] Rahul, Binoy B Nair, "Camera based Object Detection, Identification and Distance Estimation"

An automated system that can detect the presence of an obstacle, identify it as well as estimate its distance, is of great importance in ADAS (Advanced Driver Assistance Systems) applications that are now being increasingly deployed in vehicles. A few attempts have been made towards camera based distance estimation, image processing techniques were used to detect vehicles in the camera's field of vision and estimating distances. Stereo vision has been employed for robot navigation, stereo vision based detection and tracking of multiple objects in traffic has been presented. Monocular vision based cameras have also been used, but with limited accuracy. However, it is observed that traditional image processing algorithms have their limitations and hence, an attempt has been made in this study to combine state-of-theart deep learning techniques along with stereo vision to develop an embedded system based solution to the task of obstacle detection and distance estimation.

[3] Damir Demirovic, Emir Skejic , Amira Serfovic, "Performance of some images processing algorithms in TensorFlow"

Signal, image and Synthetic Aperture Radar imagery algorithms in recent time are used in a daily routine. Due to huge data and complexity, their processing is almost impossible in a real time. Often image processing algorithms are inherently parallel in nature, so they fit nicely into parallel architectures multicore Central Processing Unit (CPU) and Graphics Processing Unit GPUs. In this paper image processing algorithms were evaluated, which are capable to execute in parallel manner on several platforms CPU and GPU.

[4] Krizhevsky A, Sutskever I, Hinton GE, "ImageNet Classification with Deep Convolutional Neural Networks"

Machine learning is a system of artificial computer intelligence that provides computers with the ability to automatically learn without being programmed. In the healthcare sector, machine learning has been used to investigate skin cancer classification, to evaluate complex genetic interactions in autism, and to perform monitoring within the intensive care unit [1–3]. A recent study of diabetic retinopathy using deep machine learning revealed that machine learning exhibited high sensitivity and specificity for the detection of diabetic retinopathy.

[5] Arenado MI, Oria JMP, Torre-Ferrero C, Rentería LA, "Monovisionbased vehicle detection, distance and relative speed measurement in urban traffic"

A system combining deep learning and stereovision for detection, tagging and distance estimation of objects ahead, is presented in this study. A Convolutional Neural Network (CNN) is used to detect and identify objects in the field of vision of the stereo camera. Once the objects are detected and identified, their distance from the camera is estimated by first constructing 3D point cloud using triangulation method. The system is implemented on Nvidia® Jetson TX1 with a Zed® stereo camera, tested and found to be capable of a detection accuracy of around 84% with the average error in distance estimation less than 4.7% up to a distance of five meters

[6] Raza M, Chen Z, Rehman S, Wang P, Wang J-k, "Framework for estimating distance and dimension attributes of pedestrians in real-time environments using monocular camera"

Traffic accidents continue to increase in Korea as traffic increases, and the resulting loss of life is also on the rise. According to data surveyed by the South Korean National Police Agency,45,921 pedestrian traffic accidents were reported in 2019, resulting in 1487 deaths and 46,400 injuries. Due to the increased interest in traffic accident safety, the Advanced Driver Assistance System (ADAS) concept is rapidly developing and playing a significant role in coping with activities that are not recognized by the driver. Autonomous Emergency Braking (AEB), a representativeADAS system, is a system that is useful for preventing and mitigating accidents by braking vehicles in emergencies. For the study of AEBs' safety evaluation methods for pedestrians, a distance measurement method using a monocular camera with excellent accessibility, and measurement equipment required to validate data on the movement of vehicles, and a dummy to replace pedestrians are used. Based on the evaluation scenario considering the proposed Korea road environment, the relative distance obtained from equipment like DGPS and the relative distance using a monocular camera is compared and analyzed to verify safety. Comparative analysis shows that the minimum deviation is 2.3 cm, the third test result of 30 km/h of Car-to-Pedestrian Nearside Child (CPNC), and the maximum deviation is 25 cm, the first test result of 25 km/h of Car-to Pedestrian Nearside Adult (CPNA). The main factor in error generation is that the lane recognition in the camera image is not accurate, and the perception of small children is slow, which is why emergency braking is considered to have been slow. It is deemed that a safety assessment in weather conditions of adverse conditions will be required in the future.

[7] M. Abadi, P. Barham, J. Chen, at all."TensorFlow: a system for largescale machine learning"

TensorFlow is a machine learning system that operates at large scale and in heterogeneous environments. TensorFlow uses dataflow graphs to represent computation,shared state, and the operations that mutate that state. It maps the nodes of a dataflow graph across many machines in a cluster, and within a machine across multiple computational devices, including multicore CPUs, generalpurpose GPUs, and

custom-designed ASICs known as Tensor Processing Units (TPUs). This architecture gives flexibility to the application developer: whereas in previous "parameter server" designs the management of shared state is built into the system, TensorFlow enables developers to experiment with novel optimizations and training algorithms. TensorFlow supports a variety of applications,with a focus on training and inference on deep neural networks. Several Google services use TensorFlow in production, we have released it as an open-source project, and it has become widely used for machine learning research.In this paper, we describe the TensorFlow dataflow model and demonstrate the compelling performance that TensorFlow achieves for several real-world applications.

[8] S. Shi, Q. Wang, P. Xu and X. Chu, "Benchmarking State-of-the-Art Deep Learning Software Tools,"

Deep learning has been shown as a successful machine learning method for a variety of tasks, and its popularity results in numerous open-source deep learning software tools. Training a deep network is usually a very time-consuming process. To address the computational challenge in deep learning, many tools exploit hardware features such as multi-core CPUs and many-core GPUs to shorten the training time. However, different tools exhibit different features and running performance when training different types of deep networks on different hardware platforms, which makes it difficult for end users to select an appropriate pair of software and hardware. In this paper, we aim to make a comparative study of the state-of-the-art GPU-accelerated deep learning software tools, including Caffe, CNTK, MXNet, TensorFlow, and Torch. We first benchmark the running performance of these tools with three popular types of neural networks on two CPU platforms and three GPU platforms. We then benchmark some distributed versions on multiple GPUs. Our contribution is two-fold. First, for end users of deep learning tools, our benchmarking results can serve as a guide to selecting appropriate hardware platforms and software tools. Second, for software developers of deep learning tools, our in-depth analysis points out possible future directions to further optimize the running performance.

[9] Y. Kochura, S. Stirenko, O. Alienin, M. Novotarskiy and Y. Gordienko, "Comparative analysis of open source frameworks for machine learning with use case in single threaded and multi-threaded modes,"

This branch of AI involves the computer applications and/or systems design that based on the simple concept: get data inputs, try some outputs, build a prediction. Nowadays, ML has driven advances in many different fields[1]like pedestrian detection, object recognition, visual-semantic embedding, language identification, acoustic modeling in speech recognition, video classification, fatigue estimation[2], generation of alphabet of symbols for multimodal human-computer interfaces[3], etc. This success is related to the invention of more sophisticated machine learning models and the development of software platforms that enable the easy use of large amounts of computational resources for training such models.

[10] Z. Yang, Y. Zhu and Y. Pu, "Parallel Image Processing Based on CUDA,"

Single threaded applications cannot catch up with parallel systems when it comes to scalability mainly because of the clock frequency limitation of modern CPUs and economical reasons. Using shared memory or distributed memory architectures parallel systems can provide tremendous speed up compared to single threaded systems. CUDA, a shared memory parallel software, is considered to be a powerful language because of its easy thread management aspect and support for GPUs. Gauss blurring is a well-known image processing technique which reduces image noise and detail. Because of the high computation requirement of this technique, single threaded applications exhibit poor performance. In this paper we show that orders of magnitude speed up can be achieved by carrying out this operation on CUDA architecture with the help of high parallelism.

[11] I. K. Park, N. Singhal, M. H. Lee, S. Cho and C. Kim, "Design and Performance Evaluation of Image Processing Algorithms on GPUs,"

Deep learning is a very computational intensive task.Traditionally GPUs have been used to speed-up computations by several orders of magnitude. TensorFlow is a deep learning framework designed to improve performance further by running on multiple nodes in a distributed system. While TensorFlow has only been available for a little over a year, it has quickly become the most popular open source machine learning project on GitHub. The open source version of TensorFlow was originally only capable of running on a single node while Google's proprietary version only was capable of leveraging distributed systems. This has now changed. In this paper, we will compare

performance of TensorFlow running on different single and cloudnode configurations. As an example, we will train a convolutional neural network to detect number of cells in early mouse embryos.From this research, we have found that using a local node with a local high performance GPU is still the best option for most people who do not have the resources to design bigger system implementations.

[12] J. Lawrence,J. Malmsten, A. Rybka at all. "Comparing TensorFlow Deep Learning Performance Using CPUs, GPUs, Local PCs and Cloud."

Deep learning methods have resulted in significant performance improvements in several application domains and as such several software frameworks have been developed to facilitate their implementation. This paper presents a comparative study of four deep learning frameworks, namely Caffe, Neon, Theano, and Torch, on three aspects: extensibility, hardware utilization, and speed. The study is performed on several types of deep learning architectures and we evaluate the performance of the above frameworks when employed on a single machine for both (multi-threaded) CPU and GPU (Nvidia Titan X) settings. The speed performance metrics used here include the gradient computation time, which is important during the training phase of deep networks, and the forward time, which is important from the deployment perspective of trained networks. For convolutional networks, we also report how each of these frameworks support various convolutional algorithms and their corresponding performance. From our experiments, we observe that Theano and Torch are the most easily extensible frameworks. We observe that Torch is best suited for any deep architecture on CPU, followed by Theano. It also achieves the best performance on the GPU for large convolutional and fully connected networks, followed closely by Neon. Theano achieves the best performance on GPU for training and deployment of LSTM networks. Finally Caffe is the easiest for evaluating the performance of standard deep architectures.

3. REPORT ON PRESENT INVESTIGATION

It is developed for a particular system so we implement this algorithm for android phones. So we used TensorFlow Lite as TensorFlow is lightweight solution for mobile and embedded devices. It lets you run machine-learned models on mobile devices with low latency, so you can take advantage of them to do classification, regression or anything else you might want without necessarily incurring a round trip to a server.

4. AIM AND OBJECTIVES

The aim of this project is to eliminate the time we spend on looking and searching the information on the car. We can get the information of the car in just few clicks.

Objectives

- To develop a mobile application that will enable the user to click images of any interesting cars.
- Based on the image data received, our application will identify the car and show all relevant information about the car for the user.
- The information can include various details like the characteristics of the car, its features, its cost, its weight, images uploaded by other users, reviews.
- User will also be able to give feedback, review about the car.

5. PROBLEM STATEMENT

In now days car is important part of human life. To make your life easier, you should always find an easier and reliable mode of transportation. Therefore, your vehicle isn't just a vehicle, it's like property. So, it difficult to determine which car is suitable or gathering information of your future car. Because now days lot of cars available. User waste lot of time on unimportant trivialities and aims to detect and identify cars accordingly

6. PROPOSED SYSTEM

The Recognition-based Automobile Specification Application functions as an application on mobile phones that helps its users to identify the vehicles, such as cars, by capturing their image itself. This project renders information about the cars just by taking a clear image from any angle as an input. A person can get detailed information about the car whose photo he captures via the application. This information can include various details like the characteristics of the car, its features, its cost, its weight etc. The application would include processing on Cloud or on Device depending on the range of the application. It also allows the users to share their reviews regarding the car models. This application saves the time that we spend on unimportant trivialities and aims to detect and identify cars accordingly.

6.1 Algorithm

Convolutional Neural Networks

A convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.

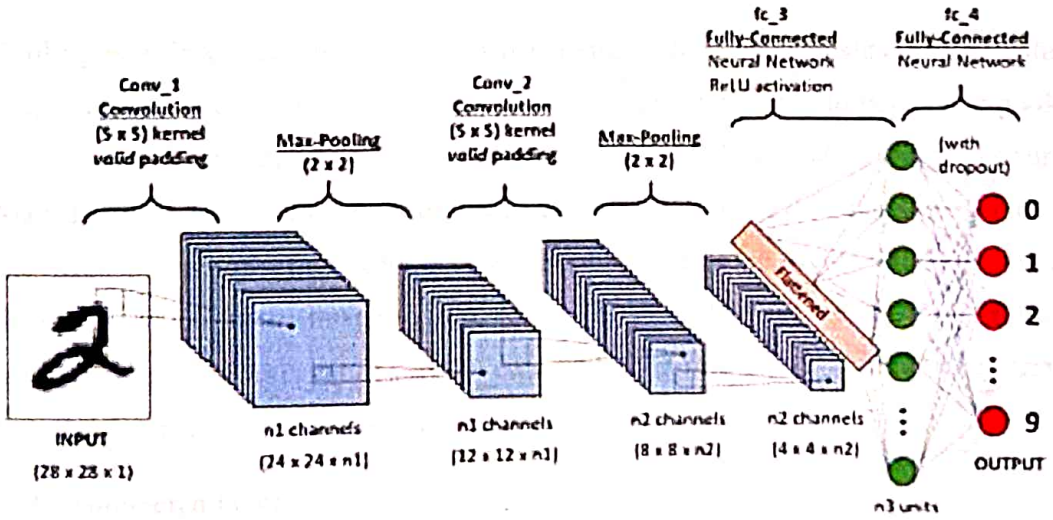

Figure No. 6.1(CNN)

A convolution neural network has multiple hidden layers that help in extracting information from an image. The four important layers in CNN are:

1. Convolution layer
2. ReLU layer
3. Pooling layer
4. Fully connected layer

Convolution Layer

This is the first step in the process of extracting valuable features from an image. A convolution layer has several filters that perform the convolution operation. Every image is considered as a matrix of pixel values.

ReLU layer

ReLU stands for the rectified linear unit. Once the feature maps are extracted, the next step is to move them to a ReLU layer. ReLU performs an element-wise operation and sets all the negative pixels to 0. It introduces non- linearity to the network, and the generated output is a rectified feature map.The original image is scanned with multiple convolutions and ReLU layers for locating the features.

Pooling Layer

Pooling is a down-sampling operation that reduces the dimensionality of the feature map. The rectified feature map now goes through a pooling layer to generate a pooled feature map. There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. The next step in the process is called flattening. Flattening is used to convert all the resultant 2- Dimensional arrays from pooled feature maps into a single long continuous linear vector.

Fully connected layer

The flattened matrix is fed as input to the fully connected layer to classify the image. Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that

space.Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training.

6.2 TensorFlow Lite

TensorFlow Lite is TensorFlow's lightweight solution for mobile and embedded devices. It lets you run machine-learned models on mobile devices with low latency, so you can take advantage of them to do classification, regression or anything else you might want without necessarily incurring a round trip to a server. It's presently supported on Android and iOS via a C++ API, as well as having a Java Wrapper for Android Developers. Additionally, on Android devices that support it, the interpreter can also use the Android Neural Networks API for hardware acceleration, otherwise, it will default to the CPU for execution. In this article, I'll focus on how you use it in an Android app. TensorFlow Lite is comprised of a runtime on which you can run *pre-existing* models and a suite of tools that you can use to prepare your models for use on mobile and embedded devices. It's not yet designed for training models. Instead, you train a model on a higher-powered machine, and then *convert* that model to the ". TFLITE" format, from which it is loaded into a mobile interpreter.

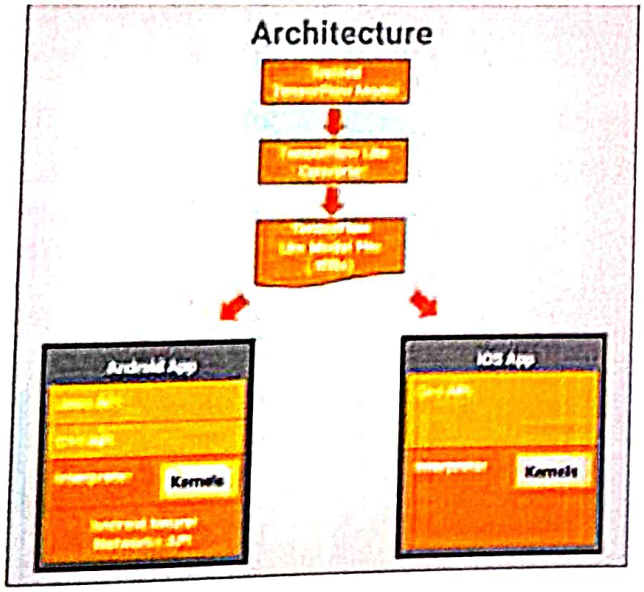

Fig No.6.2 (TensorFlow Lite Architecture)

6.3 Android Studio

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating system

Firebase

Firebase provides a Realtime database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud.

7. REQUIREMENT ANALYSIS (SRS)

7.1 Functional Requirements:

This section includes the requirements that specify all the fundamental actions of the software system.

7.1.1 Android User:

1. Download and install the app
2. Choose from a variety of algorithms to visualize
3. Add any required data or choose to generate random data for the algorithm
4. Visualize the algorithm

7.1.2 PC User

1. Visit the website using any browser
2. Choose from a variety of algorithms to visualize
3. Add any required data or choose to generate random data for the algorithm
4. Visualize the algorithm

7.2 Non-Functional Requirements:

Non-functional requirements are those that do not directly affect the functioning of the system but affect, the performance of the system. Non-functional requirements are those requirements, such as detail constraints, control mechanisms.

7.2.1. Performance Requirements

1. Performance
2. Safety
3. Reliability

7.2.2. Software Quality Attributes

1. Availability
2. Usability
3. Maintainability

8. SCOPE AND FEASIBILITY

8.1 Scope:

Nowadays, mobile technology has become important and it is easier than a personal computer or laptop, which is needed to connect to a specific location. In fact, mobile devices are becoming a natural part of daily life and routine because it is easier to carry all the time anywhere rather than a laptop or personal computer.

Further work will include increasing the accuracy of output by improving trained model by providing more dataset. Also increase the number of cars in future model.

8.2 Feasibility

8.2.1 Technical Feasibility:

Technical feasibility evaluates the technical complexity of the system. Java is being used in our project along with python training model . Firebase provides tools for tracking analytics, reporting and fixing app crashes, creating and product experiment.

8.2.2 Economic Feasibility:

Economic feasibility is a kind of cost-benefit analysis of the examined project, which assesses whether it is possible to implement it. We used android studio and Firebase which are free software .

9. METHODOLOGY

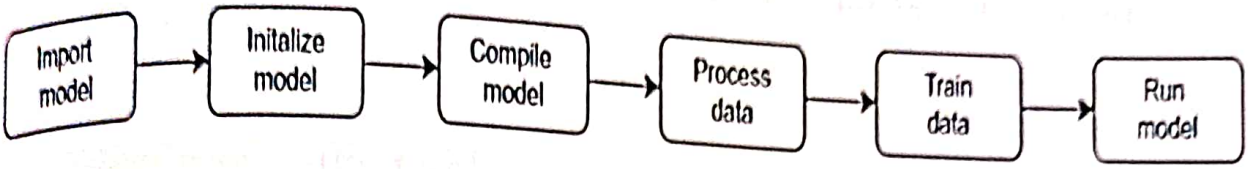

Figure 9.1: Methodology

- First user will login to the android application by entering email id and password. After successfully login user will capture photo of car and provide to the application.
- To identify image, we use concept of machine learning. So basically, we trained model for different cars to recognize the image or user input. To train module we used TensorFlow.
- We have used TensorFlow Lite for the backend to train the model.
- Android Studio is used for front end development using Adobe XD because as it is effective for making different types of layouts.
- Firebase is implemented as our Database.
- We used (CNN) algorithm to create the datasets or trained model. A convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks, based on their shared-weights architecture and translation invariance characteristics.
- Then TensorFlow model converted into TensorFlow lite format i.e. (. TFLITE). Then we create one classifier to classify the user input according to the model and provides output to the user.

There are 6 important steps:

- import model - TensorFlow library is imported.TensorFlow has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of- the-art in ML and developers easily build and deploy ML powered applications.

```
import tensorflow as tf
```

initialize model - Keras is used as a framework. In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

- Compile model - Two functions are used to compile the model.

  Loss function and optimizer which are used to guess the result and to get a good accuracy.

  Loss function: maps an event or values of one or more variables onto a real number representing some "cost" associated with the event. It's a method of evaluating how well your algorithm models your dataset. If our predictions are totally off, our loss function will output a higher number. If they're pretty good, it'll output a lower number.

  Optimizer:

  Optimization is a process of searching for parameters that minimize or maximize our functions .Optimizers shape and mold your model into its most accurate possible form by futzing with the weights.

- Data processing - In data processing, TensorFlow provides image data generator function. All the data set images are converted into one set and it labels them according to their folder name. It will rescale the images in one shape and will label the images according to their folder names.

- Training - To form the neural network according to the data, here we train our model according to the data set. This is where it will be making a guess, measuring how good or bad it is , using the optimizer to make another guess etc. It will do it for the number of epochs you specify.

10. PROJECT DESIGN

10.1 Control Flow Diagram

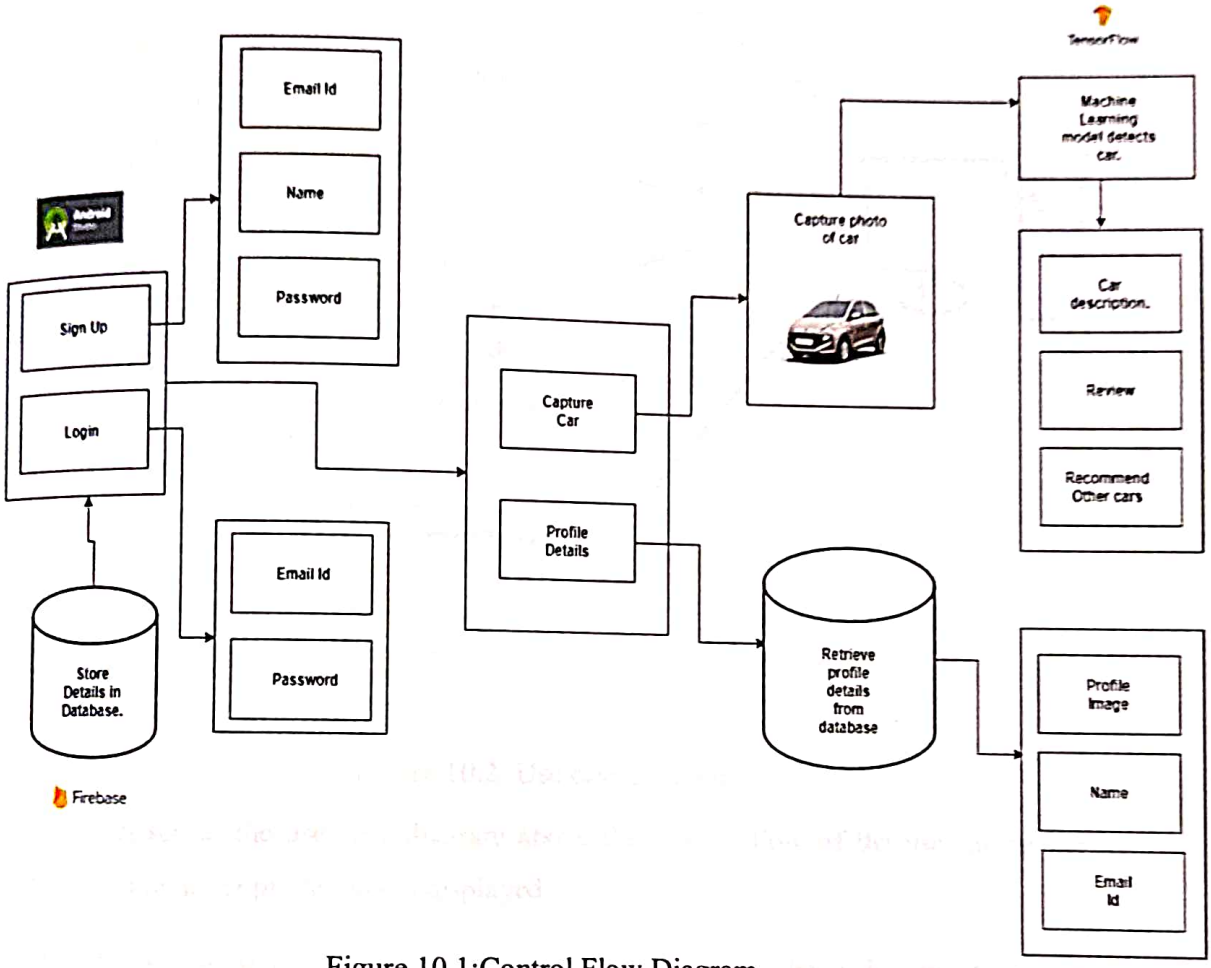

Figure 10.1:Control Flow Diagram

- First user will login to the android application by entering email id and password. After successfully login user will capture photo of car and provide to the application.
- To identify image, we use concept of machine learning. So basically, we trained model for different cars to recognize the image or user input. To train module we used TensorFlow.

10.2 Use Case Diagram

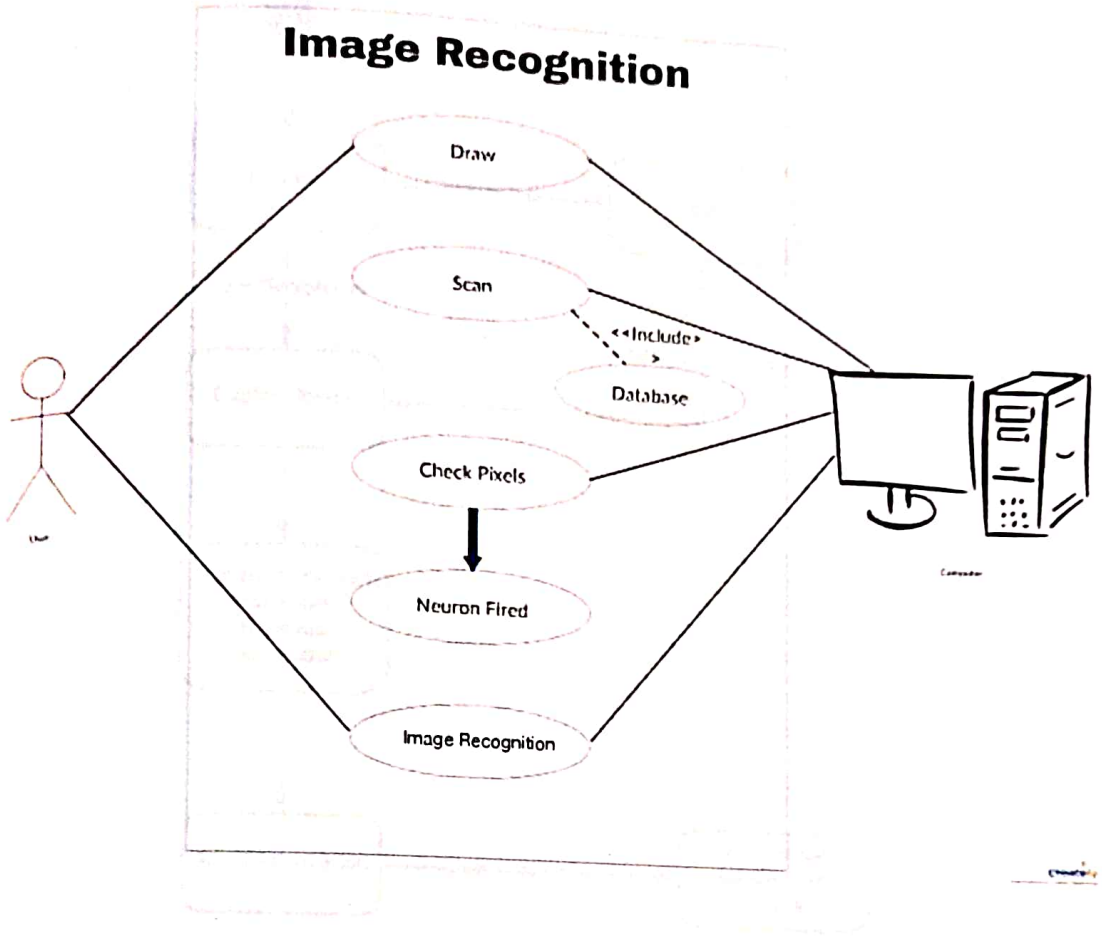

Figure 10.2: Use case Diagram

- As we can see in the use case diagram above the general flow of the user using the system and its inner processing is displayed.
- The systematic sequence of the machine is displayed in the above diagram from our initial input stage to the end state.

10.3 Activity Diagram

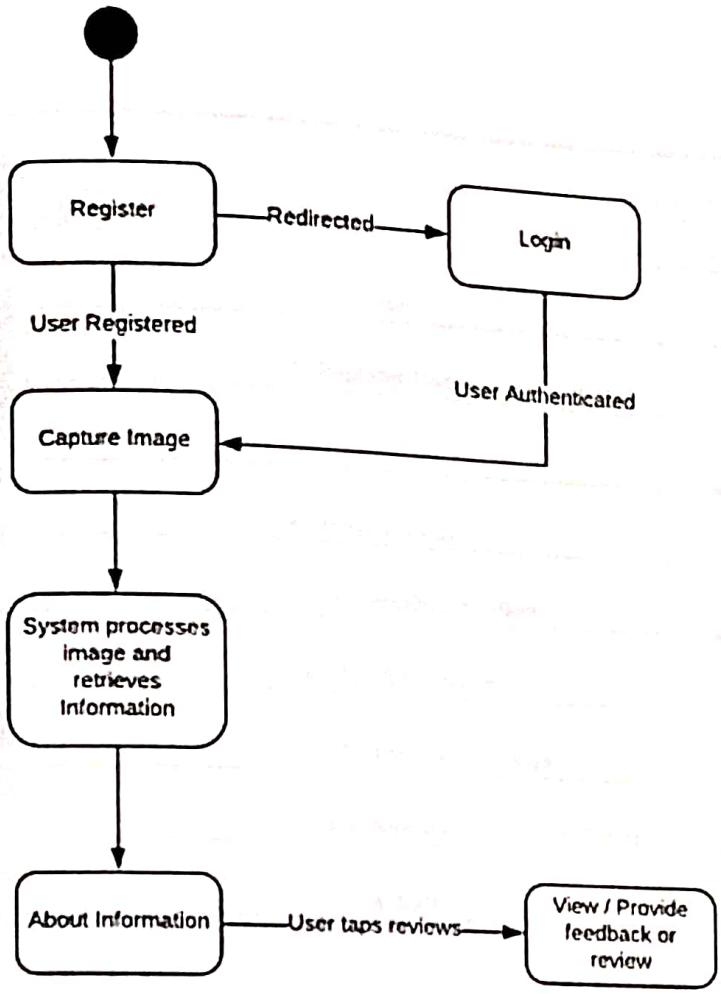

Figure 10.3: Acitvity Diagram

- The user starts off by registering on the app.
- If they have already registered then they are redirected to the login page.
- Once the user is registered or authenticated, they can capture the image of the car.
- The system then processes the image and retrieves the information based on that image and displays it for the user.
- The user can use the back button to go back to the Capture Screen to capture an image again if required.

10.4 Sequence Diagram

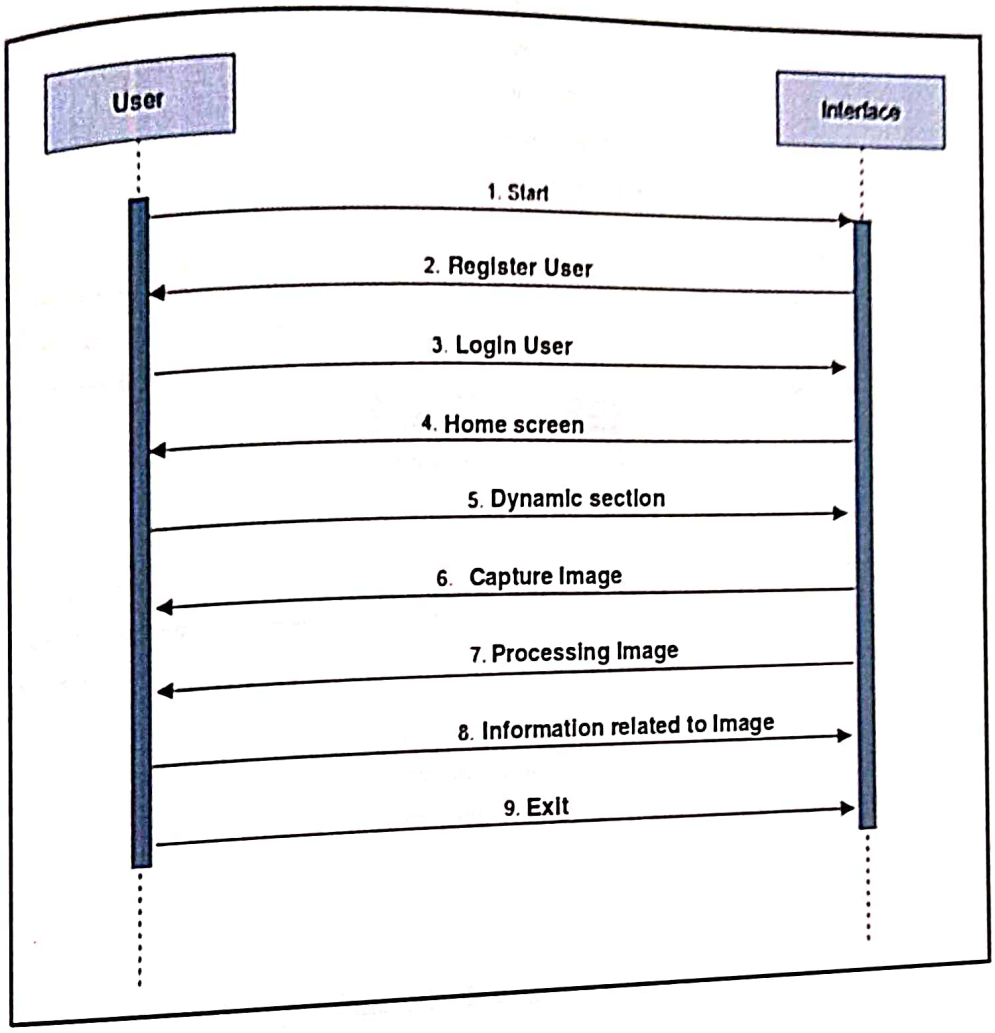

Figure 10.4: Sequence Diagram

In the sequence diagram, the objects according to their time of execution are listed below. The sequence diagram deals with how the user will flow through the system sequentially and what are the processes that will take place. The objects interact with each other by sending messages to each other.

10.5 Class Diagram

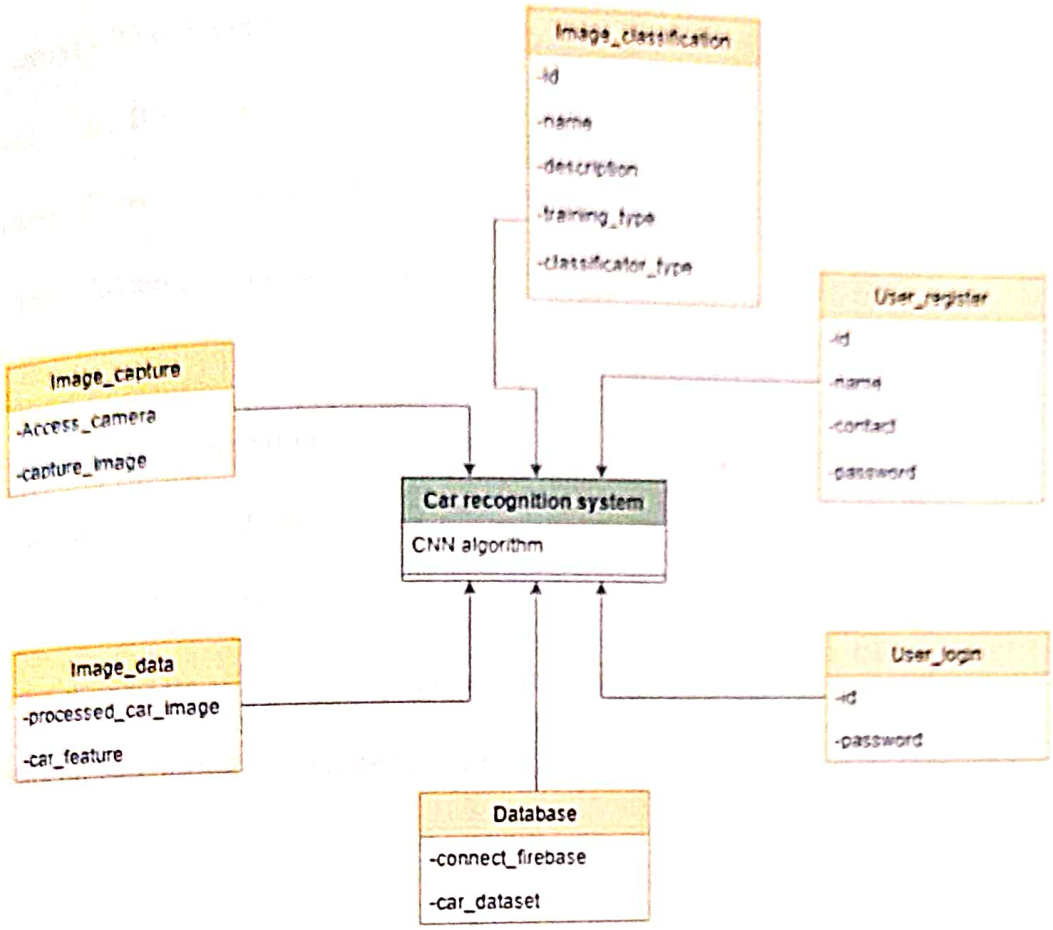

Figure 10.5: Class Diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.

11. IMPLEMENTATION AND HARDWARE SETUP

11.1 Hardware requirements

- Memory: - 20 GB and above.
- RAM : - 4 GB and above.
- Android Phone: - version (lollipop and above).
- Network Adapter : Ethernet Adapter

11.2 Software requirements

- Front End Tool : XML and Java
- Back End Tool : Firebase
- Operating System : OS X, Linux, Solaris and other platforms
- Platform: supporting a compatible JVM
- Python

11.3 Pseudo-Code

Following code is used for initialize the model with CNN :-

```
model = tf.keras.models.Sequential([
    # Note the input shape is the desired size of the image
300x300 with 3 bytes color

# This is the first convolution
tf.keras.layers.Conv2D(16,          (3,3),          activation='relu',
input_shape=(300, 3
00, 3)),
tf.keras.layers.MaxPooling2D(2, 2), # The second convolution
tf.keras.layers.Conv2D(32,          (3,3),          activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The third convolution
tf.keras.layers.Conv2D(64,          (3,3),          activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The fourth convolution
tf.keras.layers.Conv2D(64,          (3,3),          activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
# The fifth convolution
tf.keras.layers.Conv2D(64,          (3,3),          activation='relu'),
tf.keras.layers.MaxPooling2D(2,2),
#    Flatten    the    results    to    feed    into    a    DNN
tf.keras.layers.Flatten(),
#    512    neuron    hidden    layer    tf.keras.layers.Dense(512,
activation='relu'),
#    Only    3    output    neuron.    It    will    contain    3    classes)
tf.keras.layers.Dense(3, activation='sigmoid')
])
```

Sequential: That defines a sequence of layers in the neural network

Flatten: Flatten just takes that square shape images and turns it into a 1 dimensional set.

Dense: Adds a layer of neurons.

Relu effectively means "If X>0 return X, else return 0" -- so what it does it only passes values 0 or greater to the next layer in the network.

Softmax takes a set of values, and effectively picks the biggest one, so, for example, if the output of the last layer looks like [0.1, 0.1, 0.05, 0.1, 9.5, 0.1, 0.05, 0.05, 0.05], it saves you from fishing through it looking for the biggest value, and turns it into [0,0,0,0,1,0,0,0,0] -- The goal is to save a lot of coding.

Max Pooling returns the maximum value from the portion of the image covered by the Kernel.

11.4 Gantt Chart

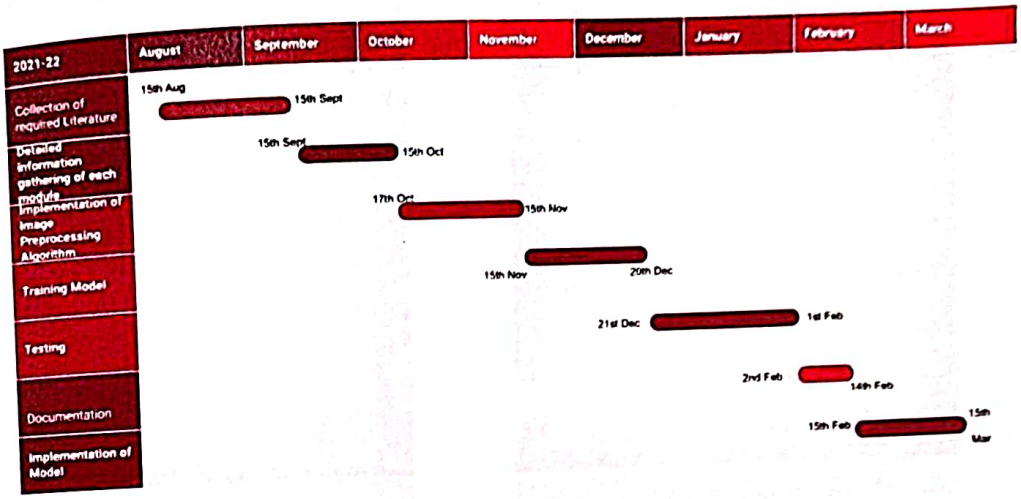

Figure 11.1: Gantt Chart

12. RESULTS

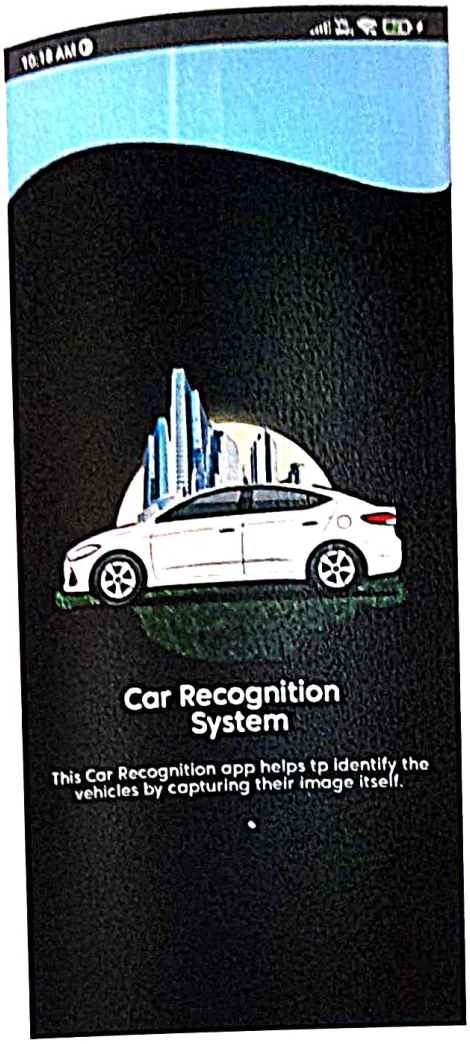

Fig 12.1 Splash Screen

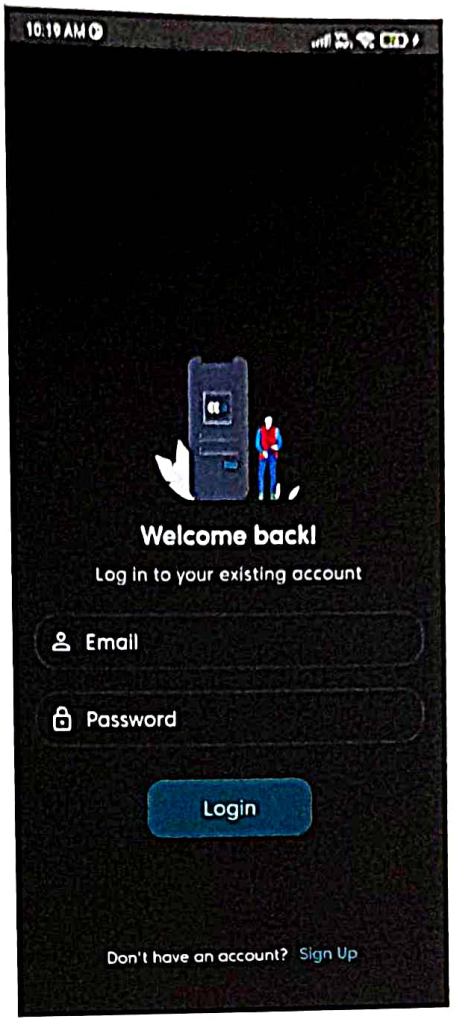

Fig 12.2 Login screen

Fig 12.3 Registration screen

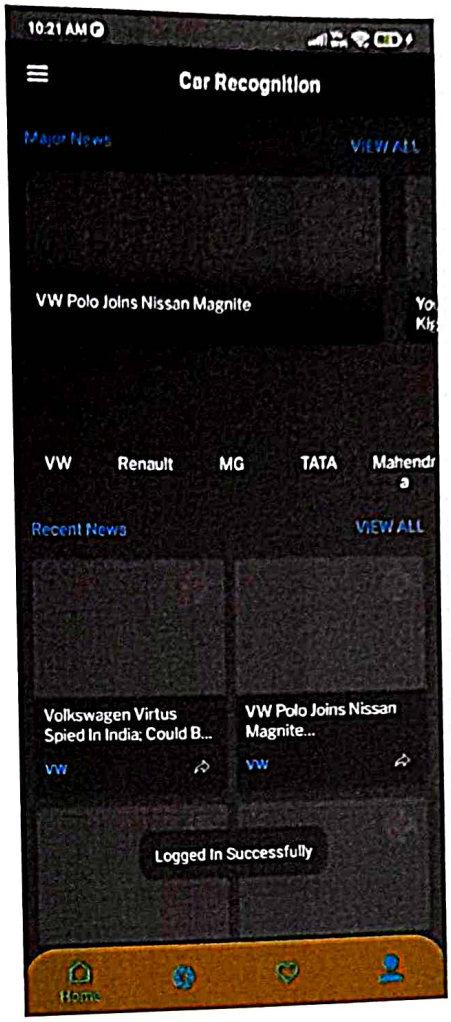

Fig 12.4 Home Screen

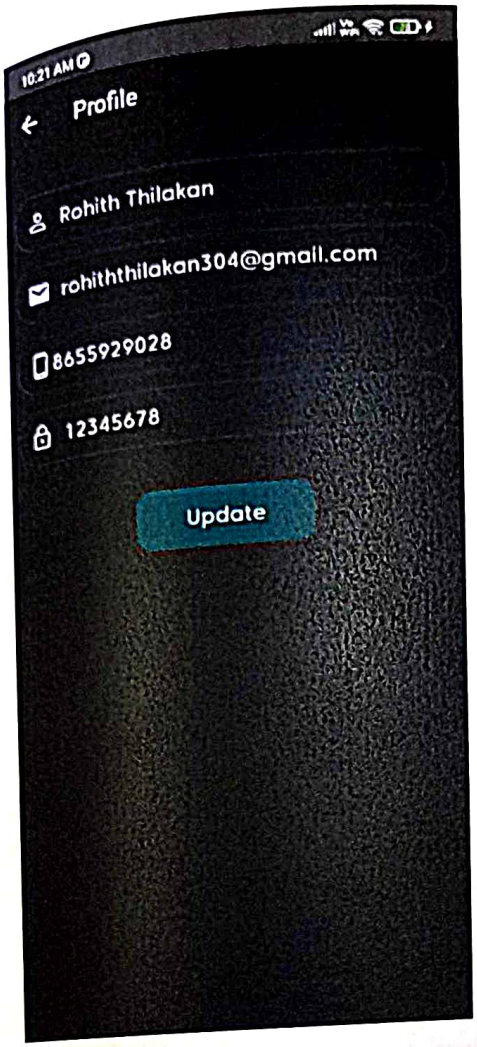

Fig 12.5 Profile image

Fig 12.6 Favourites list

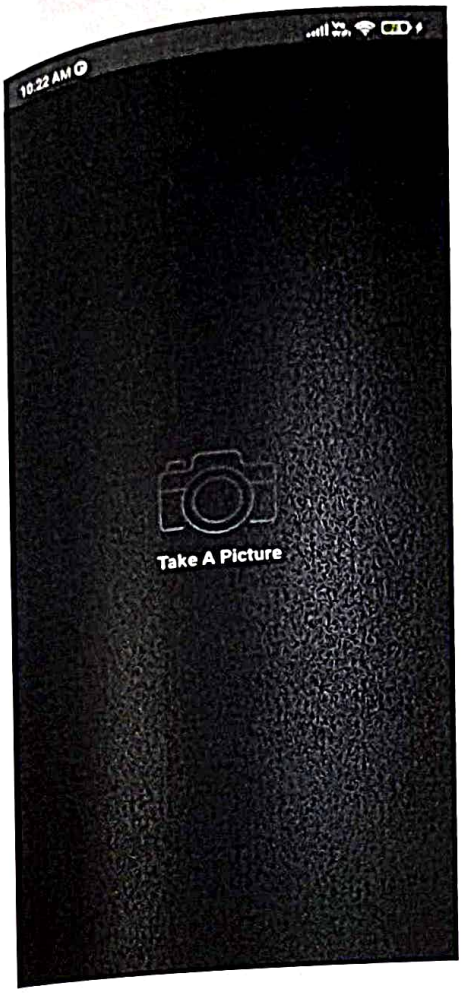

Fig 12.7 Capturing image

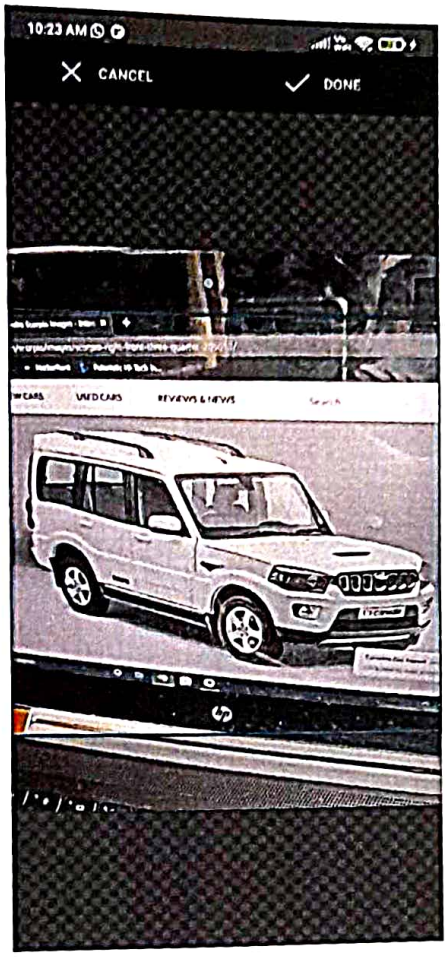

Fig 12.8 Crop image

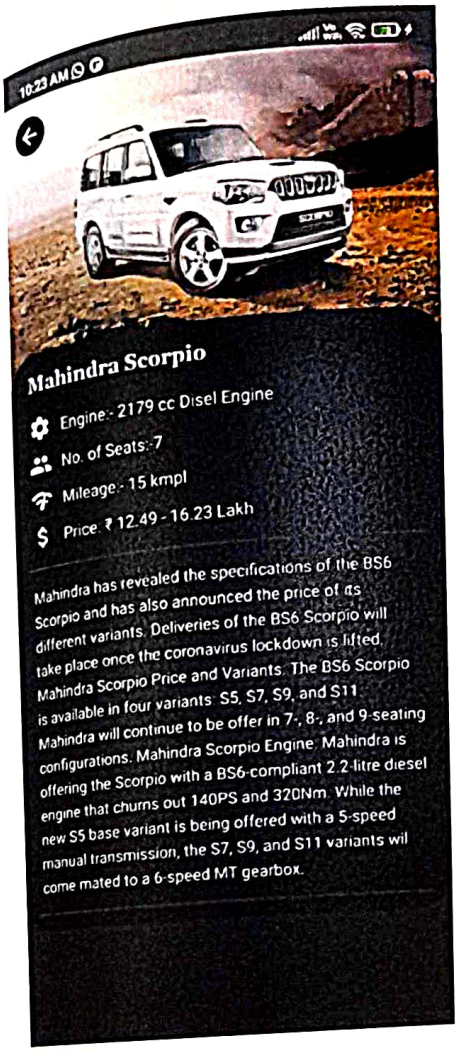

Fig 12.9 Final result

13. CONCLUSION

The project we are presenting has presented the process of image processing algorithms in TensorFlow Lite. Hence, we proposed the problem of developing a mobile application that renders information about the car just by taking their live pictures as inputs. In other words, the application should allow the user to click a photograph and based on the picture it should display information about the car. Further work will include increasing the accuracy of output by improving trained model by providing more dataset. Also increase the number of cars in future model. We also so think about some other features that helps user to identify the car and improve android layouts to make user friendly application.

14. REFERENCES

[1] Payam Goodarzi, Martin Stellmacher, Michael Paetzolad and Elmar Mathes. "Optimization of a CNN based Object Detector for fisheye Cameras", Vehicular Electronics, pp. 2278–2324, 2019.

[2] Rahul, Binoy B Nair, "Camera based Object Detection, Identification and Distance Estimation", Micro-electronics and Telecommunication Engineering, pp. 203-205, 2018,

[3] Damir Demirovic, Emir Skejic , Amira Serfovic, "Performance of some images processing algorithms in TensorFlow", computer vision and pattern recognition, pp, 799-778, 2018.

[4] Krizhevsky A, Sutskever I, Hinton GE, "ImageNet Classification with Deep Convolutional Neural Networks", Adv Neural Inf Process Syst, pp.1–9, 2012. doi:http://dx.doi.org/10.1016/j.protcy.2014.09.007.

[5] Arenado MI, Oria JMP, Torre-Ferrero C, Rentería LA, "Monovisionbased vehicle detection, distance and relative speed measurement in urban traffic", IET Intell Transp Syst, vol. 8, pp. 655–64, 2014.

[6] Raza M, Chen Z, Rehman S, Wang P, Wang J-k, "Framework for estimating distance and dimension attributes of pedestrians in real-time environments using monocular camera", Neurocomputing, vol. 275, pp. 533–545, 2018.

[7] M. Abadi, P. Barham, J. Chen, at all."TensorFlow: a system for largescale machine learning". In Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation (OSDI'16). USENIX Association, Berkeley, CA, USA, 265-283., 2016.

[8] S. Shi, Q. Wang, P. Xu and X. Chu, "Benchmarking State-of-the-Art Deep Learning Software Tools," 2016 7th International Conference on Cloud Computing and Big Data (CCBD), Macau, 2016, pp. 99-104.

[9] Y. Kochura, S. Stirenko, O. Alienin, M. Novotarskiy and Y. Gordienko, "Comparative analysis of open source frameworks for machine learning with use case in single threaded and multi-threaded modes," 2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT),

Lviv, 2017, pp. 373-376.

[10] Z. Yang, Y. Zhu and Y. Pu, "Parallel Image Processing Based on CUDA," 2008 International Conference on Computer Science and Software Engineering, Wuhan, Hubei, 2008, pp. 198-201.

[11] I. K. Park, N. Singhal, M. H. Lee, S. Cho and C. Kim, "Design and Performance Evaluation of Image Processing Algorithms on GPUs," in IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 1, pp. 91-104, Jan. 2011.

[12] J. Lawrence,J. Malmsten, A. Rybka at all. "Comparing TensorFlow Deep Learning Performance Using CPUs, GPUs, Local PCs and Cloud." 2017.

[13] S. Bahrampour, N. Ramakrishnan, L. Schott. And M. Shah, M. Comparative Study of Caffe, Neon, Theano, and Torch for Deep Learning". CoRR, abs/1511.06435. 201

15. PAPER PUBLICATION DETAILS

Title:

Recognition based Automobile Specification

Date of Publication:

April, 2022

Journal Name:

IRJMETS Publications

Name of Authors:

Rohith Kunnumakara [1], Sushmita Kushwaha [2], Omkar Lohar [3], Prof. Jignesh Patel [4]

Affiliation of authors:

[1], [2], [3] Student, Department of Computer Engineering, Atharva College of Engineering, Mumbai.

[4] Professor, Department of Computer Engineering, Atharva College of Engineering, Mumbai.