Road Accident Management System

PROJECT REPORT

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY (HONS.)

(Department of Computer Science & Engineering)

Submitted by :

Kushagra Upadhyay (2018UGCS020R)

Rishabh Kumar (2018UGCS042R)

Neha Kumari (2018UGCS056R)

Under the Supervision of

Dr. Nidhi Kushwaha

(Faculty, CSE, IIIT Ranchi)

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, RANCHI

(An Institution of National importance under act of Parliament)

(Ranchi - 834010), Jharkhand

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, RANCHI – 834010 (JHARKHAND), INDIA

Ranchi

Date: 12 Feb 2022

CERTIFICATE

This is to certify that the project titled "***ROAD ACCIDENT MANAGEMENT SYSTEM USING CONVOLUTIONAL NEURAL NETWORK***" is a record of the bonafide work done by KUSHAGRA UPADHYAY (2018UGCS020R), RISHABH KUMAR (2018UGCS042R) and NEHA KUMARI (2018UGCS056R) submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology (Hons.) in Department of Computer Science and Engineering of Indian Institute of Information Technology Ranchi, during the academic year 2021-22.

Dr.Nidhi Kushwaha
Project Guide, Deptt. of CSE
Indian Institute of Information Technology Ranchi

Dr.Shashi Kant Sharma
Faculty In-Charge: Academics
Indian Institute of Information Technology Ranchi

ACKNOWLEDGEMENT

We would like to express our gratitude to our supervisor Dr.Nidhi Kushwaha who gave us the golden opportunity to do this wonderful project on the topic, ***"ROAD ACCIDENT MANAGEMENT SYSTEM USING CONVOLUTIONAL NEURAL NETWORK",***

This project not only helped us in learning a lot of things, but also taught teamwork, time management and many more soft skills, for which we are really thankful to the institute.

We would also like to thank our friends who helped us in finalizing this project within the limited time frame.

We are overwhelmed in all humbleness and gratefulness to acknowledge each and every one who has helped us put these ideas, well above the level of simplicity and into something concrete.

Any attempt at any level cannot be satisfactorily completed without the support and guidance of other faculty members.

Thank you.

Rishabh Kumar (2018UGCS042R)
Kushagra Upadhyay (2018UGCS020R)
Neha Kumari (2018UGCS056R)

ABSTRACT

ACMS helps us to construct a smart vehicle system with minimizing the limitations of existing methods and also enhancing the security of vehicles and human beings by reducing accidental injuries. Smart vehicle system will entail a speed and other parameters of vehicle sensing mechanism which automatically messages to personal contacts with the details of vehicle position when an accident occurs using the GPS system. It is important to make an analysis of problems, and what we have . we make a proper analysis on every problem and try to resolve them. Our project is also to remove some of its negative aspects, as we thought that no one is perfect. We proposed an integrated system that has inbuilt modules with an interface. Using a Minimum number of sensors leads to fewer failures.

The Accident Control Management System is implemented using:

CNN,Faster R-CNN, R-FCN, SSD, and YOLO V2.

LIST OF FIGURES

CONTENTS

Chapter - 1

Accident Control Management System

INTRODUCTION

Motivation

As cities across the world grow and the mobility of populations increases, there has also been a corresponding increase in the number of vehicles on roads. One of the main elements that smart cities seek to contain is the issue of increasing levels of road accidents, which have resulted from increasing numbers of vehicles, leading to congestion.

The motivation behind choosing this topic is drawn from this situation. We have designed algorithms to solve this pathos using the concepts of machine learning and image processing with maximum possible accuracy.

Objectives:

- The main objective of this Accident Control Management System is to provide an efficient system to resolve the most common real world problems which are road accidents.

- This system will provide an efficient solution to all possible reasons of road accidents like drunk and drive, wrong lane, nearby car, pothole e.t.c.

- This system also includes its negative aspects and tries to resolve them.

Chapter - 2

Road Accident Statistics in India

India has a well-knit and coordinated system of transport which plays an important role in development of economic activities by promoting fair distribution of produced goods and services and movement of people. The share of the transport sector in Gross Domestic Product (GDP) of India is steadily growing. It is one of the key indicators in assessment of socio-economic development of the country. Since traffic accidents are an indicator of bottlenecks and other hindrances in smooth flow of traffic, hence NCRB collects detailed data on traffic accidents including road accidents for inferring the trend and patterns of traffic accidents for the planners to devise appropriate preventive strategies.

The Bureau collects data on 'Traffic Accidents' consisting of (i) Road Accidents (ii) Railway Accidents and (iii) Railway Crossing Accidents, as these are the major contributors of accidental deaths. Number of 'Traffic Accidents' in the country have decreased from 4,74,638 in 2018 to 4,67,171 in 2019. (However, the rate of deaths in road accidents per thousand vehicles i.e. 0.6 has remained same as it was in 2018). Maximum increase in number of traffic accidents cases in States from 2018 to 2019 it was reported in Madhya Pradesh (from 49,080 to 53,379) followed by Rajasthan (from 22,401 to 24,281) and Uttar Pradesh (from 40,783 to 42,368). On the other hand, the maximum decrease was reported in Tamil Nadu (from 66,110 to 59,499) .

These traffic accidents resulted in injuries to 4,42,996 persons and 1,81,113 deaths during 2019. State of Uttar Pradesh (27,661 deaths) followed by Maharashtra (18,524 deaths) and Madhya Pradesh (13,497 deaths) have reported maximum fatalities in traffic accidents in the country; these 3 States accounted for 15.3%, 10.2% and 7.5% of total deaths in traffic accidents respectively and collectively accounted for 33.0% (59,682 out of 1,81,113) of total fatalities reported at all India level during 2019.

The percentage share of traffic accidental deaths in total deaths due to 'Other Causes' has increased from 42.9% in 2015 to 43.9% in 2019. A rising trend was seen in the absolute number of deaths in 'Traffic Accidents' from 2015 to 2019. Number of deaths have increased by 1.3% (from 1,78,832 to 1,81,113) in 2019 over 2018. These traffic accidents comprise of 4,37,396 road accidents, 27,987 railway accidents and 1,788 railway crossing accidents and caused 1,54,732, 24,619 and 1,762 deaths respectively during 2019.

Growth in Number of Vehicles and Road Accidents in India (2015–2019)

Sl. No.	Year	Road Accidents (in thousand)	% Variation over Previous Year	Persons Injured (in thousand)	% Variation over Previous Year	Persons Killed (in Nos.)	% Variation Over Previous Year	No. of Vehicles (In Thousand)#	% Variation over previous Year	Rate of Deaths per thousand Vehicles (Col.7/Col.9)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
1	2015	464.7	3.1%	482.4	1.0%	1,48,707	5.1%	2,10,023	15.1%	0.71
2	2016	473.0	1.8%	485.5	0.6%	1,51,801	2.1%	2,30,031	9.5%	0.66
3	2017	445.7	-5.8%	456.2	-6.0%	1,50,093	-1.1%	2,53,311	10.1%	0.59
4	2018	445.5	-0.0%	446.5	-2.1%	1,52,780	1.8%	2,53,311*	-	0.60
5	2019	437.4	-1.8	439.2	-1.6	1,54,732	1.3	2,53,311*	-	0.61

'# Source: Road Accidents in India - 2018, TRW, MoRT&H, as per latest published data.

'* figures of the previous year used due to non-availability of data.

- As per data provided by States/UTs.

Fig 1 : Growth In Number of Vehicles and Road Accidents In India (2015-2019)

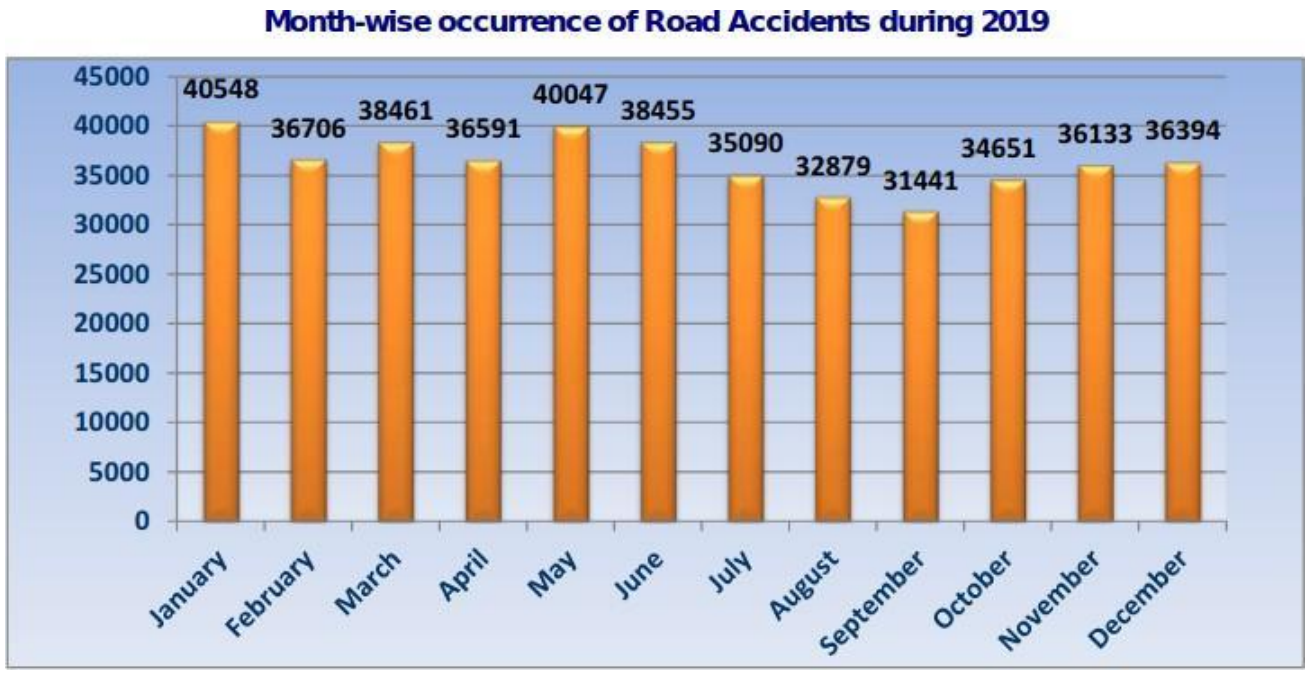


Fig 2: Month-wise Occurrence of Road Accidents during 2019

Common Causes of Road Accidents in India

Road accidents are unfortunately prevalent in India, and the majority of these accidents are caused by human fault. It is imperative to be cautious, drive carefully and follow traffic rules. However, even if you are careful on the streets does not mean that the other drivers are too. There are several incidents when the drivers are not wrong, and it's the pedestrian or the other vehicle that creates the havoc.

Listed down are the most common causes of road accidents in India -

Red Light jumping / Breaking Traffic Signals

It is a common sight at road intersections that vehicles cross without caring for the light. The main motive behind Red light jumping is saving time. The common conception is that stopping at a red signal is a waste of time and fuel. Studies have shown that traffic signals followed properly by all drivers saves time and commuters reach their destination safely and timely. A red light jumper not only jeopardizes his life but also the safety of other road users. This act by one driver incites another driver to attempt it and finally causes chaos at the crossing. This chaos at intersections is the main cause of traffic jams. Eventually everybody gets late to their destinations. It has also been seen that the red light jumper crosses the intersection with greater speed to avoid crashes and challan but it hampers his ability to judge the ongoing traffic and quite often crashes.

Drunk Driving / Drowsiness

Consumption of alcohol to celebrate any occasion is common. But when mixed with driving it turns celebration into a misfortune. Alcohol reduces concentration. It decreases the reaction time of the human body. Limbs take more time to react to the instructions of the brain. It hampers vision due to dizziness. Alcohol dampens fear and incite humans to take risks. All these factors while driving cause accidents and many times it proves fatal. For every increase of 0.05 blood alcohol concentration, the risk of an accident doubles. Apart from alcohol, many drugs, medicines also affect the skills and concentration necessary for driving. First of all, we recommend not to consume alcohol. But if you feel your merrymaking is not complete without booze, do not drive under the influence of alcohol. Ask a teetotaler friend to drop you home.

Tailgating

No matter how frustratingly slow the cars in front of you must be going, there's no excuse to get too close to them. Keep a safe distance from other vehicles so that you have enough time to react to sudden turns. All these reasons sound very basic, but they are the significant reasons behind accidents on Indian roads. Accidents come with a whole set of drawbacks Having your vehicle insured can be a breeze of relief for you. Therefore, we recommend you to have a valid car insurance or two wheeler insurance policy for your four-wheelers and bikes respectively. In case your car insurance or two wheeler insurance policy has expired, renew it within minutes online.

Few more causes of Road Accidents :-

Potholes

Potholes are bowl-shaped openings in the road that can be up to 10 inches deep and are caused by the wear-and-tear and weathering of the roads. They occur when the top layer of the road, the asphalt, has worn away and exposed the concrete base. Potholes put a huge strain on your car’s suspension and shocks (which absorb most of the impact of bumps and potholes). It can cause expensive damage to your car and cause you to make an unexpected appointment with the auto mechanic. They can also cause an impact similar to that of a 35-mph car accident, if deep enough.

Distracted Driving

One of the most common causes of road accident is distracted driving. The number of accidents occurring due to distracted driving has increased in the past decades. Undivided attention while driving is a must and drivers should inculcate the same within them. Reading messages, replying to texts, taking calls, reading, grooming, etc. behind the wheel can be fatal.

Speeding/ Reckless Driving

It is quite tempting to push the accelerator and increase the speed when you are running late or are driving on an empty road. Speeding increases the intensity of accidents and is proved to be fatal most of the time. Hence, it is advisable to drive within the legal limits even when you are running late.

Reckless driving mostly leads to horrible accidents. Take your time and remain calm behind the wheels to avoid unwanted accidents caused by mere neglect.

Rain or Wet Roads

Wet roads are slippery, and hence they can be fatal for the vehicles as the wheels lose their traction on wet roads. While you can't always avoid driving in the rain, the slippery streets should be best avoided whenever possible. Also, when the visibility is too low, you should pull over and wait until the rain subsides.

Unsafe Lane Changing

Six million accidents happen every year in India. Unfortunately, many of these accidents happen for unknown reasons. Despite this lack of knowledge concerning certain types of crashes and what drive them, we’re still able to pinpoint the exact causes of at least certain collisions.One of the most common causes of accidents is the act of changing lanes.According to official statistics, at least 33 percent of all crashes happen when vehicles change lanes or veer off the road.

CHAPTER-3

LITERATURE SURVEY

The proposed system in this report aims at reducing the loss of lives due to traffic accidents ***[International Journal of Scientific & Engineering Research, Volume 7, Issue 4, April-2016]*** and performs **four** main tasks :

(1) Detecting the current traffic signal by training our model using the COCO dataset. YOLO V2 and SSD Mobilenet merit a special mention, in that the former achieves competitive accuracy results and is the second fastest detector, while the latter, is the fastest and the lightest model in terms of memory consumption, making it an optimal choice for deployment in mobile and embedded devices. **[*Alvaro Arcos-Garcıa, Juan A. Alvarez-Garcıa, Luis M. Soria-Morillo, Evaluation of Deep Neural Networks for traffic sign detection systems, Neurocomputing (2018), doi: https://doi.org/10.1016/j.neucom.2018.08.009.*]** The evaluation and comparison of these models include mean average precision (mAP), memory allocation, running time, number of floating point operations, number of parameters of the model, and the effect of traffic sign image sizes.

(2) Identifying the proximity of nearby vehicles by object detection algorithm. It would prevent head on collision with vehicles on the highway, freeway etc. Here we are going to use **MobileNet** architecture, an efficient CNN architecture designed for mobile and embedded vision applications. *[**"Convolutional Neural Networks (MobileNet) – DeepLearning 0.1 documentation". DeepLearning 0.1. LISA Lab. Retrieved 31 August**].* This architecture uses proven depth-wise separable convolutions to build lightweight deep neural networks.

(3) Monitoring driver's drowsiness using a Computer vision system. *A computer vision system that can automatically detect driver drowsiness in a real-time video stream and then play an alarm if the driver appears to be drowsy.* [***"The most accurate method is based on human physiological activity",Driver Drowsiness Detection System Using Computer Vision, IRJET Volume: 07 Issue: 01 | Jan 2020***] Those systems shall also be designed to avoid overlap and shall not prompt the driver separately and concurrently or in a confusing manner where one action triggers both systems.(-regulation (EU) 2019/2144) ***This system is also known as EyeSight Driver Assist,Tiredness Detection Warning, AntiSleep Pilot, Fatigue detection system***.

(4) Our last proposed solution would be the system pothole detection system (PDS). It will use the mobility of the particular vehicle on which the system will be fitted, and side by side gather data from the vibrations and the GPS sensors, and further process and filter the data to monitor road surface condition *[**"A real-time pothole detection approach for intelligent transportation system", Mathematical Problems in Engineering 2015, 2015**.]*. By evaluating the R-CNN algorithm and SSD Mobilenet algorithm, the results of the test showed successful results in getting potholes from test images with a maximum confidence level of 93%.

["Pothole Detection System Using Region-Based Convolutional Neural Network," 2021 IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET), 2021, pp. 6-11]

(5) It is a natural psyche of humans to excel. But when we are sharing the road with other users we will always want to take a control. Increase in speed multiplies the risk of accident and severity of injury during accident. In our project we have used GSM/GPRS modem to locate the exact position of the vehicle. GPS is used for tracking the position of the vehicle, GSM is used for sending the message. ***["GPS beneficiary has turned into a vital piece of a vehicle. Other than utilizing as a part of different purposes, the GPS can likewise screen the speed and Distinguish a mishap" , Iot Based Automatic Accident Vehicle Location Sharing System , INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 8, ISSUE 10, OCTOBER 2019 ISSN 2277-8616]***

These models would help in minimizing the limitations of existing models and also enhancing the security of vehicles and human beings by reducing accidental injuries. This model will be useful for less memory space and gives more accurate results. Most models of this project use the CNN for processing the images.

Chapter - 4

METHODOLOGY

4.1 Detailed Methodology

Driving is a very sensitive activity that needs to be done very consciously and with high care. One should follow traffic rules and take all safety precautions while driving on the road. Recent studies show how communication capabilities should be supported by an Artificial Intelligence system capable of automating many of the decisions to be taken by emergency services, thereby adapting the resources of the rescue to the severity of the accident and reducing assistance time. The prevention method is the best method for accident avoidance. This project proposes a novel intelligence system, which is able to automatically detect the major factor of a road accident and if one of them is in action then notify us through alert messages and alarms. Our system will prevent accidents and will notify us to do better action in that situation. This project is a combination of two independent modules, first is the hardware module in which all systems will work together and managed by a master integrated board and the other is the software module.

Most of the software part of this system will work on CNN (convolutional neural network). The name "convolutional neural network" indicates that the network employs a mathematical operation called convolution. Convolutional networks are a specialized type of neural networks that use convolution in place of general matrix multiplication in at least one of their layers.
A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution. In a convolutional neural network, the hidden layers include layers that perform convolutions. Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix. This product is usually the Frobenius inner product, and its activation function is commonly ReLU. As the convolution kernel slides along the input matrix for the layer, the convolution operation generates a feature map, which in turn contributes to the input of the next layer. This is followed by other layers such as pooling layers, fully connected layers, and normalization layers.
Image processing will play an important role for this project because every model is trained on preprocessed models and after preprocessing images are stored in less memory in multidimensional array format with values 0's and 1's.
Now let's move to the hardware portion, basically it uses Arduino Uno as data and power synchronizer. It provides an environment so that sensors are run correctly. They will give alert messages as per situations.

4.2 COCO Dataset

person, bicycle, car, motorcycle, airplane, bus, train, truck ,boat,traffic light

fire hydrant,street sign,stop sign,parking meter,bench,bird,cat,dog,horse,sheep,cow,

elephant,bear,zebra,giraffe,hat,backpack,umbrella,shoe,eyeglasses,handbag,tie,

suitcase,frisbee,skis,snowboard,sports ball,kite,baseball bat,baseball glove

,skateboard,surfboard,tennis racket,bottle,plate,wine glass,cup,fork,knife

,spoon,bowl,banana,apple,sandwich,orange,broccoli,carrot,hot dog,pizza,donut

,cake,chair,couch,potted plant,bed,mirror,dining table,window,desk,toilet,door

,tv,laptop,mouse,remote,keyboard,cell phone,microwave,oven,toaster,sink,refrigerator

,blender,book,clock,vase,scissors,teddy bear,hair drier,toothbrush,hair brush,tree

,animal,clothes,pen,camera,board

4.3 Architecture

MobileNet:

We first describe the core layers that MobileNet is built on which are depthwise separable filters. We then describe the MobileNet network structure and conclude with descriptions of the two model shrinking hyperparameters, width multiplier and resolution multiplier.

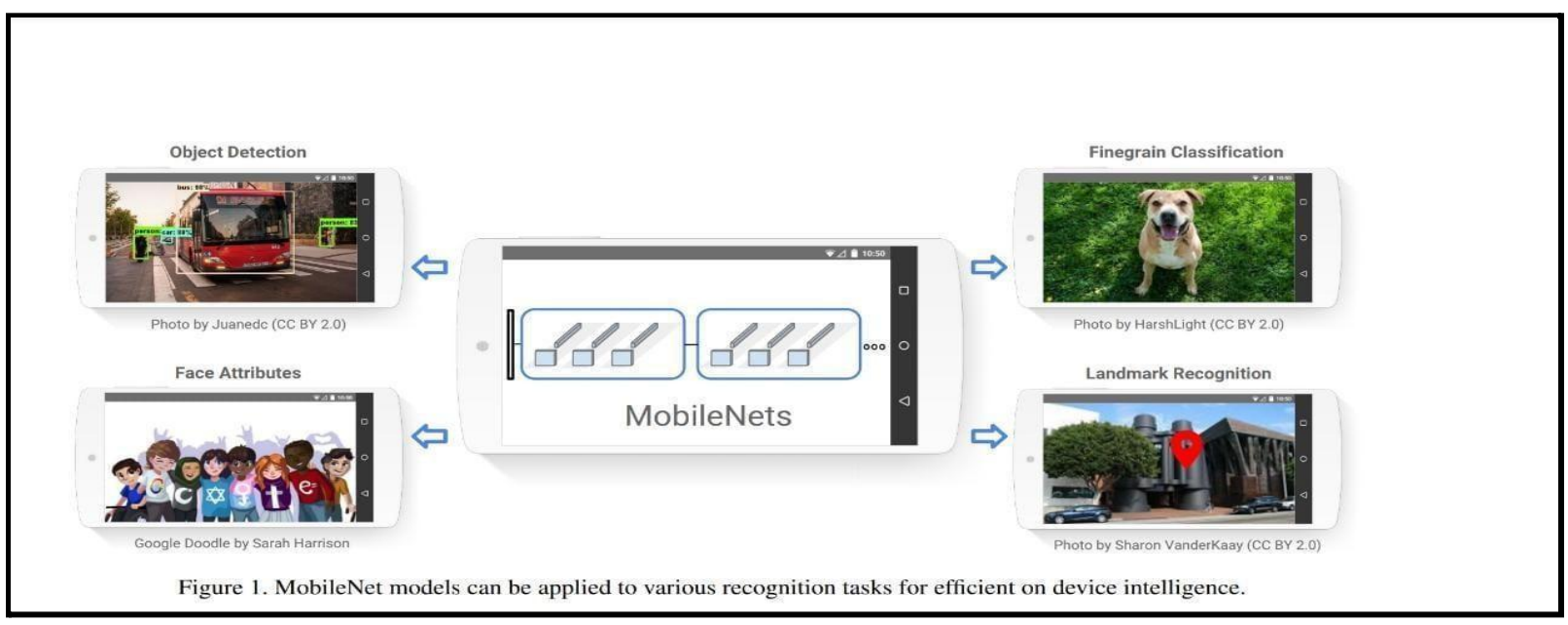


Figure 1. MobileNet models can be applied to various recognition tasks for efficient on device intelligence.

Fig 3: MobileNet Model

.1. Depthwise Separable Convolution

The MobileNet model is based on depth wise separable convolutions which is a form of factorized convolutional which factorizes a standard convolution into a depthwise convolution and a 11 convolution called a pointwise convolution. For MobileNets the depthwise convolution applies a single filter to each input channel. The pointwise convolution then applies a 11 convolution to combine the outputs the depthwise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step.

The depthwise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining.

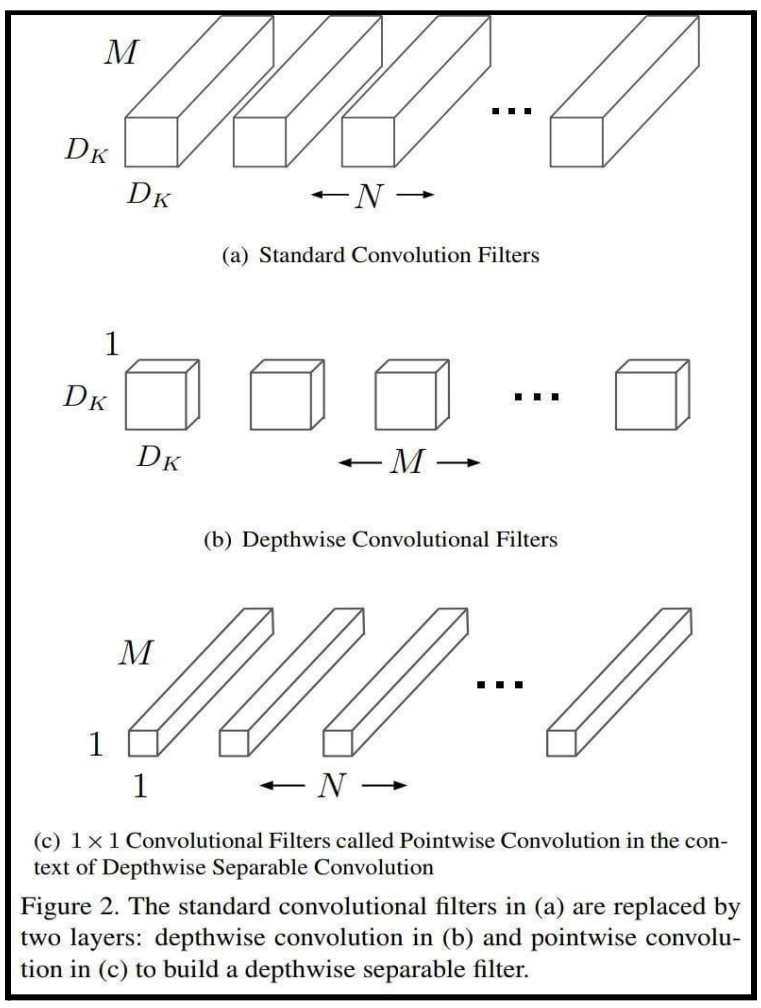


Figure 2. The standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter.

Fig 4: Standard Convolution Filters & Depthwise Convolutional Filter

2. Network Structure and Training

The MobileNet structure is built on depth wise separable convolutions as mentioned in the previous section except for the first layer which is a full convolution. By defining the network in such simple terms we are able to easily explore network topologies to find a good network. The MobileNet architecture is defined in Table 1. All layers are followed by a batchnorm [13] and ReLU nonlinearity with the exception of the final fully connected layer which has no nonlinearity and feeds into a softmax layer for classification. Figure 3 contrasts a layer with regular convolutions, batchnorm and ReLU nonlinearity to the factorized layer with depthwise convolution, 1 1 pointwise convolution as well as batchnorm and ReLU after each convolutional layer. Down sampling is handled with strided convolution in the depthwise convolutions as well as in the first layer. A final average pooling reduces the spatial resolution to 1 before the fully connected layer. Counting depthwise and pointwise convolutions as separate layers, MobileNet has 28 layers.

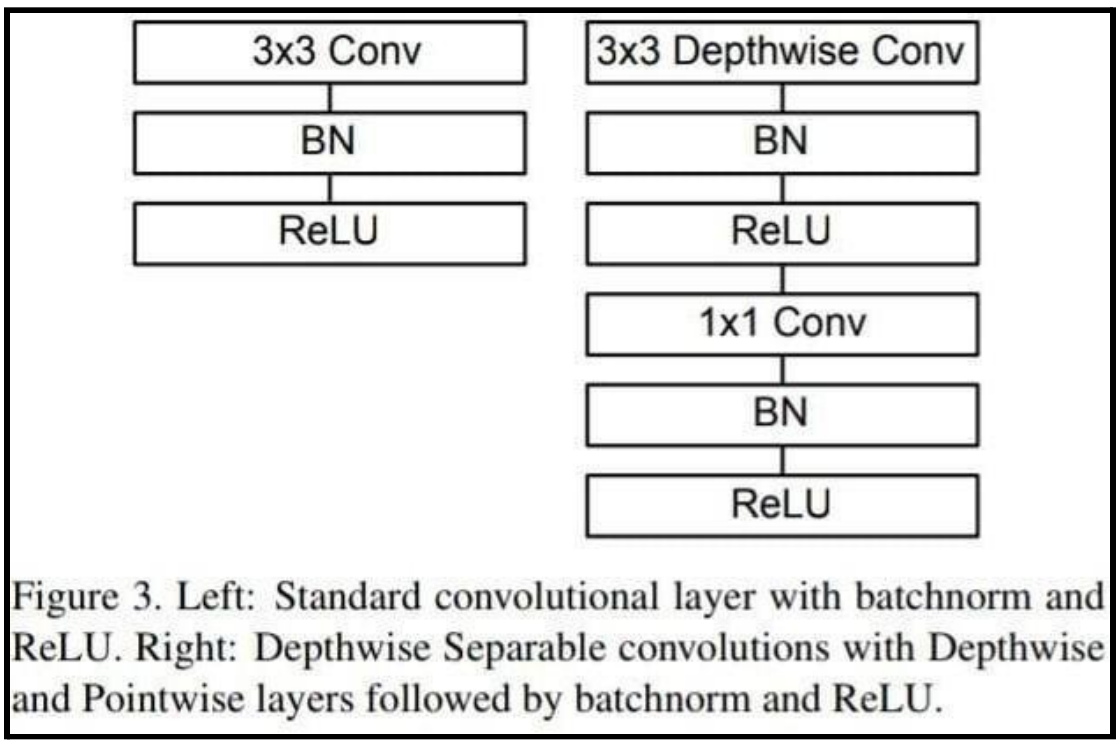


Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

Fig 5: Standard Convolution Filters & Depthwise Convolutional Filters

3. Width Multiplier: Thinner Models

Although the base MobileNet architecture is already small and low latency, many times a specific use case or application may require the model to be smaller and faster. In order to construct these smaller and less computationally expensive models we introduce a very simple parameter
called width multiplier. The role of the width multiplier is to thin a network uniformly at each layer. For a given layer.

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

Fig 6: MobileNet Body Architecture & Resource Per Layer Type

Chapter - 5

IMPLEMENTATION

Safety Models to Prevent Road Accidents

[1] Traffic Signal Detection System:

Traffic sign detection systems constitute a key component in trending real-world applications, such as autonomous driving, and driver safety and assistance. This paper analyses the state-of-the-art of several object-detection systems (Faster R-CNN, R-FCN, SSD, and YOLO V2) combined with various feature extractors (Resnet V1 50, Resnet V1 101, Inception V2, Inception Resnet V2, Mobilenet V1, and Darknet-19) previously developed by their corresponding authors. We aim to explore the properties of these object-detection models which are modified and specifically adapted to the traffic sign detection problem domain by means of transfer learning. In particular, various publicly available object-detection models that were pre-trained on the Microsoft COCO dataset are fine-tuned on the German Traffic Sign Detection Benchmark dataset. The evaluation and comparison of these models include key metrics, such as the mean average precision (mAP), memory allocation, running time, number of floating point operations, number of parameters of the model, and the effect of traffic sign image sizes. Our findings show that Faster R-CNN Inception Resnet V2 obtains the best mAP, while R-FCN Resnet 101 strikes the best trade-off between accuracy and execution time. YOLO V2 and SSD Mobilenet merit a special mention, in that the former achieves competitive accuracy results and is the second fastest detector, while the latter, is the fastest and the lightest model in terms of memory consumption, making it an optimal choice for deployment in mobile and embedded devices.

Working

1. Train the model using CNN and either a custom dataset or downloaded dataset(from kaggle).
2. Make a classification as per number of signs present in the dataset.
3. Run a customized loop and validate each frame of the input video.
4. Predict the output as per classification and take appropriate decisions.

Fig 7: output of Traffic Traffic Signal Detection System

CODE IMPLEMENTATION FOR ROAD SIGN DETECTION SYSTEM

```
import numpy as np
import cv2
from urllib.request import urlopen
#load the trained model to classify sign
from tensorflow.keras.models import load_model
model = load_model(r'my_model.h5')
########################################################

frameWidth= 640          # CAMERA RESOLUTION
frameHeight = 480
brightness = 180
threshold = 0.75          # PROBABLITY THRESHOLD
font = cv2.FONT_HERSHEY_SIMPLEX
########################################################

# SETUP THE VIDEO CAMERA
url="http://192.168.43.1:8080/shot.jpg"
def grayscale(img):
    img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    return img
def equalize(img):
    img =cv2.equalizeHist(img)
    return img
def preprocessing(img):
    img = grayscale(img)
    img = equalize(img)
    img = img/255
    return img
def getCalssName(classNo):
    if   classNo == 0: return 'Speed Limit 20 km/h'
    elif classNo == 1: return 'Speed Limit 30 km/h'
    elif classNo == 2: return 'Speed Limit 50 km/h'
    elif classNo == 3: return 'Speed Limit 60 km/h'
```

```
    elif classNo == 35: return 'Ahead only'
    elif classNo == 36: return 'Go straight or right'
    elif classNo == 37: return 'Go straight or left'
    elif classNo == 38: return 'Keep right'
    elif classNo == 39: return 'Keep left'
    elif classNo == 40: return 'Roundabout mandatory'
    elif classNo == 41: return 'End of no passing'
    elif classNo == 42: return 'End of no passing by vechiles over 3.5 metric tons'
while True:
    # READ IMAGE
    imgResp=urlopen(url)
    imgNp=np.array(bytearray(imgResp.read()),dtype=np.uint8)
    imgOrignal=cv2.imdecode(imgNp,-1)

    # PROCESS IMAGE
    img = np.asarray(imgOrignal)
    img = cv2.resize(img, (32, 32))
    img = preprocessing(img)
    cv2.imshow("Processed Image", img)
    img = img.reshape(1, 32, 32, 1)
    cv2.putText(imgOrignal, "CLASS: " , (20, 35), font, 0.75, (0, 0, 255), 2, cv2.LINE_AA)
    cv2.putText(imgOrignal, "PROBABILITY: ", (20, 75), font, 0.75, (0, 0, 255), 2, cv2.LINE_AA)
    # PREDICT IMAGE
    predictions = model.predict(img)
    classIndex = model.predict_classes(img)
    probabilityValue =np.amax(predictions)
    if probabilityValue > threshold:
    #print(getCalssName(classIndex))
        cv2.putText(imgOrignal,str(classIndex)+" "+str(getCalssName(classIndex)), (120, 35), font, 0.75, (0, 0, 255), 2, cv2.LINE_AA)
        cv2.putText(imgOrignal, str(round(probabilityValue*100,2) )+"%", (180, 75), font, 0.75, (0, 0, 255), 2, cv2.LINE_AA)
    cv2.imshow("Result", imgOrignal)

    if cv2.waitKey(1)& 0xFF == ord('q'):
        break
cv2.destroyAllWindows()
```

[2] Drowsiness Detection System:

A computer vision system that can automatically detect driver drowsiness in a real-time video stream and then play an alarm if the driver appears to be drowsy.

Driver drowsiness and attention warning and advanced driver distraction warning systems shall be designed in such a way that those systems do not continuously record nor retain any data other than what is necessary in relation to the purposes for which they were collected or otherwise processed within the closed-loop system. Furthermore, those data shall not be accessible or made available to third parties at any time and shall be immediately deleted after processing. Those systems shall also be designed to avoid overlap and shall not prompt the driver separately and concurrently or in a confusing manner where one action triggers both systems.(-regulation (EU) 2019/2144) This system is also known as EyeSight Driver Assist,Tiredness Detection Warning, Anti Sleep Pilot,Fatigue detection system....

Applications:

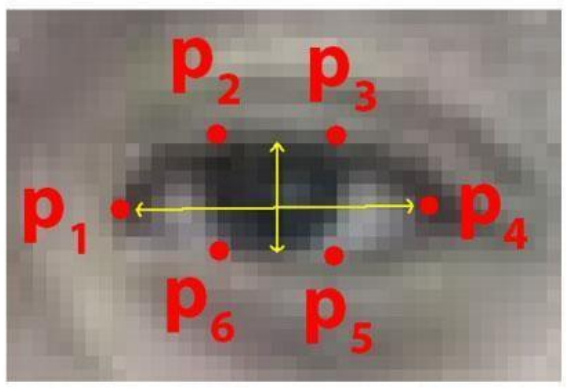
This can be used by riders who tend to drive for a longer period of time that may lead to accidents

Dependencies:

1. import cv2
2. import imutils
3. import dlib
4. import scipy

Algorithm

Each eye is represented by 6 (x, y)-coordinates, starting at the left-corner of the eye (as if you were looking at the person), and then working clockwise around the eye:.



Condition

It checks 15 consecutive frames and if the Eye Aspect ratio is less than 0.25, Alert is generated.

Relationship

$$\mathrm{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

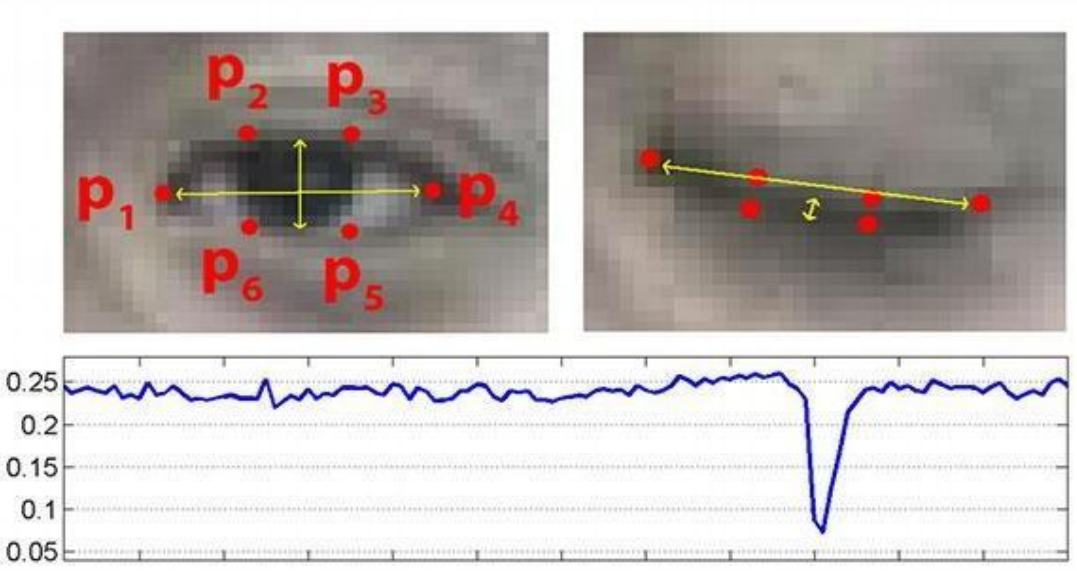


Fig 8: Eye Detection Algorithm

CODE IMPLEMENTATION OF DROWSINESS DETECTION SYSTEM

```
from scipy.spatial import distance
from urllib.request import urlopen
from imutils import face_utils
import imutils
import dlib
import cv2
import numpy as np
# import winsound
freq=0
dur=0
url="http://192.168.43.1:8080/shot.jpg"
def eye_aspect_ratio(eye):
        A = distance.euclidean(eye[1], eye[5])
        B = distance.euclidean(eye[2], eye[4])
        C = distance.euclidean(eye[0], eye[3])
        ear = (A + B) / (2.0 * C)
        return ear

thresh = 0.25
frame_check = 15
detect = dlib.get_frontal_face_detector()
predict = dlib.shape_predictor(".\shape_predictor_68_face_landmarks.dat")# Dat file is the crux of the code
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["left_eye"]
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_68_IDXS["right_eye"]
flag=0
while True:
        imgResp=urlopen(url)
        imgNp=np.array(bytearray(imgResp.read()),dtype=np.uint8)
        frame=cv2.imdecode(imgNp,-1)
        frame = imutils.resize(frame, width=450)
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        subjects = detect(gray, 0)
        for subject in subjects:
```

```
    frame = imutils.resize(frame, width=450)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    subjects = detect(gray, 0)
    for subject in subjects:
            shape = predict(gray, subject)
            shape = face_utils.shape_to_np(shape)#converting to NumPy Array
            leftEye = shape[lStart:lEnd]
            rightEye = shape[rStart:rEnd]
            leftEAR = eye_aspect_ratio(leftEye)
            rightEAR = eye_aspect_ratio(rightEye)
            ear = (leftEAR + rightEAR) / 2.0
            leftEyeHull = cv2.convexHull(leftEye)
            rightEyeHull = cv2.convexHull(rightEye)
            cv2.drawContours(frame, [leftEyeHull], -1, (0, 255, 0), 1)
            cv2.drawContours(frame, [rightEyeHull], -1, (0, 255, 0), 1)
            if ear < thresh:
                    flag += 1
                    print (flag)
                    if flag >= frame_check:
                            cv2.putText(frame, "****************ALERT!****************", (10, 30),
                                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
                            for i in range(0,6):
                                    freq+=50
                                    dur+=10
                                    # winsound.Beep(freq,dur)
            elif ear > thresh:
                    flag = 0
                    freq=0
                    dur=0

    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF
    if key == ord("q"):
            break
2.destroyAllWindows()
```

[3] Object Detection System:

This system is used to detect the distance between two vehicles so as to maintain the minimum distance limit. Basically to avoid tailgating. Object detection is a branch of computer vision which deals with the localization and the identification of an object. Object localization and identification are two different tasks that are put together to achieve this singular goal of object detection.Specifying the location of an object in an image or a video stream is called object localization and assigning the object to a specific label, class, or description is called object identification.

Some architectures that have performed tremendously well on the **COCO**

dataset. The model architectures include:

1. CenterNet

2. EfficientDet

3. MobileNet

4. ResNet

5. R-CNN

6. ExtremeNet

Here we are going to use MobileNet architecture. **MobileNet** is an object detector (2017) used as an efficient CNN architecture designed for mobile and embedded vision applications. This architecture uses proven depth-wise separable convolutions to build lightweight deep neural networks.

Optimizing the model

Freezing the model means producing a singular file containing information about the graph and checkpoint variables, but saving these hyperparameters as constants within the graph structure. This eliminates additional information saved in the checkpoint files such as the gradients at each point, which are included so that the model can be reloaded and training continued from where you left off. As this is not needed when serving a model purely for inference, they are discarded in the freezer.

Working

This module is a part of the Accident Control Management System. This will be helpful in detection and recognition of objects. With this, if anything comes in front of the vehicle then it will detect that and display the visuals. With the help of those visuals, the driver will be able to make a proper decision at the right time.This module will work on IP WebCam by default. This module will also work with a system camera. To access the system camera use below code block:-

```
import cv2
import numpy as np

thres = 0.45 # Threshold to detect object
nms_threshold = 0.2
cap = cv2.VideoCapture("run1.mp4")
# cap.set(3,1280)
# cap.set(4,720)
# cap.set(10,150)

classNames= []
classFile = 'coco.names'
with open(classFile,'rt') as f:
    classNames = f.read().rstrip('\n').split('\n')

#print(classNames)
configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
```

```
weightsPath = 'frozen_inference_graph.pb'

net =
cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5,
127.5)) net.setInputSwapRB(True)

while True:
   success,img =
   cap.read() if success:
       classIds, confs, bbox =
       net.detect(img,confThreshold=thres) bbox = list(bbox)
       confs =
       list(np.array(confs).reshape(1,-1)[0]) confs
       = list(map(float,confs))
       #print(type(confs[0]))
       #print(confs)

       indices =
       cv2.dnn.NMSBoxes(bbox,confs,thres,nms_threshold)
       #print(indices)

       for i in
           indices: i
           =
           indices[0]
           box =
           bbox[i]
           x,y,w,h = box[0],box[1],box[2],box[3]
           cv2.rectangle(img, (x,y),(x+w,h+y), color=(0, 255, 0), thickness=2)
           #
           cv2.putText(img,classNames[classIds[i][0]-1].upper(),(box[0]+10,box[1]+30)
           , #    cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
```

[4] Pot-Hole Detection System:

A pothole is a depression in a road surface, usually asphalt pavement, where traffic has removed broken pieces of the pavement. It is usually the result of water in the underlying soil structure and traffic passing over the affected area. Water first weakens the underlying soil; traffic then fatigues and breaks the poorly supported asphalt surface in the affected area. Continued traffic action ejects both asphalt and the underlying soil material to create a hole in the pavement.

Formation

According to the US Army Corps of Engineers, pothole formation requires two factors to be present at the same time: water and traffic. Water weakens the soil beneath the pavement while traffic applies the loads that stress the pavement past the breaking point. Potholes form progressively from fatigue of the road surface which can lead to a precursor failure pattern known as crocodile (or alligator) cracking. Eventually, chunks of pavement between the fatigue cracks gradually work loose, and may then be plucked or forced out of the surface by continued wheel loads to create a pothole.In areas subject to freezing and thawing, frost heaving can damage a pavement and create openings for water to enter. In the spring, thaw of pavements accelerates this process when the thawing of upper portions of the soil structure in a pavement cannot drain past still-frozen lower layers, thus saturating the supporting soil and weakening it.Potholes can grow to several feet in width, though they usually only develop to depths of a few inches. If they become large enough, damage to tires, wheels, and vehicle suspensions is liable to occur. Serious road accidents can occur as a direct result, especially on those roads where vehicle speeds are greater.Potholes may result from four main causes:

Insufficient pavement thickness to support traffic during freeze/thaw periods without localized failures Insufficient drainage Failures at utility trenches and castings (manhole and drain casings) Pavement defects and cracks left unmaintained and unsealed so as to admit moisture and compromise the structural integrity of the pavement

Working

We can classify the dataset which either download from kaggle or built your own. After that save the model for future use. Open camera so that we can detect real time pot hole. This method is also applicable on images. Taking the input from camera and processed it using CNN and predict with model which is already saved. This process will return the class name and probability. Used class name and probability as output. The output will classify into three categories that are:

1. Pot-Hole
2. Plain-Road
3. None

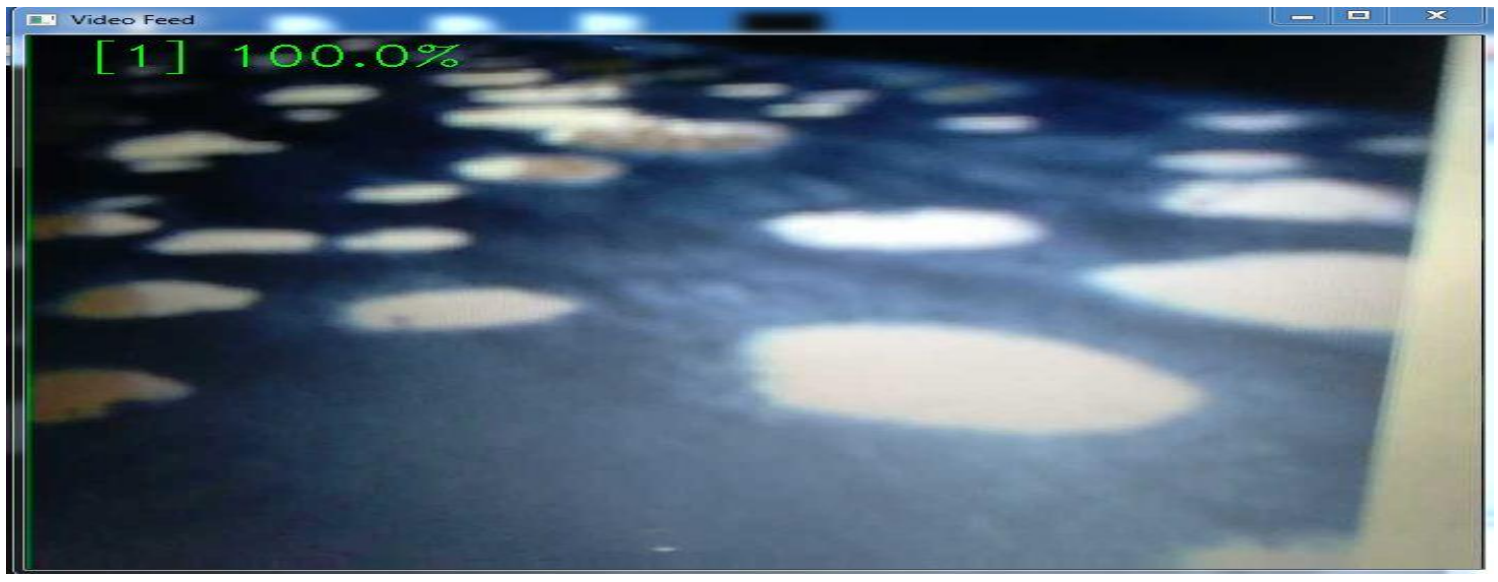


Fig 9: Pothole detection

```
1/1 [==============================] - 0s 37ms/step - loss: 8.9407e-08 - ac
Training Accuracy:  100.0 %

1/1 [==============================] - 0s 32ms/step - loss: 14.5874 - accur
Testing Accuracy:  50.0 %
Saving model weights and configuration file
```

Fig 10: Accuracy of model

```
Output exceeds the size limit. Open the full output da
1
0
1
0
train shape X (4, 300, 300, 1)
train shape y (4, 2)
```

Fig 11: Coordinates of pothole detected

Fig 12: Testing for potholes

CODE IMPLEMENTATION FOR THE POT-HOLE DETECTION SYSTEM

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import tensorflow as tf
from tensorflow.keras.layers import Flatten
from keras.models import Sequential, Model, load_model
from keras.callbacks import EarlyStopping, Callback
from keras.layers import Dense, Dropout, Activation, Flatten, Lambda, ELU,GlobalAveragePooling2D
from keras import regularizers
from keras.layers.convolutional import Convolution2D, Cropping2D, Conv2D
from keras.layers.pooling import MaxPooling2D
from tensorflow.keras.optimizers import Adam
from sklearn.utils import shuffle
from keras.utils import np_utils
import time, cv2, glob
global inputShape,size
def kerasModel4():
        model = Sequential()
        model.add(Conv2D(16, (8, 8), strides=(4, 4), padding='valid', input_shape=(size,size,1)))
        model.add(Activation('relu'))
        model.add(Conv2D(32, (5, 5), padding="same"))
        model.add(Activation('relu'))
        model.add(GlobalAveragePooling2D())
        model.add(Dense(512))
        model.add(Dropout(.1))
        model.add(Activation('relu'))
        model.add(Dense(2))
        model.add(Activation('softmax'))
        return model
size=300
 ## load Training data : pothole
potholeTrainImages = glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole1.jpeg")
```

```
        return model
size=300
 ## load Training data : pothole
potholeTrainImages = glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole1.jpeg")
potholeTrainImages.extend(glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole2.jpeg"))
potholeTrainImages.extend(glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole1.jpeg"))
train1 = [cv2.imread(img,0) for img in potholeTrainImages]
for i in range(0,len(train1)):
    train1[i] = cv2.resize(train1[i],(size,size))
temp1 = np.asarray(train1)
#  ## load Training data : non-pothole
nonPotholeTrainImages = glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole3.jpeg")
train2 = [cv2.imread(img,0) for img in nonPotholeTrainImages]
for i in range(0,len(train2)):
    train2[i] = cv2.resize(train2[i],(size,size))
temp2 = np.asarray(train2)
## load Testing data : non-pothole
nonPotholeTestImages = glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole2.jpeg")
test2 = [cv2.imread(img,0) for img in nonPotholeTestImages]
for i in range(0,len(test2)):
    test2[i] = cv2.resize(test2[i],(size,size))
temp4 = np.asarray(test2)
potholeTestImages = glob.glob("/home/lucky/Documents/Road-Accident-Management-System/Pot-Hole-Detction/pot-hole1.jpeg")
test1 = [cv2.imread(img,0) for img in potholeTestImages]
for i in range(0,len(test1)):
    test1[i] = cv2.resize(test1[i],(size,size))
temp3 = np.asarray(test1)
X_train = []
X_train.extend(temp1)
X_train.extend(temp2)
X_train = np.asarray(X_train)
X_test = []
X_test.extend(temp3)
X_test.extend(temp4)
```

```
X_test.extend(temp4)
X_test = np.asarray(X_test)
# print(train1)
y_train1 = np.ones([temp1.shape[0]],dtype = int)
y_train2 = np.zeros([temp2.shape[0]],dtype = int)
y_test1 = np.ones([temp3.shape[0]],dtype = int)
y_test2 = np.zeros([temp4.shape[0]],dtype = int)
print(y_train1[0])
print(y_train2[0])
print(y_test1[0])
print(y_test2[0])
y_train = []
y_train.extend(y_train1)
y_train.extend(y_train2)
y_train = np.asarray(y_train)
y_test = []
y_test.extend(y_test1)
y_test.extend(y_test2)
y_test = np.asarray(y_test)
X_train,y_train = shuffle(X_train,y_train)
X_test,y_test = shuffle(X_test,y_test)
X_train = X_train.reshape(X_train.shape[0], size, size, 1)
X_test = X_test.reshape(X_test.shape[0], size, size, 1)
y_train = np_utils.to_categorical(y_train)
y_test = np_utils.to_categorical(y_test)
print("train shape X", X_train.shape)
print("train shape y", y_train.shape)

inputShape = (size, size, 1)
model = kerasModel4()

X_train = X_train/255
X_test = X_test/255

model.compile('adam', 'categorical_crossentropy', ['accuracy'])
history = model.fit(X_train, y_train, epochs=1000,validation_split=0.1)
```

```
X_test = X_test.reshape(X_test.shape[0], size, size, 1)
y_train = np_utils.to_categorical(y_train)
y_test = np_utils.to_categorical(y_test)
print("train shape X", X_train.shape)
print("train shape y", y_train.shape)

inputShape = (size, size, 1)
model = kerasModel4()

X_train = X_train/255
X_test = X_test/255

model.compile('adam', 'categorical_crossentropy', ['accuracy'])
history = model.fit(X_train, y_train, epochs=1000,validation_split=0.1)

print("")

metricsTrain = model.evaluate(X_train, y_train)
print("Training Accuracy: ",metricsTrain[1]*100,"%")

print("")

metricsTest = model.evaluate(X_test,y_test)
print("Testing Accuracy: ",metricsTest[1]*100,"%")

print("Saving model weights and configuration file")
model.save('latest_full_model.h5')
print("Saved model to disk")
```

[5] Location Sharing System:

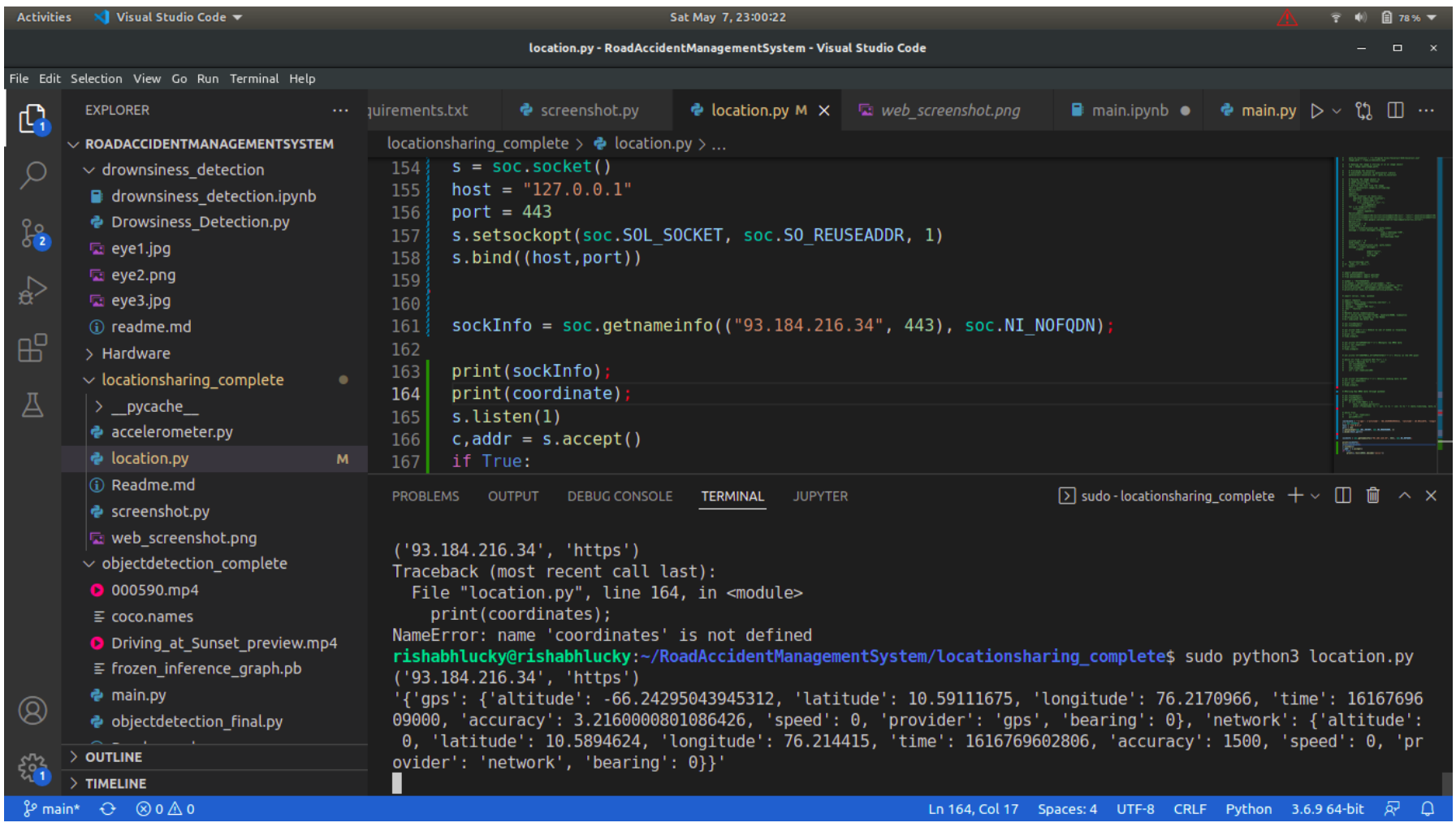
A vehicle moving on high speed will have greater impact during the crash and hence will cause more injuries. Some deaths also happen due to lack of immediate first-aid. Another problem is that the lack of information about the vehicle position. So,in our project we have used GSM/GPRS modem to locate the exact position of the vehicle. This project presents vehicle accident detection and alert system with SMS to the user defined mobile numbers.

Working :

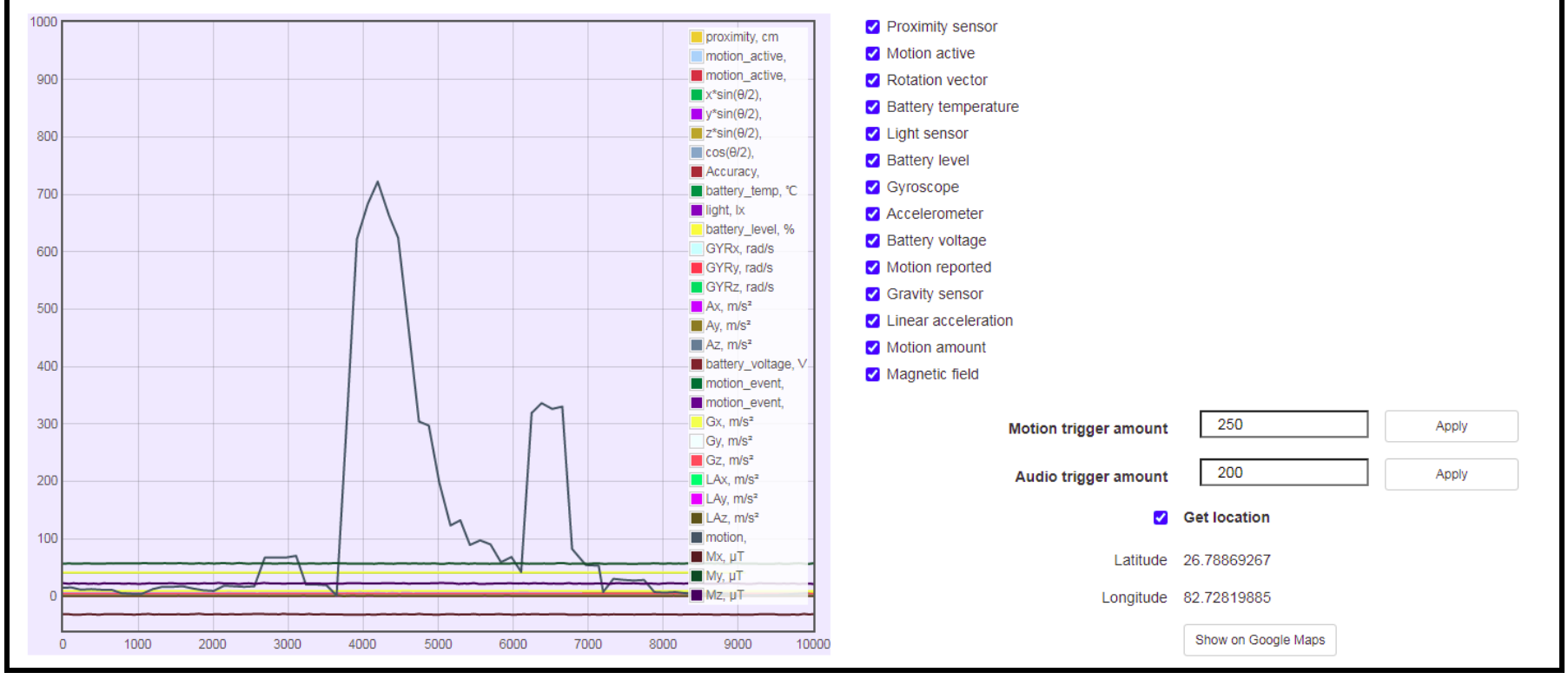
Socket programming is a way of connecting two nodes on a network to communicate with each other.
Now when the car suddenly meet with an accident and when it reaches above the range of sensor then the vibrating sensor will switch on and the location fetched containing longitude and latitude of the accident zone will be sent on the phone using Twilio.

Hence with this model implementation we can detect the position of the vehicle where the accident has occurred so that we can provide the first aid as early as possible. This project presents vehicle accident detection and alert system with SMS to the user defined mobile numbers (GPS is used for tracking the position of the vehicle, GSM is used for sending the message).

CODE IMPLEMENTATION FOR THE LOCATION SHARING SYSTEM



Sensor graph connected with our model :



Program to run same code on Android :

The program on the android device will collect all available information regarding the cell location. Location information is collected from the GPS as well as from the network. Then, the collected data is sent to another device using wireless communication.
The python program is run on the android device using Qpython.
Qpython download link.

The Python program for the Android device:

```
import androidhelper as android
import time
import socket
port=12345
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("192.168.43.70",port)) #Ip address of the connected network
droid = android.Android()
while True:
    droid.startLocating(minDistance=1000,minUpdateDistance=1)
    droid.eventWaitFor('location', int(9000))
    location = droid.readLocation().result
    data = bytes(str(location),'ascii')
    s.send(data)
    print(location)
    time.sleep(3)
droid.stopLocating()
```

The socket module is used to establish a connection between an android device and a Laptop (Wireless device).
To know more about socket module, see the documentaion here.
The andriodhelper moudle is used to access different status of the android device.
To know more about androidhelper module, see the details here.

```
location = droid.readLocation().result
```

This method `readlocation()`, will collect the location information from all available sources and returns a dictionary . A sample data is shown below.

```
{'gps': {'altitude': -66.24295043945312, 'latitude': 10.59111675, 'longitude': 76.2170966, 'time': 1616769609000, 'accurac
```

This dictionary is converted to string and then converted to bytes. After that, it is sent to the Laptop. In this program, the location information is sent once in every three seconds. The rate can be varied by changing the argument inside `time.sleep()`.

CHAPTER -6

RESULT AND ANALYSIS

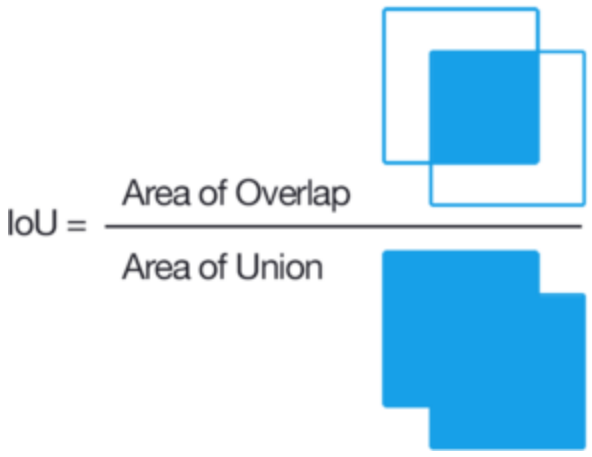
After using our software for more than 10 dozen datasets, we have compared the results and have realized that it detects the moving object and it can help us track the movement of the object in front of the driving car which can detect and warn the driver to stop tailgating and maintain proper distance. However it might miss a few objects which are not moving and which are too small to be noticed, but our model will keep updating and will learn with time.

We have analyzed that our model works 8/10 times correctly, that means it detected the object correctly in every 8 for every 10 datasets which were differently presented but having the same kind of object which gives us the accuracy of **80-85%**. This was the average calculated after running our models a few times and now we can conclude that this model is ready for use in public and can help to detect the objects in front of the vehicle so as to prevent tailgating. We have attached a few screenshots and recorded a few object movements in the next chapter so as to show the working of our model.

A object detection model produces the output in three components:

- The **bounding boxes** — x1, y1, width, height if using the COCO file format
- The **class** of the bounding box
- The **probability score** for that prediction— how certain the model is that the class is actually the predicted class

To evaluate if an object was located we use the Intersection over Union (IoU) as a similarity measure. It's given by the area of the overlap divided by the size of the union of the two bounding boxes.



Why the Average Precision isn't Enough

The mAP is good for a competition setting or to get a quick assessment of how a model is performing, but if you care about understanding what the model is doing you'll need different metrics.

The source of errors in object detection tasks can be various:

- the model could find the correct location of the bounding box but output it with an incorrect class
- the model could find an object with a correct class but be totally wrong about the location of the bounding box
- the model could miss an object entirely and so on

CHAPTER -7

IMAGE INPUT AND RESULTS

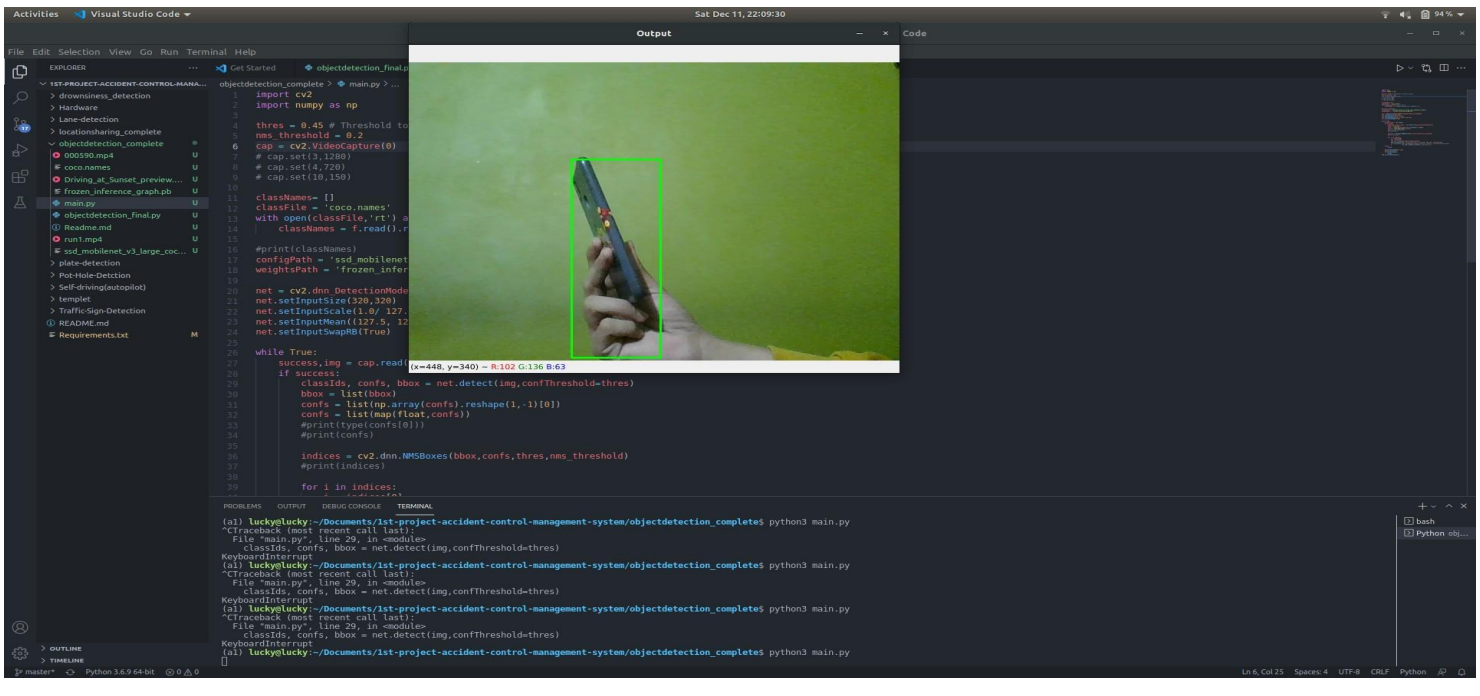
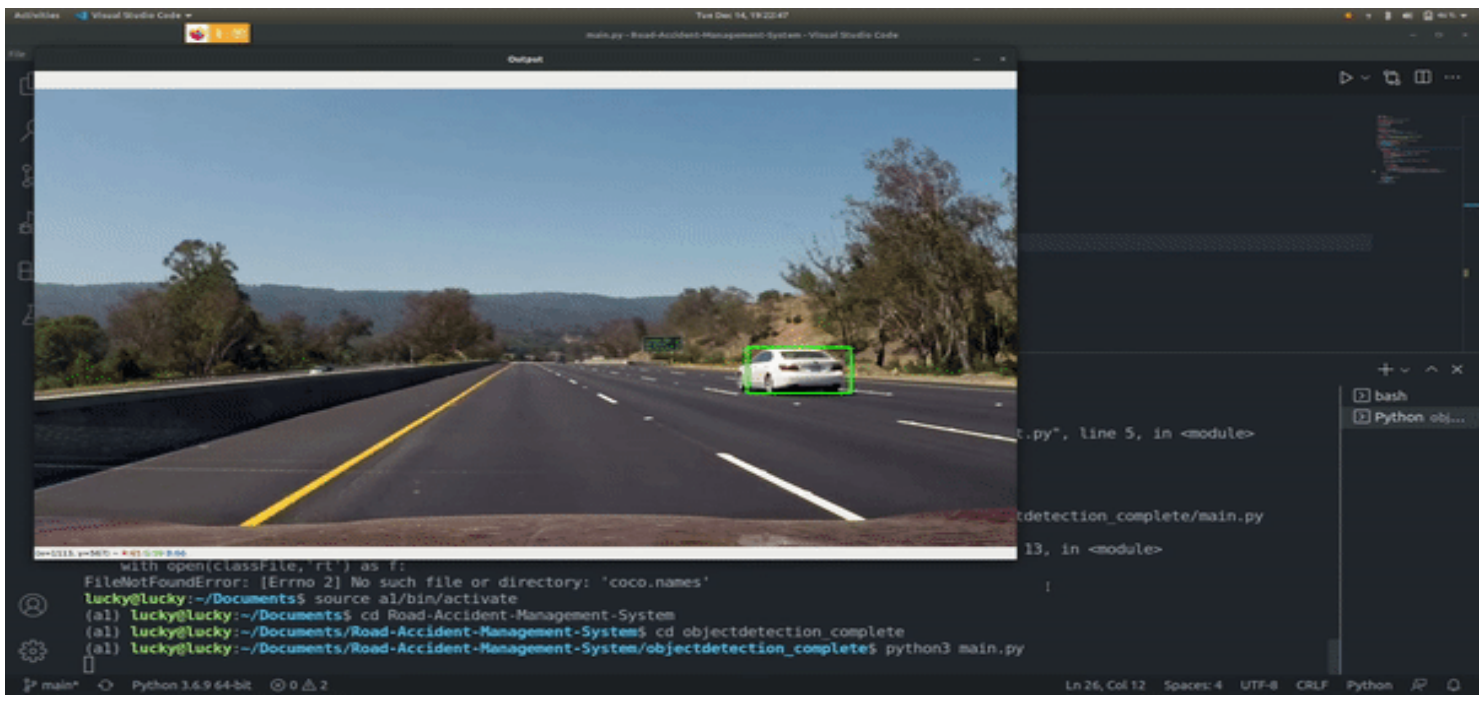
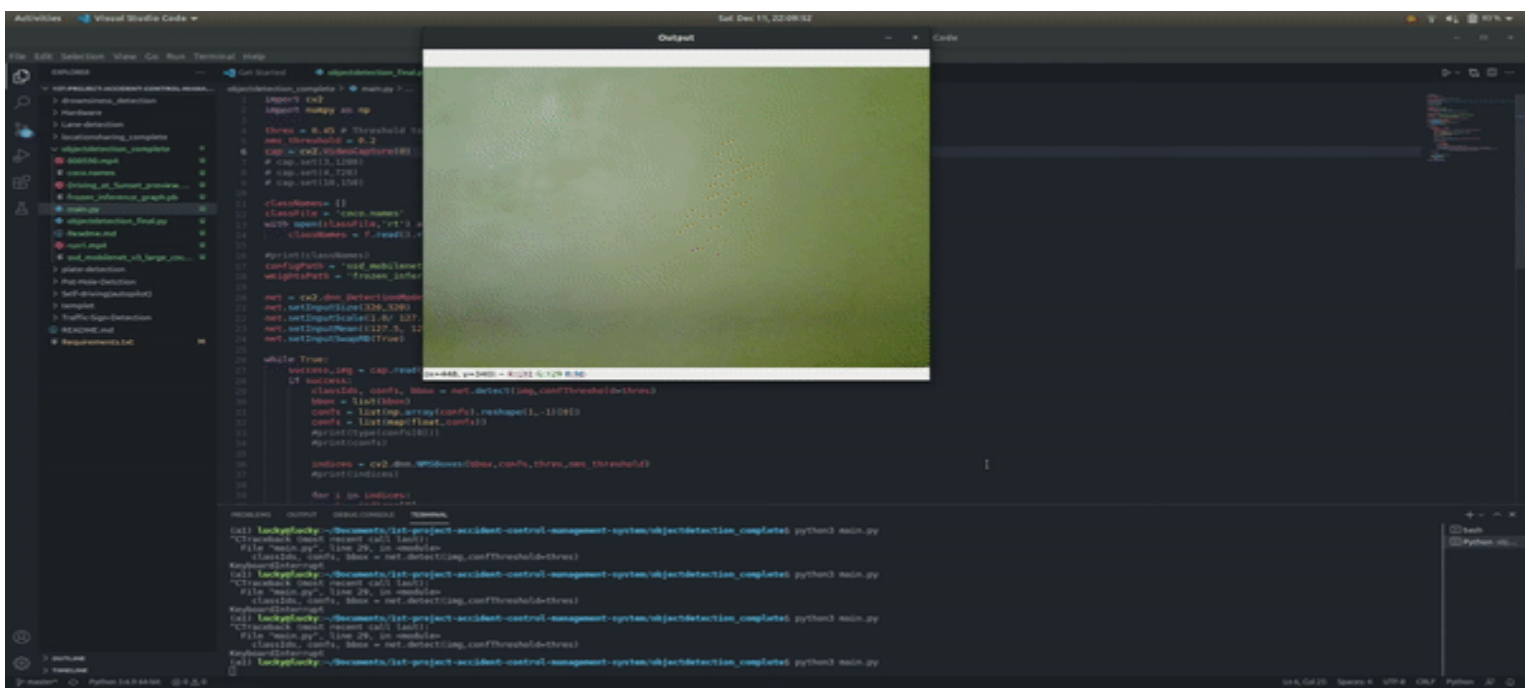


Fig 13:Output of pre-clicked Image



Gif 1: (above is a GIF for object detection working)



Gif2: (above is a GIF for movement detection)

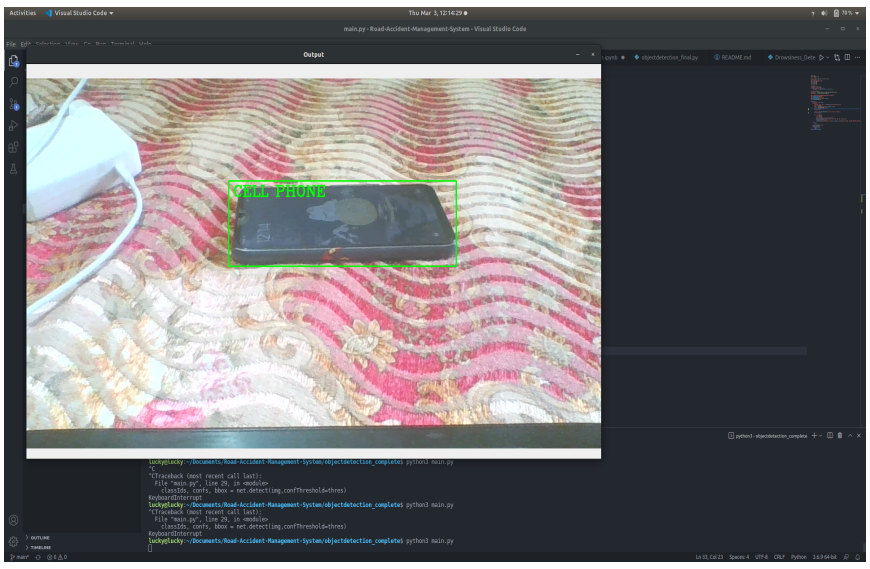


Fig 14: Cell Phone (OBJECT)

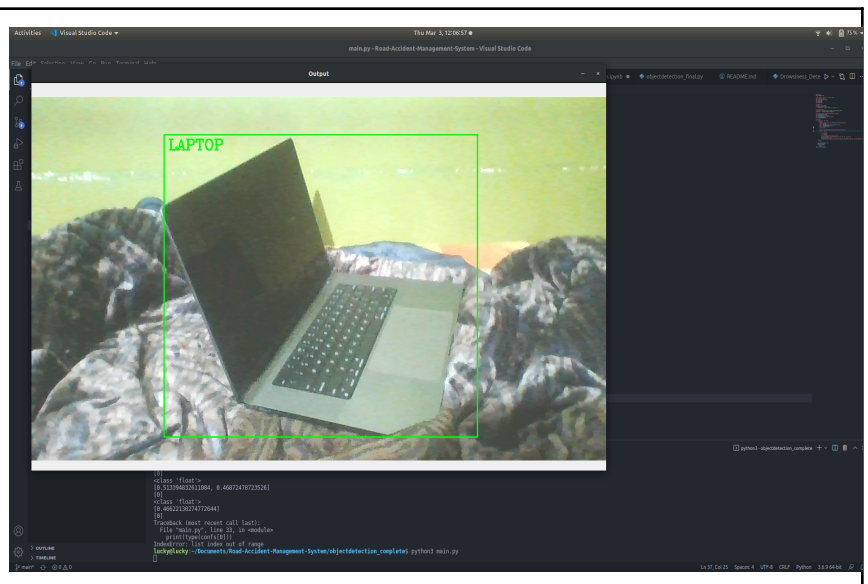


Fig 15: Laptop (OBJECT)

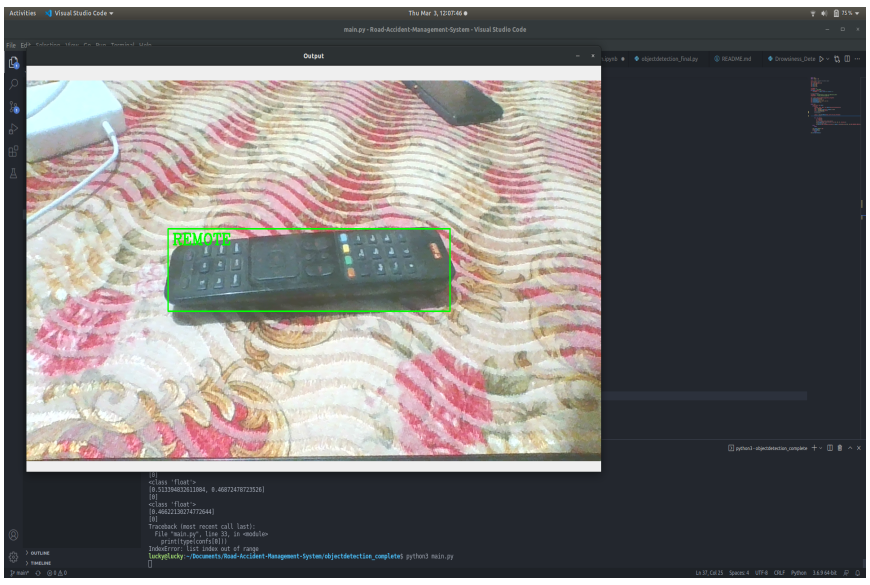


Fig 16: Remote (OBJECT)

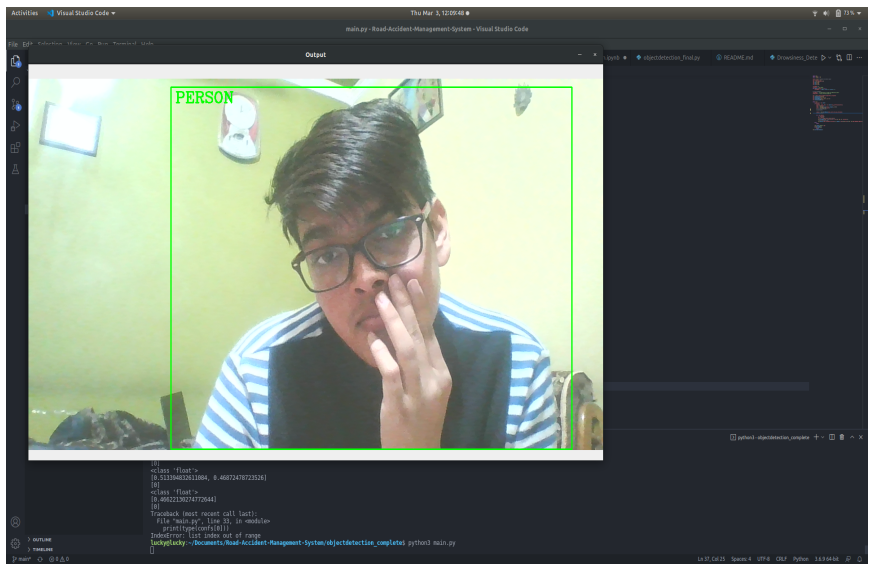


Fig 17: Person (OBJECT)

Chapter - 8

HARDWARE AND SOFTWARE REQUIREMENTS

As the project is developed in python, we have used Anaconda for Python 3.6.5 and Spyder.

• Anaconda

It is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. The Anaconda distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS.

• Spyder

Spyder (formerly Pydee) is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source software. It is released under the MIT license. Spyder is extensible with plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope.

Features include:

- editor with syntax highlighting and introspection for code completion.
- support for multiple Python consoles (including IPython).

Python libraries required :

numpy==1.19.5
opencv-contrib-python==4.5.2.54
tensorflow>=2.5.1
selenium==3.141.0
pandas==1.1.5
cmake==3.20.4
urllib3==1.26.6
dlib==19.7.0
keras==2.4.3
pytesseract==0.3.
8 sklearn==0.0
scipy==1.5.4
tqdm==4.61.1
matplotlib==3.3.4
imutils==0.5.4
requests==2.25.1
twilio==6.60.6
pillow>=8.3.2

In order to test our solution algorithms, we have also used dummy datasets from Kaggle and also OpenCV for image processing.

• Kaggle

Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges.

• OpenCV

It is a library of Python bindings designed to solve computer vision problems. OpenCV-Python makes use of **Numpy**, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All

the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

Hardware Interfaces

1. **Processor** : Intel CORE i5 processor with minimum 2.9 GHz speed.
2. **RAM** : Minimum 4 GB.
3. **Hard Disk** : Minimum 500 GB

Software Interfaces

1. Microsoft Word 2016
2. **Database Storage** : Microsoft Excel
3. **Operating System** : Linux/Windows10

CHAPTER - 9

CONCLUSION & FUTURE SCOPE

CONCLUSION

The main aim of this system is to construct a smart vehicle system with minimizing the limitations o Of existing models and also enhancing the security of vehicles and human beings by reducing Accidental injuries. Smart vehicle system will entail a speed and other parameters of vehicle sensing mechanism which automatically messages to personal contacts with the details of vehicle position when an accident occurs using the GPS system. By using image processing technology we can produce an accident free environment. This model will be useful for less memory space and gives more accurate results. Most models of this project use CNN for processing the images.

FUTURE SCOPE

In the future work, the model can be extended to:

a) **Plate detection** : Plate number detection is a combination of two parts that are plate-number-recognition and plate-number-detection. Apart from that it also uses text extraction for saving the plate-number in a csv file.

For extracting the text **OCR** (**Optical Character Recognition**) is used and Model CNN is used.

b) **Autopilot :**This system will generate an angle for steering control and the generation of angles basically depends on the movement of the camera.For more accurate results calibration of the camera is required. The camera is fixed into a particular angle and the learning rate of the model will increase and also provides an effective output.

c) **Lane Detection :** Extend the idea of deep learning in detecting the lanes by developing a multi-task deep CNN. Some of the filters which can be used are bilateral filter, gaussian filter, trilateral filter. One can change the current Hough Transformation so that it can sum up curved and straight roads respectively.

REFERENCES

Journals

[1] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Marco Andreetto, Tobias Weyand, Hartwig Adam, Efficient Convolutional Neural Networks for Mobile Vision Applications (MobileNet-architecture)

[2] **https://www.telesens.co/2018/03/11/object-detection-and-classification-using-r-cnns/** (RCNN, Fast RCNN, Faster RCNN Comparison)

[3] Alvaro Arcos-Garcıa, Juan A. Alvarez-Garcıa, Luis M. Soria-Morillo, *Evaluation of Deep Neural Networks for traffic sign detection systems*, Neurocomputing (2018), doi: https://doi.org/10.1016/j.neucom.2018.08.009.

[4] Jiali Cui, Fuqiang Chen, Duo Shi, Liqiang Liu, 2019, *Eye Detection with Faster R-CNN*, In Proceedings of the International Conference on Advances in Computer Technology, Information Science and Communications (CTISC 2019), DOI: 10.5220/0008096201110116, ISBN: 978-989-758-357-5.

[5] *"*The most accurate method is based on human physiological activity*"*, Aditya Ranjan, Karan Vyas, Sujay Ghadge, Siddharth Patel, Suvarna Sanjay Pawar, *Driver Drowsiness Detection System Using Computer Vision*, *IRJET* Volume: 07 Issue: 01 | Jan 2020.

[6] Naethan Jacob, Dr. Vishalakshi Prabhu H, 2020, Comparative Review of YOLO & MobileNet Versions: A Case Study, *(IRJET)*, ISSN: 2395-0056, Volume: 07 Issue: 04 | Apr 2020.

[7] Pothole Detection System Using Region-Based Convolutional Neural Network," 2021, *IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET)*, pp. 6-11, doi: 10.1109/CCET52649.

[8] Nathan Jennings, Socket Programming in Python (Guide), Feb 21, 2022.

[9] Tejal Palwankar, Kushal Kothari, 2022, Real Time Object Detection using SSD and MobileNet, *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, ISSN: 2278-0181, Volume 10 Issue III Mar 2022.

PROJECT DETAILS

Student1 Details

Name	**KUSHAGRA UPADHYAY**		
Register Number	2018UGCS020R	Section / Roll No	CSE/20
Email Address	kushagra.btech.cs18@iiitranchi.ac.in	Phone No (M)	9818784145

Project Details

Project Title	**ROAD ACCIDENT MANAGEMENT SYSTEM**		
Project Duration	5 months (approx.)	Date of reporting	8 MAY 2022

Organization Details

Organization Name	**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, (IIIT) RANCHI**		
Full postal address with pin code	IIIT Ranchi, Science & Technology Campus, Khojatoli, Namkum, Ranchi, Jharkhand, 834010		
Website address	http://iiitranchi.ac.in/		

Supervisor Details

Supervisor Name	**Dr. Nidhi Kushwaha**		
Designation	Faculty (CSE Department)		
Full contact address with pin code	Flat No. 402, Adhyeta Department, IIIT Ranchi, Science and Technology Campus, Sirkha Toli, Namkum, Ranchi, 834010		
Email address	nidhi@iiitranchi.ac.in	Phone No (M)	7703090242

Student2 Details

Name	**RISHABH KUMAR**		
Register Number	2018UGCS042R	Section / Roll No	CSE/42
Email Address	rishabhk.btech.cs18@iiitranchi.ac.in	Phone No (M)	7763829752

Project Details

Project Title	**ROAD ACCIDENT MANAGEMENT SYSTEM**		
Project Duration	5 months (approx.)	Date of reporting	8 MAY 2022

Organization Details

Organization Name	**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, (IIIT) RANCHI**		
Full postal address with pin code	IIIT Ranchi, Science & Technology Campus, Khojatoli, Namkum, Ranchi, Jharkhand, 834010		
Website address	http://iiitranchi.ac.in/		

Supervisor Details

Supervisor Name	**Dr. Nidhi Kushwaha**		
Designation	Faculty (CSE Department)		
Full contact address with pin code			
Email address	nidhi@iiitranchi.ac.in	Phone No (M)	7703090242

Student3 Details

Name	**NEHA KUMARI**		
Register Number	2018UGCS056R	Section / Roll No	CSE/56
Email Address	nehagaur.btech.cs18@iiitranchi.ac.in	Phone No (M)	6307562606
Project Details			
Project Title	**ROAD ACCIDENT MANAGEMENT SYSTEM**		
Project Duration	5 months (approx.)	Date of reporting	8 MAY 2022
Organization Details			
Organization Name	**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, (IIIT) RANCHI**		
Full postal address with pin code	IIIT Ranchi, Science & Technology Campus, Khojatoli, Namkum, Ranchi, Jharkhand, 834010		
Website address	http://iiitranchi.ac.in/		
Supervisor Details			
Supervisor Name	**Dr. Nidhi Kushwaha**		
Designation	Faculty (CSE Department)		
Full contact address with pin code			
Email address	nidhi@iiitranchi.ac.in	Phone No (M)	7703090242