Vision based Accident Detection System for Smart City

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Electronics and Communication Engineering

by

Ashish Raj - 20UCC024
Anmol Tuli - 20UEC023
Yash Gaur - 20UEC149

Under Guidance of
Dr. Joyeeta Singha

Department of Electronics and Communication Engineering
The LNM Institute of Information Technology, Jaipur

April 2023

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled "Vision Based Accident Detection System for Smart City", submitted by Ashish Raj (20UCC024), Anmol Tuli (20UEC023) and Yash Gaur (20UEC149) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Electronics and Communication Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2022-2023 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this report is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

Date

Adviser: Dr.Joyeeta Singha

Acknowledgments

We take this opportunity to extend our sincere gratitude and appreciation to everyone who helped our Btech project to be completed successfully.

First and foremost, we want to express our gratitude to Dr Joyeeta Singha, who served as our project advisor and gave us crucial advice and assistance during the period of project implementation. We were indebted to you for your patience and encouragement and your insights and feedback have been instrumental in simulating discussions, analyzing problems associated with our project work and for guiding us throughout our project.Projet meetings were highly informative and useful.

We would also like to thank the head of department of Electronics and Communication Engineering , Dr Nikhil Sharma and other faculty members of the department for their valuable inputs and suggestions. Their constructive criticism and expert guidance have helped us in improving the quality of our work.

We would extend our sincere thanks to our friends and colleagues for their unwavering support and motivation. Their constant encouragement and belief in me have been a source of strength during this journey.

We are immensely indebted to everyone for valuable guidance and support to everyone without whom this project was not possible.

Ashish Raj -20UCC024

Anmol Tuli -20UEC023

Yash Gaur -20UEC149

Abstract

Globally, the number of distracted driving-related traffic incidents has increased, resulting in more injuries, property damage, and even fatalities. Therefore, it is essential to track and examine driver behaviour to spot distraction and lower the accident rate. An accident victim may go neglected for a considerable amount of time on routes with extremely light and quick traffic. The goal of Vision Zero is to end all traffic-related fatalities and serious injuries. However, reaction time is crucial in the event of an accident. The road network is being watched by surveillance cameras in several nations. Real-time monitoring, however, necessitates a sizable number of personnel with attention and training.

The goal of our project is to develop a accident detection system that can recognise an accident based on a live video feed from a CCTV camera mounted on roads. The idea taken is each frame of a video should be passed through a deep learning **Convolution Neural Network** model that has been trained to distinguish between accident and non-accident related video frames. The usage of a mobile, talking on the phone, eating, sleeping, or being distracted while driving are just a few examples of behaviours that machine learning and deep learning approaches can identify. However, these methods may require high computational capacity to train the models with a massive amount of data.

In this project, we have made an effort to develop CNN based method as an proposed solution 1 to detect distracted driver and to further improve its efficiency, we have used one of the architecture as **MobileNetV2** which is proposed solution 2 and have been adopted for transfer learning. The proposed model is trained using photos from a dataset that contains various postures or circumstances of a distracted driver, and the results are analysed to confirm its efficiency. The results suggest that this model can be a valuable tool for preventing road accidents caused by distracted driving.

Keywords: *accident detection, Convolution Neural Network, machine learning, deep learning, MobileNetV2, dataset.*

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Area of Work

Artificial intelligence have gained tremendous attention in recent years due to their ability to learn and make predictions from data. The area of our work is **machine learning** which enables computers to learn from data and anticipate the future without having to be explicitly programmed. A model is trained on a dataset, and then it is applied to new, unforeseen data to produce predictions. Applications for machine learning techniques include image recognition, natural language processing, and predictive modelling.

Deep learning is a subset of machine learning that involves the use of artificial neural networks to model and solve complex problems. These networks are made up of several interconnected layers of nodes, each of which contributes to the learning process in a different way. Deep learning algorithms have been used in a variety of applications, including picture and speech recognition.

Our project uses a specific type of deep learning algorithm which is Convolutional neural networks (CNN) that are particularly good at processing and analysing image and video input. They use a succession of convolutional layers to extract characteristics from images and are motivated by the anatomy and operation of the visual brain in animals. To develop predictions about the image's content, these features are subsequently transferred across fully connected layers. Hence area of machine and deep leaning making it valuable tools in such as surveillance and autonomous systems.

1.2 Problem Addressed

- Accidents are a major cause of death in India, and most deaths occur due to the lack of timely help reaching accident victims.

- Road traffic related deaths happen in developing countries, even though these countries have only half of the world's vehicles.
- With the present data, India is on the way to the number one country in deaths from road accidents due to the poor average record of 13 deaths every hour, which is about 140,000 per year. [5]

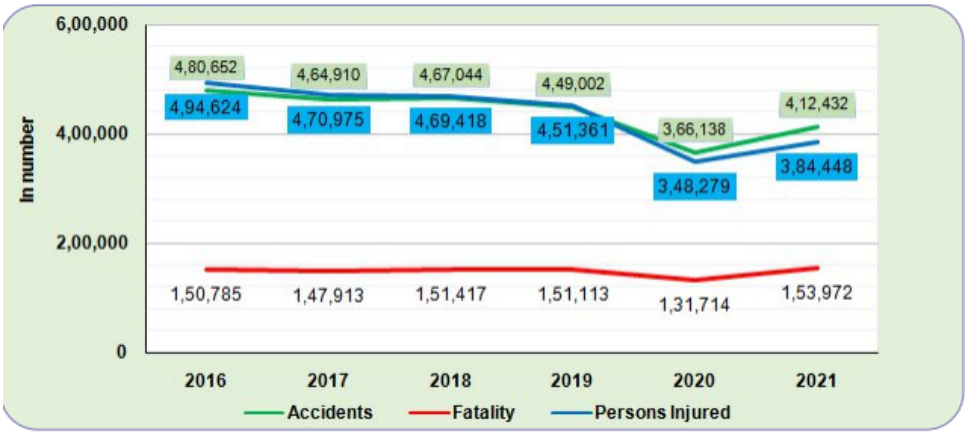

FIGURE 1.1: Trends in number of Accidents, Fatalities and Persons Injured: 2016 to 2021

The purpose of our project is to create a system which would detect an accident based on the live feed of video from a CCTV camera installed on the roads.

One of our motivation of the project is **Third Eye campaign** [6] Chennai has installed 34,293 cameras to monitor the speed activities which can be deployed for accident detection as well.

1.3 Existing System

One of the existing system is Intelligent Transport System based on Internet of Things (IOT) [7] This system detects accidents by vibration sensors, accelerometers. A signal from an accelerometer and a GPS sensor are automatically sent to the cloud and from there, an alert message will be received by whoever is subscribed to that car. The signal will indicate the severity of the accident and the GPS location. The ambulance can use the GPS coordinates to get to the scene quickly.

1.3.1 Disadvantages of Existing System

- Limited coverage area:These systems rely on the presence of sensors or devices to detect accidents and are typically placed in a specific area or location, which means that the coverage area of the system is limited.

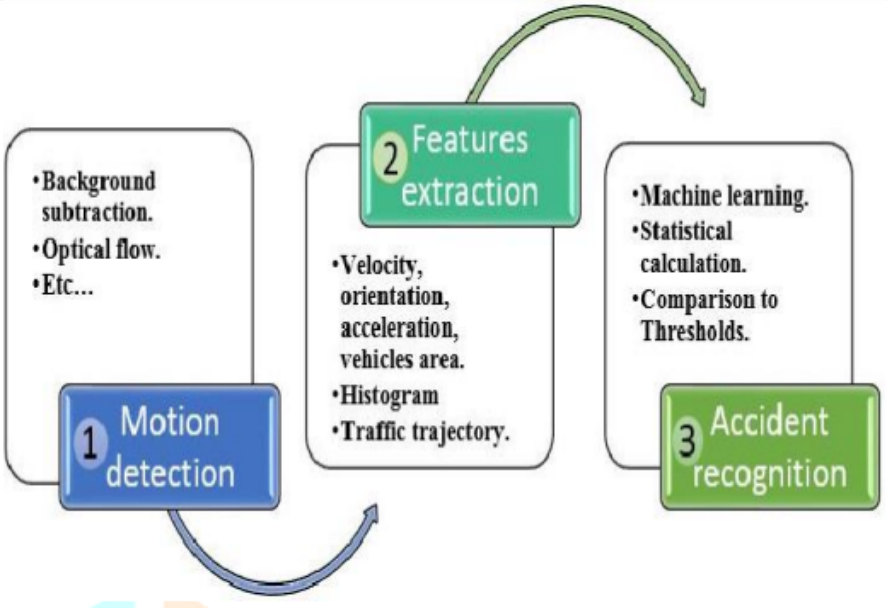

FIGURE 1.2: Existing System Architecture

- Cost and complexity: They can be expensive to install and maintain, particularly if they require specialized hardware or software. The complexity of the system can also be a significant disadvantage.
- False positives: They can sometimes generate false alarms, which can lead to unnecessary emergency responses and inconvenience for drivers.

Chapter 2

Literature Review

2.1 Accident Detection using Convolutional Neural Networks [1]

The research paper incorporates a system which is able to detect an accident form video footage provided to it using a camera. The system is designed as a tool to help out accident victims in need by timely detecting an accident and henceforth informing the authorities of the same. The focus is to detect an accident within seconds of it happening using advanced Deep Learning Algorithms which use Convolutional Neural Networks (CNN's or ConvNet) to analyze frames taken form the video generated by the camera.The proposed model is a fusion of CNN and LSTM layers for continuous video classification taken from a camera.In a CNN-LSTM network, the CNN is primarily used for feature extraction from the images, which is passed on to the LSTM for sequence prediction.

Accuracy: The paper reports an overall accuracy of 92.38 percent for the accident detection task.

Pros: The use of convolutional neural networks (CNNs) can improve the accuracy of the model in detecting accidents.

Cons: The paper does not provide information on the model's performance in detecting specific types of accidents or in different weather or lighting conditions.

2.2 Deep Learning applied to Road Accident Detection with Transfer Learning and Synthetic Images [2]

In this research paper, for testing, models with pre-trained weights were made available using the Keras Application Programming Interface (API).The accuracy of the forecast and the size of the model were the two main criteria used to choose the model. The choice of models

then involved balancing size and performance and the most effective model was built using EfficientNetB1. The framework used in the model is Tensorflow 2 along with Adam optimizer.

The metrics chosen were Accuracy, Precision, Recall, F-score (F1), and Matthews Correlation Coefficient (MCC). Also, the acronyms True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN) and Mean Average Precision (mAP) was used. The results were satisfactory for the data provided, with only 134 real images. Model MobileNetV2 had mAP of 0.87 and MCC of 0.68. Likewise, the EfficientNetB1 model had mAP of 0.89 and MCC of 0.78

Pros: Transfer learning and synthetic images can improve accuracy and reduce data needs; can be applied in real-time and integrated with traffic surveillance systems.

Limitations: Synthetic images may introduce bias or limitations; performance in detecting specific types of accidents or in different conditions is not provided.

2.3 Cnn Based Accident Detection, an AI based support system [3]

The reseach paper proposes the system using Deep Learning based on Conventional Neural Network (CNN).

The phases of proposed system includes:

- Collection of data for training purpose
- Preprocessing the data
- Model building
- Model Training
- Model validation
- Deploy as API
- Use API for prediction

The systematic steps for implementing supervised machine learning algorithms include convolution, max pooling, activation functions, model validation, and model deployment. The proposed architecture is scalable for retraining and can predict both accident occurrence and severity in a single model. Accident pictures are immediately shared with stakeholders through email or other communication methods.

2.4 Automatic driver distraction detection using deep convolutional neural networks [4]

Input Dataset:The input dataset for training the distracted driving detection model consists of thousands of images representing different distracted behaviors.

Preprocessing: Preprocessing of input images, including rotation and resizing, is necessary for accurate results in the distracted driving detection model.

Convolutional Neural Network:Different CNN architectures such as ResNet50, VGG16, and MobileNetV2 are incorporated in the distracted driving detection model to automatically extract high-level features from raw input images

Classification: CNN uses its learned experience from previous phases to identify the specific distracted behavior.

Results: The result show that MobileNetV2 provides highest training and testing accuracy around 99.98 percent and 98.12 percent respectively

Chapter 3

Proposed Work

3.1 CNN Based Accident Detection Model

3.1.1 Idea of the model

We will be first implementing the Accident Detection Model using Convolutional Neural Network which is based on Deep Learning in according to Reference Paper1[1] and Reference Paper3 [3]. We will be preparing training, validation, and testing datasets from image directories using the TensorFlow API and will build a sequential model using CNN .

Our model is trained on the preprocessed image data and then will be evaluated on a testing dataset and then we will evaluate accuracy of the model. [8]

Before discussing the model, we will talk about the python concepts used in the model.

3.1.2 Python Libraries and Frameworks

3.1.2.1 Numpy

NumPy (Numerical Python) is an open source Python library that's used in almost every field of science and engineering. It's the universal standard for working with numerical data in Python, and it's at the core of the scientific Python and PyData ecosystems. Users of NumPy range from novice programmers to seasoned academics working on cutting-edge academic and commercial research and development.

The following are NumPy's salient characteristics:

- NumPy offers several ways to generate arrays, including arrays made from lists, tuples, and other objects that are similar to arrays as well as utilising built-in functions.

- Indexing and Slicing of Arrays: NumPy offers a lot of support for indexing and slicing of arrays. This enables you to access particular array elements or subsets of those elements.
- Arrays of various sizes and forms can be subjected to arithmetic operations using NumPy's broadcasting capability.
- Mathematics: NumPy offers a wide range of mathematical operations, including as fundamental arithmetic operations, trigonometric functions, exponential functions, and more.
- Mathematical operations involving linear algebra, such as matrix multiplication and matrix decomposition, are supported by NumPy

3.1.2.2 Pandas

Pandas is an open-source data manipulation and data analysis library for the Python programming language. It offers a user-friendly and effective interface for data manipulation and analysis on top of the NumPy library.

The following are Pandas salient characteristics:

- Pandas offer a variety of data manipulation operations, including merging and reshaping datasets, grouping and aggregating data, and handling missing or duplicated information.
- Pandas' robust utilities for data visualization, time series analysis, and statistical analysis, including correlation, regression, and hypothesis testing, make it possible to do data analysis.
- Data Input and Output: A variety of data sources, including CSV files, Excel spreadsheets, SQL databases, and web APIs, can be read and written using Pandas..
- Pandas is a robust data analysis and manipulation tool since it integrates with other Python libraries like NumPy, Matplotlib, and Scikit-Learn.

3.1.2.3 Matplotlib

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. y. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, WX Python etc.

The following are Matplotlib salient characteristics:

- A broad variety of plotting options are available with Matplotlib, including those for line plots, scatter plots, bar plots, histograms, pie charts, heatmaps, and many more.

- The charts in Matplotlib can be highly customised. Almost every element of the plot, including the colors, line styles, markers, font sizes, axis labels, and titles, can be altered.
- Interactive Visualization: With the aid of the IPython shell, Jupyter notebook, and other contexts, Matplotlib offers interactive visualisation capabilities. Additionally, it supports interactive widgets and animation.
- Matplotlib integrates well with other Python libraries like NumPy, Pandas, and SciPy.

3.1.2.4 OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library.

- Processing of images and videos: OpenCV offers a variety of utilities for reading and writing photos and videos, converting colors, filtering, thresholding, edge detection, and more.
- Pre-trained models for object detection and tracking are available in OpenCV, including Haar Cascades and Deep Neural Networks. Additionally, it offers resources for building unique object trackers and detectors.
- OpenCV provides a camera calibration module that assists in computing a camera's intrinsic and extrinsic characteristics, which are crucial for applications like as 3D reconstruction, stereo vision, and augmented reality.

3.1.2.5 Tensorflow

TensorFlow is an open-source machine learning framework that is widely used in the field of artificial intelligence. It is a python library that supports many classification and regression algorithms and more generally deep learning. It is a free and open-source software library for dataflow and differentiable programming across a range of tasks. Some of its features are:

- TensorFlow is easy to use and can be used with Python, making it accessible to a wide range of developers. With the help of the TensorFlow API, developers can build and train machine learning models using pre-built models and libraries.
- TensorFlow is designed to run on multiple CPUs and GPUs, allowing for large-scale distributed computing. This capability makes it ideal for deep learning applications, which require intensive computational resources.
- TensorFlow can be used for a wide range of applications, including computer vision, natural language processing, and predictive analytics.

3.1.2.6 Keras

Keras is a popular high-level neural network API that is written in Python and built on top of TensorFlow. Here are five important points about Keras:

- Keras features a straightforward, user-friendly API and is made to be simple to use. This makes it simple for beginners to begin creating neural networks and for seasoned developers to quickly prototype and test new models.
- Keras offers a modular and adaptable architecture that makes it simple for developers to build unique neural networks and models. Additionally, it supports a variety of backends, including as TensorFlow, Theano, and CNTK, enabling developers to choose the backend that most closely matches their requirements.
- Predictive analytics, natural language processing, and picture classification are just a few of the numerous applications that Keras can be utilised for. It has a vast ecosystem of resources that may be used to create sophisticated models and find solutions.

3.1.2.7 Time module

The Python time module offers a number of functions for dealing with time-related activities, like calculating how long a programme will run, obtaining the current time, waiting a predetermined length of time, etc.

3.1.2.8 os module

The os module in Python gives users a means to communicate with the underlying operating system. For dealing with the file system, executing system instructions, and controlling processes, it provides a variety of features.

3.1.3 Methodology

The steps that are carried out in this CNN based model are as follows:

3.1.3.1 Dataset

We have collected images for the dataset using the Youtube videos and divided among three classes:

- Training Data: This will be used to train the data.We have total 791 files and they are bifurcated into two subsets : accident(369 files) and non accident(422 files). They are in the form of frames taken from a video.
- Testing Data: This will be used to test our model and improve its efficiency. We have a total of 101 files which are bifurcated into two parts: accident (47) and non accident(54).
- Validation Data: This will be used to check accuracy of our model or test reliability of our model.There are a total 98 files which will be used for validation.

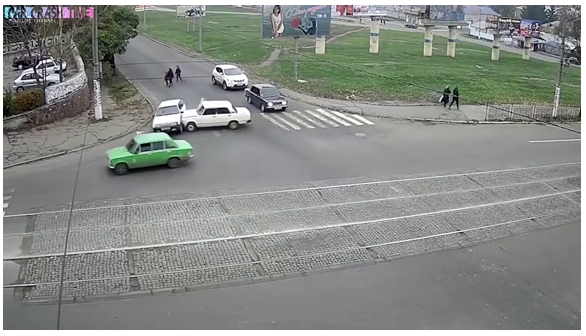

FIGURE 3.1: An Accident Image

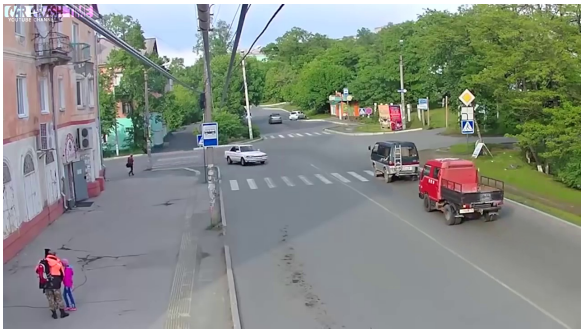

FIGURE 3.2: A Non-Accident Image

3.1.3.2 Preprocessing

In order to get reliable results, the photos must be processed before being fitted into the learning module. For instance, we must rotate and resize the input photographs for the model.

3.1.3.3 Convolutional Neural Network

A convolutional neural network (CNN) is a deep neural network used to classify images and detect object.It is used for image and video classification, segmentation, and identification applications is the convolutional neural network (CNN). It is a feed-forward neural network that takes the features from the input images using convolutional layers. The key benefits

of adopting CNNs over traditional neural networks are their capacity to handle images with varied input sizes and their automated learning of relevant features from the input data.

Similar to neural networks, CNNs [9]are composed of neurons with trainable biases and weights. Each neuron takes in a number of inputs, weights them, then sends the result through an activation function to produce an output. Each component of the network has a loss function, and all the techniques we discovered for neural networks still hold true for CNNs. More specifically, the output is produced after the image has been passed through a number of convolution, nonlinear, pooling, and fully connected layers.

Some of CNNs' main benefits include:

- Translation invariance: CNNs are more resistant to changes in the input data since they can identify objects in images regardless of their position or orientation.
- Features can be extracted automatically by CNNs from the input data, eliminating the requirement for manual feature engineering.
- Hierarchical learning: CNNs can learn progressively more complicated characteristics and representations of the input data by stacking numerous convolutional layers.
- Data augmentation: To avoid overfitting and boost generalization, CNNs can be trained on sizable datasets using data augmentation methods including random cropping, flipping, and rotation.
- Transfer learning: Using trained CNN models as feature extractors for new tasks eliminates the requirement for a lot of training data that has been labelled.

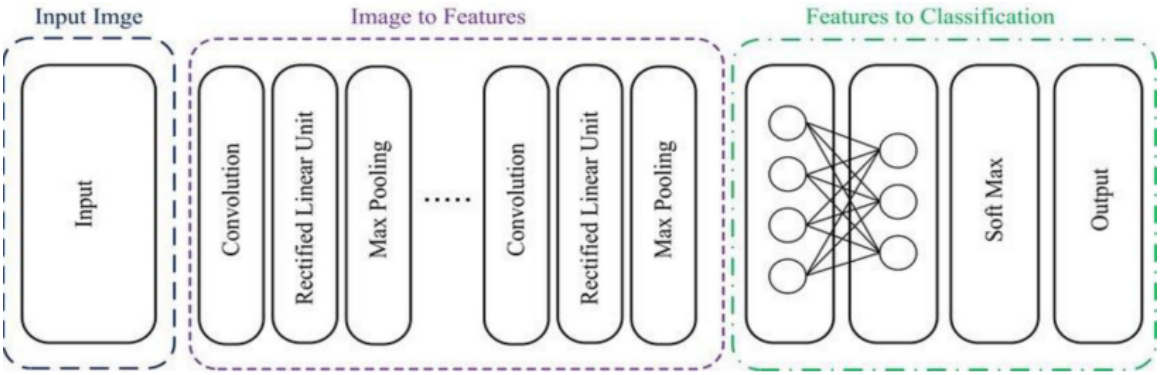

FIGURE 3.3: Architecture of Convolutional Neural Network

The above figures describes the architecture of CNN based model which consists of several layers. We will be implementing this model in our code and will describe CNN layers used in the model.

3.1.3.4 Layers in the CNN Model

A CNN basically consists of three types of layers namely: Convolutional, Pooling, Fully Connected Layer. For our CNN model we have used **Batch Normalization Layer, 3 convolutional Layers, 3 max-pooling layers , 2 dense layers and 1 flatten layer** and each are described below:

- **Normalization Layer**: A normalization layer is a layer used in neural networks to normalize the input data in order to improve the stability and convergence of the model. In order to enhance the performance of Convolutional Neural Networks (CNNs), normalisation layers are employed to normalise the activations of the preceding layer. The goal of normalisation is to increase uniformity in the activations of the layers, which could speed convergence and improve accuracy.

  In the model proposed we have used **Batch Normalization layer** , is a type of normalization layer that normalizes the input data by subtracting the batch mean and dividing by the batch standard deviation.

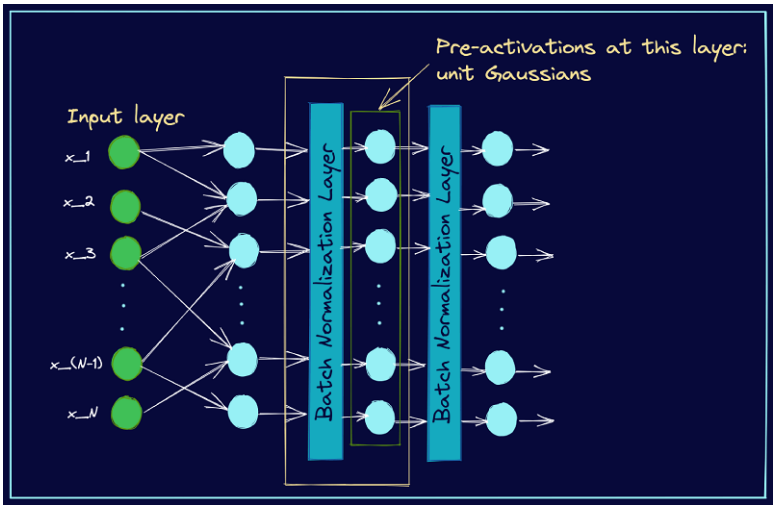

FIGURE 3.4: Section of a Neural Network with Batch Normalization Layer

  The benefits of batch normalization include faster convergence during training, improved stability of the model, and reduced sensitivity to the initialization of weights. It can also reduce the impact of vanishing and exploding gradients during training, which can lead to improved performance on tasks such as object recognition and image classification.

- **Convolutional Layer**: The convolution layer is the initial stage of image information extraction. Convolution assists in maintaining the link between the image's pixels by employing small squares of input data to learn image attributes.

  A tiny matrix known as a filter (or kernel) slides over the entire input image during the application of convolutional layers to the input image in order to extract features. The

types of features that are taken from the input image depend on the learnable weights that are part of each filter and are modified during training. Each filter moves over the input image, and a single output value—referred to as a feature map—is generated by computing the dot product of the filter weights and the associated pixel values from the image. To create the whole output feature map, the filter is then relocated to the next place and the procedure is repeated. [10]

The number of filters, size of the filters, and stride are three crucial parameters for convolutional layers. The quantity of feature maps that the layer will produce depends on the number of filters. The portion of the input image that is used to compute each output feature map depends on the size of the filter, which also affects the layer's receptive field. The stride controls how much the filter moves during each sliding operation.

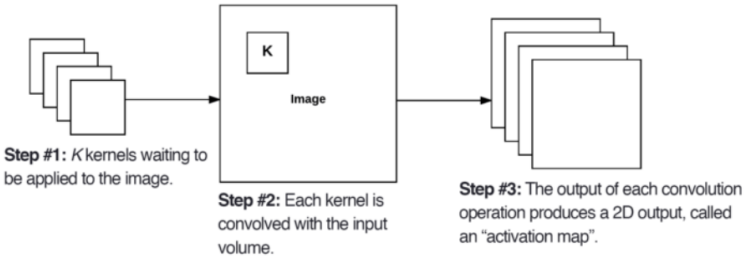

FIGURE 3.5: Working of Convolutional Layer

At each convolutional layer in a CNN, there are K kernels applied to the input volume. Each of the K kernels is convolved with the input volume. Each kernel produces a 2D output, called an activation map.

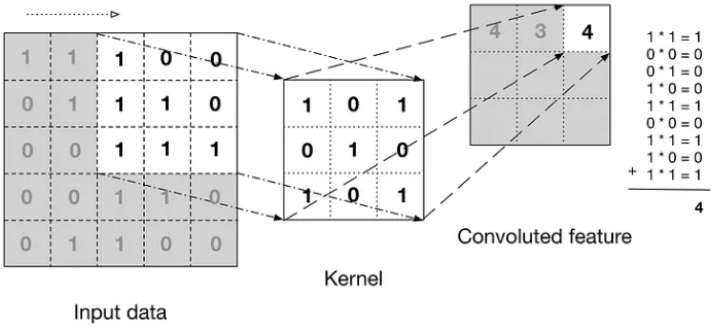

FIGURE 3.6: Convolutional Layer operation

In our model, we have used **Conv2D layer** which applies a set of learnable filters (also known as kernels) to the input image in order to detect specific patterns and features in the image. The first Conv2D layer has 32 filters, the second has 64 filters, and the third has 128 filters. Each filter produces a 2-dimensional output (a feature map) that represents the activation of that filter at that location in the input image.

- **Max-Pooling Layers**: Max pooling is a type of pooling layer commonly used in convolutional neural networks (CNNs) for image recognition tasks. A max pooling layer's

primary objective is to decrease the dimensionality of the input feature maps while keeping the most crucial features. This makes the network more resilient to changes in input and lowers the cost of processing.

Max pooling extracts the highest value possible from a sliding window with a specified size (often 2x2 or 3x3). A new output feature map with smaller spatial dimensions is created by shifting the window by a set stride and repeating the process.

MaxPooling2D used in the model : This helps to reduce the computational cost and make the model more efficient.

FIGURE 3.7: Max Pooling Layer

- **Flatten Layer**:This layer flattens the output of the previous layer into a 1-dimensional vector, which can be fed into a fully connected neural network. The flatten layer has no learnable parameters and only performs a simple operation of reshaping the input tensor. It does not affect the size or content of the input tensor in any other way.

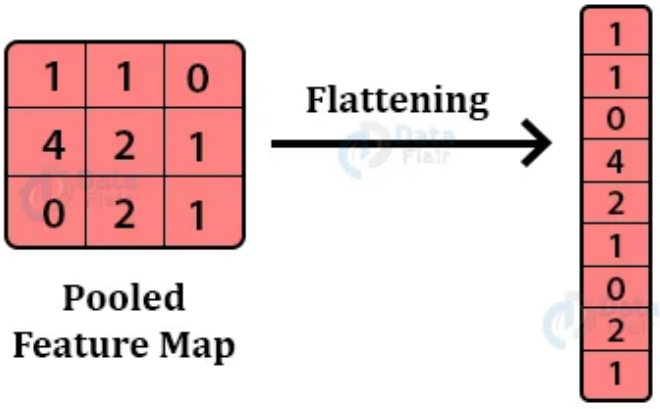

FIGURE 3.8: Flatten Layer

- **Dense Layer**: Dense layers are fully connected neural network layers that apply an activation function to the inputs and produce an output of a specified shape. In a dense layer, each neuron has a unique set of weights and biases that it acquires through training. Each neuron's output is determined by adding its bias to the weighted sum of its inputs, which is then processed by an activation function. The difficulty of the task being tackled often determines how many neurons are present in a dense layer. A more

complicated model that can capture intricate interactions between the input data and the goal variable can be created with a larger number of neurons.

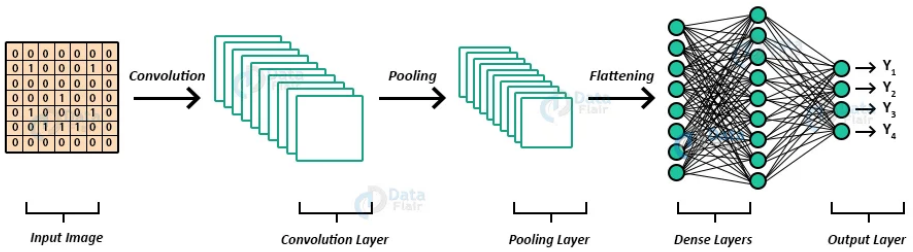

FIGURE 3.9: Dense Layer

Overall, the dense layer is an important building block of neural networks and is used extensively in deep learning models for a wide range of applications.

We have used Activation functions in CNN to introduce non-linearity to the output of a convolutional layer. In the model we have used:

- **Rectified Linear Unit (ReLU)** is an activation function that, if a number is negative, maps it to zero; otherwise, it maps it to itself. When training deep networks, the ReLU function has been proven to be particularly effective for networks with several layers since it can prevent vanishing gradients.

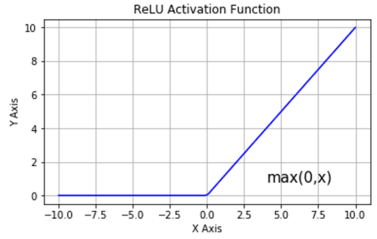

FIGURE 3.10: ReLU Activation Function

Some applications for the ReLU activation function in neural networks include the ones listed below:

ReLU has been discovered to have a faster convergence rate and to be more computationally efficient than other activation functions like sigmoid and tanh, which have a slower convergence rate because of the exponential function. ReLU provides a straightforward thresholding function, which causes training to converge more quickly.

Non-linearity: ReLU adds non-linearity to a neuron's output, enabling it to represent intricate connections between input and output.

By setting the negative values in the input to zero, ReLU promotes sparsity in the neural network. This sparsity aids in lowering overfitting and computational expense.

ReLU aids in preventing the issue of vanishing gradients during training, which is a typical problem.

- **SoftMax** is often used in the output layer of multi-class classification problems. It normalizes the output values to sum up to 1, giving the probabilities of the input belonging to each class.

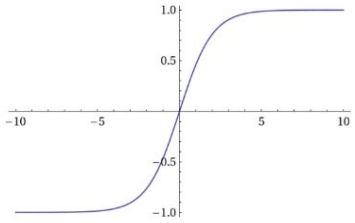

FIGURE 3.11: Softmax Activation Function

Adam Optimizer : Adam optimizer is a gradient descent optimization algorithm that is used to update the parameters of neural networks based on the training data. The Adam optimizer combines the advantages of two other optimization techniques - AdaGrad and RMSProp - and provides better results in terms of convergence speed and generalization. It is particularly useful for large datasets and deep neural networks. In our model, we will use adam optimizer to optimize the neural network during training.

Sparse categorical crossentropy : is a loss function that is typically used for multiclass classification problems where the classes are mutually exclusive. The predicted values are compared to the actual values, and the difference between them is measured using cross-entropy loss.

3.1.4 UML Diagrams

A partial graphical representation (view) of a model of a system that is being designed, being implemented, or that is already in use is called a UML diagram. A UML diagram contains graphical elements (symbols) that reflect the components of the intended system's UML model. These graphical elements are UML nodes connected with edges (also known as paths or flows). There may be additional documentation included in the system's UML model, such as use cases that have been written using a template.

UserCase Diagram : A use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. iven below is Activity Diagram of our CNN based model

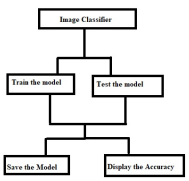

FIGURE 3.12: User Case Diagram of the CNN based model

Activity Diagram : An activity diagram is a behavioral diagram i.e., it depicts the behavior of a model. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

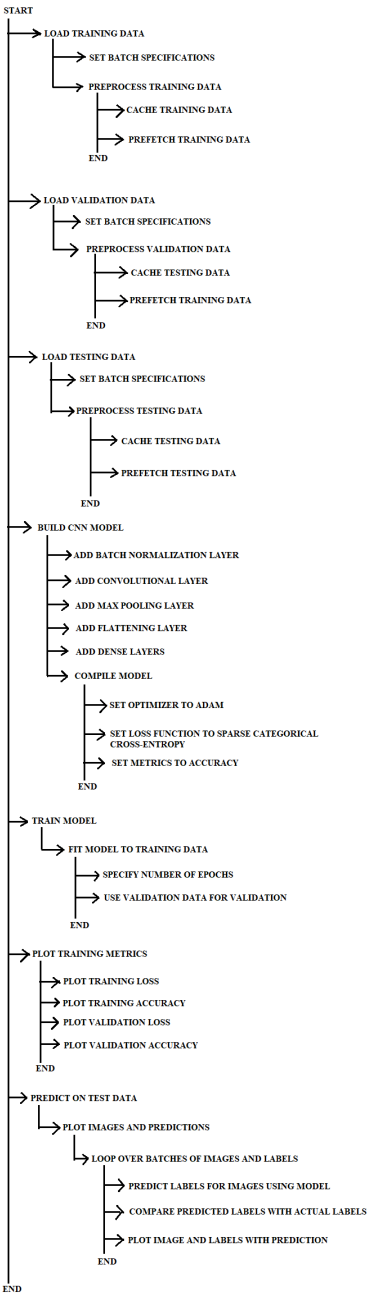

FIGURE 3.13: Activity Diagram of the CNN based model

Sequence Diagram : A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

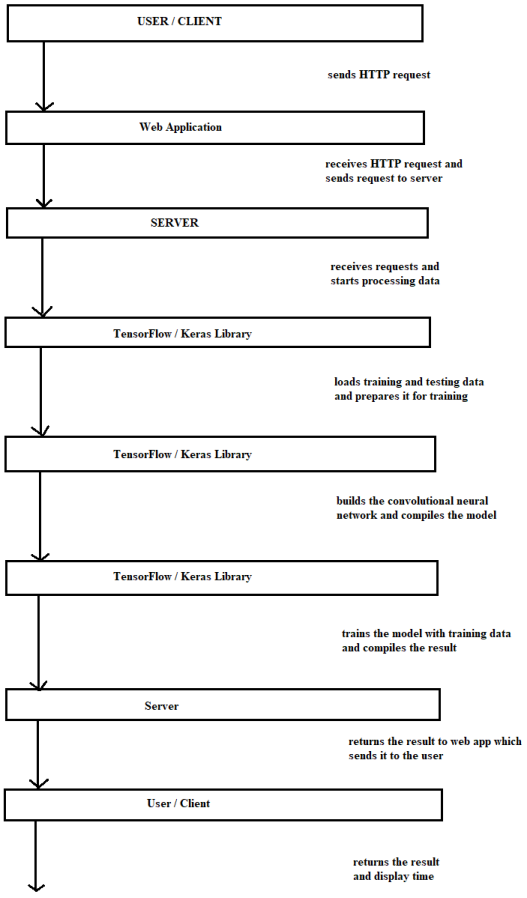

FIGURE 3.14: Sequence Diagram of the CNN based model

3.1.5 Algorithm of the CNN based Model

1. We will import the required libraries and modules such as numpy, pandas, matplotlib, tensorflow, keras layers, and os.
2. Define the batch size, image height, and image width for the dataset
3. We will load the training, validation, and testing dataset using the keras api from tensorflow framework.
4. We will get the class names of the dataset and cache and prefetch the training and testing datasets using Autotune.
5. We will define a convolutional neural network (CNN) using the keras Sequential model.
6. Compile the model by specifying the optimizer, loss function, and performance metric.
7. Fit the model to the training data while validating on the validation dataset for 10 epochs and store the returned values in retVal.

8. We will plot the training loss and accuracy using matplotlib. and define an empty accuracy vector.

9. We will iterate over the first batch of the testing dataset and make predictions using the trained model.

10. Convert the predictions into a class label and check the accuracy of the model by comparing the predicted label with the actual label.

11. We will plot the images from the testing dataset along with their predicted and actual labels and save the trained model to a file .

12. Finally we will plot the model architecture.

3.2 MobileNetV2 based System

3.2.1 Idea of proposed model

We have implemented the CNN based model and found that accuracy of the model was less. Based on the readings in the reseach paper2 [2] and reseach paper4 [4], our proposed system will be on **MobileNetV2** architecture based on transfer learning. According to the literature review, out of all architectures implemented, MobileNetV2 architecture has the best classification efficiency and therefore we are going to incorporate this in our model.

3.2.2 Methodology

To train the model with various poses, the image is first preprocessed in the proposal.Second, a pre-trained Convolutional model made up of various CNN-based deep learning architectures, MobileNetV2, has been used to detect driver distraction. Finally, the model is assessed using test data images, and the proposal's outcomes are examined.

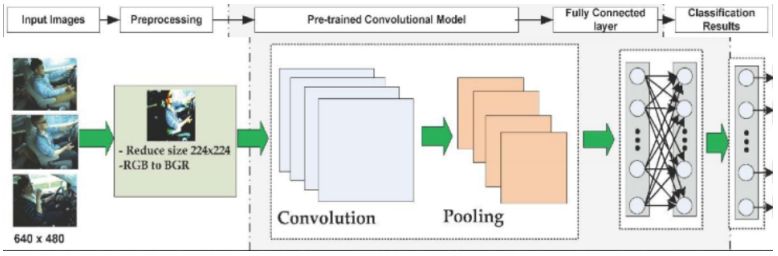

FIGURE 3.15: Work flow diagram of the MobileNetV2 based model

The steps carried out in proposal are as follows :

3.2.2.1 Dataset

We will be using this same dataset that we have used in CNN based model as discussed in Section 3.1.3.1.

3.2.2.2 Preprocessing

Preprocessing the images before fitting them into the learning module is essential to obtain accurate results. For example, we have to rotate and resize the system's input images.

3.2.2.3 Choice of the Architecture

We have studied in the reseach paper **Automatic driver distraction detection using deep convolutional neural networks** [4] , different architectures like ResNet50, VGG16, and MobileNetV2 are incorporated in their model. These architectures apply a filter to an input image to generate a feature map that summarizes the presence of detected features. Owing to the results of the architectural comparisions of the reseach paper, we have incorporated MobileNetV2 in our proposed system as it shows the maximum efficiency.

Table 1
Architectural Comparison of VGG-16, ResNet50 and MobileNetV2.

SL	Properties	VGG-16	ResNet50	MobileNetV2
1.	image	224×224×3	224×224×3	224×224×3
2.	weight	Imagenet	Imagenet	imagenet
3.	size	528	98	14
4.	Total layers	16	50	53
5.	Convolution layer	13	48	53
6.	Max pool	5	1	1
7.	Activation function	Softmax	Softmax	Softmax
8.	Total parameters	138.3 million	25.6million	3.5 million
Advantages/Limitations				
VGG16	• It is painfully slow to train. • Weights are quite large.			
ResNet50	• Lower complexity than VGG-16. • Deeper than VGG-16.			
MobileNetV2	• Faster than others. • Low-power models parameterized to meet the resource constraints. • Preferable in vision-based mobile and embedded applications.			

FIGURE 3.16: Figure taken from the research paper which shows architectural comparisions of different architectures

3.2.2.4 Classification

In this phase, the CNN identify each of the distracted behavior based on the learned experience of the previous phase.

3.2.3 Pre-trained Model Transfer Learning

To perform transfer learning, we first train a base network on a base dataset and task, after which we repurpose or transfer the gained features to a second target network, which will then be trained on a target dataset and task. The application of transfer learning can be applied to challenges with predictive modelling. The develop model approach and the pre-trained model approach are two of the most popular transfer learning strategies.

Transfer learning using a pre-trained model involves using the pre-trained model's learned features as a starting point for a new model, which is then fine-tuned to perform a specific task on a new dataset. Pre-trained CNN models learn low-level features from large image datasets, which can be used to extract deep features from new images.

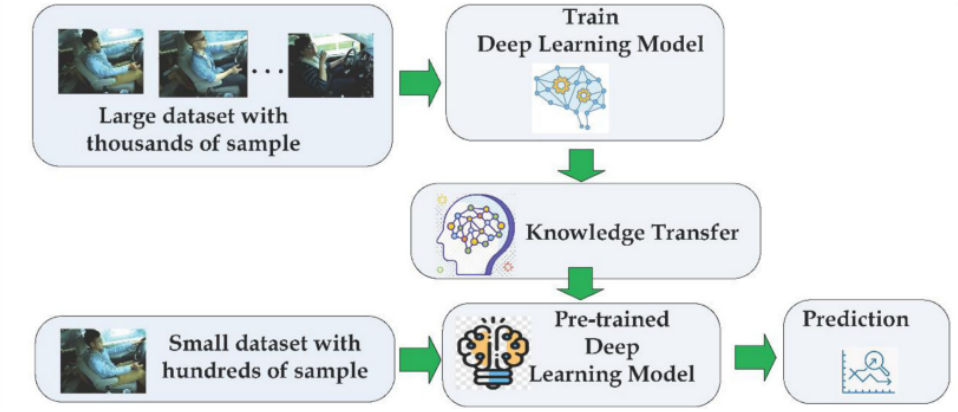

FIGURE 3.17: Transfer Learning

After being pre-trained on a sizable image dataset, MobileNetV2 architectures was able to develop a robust representation of low-level characteristics including edges, rotation, illumination, and patterns that may be utilised to extract deep features from fresh disturbing photos. In order to extract specific features utilising transfer learning principles, this model was helpful.

3.2.4 MobileNetV2 Architecture

MobileNetV2 is a deep convolutional neural network. It has 53 layers and is a powerful feature extractor that can detect and segment objects.The network is built on an inverted residual structure. [11] It has two main components:

- Inverted Residual Block
- Bottleneck Residual Block

There are two types of Convolution layers in MobileNet V2 architecture:

- 1x1 Convolution
- 3x3 Depthwise Convolution

There are Stride 1 Blocks and Stride 2 Blocks and each block has 3 different layers:

- 1x1 Convolution with Relu6
- Depthwise Convolution
- 1x1 Convolution without any linearity

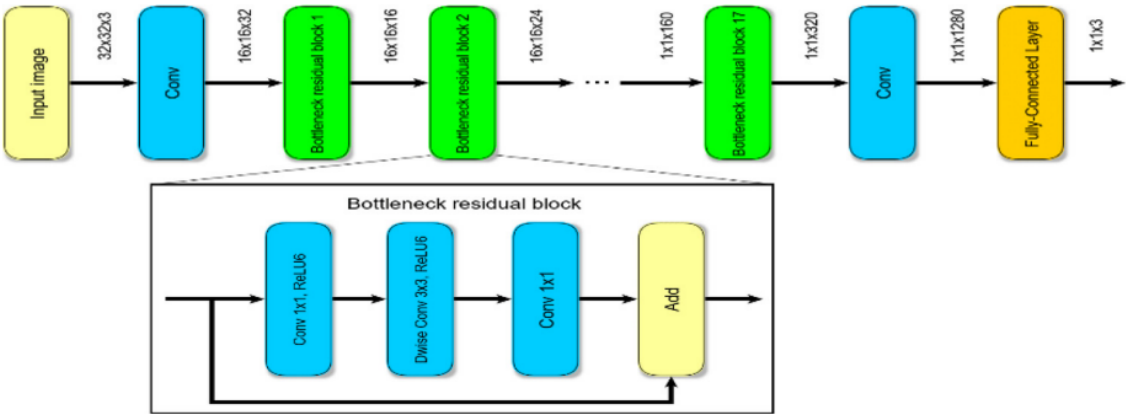

FIGURE 3.18: MobileNetV2 Architecture

MobileNetV2 uses its bottleneck layers to lower the computing cost of the network while keeping accuracy. In essence, the bottleneck layer architecture enables the network to execute a more sophisticated convolution operation with the least amount of parameters and computation needed. As a result, the network becomes more computationally efficient and is able to function on devices with lower processing capacity, such as mobile phones and embedded systems. Thus it aims to achieve a good balance between model accuracy and computational efficiency, making it well-suited for resource-constrained environments

3.2.5 Algorithm of MobilenetV2 based Model

1. Import necessary libraries which are cv2, numpy, pandas, matplotlib, tensorflow.
2. Define the batch size, image height, and image width variables.
3. Load the training, testing, and validation datasets using the Keras preprocessing module.
4. Configure the dataset for performance by caching and prefetching.
5. Define the image shape and create a base model using MobileNetV2.

6. Set the base model as non-trainable and create the final model by adding additional layers such as Conv2D, Flatten, and Dense layers.
7. Compile the final model using Adam optimizer and sparse categorical cross-entropy loss.
8. Fit the final model on the training dataset and validate on the validation dataset using the ”fit” function.
9. Plot the training and validation accuracy and loss using the ”plot” function from matplotlib.
10. Predict the labels of images from the testing dataset using the final model.
11. Plot the predicted and actual labels for a subset of images from the testing dataset.
12. Save the final model using the ”save” function from Keras.
13. Visualize the model architecture using the ”plot model” function from Keras.

3.2.6 UML Diagrams

Given below are the Usercase diagram, Activity Diagrams and Sequence Diagrams of MobileNetV2 based architecture.

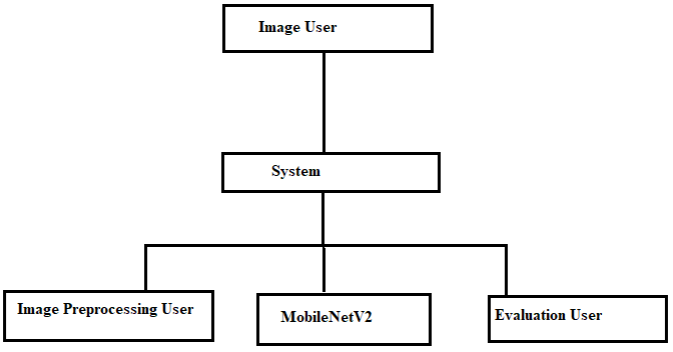

FIGURE 3.19: User Case Diagram of MobilenetV2 based model

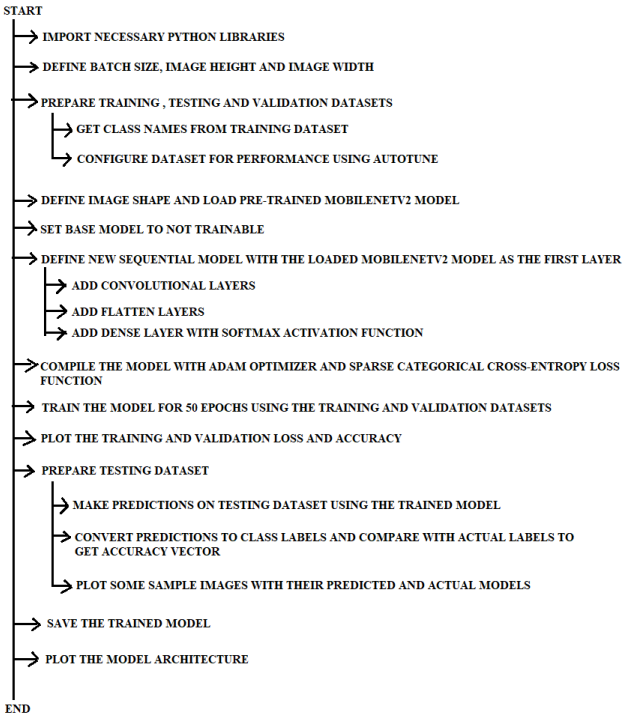

FIGURE 3.20: Activity Diagram of MobilenetV2 based Model

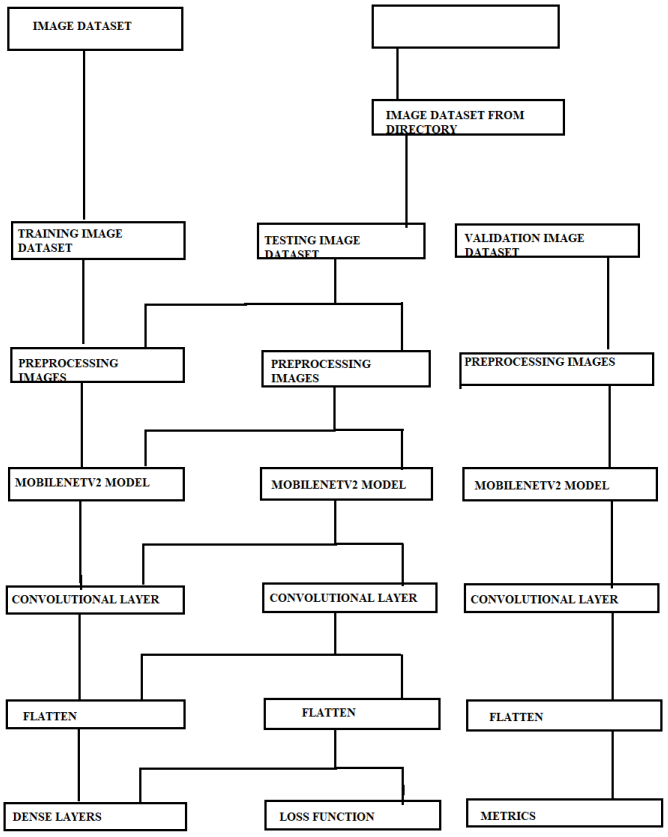

FIGURE 3.21: Sequence Diagram of MobilenetV2 based Model

Chapter 4

Simulation and Results

4.1 Simulation and Results of CNN based Model

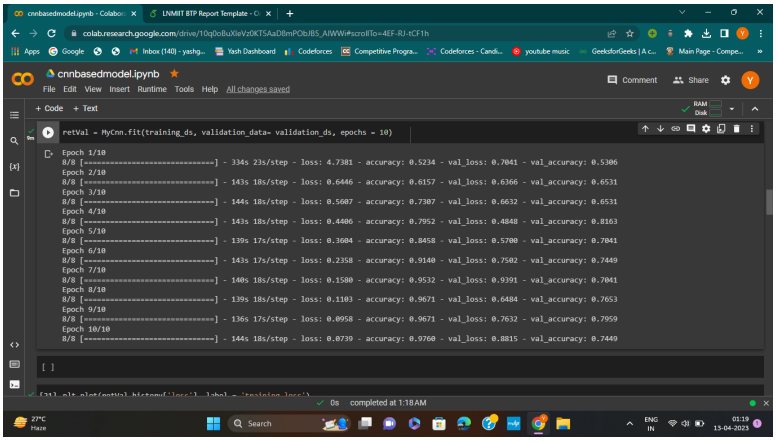

FIGURE 4.1: Training of the CNN-based Model

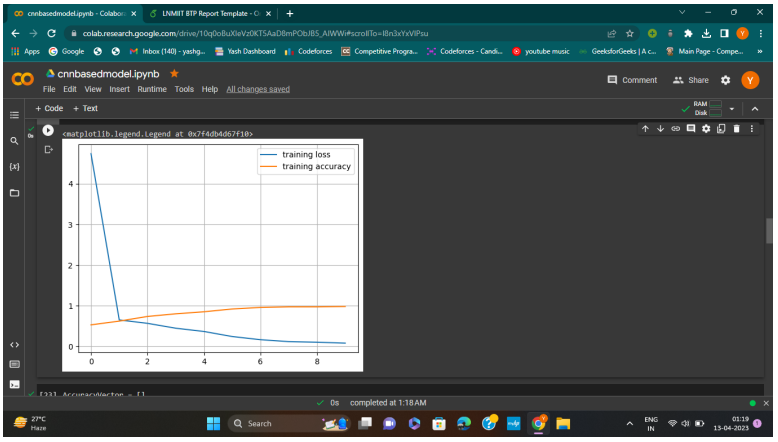

FIGURE 4.2: Graph between Training Loss and Training Accuracy

- The first figure shows the results of training of our CNN based model, that is how accurate our model is and its accuracy.

FIGURE 4.3: Visualizing Results on Testing Data

FIGURE 4.4: Sample Prediction of CNN Model

- The second figure shows the graph of Training Loss and Training accuracy .The training loss indicates how well the model is fitting the training data.
- The third figure shows the results on Testing data, what our model is depicting whether it is accident or non accident.
- The fourth figure shows the sample prediction.
- The final figure shows our implemented CNN based model

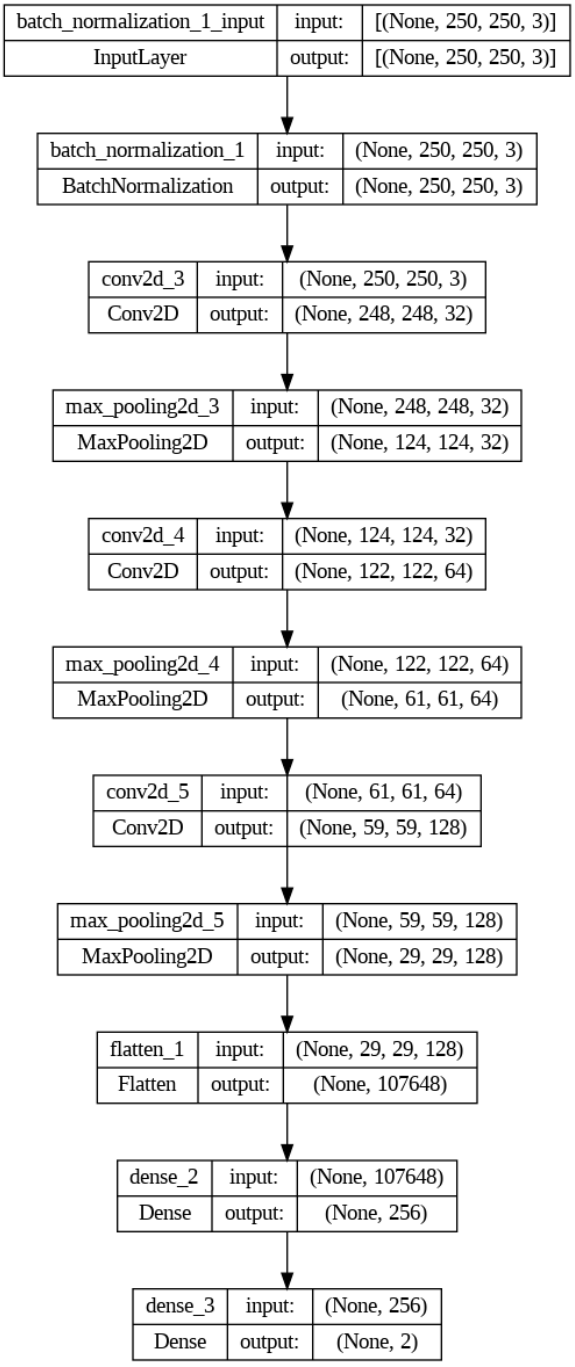

FIGURE 4.5: Architecture of our CNN model

4.2 Simulation and Results of MobileNetV2 based Model

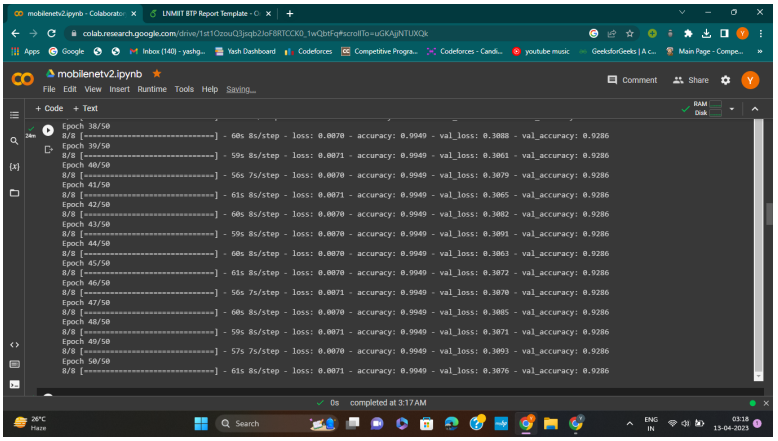

FIGURE 4.6: Training of the MobilenetV2-based Model and its accuracy

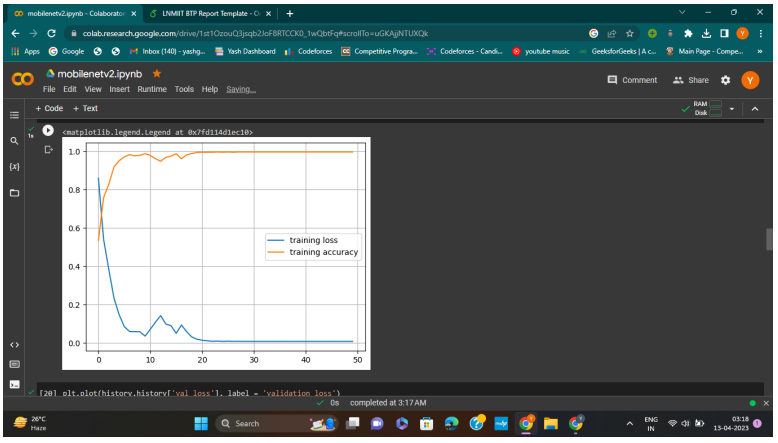

FIGURE 4.7: Graph between Training Loss and Training Accuracy

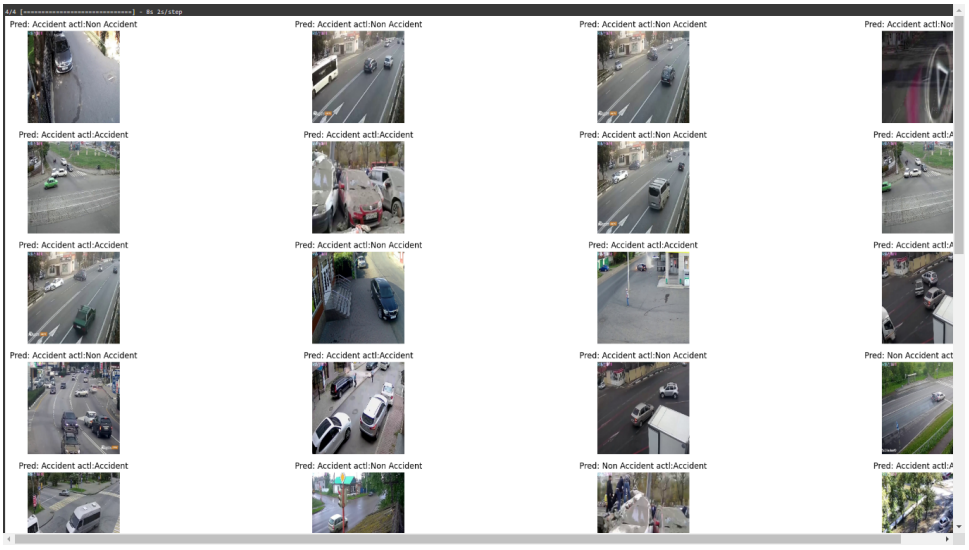

FIGURE 4.8: Visualizing Results on Testing Data

FIGURE 4.9: Sample Prediction of MobilenetV2 Model

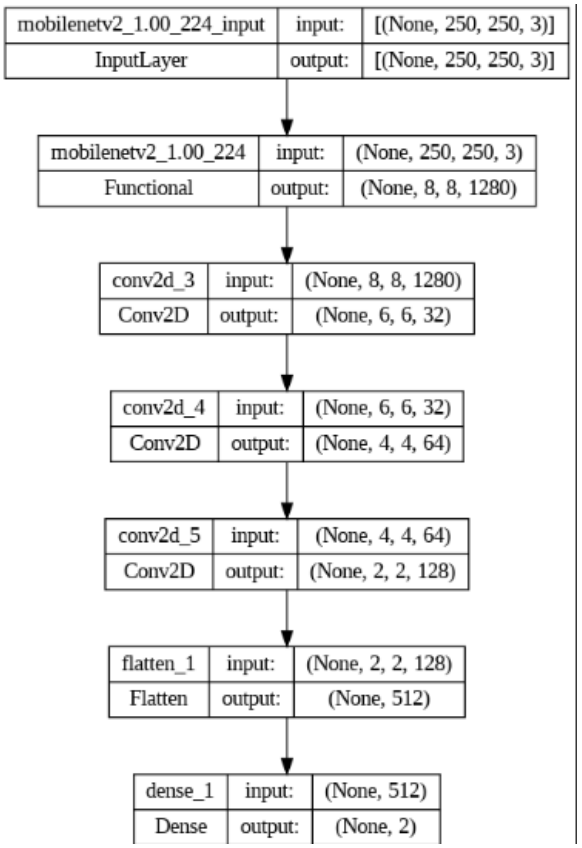

FIGURE 4.10: Architecture of our MobileNetV2 based model

Chapter 5

Conclusions and Future Work

Thee initiative to construct an Vision based Accident detection system for Smart City ” is a potential advancement in the realm of safety applications and computer vision. The model has demonstrated significant promise in properly identifying accidents and estimating the impact severity, which can aid emergency services in responding more quickly and perhaps saving lives. Other safety-related features, such lane departure warnings and pedestrian recognition, can also be added to the vehicle.

Incorporating the model with intelligent traffic management systems enables traffic controllers to react to problems rapidly and divert traffic to reduce congestion. In addition, the system can provide useful information on traffic patterns and accident hotspots that can be utilised to create more effective policies for improving infrastructure and road safety. In conclusion, our accident detection model can be a huge help in building smart cities that are safer and more effective. The technology can assist in reducing accidents, saving lives, and enhancing overall traffic management by giving useful data on traffic patterns and accidents as well as facilitating quick emergency responses.

In Future, we will implement our CNN based model and MobilenetV2 based model on long image sequencing, that is videos.

Bibliography

[1] S. J. S. Sreyan Ghosh, "Accident detection using convolutional neural networks," 2019.

[2] M. G. C. Tiago Tamaguskoa, "Deep learning applied to road accident detection with transfer learning and synthetic images," 2022.

[3] S. Prasanth, "Cnn based accident detection, an ai based support system," vol. 10, 2022.

[4] M. U. Hossaina, ""automatic driver distraction detection using deep convolutional neural networks"," 2022.

[5] M. of Road Transport and G. o. I. Highways, "Road accident in india- 2021," 2022.

[6] A. SELVARAJ, "A stye in chennai's third eye," 2021.

[7] F. Bhatti, ""a novel internet of things-enabled accident detection and reporting system for smart city environments"," 2019.

[8] A. T. S, "Accident detection and warning system," vol. 6, 2021.

[9] Y. Abouelnaga, ""real-time distracted driver posture classification"," 2018.

[10] M. Shahverdy, ""driver behaviour detection using 1d convolutional neural networks"," 2021.

[11] M. Sandler, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.