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

November 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 2023-2024 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 would want to take this opportunity to express our deep gratitude and appreciation to everyone who helped in 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 tenure of entire project. We are grateful for your understanding and support, as well as your views and comments, which have been invaluable in helping us simulate conversations, identify issues related to our project, and provide direction for the duration of the project.Project 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 insightful advice and helpful critiques have enabled us to raise the caliber 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 owe a huge debt of gratitude to everyone who helped make this project possible by providing invaluable advice and support.

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. Reaction time is important, nevertheless, in case of an accident. Several countries have surveillance cameras monitoring the road network. Real-time monitoring requires a large number of people who are trained and attentive.

Our project's objective is to create an accident detection system that can identify an incident from a live video feed provided by a CCTV camera placed on a road. The concept is to run every video frame through a deep learning **Convolution Neural Network** model that has been trained to discriminate between video frames related to accidents and those that are not.

We have attempted to create a CNN-based method as a suggested solution 1 to identify drivers who are distracted and to enhance its effectiveness., 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. Finally, we used **Transfer Learning with a SpinalNet architecture** to implement a model. The model's ability to capture complex features that are essential for accident detection is improved by the SpinalNet. A thorough preprocessing process is applied to the dataset, which includes data augmentation and normalization.

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

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Area of Work

The capacity of artificial intelligence to learn from data and make predictions has garnered a great deal of attention in recent years. Our area of expertise is **machine learning**, which allows computers to understand the world around them and predict future events without explicit programming. A model is trained on a dataset, and then it is applied to new, unforeseen data to produce predictions. Machine learning techniques find diverse applications, ranging from recognizing images and NLP to predictive modeling, showcasing their versatility in various domains.

(textbf)A subfield of machine learning called deep learning uses artificial neural networks to solve complex problems. These networks are made up of numerous interconnected node layers, each of which has a specific function in the learning process. Speech and recognition of images are just two of the many applications for deep learning algorithms.

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

- In India, accidents stand as a prominent contributor to fatalities, with a significant number of deaths attributed to the delayed arrival of assistance for accident victims.

- Developing nations account for only half of all vehicle ownership worldwide, but they experience a high rate of road traffic-related fatalities.
- With the current data, India is headed toward becoming the nation with the highest number of road accident deaths due to its dismal average record of 18 deaths per hour, or roughly 150,000 deaths annually. [6]

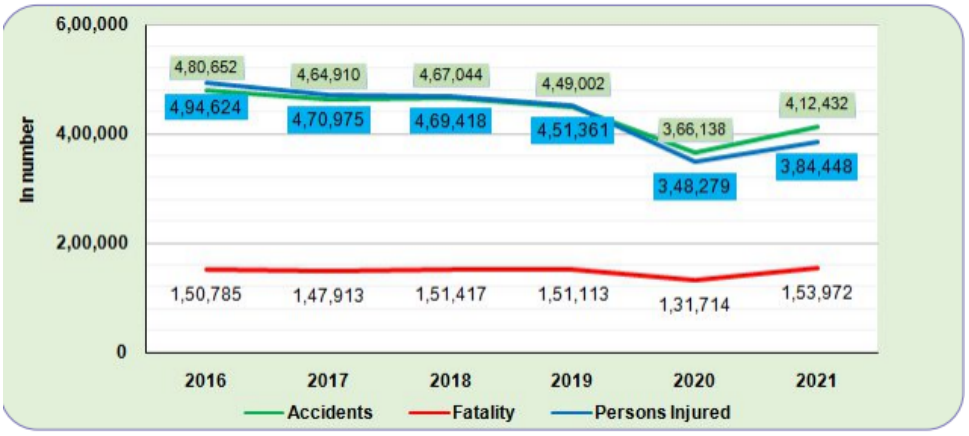

FIGURE 1.1: Trends in the quantity of collisions, fatalities, and injuries,2016 to 2021

Our project aims to develop a system capable of identifying accidents by analyzing the real-time video feed obtained from CCTV cameras on roads.

The **Third Eye campaign** is one of the reasons we are working on the project. 34,293 cameras have been installed in Chennai to monitor speed-related activities. These cameras can also be used to detect accidents.

1.3 Existing System

An established system in place is the Intelligent Transport System utilizing the Internet of Things (IoT). In this system, accidents are identified through vibration sensors and accelerometers. The system automatically transmits signals from both the accelerometer and GPS sensor to the server, where subscribed individuals receive an alert message. This message includes details on the accident's severity and the location, enabling swift response of GPS, particularly for emergency services like ambulances, which can leverage the GPS coordinates to reach the incident site

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.

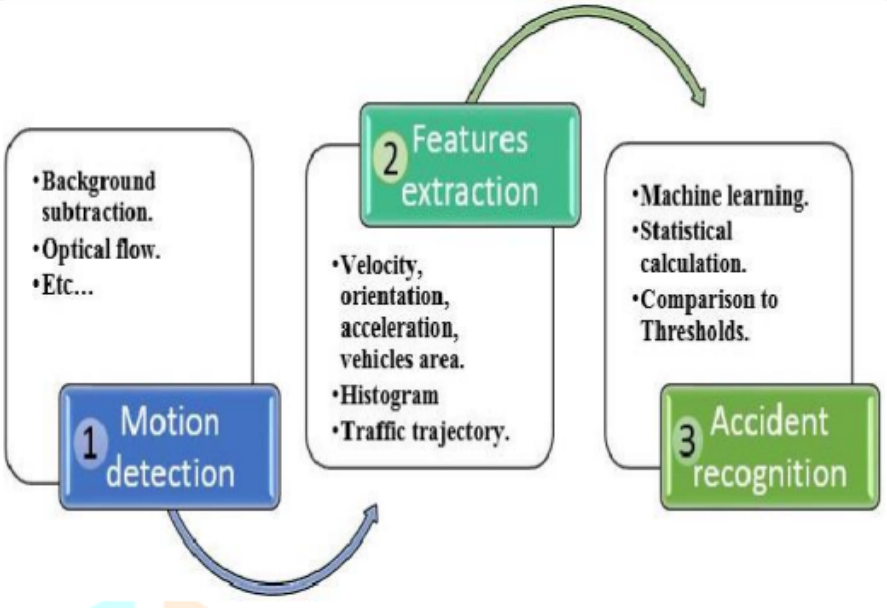

FIGURE 1.2: Existing System Architecture

- 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 Using Convolutional Neural Networks for Accident Detection [1]

A system that can identify an accident from video footage fed to it by a camera is included in the research paper. By promptly identifying an accident and alerting the authorities to it, the system is intended to assist accident victims who are in need. The goal is to use sophisticated Deep Learning Algorithms that use Convolutional Neural Networks, also known as ConvNets, to analyze frames from the video that the camera captures in order to detect an accident as soon as possible.For continuous video classification from a camera, the suggested model combines CNN and LSTM layers.A CNN-LSTM network uses the CNN mainly to extract features from the images, then sends the extracted features to the LSTM for

Accuracy: For the accident detection task, the paper reports an 92.38 results. Advantages: Using convolutional neural networks (CNNs) can increase the model's accident detection accuracy.

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 Applying Transfer Learning and Synthetic Images to Deep Learning for Road Accident Detection [2]

Using the Keras (API), models with pre-trained weights were made available for testing in this research paper.The two primary factors that went into selecting the model were its size and the forecast's accuracy. 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.

Some metrics were choosen, one of them was mcc. and model's accuracy was determined. The outcomes were deemed satisfactory given the dataset's limited size, consisting of only 134 authentic images. The MobileNetV2 model achieved a mean average precision (mAP) of 0.87 and a Matthews correlation coefficient of 0.68.

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 DL system based on Conventional Neural Network (CNN) is proposed in the research paper.

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 Automated detection of driver distraction through the use of deep convolution 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 (CNN): Various CNN architectures, including ResNet50, VGG16, and MobileNetV2, are integrated into the distracted driving detection model for the automated extraction of advanced features from raw input images.

Classification: Leveraging its acquired knowledge from earlier stages, the CNN employs its expertise to recognize distinct distracted behaviors.

Results: The results indicate that MobileNetV2 offers the highest levels of testing and training accuracy,

2.5 A Deep Neural Network with Incremental Input: SpinalNet [5]

The reasearch paper implements a SpinalNet neural network, inspired by the human spinal cord architecture, designed to achieve superior accuracy with reduced computational costs. Unlike traditional neural networks (NNs), SpinalNet introduces three splits in each layer: input, intermediate, and output. This unique structure significantly lowers incoming weights, enhancing accuracy with fewer computations.

SpinalNet offers a novel approach to neural network design with its gradual input, and likely global influence, all of which mimic the characteristics of the human spinal cord. The network architecture, which consists of layers, allows for a special configuration that strikes a compromise between computational efficiency and accuracy.

Accuracy: The paper reports an overall accuracy of 90.97 percent average and 91.47 percent best accuracy for the accident detection task.

Advantages: SpinalNet solves the computational intensiveness for large inputs problem that plagues feedforward neural network models. In contrast to conventional feedforward models, SpinalNet uses less computational power because it makes decisions locally, much like the spinal cord, gradually accepting inputs.

Cons: While the paper suggests potential applications in real-world scenarios. The challenge lies in ensuring that the model generalizes well across diverse and complex real-world datasets, and achieving this may require additional research and experimentation.

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 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 .

After our model has been trained using preprocessed image data, it will be assessed on a testing dataset to determine the model's accuracy.

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 utilized in practically all scientific and engineering domains. It is the foundation for working with numerical data in Python. From inexperienced programmers to seasoned scholars working on cutting-edge scholarly and commercial research and development, NumPy is used by a wide spectrum of users.

The key features of NumPy are as follows:

- 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.
- NumPy supports linear algebraic mathematical operations.

3.1.2.2 Pandas

Pandas is a Python programming language data manipulation and analysis library.. 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: Pandas can be used to read and write data from a wide range of sources.
- Pandas is a robust data analysis and manipulation tool since it integrates with other libraries.

3.1.2.3 Matplotlib

Matplotlib serves as a library, facilitating data visualization and graphical plotting in Python alongside its numerical extension, NumPy. This library offers an object-oriented API that allows the seamless integration of plots into applications through general-purpose GUI toolkits.

The following are Matplotlib salient characteristics:

- Matplotlib offers a wide range of plotting options, such as histograms.
- 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

- 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

An open-source machine learning framework called TensorFlow is popular in the artificial intelligence community. This Python library facilitates a wide range of classification and regression algorithms, as well as deep learning in general. It is an open-source, free software library that can be used for a variety of tasks involving dataflow and differentiable programming. Among its attributes 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 have many uses.

3.1.2.6 Keras

Built on top of TensorFlow, Keras is a well-known high-level network API written in Python. 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

Using the YouTube videos, we gathered the images for the dataset and separated them into 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

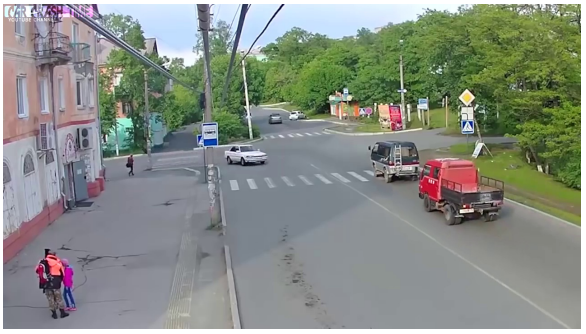

FIGURE 3.2: A Non-Accident Image

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) stands as a deep neural network utilized for image classification and object detection. Particularly applied in tasks related to image and video classification, segmentation, and identification, CNNs operate as feed-forward neural networks that extract features from input images through convolutional layers. Distinguishing themselves from traditional neural networks, CNNs offer advantages in handling images with diverse input sizes and autonomously learning relevant features from input data.

In a manner akin to neural networks, CNNs consist of neurons endowed with trainable biases and weights. Each neuron receives multiple inputs, applies weights, and passes the result through an activation function to generate an output. The fundamental principles governing loss functions and other techniques applicable to neural networks extend seamlessly to CNNs. More precisely, the network produces output after the image undergoes a sequence of convolution, nonlinear, pooling, and fully connected layers.

Some noteworthy advantages of CNNs encompass:.

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: By piling up multiple convolutional layers, CNNs are able to learn increasingly complex properties and representations of the input data.
- 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: A large amount of labeled training data is not needed when using trained CNN models as feature extractors for new tasks.

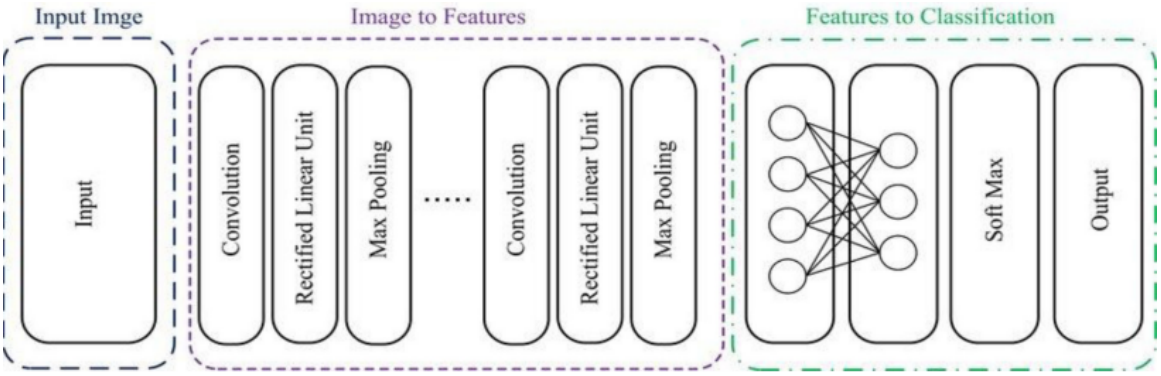

FIGURE 3.3: Architecture of Convolutional Neural Network

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

Convolutional, pooling, and fully connected layers make up the three main layer types of a CNN. We used the following layers for our CNN model: **Batch Normalization Layer, convolutional layers which are 3, max-pooling layers counting to 3, 2 dense layers, and 1 flatten layer**.

- **Normalization Layer**: To improve the stability and convergence of the model, normalizing the input data is done in the normalization layer of neural networks. Normalisation layers are used to normalise the activations of the layer before them in order to improve

the performance of convolutional neural networks (CNNs). Increasing uniformity in the layer activations is the aim of normalisation, which may hasten convergence and enhance accuracy.

In the model proposed we have used A sort of normalization layer known as a **Batch normalization layer** divides the input data by the batch standard deviation following the deduction of the batch mean.

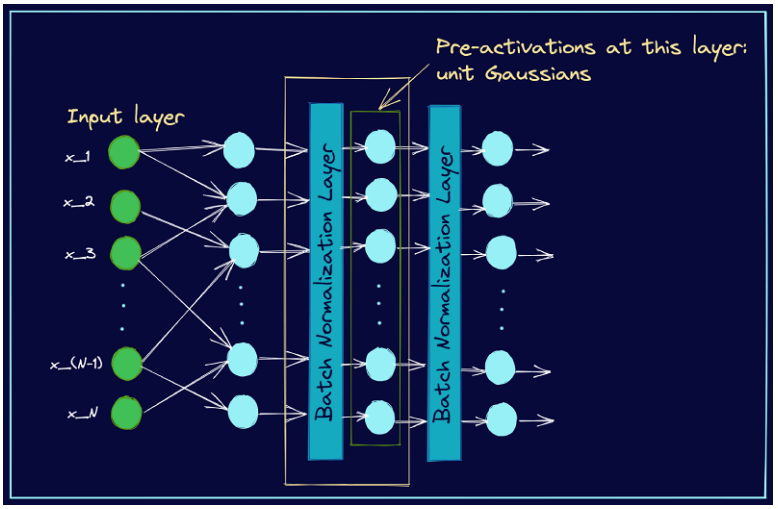

FIGURE 3.4: A Neural Network Segment Featuring Batch Normalization Layer

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 first step in extracting information from an image is the convolution layer. Convolution uses tiny squares of input data to learn about the characteristics of the image, which helps to preserve the connection between the pixels in the image.

  When convolutional layers are applied to an input image, a small matrix called a filter (or kernel) slides across the whole image 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. [7]

  The number of filters, size of the filters, and stride are three crucial parameters for convolutional layers. 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.

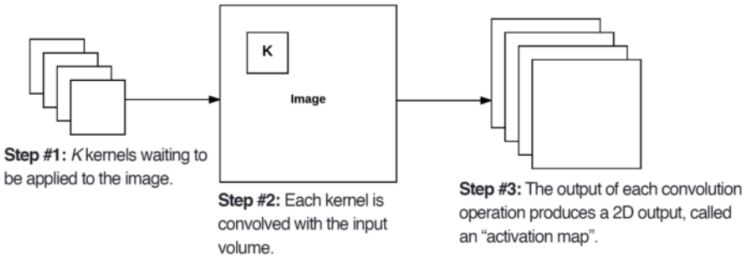

FIGURE 3.5: Working of Convolutional Layer

In a CNN, K kernels are applied to the input volume at each convolutional layer. The input volume is convolved with each of the K kernels. An activation map is the 2D output that each kernel generates.

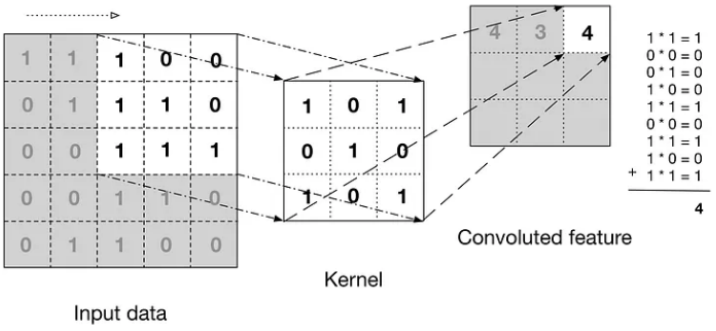

FIGURE 3.6: Convolutional Layer operation

Our model employs the **Conv2D layer**, which employs a collection of learnable filters, commonly referred to as kernels, to analyze the input image and identify particular patterns and features. There are 32 filters in the first Conv2D layer, 64 filters in the second, and 128 filters in the third. Every filter generates a two-dimensional output (a feature map) that shows where in the input image the filter is activated.

- **Max-Pooling Layers**: Convolutional neural networks (CNNs) frequently employ max pooling, a particular kind of pooling layer, for image recognition applications. Reducing the dimensionality of the input feature maps while retaining the most important features is the main goal of a max pooling layer. This reduces processing costs and increases the network's resilience to changes in input.

  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 improves the efficiency of the model.

- **Flatten Layer**: This layer creates a 1-dimensional vector from the output of the preceding layer that can be fed that is fully connected. The flatten layer only reshapes the input tensor; it does not have any learnable parameters. It has no other effect on the input tensor's size or content.

FIGURE 3.7: Max Pooling Layer

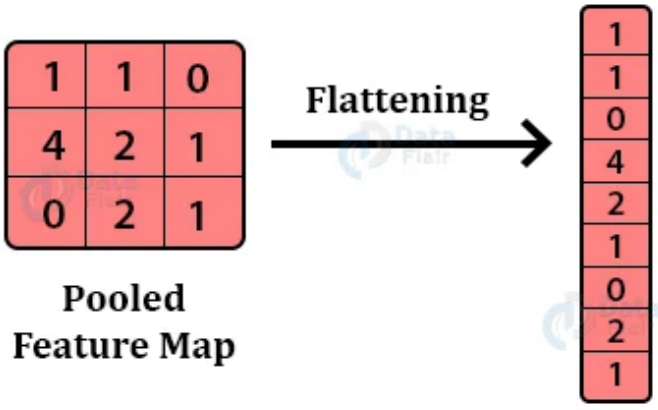

FIGURE 3.8: Flatten Layer

- **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.

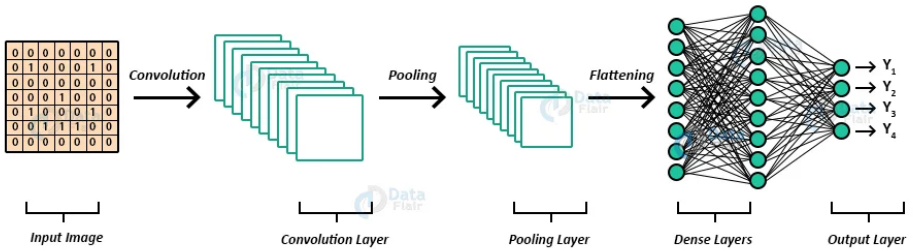

FIGURE 3.9: Dense Layer

Overall, the dense layer is an important building block of neural networks and is used extensively in many applications.

In the model:

- **T**he activation function Rectified Linear Unit (ReLU) maps a number to itself if it is positive and to zero otherwise. Because it can stop vanishing gradients, the ReLU function has been shown to be especially useful for training multi-layer deep networks.

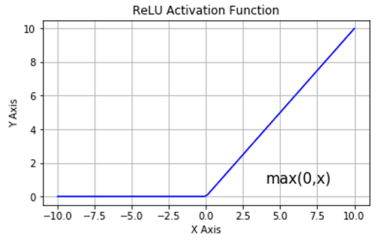

FIGURE 3.10: ReLU Activation Function

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.

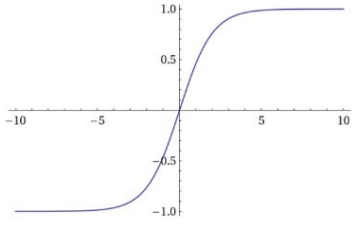

FIGURE 3.11: Softmax Activation Function

Adam Optimizer : Neural network parameters are updated using the gradient descent optimization algorithm known as the Adam optimizer, which is applied to the training data. The Adam optimizer outperforms the AdaGrad and RMSProp optimizers in terms of convergence

speed and generalization by combining their advantages. Deep neural networks and huge datasets benefit most from it. The neural network in our model will be optimized during training using the Adam Optimizer.

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 Algorithm of the CNN based Model

1. The necessary modules and libraries, including numpy, pandas, matplotlib, tensorflow, keras layers, and os, will be imported.
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. Once the optimizer, loss function, and performance metric have been specified, compile the model.
7. After validating the model for ten epochs on the validation dataset, fit the model to the training data 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 model was not so accurate. Based on the readings in the reseach paper2 [8] 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

The proposal preprocesses the image first in order to train the model with different poses.Second, driver distraction has been identified using MobileNetV2, [2], composed of multiple CNN-based deep learning architectures. Lastly, test data images are used to evaluate the model.

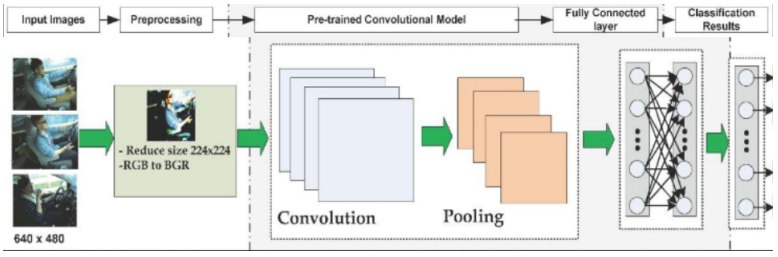

FIGURE 3.12: Work flow diagram of the MobileNetV2 based model

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

Accurate results require preprocessing the images before integrating them into the learning module. For instance, the system's input images need to be resized and rotated.

3.2.2.3 Choice of the Architecture

Several architecture are incorporated in the research paper **Automatic driver distraction detection using deep convolutional neural networks** [**?**]. These architectures take an input

image and apply a filter to show where the features are located. Based on the findings of the research paper's architectural comparisons, we have integrated MobileNetV2 into our suggested system since it exhibits the highest level of 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 3.13: Figure taken from the research paper which shows architectural comparisions of different architectures

3.2.2.4 Classification

Based on the lessons learned in the previous phase, the CNN recognizes every instance of distracted behavior in this phase.

3.2.3 Pre-trained Model Transfer Learning

Firstly, a foundational dataset and task are used to train a base network through transfer learning. Subsequently, the acquired features are either repurposed or transferred to a different target network, which undergoes training using a distinct target dataset and task. It proves particularly advantageous for addressing predictive modeling challenges. The pre-trained model method and the development model approach stand out as two widely employed strategies in transfer learning.

When a pre-trained model is used for transfer learning, its learned features are the starting point, and it is then refined to complete a specific task on a new dataset. Pre-trained CNN models, which gather low-level features from large image datasets, can be used to extract deep features from new images.

After being pre-trained on a sizable image dataset, MobileNetV2 architectures were able to produce a robust representation of designs. Deep features from fresh, unsettling images can be extracted using these features. This model proved helpful in identifying specific features through the application of transfer learning principles.

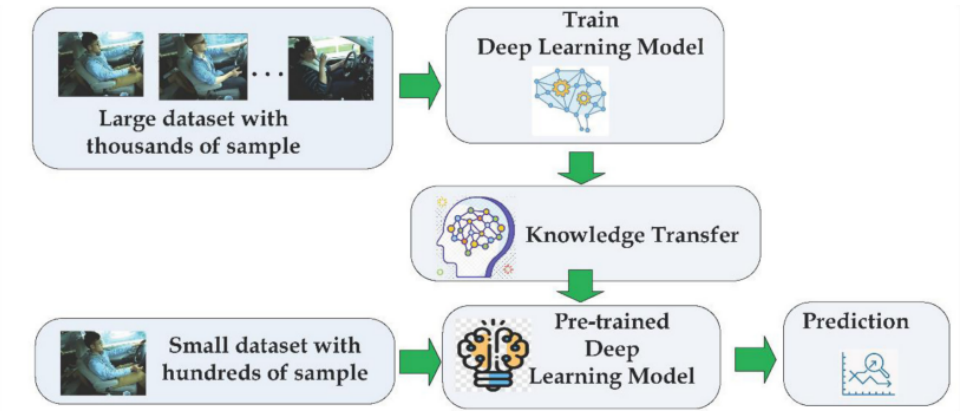

FIGURE 3.14: Transfer Learning

3.2.4 MobileNetV2 Architecture

A deep convolutional neural network is called MobileNetV2. It is a strong feature extractor with 53 layers that can identify and divide objects.On an inverted residual structure, the network is constructed. [**?**] It consists of two primary parts:

- Bottleneck Residual Block with Bottleneck structure.
- Residual Block with Inverted structure.

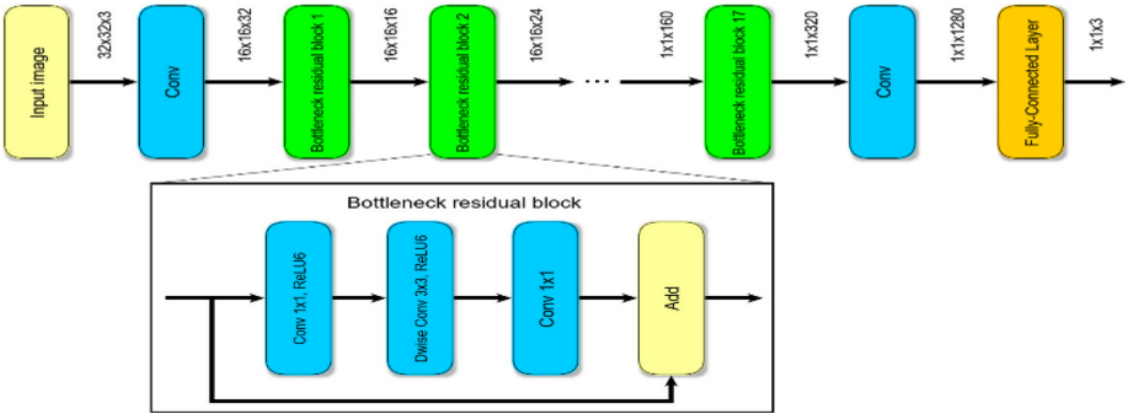

FIGURE 3.15: MobileNetV2 Architecture

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 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. Use the Adam optimizer to compile the final model.
8. Using the ”fit” function, fit the final model on the training dataset .
9. Predict the labels of images from the testing dataset using the final model.
10. Save the final model using the ”save” function from Keras.
11. Visualize the model architecture using the ”plot model” function from Keras.

3.3 Transfer learning with SpinalNet fully connected layer based System

3.3.1 Idea of proposed model

We have finally implemented the CNN and MobileNetV2 based models and found that accuracy of the model was nearly around 88 percent. Based on the readings in the research paper 5, [**?**], our final system will be on **SpinalNet Transfer Learning** based architecture. According to the literature review, out of all architectures implemented, The SpinalNet architecture, inspired by the chordate nervous system, replaces the traditional fully connected layer. It introduces a sequence of layers, each responsible for local decision-making and gradual processing of input, promoting better feature extraction and representation. The paper also predicts very high accuracy rate and hence we try to implement the following model.

3.3.2 Methodology

Preprocessing images with augmentation and resizing, organizing datasets, and applying the VGG-19 architecture enhanced with a SpinalNet fully connected layer are all part of the methodology. Stochastic gradient descent is used in training along with a learning rate scheduler to track accuracy and loss. Decision-making and feature extraction are improved by the

SpinalNet architecture. The model is tested for generalizability optionally. Fully connected layers (SpinalNet, ResNet, or VGG) can be fine-tuned in a flexible manner thanks to the code. Metrics for training and validation are visualized as part of the evaluation. Important hyperparameters are fine-tuned during training, which adds to a flexible and all-encompassing method for image classification with the goal of improved accuracy.

The steps carried out in proposal are as follows :

3.3.2.1 Dataset

We will be using this same dataset that we have used in CNN based model and MobileNetV2 based Model as discussed in Section 3.1.3.1.

3.3.2.2 Preprocessing

Ensuring accurate results in the learning module necessitates preprocessing of input images. This involves tasks such as rotating and resizing the system's input images.

3.3.2.3 Motivation for SpiralNet Architecture

The motivation for SpinalNet arises from challenges faced by traditional Deep Neural Networks (DNNs), known for their state-of-the-art performance but hindered by issues like the vanishing gradient problem and the need for extensive computation. DNNs with large input features often struggle with the size of the first hidden layer, impacting feature propagation. The vanishing gradient problem further complicates training, making it difficult as gradients become negligible near input parameters.

Inspiration is drawn from the **human somatosensory system**, where the spinal cord efficiently processes diverse tactile information. Researchers mimic the functionality of this spinal architecture to address DNN challenges. Existing solutions, such as pooling and ResNet/DenseNet architectures, have limitations, including information loss and computational expense.

By dividing each layer into input, intermediate, and output splits, SpinalNet presents a novel method. Large inputs are handled by this design effectively, guaranteeing gradual input processing. SpinalNet performs better than DenseNet because it maintains connectivity without incurring excessive computational costs. The human spinal cord's capacity to effectively receive and process a variety of bodily sensory signals served as the model for this architecture. Input, intermediate, and output splits are incorporated into each layer of the proposed SpinalNet structure, providing a specialized and effective way to process large inputs and improve neural network performance.

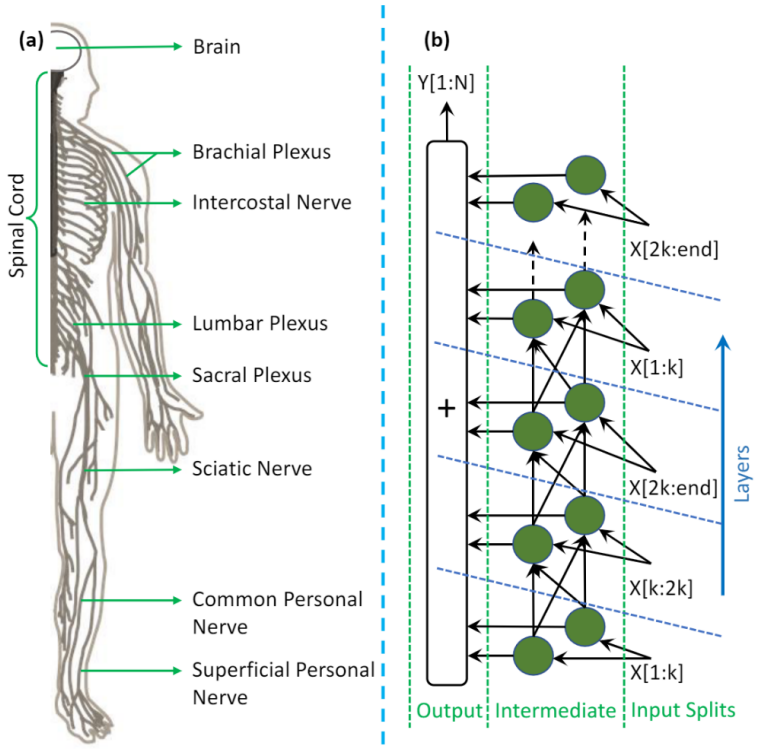

FIGURE 3.16: (a) The mechanism by which our body communicates with our spinal cord to receive and transmit sensory signals. (b) The proposed SpinalNet's design

3.3.2.4 Proposed Structure of SpiralNet using Transfer Learning

In order to emulate three important features of the human nervous system—gradual, voluntary, involuntary movements, and attention to pain intensity—the proposed SpinalNet architecture takes inspiration from the spinal cord. SpinalNet processes information gradually, akin to the varying sensitivity in different parts of the spinal cord, and mirrors the nerve plexus in the human tactile system. The architecture bears resemblance to the human spinal cord.

An output layer, intermediate sublayers, and input sublayers make up SpinalNet's structure. Gradually and repeatedly, the network processes inputs, sending modulated inputs to the global output (brain) and contributing to local outputs (reflex). The tuning of pain sensitivity in human sensory neurons is reflected in this structure. In the intermediate sublayers of the architecture, a nonlinear function is used, and in the output layer, a linear activation function. The proposed SpinalNet's theorem is shown in the following figure.

The visual proof of SpinalNet establishes its universal approximability by contrasting it with a single hidden layer neural network of considerable width. SpinalNet's simplified versions show equivalency to single hidden layer networks with different neuron counts. This proof bolsters the hypothesis that when specific weights are not frozen, deep SpinalNet retains universal approximability and can represent complex relations with fewer neurons.

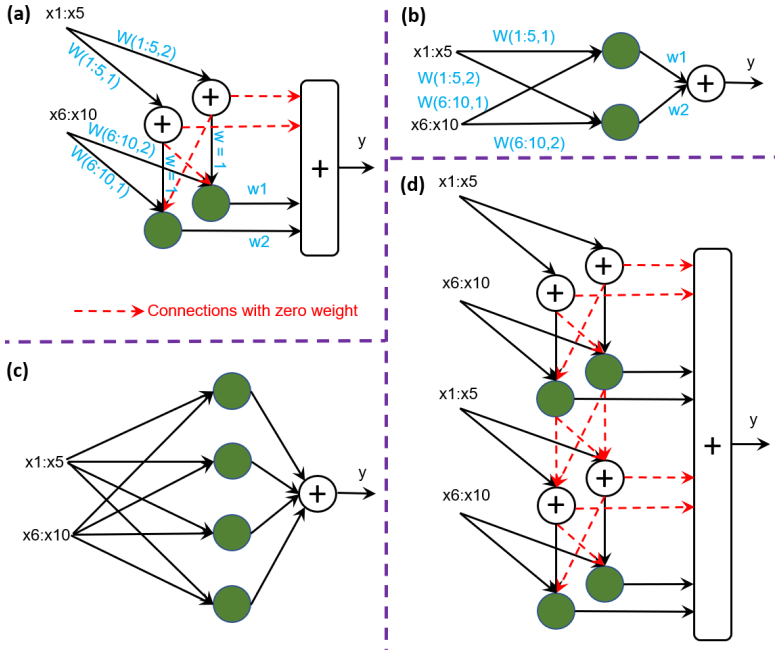

FIGURE 3.17: How SpianlNet works as simple and complex structure is shown in all figures.

A spinal hidden layer can be created from any conventional hidden layer. In (b), the spinal hidden layer replaces the conventional hidden layer found in (a). The proposed SpinalNet's structure is present in a spinal hidden layer.

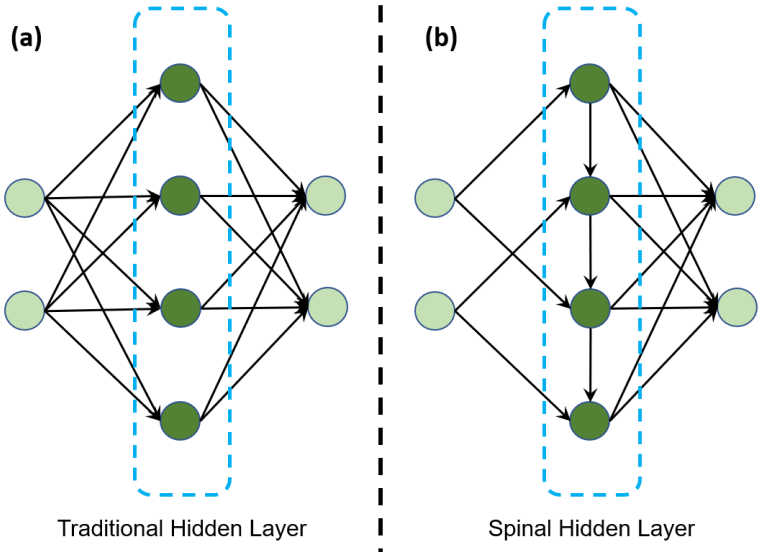

FIGURE 3.18: Transformation to Spinal Layer structure

3.3.2.5 Pretrained Models and Libraries used in architecture.

- **VGG19 bn (Batch Normalized VGG19)**: VGG19 is a deep convolutional neural network architecture. The "bn" suffix indicates the inclusion of batch normalization layers

.It has 19 layers with small (3x3) convolutional filters (16 convolutional and 3 fully connected). By normalizing inputs to each layer, batch normalization is added to enhance training stability. It is frequently utilized as a feature extractor in transfer learning and is renowned for being straightforward and efficient in image classification tasks.

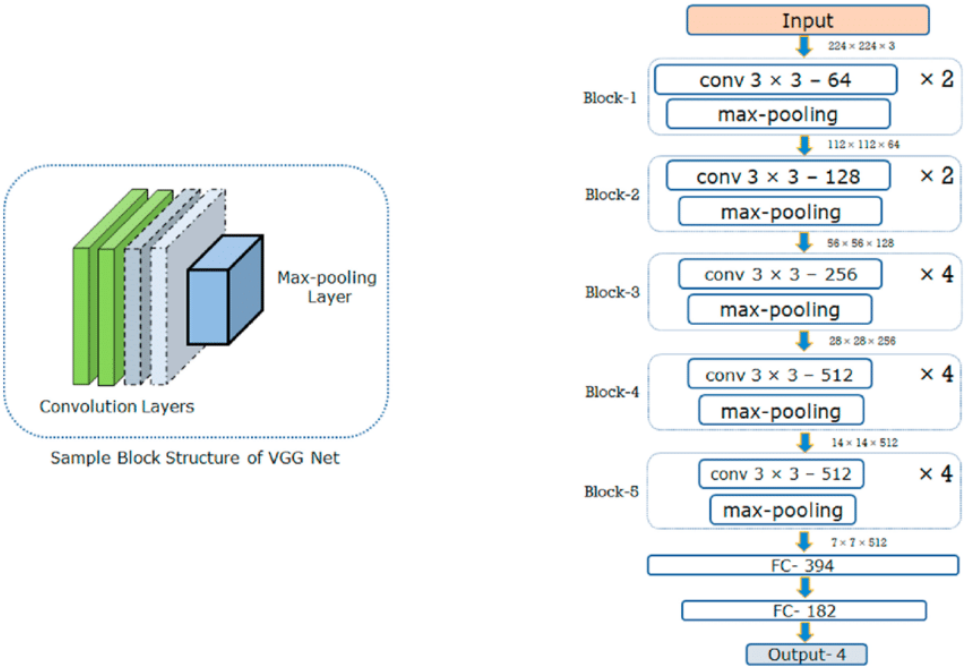

FIGURE 3.19: VGG-19 architecture and internal model.

- **torch (PyTorch):** An machine learning library called PyTorch is frequently used for deep learning applications. It improves its flexibility and ease of use.It provides a large selection of pre-built neural network construction and training functions. It also facilitates automatic differentiation., facilitating the computation of gradients.
- **torchvision**:torchvision is a PyTorch library specifically designed for computer vision tasks. It provides popular datasets (like ImageNet), pre-trained models, and image transformations. It simplifies the process of loading and preprocessing image data for deep learning tasks. It includes implementations of state-of-the-art vision models.

3.3.3 Algorithm of the model based on Transfer learning with SpinalNet fully connected layers.

1. We will import necessary libraries and modules for deep learning and data manipulation. Libraries include torch, torchvision, numpy, matplotlib, time, os, and copy..
2. Define data transformations for the training, validation, and test setsand include resizing, random rotation, random horizontal flip, conversion to tensors, and normalization.

3. We will define the directory containing the image data. and use torchvision's ImageFolder to create datasets for training, validation, and testing. Create data loaders for each dataset, specifying batch size, shuffle, and number of workers.

4. Check for the availability of a GPU (CUDA) and set the device accordingly.

5. We will implement a function imshow to visualize images in a grid.

6. We will load a batch of training data and create a grid of images using make grid from torchvision.

7. We will load the VGG19 model from torchvision and use the number of input features from the first layer of the classifier.

8. We will define two custom neural network architectures (SpinalNetResNet and SpinalNetVGG). Each has multiple fully connected spinal layers with ReLU activations.

9. We will define a classifier using a sequential model consisting of fully connected layers with dropout and replace the classifier of the pre-trained VGG19bn model with the defined SpinalNetResNet classifier.

10. To train the model, define a function train model. Train the model using a scheduler to change the learning rate over a predetermined number of epochs..

11. We will use the defined function to train the model for five epochs after initializing an optimizer with a lr of 0.001 and momentum.

Chapter 4

Simulation and Results

4.1 Results and Analysis of a CNN

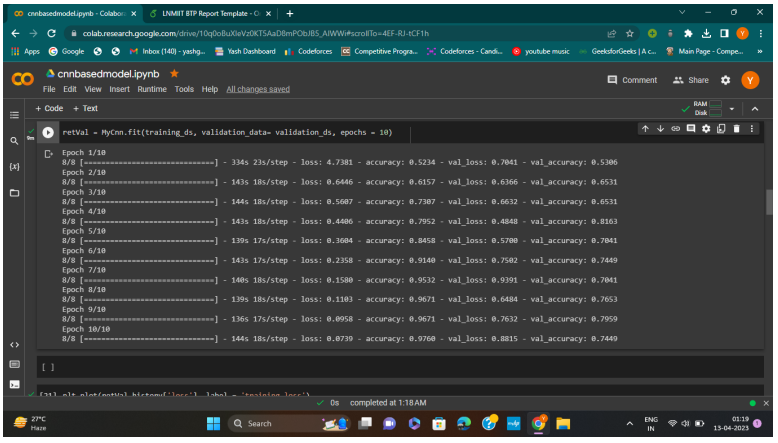

FIGURE 4.1: Training of the CNN-based Model

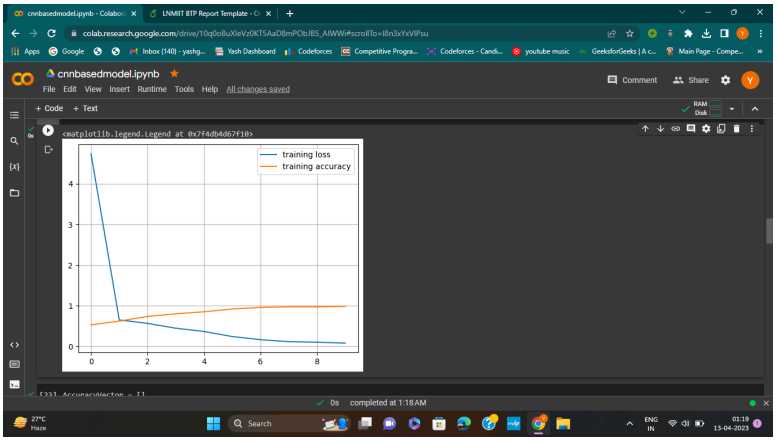

FIGURE 4.2: Graph between Training Loss and Training Accuracy

- 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 training accuracy and loss graph is displayed in the second figure.The model accuracy to the training set of data is indicated by the training loss.
- 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

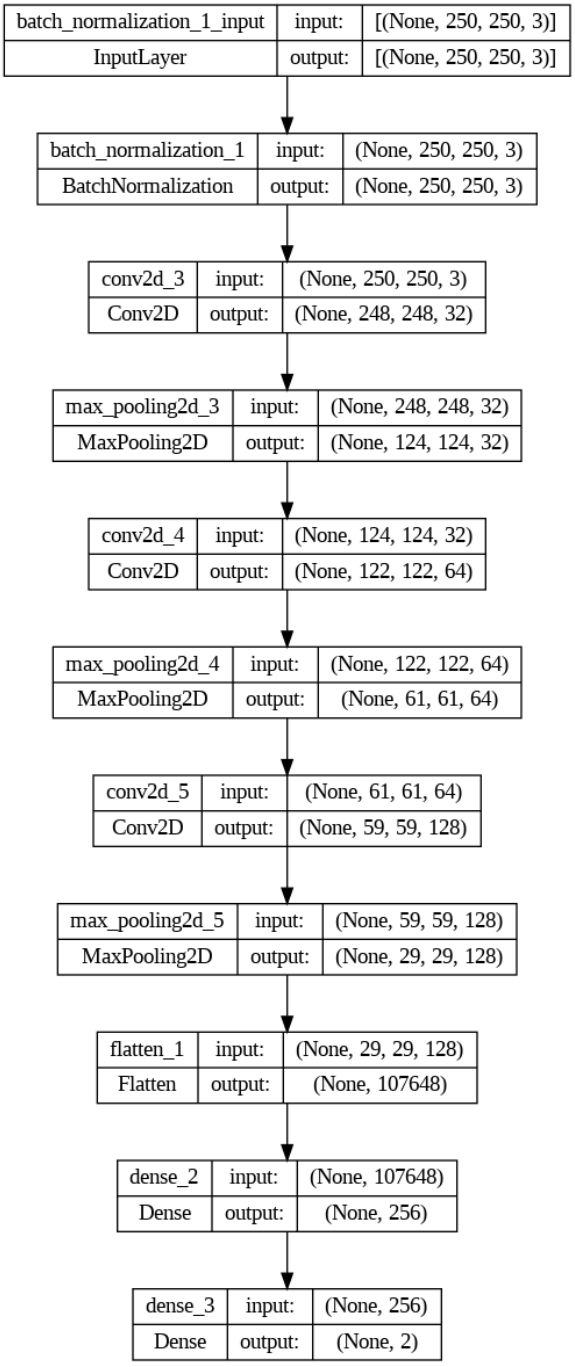

FIGURE 4.5: Architecture of our CNN model

CNNs are computationally efficient, optimizing the processing of visual data, making them ideal for tasks like image recognition and classification.CNNs employ parameter sharing and utilize specialized convolutional and pooling techniques, enabling them to run on various devices globally, enhancing accessibility.CNNs autonomously identify relevant features in data, eliminating the requirement for manual intervention, which streamlines the model training process.

We will now implement another model based on MobileNetV2 based architecture which is based on Tranfer Learning.

4.2 Simulation and Results of MobileNetV2

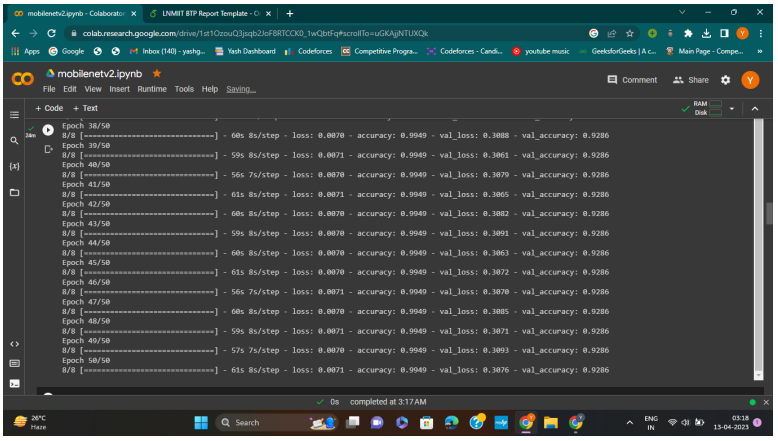

FIGURE 4.6: Training of the MobilenetV2-based Model and its accuracy

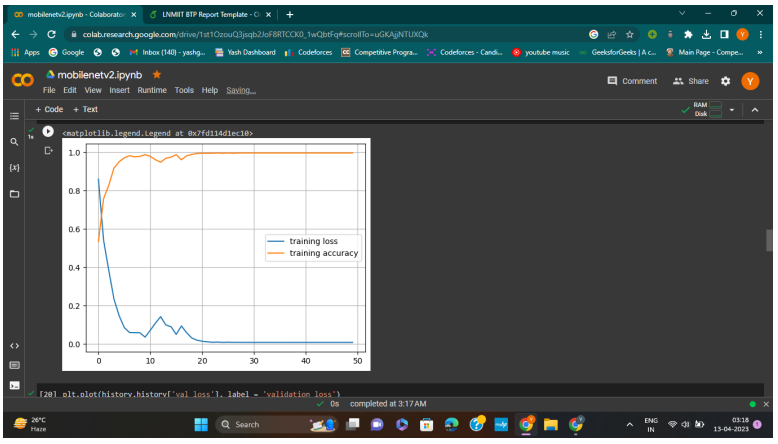

FIGURE 4.7: Graph between Training Loss and Training Accuracy

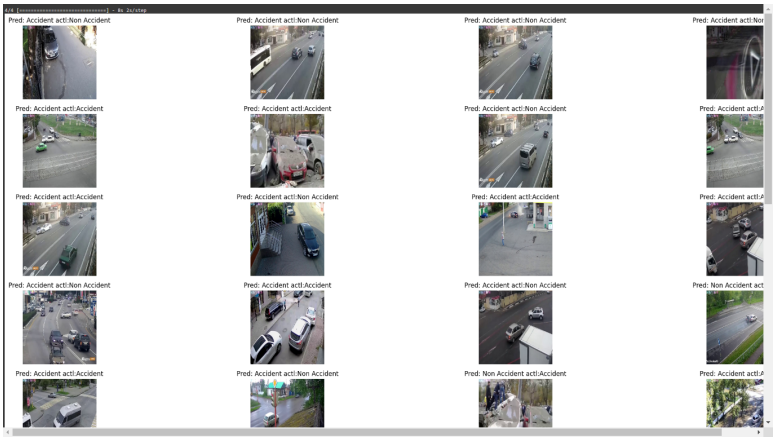

FIGURE 4.8: Visualizing Results on Testing Data

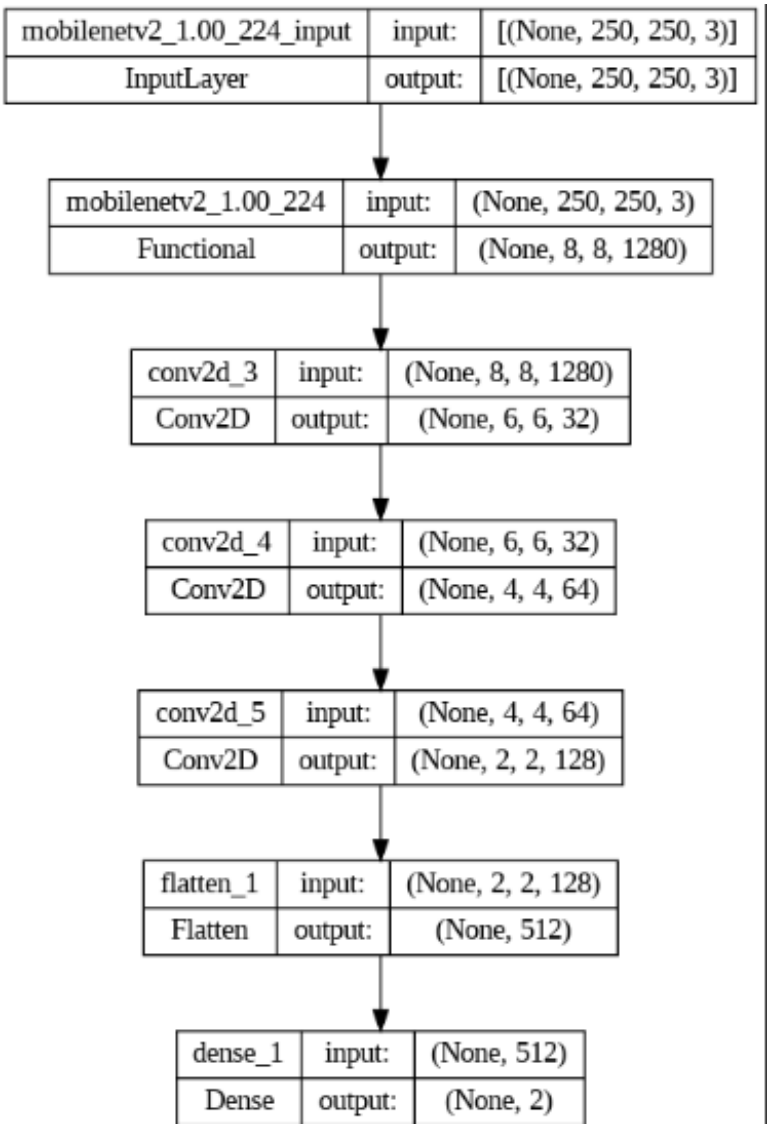

FIGURE 4.9: Architecture of our MobileNetV2 based model

4.3 Simulation and Results of model based on Transfer learning with SpinalNet fully connected layers

- The first figure shows the results of training of our model based on SpinalNet architecture, when the learning ratio is 0.01.
- The second figure shows the results of training of our model,when the learning ratio is 0.001.
- Thr third figure shows visualization of testing data.

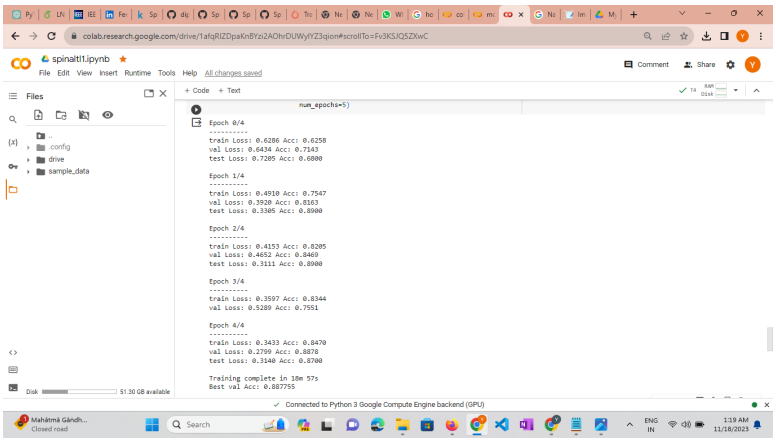

FIGURE 4.10: Proposed model's accuracy at a LR of 0.01

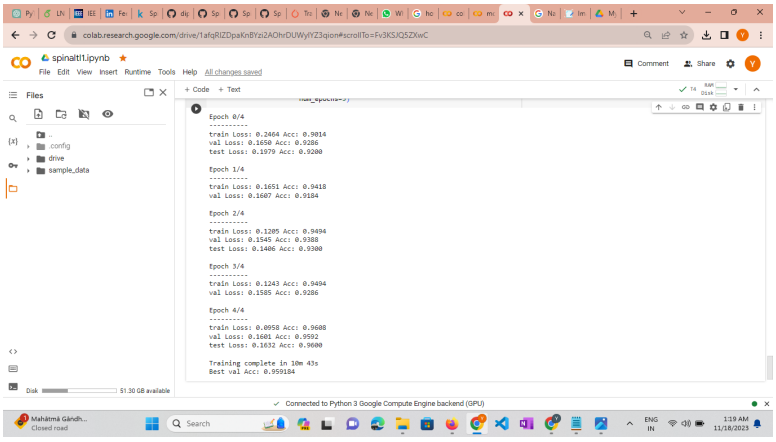

FIGURE 4.11: accuracy at a LR of 0.01

FIGURE 4.12: Visualizing Results on Testing Data

4.4 Comparative Analysis

1. CNN-Based Model (74.9 percent Accuracy):

Strengths: Traditional CNNs excel in capturing hierarchical features from images through convolutional layers. Suitable for complex image recognition tasks due to their ability to learn intricate patterns.

Weaknesses: May struggle with computational efficiency, especially on resource-constrained devices. Limited capacity to optimize performance for diverse datasets.

2. MobileNetV2-Based Model (89.80 percent accuracy):

Strengths: MobileNetV2, designed for mobile and edge devices, focuses on lightweight operations, improving computational efficiency. Efficient depthwise separable convolutions contribute to faster inference times on devices with limited resources.

Weaknesses: MobileNetV2 may sacrifice some accuracy compared to more complex models designed for high-performance environments. Limited in handling highly complex or intricate features in certain datasets.

3. SpinalNetV2-Based Model (95.9184 percent accuracy):

Strengths: SpinalNetV2 introduces spinal layers that capture and process features hierarchically, enhancing the model's ability to learn intricate patterns. Offers improved accuracy by incorporating multiple spinal layers, allowing the model to understand complex relationships in data.

Weaknesses: The increased complexity may result in higher computational demands compared to simpler models. May require more training data to fully exploit the potential of the spinal layers.

Comparasion of Models:

- Accuracy Improvement: The SpinalNetV2 model achieved the highest accuracy (95.9184 percent), surpassing both the CNN and MobileNetV2 models. The spinal layers in SpinalNetV2 contribute to its success by capturing and leveraging hierarchical features effectively.
- MobileNetV2 showed improved efficiency compared to traditional CNNs, making it suitable for deployment on resource-constrained devices. SpinalNetV2, while achieving high accuracy, may have increased computational demands due to its complex architecture.
- Feature Representation: CNNs are adept at learning hierarchical features but may fall short in representing complex relationships. MobileNetV2 focuses on lightweight operations and is efficient but may compromise on capturing intricate patterns. SpinalNetV2, with its spinal layers, excels in capturing both hierarchical and complex features, leading to superior accuracy.

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 our accident detection models, based on CNN, MobileNetV2, and SpinalNetV2, into intelligent traffic management systems empowers traffic controllers to respond promptly to issues and reroute traffic, effectively mitigating congestion. Furthermore, these models offer valuable insights into traffic patterns and accident-prone areas, enabling the formulation of data-driven policies for enhancing infrastructure and road safety. Ultimately, the integration of our accident detection models plays a pivotal role in the development of smart cities, fostering safer and more efficient urban environments.

In Future, we plan to deploy our project through an Android application, ensuring accessibility to a broader audience. Integration with emergency services will be a key aspect, ensuring rapid and efficient responses in emergency situations. This future development aligns with our commitment to leveraging technology for the betterment of urban safety and traffic management.Our focus will extend to implementing above models on extended image sequences, specifically videos. This enhancement will enable our models to operate seamlessly in real-time video feeds, offering a more comprehensive approach to accident detection and traffic management.

Bibliography

[1] S. Ghosh, S. J. Sunny, and R. Roney, "Accident detection using convolutional neural networks," in *2019 International Conference on Data Science and Communication (IconDSC)*, pp. 1–6, 2019.

[2] T. Tamagusko, M. G. Correia, M. A. Huynh, and A. Ferreira, "Deep learning applied to road accident detection with transfer learning and synthetic images," *Transportation Research Procedia*, vol. 64, pp. 90–97, 2022. International Scientific Conference "The Science and Development of Transport - Znanost i razvitak prometa".

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

[4] M. U. Hossain, M. A. Rahman, M. M. Islam, A. Akhter, M. A. Uddin, and B. K. Paul, "Automatic driver distraction detection using deep convolutional neural networks," *Intelligent Systems with Applications*, vol. 14, p. 200075, 2022.

[5] H. D. Kabir, M. Abdar, A. Khosravi, S. M. J. Jalali, A. F. Atiya, S. Nahavandi, and D. Srinivasan, "Spinalnet: Deep neural network with gradual input," *IEEE Transactions on Artificial Intelligence*, 2022.

[6] F. Bhatti, M. A. Shah, C. Maple, and S. U. Islam, "A novel internet of things-enabled accident detection and reporting system for smart city environments," *sensors*, vol. 19, no. 9, p. 2071, 2019.

[7] M. Shahverdy, M. Fathy, R. Berangi, and M. Sabokrou, "Driver behaviour detection using 1d convolutional neural networks," *Electronics Letters*, vol. 57, no. 3, pp. 119–122, 2021.

[8] A. Imre, G. Csaba, L. Ji, A. Orlov, G. Bernstein, and W. Porod, "Majority logic gate for magnetic quantum-dot cellular automata," *Science*, vol. 311, no. 5758, pp. 205–208, 2006.