A Project Report

on

AUTOMATED GENERATION OF CHALLAN BY DETECTING HELMET

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

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

19WH1A05C0	S. S. SPHOORTHY
19WH1A0597	K. CHANDRALEKHA
20WH5A0508	A. SWATHI

under the esteemed guidance of

Dr. R. Dileep Kumar
Assistant Professor

Department of Computer Science and Engineering
BVRIT HYDERABAD
College of Engineering for Women
(NBA Accredited – EEE, ECE, CSE and IT)
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Bachupally, Hyderabad – 500090

June, 2023

DECLARATION

We hereby declare that the work presented in this project entitled “AUTOMATED GENERATION OF CHALLAN BY DETECTING HELMET” submitted towards completion of Project Work in IV year of B.Tech., CSE at ‘BVRIT HYDERABAD College of Engineering for Women, Hyderabad is an authentic record of our original work carried out under the guidance of Dr. R. Dileep Kumar, Assistant Professor, Department of CSE.

S. S. Sphoorthy
(19WH1A05C0)

K. Chandralekha
(19WH1A0597)

A. Swathi
(20WH5A0508)

BVRIT HYDERABAD
College of Engineering for Women
(NBA Accredited – EEE, ECE, CSE and IT)
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Bachupally, Hyderabad – 500090

Department of Computer Science and Engineering

Certificate

This is to certify that the Project Work report on "AUTOMATED GENERATION OF CHALLAN BY DETECTING HELMET" is a bonafide work carried out by S. S. Sphoorthy (19WH1A05C0), K. Chandralekha (19WH1A0597), A. Swathi (20WH5A0508) in the partial fulfillment for the award of B.Tech. degree in **Computer Science and Engineering, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Head of the Department
Dr. E. Venkateswara Reddy
Professor and HoD,
Department of CSE

Guide
Dr. R. Dileep Kumar
Assistant Professor

External Examiner

ACKNOWLEDGEMENTS

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal**, **BVRIT HYDERABAD College of Engineering for Women**, for providing the working facilities in the college.

Our sincere thanks and gratitude to our **Dr. E. Venkateswara Reddy, Professor & HOD**, Department of CSE, **BVRIT HYDERABAD College of Engineering for Women** for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Dr. R. Dileep Kumar, Assistant Professor**, Department of CSE**, BVRIT HYDERABAD College of Engineering for Women** for his constant guidance, encouragement, and moral support throughout the project.

Finally, we would also like to thank our Project Coordinator, all the faculty and staff of **CSE** Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

S. S. Sphoorthy
(19WH1A05C0)

K. Chandralekha
(19WH1A0597)

A. Swathi
(20WH5A0508)

CONTENTS

ABSTRACT

The "Automated Generation of Challan by Detecting Helmet" is a proposed approach to address the prevalent issue of traffic law violations, particularly in densely populated countries like India. The system aims to reduce accidents and losses to life by focusing on the improper use of motorcycle helmets, which is a common cause of mishaps involving bikes.

The primary objective of the system is to enhance safety on the roads by decreasing the risk of injuries in accidents. By accurately identifying motorcyclists without helmets, the system aims to increase compliance with traffic regulations. This approach also offers benefits in terms of reducing the workload on the traffic control team by automating the process and efficiently sharing evidence for imposing fines on violators.

The proposed system utilizes deep learning techniques to detect violations. The process begins with the identification of motorcycle riders through background subtraction and object segmentation. Then, an object classifier is employed to classify violators based on their action which is not wearing helmets.

LIST OF FIGURES

1. INTRODUCTION

In countries like India, two-wheelers account for a significant number of road accidents. Not wearing a helmet significantly increases the risk of death, making helmet usage crucial for rider safety. Recognizing the importance of helmets, the government has made it compulsory to wear them while riding motorcycles. However, many people still violate this rule. To address this issue, an automated system is proposed that utilizes neural networks to generate challans (traffic violation tickets) for riders without helmets, allowing traffic police to focus on other violations.

Efficient traffic violation detection and tracking pose significant challenges. The existing system involves a substantial amount of manual work, leading to performance issues and increased errors over time. This burdens traffic police officers who must manually identify offending vehicles and contact the Regional Transport Office (RTO) to obtain vehicle data.

To overcome these challenges, the proposed system leverages neural networks to automate challan generation, facilitating the fine imposition process for helmetless riders. This automation reduces the manual workload on traffic police and enables them to concentrate on addressing other violations effectively.

The system places emphasis on the importance of helmet usage while driving. It not only detects riders without helmets but also extracts license plate information from surveillance footage. Deep learning algorithms are employed to recognize and extract characters from license plates, allowing for accurate issuance of traffic violation tickets.

With the increasing number of vehicles on the roads, automation is becoming increasingly necessary. Traffic police officers face immense challenges in efficiently monitoring and controlling violations across numerous roads. By automating the process of detecting helmetless riders and generating challans, the system aims to reduce violations and enhance road safety. This, in turn, contributes to creating a better and safer environment for both pedestrians and drivers.

The demand for transparent, quick, and hassle-free traffic violations management systems with electronic challan support has risen in countries with a large number of vehicles. The proposed system addresses this need by streamlining the detection and issuance of challans. By implementing neural networks and automated processes, the system offers a more efficient and accurate approach to identifying violators.

Automating the detection and tracking of traffic violations is a significant step towards ensuring road safety. It not only helps enforce helmet usage but also relieves traffic police officers from burdensome manual tasks. By automating the generation of challans for helmetless riders, the system allows police officers to focus on addressing other critical violations such as illegal parking, driving on the wrong side, or drunk driving.

In conclusion, the proposed system utilizes neural networks and camera surveillance to automate the detection of helmetless riders and generate challans. By simplifying the enforcement process, the system reduces the workload on traffic police officers and ensures more effective monitoring of traffic violations. This ultimately contributes to improved road safety and a better environment for all road users.

1.1 OBJECTIVE

The main aim is to design an automated system that generates challan by identifying the users without helmets through extracting the number plate of the vehicle.

1.2 PROBLEM SPECIFICATION

This model is developed here is used for traffic management. This automated system uses neural network algorithm for detecting the presence or absence of helmet. So this model helps to charge automatic challan to the violators for violating the rule of not wearing a helmet. The problem specification for this game is given below:

Problem:

Automatic Challan Generation using MobileNet SSD model which is a CNN technique for detecting if a person is wearing helmet or not.

1.3 EXISTING SYSTEM

In the existing system, traffic police takes a photo of the licence plate of the offenders who violate the traffic rules. Then E-challan is generated and sent to the associated challan website using the vehicle information retrieved from the concerned authorities. The authors in the literature used machine learning to recognize the vehicle number in the considered image to achieve the proposed task. The problem with the existing system is that, it generates challan manually where traffic authority captures the images of number plates. This system is not efficient since it is both time and effort consuming.

1.4 PROPOSED SYSTEM

The proposed system presents automated generation of challan for ease of use and to improve efficiency in processing. The system considers image processing in an automated way which serves for object detection. The objective of this project is to design an efficient deep learning mechanism that utilizes SSD MobileNet to process images and accurately identify objects extracted from the image processing. This prototype makes it easy for traffic authorities to reduce human error that exists in the current system.

2. LITERATURE REVIEW

Title: Automated Generation of Challan on Violation of Traffic Rules using Machine Learning

Authors: Shubham Kumar Chandravanshi, Hirva Bhagat, Manan Darji, Himani Trivedi

Summary:

This paper focuses upon automated generation of e-challan using machine learning approach. This model first detects the vehicle number plate then divides the number plate into segments using character segmentation following character recognition. The vehicle number plate is detected using YOLO v3 algorithm which is one of the fastest real-time object detection algorithms. Then after detecting the vehicle number plate the characters are divided into segments using YOLO v3 algorithm. The final step is to recognize the objects and generate the challan using chracter recognition. This character recognition is done using ResNet50 and DenseNet169 modules. This system promotes the era of digitization and will impose less usage of paper thus speeding up the entire process of challan generation. The machine learning models have achieved 93.86% accuracy in number plate detection, 97.38% for character segmentation and 89.11% accuracy in character recognition. This system promotes the era of digitization and will impose less usage of paper thus speeding up the entire process of challan generation. This will also generate huge amounts of data which can be used in various pattern findings.

Title: Online Challan Generation System Based On Machine Learning

Authors: Dr. Sonali Ridhorkar, Khushi Gupta, Kalyani lokhande, Prachi Bhanarkar, Vijeta Meshram, Venuhemane

Summary:

This paper focuses on E-Challan generating system using CNN, KNN (K-Nearest Neighbour). There are three major phases followed by the author. First Identifies the Violator, then detects the number Plate and finally generates the E-Challan. Identify the Violator: First detect the number plate of the violator using OCR (Optical Character Recognition). Number Plate Detection: Second extract the license number on the detected number plate, after extracting the numbers search for details of the user in the database. Generate E-Challan: Next is to check the violation details as well as generate e-challan using OCR(Optical Character Recognition) with all the information like violators name, date, time, kind of violation, with mobile number. And at the end send the e-challan on the registered mobile number of the violator as well as on the registered email. This paper emphasizes the suggested system above is to develop an automatic e-challan generation system with accurate results and functionality. The system is using python which is a great advantage as it can be elaborated in future with other features like security measures, sending help in emergency just after the detection and many more.

Title: Automated E-Challan System

Authors: Shubham Kukde , Sakshi Lokhande, Santosh Mishra, Pranoti Mahalle

Summary:

In this paper our proposed scheme works to convert the traditional challan system, our project is to prepare E-challan system so that whenever a traffic cop clicks photo of offender's vehicle it is uploaded to the server, further the traffic cop creates the E-challan with the vehicle number and then the created E-challan is made available to the motorist on the consumer side of the system, also this is notified to the motorist via SMS. When-ever any vehicle breaks any of the traffic rule, the traffic cop will click the photo of the offender's vehicle, with the fine type and whatever the rule is break by offender the cop will select the rule by giving the drop down in the application and select the fine type and this will be selected in RTO database.

Title: Smart Vehicle Recognition And E-Challan Generation System

Authors: Srinath, Raghunandan, Jayavrinda Vrindavanam, Y. R. Sumukh, L. Yashaswini, and Sangeetha S. Chegaraddi

Summary:

This system is an efficient challan generating system that helps to avoid any fraud situations by using vehicle identification number. This system uses chassis number (VIN) to generate E-Challan. First it detects the chassis number / VIN (vehicle identification number) which is unique to all the vehicles, since number plate can be easily tampered, here chassis number is detected and compared with database to get details of the owner. Then it generates the challan. Generation of challan; it generates challan using OCR and sent to offender along with the snapshot. The process as explained has been demonstrated through the chassis scan of a two-wheeler. Once the offence has been observed, the traffic official scans the aadhar card of the driver and the chassis number by using the device. This system is very time and effort consuming since it uses manual scanning.

Title: Smart Vehicle Number Plate Scanning System using Optical Character Recognition Strategy

Authors: Rajkumar, M., I. Chandra, S. Jai Ganesh, K. Dhivya, and K. Sangeethalakshmi.

Summary:

This proposed system is proved to be highly efficient when identifying the number plate characters and can be used at toll booths to verify vehicles. But first the vehicle must be registered. There are four phases followed by the author. First, the vehicle is registered with its details, using Vehicle Pattern Authorization module. This module allows the user / owner of vehicle to register details of the vehicle. Next the authority needs to check the details entered by the user are valid or not. This check is done by using Authority Registration and Scam Vehicle Block Port module. This module is used to take details of the authority that checks the violations. Number Plate Scanning Port: It scans and traces the vehicle when it crosses toll booth and if vehicle is registered then it allows the vehicle to proceed, else blocks it. The next step is to alert the system whenever a violation takes places. This is done using Gateway Alert Mechanism module. Car number plate scanning system's primary goal in applying the suggested INC approach, as well as other aims, such as resolving algorithmic and mathematical elements of computerized number plate identification systems, is to identify and fix these issues. This module uses OCR to extract number from License Plate. It also sends alert SMS to authorities. This system is used only at toll gates.

Title: Recognition of Moving Vehicle Number Plates using Convolutional Neural Network and Support Vector Machine Techniques
Authors: Fernandes, Roshan, K. Madhu Rai, Anisha P. Rodrigues, B. A. Mohan, N. Sreenivasa, and N. Megha

Summary:
This work detects better quality images of number plate by capturing videos using efficient algorithms. There are four phases followed by the author. First is to capture the video, then convert the video into frames them process these image frames. Finally train the model using CNN and SVM to extract number plate. Video Data Collection: Video of moving vehicle is captured through mobile camera. Conversion of video to audio frames: Here, video captured is converted into images of jpg format. Image preprocessing: Uses histogram equalization technique to preprocess image. Train dataset: Uses CNN and SVM to extract licence plate number by using sequential model and HOG respectively. In this system, the video is captured using a mobile camera and then the video is converted to image frames. Image enhancement is done using the histogram equalization method.

Title Licence Plate Identification and Recognition for Non-Helmeted Motorcyclists using Light-weight Convolutional Neural Netwok
Author: Darji, Meghal, Jaivik Dave, Nadim Asif, Chirag Godawat, Vishal Chudasama, and Kishor Upla

Summary:
This system identifies and recognizes the number of two-wheelers who violate traffic rules by not wearing helmets. This system identifies, extracts and recognizes the license Plate. First license plate is detected using Licence Plate Detection module. This module uses MobileNet based SSD model to detect number plate and places rectangular box over number. Then the license plate is extracted using Licence Plate Extraction module. Here, image is cropped limiting to the number in the box. Finally the OCR based Recognition module identifies characters on the plate and appends them into a text file. The proposed approach consists of three modules named as licence plate detection, extraction and recognization module. To detect the licence plates of the Indian motorcycles, we create a new dataset which consists of 1524 licence plate images of Indian motorcycle.

Title: Cognitive Number Plate Recognition using Machine Learning and Data Visualization Techniques

Author: Rahul Agrawal, Manas Agarwal, Rajalakshmi Krishnamurthi

Summary:

The proposed system proves to be both time saving and cost efficient to identify traffic rule violators using automation that makes it easy in traffic management. There are four phases in this system. The first step is to detect the number plate, then the characters are segmented and recognized. Later visualization is done to identify the maximum number of violations in a particular place. The number plate is detected using Number Plate Detection module. In this module detecting the number of the vehicle from input image. Here, the system converts the image into grey scale image by using Otsu's method. The character segmentation is done using Character Segmentation module. In this module the characters on number plate are mapped out and segmented into individual images. Then character recognition is done by Character Recognition module. Here, the supervised machine learning technique is used to recognize the characters on the number plate. Finally the visualization is done by Visualization module, where data analysis is carried out on dataset that is used for k-means clustering. This research paper provided a summary of significant advances made in this exciting area of research with a focus on number plate detection algorithms, although no such system has yet been made oriented for use in India. K-means clustering is used to find the positive correlation between variables observed.

Title: E-challan Generation Using ANPR with Web Application

Author: Shubham Kumar Sharma, Aakrati Jain, Swapnil Babar, Pushpak Mahajan

Summary:

This system is used to generate e-challan in the name of the person who owns the vehicle and details of challan are also sent to the owner as a text message. This system contains two modules which are Detecting number plate and Generation of challan. The first module includes ANPR (automatic number plate recognition) that includes capturing image of vehicles using localization, character segmentation. Next, generation of challan is done by OCR(optical character recognition) to read the vehicle number plate with the help of installed cameras and CCTV systems. The most important feature of this project is it reduces manual work done by the officials and authorities thereby smoothening the process of checking and verifying the details of vehicle in a time efficient manner to facilitate effective challan management. The challan is generated to the traffic violators using vehicle registration number so detected is looked up in the database for getting the type of vehicle and owner's information.

Title: IOT based E-Challan Automation for RTO using RFID

Author: Siddhant Shivam, Tushar Teotia, Shubham Mishra & Himanshu Mittal

Summary:

This system uses RFID technology to generate challan. This system contains two modules in it. The first module is to identify the violating vehicle and generation of challan. In Identifying vehicle module, a RFID tag will be attached to the vehicle which will have the user's information stored in them. In Generation of challan module, a challan will be imposed automatically and the mobile number stored in the vehicle's RFID tag will receive a challan notification and the challan will also be sent to the RTO's database. This up-routes manual challan and going on corruption since some traffic police don't pay proper amount to government. Use of this technology will become ubiquitous in coming day. Basically an rfid tag will be attached to the vehicle which will have the user's information stored in them. Whenever the vehicle user crosses the traffic signalthe data will be stored and sentto the server.This datawill now be sent to a decision making system which will check whether the vehicle has crossed the signal on green signal or red signalAnd will be one of the greatest contributions to development of 21st century. And RFID technology will open new volumes in the field of security against vehicle stealing. Thus we conclude that in today's scenario automation in traffic system is a necessity to avoid further violation of traffic rules. It also ensures proper collection of penalties enabling a corruption free environment. This system ensures proper collection of penalties enabling a corruption free environment, but it requires sensors attached to the vehicles to use RFID technology.

Title: Automatic challan System using RFID Technology

Author: Manish Kumar, Niranjan Kumar, Mizan Faisal, Nizamuddin, Niranjan Kumar

Summary:

The propose system identify the red signal jumping by the vehicle, generating an automatic e-challan to that vehicle owner along with RTO. This system contains two modules which are to detect the number plate and then generate the challan. In Detecting number plate module, the RFID Reader reads the information like vehicles number and automatically sends a report on the site itself through LCD. In Generation of challan module, the GSM takes message scripts which is stored in microcontroller. The GSM sends this message to that vehicle owner's mobile and RTO office.

Title: Object Detection and e-Challan Generation Systems for Traffic Violation

Author: Fabian Barreto, NeslineD.Almeida, Premraj Nadar, Ravneet Kaur, Sanjana Khairnar

Summary:

In this paper the work done in Automatic Licence Plate detection and e-challan generation systems for the past five years. For object detection, edge detection algorithms are most preferred, especially the Canny edge detection method. This is followed by OCR. e-challan generation is commonly hosted on websites and android applications and notification is sent via SMS.

Title: A Review Paper on Vehicle Number Plate Recognition

Author: Sharmishtha Mohite, Chinmayi Gurav, Vedika Kamble, Rupali Gurav, Neha S. Sakhalkar

Summary:

The system is applicable for entrances of gates in colleges and highly restricted areas. When any vehicle passes by the system the video is captured and then video is converted into images using OpenCV software. There are three modules in this system. In Real time Input module, the input can of an image or a video. Camera is used to take a realtime image or video and goes through some image processing stages. In Extraction module, the images are converted into grey images. In this paper a system captures image of vehicle and identity is verified using Raspberry pi processor that there is no change in the original image and test image. If any unregistered vehicle is detected then system gives alerts to the computer using buzzer alarm sytem. Input to the system is taken as image of vehicle and the output is detected number plate. Here the images level is extracted based on the parameters and the matrix values. In OCR based Recognition, OCR module identifies characters on the plate and appends them into a text file.

Title: Automatic Number Plate Recognition

Author: Vanshika Rai, Deepali Kamthania

Summary: In this paper an attempt has been made to develop an automatic number plate detection and recognition system for Indian vehicles. This system uses a special device called Raspberry Pi to analyze videos and capture images of vehicles' number plates. It then uses advanced techniques to extract the numbers from these plates. This system is helpful for monitoring vehicles at entrances, like college gates, and areas with restricted access. By accurately identifying vehicles through their number plates, the system improves traffic management and security measures. The proposed system first detects the vehicle and then captures the vehicle image. Vehicle number plate region is extracted using the image segmentation and characters are recognized using optical character recognition technique. This proposed system can be used for e-challan surveillance, stolen vehicle detection and many other applications.

3. THEORETICAL ANALYSIS OF THE PROPOSED PROJECT

3.1 REQUIREMENTS GATHERING

3.1.1 Software Requirements

- Operating System: Windows 10
- Programming Language: Python 3.8
- IDE: Visual Studio Code
- Packages/Libraries: Numpy, Tensorflow, PIL, OpenCV, PaddleOCR

3.1.2 Hardware Requirements

- Processor: Intel core i5
- Memory: RAM 8GB

3.2 TECHNOLOGICAL DESCRIPTION

Programming Language:

- Python: Python is a widely used programming language with excellent libraries and frameworks for data analysis, machine learning, and deep learning. It is an interpreted language. Interpreted language means "the source code is not directly translated by the target machine instead a different program like interpreter reads and executes the code". Python usually has less code than the many other languages which makes programmer to develop the software easily. Python has large set of inbuilt libraries, they are most useful for developing new models. Each library has different functions that reduce the work for the programmer. It provides the necessary tools to implement Tensorflow models, PIL, and OpenCV tasks. These tools help process images and train neural network.

Packages/Libraries:

- NumPy: NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements. NumPy is a fundamental library for numerical operations in Python. It offers multidimensional arrays and mathematical functions for efficient computation. These data structures are used for the optimal computations regarding arrays and matrices. It is used to handle and manipulate multi-dimensional arrays of image data.

- Tensorflow: TensorFlow is a popular open-source library for machine learning and deep learning in Python. It provides pre-trained models which can be fine-tuned for helmet detection. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

- OpenCV: OpenCV (Open Source Computer Vision Library) is a powerful library for computer vision tasks. It provides a wide range of functions and algorithms for image and video processing. It is used to preprocess images, apply filters, perform object detection, and track objects. OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It performs basic image operations like resizing, cropping, and rotating, and apply a wide range of image filters and transformations.

- PIL: PIL (Python Imaging Library) is a library for image processing and manipulation in Python. It is used for functions like opening, resizing, cropping, and applying filters to images, which are essential for preprocessing data before feeding it into the model. Python Imaging Library is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux.

- PaddleOCR: PaddleOCR is an open-source library that specializes in Optical Character Recognition (OCR) tasks. It provides a set of pre-trained models for text detection, which can be used to detect text in images. To perform text detection, we need to first create an OCR object using the PaddleOCR API. It provides various models and tools for text detection, recognition, and layout analysis in images and documents.

4. DESIGN

4.1 INTRODUCTION:

The main aim is to design an automated system that generates challan by identifying the users without helmets through extracting the number plate of the vehicle. Manual detection and tracking of traffic violations have been a challenging process for the authorities. This system reduces the work of traffic police officers so that they can focus on other violations.

The existing surveillance based system is not effective as this system involves huge amount of manual work which might affect the performance and increase the errors over long period of time. It reduces the manual work load on traffic police by eliminating the need of manual picking of the offending vehicles and then contacting the RTO to get the data of that particular vehicle.

This project aims to overcome this major issue, a system is proposed using neural networks with an idea of the generating an e-challan which will help the traffic police department to fine the violators without helmets, making it easier for them to focus on other violations.

4.2 ARCHITECTURE:

The architecture for the model using MobileNet SSD involves several components.

Input: The model takes as input video or image frames captured by a camera or obtained from a video source.

Preprocessing: The input frames are preprocessed to prepare them for the object detection process. This typically involves resizing, normalization, and other transformations to ensure compatibility with the MobileNet SSD model.

MobileNet Feature Extraction: The preprocessed frames are passed through the MobileNet architecture, which is a lightweight deep neural network specifically designed for mobile and embedded devices. MobileNet extracts high-level features from the frames, capturing relevant information for object detection.

MultiBox Detection: The feature maps obtained from the MobileNet backbone are used as inputs for the MultiBox detection module. This module applies a set of predefined anchor boxes to the feature maps and predicts the presence and location of objects (in this case, helmets) within these boxes. It also predicts the class label (helmet or non-helmet) for each detected object.

Challan Generation: Once the helmet detections are obtained, the model generates a challan or traffic violation ticket for each rider without a helmet. The specific details and format of the challan may vary based on the requirements of the system or local regulations.

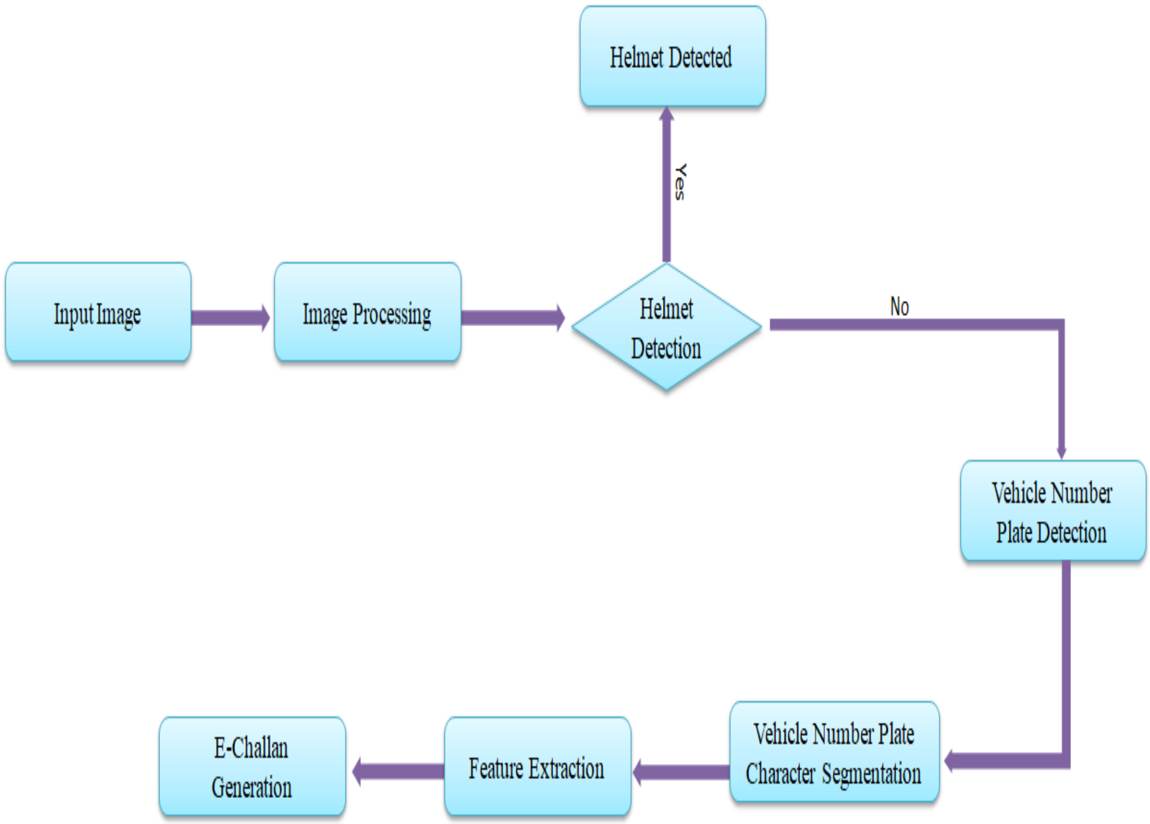

Fig. 1. Architecture

First the model takes an input image from the image source. Then the input image is processed for object detection tasks. These images are transformed into frames for this process. Then these image frames are passed through the MobileNet SSD architecture for object detection. The result obtained from MobileNet SSD is feature maps. These feature maps are used as input to Multibox detection module. It predicts class label for each detected object. The output of the MultiBox detection module may contain multiple overlapping bounding boxes for the same helmet. Suppression techniques are applied to eliminate the redundant classes. Once the helmet detections are obtained, the model generates a challan or traffic violation ticket for each rider without a helmet.

4.3 ALGORITHMS:

4.3.1 MobileNet SSD

MobileNet SSD is a specific implementation of a CNN model that detects object and computes the output bounding box and object class from the input image. Mobilenet SSD is an object detection model that computes the output bounding box and object class from the input image. This Single Shot Detector (SSD) object detection model uses Mobilenet as a backbone and can achieve fast object detection optimized for mobile devices.This Single Shot Detector (SSD) model takes only one shot to detect multiple objects present in an image. The SSD technique is based on a forward convolutional network that generates a collection of fixed-size bounding boxes and a score for the presence of object class instances in those boxes, which come behind by a non-maximal suppression step to create the final detections. Fields contain offset values (cx,cy,w,h) from the default field. The score contains confidence values for the presence of each of the object categories, with a value of 0 reserved for the background.

SSD represents multi-reference and multi-resolution detection techniques. Multi-reference techniques define a set of anchor boxes of different sizes and aspect ratios at different locations in the image and then predict a detection box based on these references. Multi-resolution techniques enable object detection at multiple scales and at different network layers. The SSD network implements an algorithm for detecting multiple object classes in images by generating a confidence score related to the presence of any object category in each default field. It also makes adjustments in the boxes to better fit the shapes of the objects. This network is suitable for real-time applications because it does not resample functions for bounding box hypotheses. This model has 28 convolution layers.

MobileNet uses two types of convolutions: depthwise convolutions and pointwise convolutions. Depthwise convolutions operate on each channel of an input tensor separately. It applies a single filter to each channel and produces a set of output channels that are combined to form the output tensor. Pointwise convolutions are also known as 1x1 convolutions. They operate on the output channels of a depthwise convolution and combine them into a smaller set of channels.

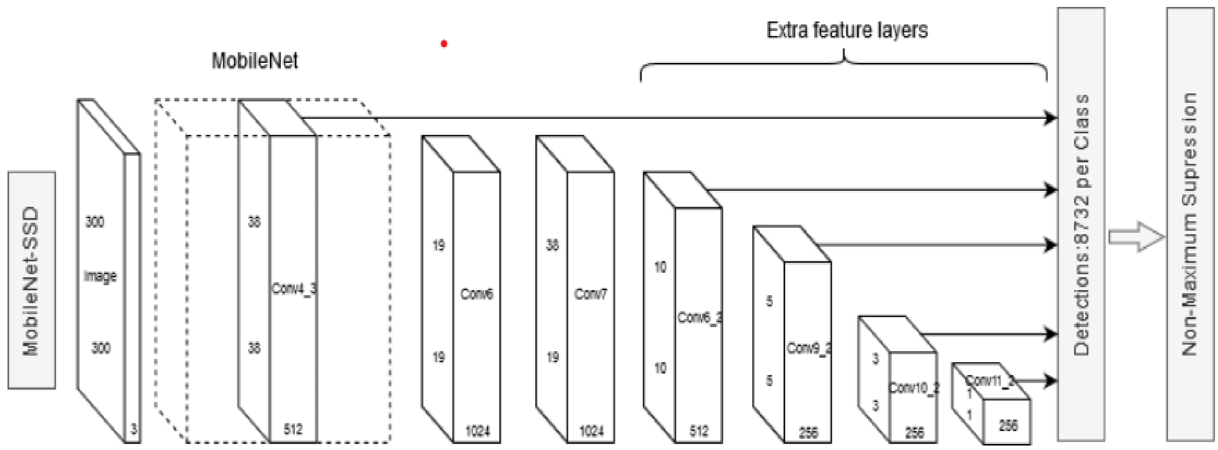

Fig. 2. MobileNet SSD

4.3.2 CNN

Convolutional Neural Network (CNN) is a type of Deep Learning architecture commonly used for image classification and recognition tasks. In deep learning, a convolutional neural network **(**ConvNet**)** is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other. Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

The CNN is another type of neural network that can uncover key information in both time series and image data. For this reason, it is highly valuable for image-related tasks, such as image recognition, object classification and pattern recognition. To identify patterns within an image, a CNN leverages principles from linear algebra, such as matrix multiplication. CNNs can also classify audio and signal data. It consists of multiple layers, including Convolutional layers, Pooling layers, and fully connected layers. The Convolutional layer applies filters to the input image to extract features, the Pooling layer downsamples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

Components of CNN:

Input layer: It is the input given to our model. The number of neurons in this layer is equal to the total number of features in our data.

Convolutional layer: It is the main building block of a CNN. It contains a set of filters, parameters of which are to be learned throughout the training.

Fully connected layer: Fully Connected Layer feed forward neural networks. The input to the fully connected layer is the output from the Convolutional Layer, which is flattened and then fed into the fully connected layer.

Output layer: The output layer converts the output of each class into probability score for its respective classes.

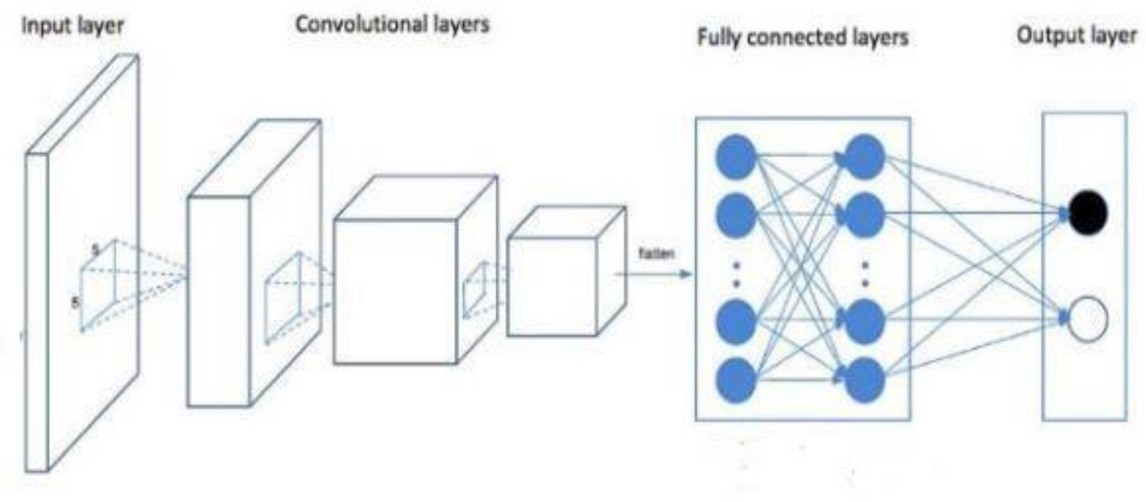

Fig. 3. CNN

4.3.3 OCR

Optical character recognition is a technology that converts typed or handwritten text and printed images containing text into machine-readable digital data format. OCR algorithms help turn large amounts of paper documents into digital files, facilitating text storage, processing, and searching. Optical Character Recognition, or OCR, gives us various options for viewing, finding, and recognizing text in images and labels. When we think of optical character recognition, we automatically think of a lot of paper.

From confidential personal documents to legal documents, not only do they take up a lot of space, but they can cause you problems if lost. This is where OCR comes in and acts as an essential part of document digitization. OCR Machine Learning is a group of computer vision problems where handwritten or typewritten text from a digital image is processed into a text readable by machines.

It is hard for computers to recognize characters because of the different fonts and variations on how one letter can be written. OCR solution operates in four main steps:

- Image acquisition
- Pre-processing
- Noise and rotation correction
- Text detection

Optical Character Recognition (OCR) is the process that converts an image of text into a machine-readable text format. For example, if you scan a form or a receipt, your computer saves the scan as an image file. You cannot use a text editor to edit, search, or count the words in the image file. However, you can use OCR to convert the image into a text document with its contents stored as text data.

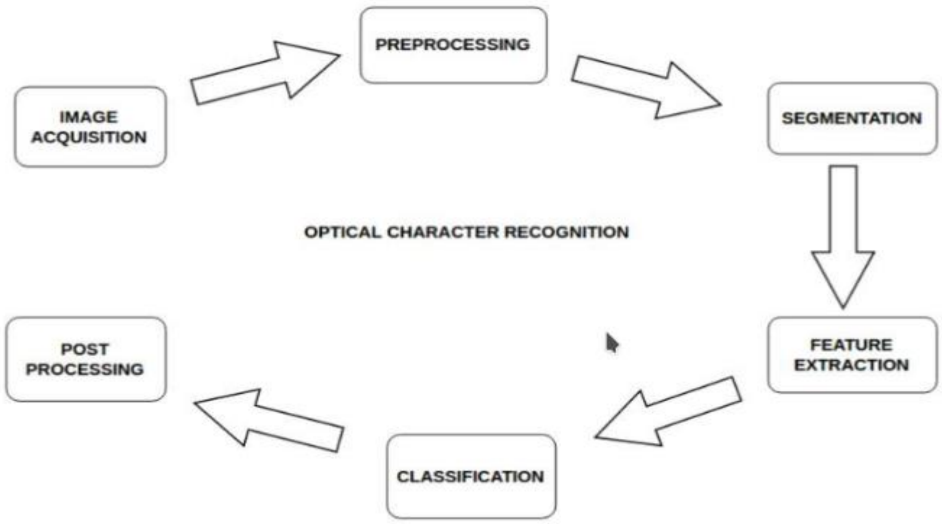

Fig. 4. OCR

4.3.4 PaddleOCR

The PaddleOCR architecture is a system that can recognize and understand text in images. It is used in various applications, such as analyzing documents or extracting information from images.It has different components that work together to perform this task. The first component is text detection, which helps find the areas in an image that contain text. Then the system moves on to text recognition. It is an ocr framework or toolkit which provides multilingual practical OCR tools that help the users to apply and train different models in a few lines of code. It uses deep learning models to read and understand the text, converting it into editable and searchable formats.

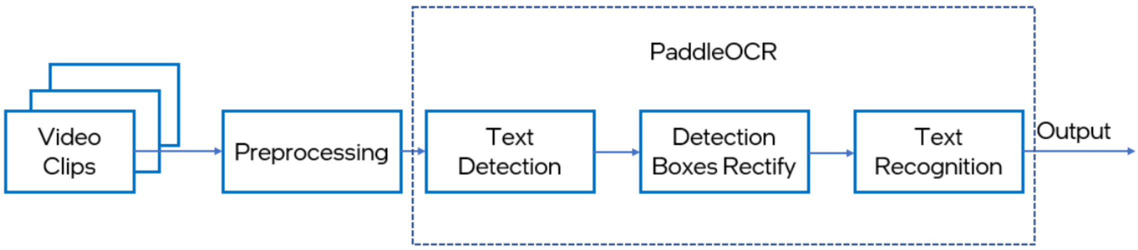

Fig. 5. PaddleOCR

5. IMPLEMENTATION

5.1 DATASET :

Dataset size: 410 MB

Dataset type: 764 Images

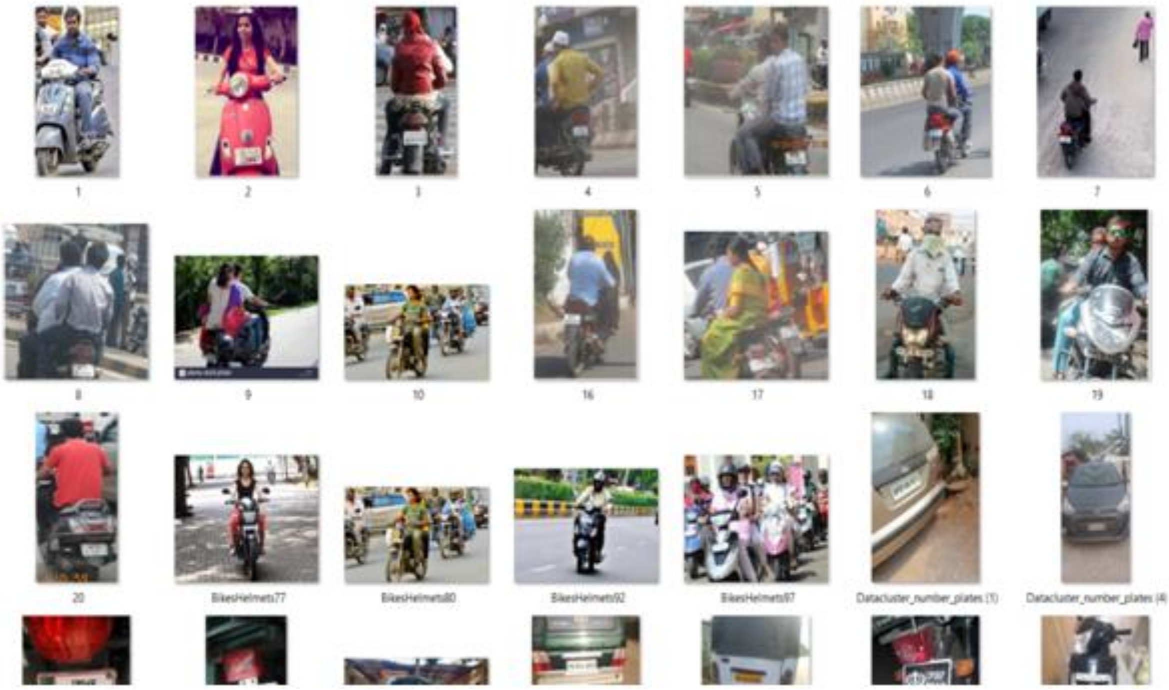

Fig. 6. Sample Dataset

5.2 RESULTS:

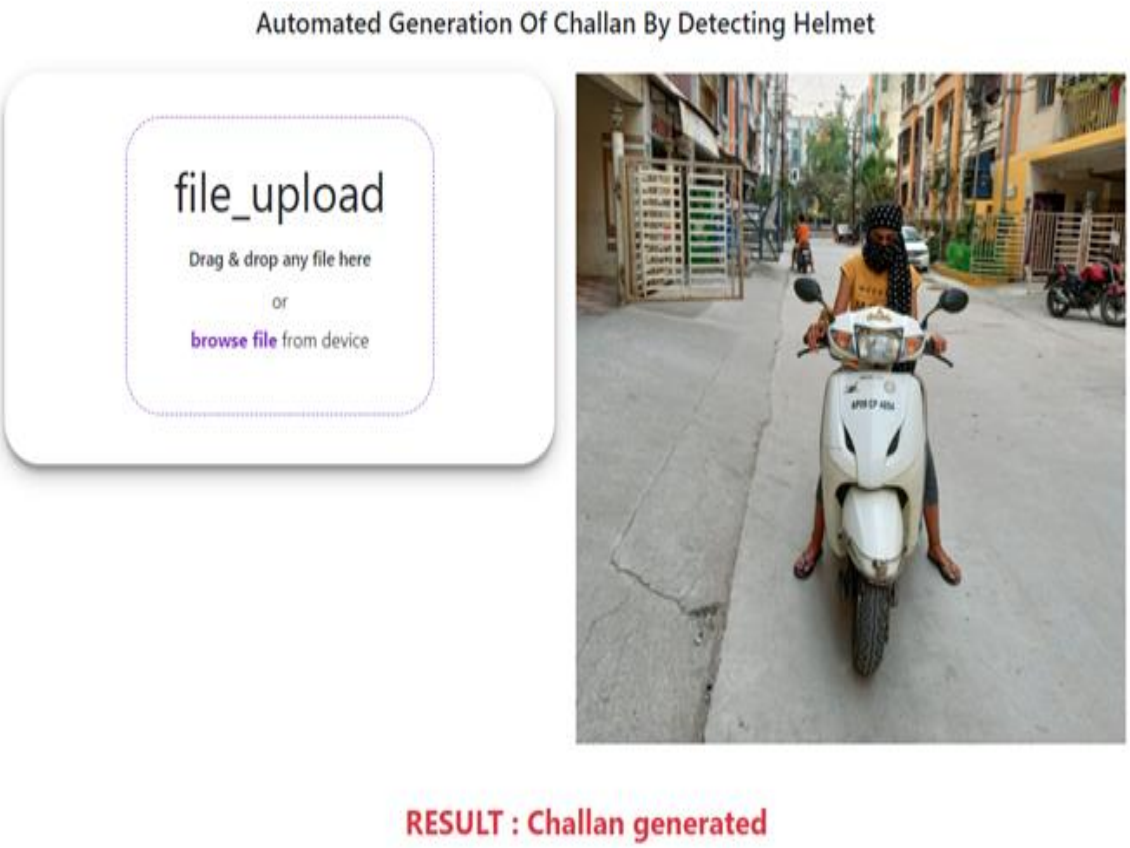

Fig. 7. Helmet not Detected

Fig. 8. Helmet Detected

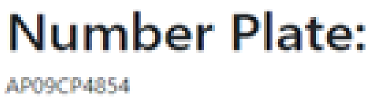

Fig. 9. License Plate Number Detection

6. CONCLUSION AND FUTURE SCOPE

The helmet detection and challan generation model using MobileNet SSD (Single Shot MultiBox Detector) is a promising solution for improving road safety and enforcing helmet usage among two-wheeler riders. The system offers several advantages, including automated detection, which reduces the burden on traffic control teams and enables efficient monitoring of violations. Furthermore, the challan generation feature automates the process of issuing traffic violation tickets, eliminating the need for manual intervention. This streamlines the overall system and enhances the speed and accuracy of penalizing violators. The use of MobileNet SSD as the underlying architecture brings efficiency and real-time performance to the model.

In conclusion, this model using MobileNet SSD provides an automated and efficient approach to enhance road safety, enforce helmet usage, and reduce traffic violations. Further, the task of real time image processing for poor quality images can be carried out for this system as future scope.

7. SOCIETAL IMPACT

In countries with a large population like India, road accidents are a significant concern. Riders without helmets are 2.5 times more likely to die in an accident than those wearing helmets. To address this issue, a proposed system can be used to quickly detect drivers not wearing helmets and issue challans.

This system can also reduce the workload of traffic police officers, allowing them to focus on other violations such as illegal parking, driving on the wrong side, and drunk driving. This prototype makes it easy for traffic authorities to reduce human error that exists in the current system.

8. REFERENCES

[1] Shubham Kumar Chandravanshi, Hirva Bhagat, Manan Darji, Himani Trivedi, ”Automated Generation of Challan on Violation of Traffic Rules using Machine Learning”, Volume 10 Issue 3, March 2021.

[2] Dr. Sonali Ridhorkar, Khushi Gupta, Kalyani lokhande, Prachi Bhanarkar, Vijeta Meshram, Venuhemane, ”Online Challan Generation System Based On Machine Learning”, Vol. 7, No. 8, 2022.

[3] Srinath, Raghunandan, Jayavrinda Vrindavanam, Y. R. Sumukh, L. Yashaswini, and Sangeetha S. Chegaraddi. "Smart Vehicle Recognition And E-Challan Generation System." In 2020 International Conference for Emerging Technology (INCET), pp. 1-4. IEEE, 2020.

[4] Rajkumar, M., I. Chandra, S. Jai Ganesh, K. Dhivya, and K. Sangeethalakshmi. "Smart Vehicle Number Plate Scanning System using Optical Character Recognition Strategy." In 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), pp. 1328-1334. IEEE, 2022.

[5] Fernandes, Roshan, K. Madhu Rai, Anisha P. Rodrigues, B. A. Mohan, N. Sreenivasa, and N. Megha. "Recognition of Moving Vehicle Number Plates using Convolutional Neural Network and Support Vector Machine Techniques." In 2021 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), pp. 93-98. IEEE, 2021.

[6] Darji, Meghal, Jaivik Dave, Nadim Asif, Chirag Godawat, Vishal Chudasama, and Kishor Upla. "Licence Plate Identification and Recognition for Non-Helmeted Motorcyclists using Light-weight Convolution Neural Network." In 2020 International Conference for Emerging Technology (INCET), pp. 1-6. IEEE, 2020.

[7] Shubham Kukde , Sakshi Lokhande , Santosh Mishra , Pranoti Mahalle , Prof. Kiran "Automated E-Challan System"

[8] Rahul Agrawal; Manas Agarwal; Rajalakshmi Krishnamurthi "Cognitive Number Plate Recognition using Machine Learning and Data Visualization Techniques"

[9] Shubham Kumar Sharma, Aakrati Jain, Swapnil Babar, Pushpak Mahajan "E-challan Generation Using ANPR with Web Application"

[10] Siddhant Shivam, Tushar Teotia, Shubham Mishra & Himanshu Mittal "IOT based E-Challan Automation for RTO using RFID"

[11] Manish Kumar, Niranjan Kumar, Mizan Faisal, Nizamuddin, Niranjan Kumar "Automatic challan System using RFID Technology"

[12] Fabian Barreto1, NeslineD.Almeida2, Premraj Nadar3, Ravneet Kaur4*, Sanjana Khairnar "Object Detection and e-Challan Generation Systems for Traffic Violation"

[13] Chinmayi Gurav, Vedika Kamble, Rupali Gurav, . Neha S. Sakhalkar, Sharmishtha Mohite "Vehicle Number Plate Recognition"

[14] Vanshika Rai, Deepali Kamthania "Automatic Number Plate Recognition"

9. APPENDIX

```
import base64
import tensorflow as tf
import cv2
import time
import label_map_util
import visualization_utils as viz_utils
import numpy as np
import pybase64

PATH_TO_LABELS = 'label_map.pbtxt'
MIN_CONF_THRESH = float(0.20)
PATH_TO_SAVED_MODEL = "saved_model"
detect_fn = tf.saved_model.load(PATH_TO_SAVED_MODEL)
category_index =
label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS,
                                                   use_display_name=True)

def mobilenet(image):
    image_expanded = np.expand_dims(image, axis=0)
    input_tensor = tf.convert_to_tensor(image)
    input_tensor = input_tensor[tf.newaxis, ...]

    detections = detect_fn(input_tensor)
    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}
    detections['num_detections'] = num_detections
    detections['detection_classes'] = detections['detection_classes'].astype(np.int64)
    image_with_detections = image.copy()
```

```
    x=viz_utils.visualize_boxes_and_labels_on_image_array(image_with_detections,
                                    detections['detection_boxes'],
                                    detections['detection_classes'],
                                    detections['detection_scores'],
                                    category_index,
                                    use_normalized_coordinates=True,
                                    max_boxes_to_draw=200,
                                    min_score_thresh=0.6,
                                    agnostic_mode=False)


    cv2.imwrite('static/output.jpg', image_with_detections)

    with open("static/output.jpg", "rb") as img_file:
        my_string = base64.b64encode(img_file.read()).decode("utf-8")
    return my_string, x

def Decode(image):
    imgdata = pybase64.b64decode(image)
    image1 = np.asarray(bytearray(imgdata), dtype="uint8")
    image1 = cv2.imdecode(image1, cv2.IMREAD_COLOR)
return image1

from detection import mobilenet, Decode
import cv2
from PIL import Image
import paddleocr
import base64
import numpy as np
from io import BytesIO
from paddleocr import PaddleOCR, draw_ocr
import re
from flask import Flask, request, jsonify
ocr = PaddleOCR()
```

```
app = Flask(__name__)

@app.route('/',methods=['GET'])
@cross_origin()
def index():
    return render_template('index.html')

@app.route('/result',methods=['GET','POST'])
@cross_origin()
def result():
    if request.method == 'POST':
        image_data = request.json['image']

        decoded_data = base64.b64decode(image_data)
        img_file = open('./num_image.jpeg', 'wb')
        img_file.write(decoded_data)
        img_file.close()
        result = ocr.ocr("./num_image.jpeg")
        for idx in range(len(result)):
            res = result[idx]
            data = []
            for line in res:
                x = " ".join(re.split(r'[A-Z]{2}[0-9]{2}|[A-Z]{2}[0-9]{2}',(line[1][0]).strip()))
                y = " ".join(re.split(r'([A-Z]{2} [0-9]{4}|[0-9]{4})',(line[1][0]).strip()))
                if x!=None or y!=None:
                    data.append(x)
                    data.append(y)

        img = Decode(image_data).copy()
        img = cv2.resize(img, (480, 320),interpolation=cv2.INTER_NEAREST)

        print("result numberplate :",data)
        new_data=[]
        for string in data:
```

```
            new_string= string.replace(" ","")
            new_data.append(new_string)
        print ("new_string =>",new_string)

        imgdata=mobilenet(img)
        datax = str(imgdata[1])
        return jsonify({"image":str(imgdata[0]),"Number":aaa,"datax":str(imgdata[1][1])})
    return render_template('index.html')

if __name__=="__main__":
    app.run(host="0.0.0.0",port="5000",debug=True)



import collections
import logging
import numpy as np
from six import string_types
from six.moves import range
import tensorflow.compat.v1 as tf
from google.protobuf import text_format
import string_int_label_map_pb2
_LABEL_OFFSET = 1

def _validate_label_map(label_map):

  for item in label_map.item:
    if item.id < 0:
      raise ValueError('Label map ids should be >= 0.')
    if (item.id == 0 and item.name != 'background' and
        item.display_name != 'background'):
      raise ValueError('Label map id 0 is reserved for the background label')
```

```
def create_category_index(categories):
  category_index = {}
  for cat in categories:
    category_index[cat['id']] = cat
  return category_index



def get_max_label_map_index(label_map):
  return max([item.id for item in label_map.item])

def convert_label_map_to_categories(label_map,
                                    max_num_classes,
                                    use_display_name=True):
  categories = []
  list_of_ids_already_added = []
  if not label_map:
    label_id_offset = 1
    for class_id in range(max_num_classes):
      categories.append({
          'id': class_id + label_id_offset,
          'name': 'category_{}'.format(class_id + label_id_offset)
      })
    return categories
  for item in label_map.item:
    if not 0 < item.id <= max_num_classes:
      logging.info(
          'Ignore item %d since it falls outside of requested '
          'label range.', item.id)
      continue
    if use_display_name and item.HasField('display_name'):
      name = item.display_name
    else:
      name = item.name
    if item.id not in list_of_ids_already_added:
```

```
      list_of_ids_already_added.append(item.id)
      category = {'id': item.id, 'name': name}
      if item.HasField('frequency'):
        if item.frequency == string_int_label_map_pb2.LVISFrequency.Value(
            'FREQUENT'):
          category['frequency'] = 'f'
        elif item.frequency == string_int_label_map_pb2.LVISFrequency.Value(
            'COMMON'):
          category['frequency'] = 'c'
        elif item.frequency == string_int_label_map_pb2.LVISFrequency.Value(
            'RARE'):
          category['frequency'] = 'r'
      if item.HasField('instance_count'):
        category['instance_count'] = item.instance_count
      if item.keypoints:
        keypoints = {}
        list_of_keypoint_ids = []
        for kv in item.keypoints:
          if kv.id in list_of_keypoint_ids:
            raise ValueError('Duplicate keypoint ids are not allowed. '
                             'Found {} more than once'.format(kv.id))
          keypoints[kv.label] = kv.id
          list_of_keypoint_ids.append(kv.id)
        category['keypoints'] = keypoints
      categories.append(category)
  return categories


def load_labelmap(path):
  with tf.io.gfile.GFile(path, 'r') as fid:
    label_map_string = fid.read()
    label_map = string_int_label_map_pb2.StringIntLabelMap()
    try:
      text_format.Merge(label_map_string, label_map)
    except text_format.ParseError:
```

```
    label_map.ParseFromString(label_map_string)
  _validate_label_map(label_map)
  return label_map

def get_label_map_dict(label_map_path_or_proto,
                       use_display_name=False,
                       fill_in_gaps_and_background=False):

  if isinstance(label_map_path_or_proto, string_types):
    label_map = load_labelmap(label_map_path_or_proto)
  else:
    _validate_label_map(label_map_path_or_proto)
    label_map = label_map_path_or_proto

  label_map_dict = {}
  for item in label_map.item:
    if use_display_name:
      label_map_dict[item.display_name] = item.id

else:
      label_map_dict[item.name] = item.id

  if fill_in_gaps_and_background:
    values = set(label_map_dict.values())

    if 0 not in values:
      label_map_dict['background'] = 0
    if not all(isinstance(value, int) for value in values):
      raise ValueError('The values in label map must be integers in order to'
                       'fill_in_gaps_and_background.')
    if not all(value >= 0 for value in values):
      raise ValueError('The values in the label map must be positive.')

    if len(values) != max(values) + 1:
```

```
    for value in range(1, max(values)):
      if value not in values:
        label_map_dict[str(value)] = value

  return label_map_dict

def get_label_map_hierarchy_lut(label_map_path_or_proto,
                                include_identity=False):
  if isinstance(label_map_path_or_proto, string_types):
    label_map = load_labelmap(label_map_path_or_proto)
  else:
    _validate_label_map(label_map_path_or_proto)
    label_map = label_map_path_or_proto

  hierarchy_dict = {
      'ancestors': collections.defaultdict(list),
      'descendants': collections.defaultdict(list)
  }
  max_id = -1
  for item in label_map.item:
    max_id = max(max_id, item.id)
    for ancestor in item.ancestor_ids:
      hierarchy_dict['ancestors'][item.id].append(ancestor)
    for descendant in item.descendant_ids:
      hierarchy_dict['descendants'][item.id].append(descendant)

  def get_graph_relations_tensor(graph_relations):
    graph_relations_tensor = np.zeros([max_id, max_id])
    for id_val, ids_related in graph_relations.items():
      id_val = int(id_val) - _LABEL_OFFSET
      for id_related in ids_related:
        id_related -= _LABEL_OFFSET
        graph_relations_tensor[id_val, id_related] = 1
```

```
    if include_identity:
      graph_relations_tensor += np.eye(max_id)
    return graph_relations_tensor

  ancestors_lut = get_graph_relations_tensor(hierarchy_dict['ancestors'])
  descendants_lut = get_graph_relations_tensor(hierarchy_dict['descendants'])
  return ancestors_lut, descendants_lut
def create_categories_from_labelmap(label_map_path, use_display_name=True):

  label_map = load_labelmap(label_map_path)
  max_num_classes = max(item.id for item in label_map.item)
  return convert_label_map_to_categories(label_map, max_num_classes,
                                         use_display_name)

def create_category_index_from_labelmap(label_map_path, use_display_name=True):

  categories = create_categories_from_labelmap(label_map_path, use_display_name)
  return create_category_index(categories)

def create_class_agnostic_category_index():

  return {1: {'id': 1, 'name': 'object'}}
```

10. ABBREVIATIONS

OCR: Optical Character Recognition

CNN: Convolution Neural Networks

YOLO: You Only Look Once

ANPR: Automatic Number Plate Recognition

SSD: Single Shot Detector