AUTOMATIC NUMBER PLATE RECOGNITION SYSTEM

A Thesis

Submitted in partial fulfilment of the

requirements for the award of the Degree of

MASTER OF COMPUTER APPLICATIONS

in

COMPUTER SCIENCE & ENGINEERING

By

DASETTY TEJASWINI

(20001F0025)

Under the esteemed guidance of

Dr. K. MADHAVI **M. Tech., Ph. D.**

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING (Autonomous)

ANANTAPURAMU-515002

ANDHRA PRADESH

2021-22

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING (Autonomous)

ANANTAPURAMU-515002

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled, **"Automatic Number Plate Recognition System",** is Bonafide work of **DASETTY TEJASWINI** bearing Admission. No: **20001F0025** submitted to the faculty of Computer Science & Engineering, in partial fulfilment of the requirements for the award of degree of **MASTER OF COMPUTER APPLICATIONS** from Jawaharlal Nehru Technological University Anantapur College of Engineering (Autonomous), Anantapuramu.

Signature of the Supervisor
Dr. K. Madhavi **M.Tech., Ph.D.**
Associate Professor
Dept. of Computer Science & Engg.,
J.N.T.U.A. College of Engineering.
Ananthapuramu– 515002.

Signature of the Head of the Department
Dr. K. Madhavi **M.Tech., Ph.D.**
Associate Professor & HOD
Dept. of Computer Science & Engg.,
J.N.T.U.A. College of Engineering.
Ananthapuramu– 515002.

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR

COLLEGE OF ENGINEERING (Autonomous)

ANANTHAPURAMU-515002

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DECLARATION

I am **DASETTY TEJASWINI** bearing Admission no. **20001F0025**, hereby declare that the project report entitled **"Automatic Number Plate Recognition System"** under the esteemed guidance of **Dr. K. MADHAVI** **M.Tech.,Ph.D.,** JNTUA College of Engineering Anantapuramu is submitted in partial fulfilment of the requirements for the award of the degree of **MASTER OF COMPUTER APPLICATIONS in** Computer Science and Engineering.

This is a record of Bonafide work carried out by me and the results embodied in this project has not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

DASETTY TEJASWINI

(20001F0025)

ACKNOWLEDGEMENT

The satisfaction and exhilaration that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I now have the opportunity to express my gratitude for all of them.

The first person I would like to thank is my guide, **Dr. K. MADHAVI**, Associate Professor of Computer Science and Engineering Department. Her wide knowledge and logical way of thinking have made a deep impression on me. Her understanding, encouragement and personal guidance have provided the basis for this thesis. She is a source of inspiration for innovative ideas and her support is known to all her students and colleagues.

My Special thanks to **Dr. K. MADHAVI,** Associate Professor and Head of the Department of Computer Science and Engineering, JNTUA College of Engineering (Autonomous), Anantapuramu. Her wide support, knowledge and enthusiastic encouragement have impressed me to be better involved in my project thesis and technical design also her ethical morals helped me to develop my personal and technical skills to deploy my project success.

I wish to thank **Prof. R. BHAVANI**, Vice Principal, JNTUA College of Engineering (Autonomous), Ananthapuramu, who has extended his support for the success of this project.

I wish to thank **Prof. P. SUJATHA**, Principal, JNTUA College of Engineering (Autonomous), Anantapuramu, for providing an excellent institutional environment and all facilities in completing this project.

I wish to thank the teaching staff and non-teaching Staff of the Computer Science and Engineering Department, JNTUA College of Engineering (Autonomous), Anantapuramu, for their support to complete my project successfully.

DASETTY TEJASWINI

(20001F0025)

ABSTRACT

Traffic control and vehicle owner identification have become a major problem in every country. Sometimes it becomes difficult to identify a vehicle owner who violates traffic rules and drives too fast. Therefore, it is not possible to catch and punish those kinds of people because the traffic personnel might not be able to retrieve the vehicle number from the moving vehicle. Therefore, there is a need to develop an Automatic Number Plate Recognition (ANPR) system as one of the solutions to this problem. There are numerous ANPR systems available today. These existing systems are based on different methodologies but still, it is a challenging task as some of the factors like high speed of the vehicle, non-uniform vehicle plate, the language of vehicle number and different lighting conditions can affect a lot in the number overall recognition rate.

The existing system for license plate recognition was developed by using Convolutional Neural Network and Optical Character Recognition. This system has the capability of detecting number plate for different font size and font style, different background colour for all Indian number plate. Despite the current developments, further improvements can still be achieved, such as deeper learning frameworks for object detection and recognition, fast coding programming frameworks, universal framework for general CNN, deep CNN models with the capacity to recognize objects from partly observed data and reliable interpretation of objects in complex conditions.

The proposed system uses deep learning frameworks called TensorFlow and Keras. TensorFlow is an open-source framework used in several algorithmic functions such as image and object detection. It uses a TensorFlow Object Detection Model that detects, locates and traces an object from a still image or video. In the training process, you will have large amounts of data they exist in very complicated format that requires a lot computation. TensorFlow involves several numerical computations and stores the data in a compact way. It also supports GPU where you need the image to be of high resolution. GPU has a faster compilation time so that the system provides higher accuracy in real time.

CONTENTS

List of figures

Chapters **Page No**

1. Introduction **1-5**

1.1 Objective 1-2

1.2 Existing System 2-2

1.3 Proposed System 2-3

1.4 Modules Description 3-4

1.5 Organization of Thesis 4-5

2. System Analysis **6-11**

2.1 Methodology & Algorithm 6-6

2.1.1 Methodology 6-6

2.1.2 Convolutional Neural Network 6-7

2.1.3 TensorFlow Object Detection Model 8-8

2.2 Feasibility Study 9-9

2.2.1 Economical Feasibility 9-9

2.2.2 Technical Feasibility 9-9

2.2.3 Social Feasibility 9-10

2.3 Software Requirement Specification 10-11

3. System Design **12-17**

3.1 UML Diagrams 12-12

3.1.1 Use Case Diagram 12-13

3.1.2 Sequence Diagram 14-14

3.1.3 Class Diagram 15-15

3.1.4 Activity Diagram 16-16

3.1.5 Data Flow Diagram 17-17

4. System Implementation **18-25**

4.1 Setup Paths 18-19

4.2 Download and install TensorFlow Object Detection Model 19-20

4.3 Creating Label Map 21-21

4.4 Train the Model 21-21

4.5 Detection from an Image 21-22

4.6 Apply OCR to Detection 22-23

4.7 Save Results 23-24

4.8 Real time Detection from your Webcam 24-25

5. System Testing **26-29**

5.1 Model Testing 26-26

5.2 Agile Testing 26-27

5.3 Unit Testing 27-27

5.4 Performance Testing 27-28

5.5 Integration Testing 28-29

6. Results **30-34**

6.1 Kaggle Dataset and its Annotations 30-31

6.2 Detection from an Image 31-31

6.3 Apply OCR to Detection 31-32

6.4 Saving Detected Images 32-32

6.5 Saving Detection Results in a CSV file 33-33

6.7 Real time Detection Results 33-34

7. Conclusion & Future Work **35-35**

References **36-36**

List of Figures

Figure No	Figure Name	Page No
2.1	Schematic Diagram of basic CNN	7
3.1	Use Case Diagram	13
3.2	Sequence Diagram	14
3.3	Class Diagram	15
3.4	Activity Diagram	16
3.5	Data Flow Diagram	17
6.1	Kaggle Dataset	30
6.2	Kaggle Dataset Annotations	30
6.3	Detection from an Image	31
6.4	Apply OCR to Detection	32
6.5	Saving Detected Images	32
6.6	Saving Detected Images Results	33
6.7	Real time Detection Results	34

Chapter 1
Introduction

The drastic increase in traffic on the roadways evokes a huge demand for technology for traffic surveillance and management. In this scenario, manual tracking the vehicles is nearly impossible nowadays due to various reasons like more vehicles, speed of the vehicles etc. There will be wastage of manpower and time for monitoring the traffic manually. Even if they are operated manually, that will reflect huge difficulties and enormous human errors. There are many number plate recognition systems already present for tracking vehicles and number plates using machine learning algorithms. But these algorithms fail due to their complexity in processing in real-time. Hence there is a need to develop a real-time automatic system that will help track the vehicles by tracing their number plates most efficiently and in real-time.

Automatic Number Plate Recognition (ANPR) plays a major role in tracking vehicles and also plays an inevitable role in systems like parking management systems, toll payment processing systems etc. In recent decades, computer vision technology has taken a great leap on several real-world issues. In an earlier period, vehicle number plates were identified using template matching techniques by detecting the width, height, contour area etc. Now several deep learningmodels trained over a huge amount of data are widely used in ANPR.

A large amount of the population who live in cities in which they need secure parking spaces to avoid the access of unauthorized persons. ANPR also helps collect the fee in tolls by identifying the vehicle's number plate. Vehicle tracking also helps with a fine collection for those who are trying to violate the traffic and road rules. It also helps in maintaining a database of the vehicles moving on the road.

1.1 Objective

The main objective of the system is to detect the number plate of the vehicles by using a deep learning Framework called TensorFlow Objection

Detection Model that could detect and traces the number plate of the vehicles and extract the number plate data in a CSV file.

1.2 Existing System

The existing system for license plate recognition was developed by using Convolutional Neural Network and Optical Character Recognition. This system has the capability of detecting number plate for different font size and font style, different background color for all Indian number plate. Despite the current developments, further improvements can still be achieved, such as deeper learning frameworks for object detection and recognition, fast coding programming frameworks, universal framework for general CNN, and deep CNN models with the capacity to recognize objects from partly observed data and reliable interpretation of objects in complex conditions.

Disadvantages:

- Does not have the capacity to recognize objects from partly observed data.
- It will not give proper accuracy in different image variations.

1.3 Proposed System

The proposed system uses deep learning frameworks called TensorFlow and Keras. TensorFlow is an open-source framework used in several algorithmic functions such as image and object detection. It uses a TensorFlow Object Detection Model that detects, locates and traces an object from a still image or video. In the training process, you will have large amounts of data they exist in very complicated format that requires a lot computation. TensorFlow involves several numerical computations and stores the data in a compact way. It also supports GPU where you need the imageto be of high resolution. GPU has a faster compilation time so that the system provides higher accuracy in real time.

Advantages:

- Faster detections and compilation time.
- Real time objects detection.
- It supports GPU where you need the image to be of high resolution.

- Provides higher accuracy in real time.

1.4 Modules Description

Our System consists of two modules:

1. Number Plate Detection
2. Optical Character Recognition

1. Number Plate Detection

The automatic number plate recognition system uses a pretrained object detection model which is developed by TensorFlow itself. This pretrained object detection model helps the ANPR system in recognizing the number plates of the vehicles. In order to train deep learning model to identify the number plate from image we have to follow following steps.

- Image Acquisition
- Upload Dataset
- Training
- Generate Results

Image Acquisition:

This is the initial phase of acquiring a picture in the given system. Input image is captured through camera.

Upload Dataset:

In this proposed system we are using the Kaggle car license plate dataset. This dataset contains 433 annotations and 433 car license plate images through which the system uses for training its number plate recognition model and the annotations of those particular images contain information about the license plate present in the image.

Data Preprocessing:

In Preprocessing step system will extract features from the dataset by using CNN algorithm.

Training:

In this model is trained by using large set of labelled data and Convolutional neural network that contains many layers. Convolutional neural network is trained by

optimizing on cross entropy loss function using mini batch gradient descent and black propagation.

Output:

After training model using convolutional neural network (CNN) this gives output containing bounding box to number plate.

2. Optical Character Recognition

The goal of Optical Character Recognition (OCR) is to classify optical patterns corresponding to alphanumeric or other characters. The process of OCR involves several steps including segmentation, feature extraction, and character recognition.

Character Segmentation:

After detection of number plate from image we have to perform character segmentation for character recognition. Character segmentation is the process of separating each character from the number plate. After completion of character segmentation, we get the bounding box on each character with its co-ordinates.

Character Recognition:

Character recognition is process of detecting and recognizing characters from input image and converting it into meaningful text in ASCII or other equivalent machine editable form. Character recognition is the process to classify the input character according to the predefined character class. In template matching, individual image pixels are used as features. Classification is performed by comparing an input character image with a set of templates from each character class.

1.5 Organization of Thesis

The rest of the Thesis is organized as follows.

Chapter 2:

This chapter explains about methodology and algorithms, feasibility study and software requirement specifications.

Chapter 3:

This chapter explains the entire system design of the project which explains how to develop the project and UML diagrams.

Chapter 4:

This chapter explains about the system implementation.

Chapter 5:

This chapter explains the software testing strategies of the project.

Chapter 6:

This chapter covers the final results of the proposed system.

Chapter 7:

Conclusion and future work of the project.

Chapter 2

System Analysis

System analysis is the process of observing systems for troubleshooting or development purposes. It is applied to information technology, where computer-based systems require defined analysis according to their makeup and design.

2.1 Methodology & Algorithm

Methodology is a set of principles, tools and techniques that are used to plan execute and manage the design. An algorithm is a set of steps to solve a problem or produce a specific outcome.

2.1.1 Methodology:

To achieve this objective discussed above, the following methodology is used:

- Input from an Image
- Data Preprocessing and training the model.
- Number Plate Detection using TensorFlow Object Detection model.
- Applying Optical Character Recognition.
- Character Recognition
- Character Segmentation
- Storing the detected images in a CSV file.
- Storing the extracted results in a CSV file.

2.1.2 Convolutional Neural Network

A convolutional neural network (CNN) is a type of artificial neural network used in image recognition and processing that is specifically designed to process pixel data. A Convolutional Neural Network is a Deep Learning algorithm which can takein an input image, assign importance to various aspects in the image and be able to differentiate one from the other.

The architecture of a CNN is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli in a restricted region of the visual field known as the Receptive Field.

A convolutional neural network can have tens or hundreds of layers that each learn to detect different features of an image. Filters are applied to each training image at different resolutions, and the output of each convolved image is used as the input to the next layer. The filters can start as very simple features, such as brightness and edges, and increase in complexity to features that uniquely define the object.

Like other neural networks, a CNN is composed of an input layer, an output layer, and many hidden layers in between. These layers perform operations that alter the data with the intent of learning features specific to the data. Three of the most common layers are: convolution, activation or ReLU, and pooling.

- **Convolution** puts the input images through a set of convolutional filters, each of which activates certain features from the images.
- **Rectified linear unit (ReLU)** allows for faster and more effective training by mapping negative values to zero and maintaining positive values. This is sometimes referred to as activation, because only the activated features are carried forward into the next layer.
- **Pooling** simplifies the output by performing nonlinear down sampling, reducing the number of parameters that the network needs to learn.

These operations are repeated over tens or hundreds of layers, with each layer learning to identify different features. The schematic diagram of basic CNN is shown in the figure 2.1.

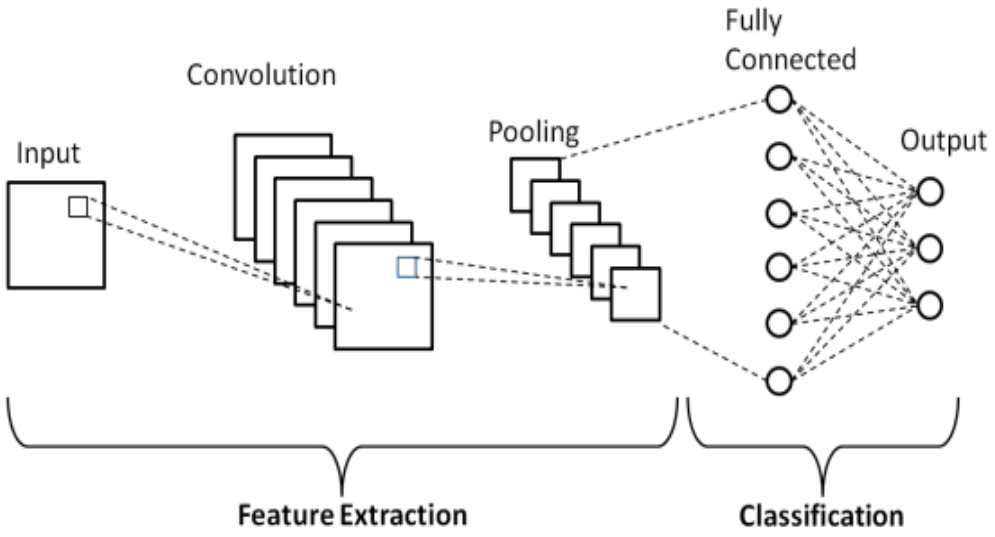


Figure 2.1 Schematic diagram of basic CNN

2.1.3 Tensorflow Object Detection Model

TensorFlow is an open-source framework used in several algorithmic functions such as image and object detection, voice search, and several others. TensorFlow boasts the application of Python and C++ APIs, as the framework hosts both Deep and Machine Learning algorithms.

TensorFlow is perfect for object detection because developers do not worry about the approach they should use to build an object detection model since both ML and DL algorithms are in TensorFlow. Object detection is a computer vision technique in which a software system can detect, locate, and trace the object from a given image or video. The special attribute about object detection is that it identifies the class of object (person, table, chair, etc.) and their location-specific coordinates in the given image.

Generally, the object detection task is carried out in three steps:

- Generates the small segments in the input. It generates large set of bounding boxes spanning the full image.
- Feature extraction is carried out for each segmented rectangular area to predict whether the rectangle contains a valid object.
- Overlapping boxes are combined into a single bounding rectangle.

In Tensorflow, all the computations involve tensors. A tensor is a vector or matrix of n-dimensions that represents all types of data. TensorFlow makes use of a graph framework. The graph gathers and describes all the series computations. Graph structure describes how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.

TensorFlow uses Python as a front-end language and runs on C++ — both languages are popular in object detection. TensorFlow object detection API prevents developers from building a model from scratch (training a model) by providing a set of existing operations. TensorFlow APIs are built on the "Model Zoo" which are already-trained models in the framework.

2.2 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysisthe feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understandingof the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

2.2.1 Economical Feasibility

2.2.2 Technical Feasibility

2.2.3 Social Feasibility

2.2.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies usedare freely available. Only the customized products had to be purchased.

2.2.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.2.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to

educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also ableto make some constructive criticism, which is welcomed, as he is the final user of the system.

2.3 Software Requirement Specification

A Software Requirements Specification (SRS) is a requirements specification for a software system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non- functional requirements are requirements which impose constraints on the design or implementation.

Functional Requirements

Functional requirement refers to the functionalities that are applicable to a system. The functional requirements of an automatic license plate recognition system are stated below. The system must be able to:

1. Load images from the system.
2. Extract frames from the image.
3. Localize the license plate region from the frames.
4. Segment characters from the localized plate.
5. Recognize the segmented characters and display them on the terminal.

Non-Functional Requirements

A non-functional requirement is a system must behave or how is the system behavior. This also specifies how the system's quality characteristics or quality attributes In order to put this constraint upon the specific system behavior, the qualities goals of the designed system should go in these:

Usability:

The usability of the system must be simple and easy. It should be capable of having minimuminteraction with the user, but at the same time provide the best results possible. It must beeasy to use.

Efficiency:

The system must be efficient to use, it should provide accurate results

and recognize thecharacters, in even unpredictable situations. It must work efficiently.

Required User Ability:

As working with Artificial Intelligence models requires some previous knowledge, the system should be designed in such a way that even people with minimal knowledge about thedomain and its reaches should be capable of using it. Users should require minimal knowledge to work with this system.

Reliability:

The system must be reliable, it should not lead to unnecessary crashes and stops Even though any kind of errors occurs, and it should have good exception handling mechanisms.

Security:

Preventive measures have to be taken to ensure that the data of the registration numbers and the data sets present in the storage units must not be allowed to be accessed by anyone. Authentication methods have to be established so that only those who have the permissions are capable of accessing the data sets and processing them.

Along with providing the functional requirements that the user expects the system to perform,certain requirements such as the above ones are to be met in order to grant the user a more friendly experience in handling and using the system.

Chapter 3

System Design

The System design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. The whole process has been divided into three stages i.e. capturing the image, plate localization and recognition of digits over the plate. The developed system first detects the vehicle and then captures the vehicle image. The vehicle number plate region is extracted using the image segmentation in an image. The optical character recognition technique is used for character recognition.

3.1 UML Diagrams

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

3.1.1 Use Case Diagram

Use case diagrams are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

A use case is a methodology used in system analysis to identify, clarify and organize system requirements.

The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. It consists of a group of elements (for example, classes and interfaces) that can be used together in a way that will have an effect larger than the sum of the separate elements combined.

The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed, the use case and goal are sometimes considered to be synonymous. The main purpose of a use case diagram is to show what system functions are performed for which actor. The figure 3.1 represents the Use Case diagram that shows the behavior and actions that are performed by the general ANPR system.

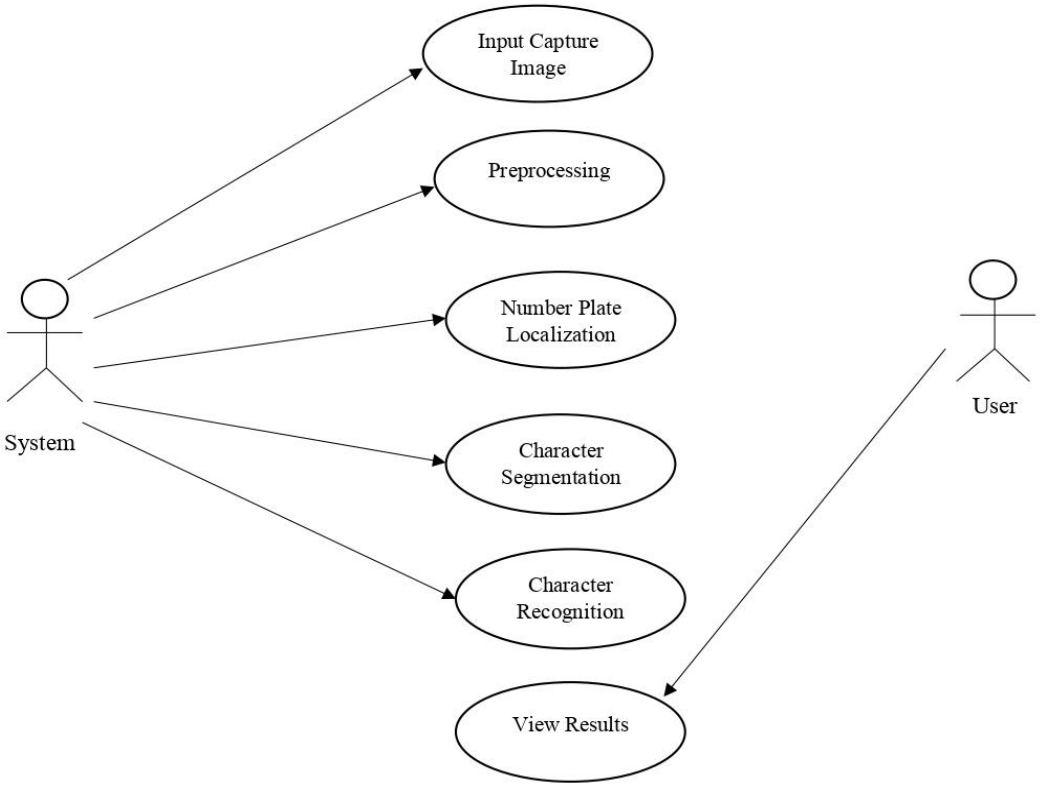


Figure 3.1 Use Case Diagram for ANPR

3.1.2 Sequence Diagram

A sequence diagram shows object interactions arranged in a time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

They capture the interaction between objects in the context of collaboration. Sequence Diagrams are time focused and they show the order of the interaction visually by using the vertical axis of the diagram to represent time, what messages are sent and when.

High-level interactions between users of the system and the system, between the system and other systems, or between subsystems (sometimes known as systemsequence diagrams). The activities that are involved in the ANPR system are arranged in a time sequence and the interactions between the objects are shown in the figure 3.2.

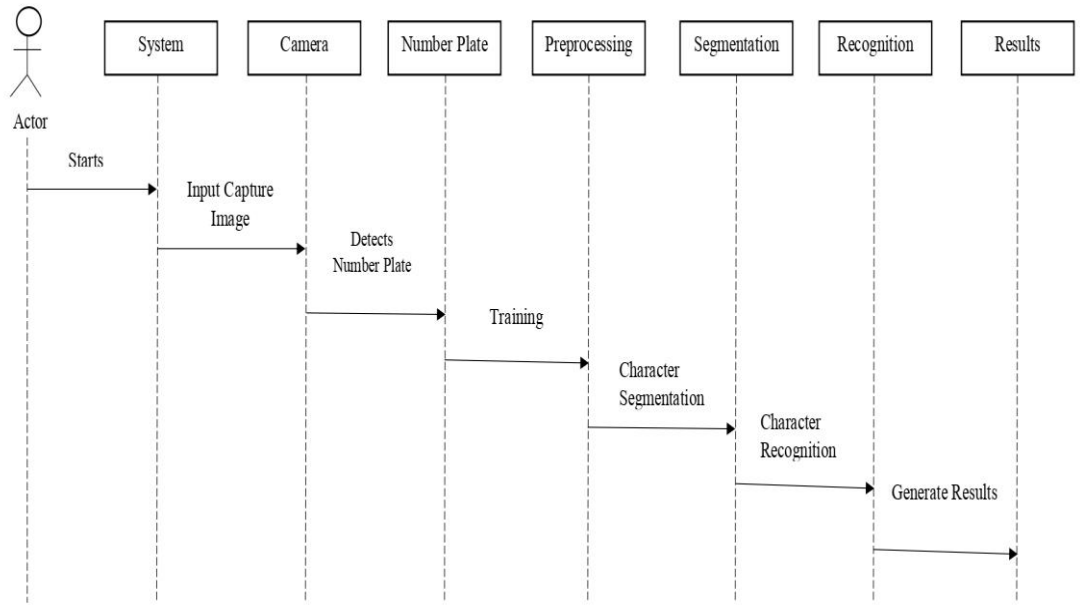


Figure 3.2 Sequence Diagram for ANPR

3.1.3 Class Diagram

The class diagram is a static diagram. It represents the static view of an application. The class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

A class diagram describes the attributes and operations of a class and also the constraints Imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

The class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. The figure 3.3 shows the overall structure of the ANPR system that represents all the classes, attributes, and methods involved in the system.

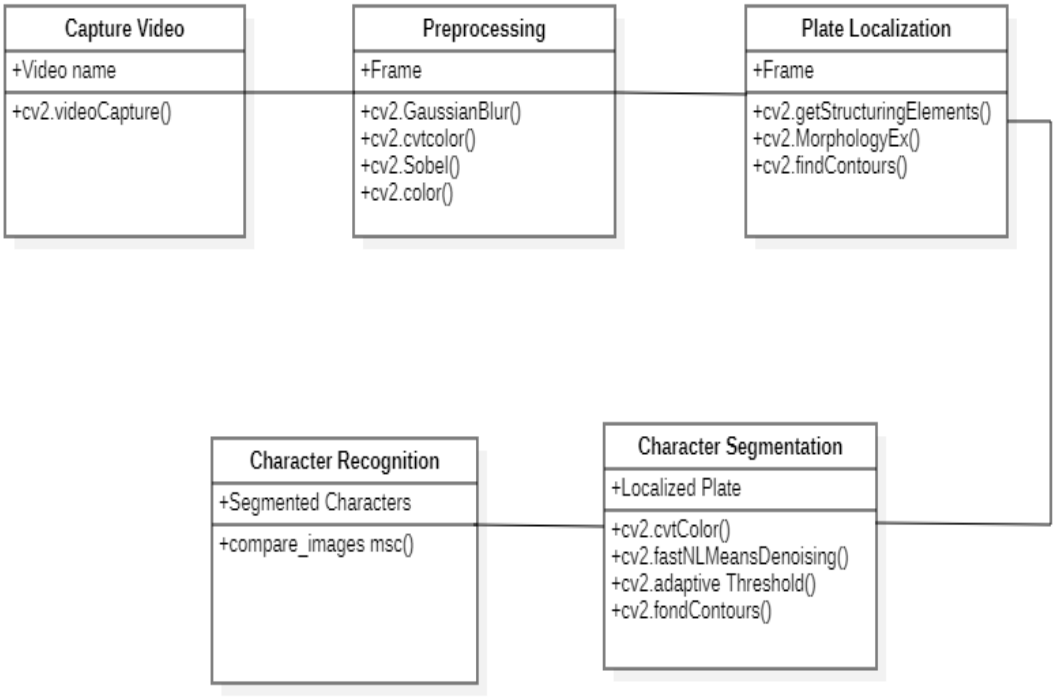


Figure 3.3 Class Diagram for ANPR

3.1.4 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. The figure 3.4 visually presents a series of actions or overall flow of control of the ANPR system.

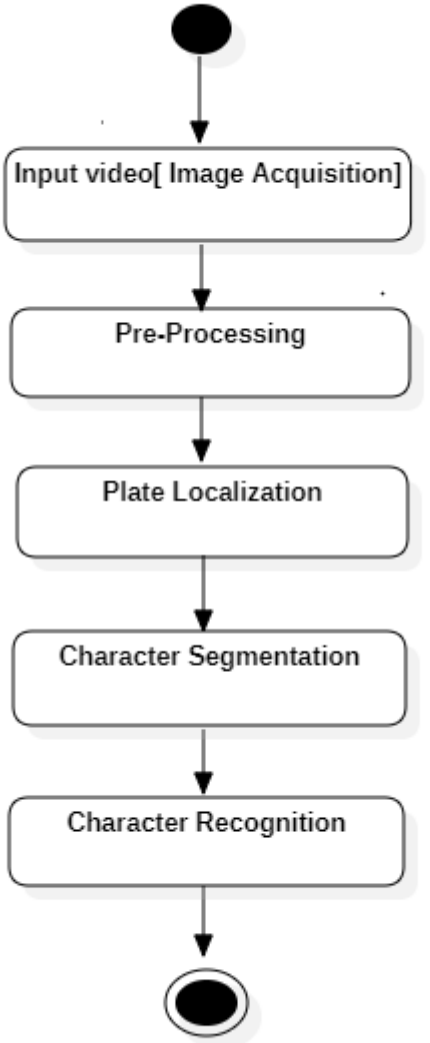


Figure 3.4 Activity Diagram for ANPR

4.1.5 Data Flow Diagram

Data flow diagrams are used to graphically represent the flow of data in a business- information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and report generation.

Data flow diagrams can be divided into logical and physical. The logical data flow diagram describes the flow of data through a system to perform certain functions of a business. The physical data flow diagram describes the implementation of the logical data flow. The figure 3.5 shows the general processes of the number plate recognition system that represents the overall flow of the data in the system.

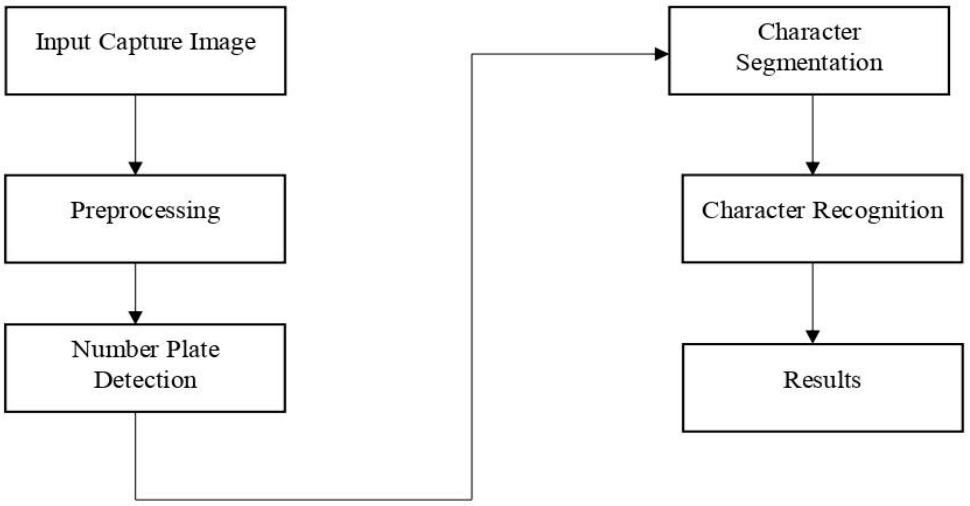


Figure 3.5 Data Flow Diagram for ANPR

Chapter 4

System Implementation

The proposed system has various modules in which the ANPR system implements essential steps to train, detect, recognize and extract the text from the vehicle number plate. ANPR System takes a long duration to train the object detection model. The system takes 1000 steps to train the model. The proposed system uses Convolution Neural Networks. There are large sets of modules in which each module consists of different levels of steps where each step involves its own processes.

4.1 Setup Paths

```
import os
CUSTOM_MODEL_NAME = 'my_ssd_mobnet'
PRETRAINED_MODEL_NAME =
'ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8'
PRETRAINED_MODEL_URL =
'http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet
_v2_fpnlite_320x320_coco17_tpu-8.tar.gz'
TF_RECORD_SCRIPT_NAME = 'generate_tfrecord.py'
LABEL_MAP_NAME = 'label_map.pbtxt'
paths = {
    'WORKSPACE_PATH': os.path.join('Tensorflow', 'workspace'),
    'SCRIPTS_PATH': os.path.join('Tensorflow','scripts'),
    'APIMODEL_PATH': os.path.join('Tensorflow','models'),
    'ANNOTATION_PATH': os.path.join('Tensorflow', 'workspace','annotations'),
    'IMAGE_PATH': os.path.join('Tensorflow', 'workspace','images'),
    'MODEL_PATH': os.path.join('Tensorflow', 'workspace','models'),
    'PRETRAINED_MODEL_PATH': os.path.join('Tensorflow', 'workspace','pre-
trained-models'),
    'CHECKPOINT_PATH': os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME),
    'OUTPUT_PATH': os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'export'),
```

```
    'TFJS_PATH':os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'tfjsexport'),
    'TFLITE_PATH':os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'tfliteexport'),
    'PROTOC_PATH':os.path.join('Tensorflow','protoc')
 }
files = {
    'PIPELINE_CONFIG':os.path.join('Tensorflow', 'workspace','models',
CUSTOM_MODEL_NAME, 'pipeline.config'),
    'TF_RECORD_SCRIPT': os.path.join(paths['SCRIPTS_PATH'],
TF_RECORD_SCRIPT_NAME),
    'LABELMAP': os.path.join(paths['ANNOTATION_PATH'],
LABEL_MAP_NAME)
}
```

The above code will create directories with names Tensorflow, workspace, and models and also creates a configuration file called as pipeline.config which contains the essential paths for the
anpr system.

4.2 Download and Install Tensorflow Object Detection Model

The automatic number plate recognition system uses a pretrained object detection model which is developed by TensorFlow itself. This pretrained object detection model helps the anpr system in recognizing the number plates of the vehicles.

```
# Install Tensorflow Object Detection
if os.name=='posix':
    !apt-get install protobuf-compiler
    !cd Tensorflow/models/research && protoc object_detection/protos/*.proto --
python_out=. && cp object_detection/packages/tf2/setup.py . && python -m pip
install .
if os.name=='nt':

url="https://github.com/protocolbuffers/protobuf/releases/download/v3.15.6/proto
c-3.15.6-win64.zip"
    wget.download(url)
```

```
    !move protoc-3.15.6-win64.zip {paths['PROTOC_PATH']}
    !cd {paths['PROTOC_PATH']} && tar -xf protoc-3.15.6-win64.zip
    os.environ['PATH'] += os.pathsep +
  os.path.abspath(os.path.join(paths['PROTOC_PATH'], 'bin'))
    !cd Tensorflow/models/research && protoc object_detection/protos/*.proto --
  python_out=. && copy object_detection\\packages\\tf2\\setup.py setup.py &&
  python setup.py build && python setup.py install
!cd Tensorflow/models/research/slim && pip install -e.
pip install tensorflow==2.4.1 tensorflow-gpu==2.4.1 –upgrade
import object_detection
if os.name =='posix':
!wget {PRETRAINED_MODEL_URL}
!mv                                    {PRETRAINED_MODEL_NAME+'.tar.gz'}
{paths['PRETRAINED_MODEL_PATH']}
!cd {paths['PRETRAINED_MODEL_PATH']} && tar -zxvf
{PRETRAINED_MODEL_NAME+'.tar.gz'}
if os.name == 'nt':
wget.download(PRETRAINED_MODEL_URL)
!move{PRETRAINED_MODEL_NAME+'.tar.gz'}
{paths['PRETRAINED_MODEL_PATH']}
!cd {paths['PRETRAINED_MODEL_PATH']} && tar -zxvf
{PRETRAINED_MODEL_NAME+'.tar.gz'}
```

After cloning the TensorFlow models, now the installation of the TensorFlow object detection model will be done. The anpr system checks whether the OS is Linux or Windows and runs the setup.py file to start installing the tensorflow object detection model.

After verifying that all the dependencies are included and all the paths are configured properly then we will restart the kernel and then run the above command to import the object_detection libraries to the jupyter notebook.

Now after importing the necessary libraries and packages we have to load the pre-trained model for licence plate detection. The above code downloads the pre-trained model which we load and use in the proposed system to make the number plate detector.

4.3 Creating Label Map

```
labels = [{'name':'licence', 'id':1}]
 with open(files['LABELMAP'], 'w') as f:
     for label in labels:
  f.write('item { \n')
  f.write('\tname:\'{}\'\n'.format(label['name']))
  f.write('\tid:{}\n'.format(label['id']))
  f.write('}\n')
```

The label map represents all the possible objects that the model will detect. The above text file is created as soon as the create label map code is executed. The label map.txt is the file that helps the system to create the label map that creates the object for what we detect i.e, License.

4.4 Train the Model

```
TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research',
'object_detection', 'model_main_tf2.py')
command = "python {} --model_dir={} --pipeline_config_path={}
--num_train_steps=10000".format(TRAINING_SCRIPT,
paths['CHECKPOINT_PATH'],files['PIPELINE_CONFIG'])
```

To train the object detection model the above code will initiate the training script. The code will generate a training command through which the developer can start the training and start the object detection training.

4.5 Detection from an Image

```
 import cv2
 import numpy as np
from matplotlib import pyplot as plt
 %matplotlib inline
 category_index =
 label_map_util.create_category_index_from_labelmap(files['LABELMAP'])
 IMAGE_PATH = os.path.join(paths['IMAGE_PATH'], 'testimg2.jpg')
 img = cv2.imread(IMAGE_PATH)
```

```
image_np = np.array(img)
input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
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
# detection_classes should be ints.
detections['detection_classes'] = detections['detection_classes'].astype(np.int64)
label_id_offset = 1
image_np_with_detections = image_np.copy()
viz_utils.visualize_boxes_and_labels_on_image_array(
      image_np_with_detections,
      detections['detection_boxes'],
      detections['detection_classes']+label_id_offset,
      detections['detection_scores'],
      category_index,
      use_normalized_coordinates=True,
      max_boxes_to_draw=5,
      min_score_thresh=.8,
      agnostic_mode=False)
plt.imshow(cv2.cvtColor(image_np_with_detections, cv2.COLOR_BGR2RGB))
plt.show()
```

Now the system is capable of detecting the number plates to do that we have to importopencv and numpy. The below code will import Numpy and OpenCV into the virtual environment. The code imports opencv and numpy libraries. The above code also imports the pyplot function from the matplotlib library. The code recognizes the licence plate and displays the label “Licence” and alongwith the accuracy.

4.6 Apply OCR to Detection

```
!pip install easyocr
```

```
!pip install torch==1.8.1+cu111 torchvision==0.9.1+cu111 torchaudio===0.8.1 -f
https://download.pytorch.org/whl/torch_stable.html
```

```
import easyocr
detection_threshold = 0.7
image = image_np_with_detections
scores = list(filter(lambda x: x> detection_threshold, detections['detection_scores']))
boxes = detections['detection_boxes'][:len(scores)]
classes = detections['detection_classes'][:len(scores)]
width = image.shape[1]
height = image.shape[0]
# Apply ROI filtering and OCR
for idx, box in enumerate(boxes):
     print(box)
      roi = box*[height, width, height, width]
      print(roi)
      region = image[int(roi[0]):int(roi[2]),int(roi[1]):int(roi[3])]
      reader = easyocr.Reader(['en'])
      ocr_result = reader.readtext(region)
      print(ocr_result)
   plt.imshow(cv2.cvtColor(region, cv2.COLOR_BGR2RGB))
for result in ocr_result:
      print(np.sum(np.subtract(result[0][2],result[0][1])))
   print(result[1])
```

4.7 Save Results

```
import csv
import uuid
'{}.jpg'.format(uuid.uuid1())
def save_results(text, region, csv_filename, folder_path):
   img_name = '{}.jpg'.format(uuid.uuid1())
       cv2.imwrite(os.path.join(folder_path, img_name), region)
       with open(csv_filename, mode='a', newline='') as f:
```

```
        csv_writer = csv.writer(f, delimiter=',', quotechar='"',
quoting=csv.QUOTE_MINIMAL)
        csv_writer.writerow([img_name, text])
region
save_results(text, region, 'detection_results.csv', 'Detection_Images')
```

4.8 Realtime Detection from Your Webcam

```
!pip uninstall opencv-python-headless -y
cap = cv2.VideoCapture(0)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
while cap.isOpened():
   ret, frame = cap.read()
   image_np = np.array(frame)
      input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0),
dtype=tf.float32)
   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
   # detection_classes should be ints.
   detections['detection_classes'] = detections['detection_classes'].astype(np.int64)
   label_id_offset = 1
   image_np_with_detections = image_np.copy()
   viz_utils.visualize_boxes_and_labels_on_image_array(
            image_np_with_detections,
            detections['detection_boxes'],
            detections['detection_classes']+label_id_offset,
            detections['detection_scores'],
            category_index,
            use_normalized_coordinates=True,
            max_boxes_to_draw=5,
            min_score_thresh=.8,
```

```
            agnostic_mode=False)
        try:
        text, region = ocr_it(image_np_with_detections, detections, detection_threshold,
region_threshold)
        save_results(text, region, 'realtimeresults.csv', 'Detection_Images')
    except:
        pass
    cv2.imshow('object detection',  cv2.resize(image_np_with_detections, (800, 600)))
        if cv2.waitKey(10) & 0xFF == ord('q'):
        cap.release()
        cv2.destroyAllWindows()
        break
```

And finally to detect a number plate in real-time the code below will start a python window which uses a webcam and captures the live video footage.The above code import the OpenCV and uses the cv2 function to access the web camera. To exit the camera window key ‘q’ should be clicked.

Chapter 5

System Testing

Testing is an integral part of developing any operational system. Testing is a method to checkwhether the actual product matches expected requirements and to ensure that product is Defect free. It involves the execution of system components using manual or automated tools to evaluate one or more properties of interest. The purpose of testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

The testing of the Detection model has to be done with the aim that, if there are any inaccuracies they have to be corrected and prevented so that they don't occur again in the future.

5.1 Model Testing

There are multiple aspects that need to be looked at while testing Automatic Number plate recognition.

- The first aspect that has to be looked at is whether the Model is capable of identifying the images of the vehicles properly. That is this is a primary need, because only when the images of the vehicles are distinguished properly and identifiably can they be usedfor further processing.
- The second aspect is whether the images of the number plates are identified and localized properly.
- The third aspect is whether the characters in the number plates are being identifiedand segmented properly.
- And the final testing is testing whether the Character Recognition model is capable of correctly recognizing the characters.

5.2 Agile Testing

Most of the Automation processes follow the agile process. Agile testing is a testing practicethat follows the rules and principles of agile software development. Unlike the Waterfall method, Agile Testing can begin at the start of the

project with continuous integration between development and testing. Agile Testing is not sequential (in the sense it's executed only after the coding phase) but continuous.

5.3 Unit Testing

Unit testing is a type of software testing where individual units or components of software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

Unit testing of the Automatic Vehicle Number Recognition contains

- Testing the input channel.
- Image input - Testing whether the images are correctly being loaded
- Test load model - see whether the model is successfully being loaded.
- Preprocessing Test
- Number Plate Localization - Checking if the number is being correctly detected.
- Number Plate Isolation - Checking if the number plate is correctly being retrieved.
- Character Segmentation Test.
- Output Display Test.

All of these minute components make up the entire system and only when all of these are successfully working seamlessly there can be a perfectly working system.

5.4 Performance Testing

Performance testing checks the speed, response time, reliability, resource usage, and scalability of a software program under its expected workload. The purpose of Performance Testing is not to find functional defects but to eliminate performance bottlenecks in the software or device. The focus of Performance Testing is checking a software programs

- Speed - The time that the system takes to detect and output the vehicle numbers.
- Scalability - The maximum load of how many input streams can the system handle ata time.
- Stability - The ability of the system to stay stable, even in the face of unpredictablesituations and unpredictable loads.

The key elements of the performance testing should be focused on the following.

1. Capacity related issues regarding the system and how much load it would be able tohandle without hindering its performance and throughput.
2. The task completion time taken by the system to work and output the results onvarious types of inputs provided.
3. The accuracy with which the vehicle numbers are predicted without any errors.

For this stage of testing, the Model was tested during both the training phases and after thetraining phases.

- While the CNNs were being trained, half the data set was used to train it while half ofit was used to validate it and improve its performance.
- After the system was integrated also, a number of test cases were run on it, to ensureits performance and accuracy never toppled.
- Diverse Images with varying aspects of composition were used to test.
- It ensured that all the possible situations were tested, including situationswhere the number plates were out of the ordinary.

5.5 Integration Testing

Integration Testing is defined as a type of testing where software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. The purpose of this level of testing is to expose defects in the interaction between these software modules when they are integrated.

Integration testing is crucial for this model because every stage is dependent on its previous stage and only on the successful working of all the stages can we have a fully functional system.

As there are multiple models involved in the complete detection of the number plate, therehas to be successful coordination between them all.

The following steps were taken to ensure the Integrated System was working properly.

- Each individual model was integrated with its immediate next model and their working accuracy was tested.
- Each individual entity in the system was iteratively tested with its neighbor's errorless performance and compatibility issues, when found they were immediately corrected.
- The system was integrated incrementally, only after ensuring that the parts were working successfully as a whole.
- Only after performing a complete set of tests the system was integrated, incrementally and its entire performance was tested again.

Chapter 6

Results

6.1 Kaggle Dataset and its Annotations

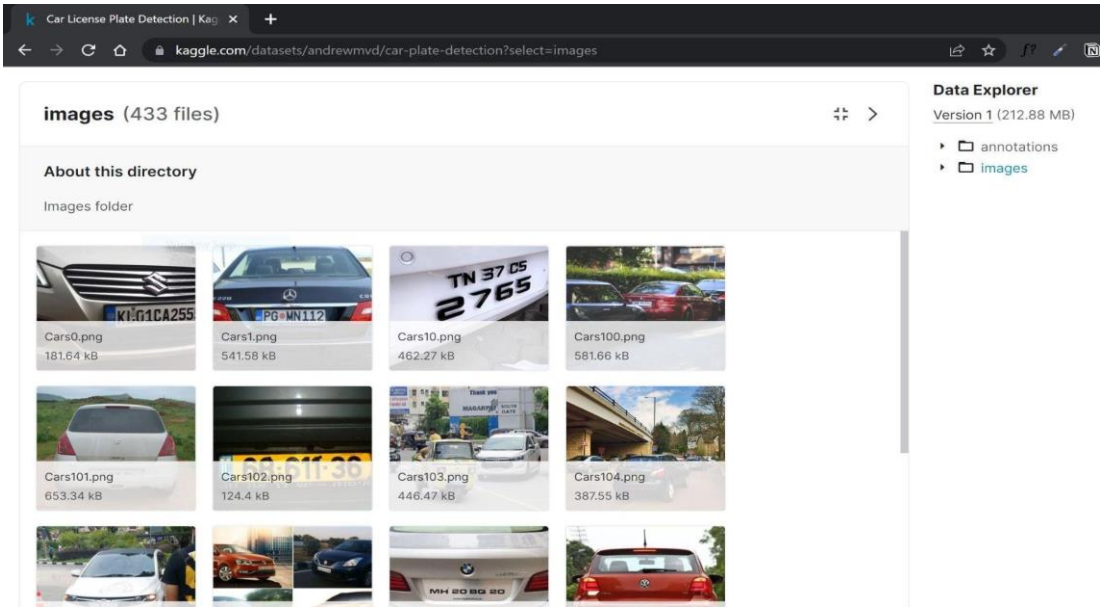


Figure 6.1 Kaggle Dataset

The figure 6.1 is Kaggle car license plate dataset. It contains 433 annotations and 433 car license plate images through which the system uses for training.

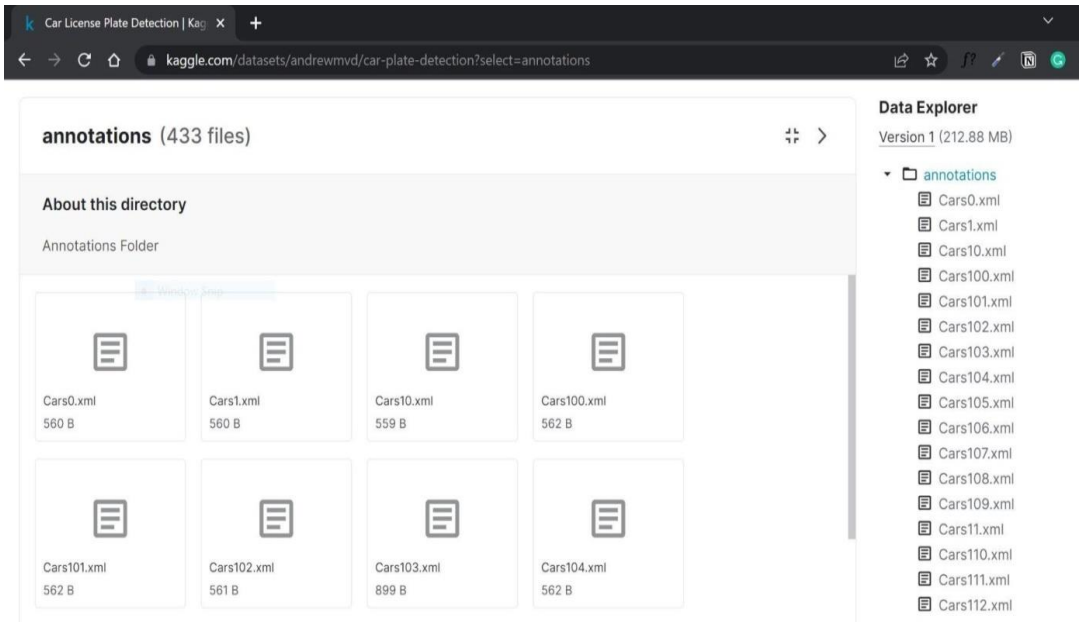


Figure 6.2 Kaggle Dataset Annotations

The figure 6.2 contains annotations of those particular images contain information about the license plate present in the image.

6.2 Detection from an Image

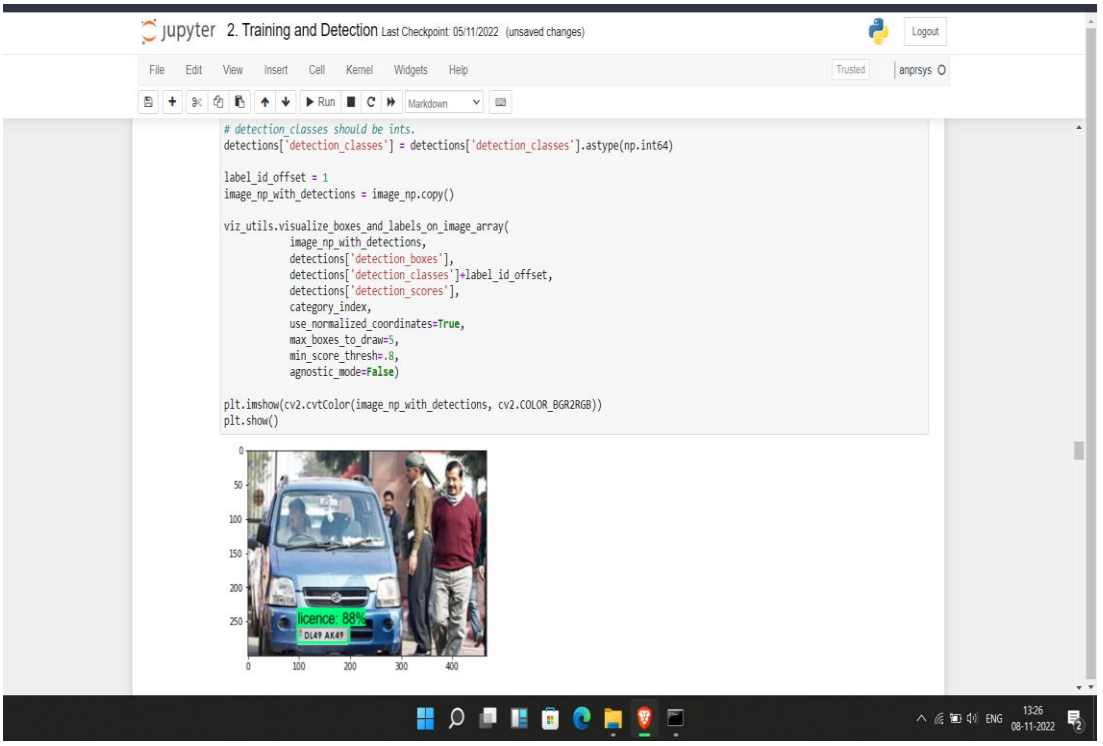


Figure 6.3 Detection from an Image

The figure 6.3 shows that the ANPR system detected the number plate and labeled it with the label name License and also showing the accuracy as 88%.

6.3 Apply OCR to Detection

The figure 6.4 shows along with the extracted number plate region the system extracts the text on the licence plate. Character recognition is done by applying OCR and the characters are extracted from the image. The characters present on the licence plate are DL49 AK49 and the ANPR System gave output ‘DL49 AK49 ’.

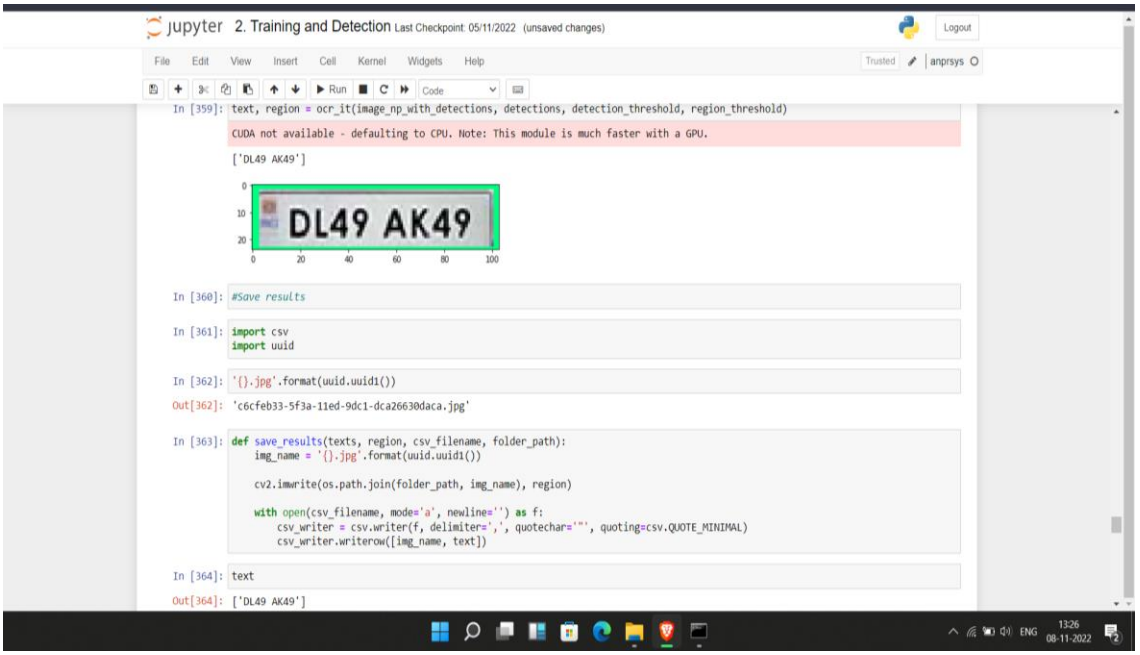


Figure 6.4 Apply OCR to Detection

6.4 Saving Detected Images

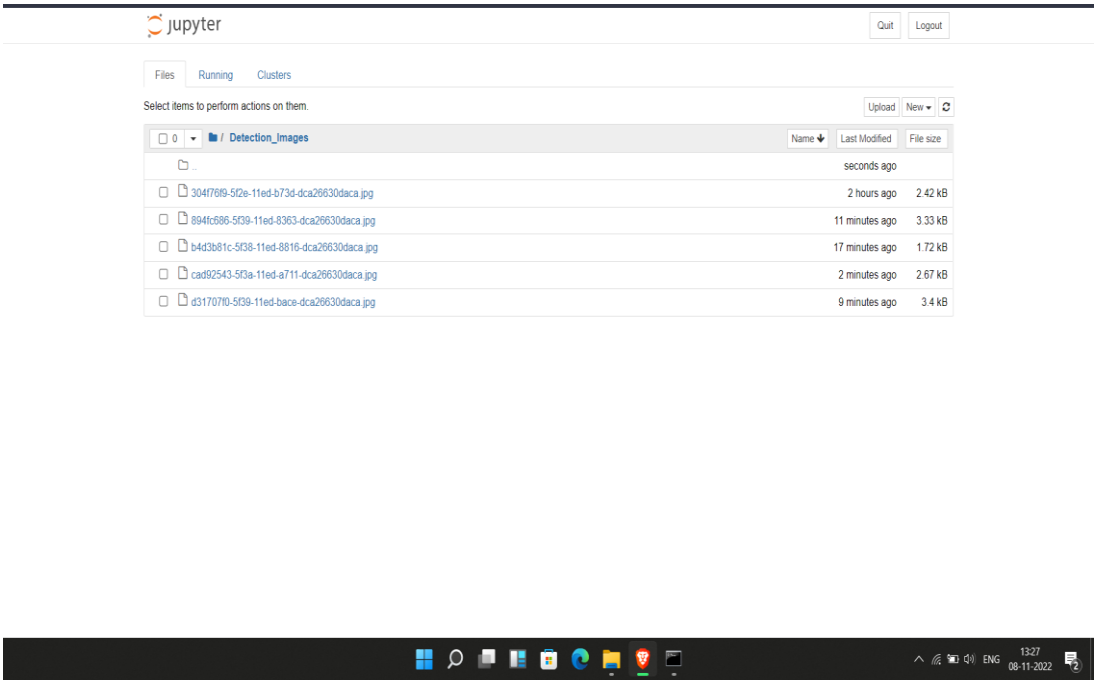


Figure 6.5 Saving Detected Images

The figure 6.5 shows that the detected number plate images are saved in a Detection_Images folder

6.5 Saving Detection Results in a CSV File

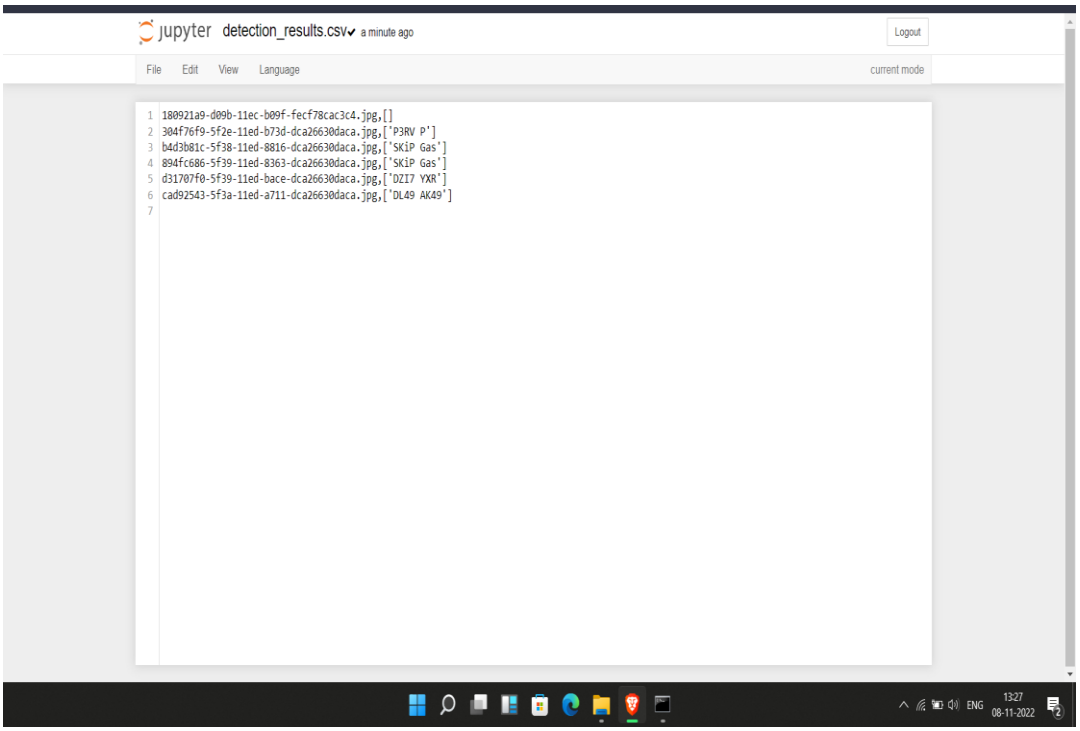
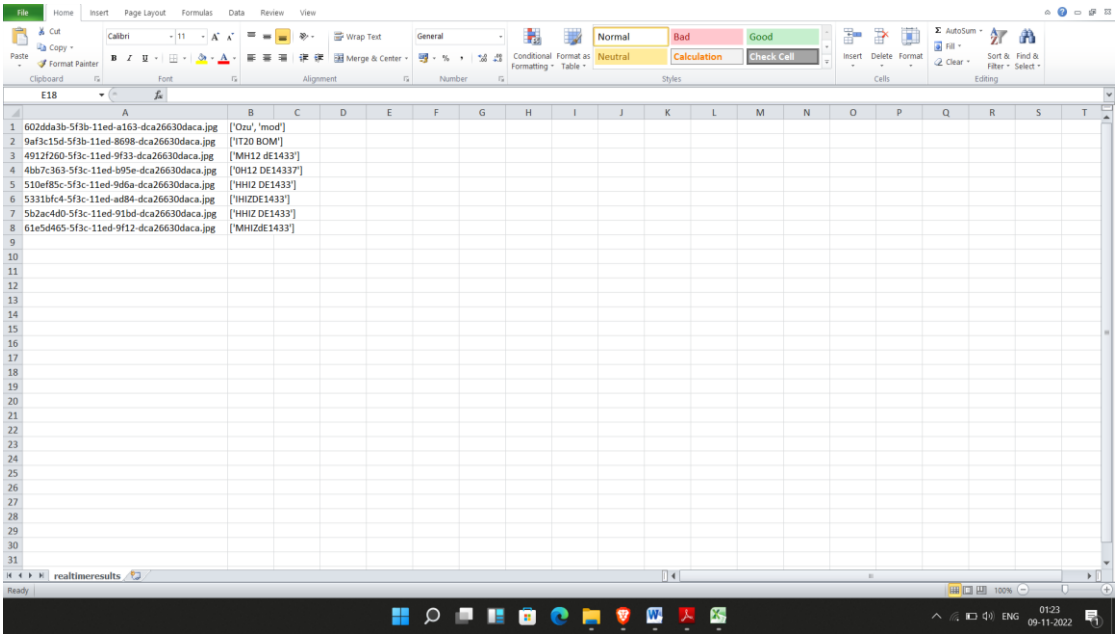


Figure 6.6 Saving Detected Images Results

The figure 6.6 the characters that are extracted from the number plates are recorded in a detection_results.CSV file.

6.6 Real time Detection Results

The figure 6.7 shows that the real time detected images file names and the extracted characters are saved in a realtime_detection_results.CSV file.



	A	B
1	602dda3b-5f3b-11ed-a163-dca26630daca.jpg	['Ozu', 'mod']
2	9af3c15d-5f3b-11ed-8698-dca26630daca.jpg	['IT20 BOM']
3	4912f260-5f3c-11ed-9f33-dca26630daca.jpg	['MH12 dE1433']
4	4bb7c363-5f3c-11ed-b95e-dca26630daca.jpg	['0H12 DE14337']
5	510ef85c-5f3c-11ed-9d6a-dca26630daca.jpg	['HHI2 DE1433']
6	5331bfc4-5f3c-11ed-ad84-dca26630daca.jpg	['IHIZDE1433']
7	5b2ac4d0-5f3c-11ed-91bd-dca26630daca.jpg	['HHIZ DE1433']
8	61e5d465-5f3c-11ed-9f12-dca26630daca.jpg	['MHIZdE1433']

Figure 6.7 Real time Detection Results

Chapter 7

Conclusion and Future Work

The project Automatic Number Plate Recognition system has successfully developed to detect and recognize the number plate by using deep learning framework TensorFlow. The system is likely to provide higher accuracy in real time by detecting the number plate using Tensorflow Object Detection model under different image variations and complex conditions. Once those plates have been detected we apply OCR to extract the text and save the results in a csv file for further use.

The project results in accurate and efficient because of GPU support, it works very fast so that time be saved. The performance of the system is better than the existing system that provides faster detections.

Thus, in the future work the OCR accuracy could be improved by using a different OCR engine and the inclusion of recognizing unwanted symbols should be improved.

References

1. Gondhalekar, Dnyanisha and Chalke, Omkar and Bansal, Siddharth and Banerjee, Soumi, Vehicle License Plate Recognition Using Neural Networks (May 7, 2021). Available at SSRN: https://ssrn.com/abstract=3866116.
2. Kirad Varad Vinay, Omkar Balaobaiah, Mujawar Sohail Mahiboob, Automatic Number Plate Recognition For Different Fonts and Non-Roman Script Vol. 5, Issue 11, ISSN No. 2455-2143, Pages 314-317.
3. International Journal of Engineering Applied Sciences and Technology, 2021 Vol. 5, Issue 11, ISSN No. 2455-2143, Pages 314-317.
4. Pratiksha Jain, Neha Chopra, Vaishali Gupta, "Automatic License Plate Recognition using OpenCV", International Journal of Computer Applications Technology and Research Volume 3– Issue 12, 756 - 761, 2014, ISSN: -2319–8656.
5. Tella Pavani, DVR Mohan, "Number Plate Recognition by using open CV-Python", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 06 Issue: 03.