AUTOMATIC NUMBER PLATE RECOGNITION AND PARKING OCCUPANCY PREDICTION

Mini Project Report

Submitted in partial fulfillment of the requirements for the award

of degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

by

SOURAV A K
MAC19CS054

MITHUN K
MAC19CS039

ADITHYAN S P
MAC19CS002

Department of Computer Science & Engineering
Mar Athanasius College of Engineering
Kothamangalam

JULY 2022

AUTOMATIC NUMBER PLATE RECOGNITION AND PARKING OCCUPANCY PREDICTION

Mini Project Report

Submitted in partial fulfillment of the requirements for the award

of degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

by

SOURAV A K
MAC19CS054

MITHUN K
MAC19CS039

ADITHYAN S P
MAC19CS002

Department of Computer Science & Engineering
Mar Athanasius College of Engineering
Kothamangalam

JULY 2022

DECLARATION

We, undersigned hereby declare that the mini project report Automatic Number Plate Recognition and Parking Occupancy Prediction, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Prof. Sukanyathara J. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Date: 22.07.2022

Kothamangalam

Sourav A K

Adithyan S P

Mithun K

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MAR ATHANASIUS COLLEGE OF ENGINEERING
KOTHAMANGALAM

CERTIFICATE

This is to certify that the report entitled **Automatic Number Plate Recognition and Parking Occupancy Prediction** submitted by **Mr. Sourav A K, Reg. No. MAC19CS054, Mr. Mithun K, Reg. No. MAC19CS039 and Mr. Adithyan S P, Reg. No. MAC19CS002** towards partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Computer science and Engineering from APJ Abdul Kalam Technological University for JULY 2022 is a bonafide record of the project carried out by them under our supervision and guidance.

Prof. Sukanyathara J	**Dr. Elizabeth Isaac**	**Prof. Joby George**
Project Guide	Project Coordinator	Head of Department

Internal Examiner (s) **External Examiner(s)**

Date:22.07.2022 Dept. Seal

ACKNOWLEDGEMENT

First and foremost, we sincerely thank the 'God Almighty' for his grace for the successful and timely completion of the project.

We express our sincere gratitude and thanks to Dr. Bos Mathew Jos, Principal and Prof. Joby George, Head of the Department for providing the necessary facilities and their encouragement and support.

We owe special thanks to our project guide Prof. Sukanyathara J for her guidance, constant supervision, encouragement, and support throughout the period of this project work.

We are grateful to the staff-in-charge Dr. Elizabeth Isaac for her corrections, suggestions and sincere efforts to co-ordinate the project under a tight schedule.

We express our sincere thanks to staff members in the department of Computer Science and Engineering who have taken sincere efforts in helping us to conduct this project.

Finally, we would like to acknowledge the heartfelt efforts, comments, criticisms, cooperation, and tremendous support given to us by our dear friends during the preparation of the project and also during the presentation without whose support this work would have been all the more difficult to accomplish.

ABSTRACT

Parking issues have been receiving increasing attention. An accurate parking occupancy prediction is a key prerequisite to optimally manage limited parking resources, thus acknowledging the drivers about the parking availability in a particular area. This project is mainly aiming to reduce the problem of parking issue in this college. Also, through image processing number plate, the vehicle of outsiders is recognized and will alert the authorized staff about their entry and will give a specific time period to such vehicles to move out from the parking area. Thus, this project will deliver a significant impact in maintaining the parking spaces efficiently and can be very effective in preventing malicious activities inside the campus. ANPR technology is used for identifying the number plate. TensorFlow library is used for training ML model and EasyOCR module is used for optical character recognition. Another feature involves predicting the future parking occupancy rate so that we can get idea on the availability of parking spots. ARIMA model is used for improving the predicting accuracy.

CONTENTS

List of Figures

List of Abbreviations

ANPR	Automatic Number Plate Recognition
CSV	Comma Separated Values
CSS	Cascading Style Sheet
JS	JavaScript
HTML	Hypertext Markup Language
ARIMA	Autoregressive Integrated Moving Average
ROI	Region of Interest
ML	Machine Language
XML	Extensible Markup Language
OCR	Optical Character Recognition
API	Application Programming Interface
ROC	Region-of-Candidates

CHAPTER 1
INTRODUCTION

The search for an empty parking spot can become an agonizing experience for the city's urban drivers. A recent article claims that drivers cruising for a parking spot in SF generate 30% of all downtown congestion. These wasted miles not only increase traffic congestion, but also lead to more pollution and driver anxiety. In order to alleviate this problem, the city armed 7000 metered parking spaces and 12,250 garages spots (total of 593 parking lots) with sensors and introduced a mobile application called SFpark, which provides real time information about availability of a parking lot to drivers.

However, safety experts worry that drivers looking for parking may focus too much on their phone and not enough on the road. Furthermore, the current solution does not allow drivers to plan ahead of a trip. We wish to tackle the parking problem by (i) predicting the occupancy rate, defined as number of occupied parking spots over total number of spots, of parking lots in a given future time (ii) working on aggregated parking lots to explore if there is estimation error reduction pattern in occupancy prediction, (iii) classifying daily parking occupancy patterns to investigate different travel behavior at different time.

1.1 MACHINE LEARNING

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers, but not all machine learning is statistical learning.

The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. Some implementations of machine learning use data and neural networks in a way that mimics the working of a biological brain. In its application across business problems, machine learning is also referred to as predictive analytics.

1.2 PROJECT OUTLINE

The project intends to develop a website for the college which predicts the availability of parking slots at a given period of time. Also, through number plate detection the vehicles entering the campus are classified as staff vehicle and visitor. The admin of the web page, that is, security staff can view the details of vehicle that are parked at that instant. This system can significantly control the parking congestion in the campus by acknowledging the staff about the availability of parking slots and thus recommends a time period for the greater availability of parking spaces. And in case of visitor, a timer of 20 minutes is set at the time of entry of such vehicle and alert the security staff about such vehicle after the time expired.

CHAPTER 2
BACKGROUND

In this chapter we discuss the necessary technologies and features that we require to build the project.

2.1 ARIMA MODEL

In statistics and econometrics, and in time series analysis, an autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA) model. Both models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). ARIMA models are applied in some cases where data show evidence of non-stationarity in the sense of mean (but not variance/autocovariance), where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity of the mean function (i.e., the trend). When the seasonality shows in a time series, the seasonal-differencing could be applied to eliminate the seasonal component. Since the ARMA model, according to the World's decomposition theorem, [is theoretically sufficient to describe a regular (a.k.a. purely nondeterministic) wide-sense stationary time series, we are motivated to make stationary a non-stationary time series, e.g., by using differencing, before we can use the ARMA model. Note that if the time series contains a predictable sub-process (a.k.a. pure sine or complex-valued exponential process), the predictable component is treated as a non-zero-mean but periodic (i.e., seasonal) component in the ARIMA framework so that it is eliminated by the seasonal differencing.

The AR part of ARIMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for "integrated") indicates that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have

been performed more than once). The purpose of each of these features is to make the model fit the data as well as possible.

Non-seasonal ARIMA models are generally denoted ARIMA(p,d,q) where parameters p, d, and q are non-negative integers, p is the order (number of time lags) of the autoregressive model, d is the degree of differencing (the number of times the data have had past values subtracted), and q is the order of the moving-average model. Seasonal ARIMA models are usually denoted ARIMA(p,d,q)(P,D,Q)m, where m refers to the number of periods in each season, and the uppercase P,D,Q refer to the autoregressive, differencing, and moving average terms for the seasonal part of the ARIMA model.

An ARIMA model can be understood by outlining each of its components as follows:

AUTOREGRESSION (AR): refers to a model that shows a changing variable that regresses on its own lagged, or prior, values.

INTEGRATED (I): represents the differencing of raw observations to allow for the time series to become stationary (i.e., data values are replaced by the difference between the data values and the previous values).

MOVING AVERAGE (MA): incorporates the dependency between an observation and a residual error from a moving average model applied to lagged observations.

2.2 OPTICAL CHARACTER RECOGNITION

Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image.

Widely used as a form of data entry from printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation – it is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted)

text-to-speech, key data and text mining. OCR is a field of research in pattern recognition, artificial intelligence, and computer vision.

Early versions needed to be trained with images of each character and worked on one font at a time. Advanced systems capable of producing a high degree of recognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs. Some systems are capable of reproducing formatted output that closely approximates the original page including images, columns, and other non-textual components.

2.3 TENSORFLOW

Machine learning is a complex discipline but implementing machine learning models is far less daunting than it used to be, thanks to machine learning frameworks - such as Google's TensorFlow - that ease the process of acquiring data, training models, serving predictions, and refining future results.

Created by the Google Brain team and initially released to the public in 2015, TensorFlow is an open-source library for numerical computation and large-scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning models and algorithms (aka neural networks) and makes them useful by way of common programmatic metaphors. It uses Python or JavaScript to provide a convenient front-end API for building applications, while executing those applications in high-performance C++. [Make sense of machine learning: AI, machine learning, and deep learning: Everything you need to know. | Deep learning explained. | Machine learning explained. | Machine learning algorithms explained. | Machine learning skills for software engineers. | Go deep into analytics and big data with the InfoWorld Big Data and Analytics Report newsletter.]

TensorFlow, which competes with frameworks such as PyTorch and Apache MX Net, can train and run deep neural networks for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation)-based

simulations. Best of all, TensorFlow supports production prediction at scale, with the same models used for training.

TensorFlow also has a broad library of pre-trained models that can be used in your own projects. You can also use code from the TensorFlow Model Garden as examples of best practices for training your own models.

2.4 CUDA

CUDA (or Compute Unified Device Architecture) is a parallel computing platform and application programming interface (API) that allows software to use certain types of graphics processing units (GPUs) for general purpose processing, an approach called general-purpose computing on GPUs (GPGPU). CUDA is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements, for the execution of compute kernels.

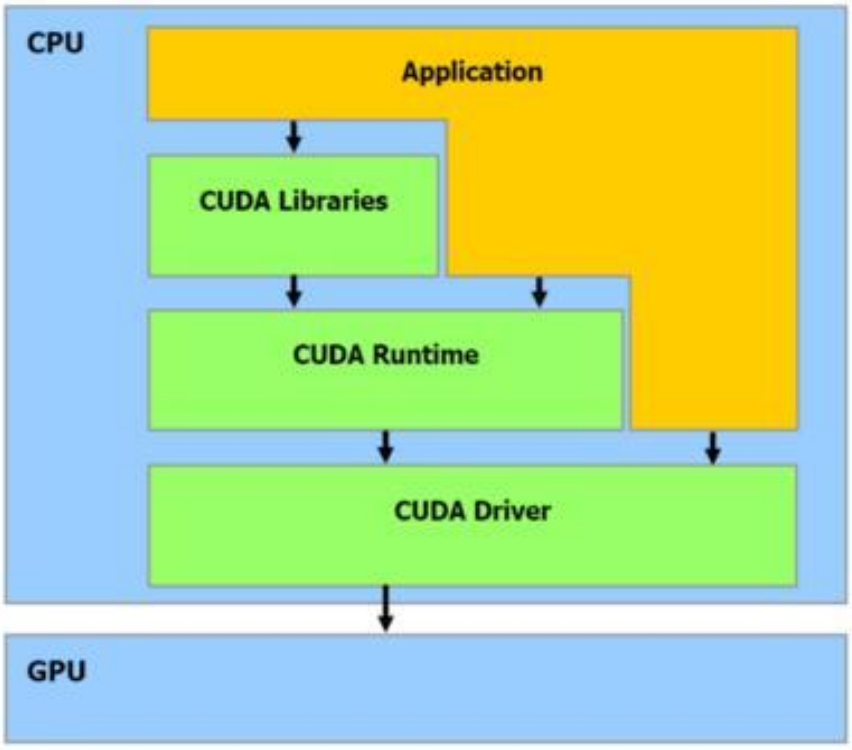

Fig 2.1 Architecture of CUDA

CUDA is designed to work with programming languages such as C, C++, and Fortran. This accessibility makes it easier for specialists in parallel programming to use GPU resources, in contrast to prior APIs like Direct3D and OpenGL, which required advanced skills in graphics programming. CUDA-powered GPUs also support programming frameworks such as OpenMP, OpenACC and OpenCL; and HIP by compiling such code to CUDA.

CUDA was created by NVidia. When it was first introduced, the name was an acronym for Compute Unified Device Architecture, but Nvidia later dropped the common use of the acronym.

2.5 PYTHON

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming.

One reason why python is suitable for machine learning applications is that it has a huge number of libraries and frameworks: The Python language comes with many libraries and frameworks that make coding easy. This also saves a significant amount of time. The most popular libraries are NumPy, which is used for scientific calculations; SciPy for more advanced computations; and scikit, for learning data mining and data analysis. These libraries work alongside powerful frameworks like TensorFlow, CNTK, and Apache Spark. These libraries and frameworks are essential when it comes to machine and deep learning projects.

Also, python is a simple language offering reliable code. Machine learning is all about complicated algorithms and versatile workflows. So, the simplicity of Python helps the developers to deal with the complex algorithms. Also, it saves the time of developers as they only require concentrating on solving the ML problems rather than focusing on the technicality of language.

2.6 APACHE BEAM

Apache Beam is an open-source unified programming model to define and execute data processing pipelines, including ETL, batch and stream (continuous) processing. Beam Pipelines are defined using one of the provided SDKs and executed in one of the Beam's supported runners (distributed processing back-ends) including Apache Flink, Apache Samza, Apache Spark, and Google Cloud Dataflow.

2.7 AVRO PYTHON

Avro is a row-oriented remote procedure call and data serialization framework developed within Apache's Hadoop project. It uses JSON for defining data types and protocols, and serializes data in a compact binary format. Its primary use is in Apache Hadoop, where it can provide both a serialization format for persistent data, and a wire format for communication between Hadoop nodes, and from client programs to the Hadoop services. Avro uses a schema to structure the data that is being encoded. It has two different types of schema languages; one for human editing (Avro IDL) and another which is more machine-readable based on JSON.

2.8 EASYOCR

EasyOCR is actually a python package that holds PyTorch as a backend handler. EasyOCR like any other OCR(tesseract of Google or any other) detects the text from images but in my reference, while using it found that it is the most straightforward way to detect text from images also when high end deep learning library(PyTorch) is supporting it in the backend which makes it accuracy more credible. EasyOCR supports 42+ languages for detection purposes. EasyOCR is created by the company named Jaided AI company.

2.9 GOOGLE API PYTHON CLIENT

Google APIs are application programming interfaces (APIs) developed by Google which allow communication with Google Services and their integration to other services. Examples of these include Search, Gmail, Translate or Google Maps. Third-party apps can use these APIs to take advantage of or extend the functionality of the existing services.

The APIs provide functionality like analytics, machine learning as a service (the Prediction API) or access to user data (when permission to read the data is given). Another important example is an embedded Google map on a website, which can be achieved using the Static Maps API, Places API or Google Earth API

2.10 GOOGLE CLOUD FIRESTORE

Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development.

2.11 IPYTHON

IPython (Interactive Python) is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history. IPython provides the following features:

- Interactive shells (terminal and Qt-based).
- A browser-based notebook interface with support for code, text, mathematical expressions, inline plots and other media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.
- IPython is a NumFOCUS fiscally sponsored project.

2.12 KIWI SOLVER

Kiwi is an efficient C++ implementation of the Cassowary constraint solving algorithm. Kiwi is an implementation of the algorithm based on the seminal Cassowary paper. It is not a refactoring of the original C++ solver. Kiwi has been designed from the ground up to be lightweight and fast. Kiwi ranges from 10x to 500x faster than the original Cassowary solver with typical use cases gaining a 40x improvement. Memory savings are consistently > 5x.
In addition to the C++ solver, Kiwi ships with hand-rolled Python bindings for Python 3.7+.

2.13 MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

2.14 NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a NumFOCUS fiscally sponsored project.

2.15 OBJECT DETECTION

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos. Well-researched domains of object detection include face detection and pedestrian detection. Object detection has applications in many areas of computer vision, including image retrieval and video surveillance.

2.16 OPEN CV

OpenCV (Open-Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. All of the

new developments and algorithms appear in the C++ interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation. Wrappers in several programming languages have been developed to encourage adoption by a wider audience. In version 3.4, JavaScript bindings for a selected subset of OpenCV functions was released as OpenCV.js, to be used for web platforms.

2.17 PANDAS

pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself. Wes McKinney started building what would become pandas at AQR Capital while he was a researcher there from 2007 to 2010.

2.18 PILLOW

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 images file formats. It is available for Windows, Mac OS X and Linux. The latest version of PIL is 1.1.7, was released in September 2009 and supports Python 1.5.2–2.7.

Development of the original project, known as PIL, was discontinued in 2011. Subsequently, a successor project named Pillow forked the PIL repository and added Python 3.x support. This fork has been adopted as a replacement for the original PIL in Linux distributions including Debian and Ubuntu

2.19 SLIM

SLIM is developed at the Space Telescope European Coordinating Facility (ST-ECF) and is a slitless spectroscopy simulator program created to produce simulated ACS grism and prism images. It is written in Python programming language and covers all spectral elements available in the Advanced Camera for Surveys (ACS) installed on board the Hubble Space

Telescope (HST). It was created to generate data that is both geometrically and photometrically realistic and is appropriate to the slitless spectroscopic modes of the ACS.

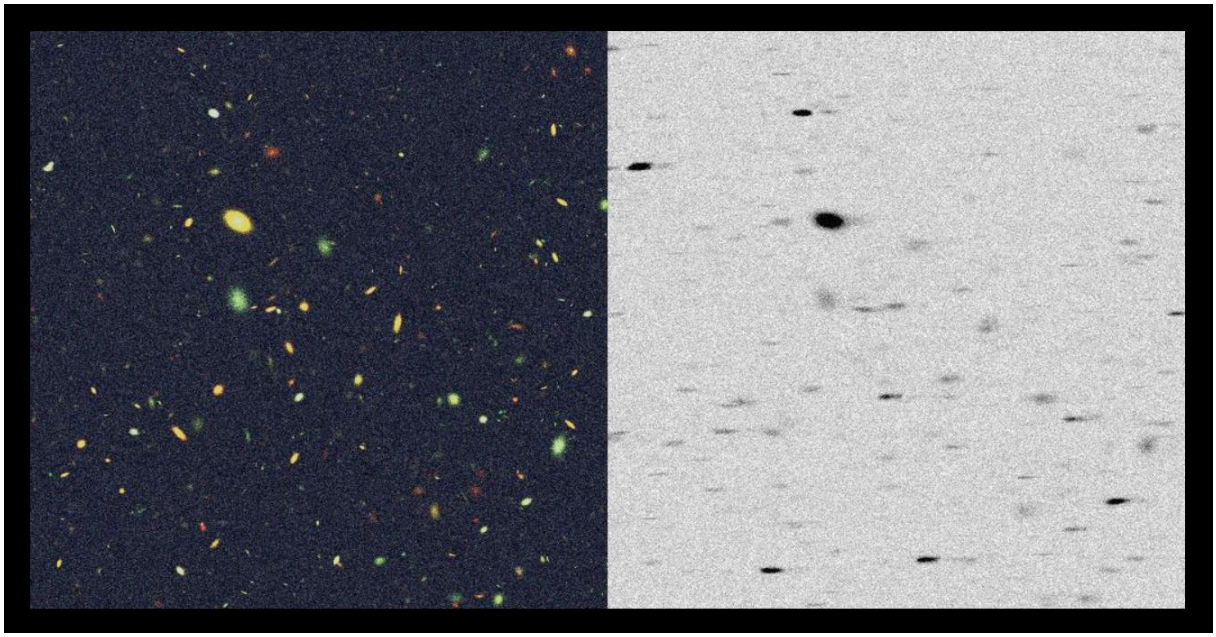

Fig 2.2 Grism observation

In figure 2.2, the left panel shows a colour composite (F435W, F606W, F814W) of a simulated ACS/WFC image of an HDF-N like field. The right panel shows a 1000 sec WFC grism observation simulated with SLIM of the field shown in the colour composite.

2.20 WHEEL

In Unix operating systems, the term wheel refers to a user account with a wheel bit, a system setting that provides additional special system privileges that empower a user to execute restricted commands that ordinary user accounts cannot access

2.21 KERAS

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model

2.22 JUPYTER NOTEBOOK

The Jupyter Notebook is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter.

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

`2.23 WEB CAMERA

A webcam is a video camera that feeds or streams an image or video in real time to or through a computer network, such as the Internet. Webcams are typically small cameras that sit on a desk, attach to a user's monitor, or are built into the hardware. Webcams can be used during a video chat session involving two or more people, with conversations that include live audio and video.

Webcam software enables users to record a video or stream the video on the Internet. As video streaming over the Internet requires much bandwidth, such streams usually use compressed formats. The maximum resolution of a webcam is also lower than most handheld video cameras, as higher resolutions would be reduced during transmission. The lower

resolution enables webcams to be relatively inexpensive compared to most video cameras, but the effect is adequate for video chat sessions.

2.24 RTX 3050 GRAPHICS CARD

The GeForce 30 series is a family of graphics processing units developed by Nvidia, succeeding the GeForce 20 series. The series was announced on September 1, 2020 and started shipping on September 17, 2020.The cards are based on the Ampere architecture and feature hardware-accelerated raytracing (RTX) with Nvidia's second-generation RT cores and third generation Tensor Cores.

The lineup, which is designed to compete with AMD's Radeon RX 6000 series of cards, consist of the entry-level and previously laptop-exclusive RTX 3050 and laptop-exclusive RTX 3050 Ti, mid-range RTX 3060, upper-midrange RTX 3060 Ti, high-end RTX 3070, RTX 3070 Ti and the enthusiast RTX 3080 10 GB, RTX 3080 12 GB, RTX 3080 Ti, RTX 3090, and RTX 3090 Ti. This is the last generation from NVIDIA to have official support for Windows 7 and 8.x as the latest drivers available for this generation require Windows 10.

2.25 HTML

HTML stands for Hyper Text Markup Language. It is used to design web pages using the markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages and markup language defines the text document within the tag that define the structure of web pages. Other technologies besides HTML are generally used to describe a web page’s appearance/presentation (CSS) or functionality/behavior (JavaScript). “Hypertext” refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web. By uploading content to the Internet and linking it to pages created by other people, you become an active participant in the World Wide Web. HTML uses “markup” to annotate text, images, and other content for display in a Web browser.

An HTML element is set off from other text in a document by “tags”, which consist of the element name surrounded by <> and </>. The name of an element inside a tag is case insensitive. That is, it can be written in uppercase, lowercase, or a mixture. For example, the

¡title¿ tag can be written as <title>, </title>, or in any other way. However, the convention and recommended practice is to write tags in lowercase.

2.26 CSS

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

CSS is among the core languages of the open web and is standardized across Web browsers according to W3C specifications. Previously, development of various parts of CSS specification was done synchronously, which allowed versioning of the latest recommendations. You might have heard about CSS1, CSS2.1, CSS3. However, CSS4 has never become
an official version.

From CSS3, the scope of the specification increased significantly and the progress on different CSS modules started to differ so much that it became more effective to develop and release recommendations separately per module. Instead of versioning the CSS specification, W3C now periodically takes a snapshot of the latest stable state of the CSS specification.

2.27 JAVASCRIPT

JavaScript (JS) is a lightweight, interpreted, or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles. Read more about JavaScript.

This section is dedicated to the JavaScript language itself, and not the parts that are specific to Web pages or other host environments. For information about APIs that are specific to Web pages, please see Web APIs and DOM.

The standards for JavaScript are the ECMAScript Language Specification (ECMA-262) and the ECMAScript Internationalization API specification (ECMA-402). The JavaScript documentation throughout MDN is based on the latest draft versions of ECMA-262 and ECMA-402.

Do not confuse JavaScript with the Java programming language. Both “Java” and “JavaScript” are trademarks or registered trademarks of Oracle in the U.S. and other countries. However, the two programming languages have very different syntax, semantics, and use.

CHAPTER 3

RELATED WORKS

Currently, there are different approaches and methodologies to predict the parking slot availability but most of them are either time consuming or expensive. The conventional approach in prediction tends to be inaccurate because irregular pattern in the parking behavior. Below are the detailed explanations of different methods and their drawbacks.

3.1 MULTIPLE LINEAR REGRESSION

This method is performed on a dataset to predict the response variable based on a predictor variable or used to study the relationship between a response and predictor variable, for example, student test scores compared to demographic information such as income, education of parents, etc.

3.2 K-NEAREST NEIGHBORS

Like the classification method with the same name above, this prediction method divides a training dataset into groups of k observations using a Euclidean Distance measure to determine similarity between “neighbors”. These groups are used to predict the value of the response for each member of the validation set.

3.3 REGRESSION TREE

A Regression tree may be considered a variant of a decision tree, designed to approximate real-valued functions instead of being used for classification methods. As with all regression techniques, XLMiner assumes the existence of a single output (response) variable and one or more input (predictor) variables. The output variable is numerical. The general regression tree building methodology allows input variables to be a mixture of continuous and categorical variables. A decision tree is generated when each decision node in the tree contains a test on some input variable's value. The terminal nodes of the tree contain the predicted output variable values.

3.4 NEURAL NETWORK

Artificial neural networks are based on the operation and structure of the human brain. These networks process one record at a time and "learn" by comparing their prediction of the record (which as the beginning is largely arbitrary) with the known actual value of the response variable. Errors from the initial prediction of the first records are fed back into the network and used to modify the networks algorithm the second time around. This continues for many, many iterations.

PREDICTION METHODS SUMMARY

- A technique performed on a database either to predict the response variable value based on a predictor variable or to study the relationship between the response variable and the predictor variables.
- XLMiner supports the use of four prediction methods: multiple linear regression, k-nearest neighbors, regression tree, and neural network.

CHAPTER 4

DESIGN AND IMPLEMENTATION

This chapter describes the implementation of ANPR system, and the functionalities implemented in phase 1 also.

The proposed system is capable of detecting the number plates, detecting the numbers and store in a database. The system has both cloud and local database. For faster access and processing the local database can be used and for network-based activities cloud-based database can be used. The detected number plates are displayed on a web dashboard which is accessible for verified admins only. The admin page consists of an authentication step to access the web dashboard so that the information about vehicles inside the organization or building are safe.

Outsiders or staffs in the case of an organization, can see the details like parking occupancy prediction with a web dashboard specially designed for users. This dashboard will be made available on the network so that outsiders or staffs can visit this web page to see all these data for better parking slot availability prediction. This ensures a better user experience and guarantees that the data is safe and can be later transferred to the college or organization database.

The parking prediction is done using the ARIMA model algorithm. This model can achieve a greater accuracy. The predicted values are updated to the cloud database. The online and offline database can sync over a certain period of time. With huge amount of data of vehicles and parking slots availability, the prediction model is trained so far.

The system mainly consists of a camera and the computer system. The main camera module has to be placed in front of the entry gate pointing the number plate of entering vehicles. The camera can capture the live visuals of everything that passes in front of it. This system requires a high-end device for processing since live visual detection is used.

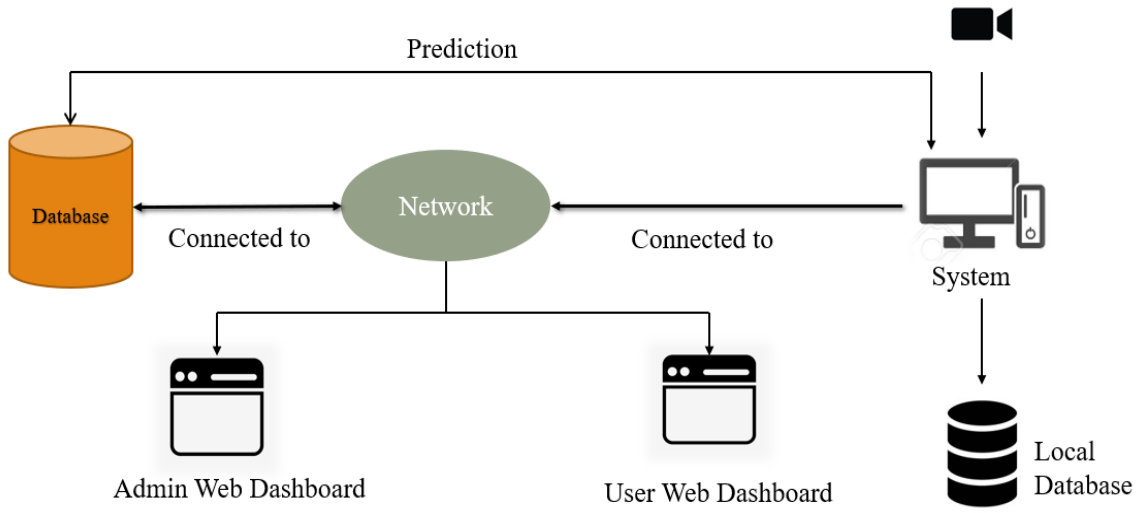

Fig. 4.1 Architecture of the proposed system

The TensorFlow and EasyOCR will be responsible for extracting the number plate and number from the visuals and ARIMA algorithm will be responsible for parking prediction with previous data.

4.1 DETECTING THE NUMBER PLATE AND EXTRACTING NUMBERS

Phase 1 of the project included the detection of number plate and recognizing the numbers from the detected image of number plate.

4.1.1 DETECTING VEHICLE NUMBER PLATE

The number plate detection consists of several steps. First step is to detect the ROI (Region of Interest). A camera is used to capture the live images of vehicle. The live visuals are passed to the computer system. The system consists of different software modules. They are training module and testing module. The train module can be used to train the system to detect the number plates and to test the detection system, testing module can be used. The training module has two phases, data collection and training. Previously defined data sets are used for the training and testing phase. Each data set item consists of an image and a .xml file. The position of number plate of corresponding image will be in the .xml file. Example is shown in fig 4.1 and 4.2

Fig 4.2a Image of a car (img200.png)

```
<annotation>
    <folder>images</folder>
    <filename>Cars201.png</filename>
    <size>
        <width>590</width>
        <height>350</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>licence</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <occluded>0</occluded>
        <difficult>0</difficult>
        <bndbox>
            <xmin>216</xmin>
            <ymin>171</ymin>
            <xmax>367</xmax>
            <ymax>208</ymax>
        </bndbox>
    </object>
</annotation>
```

fig 4.2b xml file of the image img200.png

The projects involved three types of number plates. Black text on yellow plate, black text on white number plate, white text on green number plate. Detecting these types of number plates is a challenging one. So, the detector should be capable of detecting all types of number

plates. ANPR uses optical character recognition (OCR) on images taken by cameras. When Dutch vehicle registration plates switched to a different style in 2002, one of the changes made was to the font, introducing small gaps in some letters (such as P and R) to make them more distinct and therefore more legible to such systems. Some license plate arrangements use variations in font sizes and positioning—ANPR systems must be able to cope with such differences in order to be truly effective. More complicated systems can cope with international variants, though many programs are individually tailored to each country.

The cameras used can be existing road-rule enforcement or closed-circuit television cameras, as well as mobile units, which are usually attached to vehicles. Some systems use infrared cameras to take a clearer image of the plates. For detecting the number plates, high quality camera is required. The live visuals are passed to the system for further processing like detecting the number plate, recognizing the character on the number plate, local and cloud-based storage, web view, etc.

4.1.2 RGB TO GRAYSCALE CONVERSION

The live visuals from the system are used as the input for this process. These visuals are processed with the help of graphics card and graphics drivers. Whenever a vehicle is detected, a ROI (Region of Interest) is being identified and extracted using the grid methodology.

The ROI is identified by converting the image from camera to a grey formatted image. The python

From the extracted ROI, further segmentation procedure is done. The image conversion form RGB to gray can be done with following methods.

- Convert an Image to Grayscale in Python Using the image.convert() Method of the pillow Library
- Convert an Image to Grayscale in Python Using the color.rgb2gray() Method of the scikit-image Module
- Convert an Image to Grayscale in Python Using the cv2.imread() Method of the OpenCV Library

- Convert an Image to Grayscale in Python Using the Conversion Formula and the Matplotlib Library

Fig 4.3 Raw Image in RGB format

This image in RGB format can be converted into grayscale mode to detect the number plate. The number plate is identified by finding the white area in rectangular shape from the gray mode image. After converting to gray scale mode, the image will look like fig. 5.4

Fig 4.4 Gray scale image of fig 4.3

4.1.3 IDENTIFYING THE ROI

ROI detection has been studied for many years. Most algorithms use either feature-based or object-based approaches. Feature-based methods find pixels that share significant optical features with the target and aggregate them to form ROIs. These methods can capture most of the target pixels on the basis of optical feature similarity. However, not all target pixels have strong optical features, so the detected ROI usually fails to encompass the entire target. In addition, feature-based methods cannot distinguish between targets, which can cause confusion in subsequent stages of processing.

The HROID method includes five main processing steps (see Fig 4.5 and 4.6). First, use prior information to divide the targets into groups and to determine the lowest detection resolution for each group. We then down sample the image to the required resolution using wavelet transformation. This is followed by detecting the pixels with strong features for each group, employing morphological processing to improve the chances that each region, which is called a region-of-candidates (ROC), corresponds to only one target. We then combine the ROCs for all groups by a voting procedure that maximizes the number of pixels in these regions. Finally, we extend the surviving ROCs using prior information so that target pixels lacking strong

optical features are included within the appropriate ROC. The extended ROCs are deemed to be ROIs.

Fig 4.5 Plotted number plate

Fig 4.6 ROI – Region of Interest

4.1.4 EXTRACTING THE CHARACTERS

The ROI (number plate) from the above step is segmented by character by character by grid methodology and each character in grid is recognized using OCR technology.

An OCR program extracts and repurposes data from scanned documents, camera images and image-only pdfs. OCR software singles out letters on the image, puts them into words and

then puts the words into sentences, thus enabling access to and editing of the original content. The main benefit of optical character recognition (OCR) technology is that it simplifies the data-entry process by creating effortless text searches, editing and storage. OCR allows businesses and individuals to store files on their computers, laptops and other devices, ensuring constant access to all documentation.

4.1.5 LOCAL AND CLOUD STORAGE

The extracted number plate from the ROI is to be saved in two locations. One is local storage which is for the fast processing of data and cloud storage for security of data and for web view. The data stored in both local and cloud are used for the next phase, that is prediction of parking occupancy.

The storage is done on two types of files. One is a .txt file which keeps track on the entry and exit details of each and every vehicle that are detected by the camera. The text file gets updated each and every time a vehicle number plate is being detected. The text file consists of serial number, vehicle number plate, entry time, exit time, date and owner type. The other one is an .csv file for the prediction phase. The csv file gets updated every 15 minutes. The csv file contains the total number of parking slots, total number of parking slots available, date and time.

For cloud storage, firebase is used. The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, real-time events continue to fire, giving the end user a responsive experience.

Firestore is a NoSQL document database built for automatic scaling, high performance, and ease of application development. While the Firestore interface has many of the same features as traditional databases, as a NoSQL database it differs from them in the way it describes relationships between data objects.

The detected number plate, time, etc. are uploaded and stored on firebase using firebase-admin and firebase module of python. Firebase provides security and efficient data storage and retrieval with certain commands.

4.2 PREDICTION

Phase 2 of the project included the prediction of parking occupancy over time and updating these predicted data to cloud and store locally.

4.2.1 DATA SET

We identified being highly relevant in predicting parking availability are day, time, event, distance, parking price, etc. For day, we set an indicator function for each day in the week. For time, we discretize time into 24 intervals and set an indicator function for each time interval (ta, tb) (inclusive on ta and exclusive on tb). The single distance feature is a distance in miles between the lot location and the cluster centroid to which it belongs (calculated by K-means we will mention in next section). Three event features - sports, concert, and race - are also represented by indicator functions. We mark an event feature as 1 if there is an event either happening at the time of interest or will happen within four hours (to account for people arriving early to an event). For example, the time stamp corresponds to Monday, 08:00 am. Also suppose that a major race is starting at 10:00 am (within four hours of 08:00 am) that same day. Moreover, we check the PCA on timeseries to discover occupancy feature in another space. And we also input the first 3 PCA features because it already explains over 95% variance. Thus, a feature vector for time t can be expressed as vt = T where i is the time step lag (we pick previous 24 hours occupancies, i.e., i = 24), P C1 stands for first principal component, similarly for P C2, P C3.

4.2.2 MODEL SELECTION

ARIMA

We implement ARIMA(2, 0, 1) × (1, 1, 0)24 model to investigate the relationship between prediction error and aggregation level. Here seasonal factor is set to be 24 due to the daily period. We tuned the parameters by picking the lowest AIC value among different p, d, q, P, D, Q combinations. And performance of MAPE is computed in prediction error curve is shown in Figure 3a. Originally the prediction error goes over 15%. But it drops significantly after aggregating more than 100 spots, the aggregation effect become flatten out when adding over 160 spots.

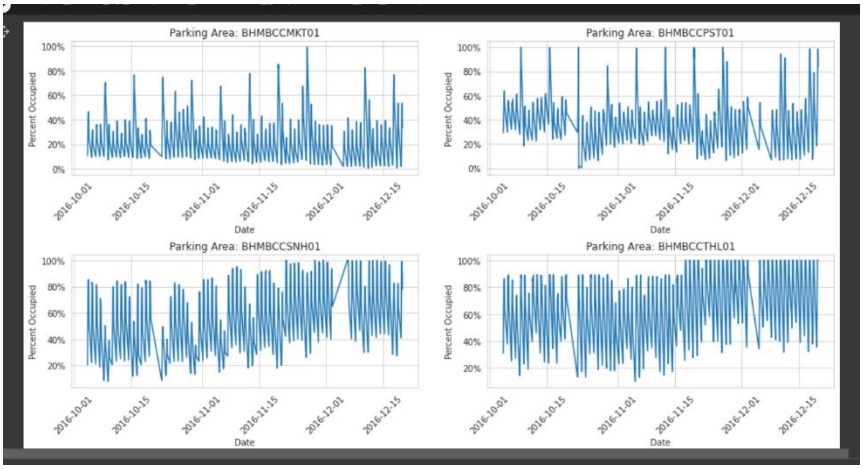

Fig 4.7 Arima based prediction result graph

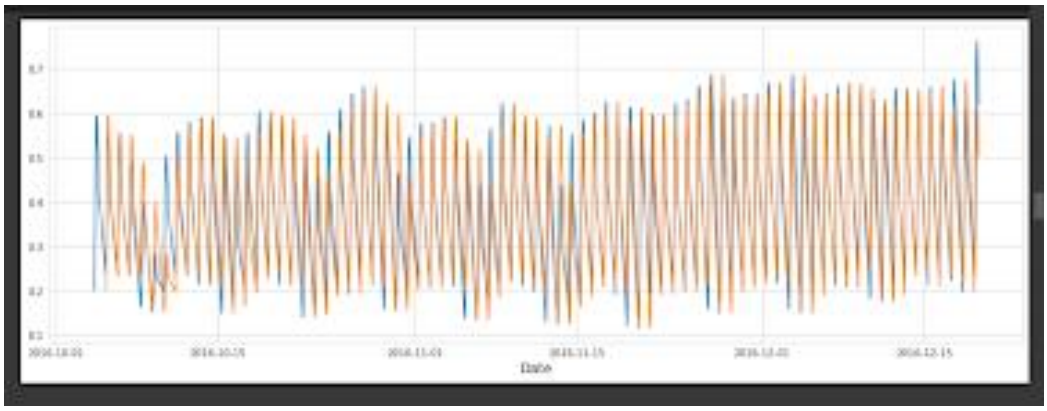

Fig 4.8 Look for seasonality and test for stationarity

LINEAR REGRESSION (OLS)

Given y(t) is the time series, we input previous time step value together with the dummy variable to indicate current time is allocated at what hour in the day, and what day of the week to fit the value at current time.

The y(t) can be expressed as

$$y(t) = c_0 + \sum_{i=1}^{24} \beta_i \mathbf{1}\{HoD(t) = i\} + \sum_{j=1}^{5} \theta_j \mathbf{1}\{DoW(t) = j\} + \alpha y(t-1) + \epsilon(t) \quad (2)$$

where HoD() is a function takes in time t and return the hour of the day (from 1 to 24), DoW() is a function return the day of week (from Monday to Friday). β, θ are dummy variables, α, c0 are corresponding parameters. We fit the model by getting least square error of (t) The R2 value for Linear Regression Model is over 92%. After aggregation of 120 spots, it gives MAPE close to 8%.

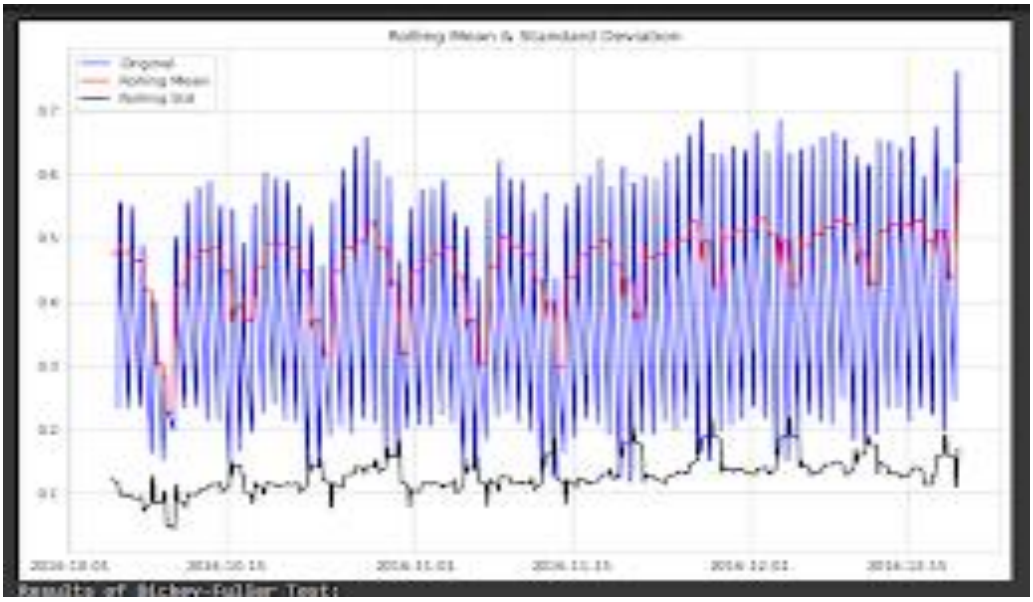

Fig 4.9 Parking mean and standard deviation

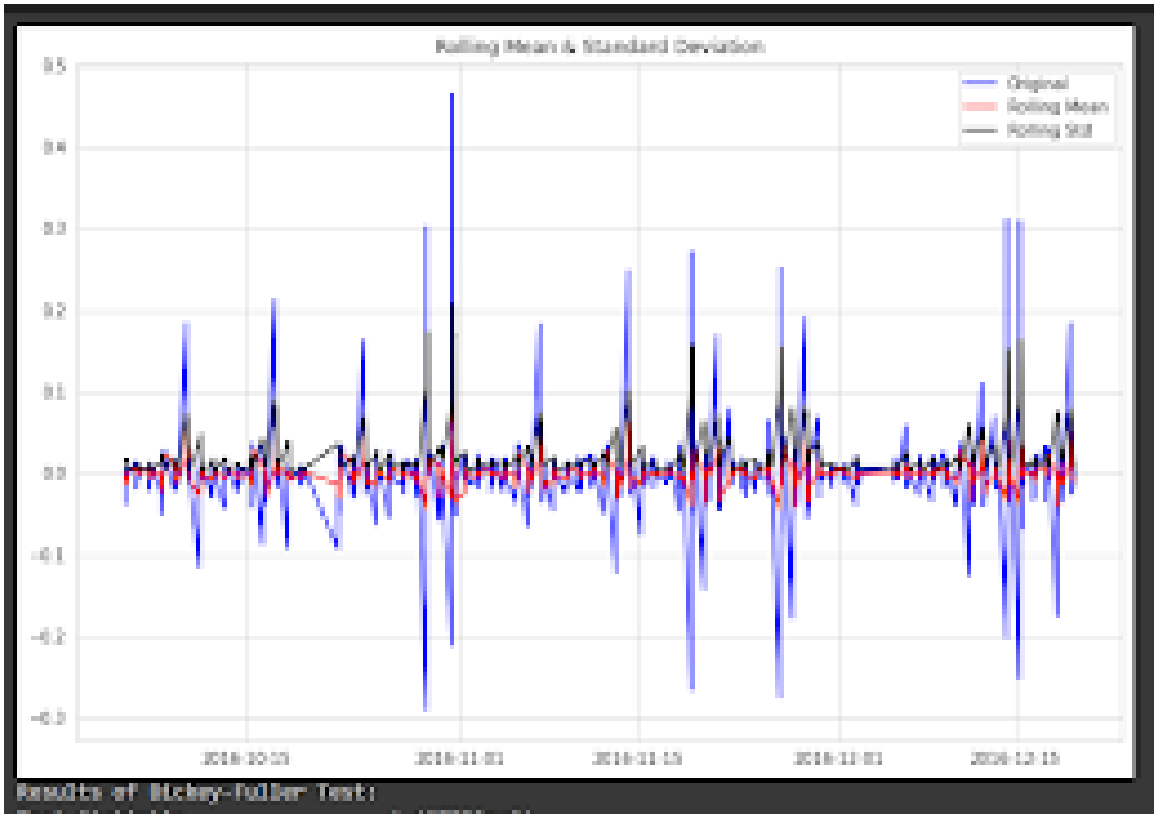

Fig 4.10 Result of Dickey-Fuller Test

SUPPORT VECTOR REGRESSION

Let a vector xk(t) holds all sample data from y(t − k + 1), y(t − k + 2), . . . , y(t), where k is the vector length (also known as time lags). SVR builds a nonlinear map between y(t) and xk(t − 1) as follows:

$$y(t+1) = w^T \phi(\mathbf{x}_k(t)) + b + \epsilon(t)$$

The kernel function φ is chosen particularly depending on problem. Here the kernel function is radial basis function. We pick previous 24 sample observations to generate current prediction value, which is y(t) relates to y(t − 24), y(t − 23), . . . , y(t−1). It gives best performance when aggregating about 100 spots, generating MAPE with value of 7.21% for testing.

CHAPTER 5

RESULT AND EVALUATION

In this chapter we discuss the output obtained by the ANPR system against some test cases.

5.1 NUMBER PLATE DETECTION

The project is tested on various pictures and was successful with certain pictures that were available under certain dimension frame. The image was successfully enhancing the image quality and was converting the picture to gray scale image. The project extracts the number plate from the car image and display it separately.

Fig 5.1 image captured by system

Fig 5.2 Grayscale image

Fig 5.1 shows the captured image by the system and this image is then converted to grayscale image for better processing and fast recognition of characters from the image.

Fig 5.2 is the corresponding grayscale image of fig 5.1

Fig 5.3 Detecting ROI

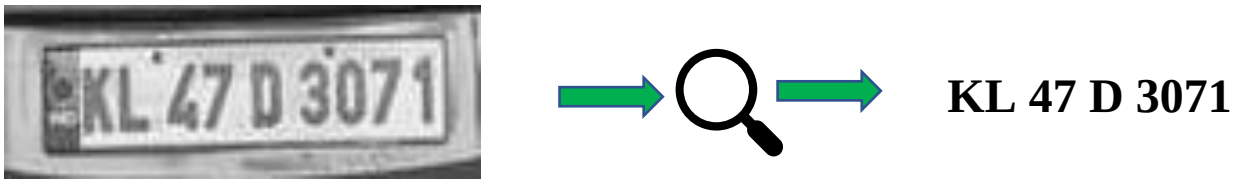

Fig 5.4 Character extraction from ROI

From the grayscale image (fig 5.2) the region of interest is identified and from the ROI, the characters are extracted by segmentation and recognition processing. Fig 5.3 represents the identified ROI of the car and fig 5.4 represent the character recognized from the ROI.

These extracted number plates are then furtherly used for the prediction phase.

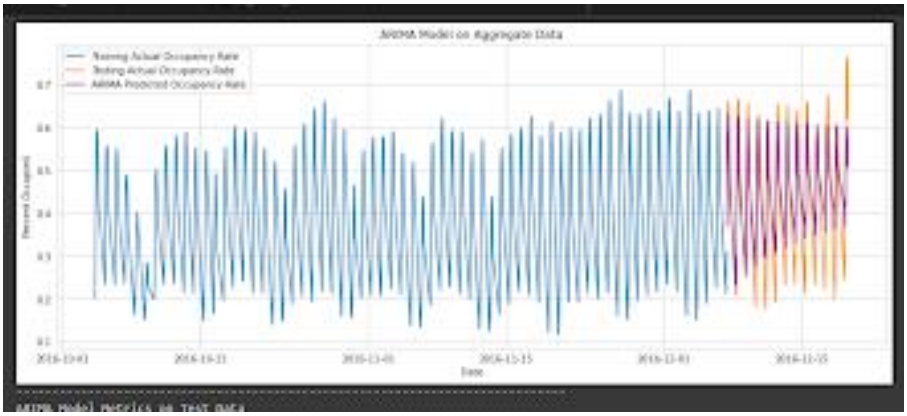

Fig 5.5 ARIMA model test Result

Fig 5.5 represents the predicted parking occupancy by analyzing the given data set. Using the data set, a pattern is obtained and according to the pattern a line graph is plotted for the future.

5.2 NUMBER PLATE DETECTION AND RECOGNITION RESULTS

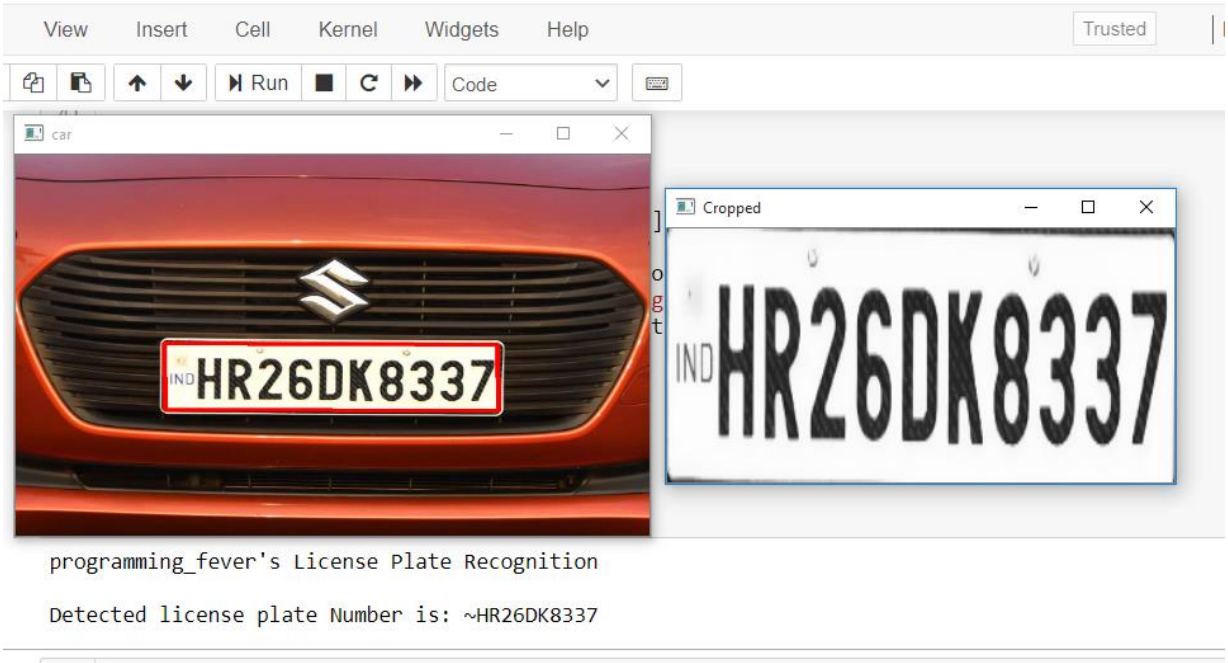

Fig 5.6 Test result 1

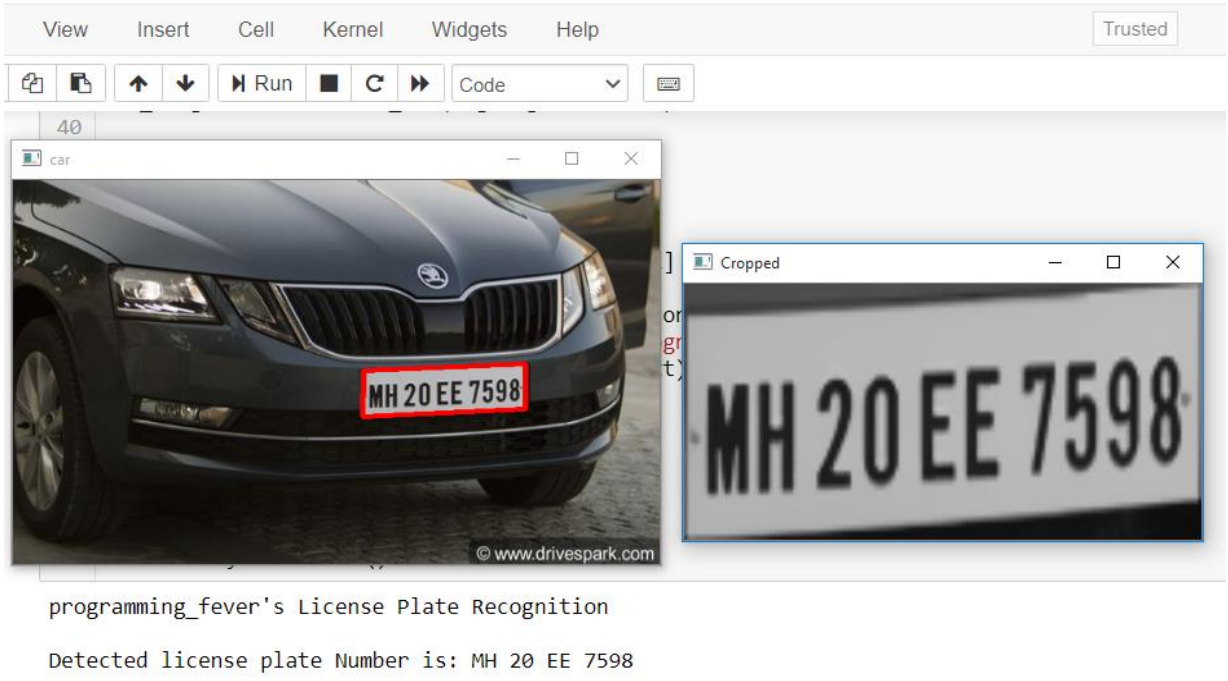

Fig 5.7 Test result 2

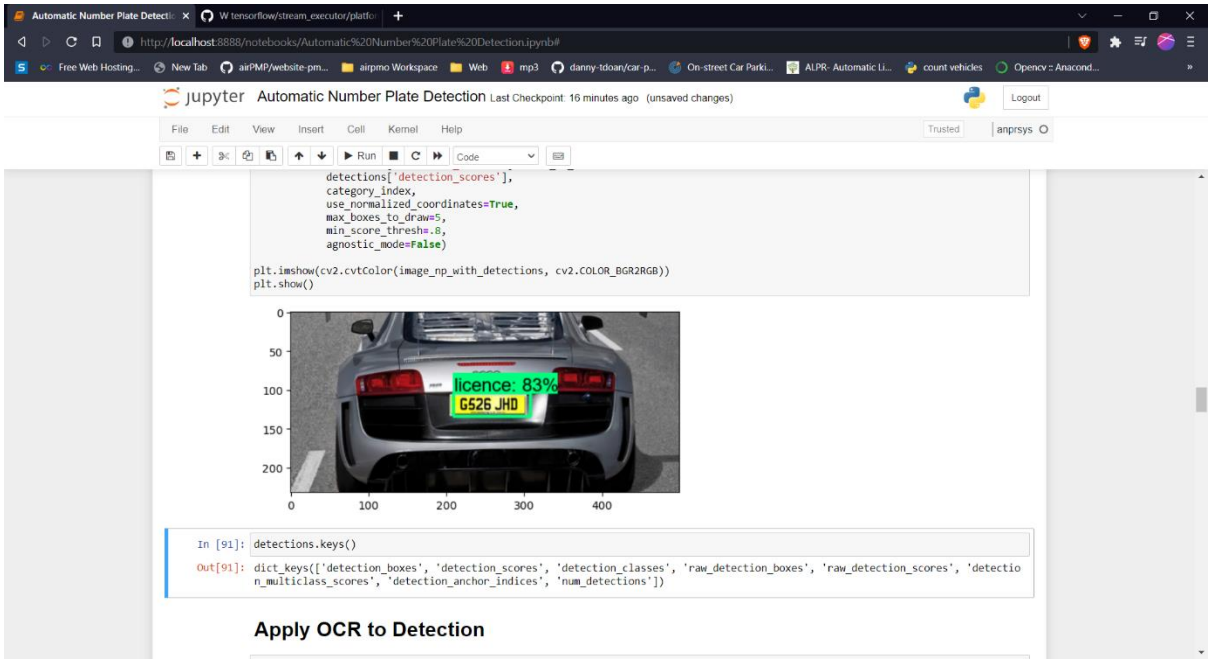

Fig 5.8 Test result 3

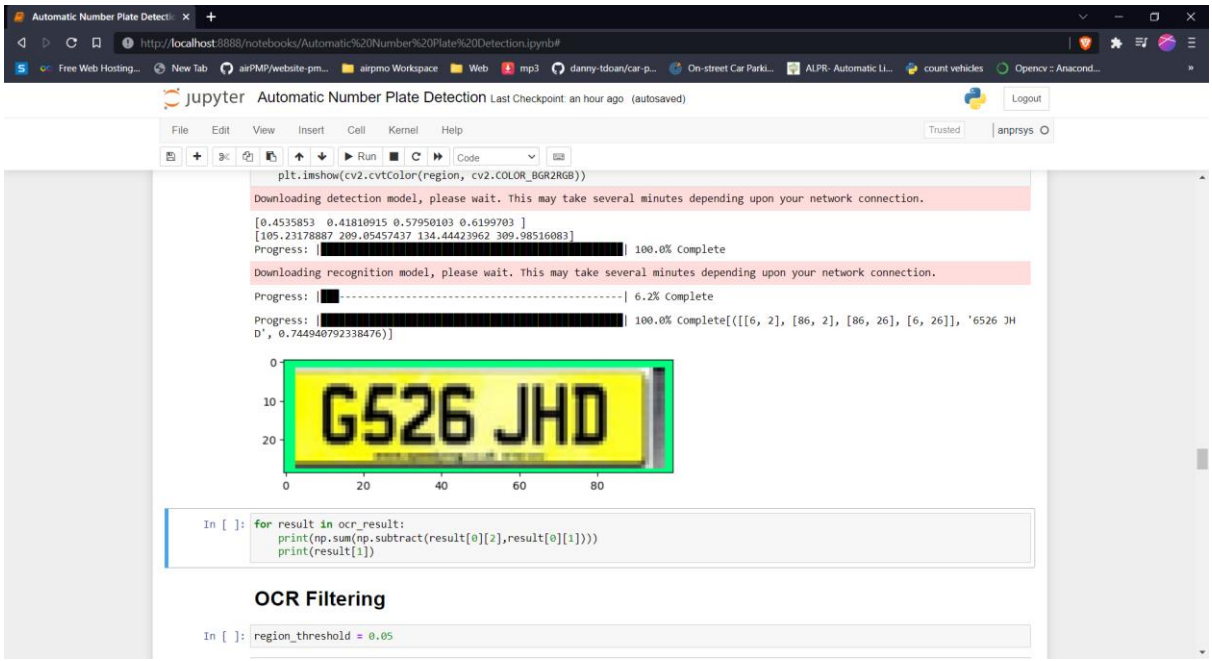

Fig 5.9 Test result 4

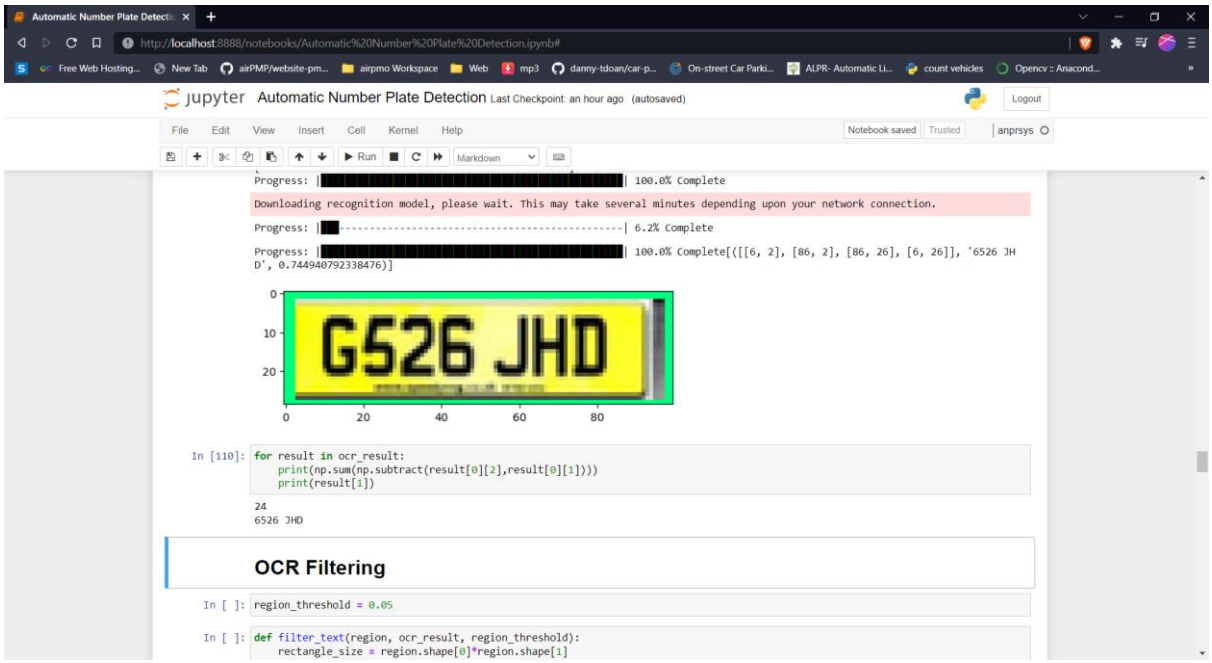

Fig 5.10 Test result 5

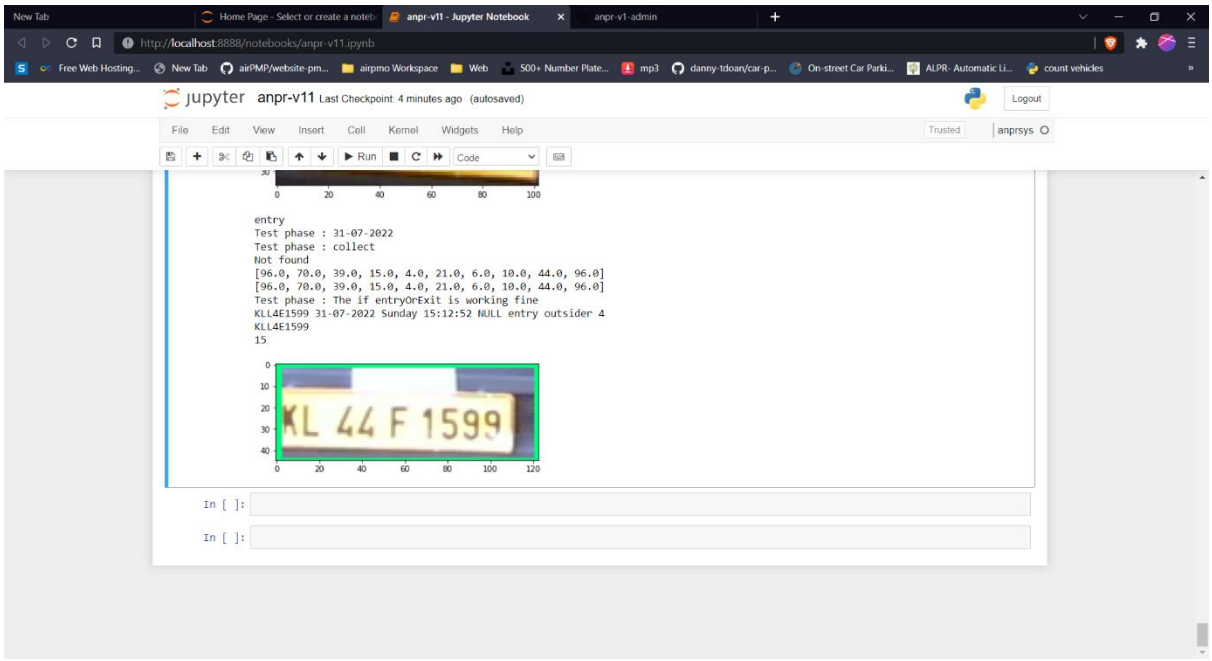

Fig 5.11 Test result 6

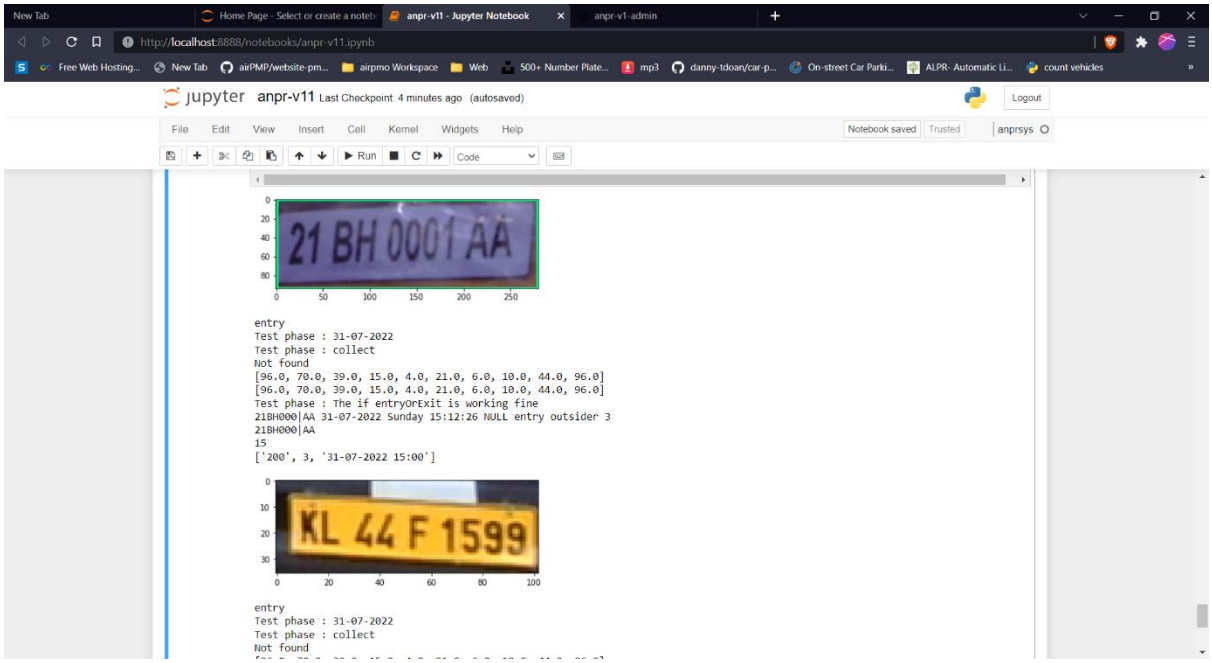

Fig 5.12 Test result 7

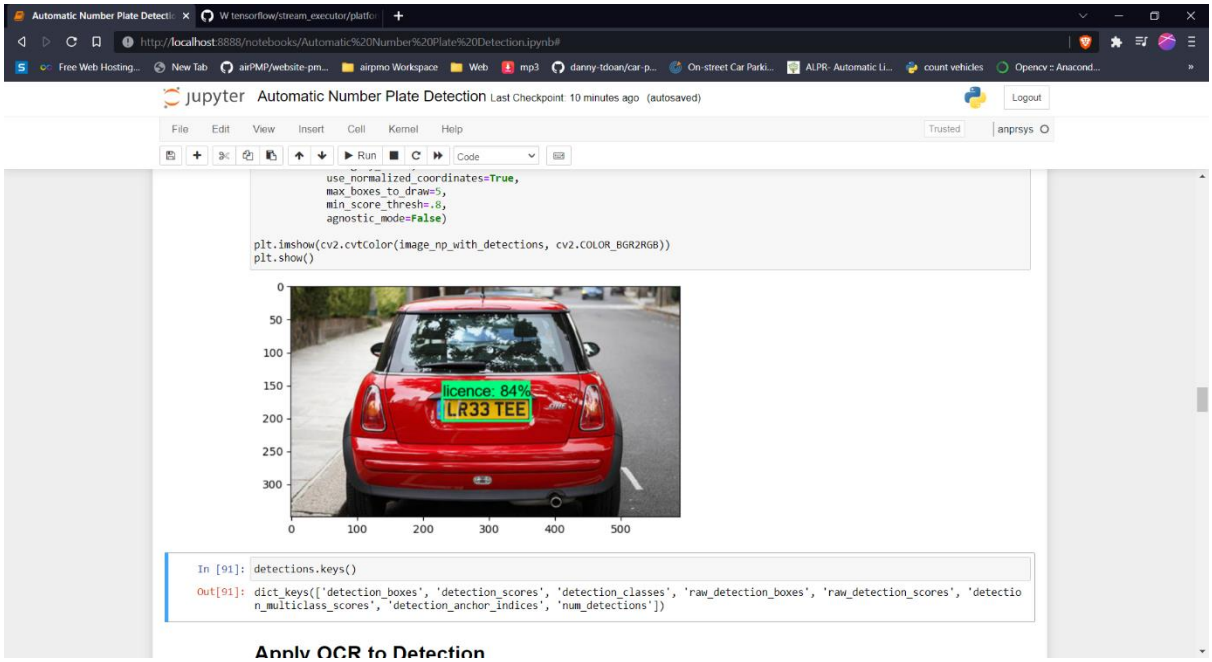

Fig 5.13 Test result 8

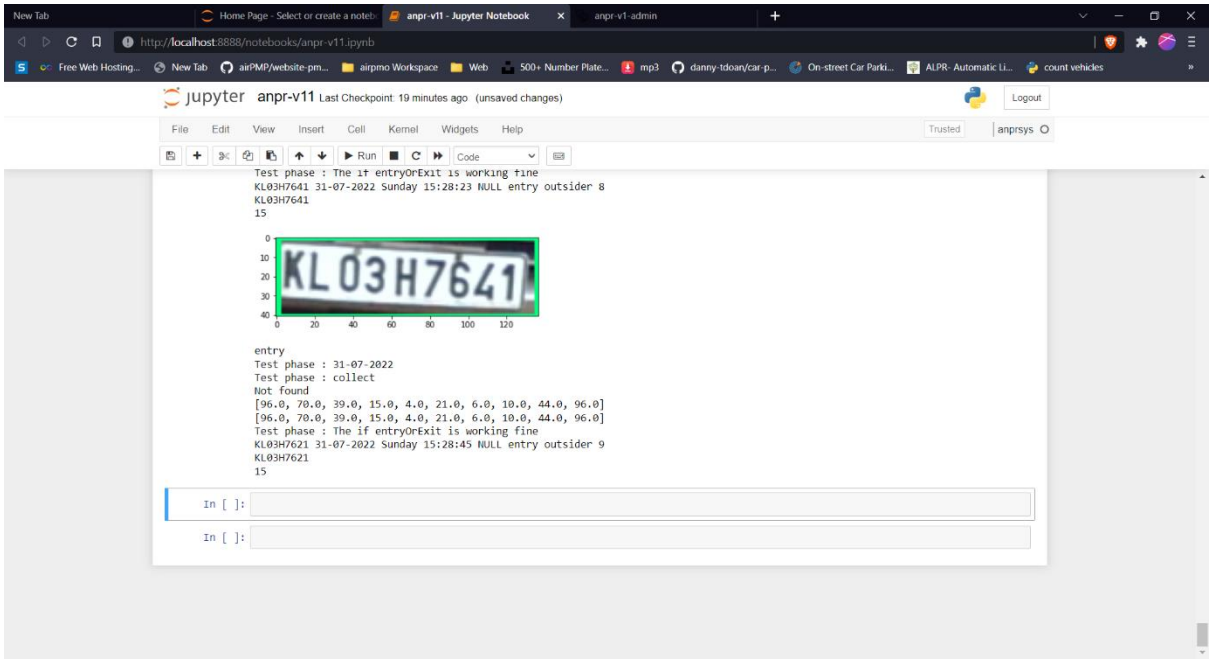

Fig 5.14 Test result 9

5.3 USER PAGE RESULTS

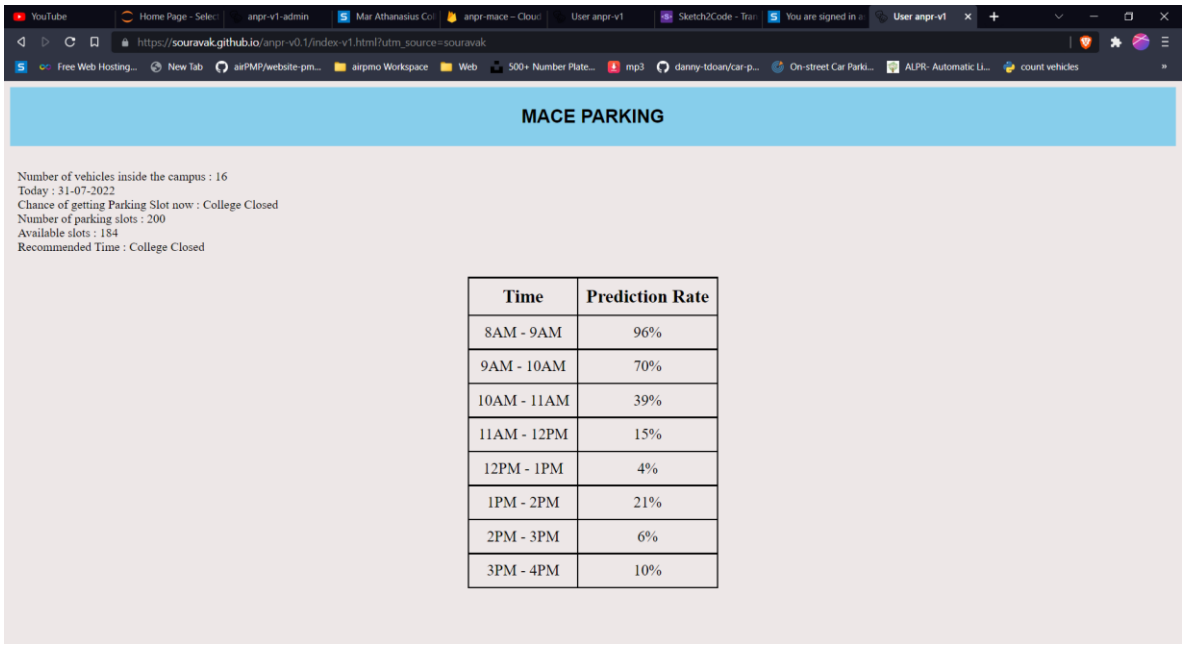

Fig 5.15 User page View

5.4 ADMIN PAGE RESULTS

Fig 5.16 Admin page View

CHAPTER 6
FUTURE SCOPE

Presently the project is built for the college. The college gate security can access the webpage and monitor the entry and exit of vehicles inside the campus. But in future this project can be used in lots of different scenarios.

The future scope is that the automatic vehicle recognition system plays a major role in detecting threats to defense. Also it can improve the security related to the women's as they can easily detect the number plate before using cab or other services. The system robustness can be increase if bright and sharp camera is used. Government should take some interest in developing this system as this system is money-saving and eco- friendly, if applied effectively in various areas.

This project can be used to collect tolls from vehicles without interfering the journey of passengers. And also, this project can be implemented in large scale shopping malls, colleges, public parking slots, etc. By this people can know the parking availability of selected parking slots, day and time. Thus, reducing the parking as well as the traffic congestion. Moreover, this model can be utilized to justify a detailed understanding of how aggregate occupancy patterns are formed and verified on higher resolution data. Also, more features can be incorporated into our current prediction models, such as humidity or weather, population density in corresponding region, etc.

CHAPTER 7

CONCLUSION

With this project, an effective method of identification of vehicle number plate is proposed which is less time consuming and applied to various types of pictures. Edges could be recognized here through the use of the SOBEL edge detection method, and also the holes are filled but with far less than 8 pixels. To retrieve the vehicle's number plate, we delete attached parts / pieces and under 1000 pixels. Our proposed set of computer instructions is mainly based on Indian car number plate scheme, the accuracy of extracting the number plate for low quiet mood can be increased, as well as we can detect the number plate that has different font size and also different font type.

REFERENCES

[1] S.A.Daramola EmmanuelAdetiba,AnthonyUwakhonyeAdoghe, JokeA.Badejo(2011):Automatic vehicle identification system using license plate

[2] • H. K. Sulehria, Ye Zhang and D.Irfan (2007):MathematicalMorphology Methodology forExtraction ofVehicleNumber Plates, International Journal of Computers Issue 3, Volume 1.

[3] S.H. Mohades Kasaei, S.M. Mohades Kasaei and S.A.Monadjemi(2009): A Novel Morphological Method forDetection and Recognition of Vehicle License Plates, American Journal of Applied Sciences 6 (12)

[4] • A. O. Khalifa, S. Khan, R. Islam and Suleiman (2007): Malaysian Vehicle License PlateRecognition, The International Arab Journal of Information Technology, Vol. 4, No. 4.

[5] M. Welling. Support vector regression.

[6] D. Basak, S. Pal, and D. C. Patranabis, “Support vector regression,” Neural Information Processing-Letters and Reviews, vol. 11, no. 10, pp. 203–224, 2007.

[7] P. Auer, H. Burgsteiner, and W. Maass, “A learning rule for very simple universal approximators consisting of a single layer of perceptrons,” Neural Networks, vol. 21, no. 5, pp. 786–795, 2008.

[8] M. Richtel, “Now, to find a parking spot, drivers look on their phones,” NYTimes Article, May 2011.