FINAL REPORT

TCS REMOTE INTERNSHIP PROGRAM 2018

License Plate Recognition System using Python and OpenCV

Submitted by

Vishal Polley (CT20172176247)
Abhay Pandey (DT20173820470)

INSTITUTE OF ENGINEERING AND TECHNOLOGY, LUCKNOW

Prof. Manik Chandra
(Faculty Advisor)

Mr. Deepanshu Kukreja
(Mentor)

TATA CONSULTANCY SERVICES

TABLE OF CONTENT

INTRODUCTION

License Plate Recognition Systems use the concept of optical character recognition to read the characters on a vehicle license plate. In other words, LPR takes the image of a vehicle as the input and outputs the characters written on its license plate.

LPR also called ALPR (Automatic License Plate Recognition) has 3 major stages.

1. *License Plate Detection:* This is the first and probably the most important stage of the system. It is at this stage that the position of the license plate is determined. The input at this stage is an image of the vehicle and the output is the license plate.
2. *Character Segmentation:* It's at this stage the characters on the license plate are mapped out and segmented into individual images.
3. *Character Recognition:* This is where we wrap things up. The characters earlier segmented are identified here. We have used machine learning for this.

TECHNOLOGIES USED

OS - *Ubuntu 16.04 :*

Ubuntu is a free and open source operating system and linux distribution based on debian .

is the most popular operating system for the cloud .There is python installed in it which makes our work more easier .

Python - *3.5 or Up :*

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. It's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The interpreter and the extensive standard library are freely available in source or binary form .

IDE - *Atom :*

Atom is a desktop application built using web technologies. It is free and open source text and source code editor for linux .

It is based on Electron ,a framework that enables cross-platform desktop applications using

Chromium and Node.js . It is written in CoffeeScript and Less . It can also be used as an Integrated Development Environment (IDE).

Database - *SQLite3 :*

SQLite is a relational database management system contained in a C programming library. In contrast to many other database management systems, SQLite is not a client-server database engine.It is embedded into the end program.
SQLite is ACID-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity.

Front End - *Tkinter :*

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter outputs the fastest and easiest way to create the GUI applications.

Back End - *Python :*

Python is an interpreted high-level programming language.
It provides constructs that enable clear programming on both small and large scales .
It is meant to be an easily readable language. Writing programs in Python takes less time than in some other languages.

MODULE'S INFORMATION

Python Modules - *scikit-learn, scikit-image, OpenCV, scipy, Pillow, numpy, matplotlib*

We have build our project over isolated python virtual environment. This makes it easy to manage our project's dependencies and packages. We have used virtualenv package to create a virtual environment, which can be installed and activated by running these commands -

```
cd lpr/
virtualenv -p python3 env
source env/bin/activate
```

We have included a requirements file named as ***requirements.txt*** inside our project folder. To install all the modules and dependencies required for the project run the following command in terminal as -

```
pip install -r requirements.txt
```

1. ***scikit-learn*** *:* scikit-learn is a Python module for machine learning built on top of SciPy. It provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

2. ***scikit-image*** *:* For performing *Image Processing* we have used scikit-image. It's a Python package for image processing.

3. ***scipy*** *:* SciPy is a free and open-source Python library used for scientific computing and technical computing. It contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

4. ***OpenCV*** *: OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.*

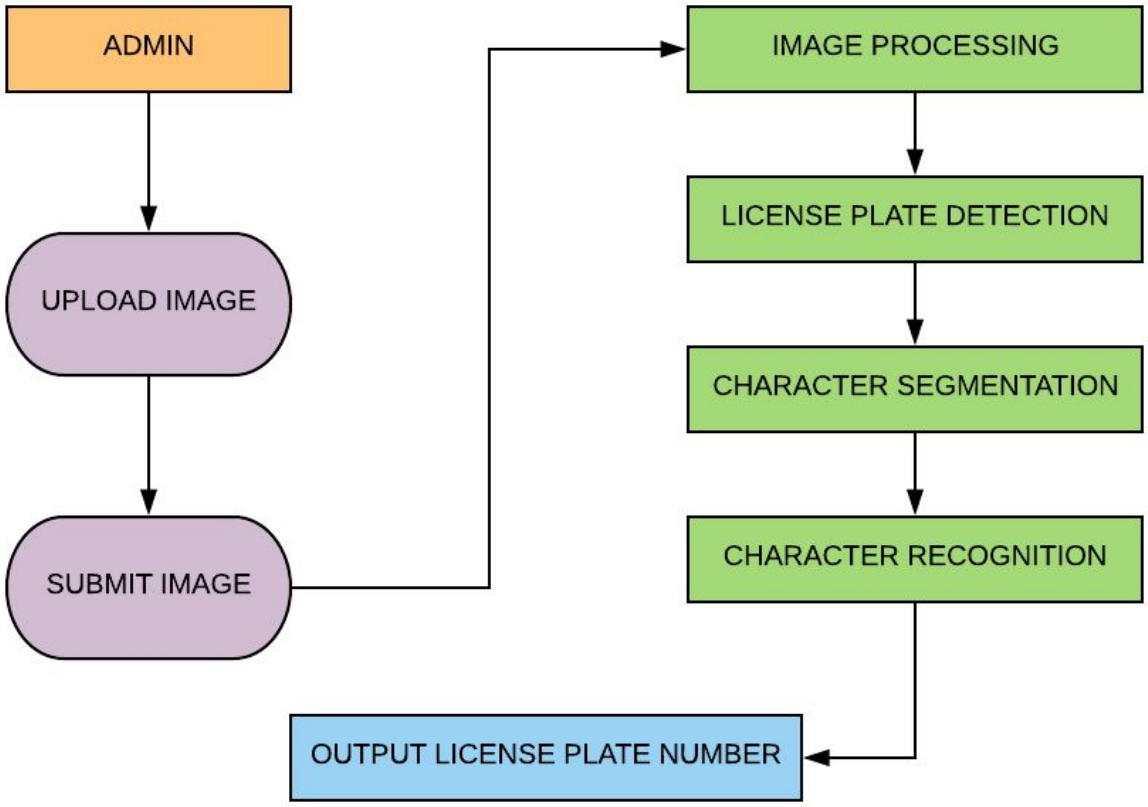
5. ***Pillow*** *:* Python Imaging Library (abbreviated as *PIL*) is a free library for the Python programming language that adds support

for opening, manipulating, and saving many different image file formats.

6. ***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.

7. ***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+.

DATA FLOW DIAGRAM



TEST CASES

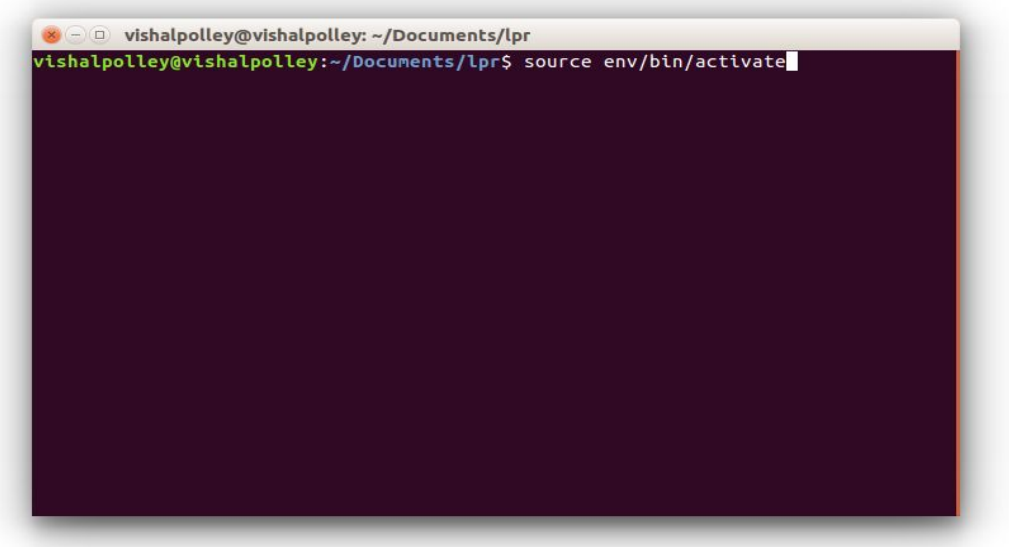
INPUT -

OUTPUT -

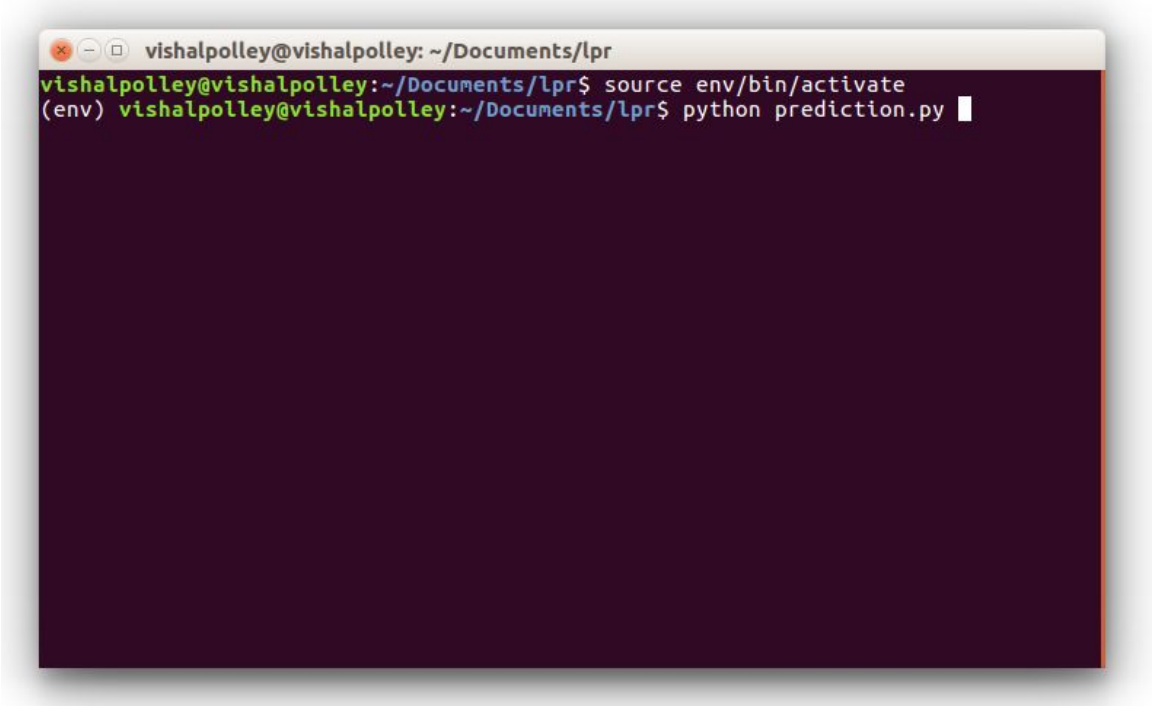
DEMONSTRATION AND SCREENSHOTS

In the first step, open terminal (Python Bash) and activate the virtualenv (Python virtual environment) by running the following command inside the project folder -

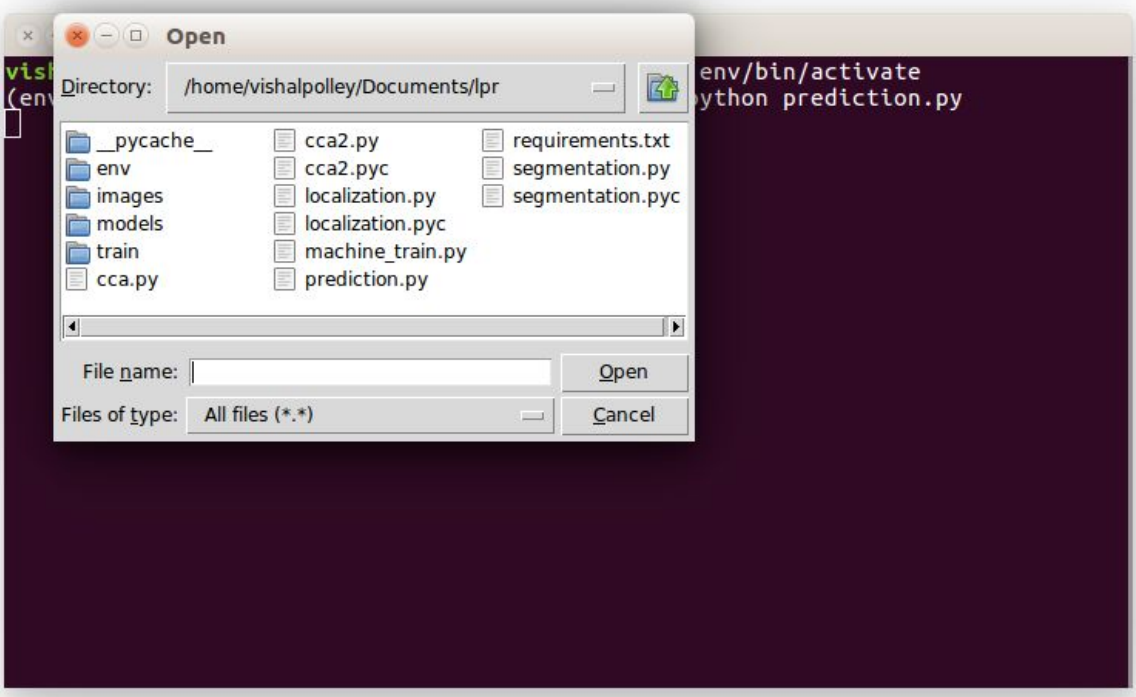
```
source env/bin/activate
```



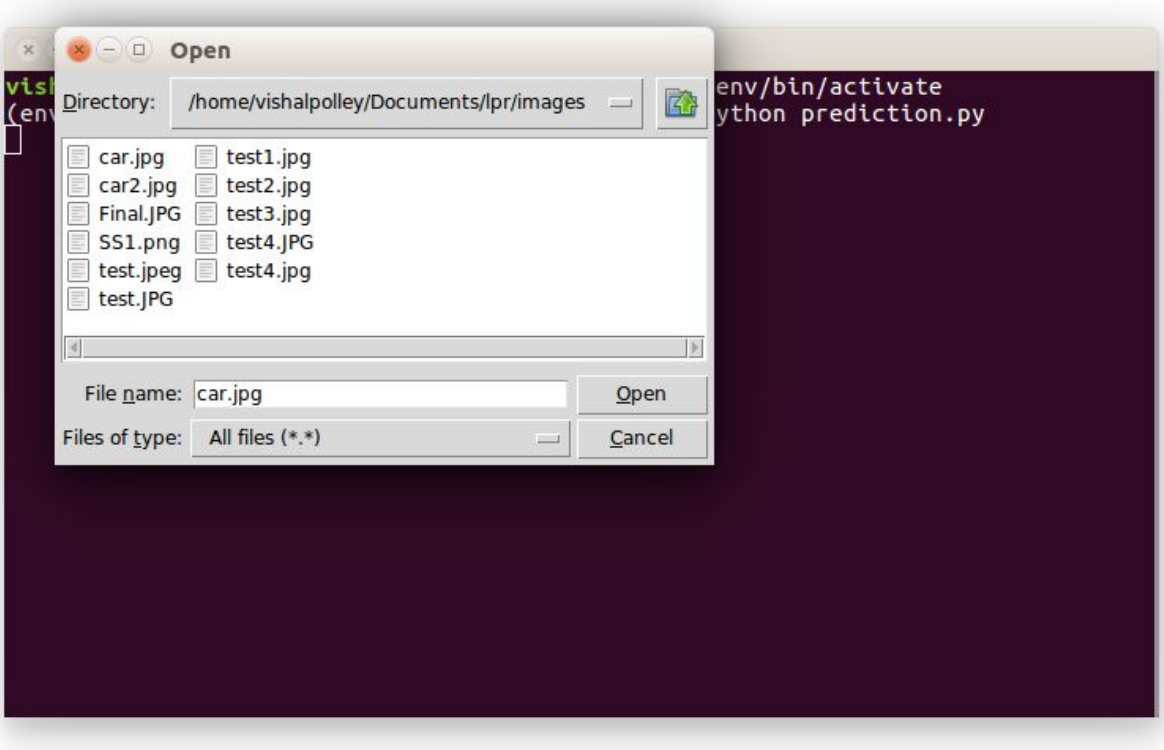
Now run the python project by executing python script named ***prediction.py*** in the terminal (Python Bash)



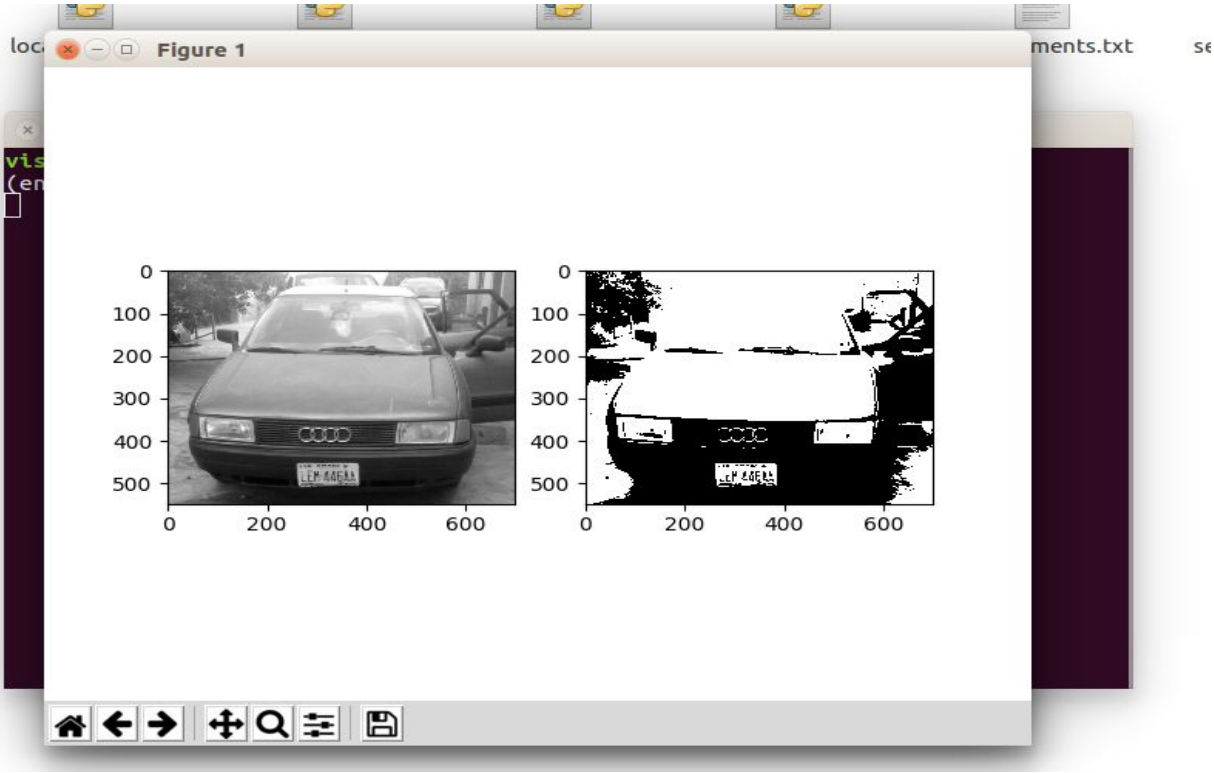
The tkinter image file input dialog box will now open.



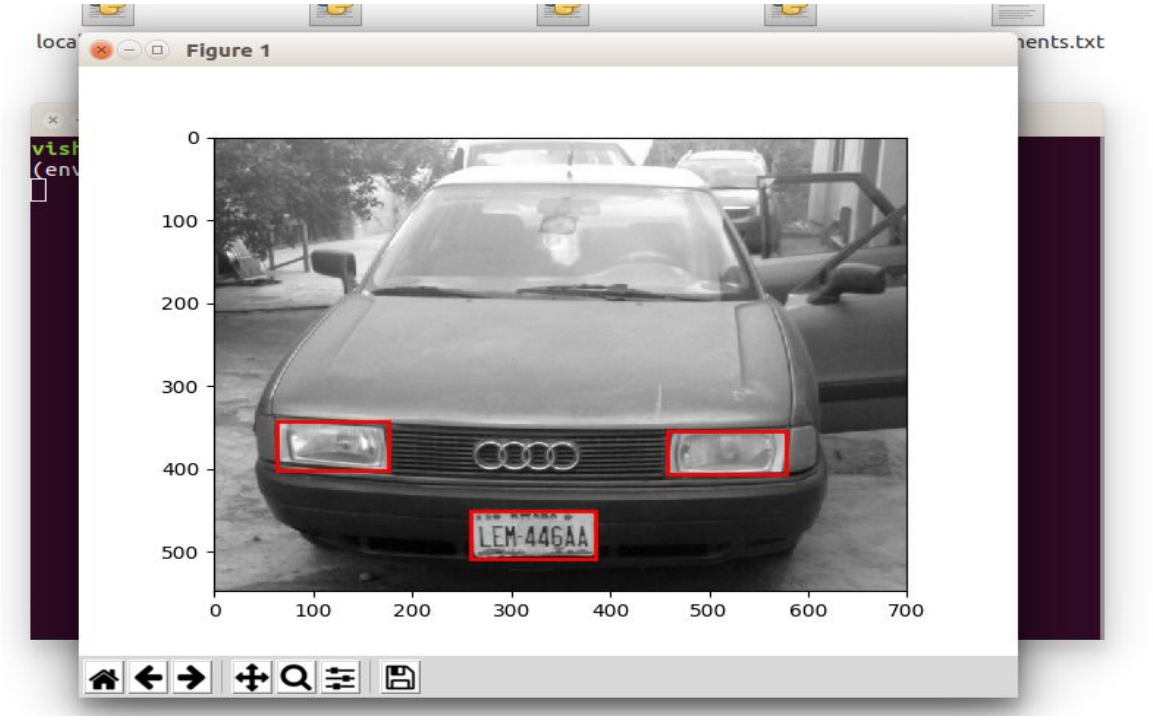
Now open any car image placed inside ***images*** folder in the project folder.



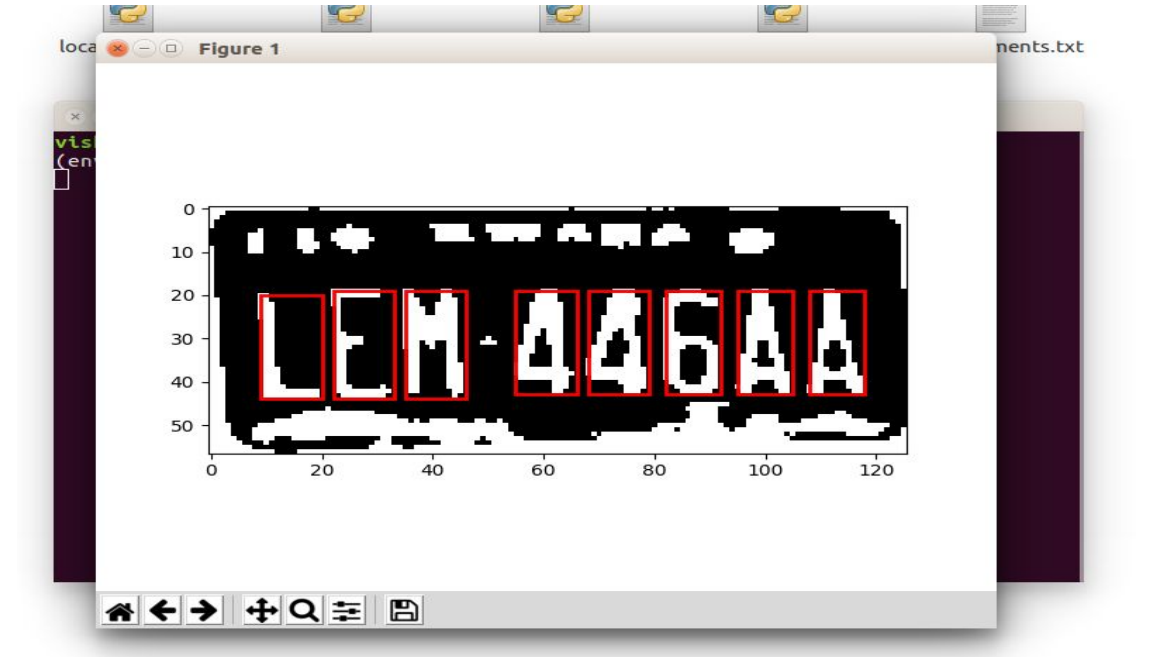
The next step displays the license plate detection process (plate localization). In this process the original image is converted to its grayscale version. Now to localize license plate from the image a specific threshold is applied to the grayscale image. The following image shows a comparison between the grayscale image and the threshold image in the matplotlib pyplot.



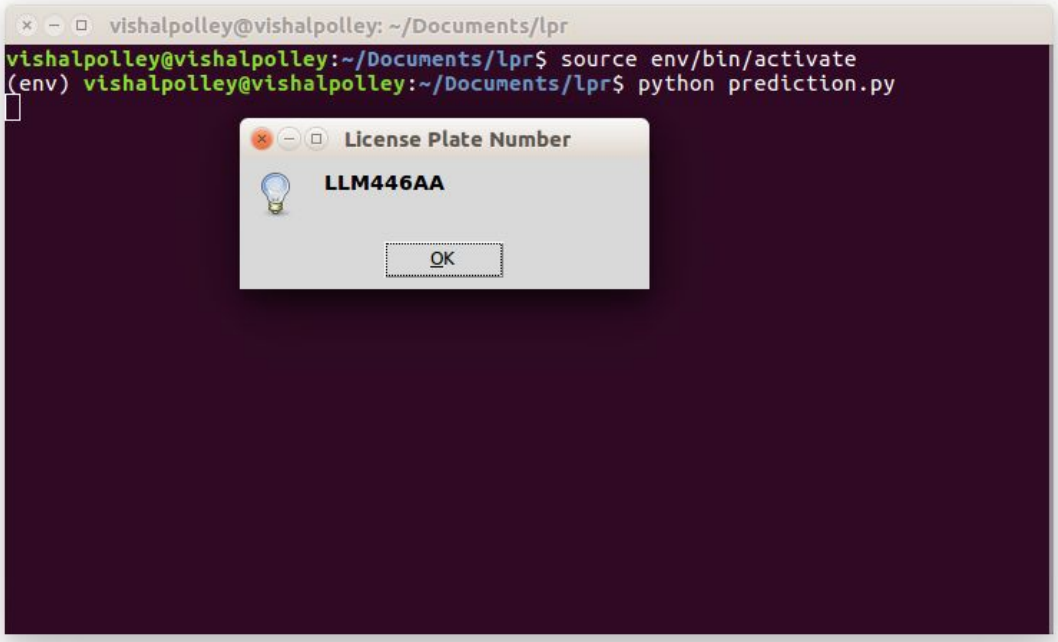
Now after localizing license plate from the original image, the next image shows the process of identifying all the connected regions in the image using the concept of *connected component analysis* (CCA). It basically helps us group and label connected regions on the foreground. A pixel is deemed to be connected to another if they both have the same value and are adjacent to each other.



In the next step we have mapped out all the characters from the image using character segmentation process and CCA.



In the final step we have used supervised machine learning to detect the possible character present on the license plate. It makes use of a known dataset (called the training dataset) to make predictions and thus the license plate number is detected and displayed inside a new dialog box as output.



FUTURE ENHANCEMENTS

- The project currently works over still captured images only, and can be modified in future to be implemented to extract license plate information over live video feeds.
- Efficiency of the project can be increased by improving the character segmentation algorithm so it can be applicable to various types of car's images.
- Image Processing speed can be increased by installing faster processors.
- The project can be implemented with Raspberry-Pie so as to use it for real life conditions.
- Project currently have a simple GUI based on tkinter but it can be made much more user friendly and easily navigable by using many other modules.
- We are currently using pre build Machine Learning libraries for recognizing and detecting license plate numbers. Self written machine learning codes can further enhance the speed and process for images of all conditions.
- More number of character datasets can be trained with the project, so to detect and recognize characters of regional languages and hand written license plates.

SOURCES

1. **Google :** For searching various queries.
2. **scikit-learn :** For implementing various machine learning algorithm in python.
   http://scikit-learn.org/stable/
3. **scikit-image :** For implementing various image processing algorithms in python.
   https://scikit-image.org/
4. **tkinter :** For building graphical user interface in python.
   https://wiki.python.org/moin/TkInter
5. **Github :** For searching various projects related to optical character recognition for learning purposes.
   https://github.com/search?q=optical+character+recognition+python