Developing Open-Source License Plate Recognition Solutions using deep learning modules and improving its reliability

Author: Dayesh Raval, Data Science Intern, Aunalytics

4th year student of Computer Science, with specialization in AI & ML

at Vellore Institute of Technology (VIT), Bhopal, India

• EXECUTIVE SUMMARY

License Plate Recognition (LPR) involves capturing photographic video or images of license plates and converting the optical data into digital information in real-time. Despite its perception as a relatively modern innovation, LPR technology has been around since. The primary objective of this research and development project is to create an open-source MVP for a License Plate Recognition System. Extensive research has been conducted to identify optimal approaches to develop a Most Valuable Program (MVP). The goal of the MVP is to convert an input license plate image into text, providing essential functionality without demanding strict accuracy standards.

• BACKGROUND OF THE RESEARCH

The history of license plate recognition technology can be traced back to an overseas application developed by the Police Scientific Development Branch in the UK approximately forty-five years ago. License Number-plate recognition (LPR) technology utilizes optical character recognition on images to read vehicle registration plates and generate vehicle location data. Automatic number-plate recognition systems can be used to store the images captured by the cameras as well as the text from the license plate. The initial implementations of LPR were observed at the Dartford Tunnel and the A1 road in Great Britain. However, the system did not gain widespread use due to its cost implications and user challenges. Throughout the 1990s, the technology underwent major advancements which resulted in it evolving into a more affordable, user-friendly application. Modern LPR software has streamlined the process into a cost-effective, accessible technology that is widely used in multiple applications and is supported by the integration of machine learning and deep learning. Presently, the image data we receive is superior in terms of quality and size as compared to earlier times, rendering the existing and outdated technologies ineffective. The emergence of Deep learning and its diverse frameworks has revolutionized the understanding, processing and recognition of this vast image data.

- **PROBLEM STATEMENT**

The research aims to develop an open-source Minimum Viable Product (MVP) using Artificial Intelligence, machine learning, deep learning frameworks, etc., with the given image data. Despite the importance of robust and accurate AI/ML LPR systems, the literature on AML and machine learning is relatively limited. The general procedure for License Plate detection and recognition involves four main steps to obtain the necessary information, i.e., **image acquisition, plate localisation, character segmentation, and character recognition**.

Previously, many LPR systems claimed accuracy when trained to match plates from a single jurisdiction or region, but failed when trying to recognise plates from other jurisdictions due to variations in font and shape. These significant changes in the license plate protocol affected OCR(Optical Character Recognition) accuracy, specifically when additional characters or new license plate designs were introduced. The earlier models were able to recognise only clean and clear images while struggling with varied image qualities such as noisy, blurred and cropped images. Overall, ALPR(Automatic license plate recognition) systems were not able to adapt to the above changes quickly in order to be effective. Hence, the research endeavours to develop an MVP solution to overcome the problem statement and shortcomings in existing LPR systems.

- **PROPOSED SOLUTIONS TO THE PROBLEM STATEMENT**

The core of our study and model development was centred around the stated LPR solutions:-

1. **Evaluating EasyOCR & OpenCV for LPR**
2. **Evaluating YOLOv8 & EasyOCR for LPR**

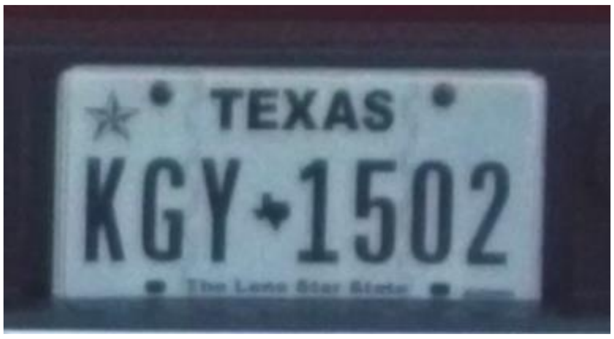
This was made possible due to the availability of various quality and types of license plate images from a car wash company. The dataset consisted of around 100 images. Of these, **37 images** served no purpose to our problem statement and were discarded. The final dataset included **63 license** plate images of different qualities and resolutions.

Evaluating EasyOCR & OpenCV for LPR

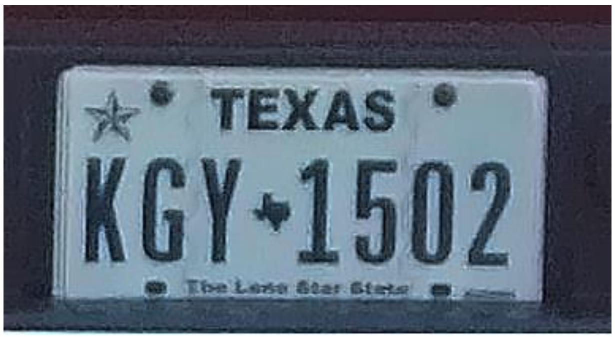
The first step of the open-source solution-building process was checking the images which were of different qualities and resolutions. Based on these, we passed the images through various transformation steps using the OpenCV library for better detection and recognition accuracies for the license plate numbers. The following transformation order was followed: -

1. **Adding Sharpness and removing the noise from the input image**
2. **Adding brightness and contrast to the image**
3. **Smoothening of the image for further noise removal**

The transformations worked very well on the images and an example is shown below: -



Original Image



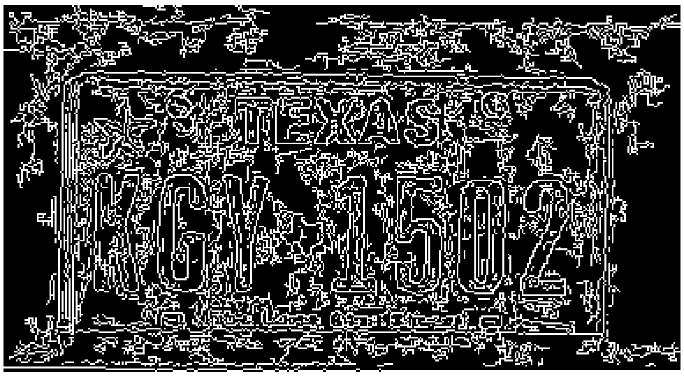
Transformed Image

Post the transformations, the subsequent steps were performed for plate recognition and detection: -

1. **Finalising edges for localisation**

2. **Developing and finalising contours, their locations, and applying masking operations**

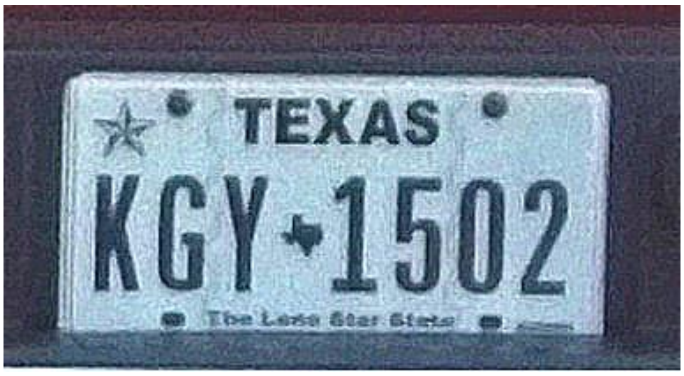
3. **Finally, we obtained the detected license plate image and used the EasyOCR library for character recognition and its corresponding confidence score**



Edges detected using Canny Edge detector

RESULTS OF THE METHODOLOGY: -

1. **An accuracy of 82% was achieved from the set of images of different kinds.**
2. **This model worked very well for clear and cropped images, and demonstrated considerable accuracy with dark images, but fell short while working with noisy and cropped images.**
3. **The results are documented in project documentation using the Confluence software and an Excel sheet with text and number data from the license plate, along with their respective confidence scores.**



Results: -

```
([[125, 47], [253, 47], [253, 85], [125, 85]], 'TEXAS', 0.9995248166824805), ([[39, 87],
[335, 87], [335, 177], [39, 177]], 'KGY-1502', 0.6462661924071503)]
```

Evaluating YOLOv8 & EasyOCR for LPR

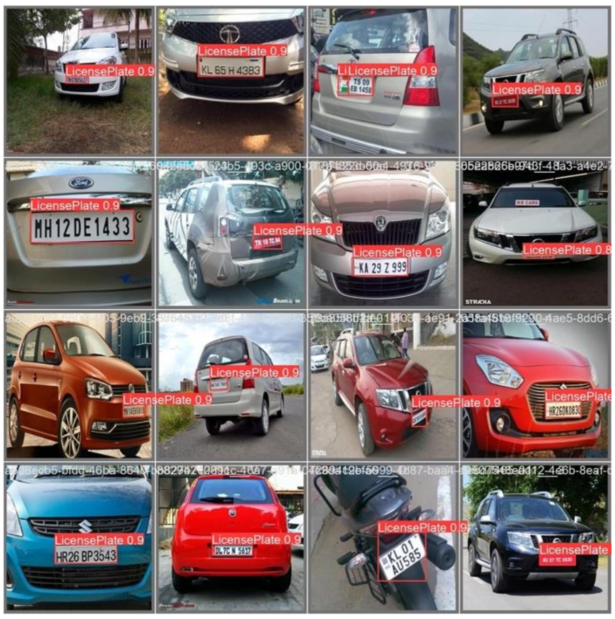
The first step of the open-source solution-building process was checking the images which were of different qualities and resolutions. To initiate the analysis, we cloned the YOLO8 model files from a GitHub account into our notebook. The next steps involved:-

1. Training the model using a custom dataset from RoboFlow of license plate images.

2. We need to use the Roboflow Python library to work with a dataset, install the Roboflow library, initialise it with an API key present in our Roboflow account, and then access a specific project and dataset within your Roboflow workspace.

3. We must have the necessary dataset, model file, and configuration files in the specified paths. These required dependencies can be installed by downloading the requirements.txt file from the cloned account to the notebook.

The model was for around 40 epochs, and it performed very well on the custom dataset. The model was then used for detecting and recognising our license images. There was predicted with OCR.py, which contained the code for recognising characters on the license plates using the EasyOCR library and its integration with the YOLOv8 framework.



Results of detection on the custom dataset

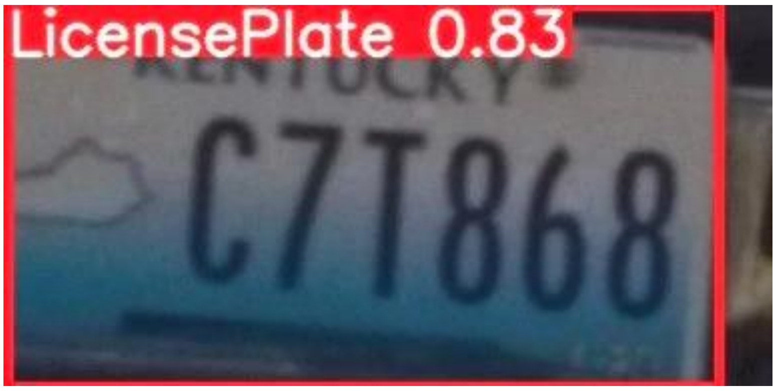
RESULTS OF THE METHODOLOGY: -

1. The model could only detect the license plates with good confidence scores but did not recognise the characters on our dataset images and live videos.

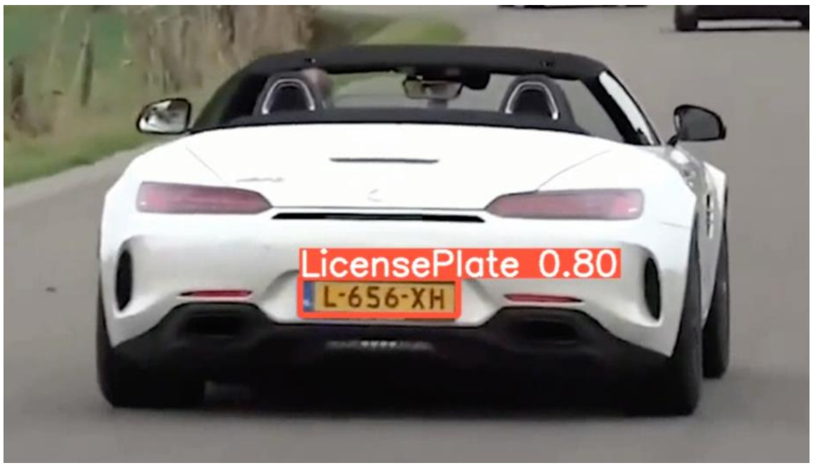
2. The EasyOCR integration with YOLOv8 does not suit the license plate images we have ingested into the model.

3. As per my discussion with the chief Data Scientist, this part of the solution for license detection can be integrated with the first methodology covering the transformations and character recognition of the images for better recognition with OpenCV and EasyOCR respectively.

4. Till then, this can be deployed on a version control system and worked on for improvements.



License Plate Detected, but characters not recognized!!



License detection in a live video, but no character recognition

- **CONCLUSION TO THE RESEARCH FINDINGS**

The accuracy of recognition is of utmost importance in this system, and this application should be optimized and modified to overcome the accuracy limitations. In order to make the recognition more precise, we should add some pre-processes to remove the interferences. Therefore, the above-mentioned LPR solutions can be improved using: -

1. Certain advanced text-extract algorithms must be utilised for better training and confidence scores across all types of images.
2. Advanced deep learning frameworks need to be studied and employed in the solutions for better detection capabilities for various qualities of images.
3. The search for the MVP open-source solution for LPR still continues…

Moreover, the future scope of the licence plate recognition study involves delving into complex environments like vehicles on dark nights or in heavy rains. If we accomplish all of the objectives, this application will have a very promising future.

Today advances in technology have taken Automatic Number Plate Recognition (ANPR) systems from complicated and expensive setups to simple mobile applications, enabling a 'point to shoot' method.

Going forward, the automatic vehicle recognition system could potentially play a significant role in enhancing defence-related threat detection and improving safety for women, as they can easily detect the number plate before using a cab or other service.

The OCR (Optical Character Recognition) method is sensitive to misalignment and different dimensions, so must generate different types of templates for different RTO specifications. Statistics can also be used to verify and confirm traffic card results. There are some restrictions such as Currently speed, text on bus, slope in image etc. These limitations can be removed by further improving the algorithm

- **<u>CITATIONS TO THE RESEARCH METHODOLOGY- BIBTEX</u>**

1. **YOLOv8 Model: -**

```
@software{yolov8_ultralytics,
  author = {Glenn Jocher and Ayush Chaurasia and Jing Qiu},
  title = {Ultralytics YOLOv8},
  version = {8.0.0},
  year = {2023},
  url = {https://github.com/ultralytics/ultralytics},
  orcid = {0000-0001-5950-6979, 0000-0002-7603-6750, 0000-0003-3783-7069},
  license = {AGPL-3.0}}
```

*The usage of the software is in accordance with the AGPL-3.0 license.

2. OpenCV Model: -

```
% to cite the sofware library
@misc{itseez2015opencv,
  title={Open Source Computer Vision Library},
  author={Itseez},
  year={2015},
  howpublished = {\url{https://github.com/itseez/opencv}}
}
% to cite the manual
@manual{itseez2014theopencv,
  title = {The OpenCV Reference Manual},
  organization = {Itseez},
  edition = {2.4.9.0},
  month = {April},
  year = {2014},
  notes = {See \url{http://opencv.org/}}
  }
```

3. EasyOCR Model: -

There is no such citation for EasyOCR;
Some acknowledgements and references that have been added, hope this works out: -

All deep learning part is based on Pytorch. :heart:

Detection part is using CRAFT algorithm from this official repository and their paper (Thanks @YoungminBaek from @clovaai). We also use their pretrained model.

Recognition model is CRNN (paper). It is composed of 3 main components, feature extraction (we are currently using Resnet), sequence labeling (LSTM) and decoding (CTC). Training pipeline for recognition part is a modified version from deep-text-recognition-benchmark. (Thanks @ku21fan from @clovaai) This repository is a gem that deserved more recognition.

Beam search code is based on this repository and his blog. (Thanks @githubharald)

Data synthesis is based on TextRecognitionDataGenerator. (Thanks @Belval)