A close-up of a logo

Description automatically generatedA white background with black text

AI-generated content may be incorrect.

Automatic Vehicle Plate Number Detection & Extraction System

24CSAI02I

***Project Number: 17***

**Name:** Abdelrahman Elgewily

**ID:** 229214

**Table of Contents**

[**1)** **Abstract:** 3](#_Toc196132450)

[**2)** **Introduction:** 3](#_Toc196132451)

[**3)** **Tools & Technologies Used:** 3](#_Toc196132452)

[**4)** **Implementation Details:** 4](#_Toc196132453)

[**5)** **Challenges:** 4](#_Toc196132454)

[**6)** **Results:** 5](#_Toc196132455)

[**7)** **Conclusion:** 5](#_Toc196132456)

[**8)** **Reference Pictures:** 6](#_Toc196132457)

# **Abstract:**

This paper presents a license plate detection system that detects and locates the license plate of cars in images. The system uses the YOLO model to accurately locate the license plate in the images. The system also uses some preprocessing techniques like gray scaling, cropping, and thresholding to improve the license plate in the images to prepare it before using OCR. OCR is employed using Tesseract to perform OCR to extract characters from the license plate image. The extracted text is written into a .txt file so that it can be used whenever needed. The system is an easy model, but can further be enhanced to implement more features like country identification, and can be developed to recognize different forms of other license plates for various countries.

# **Introduction:**

License plate recognition is an important technology in traffic, helping with monitoring, security, and safety. This system automatically recognizes license plates from images using computer vision and optical character recognition (OCR).

In this project, the developed license plate detection system is capable of recognizing license plates in static images. The system used in this project employs the YOLO model for object detection, specifically identifying the license plate.

Lastly, once the license plate is detected, the coordinates of the plate in the image are cropped and sent to the OCR, which prints the characters in the cropped image and saves them to a .txt file for further use.

# **Tools & Technologies Used:**

|  |  |
| --- | --- |
| **Python** | The main language utilized in the project's development is due to its simplicity and its capability to support machine learning and computer vision. |
| **Open CV** | It is utilized in image processing tasks, such as reading images, pre-processing images, and displaying them. |
| **YOLO** | It is a real-time object detection system, utilized to recognize and extract license plates in images. |
| **Ultralytics** | It utilizes YOLOv8 to load, train, and predict. |
| **Paddle OCR** | It’s an optical character recognition engine, used to extract characters from an image (Arabic - English) |
| **Tesseract OCR** | It is an optical character recognition engine, used to extract text from images. |
| **Regular Expressions** | It is used to remove unwanted characters from the extracted text. |
| **Jupyter Notebook** | It’s the environment in which the project was run and tested. |

# **Implementation Details:**

The license plate detection system starts by loading an image of a vehicle using OpenCV. The image is then read from the image path given to it by the user. After the model was trained on the dataset, it takes the image path and, using the YOLOv8 model, it performs object detection to find the license plate. After the model detects the license plate in the image, the system crops the license plate region and saves it into 4 variables x1, y1, x2, and y2. After that, the system takes the license plate position in the image and passes it to the OCR, but before extracting the text, some preprocessing is done to improve the readability of the text by decreasing noise and increasing contrast of the image, which improves the accuracy of the OCR. Once the preprocessing is done, it passes it to Tesseract OCR. After the text is extracted, it is cleaned using regular expressions to remove any unnecessary characters like @ symbols and similar characters. Finally, the license plate number is printed in the console and saved to a .txt file, and each new entry is entered into a new line in the same file. This system is a simple representation and can be updated to work on videos or live feeds. The other OCR used paddle was used to compare which is best paddle or tesseract and with no preprocessing other than BGR to RGB is has better accuracy than tesseract and can get the charecters even if the image is taken from an angle unlike tesseract who has a high chance of not detecting the text even with preprocessing.

# **Challenges:**

While developing this license plate detection model, many challenges were faced. These challenges impacted the implementation process and the system's performance. The first challenge was making sure the model had accurate detections because YOLO is dependent on the quality of the dataset, which is why preprocessing is very important, as noisy or blurry images can make it difficult for the model to accurately detect the license plate's location.

Another challenge faced was with the preprocessing for the OCR, since license plates have different fonts, colors, and lighting conditions in the images, making it difficult for the OCR to accurately recognize the characters and extract them. Applying grayscale and thresholding was very important in making sure the OCR was able to recognize the characters in the images, no matter the image conditions.

Multiple OCRs were tested like fast plate OCR but it can only recognize Argentinian plates, it was functional but wasn’t helpful to achieve the final goal of recognizing text from any plate I give it. Trying to train a model on text yielded mixed results as it wasn’t accurate with images that were at an angle, so it wasn’t any different from using the tesseract OCR.

# **Results:**

The developed license plate recognition system accomplishes the objectives of license plate information detection and extraction from images. Multiple scenarios of differing lighting and image quality is managed reliably by the YOLO model for license plate detection as well as position identification between multiple images. The model also provides the bounding box coordinates of where the license plate region is located within the image, which accuracy of cropping for OCR processes later on.

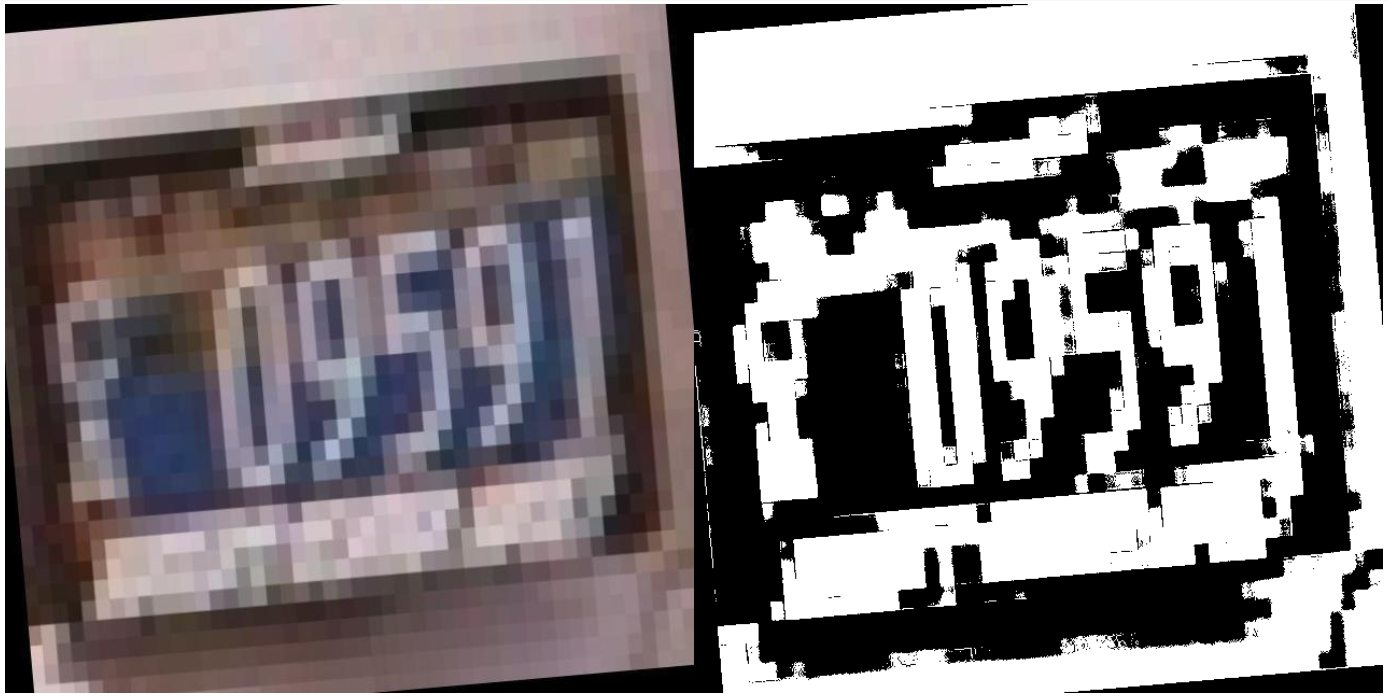
When coordinate information of the license plate inside the image is provided to the OCR, several preprocessing steps are performed, such as gray scaling and applying a threshold to enhance the legibility of the characters, thereby significantly improving the OCR's ability to accurately extract text from the image. Using Paddle OCR yielded better results and was compatible with both English and Arabic which were my target languages. Paddle OCR can extract characters from multiple angles without data aggregation or preprocessing to a certain extent, but still far better than Tesseract OCR.

# **Conclusion:**

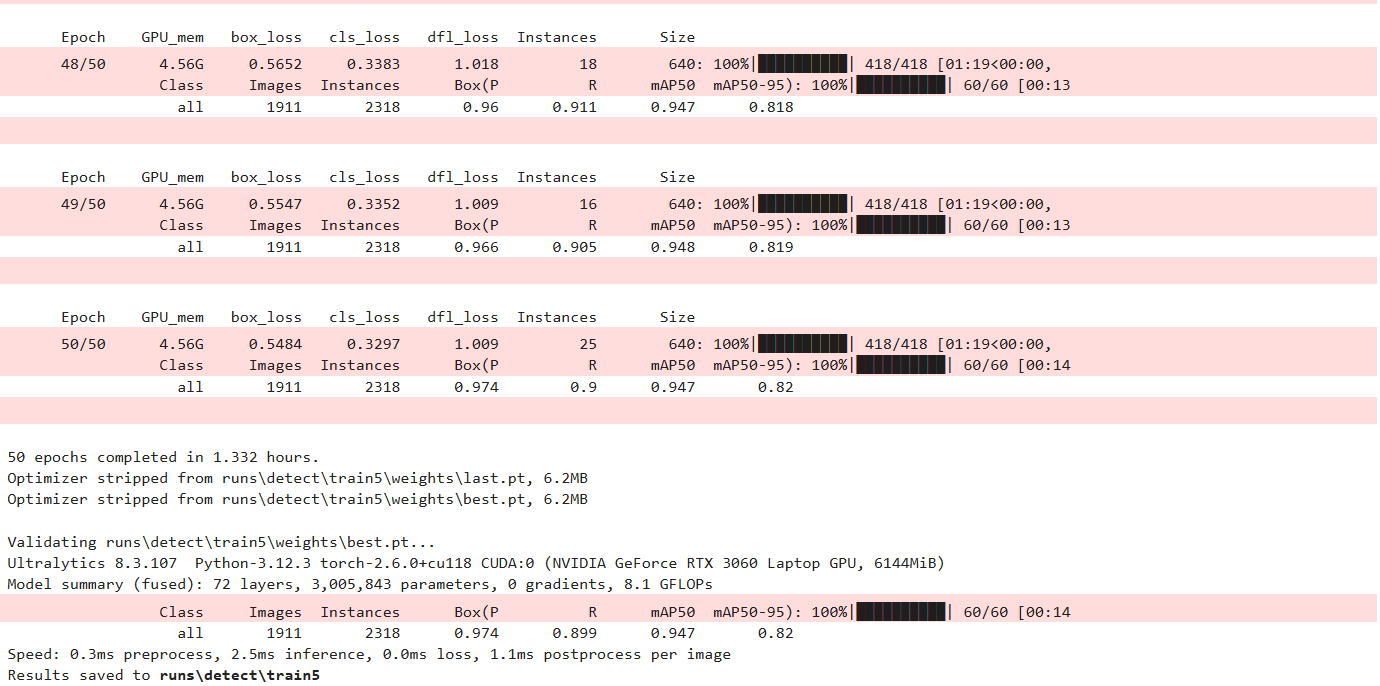
In conclusion, the license plate detection system functions accurately and successfully made using optical character recognition techniques (OCR) and some deep learning techniques. The system is made up of two parts Yolo to train the model and detect the location of the license plate in the image and the other part is giving the positions of the license plate to the OCR to analyze and extract the characters. The use of preprocessing techniques was crucial to achieving the accurate results from the model and the OCR.

The system is a simple demonstration of making a functioning license plate recognition system as this model can be enhanced/upgraded to be able to analyze videos or live CCTV feeds which would be best in real world situations.

# **Reference Pictures:**



**Fig 1.** Image Preprocessing using a function called enhance before model training. This is the original image and the preprocessed image.

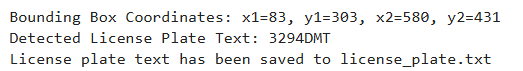


**Fig 2.** The training of the model completed results and accuracy with 50 epochs





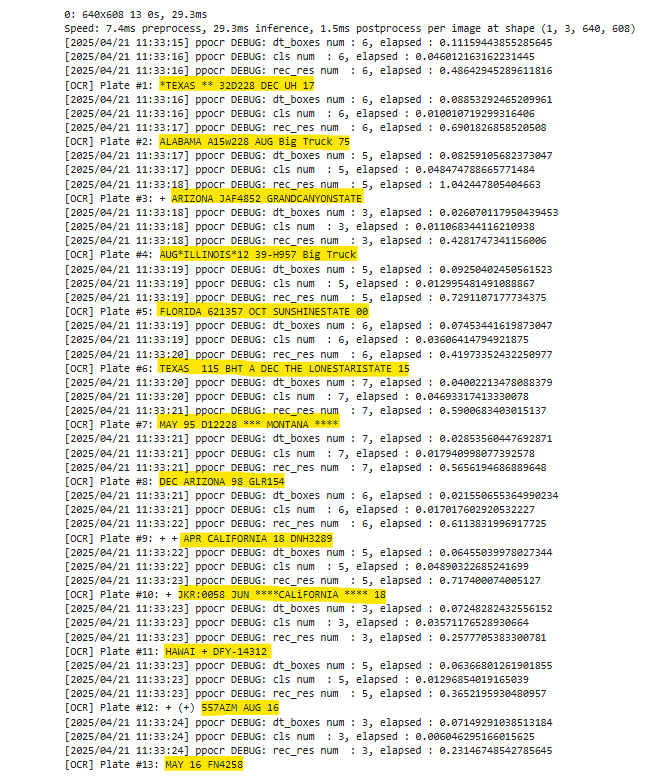
**Fig 3.** (Output of the trained model using the best accuracy model detecting the license plate on 2 different images.)



**Fig 4.** The license plate info printed and saved to a .txt file.



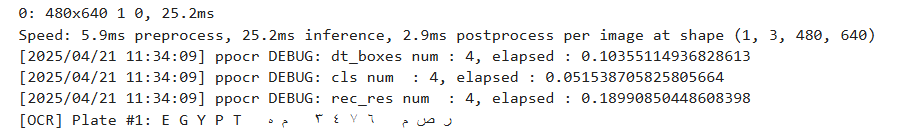
**Fig 5.** Image with Multiple US States License Plates

****

**Fig 6.** License Plate output from (**Fig 5**)



**Fig 7.** Egyptian license plate

****

**Fig 8.** Character output on (**Fig 7**)