License Plate Detection & Identification

Kyle Williams, Tyler VanderMate, Wen-Kai Chen, Ashley Yang

December 6th, 2023

Abstract

This report detailed the implementation of license plate detection and identification. Leveraging three stages our system performed image localization, image preprocessing and optical character recognition. YOLOv8 was trained for image localization to pinpoint the license plate within the visual data, with a dataset hosted on Roboflow. Subsequently, the option to perform image preprocessing to enhance the quality of captured license plate was provided through either OpenCV perspective transformation or image rectification. Finally, Easy OCR was utilized to extract alphanumeric characters from the plates for identification. The combination of all three stages was important for providing an effective solution for license plate detection and identification.

1 Introduction

License plates serve as crucial identifiers for vehicles. The demand for efficient license plate detection and identification systems has increased with traffic demands. This project provided an implementation of license plate identification and sought to improve the results through image preprocessing. The report highlighted image processing techniques, including perspective transformation and image rectification. Perspective transformation adjusted for spatial distortions, while rectification refined and aligned extracted information, culminating in more accurate license plate identification. The system emerged as a potent real-world solution. For example law enforcement benefits from more accurate vehicle identification for tracking and surveillance, while toll road management gains more accurate toll collection. Parking ticket implementation can also be expedited which overall contributes to more efficient urban management. As the report delved into the system's stages and results, a clear understanding of its functionality and potential emerged. These insights highlights the system's effectiveness and uses while paving the way for continued improvements and usage of license plate identification.

By combining multiple components our project achieved better accuracy than just using the EasyOCR library by itself. YOLOv8 was used to establish the area of interest for subsequent processing and identification. Without proper localization of the area of interest the identification stage would have had the possibility of identifying not only text from the vehicle but text from other objects in the image. After localization of the area of interest the image had the possibility of undergoing preprocessing that improved the end result. Two major preprocessing used were perspective transformation and image rectification. These processes helped improve identification of license plates at steep angles and distant perspective. The system's core utilized the EasyOCR library, a robust tool for text recognition, as the final stage. This integration allowed the system to decipher textual information after preprocessing and localization of the images. The plate detection model was tailored to license plate recognition of images but can in the future be expanded to videos and live footage with object detection and tracking. The synergy between the EasyOCR library and the preprocessing, orchestrated through Python scripts, ensured better accuracy overall for identification.

2 Method

There are 3 stages to this project. The area of interest is localized with Roboflow. Then given the coordinates from stage 1, the image is projected to show the localized license plate with perspective projection or rectification. Additionally EasyOCR was used to recognize the license plate number.

2.1 Dataset consolidation

The main purpose of dataset consolidation is for object detection model training and result evaluation. We need to search for several datasets to keep the diversity. For example, we can search for licence plate images from Vietnam, Europe, and the USA. Moreover, we need to find some license plate datasets with different angles, so that we can make sure our perspective projection is successful. We can make sure that every image has its relative annotation. Moreover, those annotation should contain complete bounding box information. Then we will separated the image file form the XML file to make them easier for our programs to read. The dataset we found can be located in the "Dataset" section of Experiments

2.2 Localization of License Plate

Once we found a dataset that matched that description (see 3.1) and it was properly processed, we could train the AI model to perform object detection. For this project, we decided to use YOLOv8 to perform our object detection needs. YOLOv8 is made by Ultralytics, and is one of the most popular object detection models in the ecosystem. YOLOv8 can cause performance decreases when using it for real-time work, but since we will be using single images of license plates, that's a nonissue. The model is well optimized for object detection, and will be extremely successful on a detection test as straightforward as license plates.

In this project, we used Roboflow to host our AI model. While you must upload the dataset, manually preprocess, and manually run YOLOv8 on Python code, Roboflow can host the final model weights and provides a direct way to implement our code into the pipeline, in addition to being one of the most popular AI hosting services in circulation. We first tested our dataset with the Roboflow 3.0 model, and it was extremely successful sporting a mAP in the 0.8s.

With proof that our dataset works, we ran YOLOv8 on Google Colab. To run YOLOv8 on Google Colab using a dataset hosted on Roboflow, you install the pip package ultralytics, the python display, and pip package roboflow. Then, you must load in your custom dataset using the API key and project version. From there, you can use the code to run YOLOv8. We ran the project for 50 epochs, and the results can be viewed in the "Results" section, but it's fair to say it was an extremely accurate model. Then you can upload the weights back into Roboflow for pipeline implementation

2.3 Perspective Transformation

The primary usage of perspective transformation in the preprocessing stage is to take the in coordinates and perform transformation to focus on the area of interest. The OpenCV perspective transformation function was utilized for this purpose. The area of interest is given in four coordinates by localization in stage one. Then after defining the source and destination points based on these four coordinates, the perspective transformation matrix is calculated using OpenCV function getPerspectiveTransform(). Next the OpenCV warpPerspective() function is applied to the image to achieve the final image that will be used in the next stage. This stage will work for videos as well as images. There is also the option to search for contours within the localized image.

2.4 Rectification

We import the Image Rectification repository and use the function for the calculation. First, we calculate the horizontal edges and try to find out the horizontal vanish point. Then, we do the same thing on the vertical direction. After we get the result, we compute a homography matrix and apply it on the input image.

2.5 Digit Recognition

For the digit recognition we use the open source code from EasyOCR. EasyOCR is a well known OCR model which support several languages and has a pretty good accuracy. First, we get the result from license plate localization and perspective projection which will help the OCR model to focus on the license plate and improve the performance. Next, we convert it from GRB to grayscale and apply

gaussian blur. And than we input the image inside EasyOCR model. We set a constraint for the model which the model only can detect English letter, number, dot, space and dash. After we get the result, we choose the one with the highest probability and extract the text imformation for the output.

3 Experiments

3.1 Dataset

For the dataset, we wanted to find a variety of license plates, from many different countries and angles. This was for our license plate localization, so these datasets were required to have bounding boxes around the license plates in the photos. We searched on three popular sites for Dataset hosting for object detection — Kaggle, HuggingFace, and Roboflow Universe. After searching, found two datasets that work, one with around 500 images and the other with 2k. Both are credited at the bottom of this document.

In our dataset, we also had image level augmentation. This is a useful step where we can alter the image so the AI model is better at recognizing blurrier images and images with a lot of noise. So, the Noise filter and the rotation filter was added to the dataset so it could recognize these images better. Once they were preprocessed into Train, Valid, and Test, and uploaded to Roboflow for hosting, we could train our dataset on the model.

3.2 Evaluation metrics

3.2.1 Easy OCR

We use F1 score to evaluate different modification of the EasyOCR model. We used the "License Plate Annotated Image Dataset" from kaggle for this evaluation. This dataset contained about 400 images with annotaion. This is the only few dataset of license plate that has license number in the notation. During the evaluation we compared the result from different EasyOCR setting with the ground truth. Then, we calculate the F1 score by this equation:

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

The only difference between the setting is gaussian blur and letter restriction of dot, space and dash. After evaluation, the setting with no gaussian blur and no letter restriction got 46.45% F1 score. The setting with only letter restriction got 49.95% F1 score. The setting with gaussian blur but no letter restriction got 51.41% F1 score. Finally, The setting with gaussian blur and letter restriction got 56.07% F1 score.

3.3 Results

There are three different checkpoints in this project that give us clear results to analyze. The first checkpoint in the project is the license plate localization done by the AI model. The second checkpoint is the perspective translation and image rectification to render the image readable. And the final checkpoint is the implementation of EasyOCR for identification of the digits on the license plate.

3.3.1 Localizing to License Plates

Some example results of the YOLOv8 model on our dataset are below. This was our final model working on some test images. You can see examples of our dataset in the image too, with some shearing, some rotation, and some noise. This was trained on the dataset of 2,000 images. Evaluation of the numbers is in the "Analysis and Discussions" section.

Figure 1: YOLOv8 on Custom Dataset E

3.3.2 Perspective Transformation and Image Rectification

For the image transformation of the preprocessing stage there are two options, perspective transformation or image rectification. Perspective transformation was used in the case that the license plate had other objects that interfered with the image rectification algorithm and in the cases where the license plate was not at a steep angle. For example the bird in figure 6 caused the image rectification to fail and did not produce a usable image. While for figure 9, image rectification produced a better preprocessing result for an image that had a steep angle of the license plate.

Figure 2: Rectification on figure 6

Figure 3: Perspective Transformation on figure 6

Figure 4: Rectification on figure 9

Figure 5: Perspective Transformation on figure 9

3.3.3 License Plate Detection and Identification Results

The final step of the project was implementing EasyOCR into the project. Easy OCR can take the text remaining in the image and attempt to read it. Before the image localization, image rectification and perspective transformations, EasyOCR was unable to read the license plate with any accuracy. Now, it's able to read the license plate nearly flawlessly. Our project can take an image of a car and

output a result like Figure 11 on any image you give it with a car in it, no matter how angled, with accurate results. Example results are below!

Figure 6: Original Photo

Figure 7: No preprocessing OCR result

Figure 8: Perspective projection Easy OCR result

Figure 9: Original Photo

Figure 10: No preprocessing Easy OCR Result

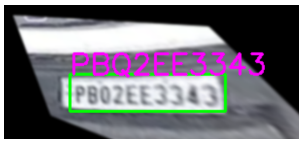

Figure 11: Rectification Easy OCR result

3.4 Analysis and discussions

3.4.1 YOLOv8 Training Analysis & Results

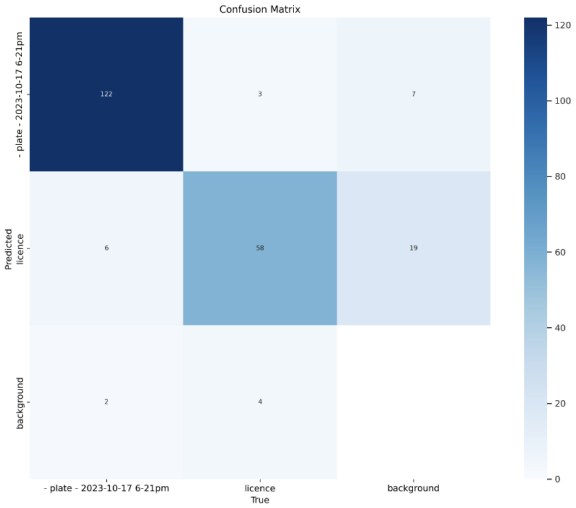

Figure 12: YOLOv8 on Custom Dataset Confusion Matrix

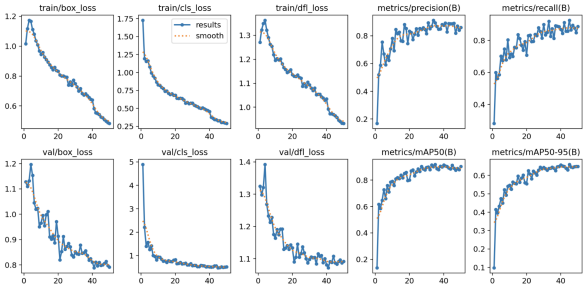

Figure 13: YOLOv8 on Custom Dataset Results

Our final crucial values of the YOLOv8 model trained on our dataset of 2,000 images:
box_loss: 0.4978
cls_loss: 0.2975
dfl_loss: 0.9331
mAP50: 0.903
mAP50-95: 0.648

These are the numbers that matter the most for an AI model training in object detection, and our AI model passed with flying colors. mAP is a particularly crucial value to pay attention to. Usually

in object detection, a mAP50 above 0.6 is considered a "good" mAP value, so having ours be in the 0.9 means that it's an extremely accurate model. License plates are very clearly defined, so this may have been expected when using a model as powerful as YOLOv8, but it's a great result. Looking at the test images, it's clear the detection is top-notch and accurate.

Our first iteration of the model, using the Roboflow Custom Model 3.0, has a 0.8542 mAP50. This is also a very high mAP50, but we were able to increase it by nearly 0.05 by doing some simple preprocessing and changing the model to the more effective YOLOv8!

4 Conclusion

In this project, we combined numerous different processes and systems to accurately predict license plates numbers based off of any image of a car. First, we use YOLOv8's object detection on an image to localize where the license plate is. Then, we perform image preprocessing on the localized image using perspective transformation and image rectification, to warp the license plate to render it readable. Then, we use EasyOCR to read the fixed letters and numbers on the plate to output the license plates. All systems work together to output an extremely accurate result.

Demand for accurate and automatic license plate identification is important for a wide array of industries, from law enforcement for criminal recognition to toll companies to send small fees for driving on highways. This license plate reader can take any image and output a (mostly) accurate license plate number, even if the image is warped or taken at extremely high angles, because of the preprocessing implemented. This tool is going to be useful for a long time, as long as license plates are used to identify cars.

5 Contribution

Each of us are required mention the parts that we worked on so this section will be different for each of us.

5.1 Code

What section I worked on in code:
Localization of License Plate & AI Model - (Kyle)
Dataset Selection and Handling - (Kyle)

In this project, my task was to be the sole contributor of the Object Detection part of the project. That meant I was in charge of the image localization using an AI model. I scoured the web for quality datasets, preprocessed the images, and put them in "train", "test", and "valid". From there, I worked on the AI model. I first implemented the Roboflow 3.0 model for proof of concept, and wrote some code to implement that model into our pipeline. Then, I decided between a handful of AI models to land on YOLOv8, for its exceptional object detection. I personally edited the code and ran YOLOv8, got the results graphs and re-uploaded the weights to Roboflow. I then changed the code to put my object detection into the pipeline.

5.2 Report

All sections below were sections I individually wrote.
2.2 Localization of License Plate - (Kyle)
3.1 Dataset (Kyle)
3.3 Results (Kyle) 3.3.1 Localizing to License Plates (Kyle)
3.4 Analysis and Discussions (Kyle)
3.4.1 YOLOv8 Training Analysis Results (Kyle)
4 Conclusion (Kyle)
Contribution (Kyle)

6 References

1. SCH. (2023, June 16). *Plate Object Detection*. YouTube. Retrieved December 5, 2023, from https://universe.roboflow.com/sch-pp85c/plate-641xp/dataset/2

2. Üstünkök, T. (2019). *License Plate Detection*. Kaggle. Retrieved December 5, 2023, from https://www.kaggle.com/code/tustunkok/license-plate-detection

3. Jelal Sultanov. (2020). *License Plate Image Dataset*. Kaggle. Retrieved December 5, 2023, from https://www.kaggle.com/datasets/aladdinss/license-plate-annotated-image-dataset

4. Jaided AI. (2020). *Easy OCR*. Github. Retrieved December 5, 2023, from https://github.com/JaidedAI/EasyOCR

5. Chaudhury, Krishnendu, Stephen DiVerdi, and Sergey Ioffe. (2014). *Auto-rectification of user photos.* IEEE International Conference on Image Processing (ICIP). IEEE, 2014.