Gate Access Controller

Project Report

Name	Code
Muhammad Amr	9220728
Marwan Muhammad	9220808
Youssef Tarek	9220990

Presented to

Prof. Youssef Ghatas

Eng. Muhammad Shawky

Contents

1 Introduction

This report presents the second phase of the Gate Access Controller project. This phase builds on the project proposal by providing details on the algorithms used, experimental results, system performance, and team contributions. We Integrated the model and algorithms used with a fancy UI and dynamic database to be able to verify and determine if a certain car has gate access.

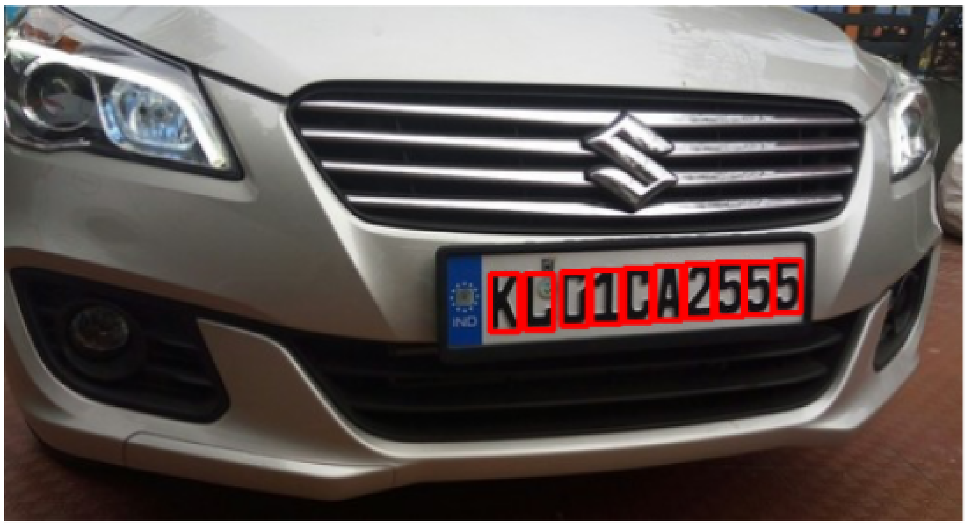
2 Used Algorithms

2.1 Car Plate Detection

The system uses the following algorithms for car plate detection. This involves:

- Preprocessing the input image to enhance quality
  - Resize the image to standard dimensions.
  - Converts color images to grayscale for easier processing.
  - Applying bilateral filter for smoothing.
  - Applying morphological operations like TOP_HAT and BLACK_HAT.
  - Applying adaptive thresholding.
- Plate localization
  - Finding most reasonably near contours.
  - Removing duplicate contours, as sometimes we got double rectangles above each others.
  - Divide contours into groups vertically with some tolerance.
- Character Segmentation
  - Get all characters on the same horizontal line, and while checking appropriate aspect ratio.
  - Making sure all characters have the same height with some tolerance, to further remove any remaining noisy inputs.
  - Applying morphological operations like TOP_HAT and BLACK_HAT and doing adaptive thresholding.
- OCR
  - Made a dataset using another pretrained OCR model to take it's output, validate it and then train our OCR on it.
  - Split the data into training and test data with [4:1] ratio.
  - Feature extraction by resizing then flattening the images.
  - Used different machine learning algorithms like linear and rbf SVM, with different c and gamma values. also tried KNN.
  - Chose the highest accuracy model which was rbf SVM with C=1 and gamma='scale'.

car plate detection image:



2.2 Access Control

The recognized plate number is checked against a predefined list of allowed plates stored in a database. The system:

- Fetches the detected plate number.
- Compares it to the access control list.
- Provides an access decision (granted or denied).

3 Experiment Results and Analysis

3.1 Test Case Variety

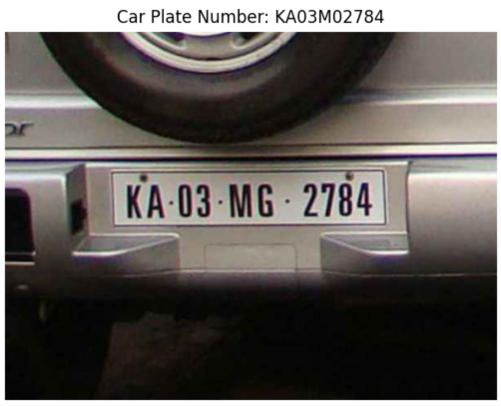
We tested the system on a diverse set of images, including:

- Images taken in different lighting conditions (daytime, nighttime).
- Plates with varying fonts, sizes, and cleanliness.
- Images with multiple vehicles.

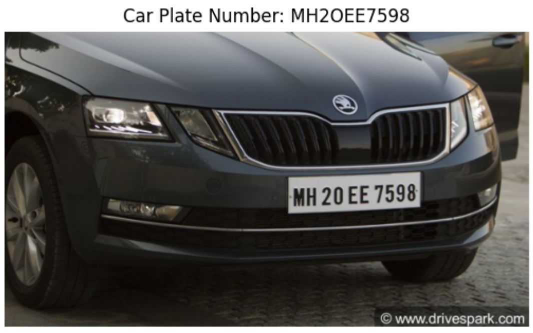
3.2 Comparison Metrics

The following metrics were used to evaluate system performance:

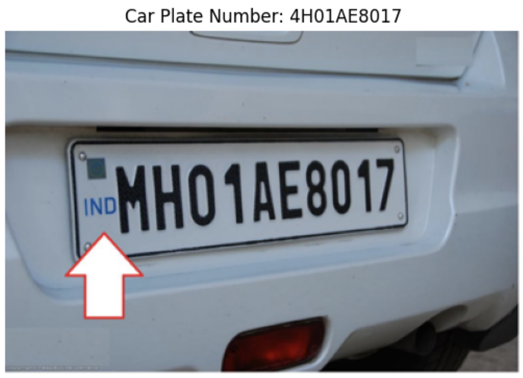
- **Accuracy:** The percentage of correctly identified plates.
- **Recall:** The ratio of correctly detected plates to the total plates in the dataset.
- **Processing Time:** Average time taken to process an image.



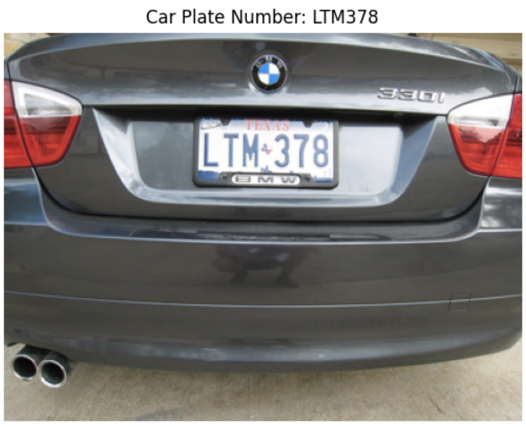
(a) Test 1



(b) Test 3



(c) Test 3



(d) Test 4

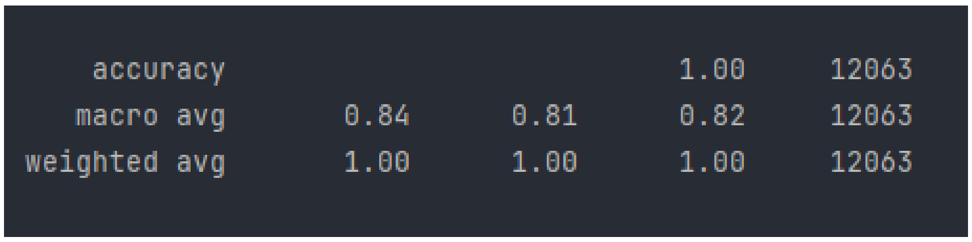
Figure 1: Test Cases

3.3 Performance and Accuracy

The experimental results indicate:

- OCR accuracy: [99.65%]
- Weighted Average Recall: [1.00]
- Weighted Average Precision: [1.00]
- Weighted Average F1-score: [1.00]

Performance graph for OCR:

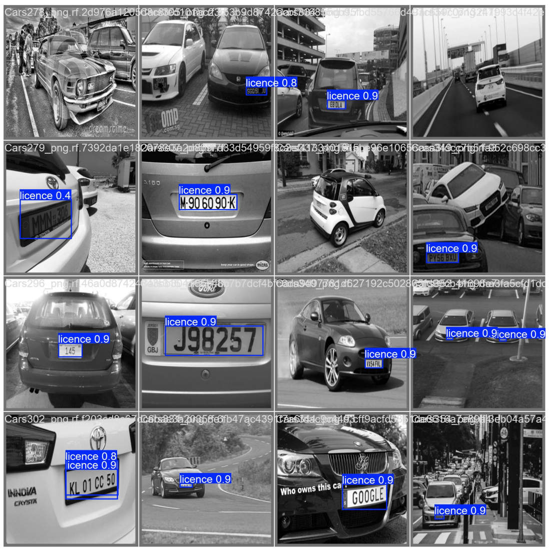


```
     accuracy                           1.00     12063
    macro avg       0.84      0.81      0.82     12063
 weighted avg       1.00      1.00      1.00     12063
```

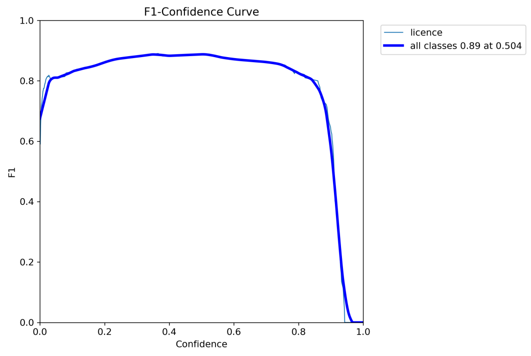
3.4 Pre-trained model

We used Yolo version 11 to detect the car plate location on the image. We fine-tuned the model on a dataset from Roboflow and also used py-tesseract OCR to convert the characters in the image to string. The training of Yolo is done on GPU P100 on Kaggle and the model took 1.5hrs for training.

performance graphs for pre-trained model (YOLO):



(a) Validation



(b) F1 Curve

Figure 2: Test Cases

3.5 Strengths and Weaknesses

Strengths

- High accuracy in detecting plates under standard conditions.
- Robust access control logic.
- Efficient processing for real-time applications.

Weaknesses

- Reduced accuracy under poor lighting conditions.
- Challenges with highly reflective plates.
- Occasional false positives in multi-vehicle images.

4 Work Division Between Team Members

- **Marwan Mohamed** Input Preprocessing, Plate localization, Optimizations and analysis, Adding pre-trained models.
- **Muhammad Amr** Character segmentation and frontend.
- **Youssef Tarek:** Developed OCR, backend and Integration.

5 Conclusion Additional Comments

5.1 Conclusion

Regardless of the limitations of not learning machine intelligence on a deep level, we managed to get really good results that were close to YOLO which is a pre-trained deep learning model. We are looking to further enhance the model to be ready for production.

5.2 Additional Comments

- Future enhancements include supporting Arabic letters.
- Improving OCR after deep diving into machine learning.
- Implementing video features which we didn't do due to not finding high quality dataset.

References

[1] Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2013). Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. *IEEE Transactions on Circuits and Systems for Video Technology*, 23, 311-325. https://doi.org/10.1109/TCSVT.2012.2203741

[2] Zhao, H., Zhang, H., Wang, W., Zhai, J., & Wu, Y. (2016). A Robust Method for License Plate Detection Based on Shape and Color Information. *IEEE Access*, 4, 1886-1895. https://doi.org/10.1109/ACCESS.2016.2521703

[3] Read, J., Choudhury, S., Gupta, S., Raj, S., & El-Husseiny, M. (2014). Design of Automatic License Plate Recognition System Based on Neural Networks. *Journal of Computer Science*, 10(3), 392-399. https://doi.org/10.3844/jcssp.2014.392.399