A Detailed License Plate Recognition System Using YOLO, CNN and OCR

Hira Yalçın Beren Köse Dilara Yılmaz Ebrar Yılmaz

December 31, 2025

Abstract

License Plate Recognition (LPR) is a fundamental computer vision problem widely used in intelligent transportation systems, traffic surveillance, parking management, and security applications. This paper presents a comprehensive and end-to-end LPR system that integrates deep learning-based object detection, image preprocessing, Convolutional Neural Networks (CNN), and Optical Character Recognition (OCR). The proposed approach utilizes YOLO for accurate license plate localization, followed by advanced preprocessing techniques to enhance character visibility.

Experimental evaluations demonstrate that the integration of deep learning and image processing significantly improves recognition accuracy under varying illumination and environmental conditions.

1 Introduction

With the rapid increase in the number of vehicles worldwide, automated traffic monitoring systems have become an essential component of modern transportation infrastructure. License Plate Recognition (LPR) plays a critical role in these systems by enabling automatic vehicle identification without human intervention.

Traditional manual monitoring methods are inefficient, time-consuming, and prone to error. In contrast, automated LPR systems provide real-time, accurate, and scalable solutions. Recent advances in deep learning and computer vision have significantly improved the performance of LPR systems, particularly in challenging real-world environments.

This study aims to design a robust LPR system capable of handling variations in illumination, plate orientation, noise, and background complexity.

2 Problem Definition and Challenges

Despite technological progress, license plate recognition remains a challenging task due to several real-world factors:

- Illumination changes caused by sunlight, shadows, or night conditions
- Motion blur resulting from fast-moving vehicles
- Variations in license plate fonts, sizes, and formats
- Occlusions, dirt, or damaged plates
- Complex backgrounds and reflections

The primary objective of this project is to overcome these challenges by leveraging deep learning-based detection and recognition techniques.

3 Related Work

Early LPR systems relied heavily on handcrafted features and rule-based approaches such as edge detection, color analysis, and connected component labeling. While these methods worked under controlled conditions, their performance degraded significantly in real-world scenarios.

Recent studies have adopted CNN-based approaches for both detection and recognition tasks. Object detection frameworks such as YOLO, SSD, and Faster R-CNN have demonstrated superior accuracy and speed. OCR engines combined with preprocessing techniques further enhance recognition performance.

4 System Architecture

The proposed LPR system follows a multi-stage pipeline illustrated below:

1. Image acquisition
2. License plate detection using YOLO
3. Plate region extraction
4. Image preprocessing
5. Character recognition using OCR and CNN
6. Output generation

Each stage plays a crucial role in improving the overall system performance.

5 Dataset Description

The dataset consists of vehicle images collected under different environmental conditions, including daylight, nighttime, and varying weather conditions. Each image contains at least one visible license plate.

5.1 Annotation Process

License plates were manually annotated using bounding boxes to train the YOLO detection model. Accurate annotation is essential for achieving high detection precision.

5.2 Dataset Splitting

The dataset was divided into training (70%), validation (15%), and test (15%) sets. The validation set is used to tune hyperparameters and monitor model generalization.

6 License Plate Detection Using YOLO

YOLO (You Only Look Once) is a single-stage object detection algorithm that performs detection and classification in one forward pass.

6.1 YOLO Architecture

YOLO divides the input image into a grid and predicts bounding boxes and confidence scores for each grid cell. This unified architecture allows real-time detection with high accuracy.

6.2 Training Strategy

The YOLO model was trained for multiple epochs using annotated images. Data augmentation techniques such as rotation and brightness adjustment were applied to improve robustness.

7 Image Preprocessing Techniques

Preprocessing is a critical step to enhance the quality of the detected license plate region before OCR.

7.1 Grayscale Conversion

Converting images to grayscale reduces computational complexity and focuses on intensity information.

7.2 Gaussian Blur

Gaussian Blur smooths the image by reducing high-frequency noise that may interfere with thresholding.

7.3 Adaptive Thresholding

Adaptive thresholding computes local threshold values, making it robust to uneven illumination.

7.4 Morphological Operations

Erosion removes small white noise, while dilation restores character thickness and connectivity.

8 Optical Character Recognition

OCR converts image-based characters into machine-readable text. The accuracy of OCR depends heavily on the quality of preprocessing and segmentation.

8.1 OCR Challenges

Incorrect segmentation, low contrast, and noise can significantly reduce OCR performance.

9 Convolutional Neural Networks

CNNs automatically learn spatial features such as edges, shapes, and textures.

9.1 Epoch and Training Process

An epoch represents one full pass through the training dataset. Increasing the number of epochs improves learning but may lead to overfitting if not properly monitored.

9.2 Validation Strategy

Validation loss is used to determine the optimal stopping point during training.

10 Implementation Details

10.1 YOLO Detection Code

```
from ultralytics import YOLO
model = YOLO("yolov8n.pt")
results = model("vehicle.jpg")
```

10.2 Preprocessing Code

```
import cv2
img = cv2.imread("plate.jpg", 0)
blur = cv2.GaussianBlur(img, (5,5), 0)
thresh = cv2.adaptiveThreshold(
    blur, 255,
```

```
    cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
    cv2.THRESH_BINARY, 11, 2)
```

10.3 OCR Code

```
import pytesseract
text = pytesseract.image_to_string(thresh,
   config='--psm␣7')
print(text)
```

11 Evaluation and Results

The proposed system was evaluated using detection accuracy and recognition accuracy. Experimental results indicate that preprocessing and deep learning-based detection significantly improve OCR performance.

12 Discussion

The integration of YOLO, CNN, and OCR creates a robust pipeline capable of handling real-world challenges. Adaptive thresholding and morphological operations were found to be particularly effective.

13 Conclusion

This paper presented a detailed and comprehensive license plate recognition system using modern deep learning and image processing techniques. The results demonstrate the effectiveness of the proposed approach in real-world scenarios.

14 Future Work

Future improvements may include real-time video processing, transformer-based OCR models, and larger datasets.

15 References

- Redmon et al., YOLO: You Only Look Once, 2016.
- Gonzalez and Woods, Digital Image Processing.
- OpenCV Documentation.
- Tesseract OCR Documentation.