AI-Powered Number Plate Detection and Recognition Using customised YOLOv8 and OCR

Project Report Submitted in Partial

fulfilment of the requirement for the

award of Degree of

MASTER OF COMPUTER APPLICATION (MCA)

Submitted by

Kondepudi Satish

Reg No: 2314106728

Under the guidance of

<Guide's Name>

Guide Reg No:

MANIPAL UNIVERSITY JAIPUR (MUJ)

CENTRE FOR DISTNACE & ONLINE EDUCATION

February 2025

BONAFIDE CERTIFICATE

Certified that this project report titled “AI-Powered Number Plate Detection and Recognition Using YOLO and OCR“ is the bonafide work of “Kondepudi Satish” who carried out the project work under my supervision in the partial fulfilment of the requirements for the award of the Master of Computer Application (MCA) degree.

SIGNATURE

Name of the Guide

Guide Registration Number

DECLARATION BY THE LEARNER

I, Kondepudi Satish bearing Reg. No 2314106728 hereby declare that this project report entitled "AI-Powered Number Plate Detection and Recognition Using YOLO and OCR" has been prepared by me towards the partial fulfilment of the requirement for the award of the Master of Computer Application (MCA) Degree under the guidance of……

I also declare that this project report is my original work and has not been previously submitted for the award of any Degree, Diploma, Fellowship, or other similar titles.

K. Satish

Place: Hyderabad — KONDEPUDI SATISH

Date:14/02/2025 — Reg. No. 2314106728

Table of Contents

1. Introduction and Objectives of the Study

Automated vehicle identification is crucial for traffic management, security, and law enforcement. AI-powered Number Plate Detection and Recognition (NPD-R) is a vital technology that enables automatic vehicle monitoring by detecting and recognizing license plates. Traditional methods often struggle with environmental challenges such as lighting variations, occlusions, and diverse plate designs. To overcome these limitations, this project aims to develop an advanced NPD-R system using a customized YOLOv8 deep learning model for accurate real-time plate detection and Optical Character Recognition (OCR) for reading license plate numbers.

The primary objectives of this study are:

- To design a robust and efficient number plate detection and recognition system capable of handling various conditions.
- To improve detection and recognition accuracy under diverse environmental scenarios, such as poor lighting, motion blur, and occlusions.
- To integrate OCR technology with YOLOv8 for automatic text extraction from detected number plates.
- To optimize the YOLOv8 model for real-time applications with minimal computational overhead, ensuring efficient deployment on resource-limited devices.

By leveraging deep learning, computer vision, and OCR, this project seeks to enhance vehicle identification, contributing to improved law enforcement, toll collection, and intelligent traffic monitoring systems. The customized YOLOv8 model with integrated OCR will ensure superior detection and recognition performance, making it highly applicable in real-world scenarios.

2. System Analysis:

2.1. **Identification Of Need**:

Traditional number plate detection methods, such as edge detection, template matching, and morphological operations, have been fundamental in vehicle identification systems. While these approaches provided initial success, they struggle with varying lighting conditions, occlusions, and low-resolution images, leading to inaccurate detections. Recent advancements in deep learning have introduced object detection models, such as **YOLO** (You Only Look Once), which have significantly improved detection accuracy and speed. However, challenges such as detecting plates under adverse weather conditions, varying plate designs, and complex backgrounds remain. Additionally, extracting readable text from plates remains a challenge due to distortions and font variations, necessitating **OCR i**ntegration for enhanced recognition.

2.2. **Preliminary Investigation:**

Several research studies have explored the use of deep learning for license plate detection and recognition. Setiyono et al. proposed an efficient ALPR system leveraging YOLO for plate detection and CNNs for further recognition. Their approach improved detection under different environmental conditions using data augmentation techniques. However, computational complexity remained high, making real-time deployment challenging [1].

Laroca et al. emphasized dataset diversity and augmentation techniques to improve YOLO-based detection accuracy. While achieving high precision, the model struggled under harsh weather conditions and low-light scenarios, necessitating further enhancements to the YOLO model [2].

Adak et al. introduced a two-step YOLOv3-based pipeline where one model detected vehicles and another localized number plates. While this improved accuracy, sequential detections increased computational costs, making real-time applications difficult. Additionally, small-scale and partially occluded plates remained a challenge [3].

Rayson et al. proposed an efficient and layout-independent ALPR system incorporating OCR for text extraction. Their model achieved a 96.9% recognition rate across eight datasets but struggled under extreme weather conditions, requiring additional training for generalization [4].

Building upon these studies, this project customizes YOLOv8 for accurate plate detection and integrates OCR for real-time recognition, addressing both detection and recognition challenges in complex environments.

2.3. Project Flow Chat:

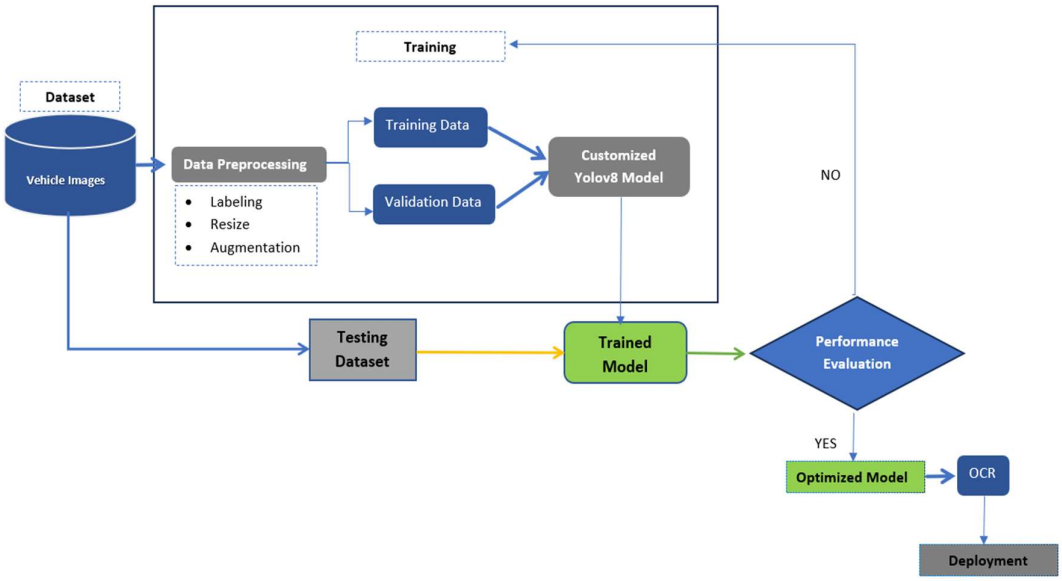


Fig 1: Model Development

3. Methodology

Based on the image Fig 1 we can see the methodology for developing the AI-powered NPD-R system consists of systematic steps ensuring accuracy, efficiency, and robustness. The development process includes data collection, preprocessing, model development, training and optimization, evaluation, and deployment. Each of these stages plays a crucial role in building a reliable and high-performing system for automated number plate detection and recognition.

a. Data Collection: A comprehensive dataset of vehicle images containing number plates will be collected from various sources. The dataset will include images captured under diverse conditions such as daylight, nighttime, fog, and rain to ensure the model generalizes well across different environments. Additionally, number plates from different angles, vehicle types, and regions with varying plate formats will be included. This diversity will help train the model to accurately detect and recognize number plates regardless of external factors

b. Preprocessing: Preprocessing techniques will be applied to enhance image quality and consistency. Noise reduction methods, such as Gaussian filtering, will be used to remove unwanted artifacts. Contrast enhancement techniques, such as histogram equalization, will improve visibility, especially in low-light conditions. The images will be resized to a standard resolution for uniformity during training.

c. Model Development: The NPD-R system will employ a deep learning-based approach using the customised YOLOv8 object detection model for number plate detection. A customized version of YOLOv8 will be developed with additional feature extraction layers like DW convo to focus specifically on number plate detection. In addition to YOLOv8, an OCR engine (such as Tesseract OCR or CRNN) will be integrated to extract alphanumeric characters from detected plates, ensuring precise text recognition. Post-processing techniques, such as language modeling or spell-checking, may be applied to correct potential OCR errors.

d. Training and Optimization: The model will be trained on labeled datasets, with bounding boxes around number plates and corresponding plate number annotations for OCR. The training process will utilize backpropagation and optimization techniques, such as Adam optimizer, to minimize loss functions. Hyperparameter tuning will be conducted to optimize learning rates, batch sizes, and anchor box configurations to enhance accuracy and computational efficiency.

e. Evaluation: To assess the effectiveness of the NPD-R system, multiple performance metrics will be calculated. Precision, recall, and F1-score will be used to measure detection performance. Mean Average Precision (mAP) will evaluate the overall detection accuracy of the YOLOv8 model. Additionally, OCR performance will be assessed using character-level and word-level accuracy to determine the quality of text recognition.

f. Deployment: Once the model achieves satisfactory performance, it will be deployed for real-time applications. Deployment will include edge computing integration for efficient processing on embedded devices such as traffic cameras or mobile systems. Web-based API services using Flask or FastAPI will also be implemented to allow external applications to access the system's functionality.

By following this structured methodology, the AI-powered NPD-R system aims to provide a scalable, accurate, and efficient solution for automated number plate detection and recognition. The integration of YOLOv8 and OCR will enable advanced vehicle identification, contributing to improved traffic management, law enforcement, and automated toll collection systems.

4. System Design

4.1. Software Engineering Paradigm Applied

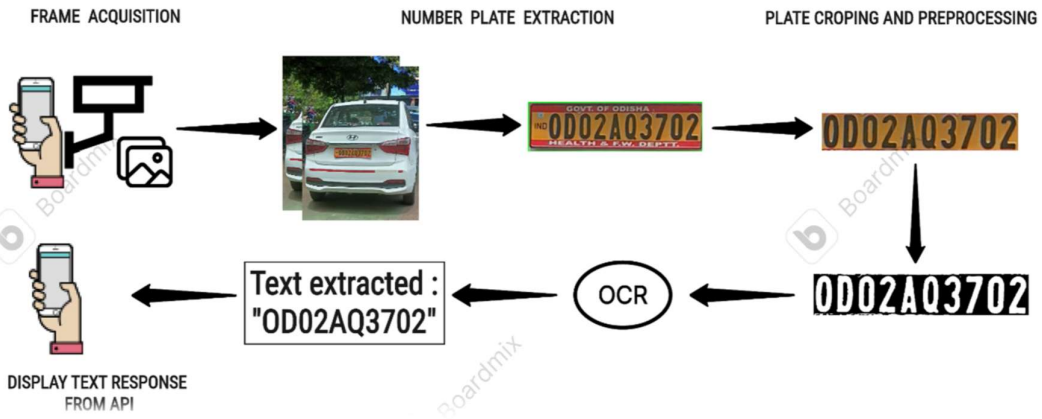
The AI-powered NPD-R project will integrate advanced artificial intelligence (AI) and computer vision technologies to ensure high accuracy and real-time performance. The system is designed to be highly efficient and adaptable, leveraging cutting-edge tools and methodologies for superior vehicle number plate detection and recognition.

The core technologies used in this project are as follows:

- **Deep Learning Frameworks:** PyTorch and TensorFlow will be utilized for model development, training, and deployment. These frameworks provide extensive libraries and tools for building deep learning models, ensuring flexibility and GPU acceleration for optimized performance.
- **Object Detection Models:** YOLOv8 will be the primary model for real-time number plate detection due to its exceptional speed and accuracy. The model's efficiency in detecting small objects within complex backgrounds makes it ideal for this application.
- **OCR Engine:** Tesseract OCR or a deep learning-based OCR system, such as Convolutional Recurrent Neural Network (CRNN), will be integrated for extracting text from detected number plates. This ensures accurate and reliable number recognition under various conditions.
- **Programming Languages:** Python will be the main programming language, complemented by OpenCV for image preprocessing, feature extraction, and transformations. These tools enhance the clarity and usability of input images for improved model accuracy.
- **Hardware:** High-performance GPUs will be employed to accelerate deep learning model training and inference. Additionally, cloud-based computing solutions will be explored for scalability, allowing seamless remote access and improved computational efficiency.
- **Deployment Tools:** Flask or FastAPI will be used for deploying the trained model as a web-based API. Edge computing will be implemented to enable real-time processing on traffic cameras or embedded devices, reducing latency and ensuring rapid response times.
- **Architecture:** The system follows a Client-Server Architecture where the client (user interface) interacts with the server (back-end processing and database).
- **Development Methodology:** The Agile methodology is applied, with iterative cycles and incremental improvements.

By integrating YOLO for plate detection and OCR for recognition, the AI-powered NPD-R system will provide a robust, scalable, and efficient solution for automated vehicle identification. This technology will significantly enhance traffic management, law enforcement, and automated toll collection systems, making transportation infrastructure more intelligent and secure.

4.2. System Architecture



4.3. Modularization Details

The Vehicle Number Plate Detection System can be broken down into the following distinct modules:

- **Frame Acquisition/Image Preprocessing**
  The first step in the system involves preprocessing the input images. This module focuses on enhancing image quality and preparing it for plate detection. The steps include:
  - **Resizing**: The input image is resized to a fixed resolution to ensure that it matches the model's input requirements (e.g., for YOLO). This ensures consistency and helps improve processing time.
  - **Noise Reduction**: Often, input images are noisy due to poor lighting, reflections, or camera quality. Noise reduction techniques like Gaussian blur, median filtering, or bilateral filtering can help smooth out the image and remove unnecessary noise.
  - **Contrast Enhancement**: Enhancing contrast can help make the vehicle's number plate more distinguishable from the background.
- **Number Plate Detection**
  The YOLOv8 with DW convolution efficiently detects number plates in different environments, whether the image quality is high or low, by ensuring minimal computational overhead while maintaining the integrity of the detected plates. The module involves:

        - **Bounding Box Detection**: The model predicts bounding boxes around potential plates, providing the coordinates (x, y) of the plate.
        - **Post-Processing**: After detection, the bounding boxes are refined, and irrelevant objects (non-plate areas) are filtered out. This ensures that the detected area corresponds only to a number plate.
    - **Text Recognition**
      After detecting the number plate, the **Text Recognition** module uses Optical Character Recognition (OCR) to extract the characters from the plate. The OCR engine, such as **Tesseract** or a custom-trained neural network, reads the characters in the bounding box. This step involves:
        - **Character Segmentation**: The characters on the plate are segmented (i.e., individual characters are isolated).
        - **OCR Processing**: The isolated segments are passed through an OCR engine, which converts the visual characters into machine-readable text (e.g., "ABC123").
        - **Error Correction**: The system may include a post-processing step to correct common OCR errors (like misread characters due to noise).
    - **User Interface (UI)**
      The **User Interface** module provides a platform (web or mobile app) where users can interact with the system. This module includes:
        - **Image Upload**: Users can upload an image of the vehicle to the system for processing.
        - **Result Display**: After processing, the system displays the detected number plate and, optionally, vehicle information retrieved from a database. The UI should be designed to provide feedback on the processing status, allowing users to see results quickly and clearly.
        - **User Experience**: The UI should be intuitive, easy to navigate, and provide an option to upload multiple images or retry with different images.

4.4. Data Integrity and Constraints

Data Integrity is crucial in ensuring that the system operates reliably and provides accurate results. This can be maintained through the following constraints:

- **Plate Uniqueness**: The system must ensure that no duplicate vehicle numbers are stored in the database. This can be enforced by implementing a unique constraint on the number plate field within the database.

- **Valid Plate Formats**: Number plates should adhere to a standard format (e.g., "0D 02 AY 9999" or "TS 01 A 1111"). The system must validate the format of the detected plate to ensure that it follows the expected pattern.
- **Data Consistency**: The system must check the consistency of vehicle records (e.g., a plate number should not belong to two different vehicles) to ensure that the database contains reliable and non-contradictory information.

4.5. User Interface Design

A **clean, intuitive UI** is essential for user engagement. The user interface should:

- **Provide Clear Instructions**: The user should easily understand how to upload an image and what to expect as a result.
- **Feedback Mechanism**: Show the processing status (e.g., loading spinner, success/failure message).
- **Error Handling**: If the uploaded image cannot be processed (e.g., unreadable or unclear), the system should prompt the user with an error message and offer guidance on improving the image.
- **Results Display**: Once the number plate is detected, the system should display the plate text along with any associated vehicle details (if available). The UI should ensure these results are clearly legible and easy to interpret.

4.6. Test Cases

- **Unit Test Cases**:
  - **Image Loading**: Ensure that the system correctly handles different image formats (JPEG, PNG) and resizes the image to the required dimensions without errors.
  - **Plate Detection**: Test the detection accuracy for different vehicle types, environments, and image qualities. Ensure that the bounding boxes correctly identify the number plate.
  - **OCR Functionality**: Validate that OCR extracts the correct text from a variety of plates and can handle different fonts, sizes, and plate designs.
  - **Database Integrity**: Test that number plates are correctly stored in the database without duplication and that queries return the correct data.
- **System Test Cases**:
  - **End-to-End Workflow**: Test the entire process from uploading an image to displaying the recognized number plate. Ensure all modules interact correctly and handle edge cases (e.g., unclear plates, multiple plates in a single image).

- **Performance Testing**: Test the system's responsiveness under load, such as processing multiple images simultaneously.
- **Error Handling**: Test the system's behavior when an error occurs (e.g., OCR failure, no plate detected). Ensure proper error messages are shown to the user.

These detailed test cases ensure that each function of the system works as expected and the full system functions cohesively.

5. Coding

```
# Load YOLO Model
logger.info("Loading YOLO model...")
license_plate_detector = YOLO(r'./weights/bestlicence.pt')
license_plate_detector.model = license_plate_detector.model.to(device)
license_plate_detector.model.eval()
logger.info("YOLO model loaded successfully.")
```

```
logger.info("Initializing OCR reader...")
text_recognizer = TextRecognizer()
logger.info("OCR reader initialized successfully.")
```

```
# Define the main detection function
def detect_number_plate(frame):
    """
    Detects a number plate in the given frame, applies transformations, and extracts text.
    """
    start = time.time()
    logger.info("Starting number plate detection.")

    logger.info("Running YOLO inference on the input frame...")
    with torch.inference_mode():  # Ensure no gradients are computed
        results = license_plate_detector.predict(frame, device="cpu")  # Explicitly set the d

    # results = license_plate_detector(frame)
    best_detection = None
    plate_confidence = 0.2
    prediction = {"confidence": 0}
    filtered_text = "No Plate Detected : XXXXXXXXX"
    expanded_region = None
    masked_region = None
    transformed_plate = None
    plate_image_cropped = None

    # Check for detections
    if results[0].boxes is None or len(results[0].boxes.data) == 0:
        logger.info("No detections found.")
        return filtered_text, prediction["confidence"], []
```

```python
# Perform OCR on the cropped plate
prediction = text_recognizer.read(plate_image_cropped)
filtered_text = ''.join(re.findall(r'[A-Z0-9]', prediction["text"]))
logger.info(
    f"Predicted text: {prediction['text']} | Filtered text: {filtered_text} | Confidence: {prediction['confidence']}"
)

# Validate and format the detected text
best_text = filtered_text.strip()
text_confidence = int(prediction["confidence"] * 100)

if license_complies_format(best_text):
    text = format_license(best_text)
    state_code = state_code_formating(text[:2])
    plate_number = text[2:]
    best_text = combine_state_code_and_plate(state_code, plate_number)
    logger.info(f"Formatted license plate text: {best_text}")
else:
    logger.warning("License plate format non-compliant.")

# Log inference time
end_time = int((time.time() - start) * 1000)
logger.info(f"Inference time: {end_time} milliseconds")
```

5. References

[1] Setiyono, Budi, Dyah Ayu Amini, and Dwi Ratna Sulistyaningrum. "Number plate recognition on vehicle using YOLO-Darknet." *Journal of Physics: Conference Series*. Vol. 1821. No. 1. IOP Publishing, 2021.

[2] Laroca, Rayson, et al. "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector." *IET Intelligent Transport Systems* 15.4 (2021): 483-503.

[3] Adak, Rajdeep, et al. "Automatic number plate recognition (ANPR) with YOLOv3-CNN." *arXiv preprint arXiv:2211.05229* (2022).

[4] Laroca, Rayson, et al. "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector." *IET Intelligent Transport Systems* 15.4 (2021): 483-503.

[5] Gonçalves, Gabriel Resende, et al. "Real-time automatic license plate recognition through deep multi-task networks." *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*. IEEE, 2018.

[6] Quraishi, Abdul Awal, Farid Feyzi, and Asadollah Shahbahrami. "Detection and recognition of vehicle licence plates using deep learning in challenging conditions: a systematic review." *International Journal of Intelligent Systems Technologies and Applications* 22.2 (2024): 105-150.

[7] Selmi, Zied, Mohamed Ben Halima, and Adel M. Alimi. "Deep learning system for automatic license plate detection and recognition." *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*. Vol. 1. IEEE, 2017.

[8] Jamtsho, Yonten, Panomkhawn Riyamongkol, and Rattapoom Waranusast. "Real-time license plate detection for non-helmeted motorcyclist using YOLO." *Ict Express* 7.1 (2021): 104-109.

[9] Moussaoui, Hanae, et al. "Enhancing automated vehicle identification by integrating YOLO v8 and OCR techniques for high-precision license plate detection and recognition." *Scientific Reports* 14.1 (2024): 14389.

[10] Al-Hasan, Tamim Mahmud, Victor Bonnefille, and Faycal Bensaali. "Enhanced YOLOv8-Based System for Automatic Number Plate Recognition." *Technologies* 12.9 (2024): 164.

[11] Chopade, Rohan, et al. "Automatic Number Plate Recognition: A Deep Dive into YOLOv8 and ResNet-50 Integration." *2024 International Conference on Integrated Circuits and Communication Systems (ICICACS)*. IEEE, 2024.

[12] Amin, Arslan, et al. "Next-Generation License Plate Detection and Recognition System Using YOLOv8." *2023 IEEE 20th International Conference on Smart Communities: Improving Quality of Life using AI, Robotics and IoT (HONET)*. IEEE, 2023.