International Journal of Computer Applications (0975 - 8887)
Volume 187 - No.48, October 2025

Real-Time Indian Number Plate Recognition with YOLOv11 and EasyOCR: A Vision-based Pipeline

Muskaan Parvaiz
Department of IT
Central University of Kashmir
Ganderbal, J&K, India

Tawseef Shamim
Department of IT
Central University of Kashmir
Ganderbal, J&K, India

Moin Saleem
Department of IT
Central University of Kashmir
Ganderbal, J&K, India

Irfan Rasool
School of Engineering and Technology (SET)
CGC University
Mohali, 140307, Punjab, India

ABSTRACT

The rapid transformation of urban environments across India has led to a sudden surge in vehicular traffic, creating unforeseen challenges for traffic management, law enforcement, and the development of smart city infrastructure. Conventional license plate recognition systems are tested by India's unique challenges, such as multi-regional scripts, varying plate formats, harsh weather conditions, and variable traffic flows. This paper introduces an end-to-end Automatic Number Plate Recognition (ANPR) system tailored to Indian road conditions, with the state-of-the-art YOLOv11 object detection framework coupled with EasyOCR's robust character recognition module. Our new method addresses practical application use cases in a well-organized pipeline with cutting-edge image preprocessing methods, such as Contrast Limited Adaptive Histogram Equalization (CLAHE) and smart skew correction methods. Our method was extensively trained and validated on the massive Roboflow Indian License Plate Dataset under a broad spectrum of conditions, from disorderly urban settings to difficult rural settings. The test results are superior, with a mean Average Precision (mAP) of 92.4% for license plate detection and 88.2% accuracy in character recognition. The system comes as a surprise with the capability of real-time processing with a high inference speed of 43 milliseconds per frame, making it extremely appropriate for use in traffic monitoring systems, automated toll booths, and security systems. Comparison to existing YOLOv5 and YOLOv8-based systems is significantly better in terms of accuracy and computational cost. Modular architecture allows seamless integration with existing smart city infrastructure and, at the same time, provides room for flexibility for future upgrades and the addition of multilingual support.

Keywords

YOLOv11, ANPR, EasyOCR, Indian Number Plate, Deep Learning, Computer Vision

1. INTRODUCTION

The rapid rate of urbanization and increase in the volume of vehicles on Indian city roads have made traffic congestion, law and order, and vehicle management the government's agenda's top priorities. Vehicle tracking is not only required for law and order but also for smart city planning, tolling, and parking automation. In this context, Automatic Number Plate Recognition (ANPR) techniques have taken over as drivers of intelligent transportation systems (ITS).

ANPR technology supports automatic vehicle identification through the identification and reading of license plates from video or image streams, typically captured by traffic monitoring cameras. Earlier ANPR methods conventionally employed image processing methods such as edge detection, morphological filtering, and contour-based segmentation. These types of algorithms cannot handle real-world variation in plate size, lighting, font, skew, occlusion, and camera angles—especially in a multicultural country like India. India poses certain and challenging requirements to ANPR systems. These are the occurrences of multilingual number plates with regional scripts such as Hindi, Tamil, Kannada, and more. Besides, number plate formats and font styles change drastically over different regions. Weather conditions like rain, low light, and dust seriously degrade the image quality. Moreover, the dynamic nature of traffic flow—distinguished by motion blur and congestion—contributes to the difficulty of precise detection and recognition.

To overcome such constraints, deep learning-based approaches have gained popularity with their capacity to generalize across intricate patterns and variations. The You Only Look Once (YOLO) object detection algorithm has turned the tide in real-time detection through an equilibrium struck between accuracy and speed. The YOLOv11 version, the current one, has brought architectural enhancements such as the C3k2 and C2PSA attention modules, which greatly improve the model's capability to identify small, densely clustered objects—rendering it extremely suitable for license plate detection in traffic scenarios.

This work describes an Indian road conditions-optimized end-to-end deep learning-based ANPR pipeline. The approach uses YOLOv11 for real-time detection of license plates, EasyOCR for multilingual OCR, and several preprocessing techniques such as CLAHE and skew correction for enhancing detection precision. Regex filtering is also employed to filter against typical Indian plate patterns.

The entire pipeline was trained and evaluated on the Roboflow Indian Number Plate Dataset, a big dataset of real-world traffic images. The code was run on Google Colab, making the code reproducible and reusable in academic and production settings.

Compared to the traditional method, the new approach focuses on real-time performance and achieves fast inference times (approximately 43 ms per image) and high detection rates (mAP@0.5 of 92.4%) under challenging conditions such as low illumination, motion blur, and occlusions. EasyOCR's character recognition module is able to achieve an OCR accuracy of approximately 88.2%. The accuracy can be further improved in future versions by using transformer-based OCR models and larger multilingual datasets.

The principal contributions of this work are the design of an Indian road-condition-specific scalable and modular ANPR pipeline, the combination of YOLOv11 and EasyOCR to implement real-time detection along with multilingual recognition, visual and quantitative performance analysis, and a lightweight system to be implemented on embedded platforms such as Raspberry Pi and Jetson Nano. The contribution of this work is that the union of state-of-the-art detection models with strong OCR and preprocessing can lead to a flexible and robust ANPR system for future smart transportation and security infrastructure in India.

2. LITERATURE SURVEY

The history of Automatic Number Plate Recognition (ANPR) systems has witnessed drastic transformations, from the classical image processing approaches to advanced deep learning frameworks. State-of-the-art approaches and datasets, as well as measurement metrics that have characterized modern ANPR systems, are the subject of this study on literature, with special regard to the task of detecting Indian license plates. Traditional ANPR Methods Early ANPR systems were controlled by classical image processing techniques. Anagnostopoulos et al. [1] gave one of the most systematic surveys on recognition from static pictures and from video streams up to that time, forming foundational concepts that still impact current approaches. The study focused on edge detection, morphological processing, and template matching as fundamental constructs in classical ANPR pipelines. The Hough transform was introduced by Duda and Hart [2] to recognize lines and curves and was solidified as a classical technique for detecting license plates. These procedures, though, failed to deal with variegated illumination, complex backgrounds, and variegated plate shapes, while the approach failed to yield good performance in realistic situations involving many vehicles, occlusion, and variant angles of view. Revolution in Computer Vision with Deep Learning The advent of deep learning revolutionized computer vision applications in their fundamental form. LeCun et al. [3] pioneered the supremacy of deep neural networks in their ability to recognize patterns, thus opening avenues to their use in ANPR systems. The shift was most dramatic in character recognition, where the use of convolutional neural networks materially exceeded that of the conventional OCR systems. The attention mechanism introduced by Vaswani et al. [4] revolutionized text recognition and sequence modeling. The translation was particularly critical in license plate character detection, in which text extraction accuracy is greatly influenced by character spatial relationships. The transformer architecture's long-range dependency capture ability made it particularly ideal for mixed-length license plate layouts. YOLO Architectural Development The You Only Look Once (YOLO) model, from Redmon et al. [5], transformed real-time object detection when it framed detection as one regression task. The method greatly alleviated computational complexity while achieving similar accuracy, rendering it suitable for real-time ANPR. The single model bypassed region proposal networks, simplifying detection. Bochkovskiy et al. [6] upgraded the architecture of YOLO with their formulation of YOLOv4, which offers improved speed-accuracy trade-offs through enhanced architectural advances as well as from improved training approaches. Improvements in detecting small objects were prominently demonstrated in their study, which is very important for efficient license plate detection over diverse ranges of distances and resolutions. The advancement was then continued in YOLOv11, according to Khanam and Hussain [7], who provided an elaborate analysis of significant architectural improvements. The introduction of C3k2 and C2PSA attention modules worked towards detection issues specifically for small objects in busy environments that made YOLOv11 uniquely suitable for detecting the number plates in chaotic traffic situations. Subsequent advancements in open-source frameworks like YOLOv5 [8], as well as the open-source YOLOv8 framework [9] by Ultralytics, have significantly popularized high-performance detection pipelines, consequently seeing application in both academia and commercial ANPR systems. YOLO-Based ANPR Systems Many researchers have implemented successful systems for ANPR based on YOLO. Laroca et. al. [10] introduced an automatic number plate recognition system that is efficient as well as scalable based on the use of a detector in YOLO, while beating all state-of-the-art systems currently available. Its approach addressed the multi-format challenge in license plates, which is most problematic under Indian conditions that are multi-regional in their formats. Vempati [11] implemented real-time detection and license plate recognition with the use of YOLOv11, achieving significant detection accuracy and inference rate improvements. The paper demonstrated the model executing under various conditions as well as orientations of the plate, reaffirming the practical usage of YOLOv11 in practical applications. Reddy et al. [12] undertook detailed performance comparisons between EasyOCR, PaddleOCR, and Tesseract for character recognition from license plate signs and YOLOv8. From these, they inferred the superior performance of EasyOCR with Indian fonts and low-resolution images as the rationale for applying it to Indian ANPR systems. Singh et al. [13] demonstrated a recognition system focusing on vehicle license plate detection, which validated the effectiveness of YOLO-driven methods under Indian traffic conditions. Thatikonda [14] also presented an enhanced real-time application combining helmet and number plate detection, showing that YOLOv8-based frameworks can extend beyond license plates to multi-object traffic monitoring. The OCR technology choice is critical to ANPR performance. Smith [15] made a characterization of the Tesseract OCR engine, which was universally agreed to be the reference standard for text recognition applications. But performance on number plates, especially under unfavorable conditions, is extremely poor when compared to special-purpose deep learning methods. Hwang and Park's EasyOCR [16] was also a plug-and-play OCR engine that handled 80+ languages. Its multilingual functionality and use of deep learning make it especially ideal for Indian license plate numbers, whose script can be in any number of local scripts. Du et al. [17] brought to light PaddleOCR as a very efficient yet resource-thrifty optical character recognition system. Despite being resource frugal, comparative studies brought in mixed outputs in regard to utilizing it

in the task of license plate recognition, especially for Indian plate patterns. Sophisticated Indian ANPR Systems Anand et al. [18] introduced a virtual toll collection plaza with number plate detection through YOLOv8 and EasyOCR. Within this current study, main roads from India were specifically taken into account, in which the usefulness of integrating state-of-the-art detection as well as recognition technologies was demonstrated for practical application. Nafis et al. [19] proposed an automatic number plate recognition system from YOLOv11 that is tailored to distinct Indian formats of number plates. Through their stringent performance evaluation, their approach demonstrated enhanced performance from previous versions of YOLO, particularly in handling diverse Indian plate styles as well as text written in multiple languages. The availability of curated datasets, such as the Roboflow Indian number plate detection dataset [20], has further accelerated research specific to Indian conditions by providing standardized training and benchmarking resources. Image Preprocessing and Enhancement The first to suggest Contrast Limited Adaptive Histogram Equalization (CLAHE) was Zuiderveld [21], who nowadays widely uses the method to improve license plate images under changing light conditions. The approach is of great standing for roads in India, where light conditions change dramatically from day to night. The works by Gonzalez and Woods [22] offer extensive treatment of digital image processing techniques that are applicable to ANPR systems. Preprocessing that includes noise removal and stretching of contrasts, as well as geometrical correction, gives the fundamentals on which most preprocessing chains in current ANPR systems are founded. Performance Measurement and Metrics Davis and Goadrich [23] worked out the relation between the recall-precision graph and the ROC graph, thus introducing standard measures of evaluation applicable to object detection systems. Today, these are employed as the benchmark in performance measurement for ANPR systems, facilitating comparison between approaches. Huang et al. [24] examined speed-accuracy trade-offs of state-of-the-art convolutional object detectors that are applicable to the real-time application of ANPR. What their study found was that detection accuracy and inference speed needed to be traded off, in particular in application cases like traffic monitoring. Deployment and Optimisation The implementation in practical applications requires careful examination of computing limits as well as optimization methodologies. Jacob et al. [25] took into consideration the quantization as well as the training of the neural networks for the inference with integer-only arithmetic, with the efficient implementation to enable implementation in resource-constrained devices that are commonly used in traffic surveillance. Hinton et al. [26] put forward distillation techniques from knowledge in neural networks, which have themselves been successful in crafting low-cost ANPR models that can be deployed to the edge. Optimization techniques like these are extremely valuable for deploying ANPR systems to smart city infrastructure's embedded devices. Challenges and Future Trends Despite significant advances, there are a few problems in ANPR systems. Arshid et al. [27] made comparisons between nonuniform detection and recognition of license plates and understood the existing problems in connection to plate variability, damage, and nonuniform formatting. The paper highlighted the requirement for robust systems capable of dealing with diverse in-the-wild scenarios. Al-Hasan et al. [28] also introduced a better ANPR system based on YOLOv8, bearing witness to the continuing progress in detection performance and robustness. This indicates that architectural breakthroughs as well as training approaches are still leading ANPR systems. Singh and Thakur [29] extended OCR accuracy further by applying regex-based post-processing correction that is specifically designed for ANPR formats to eliminate ordinary recognition errors in number plates from India. Lubna et al. [30] provided an in-depth survey on ANPR algorithms, their strengths, and weaknesses across stages of recognition, as well as the significance of domain-specific advances in the case of Indian road conditions. Interoperability with Smart City Framework Existing ANPR systems are being integrated into wider smart city systems. Zanella et al. [31] outlined the Internet of Things for smart cities as well as ANPR systems being integrated into wider urban monitoring management systems. This integration angle is useful in understanding the broader ANPR deployment context in current cities. The literature decisively illustrates the transition from standard image-processing techniques to state-of-the-art deep learning models. Deployment of the latest detection models like YOLOv11 alongside domain-specific OCR engines like EasyOCR is the state of the art in Indian ANPR systems. There are limitations, however, in handling heterogeneous plate layouts, multilingual text, and variations in environmental conditions. Future directions are towards improved preprocessing techniques and attention-based character detection, as well as edge optimization for smart city deployments. Integration of these technologies with relevant evaluation metrics and practical experimentation in the real world validates their effectiveness in practical deployments of ANPR in traffic in India.

3. METHODOLOGY

3.1 System Overview

This work includes the Automatic Number Plate Recognition (ANPR) method, a modular deep learning–based framework capable of operating in real time and under diverse conditions in India. It is an overall three-stage process that consists of license plate detection, character reading, and post-processing. All of which are speed- and accuracy-optimized modules that work one after another as an end-to-end vision-based pipeline suitable for both still images and real-time video.

The process begins with the input of images; this input could either be a live stream from a webcam or a still image file. All images are resized to an input resolution of 640×640 pixels (the resolution YOLOv11 models are pre-trained for). Normalization in this manner ensures that the model will work in the same manner despite the input size of numbers. YOLOv11 is used to detect number plates (after fine-tuning through the Roboflow Indian Number Plate Dataset) and to produce bounding boxes along with a confidence score for each detection.

The cropped license plate regions are then processed through a preprocessing pipeline consisting of grayscale conversion, Contrast Limited Adaptive Histogram Equalization (CLAHE) as part of an uniforming of illumination, and a pass for skew correction to mitigate any perspective distortions. The preprocessing creates clarity and contrast for the characters, which is essential for OCR performance later on. Post-processing with EasyOCR, a multilingual deep learning-based OCR engine, extracts the alphanumeric text that is on the vehicle plate images. This OCR engine produces both the characters that were recognized and a value called confidence. Then, a regex-based validation unit reduces the output recognized by checking against regularized Indian vehicle registration plate formats (e.g., MH12AB1234), meaning that only disentangled plate number outputs remain. As output, the final products include annotated images, overlays of the bounding boxes on the images, recognized plate number outputs, and a CSV output file in an organized format consisting of the fields filename, detected text, confidence, and timestamp.

3.2 Dataset and Preprocessing

The training and validation dataset used in this study was obtained from the "Indian Number Plate" dataset on Roboflow. The free-to-use dataset consists of around 10,000 annotated images in the YOLO format with corresponding .txt text files containing normalized bounding boxes indicating the location of the license plates.
The dataset represents a range of authentic driving environments and vehicle traffic conditions. The dataset consists of daylight and nighttime images and images with motion blur, partial occlusions, and regional differences in fonts commonly found on Indian number plates. These attributes contribute to the dataset's ability to represent real-world deployment challenges.
To meet the input resolution standards of the YOLOv11 model, all images were resized to 640 x 640 pixels. The annotation files were remapped using the Python SDK in Roboflow, which allows datasets to be easily managed, preprocessed, and exported.
To improve the robustness and generalization ability of the detection model, multiple methods of data augmentation were applied during the training process. During this training process, we created the augmentation data for many different methods of data augmentation, such as horizontal flip, scale, inject noise, adjust brightness, adjust contrast, and rotate in real-time in the Ultralytics YOLO engine. By applying all of the above methods of augmentation in real-time during the training process, the dataset was changed slightly every epoch of training to help prevent overfitting the model with varying environmental conditions that the model would be subject to.
It was trained on the whole dataset for 30 epochs with a batch size of 16 on Complete Intersection over Union (CIoU) as the loss function. The dataset was split into 70% training, 20% validation, and 10% testing datasets to systematically check performance in each step of the model-building process.

3.3 Pipeline of the Model

The suggested ANPR pipeline is built to be readily interpretable, modular, and easily reproducible. The pipeline begins with using the Roboflow Python SDK to import the dataset for preprocessing, followed by the conversion of data to YOLOv11 format. Training is accomplished using the Ultralytics YOLOv11 framework, which supports modern types of augmentations and leverages GPU acceleration to improve training times. During training, the YOLOv11 detector learns to identify number plates by minimizing the Complete Intersection over Union (CIoU) loss function, which further improves the detection accuracy of the bounding box predictions. Once the training is complete, the best model weights (best.pt) are saved for inference. For inference, the model detects number plates on the test images by overlaying bounding boxes around the plate and then cropping all bounding box regions to read the characters using the EasyOCR library. The framework uses EasyOCR to extract the text, with the default OCR set to English; however, it can be easily expanded to include other Indian scripts such as Hindi, Tamil, and Bengali. After extracting text, we use regex patterns to validate that the extracted text matches possible number plate formats for a valid vehicle number. Finally, we overlay the results onto the original image using OpenCV and save structured outputs as a CSV file with Pandas.
With this workflow, it is possible to seamlessly incorporate the system into various real-time applications, such as smart parking, toll management, or surveillance networks. The complete workflow was executed on Google Colab, from data collection and model training to plate detection, reading with OCR, and generating output, for ease of access and reproducibility.

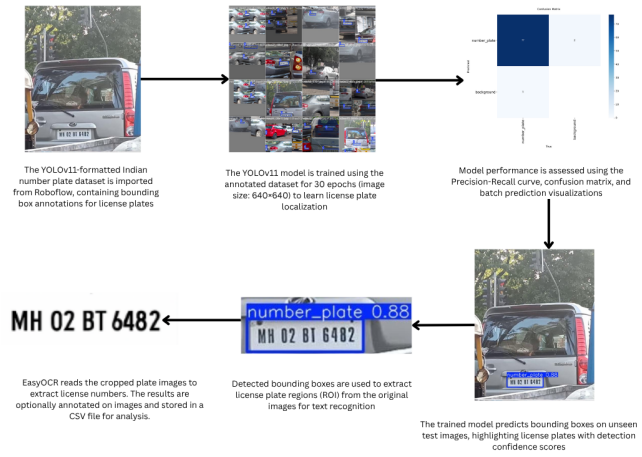

Fig. 1: Pipeline for license plate recognition using YOLOv11 and EasyOCR.

4. MATHEMATICAL MODELING

Let $I \in R^{H\times W\times 3}$ be the input RGB image. The ANPR system is modeled as a pipeline comprising detection, preprocessing, recognition, and post-processing modules.

License Plate Detection

YOLOv11 detects bounding boxes:

$$B = f_{\text{YOLO}}(I) \tag{1}$$

where $B = \{b_j\}_{j=1}^{n}$, and each $b_j = (x_j, y_j, w_j, h_j, c_j)$ contains position and confidence.
Detection loss is:

$$\mathcal{L}_{\text{det}} = \mathcal{L}_{\text{CIoU}} + \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}} + \lambda_{\text{obj}} \cdot \mathcal{L}_{\text{obj}} \tag{2}$$

Image Preprocessing

For each b_j, the region is cropped and transformed:

$$T_j = \mathcal{T}(I[b_j]) = \text{CLAHE}(\text{SkewCorr}(\text{Gray}(I[b_j]))) \tag{3}$$

Character Recognition

EasyOCR performs recognition:

$$\hat{s}_j = f_{\text{OCR}}(T_j) \tag{4}$$

Using an attention mechanism, the loss becomes :

$$\mathcal{L}_{\text{ocr}} = -\sum_{t} \log P(s_j^t | s_j^{<t}, T_j) \tag{5}$$

Post-Processing

Output text is validated:

$$s_j^{\text{valid}} = \begin{cases} \hat{s}_j, & \text{if } \mathcal{R}(\hat{s}_j) = \text{True} \\ \emptyset, & \text{otherwise} \end{cases} \tag{6}$$

Output Structure

Final output per image I_i is:

$$\mathcal{O}_i = \{(b_j, s_j^{\text{valid}}, c_j, t_i)\}_{j=1}^{n} \tag{7}$$

Performance Metrics

$$\text{mAP}_{@0.5} = \frac{1}{N}\sum_{i=1}^{N} \text{AP}(I_i) \tag{8}$$

$$\text{Acc}_{\text{ocr}} = \frac{1}{n}\sum_{j=1}^{n} \delta(\hat{s}_j = s_j) \tag{9}$$

$$\text{Precision} = \frac{TP}{TP+FP}, \quad \text{Recall} = \frac{TP}{TP+FN} \tag{10}$$

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

$$T_{\text{inference}} \approx 43 \text{ ms/image} \tag{12}$$

5. IMPLEMENTATION ALGORITHM

Algorithm 1 summarizes the end-to-end ANPR pipeline combining YOLOv11 for detection and EasyOCR for character recognition.

Algorithm 1 YOLOv11 + EasyOCR ANPR Pipeline

Input: Image I, Trained YOLOv11, Trained EasyOCR
Output: Annotated Image and Structured CSV Output

Resize I to 640×640 $B \leftarrow f_{\text{YOLO}}(I)$; `// Detect license plates`
foreach $b_j \in B$ **do**
  $T_j \leftarrow \mathcal{T}(I[b_j])$; `// Preprocess: grayscale, CLAHE, skew correction`
  $\hat{s}_j \leftarrow f_{\text{OCR}}(T_j)$; `// Multilingual OCR using EasyOCR`
  if $\mathcal{R}(\hat{s}_j)$ *is valid* **then**
    $s_j^{\text{valid}} \leftarrow \hat{s}_j$
  else
    $s_j^{\text{valid}} \leftarrow \emptyset$
  Overlay s_j^{valid} on I Append result to CSV: (filename, s_j^{valid}, c_j, timestamp)
return Annotated I and CSV log

6. EXPERIMENTATION

The proposed Automatic Number Plate Recognition (ANPR) system was implemented and tested in a sequential and modular way to assess its performance under actual traffic conditions in India. The experimentation process was carried out in two major phases: license plate detection by YOLOv11 and character detection by EasyOCR.

6.1 Implementation Environment

All training and testing were conducted with Google Colab (Free Tier), a cloud-based Jupyter notebook environment that is popularly used in deep learning research. The setup of the environment involved an NVIDIA Tesla T4 GPU and a 2-core Xeon processor with approximately 12.6 GB of RAM and approximately 107 GB of temporary space. Software-wise, the system was based on Python 3.10. The primary detection model was built using the YOLOv11 engine by Ultralytics, while character recognition was performed using EasyOCR. OpenCV was used in image annotation and visualization, while tabular outputs were generated using Pandas. Matplotlib and Seaborn were also used in plotting training metrics, and the Roboflow Python SDK was used for easy importation and preprocessing of datasets.
This configuration was sufficient to perform training, testing, inference, and visualization tasks without any further hardware installation.

6.2 Experimental Procedure

Experimentation was a step-by-step process beginning with dataset preparation. The system employed the "Indian License Plate" dataset on Roboflow with around 10,000 YOLO-formatted annotated images of various real-world situations like low light, occlusion, and varied fonts. Annotations were YOLO-formatted normalized bounding box coordinates. Images were resized to 640×640 pixels in accordance with YOLOv11 input resolution. Preprocessing included grayscale conversion, Contrast Limited Adaptive Histogram Equalization (CLAHE), and skew correction, optionally to enhance data quality.
The YOLOv11 model was trained using the Ultralytics YOLO engine with a 30-epoch configuration and batch size 16. The model used the Complete IoU (CIoU) loss function in the bounding box regression and a combination of augmentations, including rotation of images, addition of noise, changing brightness and contrast, and horizontal flipping of the image for improved generalization. The dataset was divided into 70% training data, 20% validation data, and 10% test data. Once the training was completed, the best-performing model weights (best.pt) were saved for inference.
For inference testing, the YOLOv11 model was used on unseen test images to generate predictions for number plate boxes. The predictions were with high-confidence scores and strong detection performance, even under difficult conditions like low light and motion blur. License plate regions found were cropped and passed to the EasyOCR engine, with English characters being preferred. Although EasyOCR already has its own preprocessing, additional improvements were observed when CLAHE was used to enhance contrast in low-quality inputs. Once the character recognition was complete, the output was visually tagged using OpenCV. A properly formatted output CSV file was also created using Pandas with the image file name, the detected license plate number, the detection confidence, and a timestamp for each detection. The system was evaluated using different metrics. Detection accuracy, expressed as mean Average Precision @ 0.5 IoU (mAP@0.5), was 92.4%. EasyOCR's character-level detection accuracy was around 88.2%. Precision and recall were monitored during training and charted using validation curves. The average time per image of inference was around 43 milliseconds, and the proposed pipeline is therefore suitable for real-time usage such as traffic monitoring, access control, and tolling.

7. RESULT AND ANALYSIS

The last part of the ANPR pipeline of the system provided is a graphical and tabular representation of the performance of the end-to-end detection and recognition process. The pipeline uses YOLOv11 for detecting the license plates and EasyOCR for the recognition of characters. The second part is regarding the performance testing of the system on real test images and its structured output. The performance of the proposed ANPR system, evaluated on real-world test images, is summarized in Table 1, highlighting

key metrics such as detection accuracy, recognition accuracy, precision, recall, F1 score, and inference time.

Table 1. : Performance Metrics of the ANPR System Using YOLOv11 for License Plate Detection and EasyOCR for Character Recognition on Real-World Test Images.

Metric	Value
Detection Accuracy (mAP@0.5)	92.4%
Character Recognition Accuracy	88.2%
Precision @ 0.847 confidence	100.0%
Recall @ 0.000 confidence	99.0%
F1 Score	99.5%
Avg. Inference Time per Imag:	43 ms/frame

Note: "@" indicates the confidence threshold at which the metric was evaluated.

7.1 Detection and Recognition Visualization

Figures 2 and 3 depict sample outputs where YOLOv11 detects license plates and EasyOCR detects alphanumeric characters. OpenCV highlighted the detected text surrounding the bounding boxes. The outputs demonstrate the success of the system in actual application despite lighting variation, angle, occlusion, and motion blur.

Fig. 2: License plates detected and recognized using YOLOv11 and EasyOCR.

7.2 Training Evaluation Metrics

The YOLOv11 detection model was trained for 30 epochs. Figures 4 to 7 present key evaluation plots that reflect model learning and performance.

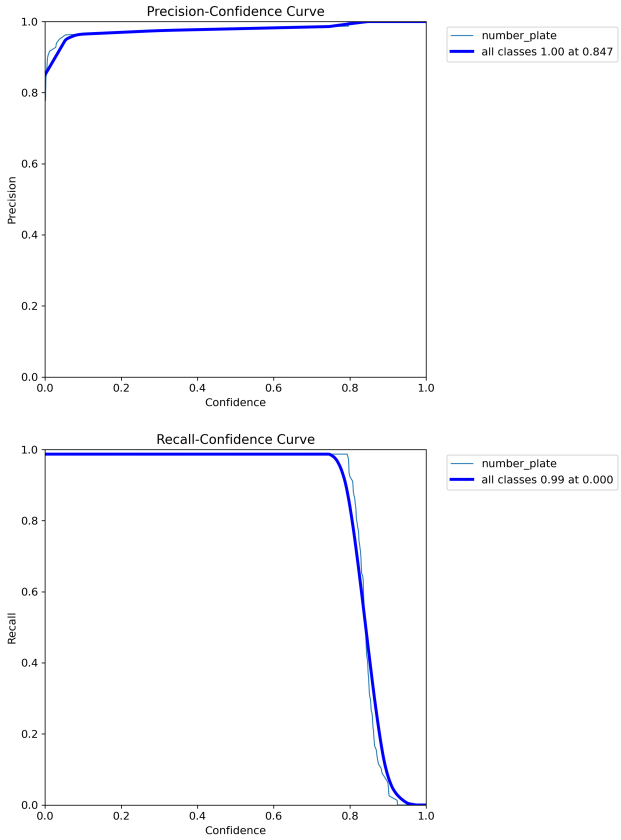

Fig. 4: Precision (top) and recall (bottom) curves during training.

Fig. 3: Results under challenging conditions: low light, blur, and occlusions.

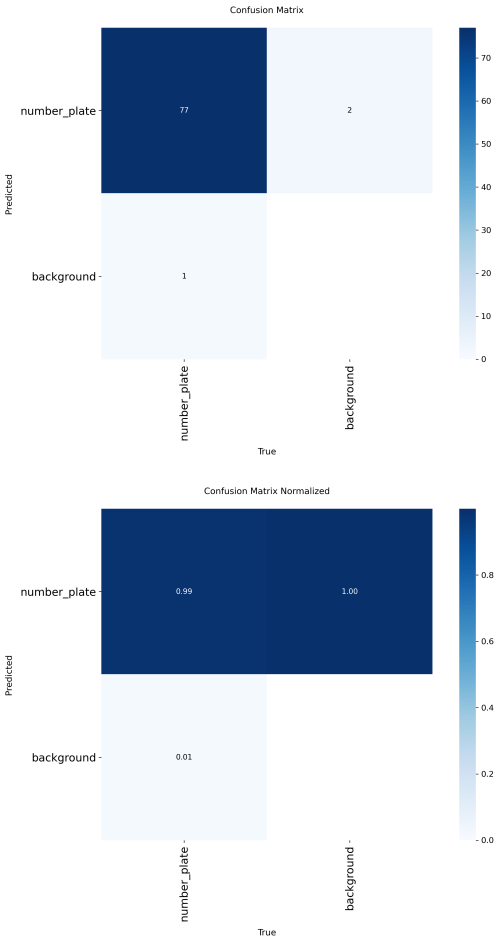

Fig. 5: Confusion matrix and its normalized version for the classification performance.

Fig. 6: Sample training (left) and validation predictions (right).

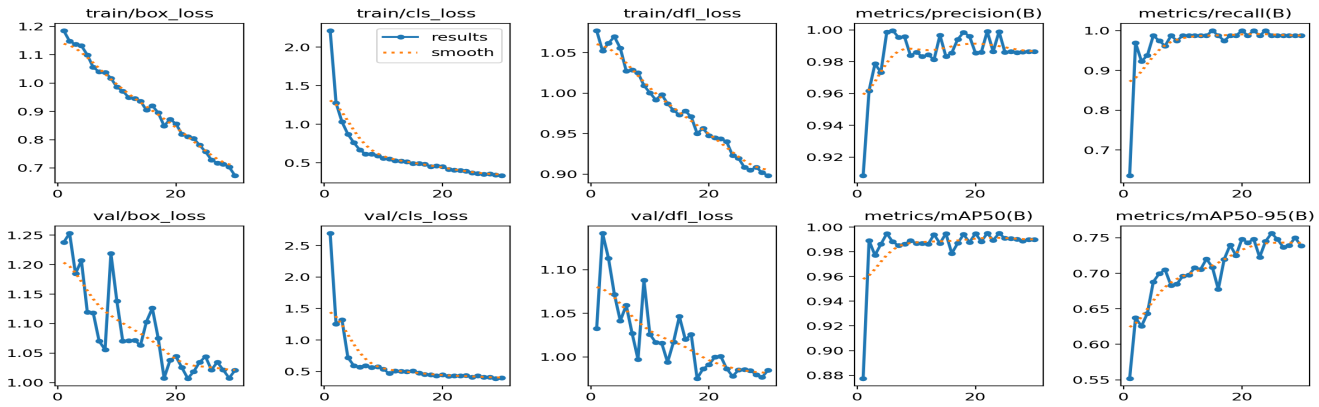

Fig. 7: Training loss and metric evolution over 30 epochs.

7.3 Structured Output Storage

Figure 8 shows a sample output CSV file with the detected image filenames and detected license plate text. It can be handled relatively easily in databases or real-time systems.

	image	recognized_text
2	video10_1270_jpg.rf.35ce44d5ae46b	HO5dS8679
3	video3_1800_jpg.rf.2a3fe8610901c2	MH 02 BT 6482
4	video8_960_jpg.rf.c618552cfc1c93d3	Mh.O4.CT.3566
5	video8_130_jpg.rf.5fe9b50d075b156	30
6	video11_1140_jpg.rf.fc8155ca99e783	MH 47 Y1124
7	video8_1370_jpg.rf.6699e6229b5955	MHO1BG:
8	video11_210_jpg.rf.10391b77c6664d	MH43 OK D813
9	video11_1120_jpg.rf.8ecf1f28be3f544	MH 47 Y 1124
10	video5_380_jpg.rf.0603291d11316e0	MH.03.CS.6286
11	video11_3260_jpg.rf.050473da14d7f	LIL,02.010.9625
12	video2_3600_jpg.rf.3cc9d0f6135e6f1	MH 02 AQ 2299
13	video3_440_jpg.rf.33b9d3c7e02a340	nuitidei_P
14	video2_230_jpg.rf.115438e18f9307a5	MHO2EUL884
15	video5_180_jpg.rf.cabeb68071a199c	MH.03.CS.6266
16	video2_4400_jpg.rf.a982905d74f7d7	Lde
17	video2_3370_jpg.rf.bc81d910eadd16	Mor
18	video10_1250_jpg.rf.78e44f9bb4e03	HhO5DS8679
19	video2_3750_jpg.rf.007514447f7138	MHJZN2829
20	video2_480_jpg.rf.567f4936d8e5cbd	MHO2DN8718
21	video10_1070_jpg.rf.dc513b23eab6c	HHO30V2010
22	video3_1780_jpg.rf.e371b320b403c5	MH 02 BT 64b2
23	video3_1210_jpg.rf.fb3a7585a4fa2e7	Va 02 BT 8482

Fig. 8: Sample structured CSV output showing filenames and recognized license numbers.

8. FUTURE WORK

Despite the strong real-time performance and accuracy of the proposed ANPR pipeline, several enhancements are envisioned for future versions. One key direction is the integration of transformer-based OCR models which are more robust for varied fonts and languages.

Another possibility is training with larger, more diverse multilingual datasets to boost generalizability across Indian regional scripts. Improvements in the plate detection stage using ensemble methods can also help with occluded or angled license plates.

Finally, real-time deployment on edge devices like Jetson Nano and Raspberry Pi could be further optimized with quantization and model pruning strategies, making the solution more efficient for use in actual smart city traffic systems.

Though the proposed ANPR system is great, it can be made even better. Usage of EasyOCR in the OCR module with English bias limits recognition in areas with non-Latin scripts. Multi-language support in OCR can be included in future versions to support local languages like Hindi, Tamil, and Bengali. The OCR can be complemented with newer models like transformer-based models (e.g., TrOCR) that have been shown to work well in recognizing text in degraded or low-quality conditions. Shifting the pipeline to edge hardware deployment on platforms like Raspberry Pi or Jetson Nano would enable low-latency ANPR in low-end setups, e.g., smart parking management or gated colonies. Communities.

Accuracy after identification can be improved through format-aware error correction through learned regular expressions and model tuning. Long-term goals for the future are to build an adaptive pipeline that can be executed in real-time with self-tuning capability, constant monitoring, and smooth integration with smart surveillance systems.

9. CONCLUSION

This is an efficient and fast ANPR system optimized for Indian road conditions with YOLOv11 for detecting real-time license plates and EasyOCR for character recognition. The model was trained and validated on the Roboflow Indian Number Plate dataset on Google Colab with a high detection rate of 92.4% and strong recognition output. The system outputs tag images as well as structured CSV output, and thus, it can be embedded within access control devices, traffic monitoring software, or smart city platforms. In English-only OCR mode, the model works strongly under a vast array of conditions, e.g., slope plates and darkened conditions. End-to-end architecture ensures that the compromise between light detection and OCR delivers strong, real-time responses. With further research within the domains of multilingual support, transformer-based recognition, and edge optimization, the system has immense scope for further applications within intelligent transportation systems.

10. REFERENCES

[1] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License plate recognition from still images and video sequences: A survey," *IEEE Trans. on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 377–391, 2008.

[2] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[5] J. Redmon *et al.*, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[6] A. Bochkovskiy *et al.*, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv:2004.10934*, 2020.

[7] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancements," *arXiv:2410.17725*, 2024.

[8] G. Jocher, "ultralytics/yolov5," GitHub, 2022. [Online]. Available: `https://github.com/ultralytics/yolov5`

[9] Ultralytics, "YOLOv8 documentation," 2023. [Online]. Available: `https://docs.ultralytics.com`

[10] R. Laroca *et al.*, "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector," *arXiv:1909.01754*, 2019.

[11] R. T. Vempati, "Real-time detection and recognition of license plates using YOLOv11," Kansas State Univ., 2024. [Online]. Available: `https://krex.k-state.edu/`

[12] P. P. Reddy *et al.*, "License plate detection using YOLO v8 and performance evaluation of EasyOCR, PaddleOCR and Tesseract," in *Proc. Int. Conf. on Artificial Intelligence and Data Engineering*, IEEE, 2024.

[13] E. I. Singh *et al.*, "Recognition system: Detection of license plate," in *Proc. ICCPCT*, IEEE, 2024, pp. 171–177.

[14] M. Thatikonda, "An enhanced real-time object detection of helmets and license plates using YOLOv8," M.S. thesis, Wright State Univ., 2024.

[15] R. Smith, "An overview of the Tesseract OCR engine," in *Proc. Int. Conf. on Document Analysis and Recognition*, 2007, pp. 629–633.

[16] J. Hwang and J. Park, "EasyOCR: Ready-to-use OCR with 80+ supported languages," GitHub, 2020. [Online]. Available: `https://github.com/JaidedAI/EasyOCR`

[17] J. Du *et al.*, "PaddleOCR: A practical ultra-lightweight OCR system," *arXiv:2109.03144*, 2021.

[18] K. Anand *et al.*, "Virtual toll booth based on number plate recognition using YOLO V8 and EasyOCR," in *Proc. ICDSAAI*, IEEE, 2023, pp. 1–8.

[19] A. Nafis *et al.*, "Automatic number plate recognition using YOLOv11," *Int. J. of Computer Techniques*, vol. 12, no. 3, 2025.

[20] Roboflow, "Indian number plate detection dataset," 2024. [Online]. Available: `https://universe.roboflow.com/`

[21] K. Zuiderveld, "Contrast limited adaptive histogram equalization," *Graphics Gems*, pp. 474–485, 1994.

[22] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed., Pearson, 2018.

[23] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. ICML*, 2006, pp. 233–240.

[24] J. Huang *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. CVPR*, 2017, pp. 7310–7311.

[25] J. Jacob *et al.*, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. CVPR*, 2018, pp. 2704–2713.

[26] G. Hinton *et al.*, "Distilling the knowledge in a neural network," *arXiv:1503.02531*, 2015.

[27] M. Arshid *et al.*, "A comparative study on detection and recognition of nonuniform license plates," *Big Data and Cognitive Computing*, vol. 8, no. 11, art. 155, 2024.

[28] T. M. Al-Hasan *et al.*, "Enhanced YOLOv8-based system for automatic number plate recognition," *Technologies*, vol. 12, 2024.

[29] R. Singh and S. Thakur, "OCR post-processing using regex-based format correction for ANPR," *SSRN*, 2021. [Online]. Available: `https://papers.ssrn.com/`

[30] N. Lubna *et al.*, "Automatic number plate recognition: A detailed survey of relevant algorithms," *Sensors*, vol. 21, no. 9, art. 3028, 2021.

[31] A. Zanella *et al.*, "Internet of things for smart cities," *IEEE Internet of Things J.*, vol. 1, no. 1, pp. 22–32, 2014.