



AI Traffic Violation Detection System

PRESENTED BY:

Chiranjeet (ET22BTHCS012)	Sangamlung (ET22BTHCS036)
Deep Das (ET22BTHCS021)	Simanta (ET22BTHCS043)
Marcel (ET22BTHCS028)	Vikosalie (ET22BTHCS070)

INTERNAL GUIDE:

Dr. Bondita Paul

CONTENT	SLIDE NUMBER
INTRODUCTION	3-5
PROJECT OBJECTIVES	6
LITERATURE REVIEW	7-11
IDENTIFIED RESEARCH GAPS	12
PROPOSED SYSTEM OVERVIEW	13-14
EXPECTED OUTCOMES	15
TOOLS & TECHNOLOGIES	16
PROJECT PLAN & TIMELINE	17
INITIAL TRAINING METRICS & RESULTS	18-25
REFERENCES	26-29
CONCLUSION & FUTURE SCOPES	30

Introduction / Problem Background

AI TRAFFIC VIOLATION DETECTION SYSTEM

AI-based traffic monitoring systems use artificial intelligence, particularly computer vision, to analyze real-time video feeds from cameras. This allows them to automatically identify vehicles, detect traffic violations (like speeding or running red lights), monitor congestion, and gather data to improve traffic flow and safety.

AI-based traffic monitoring systems in India

Major Indian cities like Bengaluru, Delhi, and Hyderabad have successfully implemented AI-driven systems for automated traffic law enforcement..

Challenges in deployment

Despite their success, these commercial systems are prohibitively expensive and require heavy computational infrastructure.

They are often "black-box" solutions, making them unsuitable for educational study or small-scale experimentation.

Introduction / Problem Background

AIM

To develop a simplified, cost-effective learning prototype that replicates the core "detection-to-action" pipeline of professional systems

Unlike commercial products, this project focuses on understanding the *integration* of object detection (YOLOv5), character recognition (OCR), and automation logic

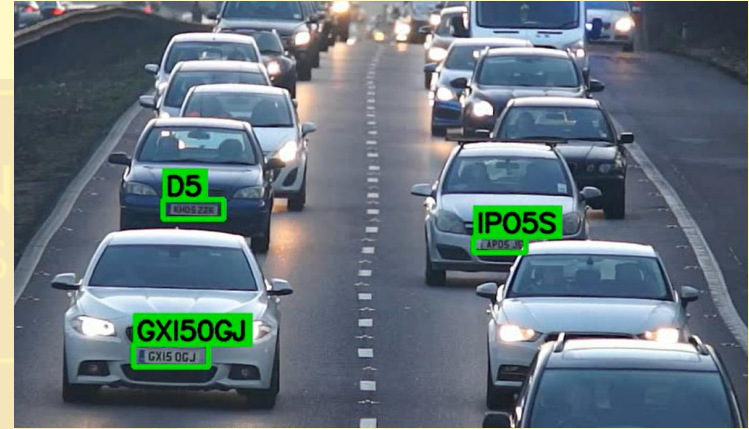
Educational project focus

Our goal is to understand and replicate key AI-based traffic enforcement functionalities within a simplified academic framework.

● Introduction / Problem Background

Automated detection of common violations (SCOPE)

The system can detect **helmetless riding**, **triple riding**, and **signal jumping** using object detection and image analysis algorithms.



Project Objectives

- Learning and Implementing AI-based Traffic Violation Detection
 - **Real-time object detection using YOLOv5:** Understand and implement YOLOv5 for detecting vehicles, helmets, and riders in traffic video streams.
 - **Detect multiple violation types:** Develop detection logic for common violations including helmetless riding, triple riding, wrong-lane driving.
 - **Number plate recognition via OCR:** Use Optical Character Recognition (OCR) to extract vehicle numbers from detected license plates.
 - **Automated e-Challan generation:** Integrate detection and OCR modules to automatically generate challans based on identified violations.
 - **Performance analysis:** Evaluate system accuracy and efficiency under varying lighting and traffic conditions using standard metrics.

Literature Review

- Summary of 33 Papers Read :

Use of CNN and YOLO architectures: Recent studies rely on deep learning models like YOLOv3–v8 [Paper 3, 5] and CNNs for detecting traffic violations [Paper 1, 8, 12, 23] and recognizing number plates [Paper 2, 18, 26, 28, 31].

High accuracy under controlled conditions: YOLO-based models achieve up to 97% accuracy (and higher) for helmet detection [Paper 12, 15, 16], ALPR tasks [Paper 2, 4, 12, 18, 28, 29], and other specific violations [Paper 8, 11, 22] in daylight and well-lit scenarios.

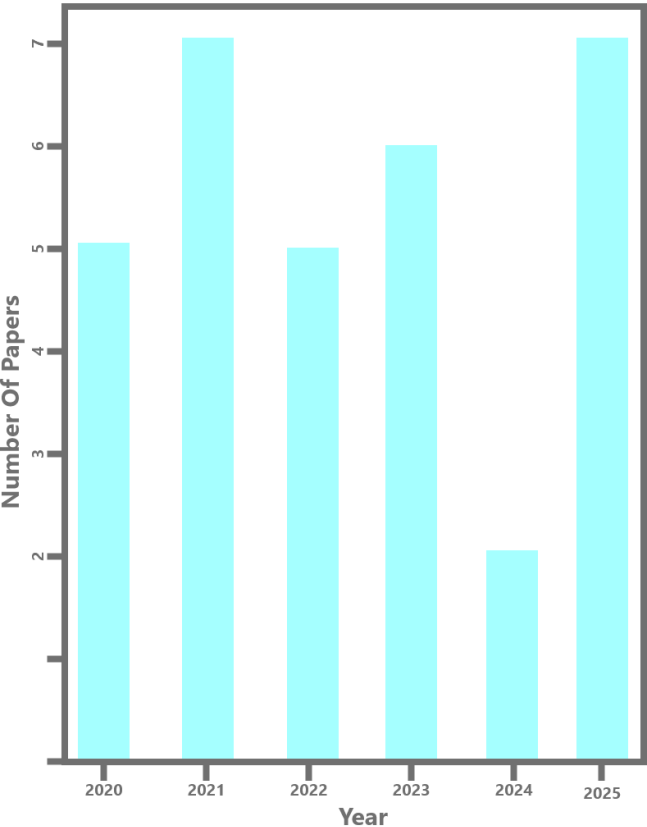
Challenges in low-light and dense traffic: Detection accuracy significantly drops in low-light environments [Paper 1, 4, 15, 17, 18, 23, 30] or when multiple vehicles overlap in crowded urban scenes [Paper 12, 13, 16, 20, 23, 27].

Limited integrated systems: Most studies focus on single-violation detection (such as ALPR-only [Paper 2, 18, 28, 29]) instead of unified, multi-violation approaches [Paper 1, 9, 11, 12, 15, 20, 21, 23].

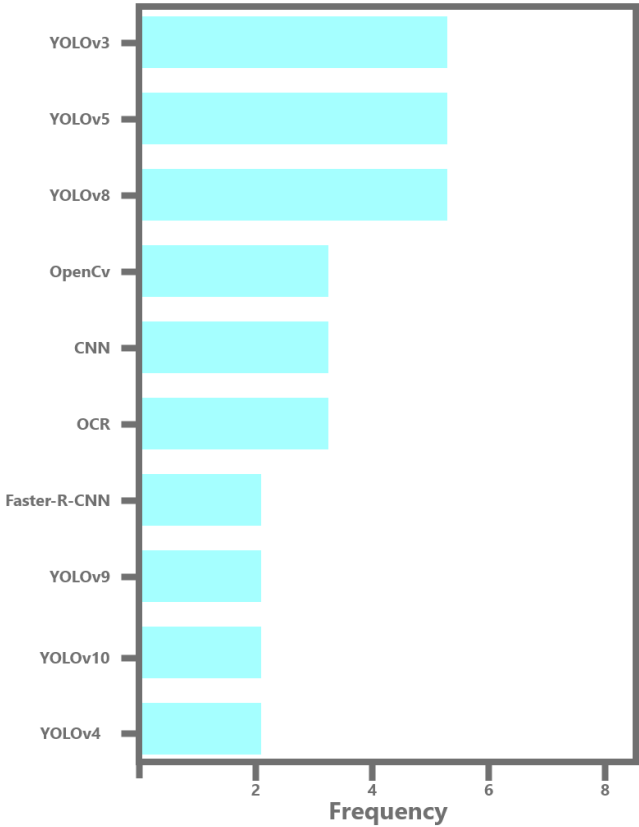
Dataset accessibility issues: Lack of open, localized Indian traffic datasets [Paper 5] remains a bottleneck. This is evidenced by the large number of studies relying on custom-built, non-public datasets [Paper 1, 12, 16, 18, 21, 23, 31] or identifying a lack of data as a general limitation [Paper 11].

Literature Review (Visualization)

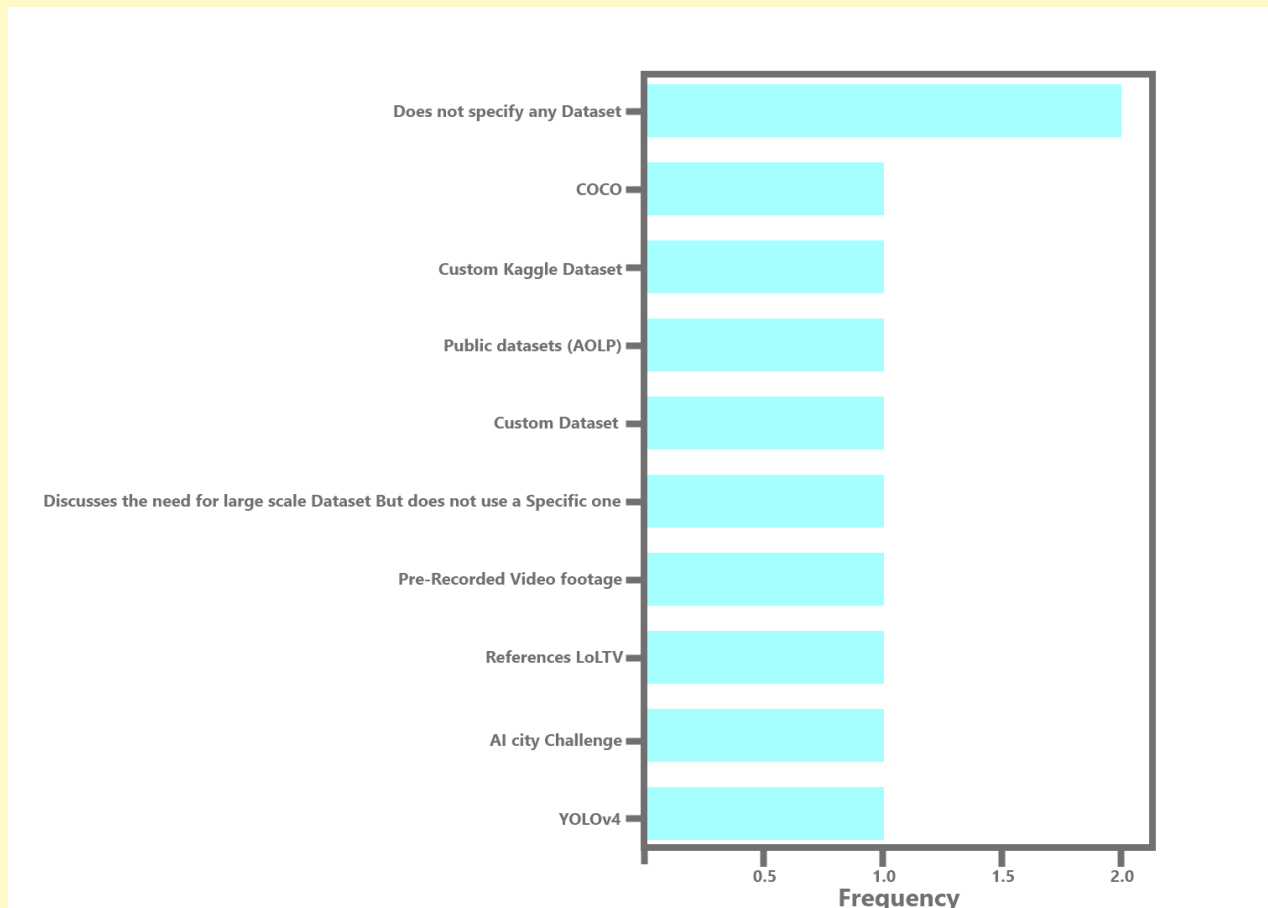
Papers and Published Year:



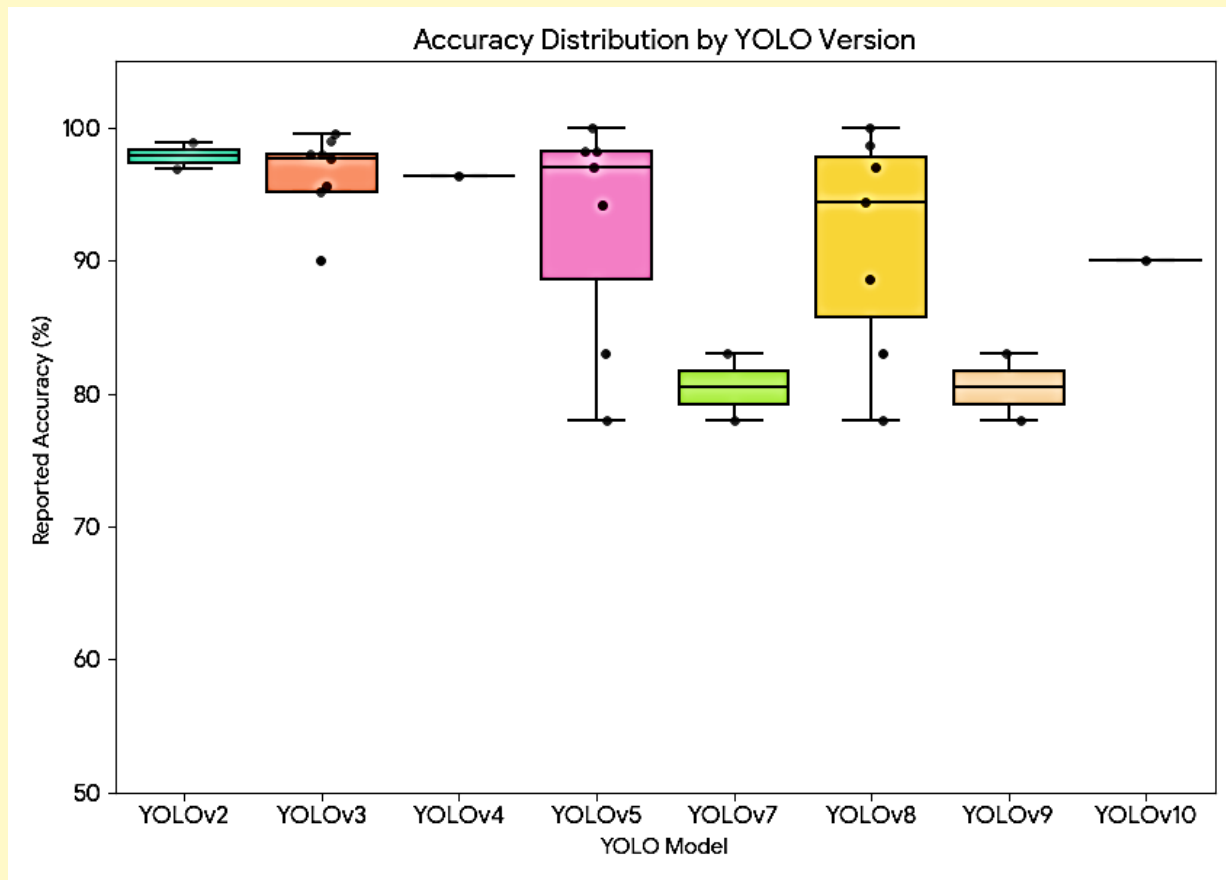
Top algorithms:



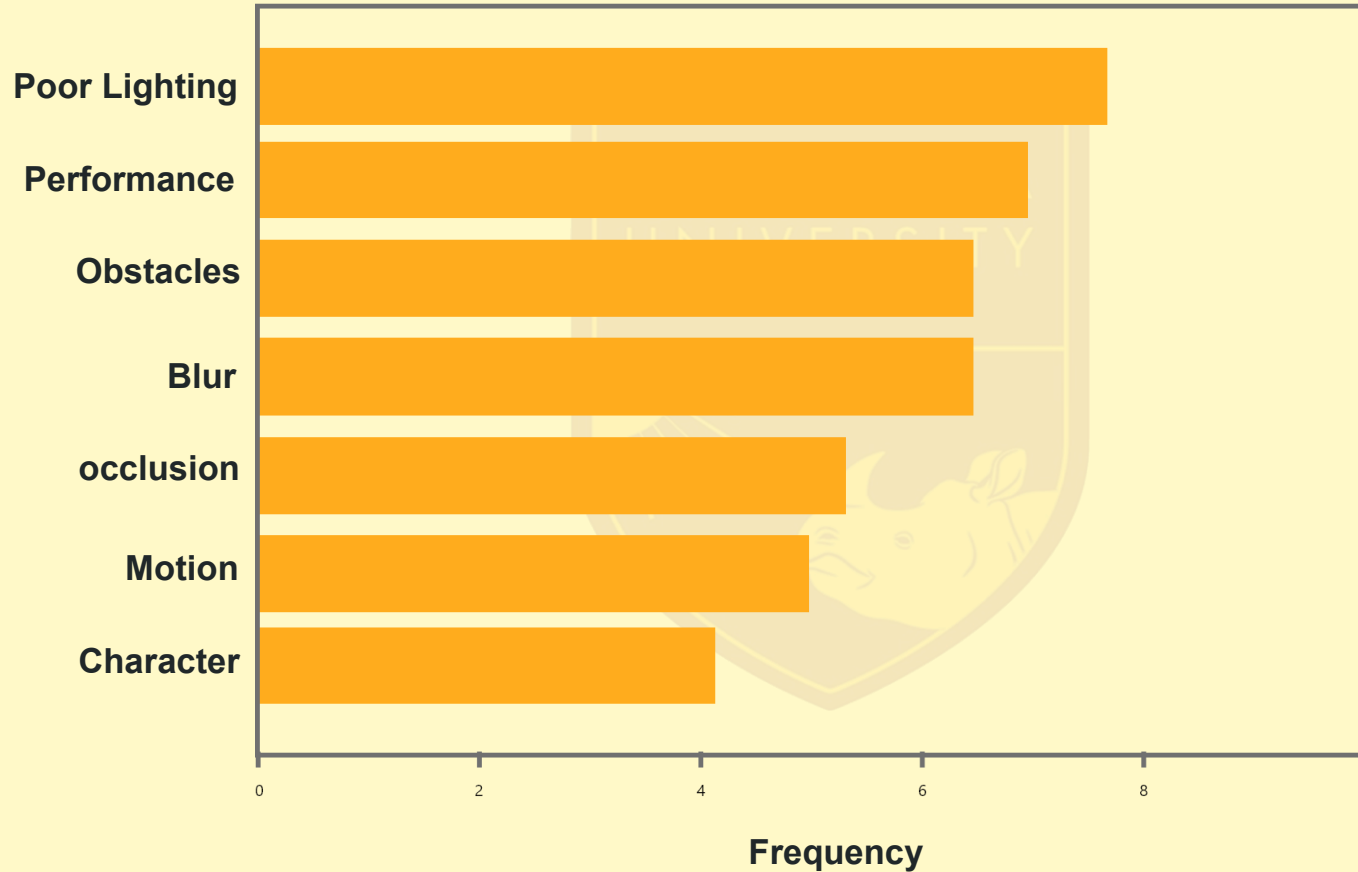
Literature Review (Visualization)



Literature Review (Visualization)



Common terms in limitations:



Identified Research Gaps

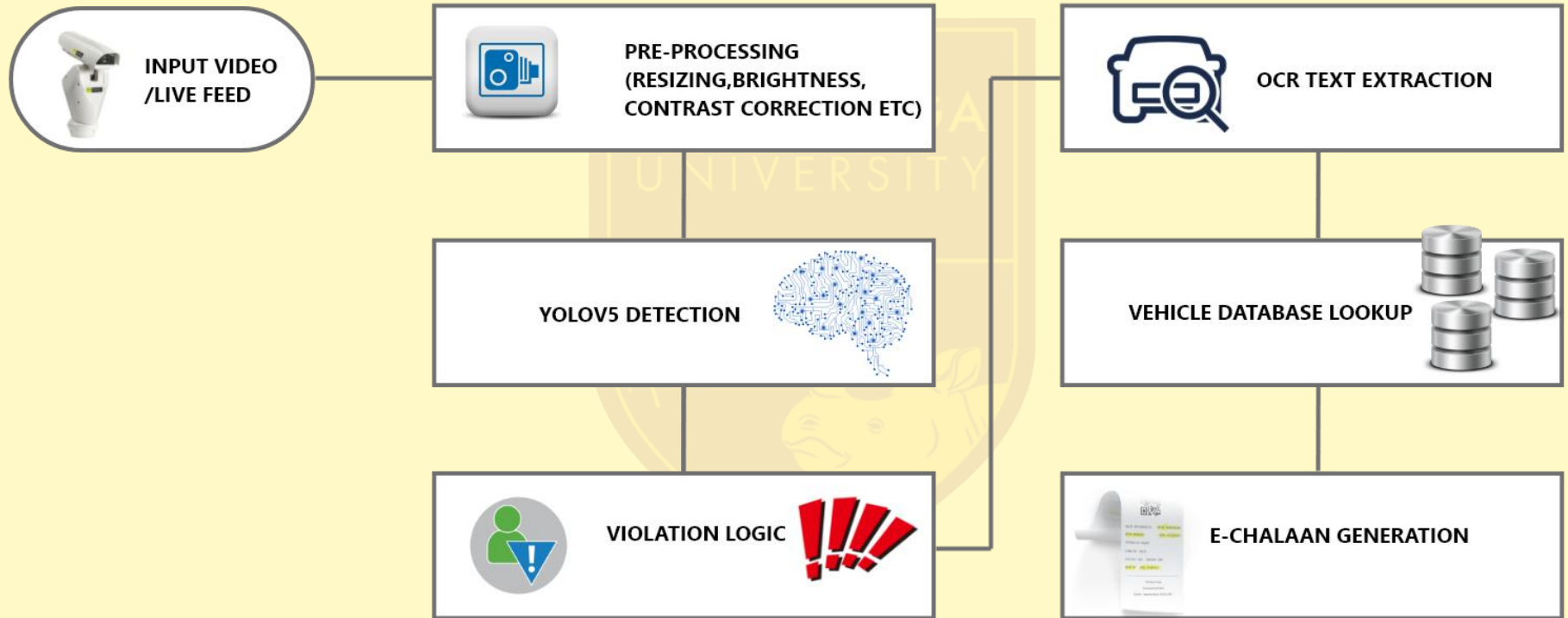
● Challenges in Current AI-based Traffic Enforcement Systems

- **Low-light detection limitations:** Existing AI models show degraded performance under poor lighting, motion blur, and night-time conditions.
- **Single violation focus:** Most systems target only one violation such as helmet detection or license plate recognition instead of multi-violation detection.
- **High infrastructure costs:** Complex hardware and cloud-based computation make deployment challenging for small cities or educational projects.
- **Lack of localized datasets:** Few publicly available Indian traffic datasets exist for model training, affecting localization and accuracy.
- **Limited end-to-end automation:** Few systems connect violation detection directly to automated e-challan generation and reporting modules.

Proposed System Overview

- Workflow of AI-based Traffic Violation Detection System
 - **Input Video Acquisition:** Collect CCTV or recorded traffic footage for system testing and model validation.
 - **Pre-processing:** Enhance video frames through resizing, brightness correction, and low-light adjustment using OpenCV.
 - **YOLOv5 Detection:** Detect vehicles, riders, helmets, and other relevant objects for traffic analysis.
 - **Violation Logic and OCR:** Apply rule-based logic to identify violations, extract plate numbers using OCR, and link with vehicle data.
 - **Automated e-Challan Generation:** Integrate outputs to generate challans automatically, simulating real-world enforcement systems.

Proposed System Block Diagram



Expected Outcomes

- Deliverables and Evaluation Metrics

- **Functional prototype:** A working system capable of detecting 3–4 common traffic violations such as no helmet, triple riding, wrong-lane, and red-light jumping.
- **OCR-based number plate recognition:** Accurately reads vehicle registration numbers from detected frames using trained OCR models.
- **Automated e-Challan generation:** Produces a PDF or on-screen challan once a violation is detected and verified against the simulated database.
- **Performance evaluation:** Assesses results using metrics like Precision, Recall, mAP, and FPS for model effectiveness and efficiency.
- **Demonstration and visualization:** Displays detection results in GUI-based visualization for clear presentation of system output.

Tools & Technologies

- Frameworks, Platforms, and Libraries Used

- **Programming Language:** Python — primary language for implementing AI models and system logic.
- **Frameworks and Libraries:** PyTorch for YOLOv5 training, OpenCV for image processing, NumPy and Pandas for data handling.
- **Platforms and Environment:** Google Colab used for model training with GPU support and seamless dataset access.
- **OCR and Detection Models:** EasyOCR and Tesseract integrated for number plate recognition and text extraction.
- **Datasets and Hardware:** Custom datasets plus LoLTV dataset; development on laptops and Jetson Nano for future deployment.

Project Plan & Timeline

- Phased Execution Strategy

Phase 1 – Current Phase

Focus on problem study, literature review, and dataset collection. Begin initial YOLOv5 training on sample datasets.

Deliverables

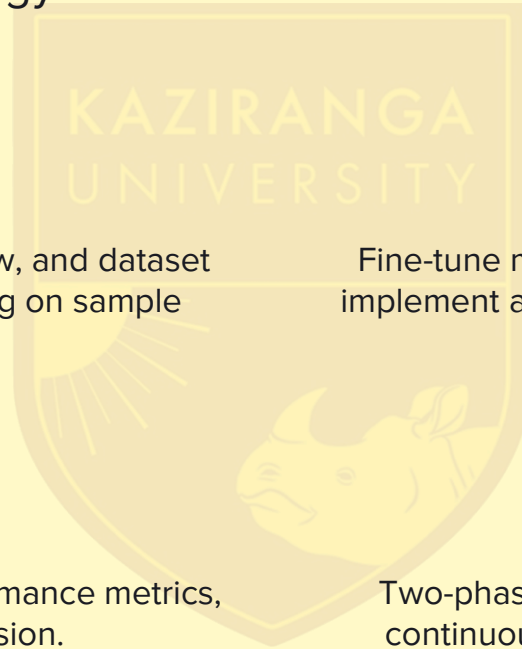
Fully integrated system prototype, performance metrics, and documentation for submission.

Phase 2 – Final Phase

Fine-tune model, integrate OCR and violation logic, implement automated e-challan module, and conduct testing.

Duration

Two-phase plan executed over 2–3 months with continuous evaluation and improvement cycles.



Initial Training Metrics (YOLOv5)

TRAINING CONFIGURATION

- **Dataset used:** Dhaka-AI-Yolo-Formatted-Dataset (Kaggle)
- **Dataset size :** 330 MB (2390 training images & 600 validation images)
- **Train-val ratio :** 80:20
- **Pre-processing Size:** Resized all training images to 640x640 pixels
- **Model Architecture:** YOLOv5
- **Total Epochs:** 100
- **Optimizer:** SGD(Stochastic Gradient Descent)
- **Platform:** Google Collab

PERFORMANCE METRICS

- **Precision :** 71.9%
- **mAP@0.5 :** 43.5 %
- **Recall :** 37.5 %
- **Loss Reduction:**
Box loss: Reduced from 0.089- 0.029
Objectness Loss: reduced from 0.061 – 0.036

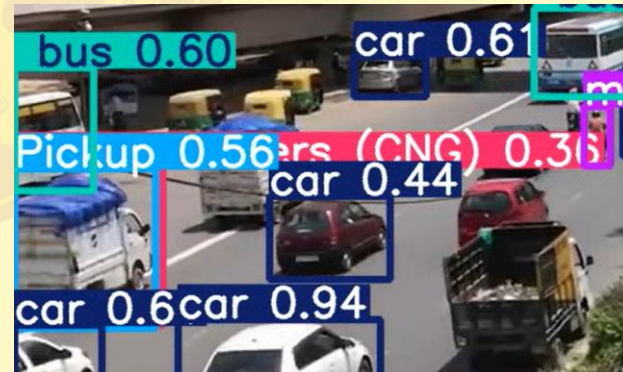
Training Image :



Labels(Object ID and its coordinates)

```
3 0.800000 0.743750 0.211667 0.512500
3 0.799583 0.424375 0.097500 0.138750
3 0.799583 0.330000 0.084167 0.057500
13 0.663750 0.595625 0.059167 0.158750
13 0.668750 0.483125 0.042500 0.056250
13 0.123750 0.483750 0.047500 0.090000
13 0.176250 0.474375 0.055833 0.098750
13 0.471667 0.310000 0.025000 0.047500
13 0.716250 0.281875 0.017500 0.036250
13 0.719167 0.249375 0.015000 0.021250
13 0.346667 0.335000 0.035000 0.045000
13 0.532917 0.280625 0.022500 0.038750
9 0.727083 0.318125 0.015833 0.033750
4 0.211250 0.335625 0.059167 0.041250
4 0.427500 0.287500 0.030000 0.035000
```

Test Image :



Initial Training Metrics (YOLOv5)

Learning Rate

We used a **learning rate schedule**. It started at **0.0097** and gradually decayed to **0.000298** by the final epoch. This decay helps the model find a more precise solution as training progresses.

Batch Size

A batch size of **16** was used. This means the model looked at 16 images at a time before updating its weights. This size is a good balance between learning speed and memory usage on Google Colab.

WHY USE SGD (STOCHASTIC GRADIENT DESCENT)?

While Adam is another popular optimizer, we used SGD for two main reasons:

1. **YOLOv5 Architecture:** The entire YOLOv5 architecture, including its learning rate schedules and default hyperparameters, is built and optimized around using SGD. Switching to Adam would require re-tuning all these other settings from scratch to get comparable results.
1. **Better Generalization:** Adam often converges *faster* in the initial epochs, but SGD often leads to a *better* and more reliable final model. For a research project, "better generalization" is almost always the goal.

Initial Training Metrics (YOLOv5)

Formulas used :

Precision

Measures how many of the *detections* were correct. (High precision = low false alarms).

$$\text{Precision} = \frac{TP}{TP + FP}$$

TP (True Positive): Model predicted "car" and it was a car.

FP (False Positive): Model predicted "car" but it was a bush.

Recall

Measures how many of the *actual objects* were found. (High recall = low missed objects).

$$\text{Recall} = \frac{TP}{TP + FN}$$

FN (False Negative): There was a car, but the model missed it.

Loss Reduction (Calculation)

Shows how much the model learned. We calculate the percentage drop from the start to the end of training.

$$\% \text{Reduction} = \frac{(\text{Initial loss} - \text{Final loss})}{\text{Initial loss}} \times 100$$

Example (Box Loss): $(0.089 - 0.029)/0.089 = 67.4\%$
Reduction

mAP@0.5 (Mean Average Precision)

The average precision across all classes, where a detection is "correct" only if it overlaps the true object by > 50% (IoU).

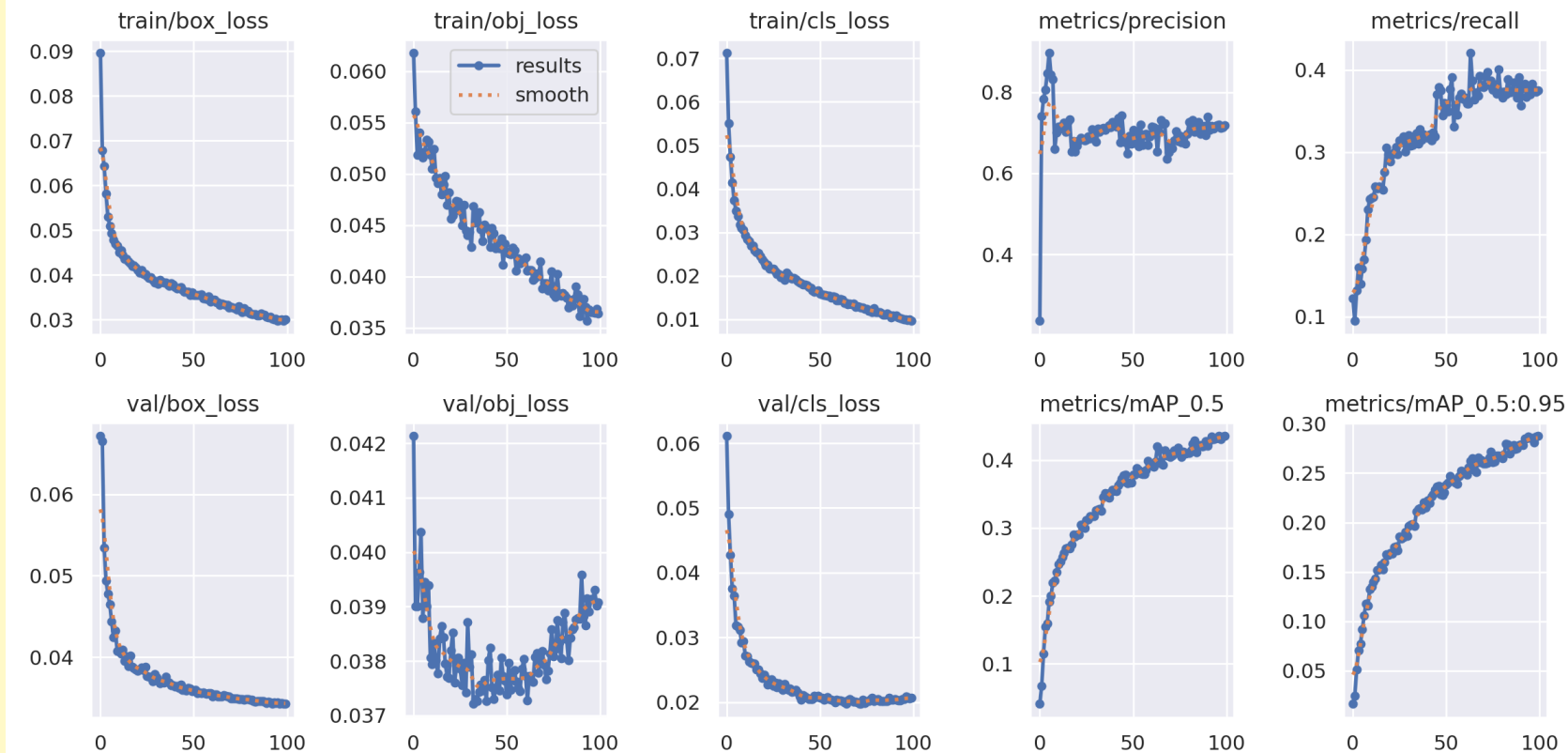
Why the scores are Low:

The low recall is likely due to the small and poor quality of dataset ('Dhaka-AI')

We will address this in Phase 2 by fine-tuning the model on a custom data adding more varied training data and applying data augmentation.

Initial Training Metrics (YOLOv5)

Results.png :



Real-time detection of vehicles and riders in test footage (Source- Youtube, Location- NewDelhi)



Why YOLOv5 ?

YOLOv5 is a more mature, stable, and well-documented framework, making it ideal for a learning prototype where integration is the main goal.

YOLOv5 has lower computational overhead, allowing for faster training and iteration on the hardware available to us (Google Colab)

References :

- [**PAPER 1**] A Deep Learning Approach for Helmet Detection and Fine Generation System, Sulbha Yadav et al., 2025
- [**PAPER 2**] An Efficient and Layout-Independent Automatic License Plate Recognition System Based on the YOLO detector, Rayson Laroca et al., 2021
- [**PAPER 3**] Automated Traffic Violation Detection Using AI and Machine Learning, Angelina Grace, 2025
- [**PAPER 4**] Automatic Traffic Rules Violation Detection and Number Plate Recognition System for Bangladesh, Raian Shahrear et al., 2020
- [**PAPER 5**] AI-Based Integrated Traffic Violation Detection and Smart Traffic Management System: A Comprehensive Review, Avadhut Dilip sutar, 2025
- [**PAPER 8**] LoLTV: A Low Light Two-Wheeler Violation Dataset With Anomaly Detection Technique, Samprit Bose et al., 2023
- [**PAPER 9**] NEXT-GEN TRAFFIC SURVEILLANCE: AI-ASSISTED MOBILE TRAFFIC VIOLATION DETECTION SYSTEM, Dila Dede et al., 2023

References :

- [**PAPER 11**] TRAFFIC SIGNAL VIOLATION DETECTION USING ARTIFICIAL INTELLIGENCE AND DEEP LEARNING, Dr. S. Raj Anand et al., 2021
- [**PAPER 12**] Two-Wheeler Traffic Violations Detection and Automated Penalty Issuance System, Batti Tulasidasu et al., 2025
- [**PAPER 13**] UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking, Longyin Wen et al., 2020
- [**PAPER 15**] An Intelligent System for Traffic Violation Detection: A Deep Learning Approach, M. Jayanthi et al., 2021
- [**PAPER 16**] An Automated System to Detect Helmet on Motorcyclists and Pillion Rider Using YOLOv5 and Deep SORT, Arsalan Hasnain Khan et al., 2023
- [**PAPER 17**] Real Time Traffic Signal Violation Detection System using Machine Learning, Anshul Sharma et al., 2021
- [**PAPER 18**] Real-Time Automatic Number Plate Recognition based on YOLOv5, Ayush Roy et al., 2022

References :

- [**PAPER 20**] Real-time Traffic Violation Detection System based on YOLO and Centroid Tracking, Yifan Wang et al., 2020
- [**PAPER 21**] A Novel Approach for Traffic Violation Detection using Deep Learning, Ashish Kumar et al., 2021
- [**PAPER 22**] Automatic Traffic Violation Detection: A Smart and an Intelligent Solution, Vishnu B et al., 2021
- [**PAPER 23**] An Automated System for Monitoring of Traffic Rule Violations using Deep Learning, H.N. Harish et al., 2022
- [**PAPER 26**] License Plate Detection Based on Improved YOLOv8n Network, Ruizhe Zhu et al., 2025
- [**PAPER 27**] Fast Helmet and License Plate Detection Based on Lightweight YOLOv5, Chenyang Wei et al., 2023
- [**PAPER 28**] End-to-End High Accuracy License Plate Recognition Based on Depthwise Separable Convolution Networks, Song-Ren Wang et al., 2022

References :

- [**PAPER 29**] Deep Learning Based Framework for Iranian License Plate Detection and Recognition, Mojtaba Shahidi Zandi & Roozbeh Rajabi, 2022
- [**PAPER 30**] Cloud-Based License Plate Recognition: A Comparative Approach Using You Only Look Once Versions 5, 7, 8, and 9 Object Detection, Christine Bukola Asaju et al., 2025
- [**PAPER 31**] Automated Indonesian Plate Recognition: YOLOv8 Detection and TensorFlow-CNN Character Classification, Windu Gata et al., 2025



Conclusion & Future Scope

- Project Summary and Forward Vision

- **Project Summary:** Developed a learning-oriented AI prototype replicating core components of real-world traffic enforcement systems using YOLOv5 and OCR.
- **Key Achievements:** Completed literature review, initial model training, system design, and partial testing to understand detection-to-automation workflows.
- **Future Improvements:** Enhance low-light detection, expand datasets, and integrate edge deployment using Jetson Nano or Raspberry Pi for real-time processing.
- **Extended Applications:** Potential to develop web dashboards for violation monitoring, data analytics, and automated record management.
- **Learning Outcome:** Provided hands-on experience with AI model training, system integration, and performance analysis in an applied research context.



**THANK
YOU**