# Helmet Violation Detection and Number Plate Recognition (Engineering Track)


## Abstract

*Road safety is a major concern, especially in accident-prone regions where helmet-less riding leads to fatalities. This project develops a system for detecting helmet violations and respective number plates and extracting vehicle license numbers. We employ detection models for violation and number plate detection and an OCR model for alphanumeric extraction. The concerned authorities can further use these license numbers to penalize the violators and enforce safety guidelines on roads.*


## 1. Problem Statement

This system is designed only for outdoor environments such as roads and highways, with input coming from a CCTV camera, and the CCTV footage must capture the two-wheeler vehicle, the rider and the number plate. The output is the extracted vehicle number plate if the rider does not wear a helmet.

For this project, since we do not have access to live CCTV footage, we will provide a demonstration of this pipeline system on a stock video or some simulated video.

## 2. User Interface

### 2.1. User Interface Design

The user interface for our system is designed to be simple, interactive, and efficient. It enables users to upload traffic surveillance videos, process frames to detect helmet violations and number plates, and view the results in a structured format. Figure 1 illustrates the workflow of our system, depicting the different stages of processing from video upload to report generation.

For now, we only have a basic user interface ready where one can upload videos.

### 2.2. AI-Based Coding Tools Used

We have used ChatGPT and GitHub Copilot to generate code snippets and explanations and for debugging purposes.

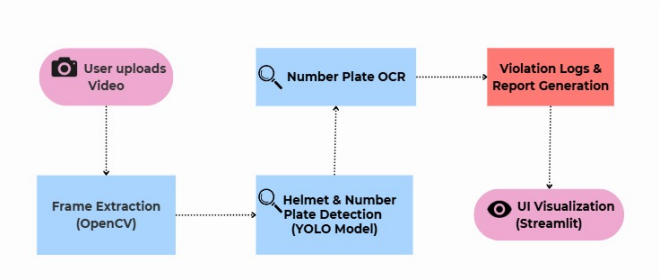


Figure 1. System Workflow for Helmet and Number Plate Detection

We are also using Streamlit for UI development and visualisation of results as it provides an easy-to-use interface for users to interact with the system.

## 3. Related Work

Meenu R et al. (2020) [1] proposed a system where they performed helmet detection and number plate extraction using Faster Region-Convolutional Neural Network (Faster R-CNN). They used CCTV footage and then split it into frames for analysis. Their methodology was split into four stages: motorcycle detection, head detection, helmet detection, and then number plate detection.

Pushkar Sathe et al. (2022) [2] used YOLOv5 for helmet detection. They are using two methods to check if the rider is wearing a helmet. Firstly, they check with the help of overlapping boxes of the helmet, number plate, and the person and verify through a set of conditions if the person is wearing a helmet or not. The second method uses a range of motorcycle coordinates to check for helmets. Finally, they use EasyOCR to recognize number plates.

J Mistry et al. (2017) [3] used YOLOv2 for first detecting persons in a frame, citing the better performances in detecting a person rather than a motorcycle of the model. It then proceeds to detect the helmet, and if it is not found, then it goes for the number plate. For no number plate detected, the model infers that the person detected is a pedestrian.

V Sri Uthra et al. (2020) [4] presented significant findings where the paper proposed a motorcycle detection and classification method, helmet detection, and license plate recognition. Vehicle Classification was performed using an SVM classifier. Helmet detection was done by applying convolutional Neural Network (CNN) algorithms to extract image attributes, followed by classification using the SVM classifier. License plate recognition was done using Optical Character Recognition (OCR).

Mamidi Kiran Kumar et al. (2023) [5] use the YOLO Darknet deep learning framework to automate the detection of motorcycle riders wearing helmets from images, simultaneously triggering alerts for non-compliance. Through bounding boxes and confidence scores, the model identifies regions of interest like riders, helmets, and number plates. The strengths of the model lie in its automated and efficient solution for helmet detection, eliminating the need for manual checks, and its utilization of the YOLO Darknet framework, enabling real-time detection and alert generation.

# 4. Datasets and Evaluation Metrics

## 4.1. Helmet Violation Detection Dataset

The dataset [6] used for violation detection is an Open Source Dataset from Roboflow. The dataset has annotations for two classes: with_helmet and without_helmet. These annotations are bounding boxes containing the vehicle as well as the rider.

The training set consists of 2,862 images, with augmentations applied including noise to 0.5% of pixels and grayscale to 10% of images, creating two versions of each source image (1,431). There are 477 images in the validation set.

The training set has 2,310 annotations (0.4) for with_helmet and 3,274 annotations (0.6) for without_helmet and the validation set has 423 annotations (0.45) for with_helmet and 512 annotations (0.55) for without_helmet. There is a slight imbalance among classes but this is tolerable and won't have any significant impact on the results.

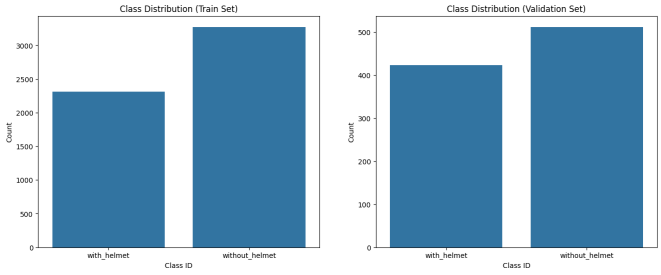


Figure 2. Class distribution across Violation Detection Dataset

The related evaluation metrics are precision, recall and mAP50-95. Precision [TP/(TP+FP)] answers "of all predicted bounding boxes, how many are actually correct?". Recall [TP/(TP+FN)] answers "of all actual objects/bounding boxes, how many were detected?". mAP50-95 evaluates the model's performance by considering both precision and recall across IoU thresholds [0.50, 0.55, ..., 0.95].

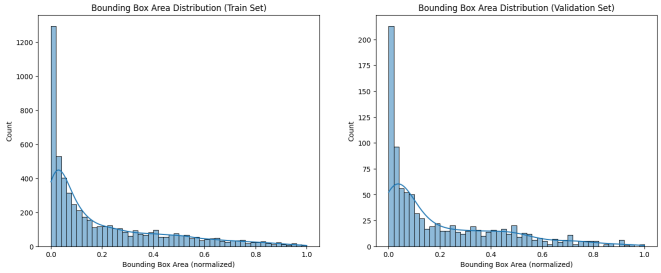


Figure 3. Bounding Box Size distribution across Violation Detection Dataset

## 4.2. Additional Dataset Analysis and Segregation

Before training our detection models, we performed a detailed exploratory analysis on the raw images to quantify lighting, contrast, and blur characteristics, then automatically reorganized the dataset into folders for each condition combination.

**Feature extraction.** For every image we computed:

- **Brightness:** mean of the HSV Value channel.
- **Contrast:** standard deviation of the grayscale intensities.
- **Color variance:** per-channel (B, G, R) 256-bin histograms.
- **Blur measure:** variance of the Laplacian on the grayscale image (low variance = blurry, high variance = sharp).

**Adaptive binning.** We applied $k$-means ($k = 3$) separately to the brightness and contrast distributions to define "low," "medium," and "high" ranges. The blur threshold was selected from the valley between the two peaks of the Laplacian-variance histogram.

**Composite categories.** Each image was labeled with brightness, contrast, and blur categories (e.g. bright, high_contrast, blurry) and then grouped into a folder names.

All scripts and the updated, segregated dataset are available in our GitHub repository.

## 4.3. License Plate Detection Dataset

The dataset [7] used for license plate detection is an Open Source Dataset from Roboflow. It consists of a large collection of images with varying backgrounds, lighting conditions and angles, and annotated bounding boxes indicating

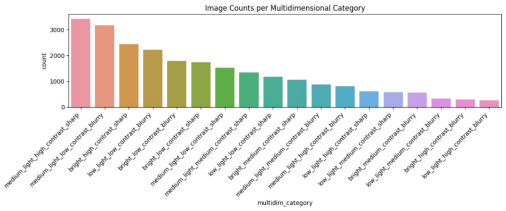


Figure 4. Number of images in each composite category (e.g. bright_high_contrast_blurry, medium_light_low_contrast_sharp, etc.).

the location of the license plates. Samples are divided into three sets: 21,173 images in the training set, 2,046 images in the validation set, and 1,019 images in the test set.

Preprocessing included automatically correcting the image orientation and resizing all images to $640 \times 640$ pixels to ensure uniform input size. To improve model robustness, each training image was augmented with up to 3 transformations, including horizontal flips, zoom (up to 15%), random rotation ($\pm 10°$), shear ($\pm 2°$) and grayscale (10%).

Bounding box area distribution analysis, shown in Figure 5, reveals that most license plates occupy a small fraction of the image, emphasizing the need for precise models to detect small objects.

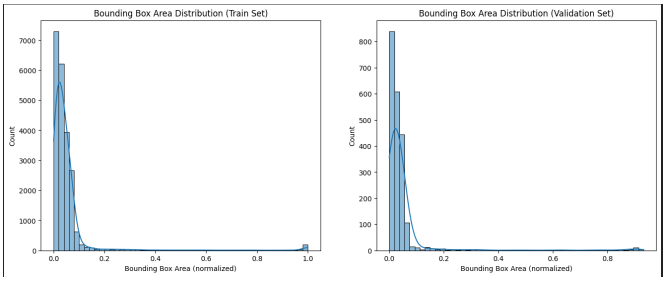


Figure 5. Bounding box area distribution across the License Plate Detection dataset

The related evaluation metrics are the same as the Violation Detection Dataset, i.e. precision, recall and mAP50-95.

### 4.4. Resource Requirements for the system

The system requires around 8GB RAM and around 1GB GPU memory (optional, but absence affects inferencing time) to use both YOLO models for inferencing and processing each frame. Storage requirements is around 1GB.

## 5. Analysis of Results

### 5.1. Helmet Violation Detection

We trained five YOLO models for the detection of with_helmet and without_helmet classes on the dataset for 100 epochs and a patience counter of 5. The overall results (for both classes) on validation set are as follows

| Model | Precision | Recall | mAP50-95 |
|---|---|---|---|
| YOLOv10m | 0.753 | 0.65 | 0.464 |
| YOLOv10b | 0.736 | 0.657 | 0.473 |
| YOLOv10l | 0.724 | 0.628 | 0.45 |
| YOLOv11m | 0.739 | 0.655 | 0.457 |
| YOLOv11l | 0.754 | 0.699 | 0.481 |

YOLOv11l model has the best scores across all metrics.

### 5.2. Additional Results in Helmet Violation Detection

| Model | Precision | Recall | mAP50-95 |
|---|---|---|---|
| YOLOv8m | 0.752 | 0.701 | **0.489** |
| YOLOv8l | **0.769** | 0.694 | 0.485 |
| YOLOv9m | 0.702 | **0.704** | 0.479 |
| YOLOv9c | 0.722 | 0.68 | 0.475 |

YOLOv8m model has the best mAP50-95 among all models. Figure 6 shows the training statistics and test results.

Training Statistics on Train Set and Results on Validation Set:

| YOLO Model | GPU Memory | Early Stopping at Epoch | with helmet (P, R, mAP50-95) | without helmet (P, R, mAP50-95) | all classes (P, R, mAP50-95) |
|---|---|---|---|---|---|
| v8m | 8.25 GB | 26 | 0.71, 0.655, 0.442 | 0.794, 0.748, 0.535 | 0.752, 0.701, 0.489 |
| v8l | 11.7 GB | 31 | 0.732, 0.674, 0.441 | 0.805, 0.715, 0.53 | 0.769, 0.694, 0.485 |
| v9m | 9.56 GB | 26 | 0.743, 0.61, 0.431 | 0.661, 0.798, 0.526 | 0.702, 0.704, 0.479 |
| v9c | 12.3 GB | 28 | 0.621, 0.683, 0.432 | 0.823, 0.678, 0.519 | 0.722, 0.68, 0.475 |
| v10m | 10.3 GB | 31 | 0.751, 0.585, 0.416 | 0.755, 0.715, 0.511 | 0.753, 0.65, 0.464 |
| v10b | 13.2 GB | 34 | 0.673, 0.642, 0.428 | 0.799, 0.672, 0.518 | 0.736, 0.657, 0.473 |
| v10l | 15.1 GB | 21 | 0.684, 0.596, 0.398 | 0.763, 0.661, 0.503 | 0.724, 0.628, 0.45 |
| v11m | 10 GB | 24 | 0.671, 0.655, 0.423 | 0.806, 0.655, 0.492 | 0.739, 0.655, 0.457 |
| v11l | 12.2 GB | 39 | 0.7, 0.674, 0.433 | 0.807, 0.724, 0.529 | 0.754, 0.699, 0.481 |

Figure 6. Training Statistics and Results for Violation Detection Model

### 5.3. License Plate Detection

We trained YOLOv11n model for license plate detection on the dataset for 50 epochs with a patience counter of 5. It achieved a Precision of 0.988, Recall of 0.96 and mAP50-95 of 0.709 on the validation set.

### 5.4. Additional Results under Segmented Conditions

The dataset was segmented as described earlier into lighting, contrast, and blur categories. We evaluate the license plate detection model on four representative extreme conditions.

- **Rationale:** We use AP@[.50:.95] as a robust indicator of overall detection quality across multiple IoU thresholds. AP@.50 evaluates detection recall (looser match), and AP@.75 reflects stricter localization accuracy.

Table 1. Comparison of detection performance

| Condition | AP@[.50:.95] | AP@.50 | AP@.75 |
|---|---|---|---|
| bright_low_contrast_sharp | 0.6927 | 0.8502 | 0.7860 |
| low_light_low_contrast_blurry | 0.6005 | 0.9301 | 0.6586 |
| bright_low_contrast_blurry (best) | **0.7642** | **0.9587** | **0.9180** |
| bright_medium_contrast_blurry (worst) | **0.4243** | **0.8327** | **0.3891** |

- **Observations:**
  - The bright_low_contrast_blurry condition achieves the highest performance (AP@[.50:.95] = 0.7642), suggesting blur is not a limiting factor in well-lit environments.
  - The lowest score (0.4243) is seen under bright_medium_contrast_blurry, potentially due to confusing textures and insufficient lighting gradient.
  - bright_low_contrast_sharp performs better than low_light_low_contrast_blurry, indicating lighting plays a more significant role than sharpness.
  - Uneven distribution of images per condition may affect metric reliability and is a limitation in interpreting comparative results.

## 6. Compute Requirements

The compute requirements have been effectively managed for our project. The datasets were available as an open-source resource, which provided a robust collection of annotated images. This accessibility allowed us to quickly train and validate our models without encountering major resource constraints. Consequently, our computational infrastructure has sufficiently supported the training of various YOLO models and the execution of detection and OCR tasks, eliminating the need for any significant adjustments.

## 7. Conclusion and Future Work

This work presented a two-stage pipeline for detecting helmet violations and recognizing license plates using YOLOv8. We evaluated performance under diverse lighting and image quality conditions using COCO-style metrics. The model performed best in bright_low_contrast_blurry settings and worst in bright_medium_contrast_blurry, showing that visual conditions significantly impact detection accuracy. Results may be skewed due to imbalance in dataset splits across conditions.

For future works we will propose to:

- Fine-tune the detector on more balanced subsets to improve generalization across visual domains.
- Integrate OCR confidence filtering to improve license plate text accuracy.
- Extend the pipeline to support batch evaluation of videos or frame sequences.

## 8. Individual Tasks

| Name | Tasks |
|---|---|
| Himanshu Raj | Pipeline, Helmet Violation detection model, Finding Datasets and relevant Research papers, Report |
| Aarya Khandelwal | Analysis of Datasets and Results, License Plate detection model, Finding Datasets and relevant Research papers, Report, PPT |
| Mahi Mann | Analysis of Datasets and Results, User interface, Finding Datasets and relevant Research papers, Report, PPT |

## References


1. Meenu R, Sinta Raju, Smrithi P Paul, Swathy Sajeev, Alphonsa Johny. "Detection of Helmetless Riders Using Faster R-CNN" Volume. 5 Issue. 5, - 2020, International Journal of Innovative Science and Research Technology (IJISRT), www.ijisrt.com. ISSN - 2456-2165, PP :- 1616-1620.
2. P. Sathe, A. Rao, A. Singh, R. Nair and A. Poojary, "Helmet Detection And Number Plate Recognition Using Deep Learning," 2022 IEEE Region 10 Symposium (TENSYMP), Mumbai, India, 2022, pp. 1-6, DOI: 10.1109/TENSYMP54529.2022.9864462.
3. J. Mistry, A. K. Misraa, M. Agarwal, A. Vyas, V. M. Chudasama and K. P. Upla, "An automatic detection of helmeted and non-helmeted motorcyclist with license plate extraction using convolutional neural network," 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), Montreal, QC, Canada, 2017, pp. 1-6, DOI: 10.1109/IPTA.2017.8310092.
4. Sri U. V., Sariga D. V., Vaishali K. S., Padma, P. S. (2020). "Helmet violation detection using deep learning." International Research Journal of Engineering and Technology, Vol. 7, pp. 3091–3095.
5. Kiran-Kumar, M., Sanjana, C., Shireen, F.,Harichandana, D., Sharma, M., Manasa, M.(2023). "Automatic number plate detection for motorcyclists riding without helmets." E3S Web of Conferences, Vol. 430, pp. 01038. DOI: 10.1051/e3sconf/202343001038.
6. "Helmet - No Helmet Detection Dataset" (Programa Delfn, 2024) Open Source Dataset on Roboflow Universe. URL
7. "License Plate Recognition Dataset" (Roboflow Universe Projects, 2024) Open Source Dataset on Roboflow Universe. URL