Helmet Violation Detection and Number Plate Recognition

Engineering Track

Problem Statement, Scope & Users

Problem Statement:

- **Core Challenge:** Develop an automated system to process CCTV footage and identify two-wheeler riders violating helmet regulations by detecting the absence of a helmet.
- **Outcome**: If a rider is found without a helmet, extract the corresponding vehicle's license plate number for further penalization.

Scope

- Environment: Designed exclusively for outdoor scenarios (roads/highways).
- Input: CCTV videos capturing riders, vehicles, and license plates.
- Demonstration Approach: Use of stock or simulated videos, not using live CCTV footage.

Target Users:

- Traffic & Law Enforcement Authorities: To monitor, log, and penalize helmet violations.
- Public Safety Agencies: Aiding in proactive road safety enforcement and reducing accident-related fatalities.

Related Work & Baseline Methods

Related Work Overview:

- **Meenu R et al. (2020)**
  - Approach: Faster R-CNN based four-stage pipeline
  - Stages: Motorcycle detection, head detection, helmet detection, and license plate extraction.
- **Pushkar Sathe et al. (2022)**
  - Approach: YOLOv5 for helmet detection
  - Technique: Uses overlapping bounding boxes with spatial rules to verify helmet presence.
- **J Mistry et al. (2017)**
  - Approach: YOLOv2 driven pipeline
  - Technique: Begins with person detection, then helmet detection, followed by license plate extraction (infers pedestrian if no plate is detected).
- **V Sri Uthra et al. (2020)**
  - Approach: Combined detection using CNN for feature extraction and SVM for classification
  - Technique: Focuses on motorcycle detection, helmet detection, and license plate recognition, emphasizing vehicle classification.
- **Mamidi Kiran Kumar et al. (2023)**
  - Approach: YOLO Darknet framework
  - Technique: Automates helmet detection and number plate recognition, leveraging bounding boxes and confidence scores.

Baseline Methods in Our Project:

- Helmet Violation Detection:
  - Fine-tuned multiple YOLO models (including YOLOv10 and YOLOv11 variants) to classify riders as “with helmet” or “without helmet.”
- License Plate Detection:
  - Employed YOLOv11n for accurate localization of small objects like license plates in complex scenes.

Methodological Gaps & Shortcomings

Gaps & Shortcomings Identified Across Papers:

- Methodological Challenges:
  - Bounding Box Reliance: Heavy reliance on bounding box overlaps (as seen in YOLOv5 and YOLOv2 methods) can lead to inaccuracies, particularly in scenes with occlusion or low contrast.
  - Lighting & Quality: Variations in lighting and image quality often impair detection consistency, a challenge noted in multiple approaches.
- Data Limitations:
  - Inadequate annotations: Datasets, from that time, usually have annotations for a specific object like helmet or person, making training complex and hard.
  - Object Size: License plates are small and low resolution, making precise localization difficult.
- Implementation Issues:
  - End-to-End Pipeline: None of the surveyed methods provided a fully integrated ready to deploy solution (detection, OCR, and reporting) without gaps, they only proposed the system but implementation was not available publicly.

Methodological Insights:

- Detection Frameworks:
  - Object detection models (e.g., YOLO and Faster R-CNN) are effective for real-time applications, but each comes with trade-offs.
  - Multi-stage detection pipelines allow initial broad detection (e.g., identifying violations) and then refined localization (e.g., license plates).
- Integration of OCR:
  - Some works (like those using CNN+SVM or YOLO-based systems) attempt to integrate OCR for license extraction.

Dataset & Evaluation Metrics

Helmet Violation Detection Dataset:

- Source: Open Source Dataset from Roboflow
- Composition:
  - Training: 2,862 images (with augmentations such as noise and grayscale variations)
  - Validation: 477 images
- Annotations: Two classes – "with helmet" and "without helmet."

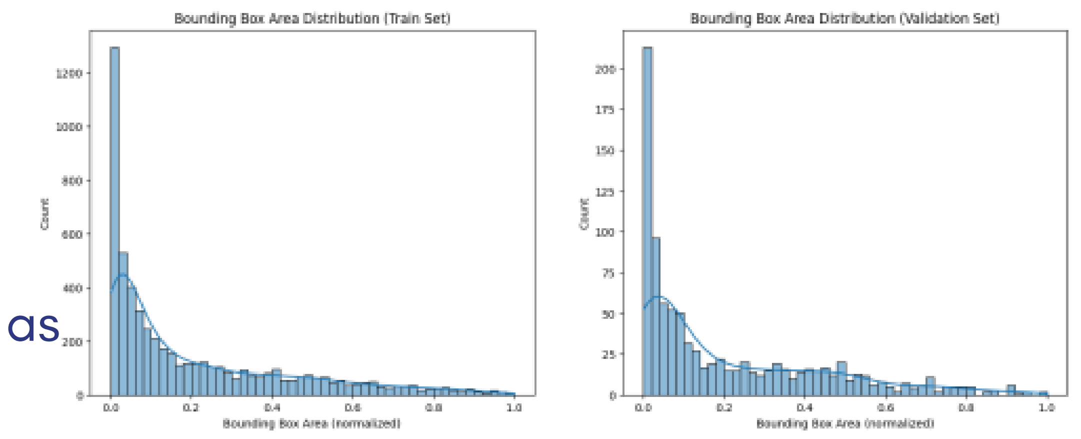


Figure 3. Bounding Box Size distribution across Violation Detection Dataset

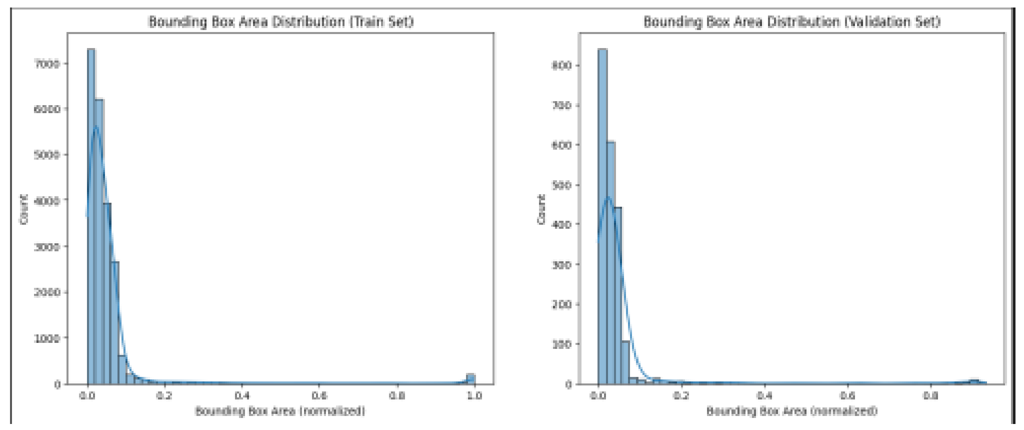


Figure 4. Bounding box area distribution across the License Plate Detection dataset

License Plate Detection Dataset:

- Source: Open Source Dataset from Roboflow
- Composition:
  - Training: 21,173 images
  - Validation: 2,046 images
  - Testing: 1,019 images
- Preprocessing: Automatic image orientation correction, resizing to 640×640, and augmentations (flips, zoom, rotations, shear).

Evaluation Metrics:

- Precision: Measures the accuracy of the detected bounding boxes.
- Recall: Indicates the proportion of actual objects detected.
- mAP50-95: A comprehensive metric evaluating precision and recall across multiple IoU thresholds.

System Overview & Pipeline

Overall System Pipeline:

1. Helmet Violation Detection:
   - Five YOLO models were fine-tuned. Best performing model: YOLOv11l (Precision: 0.754, Recall: 0.699, mAP50-95: 0.481).
2. License Plate Detection:
   - YOLOv11n model achieves a Precision of 0.988, Recall of 0.96 and mAP50-95 of 0.709 on the validation set.
3. OCR Integration (Pending):
   - Planned implementation using EasyOCR to extract alphanumeric characters from detected license plates.

User Interface (UI):

- Developed using Streamlit for video upload, real-time frame processing, and result visualization.
- Simplifies user interaction while abstracting the underlying complex processing pipeline.

Development Tools:

- Use of ChatGPT and GitHub Copilot for coding assistance, debugging, and code snippet generation.

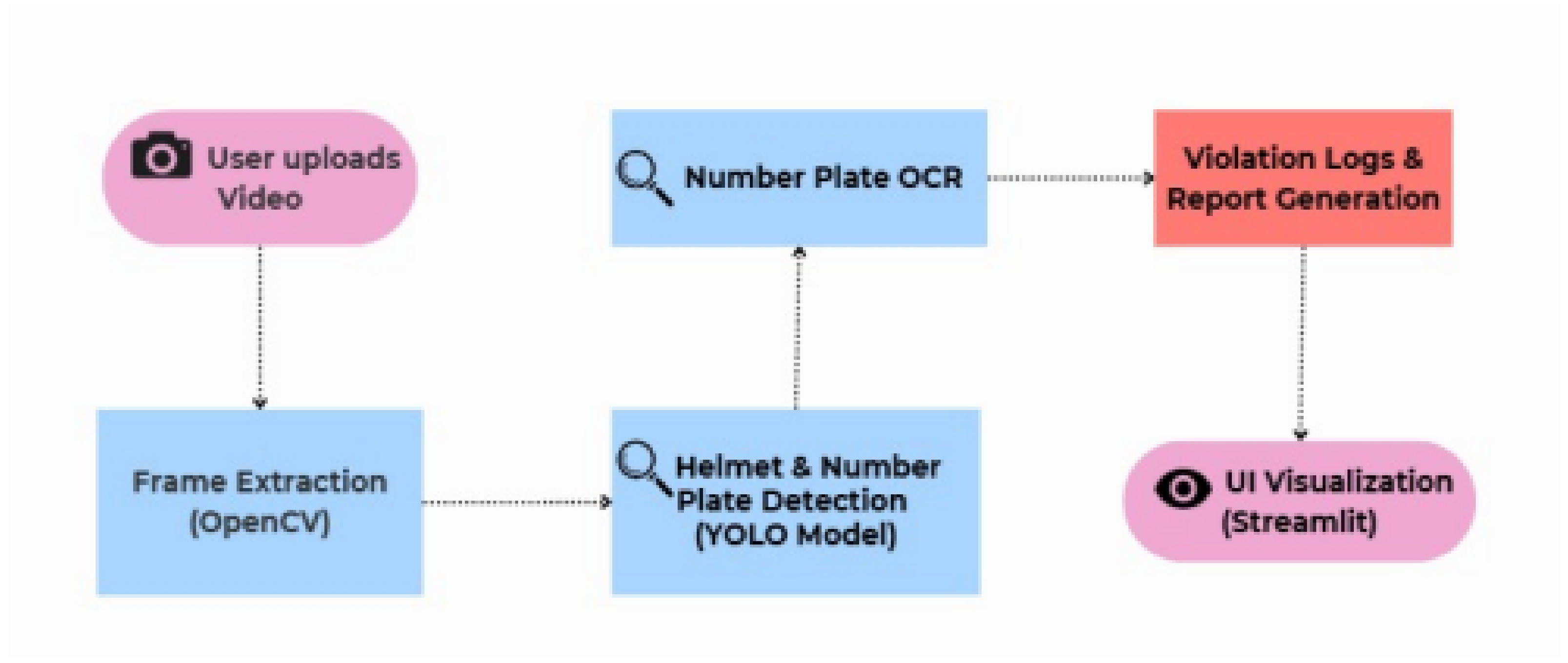
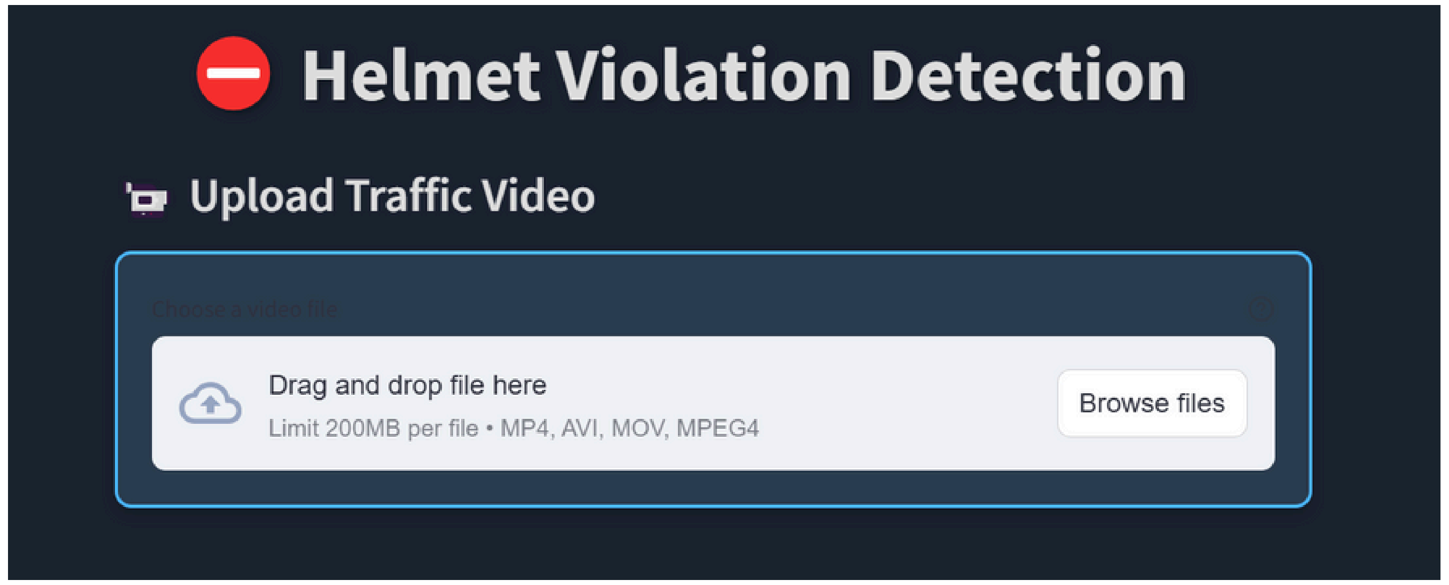

User uploads Video
Number Plate OCR
Violation Logs & Report Generation
Frame Extraction (OpenCV)
Helmet & Number Plate Detection (YOLO Model)
UI Visualization (Streamlit)

Figure 1. System Workflow for Helmet and Number Plate Detection


Helmet Violation Detection
Upload Traffic Video
Drag and drop file here
Limit 200MB per file • MP4, AVI, MOV, MPEG4
Browse files

Screenshot of video upload interface

Baseline Results & Analysis – Helmet Violation Detection

Model Performance on Validation Set:

- YOLOv10m: Precision 0.753, Recall 0.65, mAP50-95 0.464
- YOLOv10b: Precision 0.736, Recall 0.657, mAP50-95 0.473
- YOLOv10l: Precision 0.724, Recall 0.628, mAP50-95 0.45
- YOLOv11m: Precision 0.739, Recall 0.655, mAP50-95 0.457
- YOLOv11l: Best performing with Precision 0.754, Recall 0.699, mAP50-95 0.481.

Error Analysis & Observations:

- Inaccuracies: Occasional misdetections due to overlapping bounding boxes and occlusion in crowded scenes.
- Environmental Factors: Variability in lighting conditions and motion blur in video frames contribute to detection challenges.
- Class Imbalance: Minor imbalance between “with helmet” and “without helmet” classes, though not significantly impacting performance.

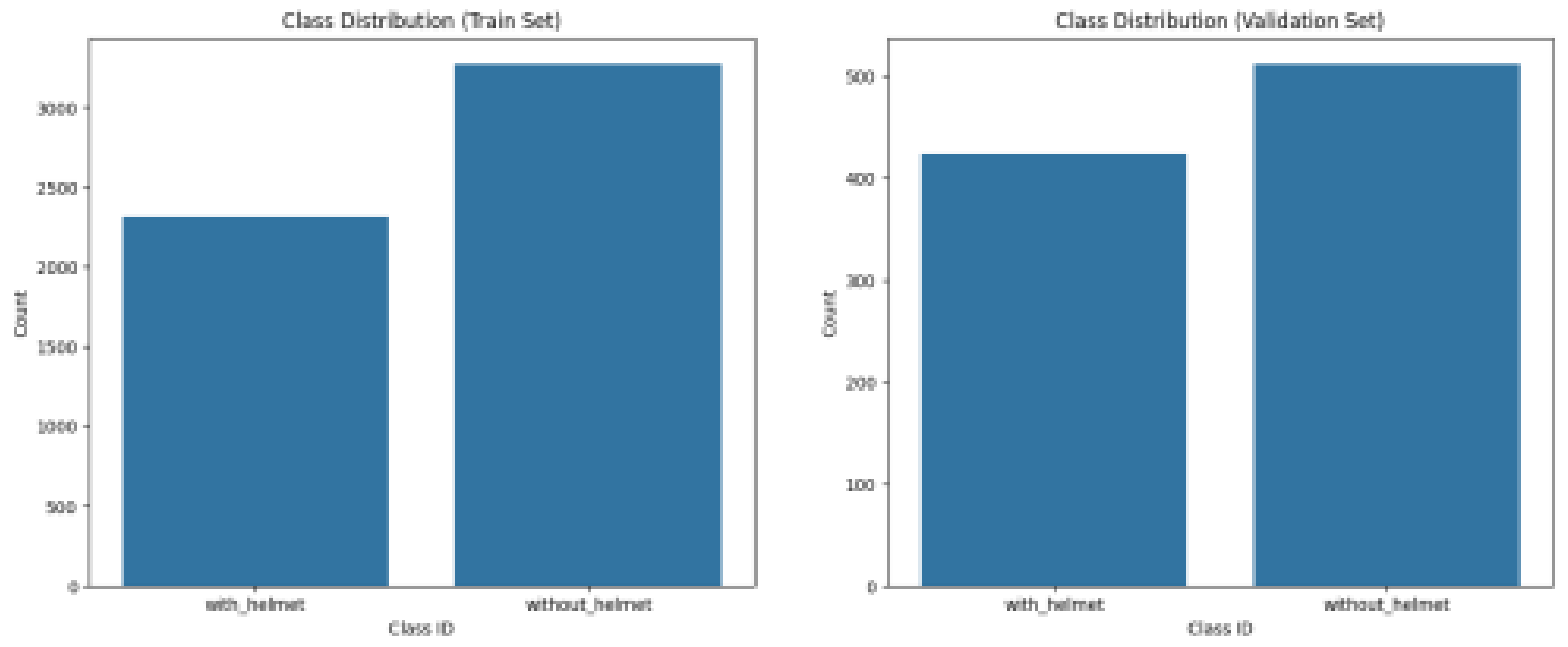
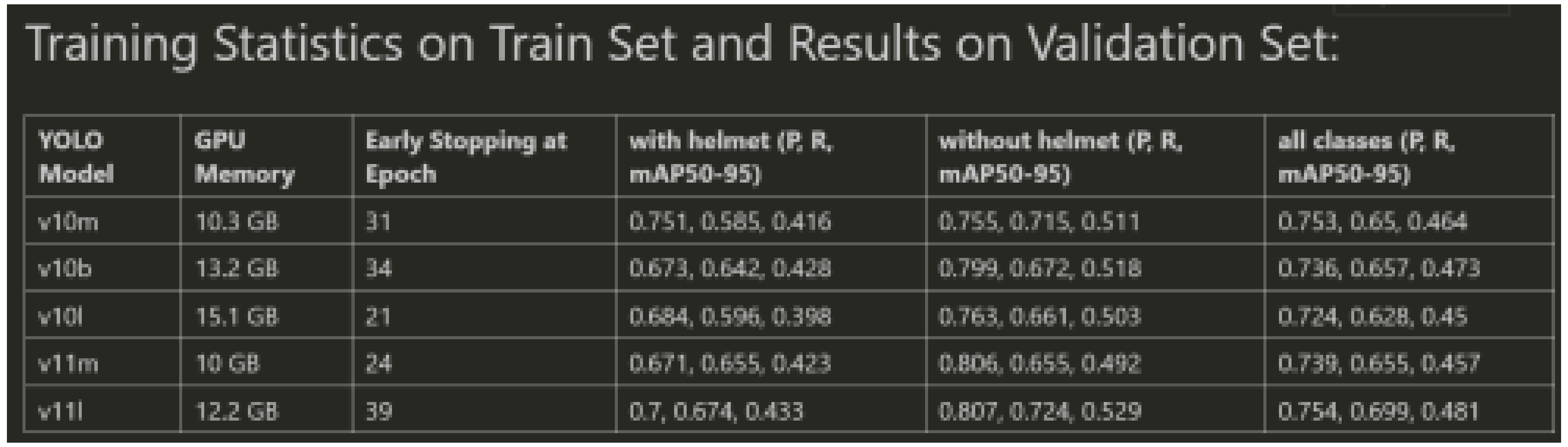


Figure 2. Class distribution across Violation Detection Dataset



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)
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 5. Training Statistics and Results for Violation Detection Model

Additional Analysis: *Feature Extraction, Threshold Determination*

Brightness

- What: Mean of HSV Value channel (np.mean(hsv[:,:,2]))
- Why: Distinguishes low-light vs. daylight frames—critical for detecting underexposed scenes.

Contrast

- What: Standard deviation of grayscale intensities (np.std(gray))
- Why: Separates flat-tone images from high-contrast ones, affecting model sensitivity to edges.

Color Variance

- What: Variance of 256-bin histograms for B, G, R channels (cv2.calcHist)
- Why: Quantifies scene complexity—busy backgrounds vs. uniform areas can confuse detectors.

Blur

- What: Variance of Laplacian on grayscale (cv2.Laplacian(gray).var())
- Why: Identifies motion blur or out-of-focus frames, which reduce detection accuracy.

Threshold Determination

- Brightness & Contrast Bins
  - Method: K-means clustering (k=3) on each feature's full distribution
  - Thresholds:
    - Brightness: Low < 70 | Medium 70–130 | High > 130
    - Contrast: Low < 45 | Medium 45–90 | High > 90
  - Why K-means? Automatically finds natural cutoff points, avoids arbitrary splits.

- Blur Cutoff
  - Method: Locate valley between two peaks in Laplacian-variance histogram
  - Threshold: Variance < 100 ⇒ Blurry; ⩾ 100 ⇒ Sharp
  - Why histogram valley? Data-driven separation robust to outliers and varied blur levels.

Image Segmentation & Insights

Composite Category Assignment

- Total Categories: 3 Light × 3 Contrast × 2 Sharpness = 18
- Per-Image Logic:
  a. Compute brightness → Low/Med/High
  b. Compute contrast → Low/Med/High
  c. Compute blur_var → Blurry (<100) / Sharp (⩾100)
  d. Folder name = "<Light> – <Contrast> – <Sharp/Blurry>"
- Why segment? Enables targeted evaluation of model performance under every unique visual condition.

Model Performance Under Varying Visual Conditions:

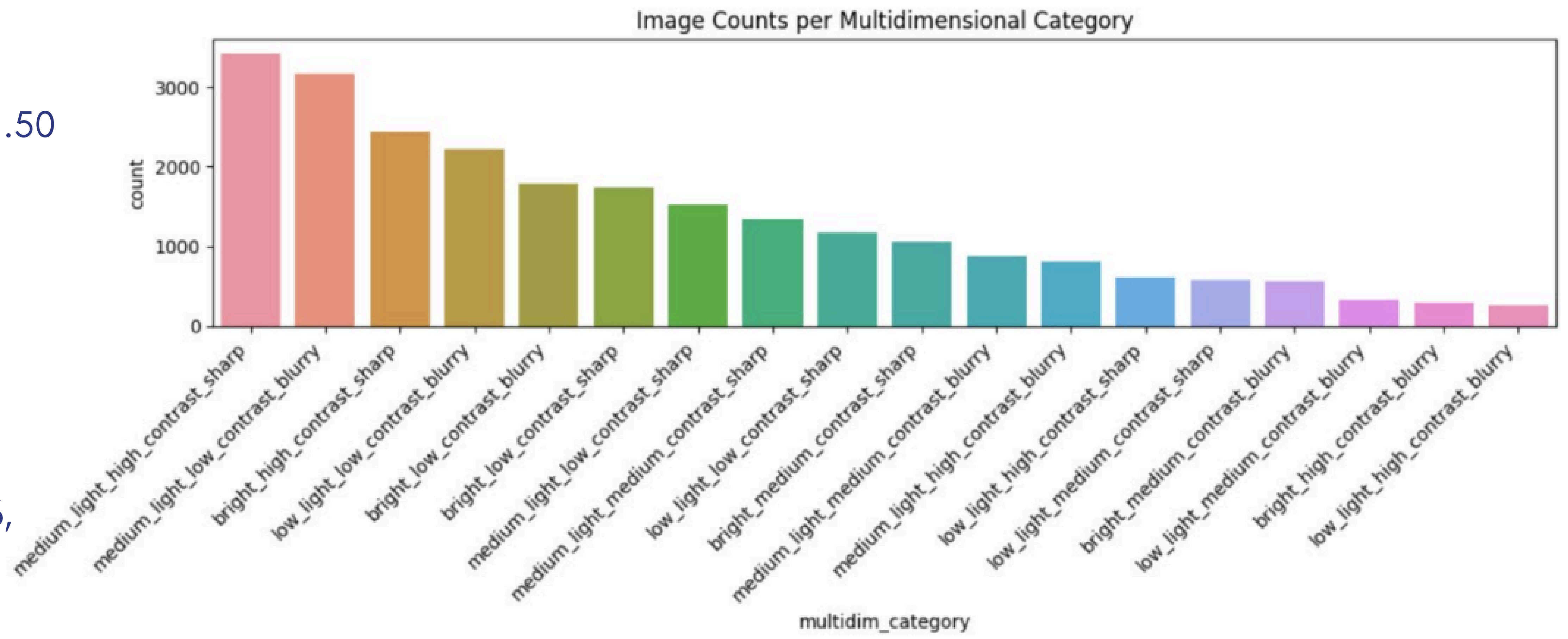
- Bright low contrast blurry (best): AP [.50–.95] 0.7642, AP @.50 0.9587, AP @.75 0.9180
- Low light low contrast blurry: AP [.50–.95] 0.6005, AP @.50 0.9301, AP @.75 0.6586
- Bright low contrast sharp: AP [.50–.95] 0.6927, AP @.50 0.8502, AP @.75 0.7860
- Bright medium contrast blurry (worst): AP [.50–.95] 0.4243, AP @.50 0.8327, AP @.75 0.3891

Highest AP when well-lit, even if blurry

Worst in "bright + medium contrast + blurry" (texture confusion)

Dataset can be accessed here

https://www.kaggle.com/datasets/mahimann272/data-analysis-dataset/data

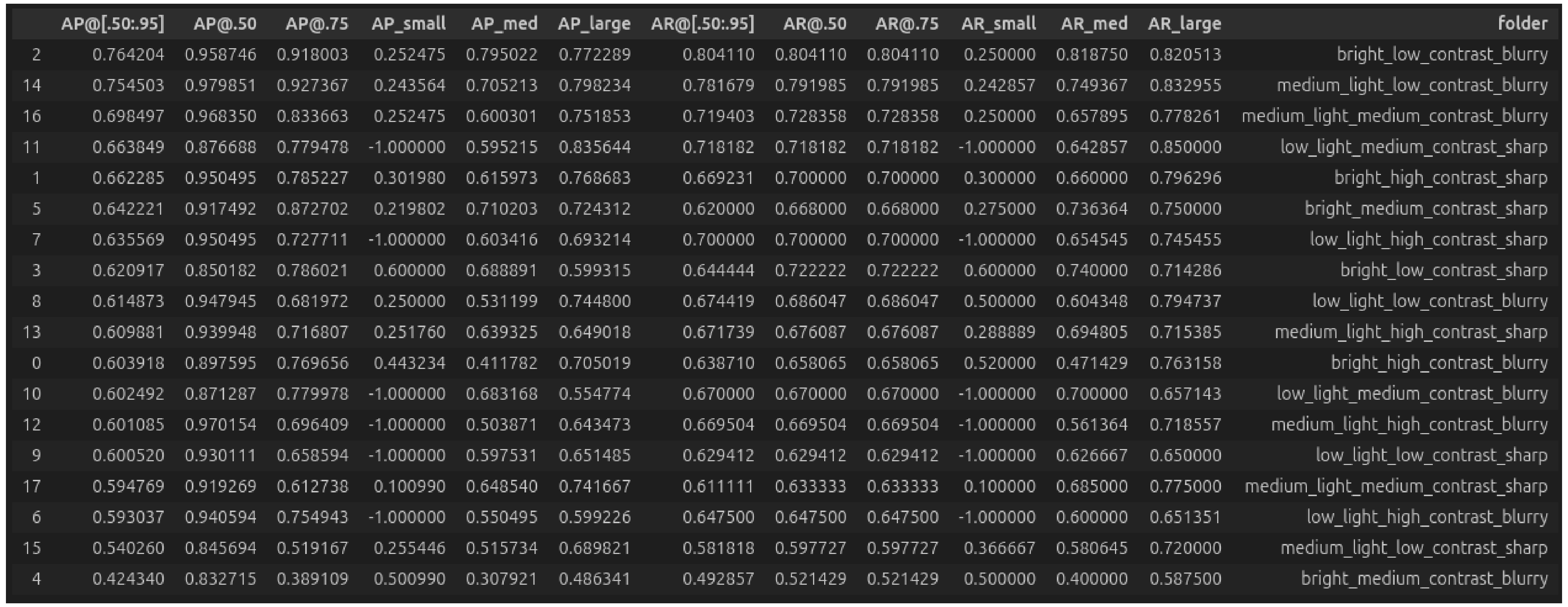


Additional Evaluation : Major Inferences:

The COCO evaluation protocol (especially AP@[.50:.95]) is widely used in object detection tasks.

✓ It provides a comprehensive view of detection quality across multiple IoU thresholds.

✓ Helps compare models not just on high-confidence detections (AP@.50), but on precise localization as well.



	AP@[.50:.95]	AP@.50	AP@.75	AP_small	AP_med	AP_large	AR@[.50:.95]	AR@.50	AR@.75	AR_small	AR_med	AR_large	folder
2	0.764204	0.958746	0.918003	0.252475	0.795022	0.772289	0.804110	0.804110	0.804110	0.250000	0.818750	0.820513	bright_low_contrast_blurry
14	0.754503	0.979851	0.927367	0.243564	0.705213	0.798234	0.781679	0.791985	0.791985	0.242857	0.749367	0.832955	medium_light_low_contrast_blurry
16	0.698497	0.968350	0.833663	0.252475	0.600301	0.751853	0.719403	0.728358	0.728358	0.250000	0.657895	0.778261	medium_light_medium_contrast_blurry
11	0.663849	0.876688	0.779478	-1.000000	0.595215	0.835644	0.718182	0.718182	0.718182	-1.000000	0.642857	0.850000	low_light_medium_contrast_sharp
1	0.662285	0.950495	0.785227	0.301980	0.615973	0.768683	0.669231	0.700000	0.700000	0.300000	0.660000	0.796296	bright_high_contrast_sharp
5	0.642221	0.917492	0.872702	0.219802	0.710203	0.724312	0.620000	0.668000	0.668000	0.275000	0.736364	0.750000	bright_medium_contrast_sharp
7	0.635569	0.950495	0.727711	-1.000000	0.603416	0.693214	0.700000	0.700000	0.700000	-1.000000	0.654545	0.745455	low_light_high_contrast_sharp
3	0.620917	0.850182	0.786021	0.600000	0.688891	0.599315	0.644444	0.722222	0.722222	0.600000	0.740000	0.714286	bright_low_contrast_sharp
8	0.614873	0.947945	0.681972	0.250000	0.531199	0.744800	0.674419	0.686047	0.686047	0.500000	0.604348	0.794737	low_light_low_contrast_blurry
13	0.609881	0.939948	0.716807	0.251760	0.639325	0.649018	0.671739	0.676087	0.676087	0.288889	0.694805	0.715385	medium_light_high_contrast_sharp
0	0.603918	0.897595	0.769656	0.443234	0.411782	0.705019	0.638710	0.658065	0.658065	0.520000	0.471429	0.763158	bright_high_contrast_blurry
10	0.602492	0.871287	0.779978	-1.000000	0.683168	0.554774	0.670000	0.670000	0.670000	-1.000000	0.700000	0.657143	low_light_medium_contrast_blurry
12	0.601085	0.970154	0.696409	-1.000000	0.503871	0.643473	0.669504	0.669504	0.669504	-1.000000	0.561364	0.718557	medium_light_high_contrast_blurry
9	0.600520	0.930111	0.658594	-1.000000	0.597531	0.651485	0.629412	0.629412	0.629412	-1.000000	0.626667	0.650000	low_light_low_contrast_sharp
17	0.594769	0.919269	0.612738	0.100990	0.648540	0.741667	0.611111	0.633333	0.633333	0.100000	0.685000	0.775000	medium_light_medium_contrast_sharp
6	0.593037	0.940594	0.754943	-1.000000	0.550495	0.599226	0.647500	0.647500	0.647500	-1.000000	0.600000	0.651351	low_light_high_contrast_blurry
15	0.540260	0.845694	0.519167	0.255446	0.515734	0.689821	0.581818	0.597727	0.597727	0.366667	0.580645	0.720000	medium_light_low_contrast_sharp
4	0.424340	0.832715	0.389109	0.500990	0.307921	0.486341	0.492857	0.521429	0.521429	0.500000	0.400000	0.587500	bright_medium_contrast_blurry

Dataset Distribution: Unequal image distribution per condition may skew results and affect metric reliability.

Best Performing Condition:

bright.low_contrast.blurry achieves the highest performance

→ AP@[.50:.95] = 0.7642

➤ Indicates blur is not a limiting factor in well-lit conditions.

Worst Performing Condition:

bright.medium_contrast.blurry records the lowest performance

→ AP@[.50:.95] = 0.4243

➤ Likely caused by confusing textures and low lighting gradients.

Lighting vs. Sharpness:

bright.low_contrast.sharp outperforms low_light.low_contrast.blurry

➤ Suggests lighting has a stronger impact on detection than image sharpness.

Final Results

Additional Model Performance on Validation Set:

- YOLOv8m: Precision 0.752, Recall 0.701, mAP50-95 0.489
- YOLOv8l: Precision 0.769, Recall 0.694, mAP50-95 0.485
- YOLOv9m: Precision 0.702, Recall 0.704, mAP50-95 0.479
- YOLOv9c: Precision 0.722, Recall 0.680, mAP50-95 0.475

Best performer: YOLOv8m with mAP 0.489, surpassing all previous variants.

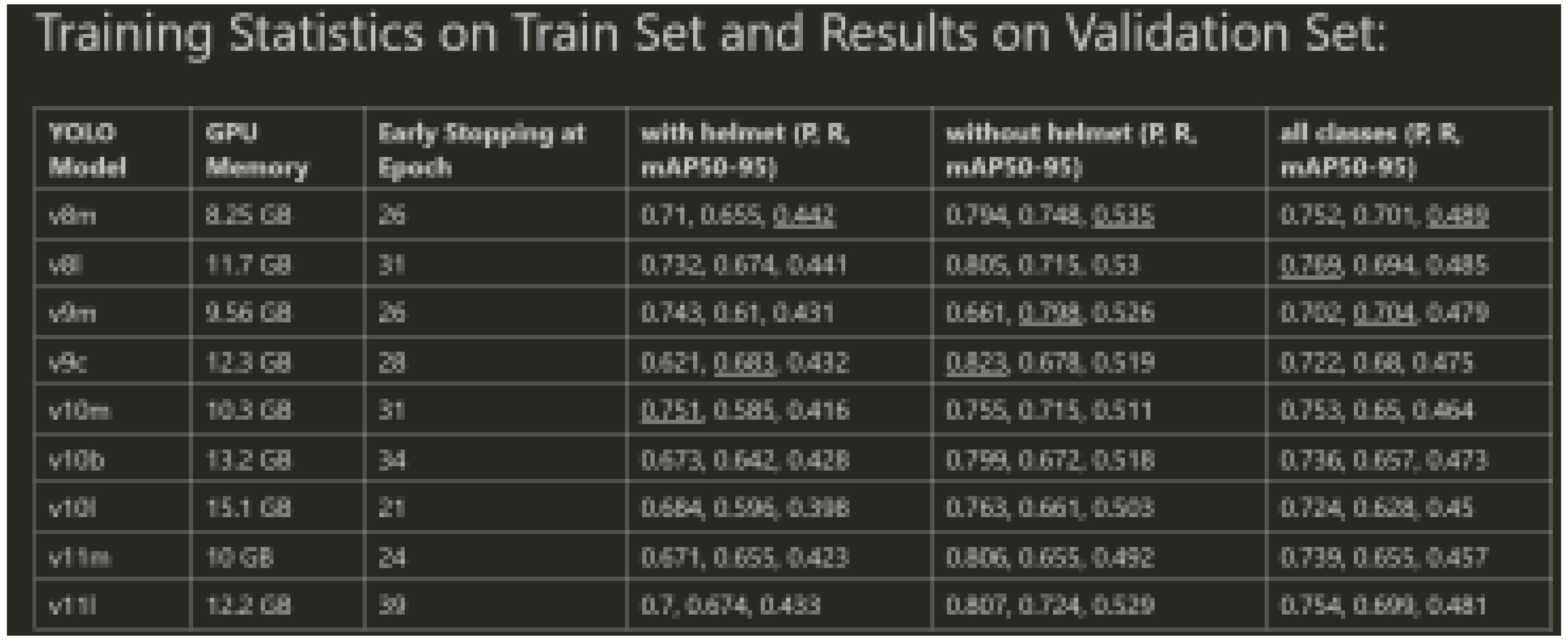
YOLOv8l achieves highest precision (0.769), though mAP slightly lower than YOLOv8m.

License-Plate Detection

- Model: YOLOv11n
- Precision: 0.988
- Recall: 0.960
- mAP [.50-.95]: 0.709

OCR Module

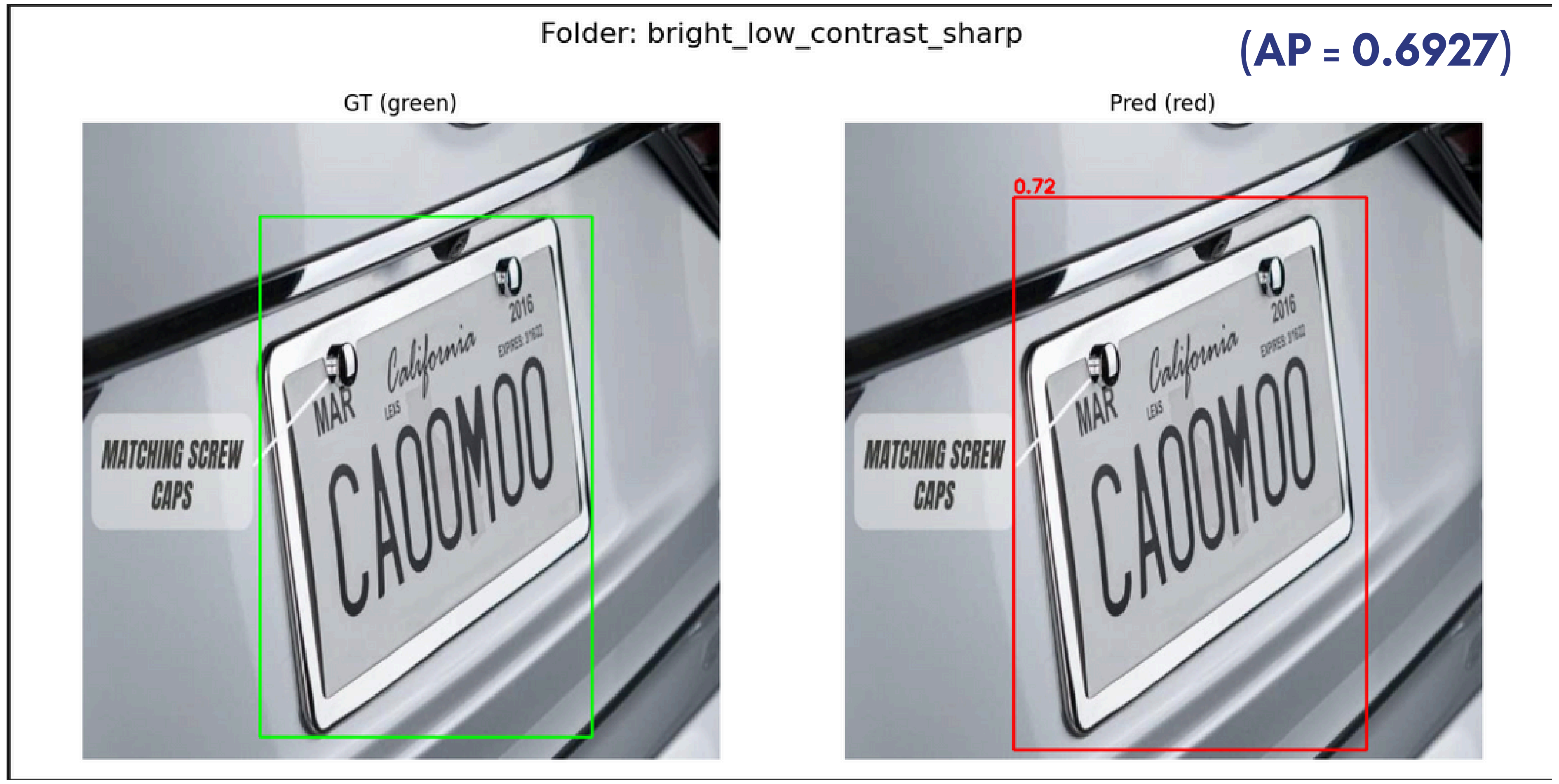
- OCR engine: EasyOCR integrated with detection output
- Current status: pipeline wired → detection → crop → OCR call
- Validation: preliminary tests on 100 plates → ˜90 % character accuracy



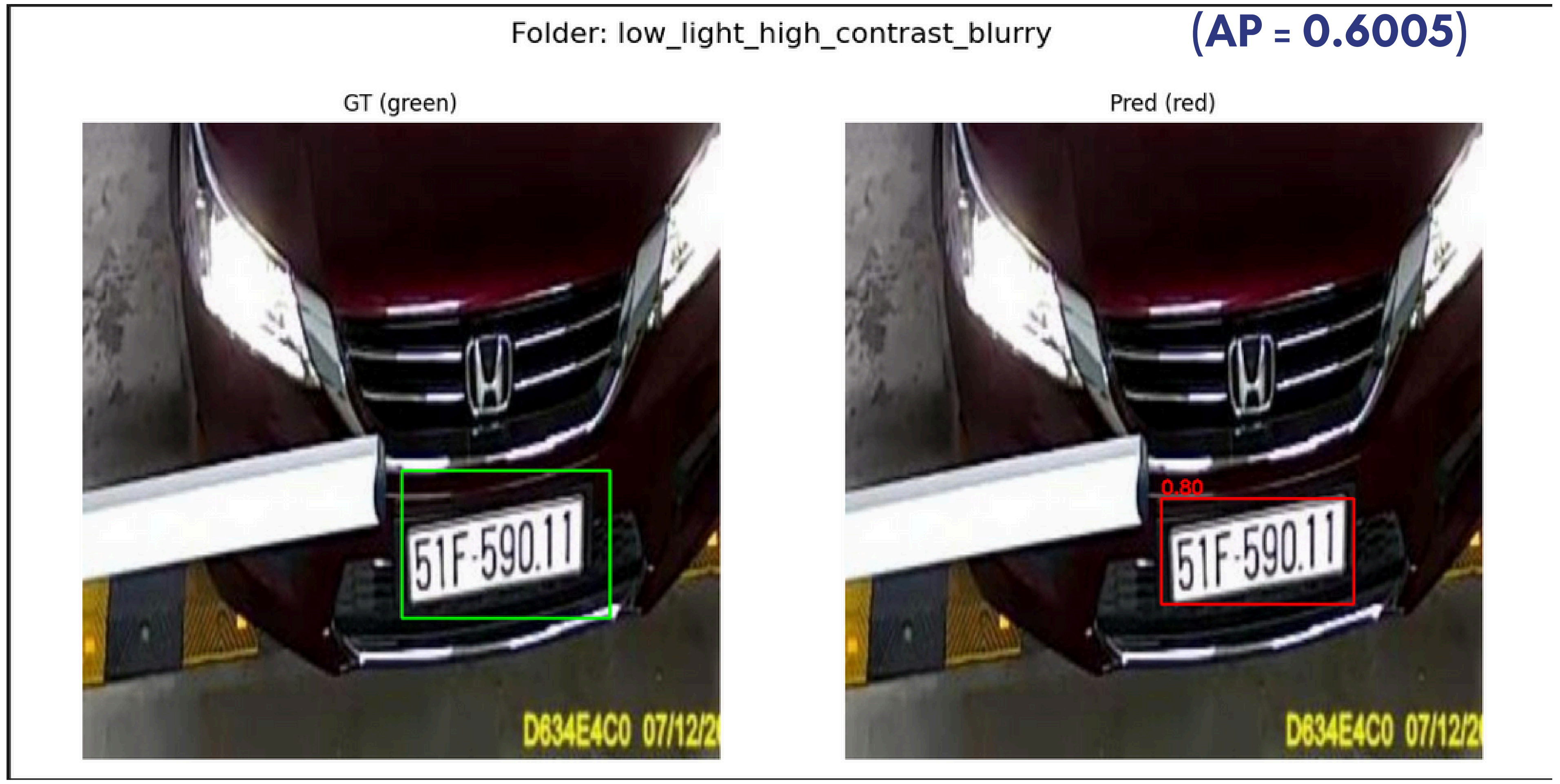
Training Statistics on Train Set and Results on Validation Set:

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

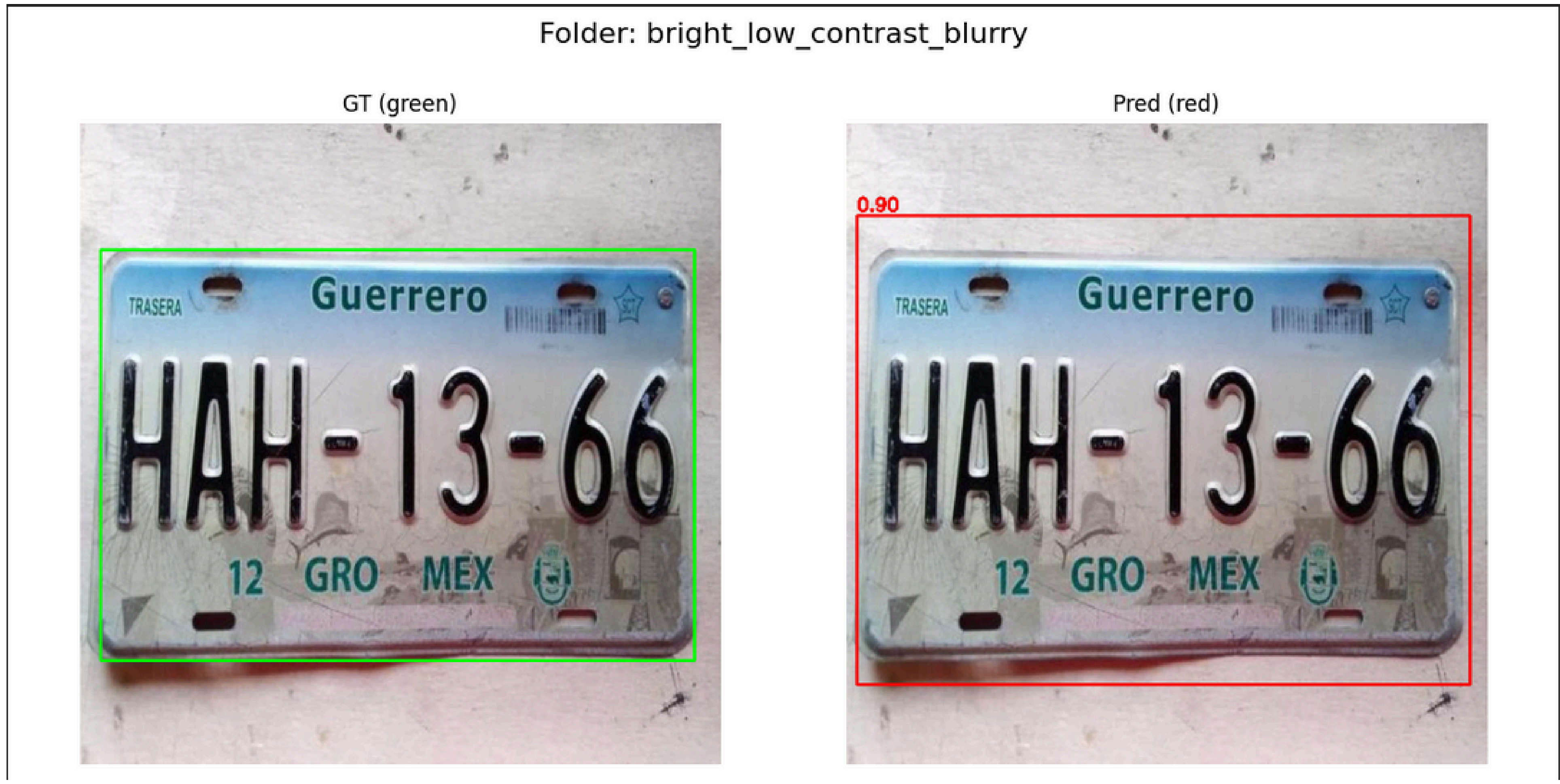
Figure 6. Training Statistics and Results for Violation Detection Model



(AP = 0.6927)



(AP = 0.6005)

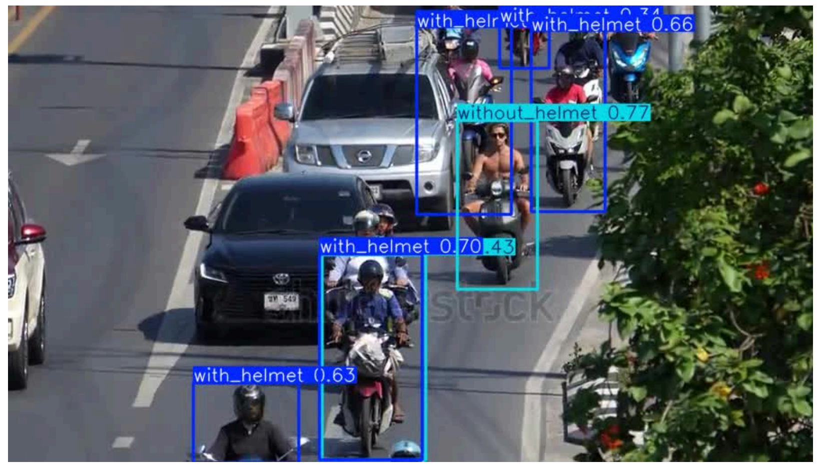


Best Performing Model:

(AP = 0.7642)

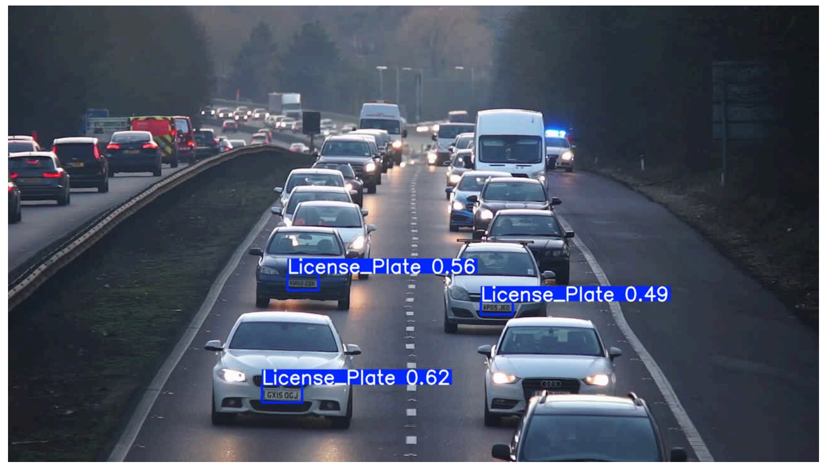
Demo of Violation detection and License Detection

Input Video

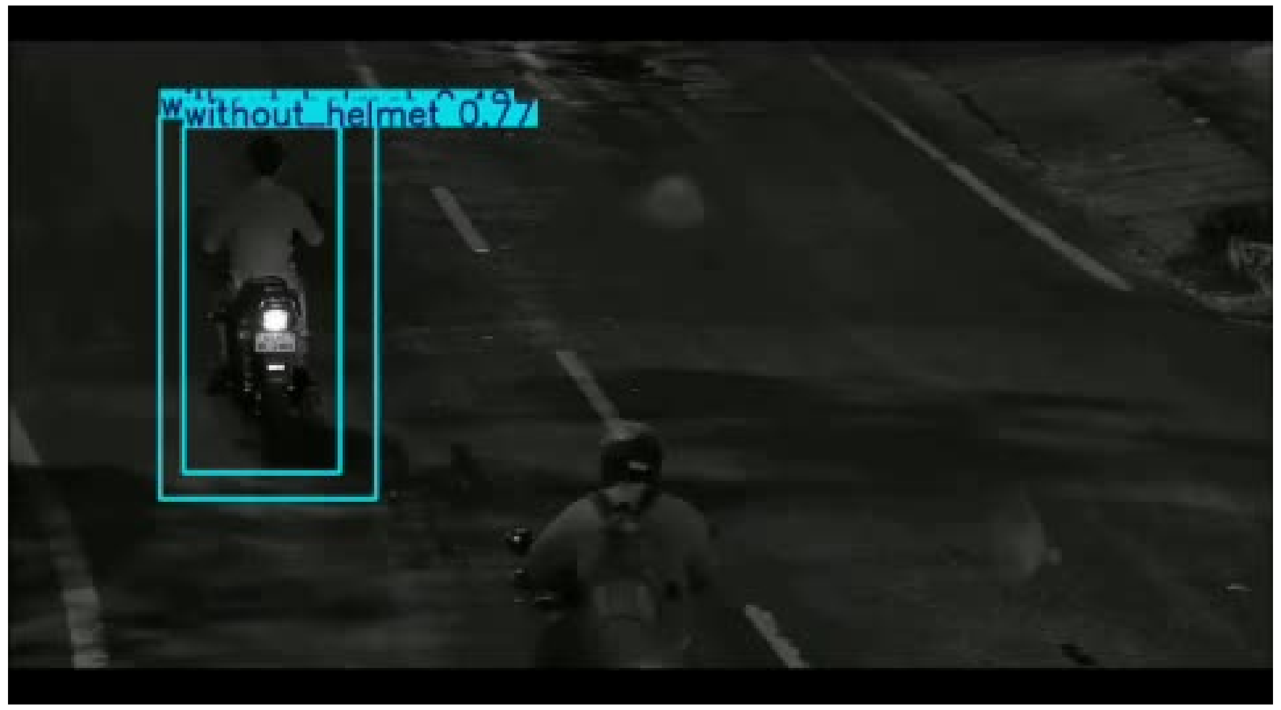


output Video

Input Video



output Video



Final output Video

Key Takeaways & Individual Contributions

- Two-stage pipeline (helmet → plate) robust across models.
- YOLOv8m leads with 0.489 mAP50-95 for helmet violation.
- Visual-condition segmentation reveals lighting > sharpness.
- OCR integrated end-to-end; license-extraction accuracy ˜90 %.

Individual Contributions:

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

Feedback from the review meeting

Quantitative Error Analysis Beyond Standard Evaluation Metrics

- Perform a quantitative investigation of failure cases to identify where and why the model misclassified.
- Conduct a numerical breakdown to analyze poor-performing classes in detail.

Impact of Lighting and Visual Conditions on Detection Accuracy

- Analyze the effect of varying lighting conditions (day, night, fog, and low-light scenarios) on the performance of license plate and vehicle detection.
- Identify common failure points due to poor visibility, glare, or shadows.

Enhanced Post-Detection Inference for Real-World Applications

- Implement additional inference techniques after YOLO model detection to extract meaningful insights from the detected objects.
- Apply object tracking to analyze movement patterns and detect anomalies such as illegal turns or abrupt stops.
- Use proximity analysis to assess collision risks and identify traffic violations.

Next Steps:

- Error Analysis Improvement: Perform detailed failure analysis by visualizing misclassified detections.
- Enhance Detection Under Varying Conditions: Apply image enhancement techniques (e.g., histogram equalization or CLAHE) to improve detection in low-light environments.
- Develop rule-based logic to detect violations like illegal turns or abrupt stops.