CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

Conference Paper

YOLO-Powered Detection of Helmets and Number Plates for Enforcing Safety on Wheels

Nithish Reddy
Radha Reddy*
Rajesh Mahadeva

*CISTER Research Centre

CISTER-TR-251101

2025/11/17

YOLO-Powered Detection of Helmets and Number Plates for Enforcing Safety on Wheels

Nithish Reddy, Radha Reddy*, Rajesh Mahadeva

*CISTER Research Centre

Polytechnic Institute of Porto (ISEP P.Porto)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail: mamidi.reddy@learner.manipal.edu, reddy@isep.ipp.pt, rajesh.mahadeva@manipal.edu

https://www.cister-labs.pt

Abstract

Road safety is becoming an increasingly important issue, particularly in a country like India, where two-wheelers are a common mode of transport. A major concern is that many riders do not wear helmets, which can lead to serious injuries or even death in the event of an accident. This paper proposes an intelligent system that automatically recognizes riders not wearing helmets and saves their images and vehicle number plates using a combination of computer vision and deep learning. This system uses the YOLO object detection algorithm to locate riders, helmets, and number plates in real-time video streams. A custom-trained optical character recognition model is then used to extract and recognize the characters on the plate. All recognized number plates are stored in a CSV file, from which we can send fines to the corresponding riders.

YOLO-Powered Detection of Helmets and Number Plates for Enforcing Safety on Wheels

Nithish Reddy
Manipal Institute of Technology
Manipal Academy of Higher Education
Manipal, India
mamidi.reddy@learner.manipal.edu

Radha Reddy*
Manipal Institute of Technology
Manipal Academy of Higher Education
Manipal, India
CISTER Research Centre, Porto, Portugal
radhakrishna.reddy@manipal.edu

Rajesh Mahadeva*
Manipal Institute of Technology
Manipal Academy of Higher Education
Manipal, India
rajesh.mahadeva@manipal.edu

2025 Artificial Intelligence and Smart Technologies for Sustainability Conference (AISTS) | 979-8-3315-9852-5/25/$31.00 ©2025 IEEE | DOI: 10.1109/AISTS66100.2025.11233121

***Abstract*—Road safety is becoming an increasingly important issue, particularly in a country like India, where two-wheelers are a common mode of transport. A major concern is that many riders do not wear helmets, which can lead to serious injuries or even death in the event of an accident. This paper proposes an intelligent system that automatically recognizes riders not wearing helmets and saves their images and vehicle number plates using a combination of computer vision and deep learning. This system uses the YOLO object detection algorithm to locate riders, helmets, and number plates in real-time video streams. A custom-trained optical character recognition model is then used to extract and recognize the characters on the plate. All recognized number plates are stored in a CSV file, from which we can send fines to the corresponding riders.**

***Index Terms*—Helmet detection, rider monitoring, number plate recognition, and road safety.**

I. Introduction

In India, motorized two-wheelers (MTW) play a significant role in cities where people ride them daily. According to the Ministry of Road Transport and Highways of India [1] 2018-19, MTW continued to dominate the Indian vehicle landscape, accounting for 74.8% of all registered motor vehicles and 79.5% of new vehicle registrations. They also made up 83.3% of all non-transport vehicles nationally and 79% in millions of cities, highlighting their widespread use for personal mobility. Additionally, MTW represented 73.9% of India's total automobile exports, underscoring their importance domestically and in global markets. The other side of the story is that they account for 28.8% of total road injuries [2], and 37% of those killed in road accidents in 2019 were MTW riders. More than 43,000 MTW riders died in road accidents in 2022 alone, and many of those fatalities were attributed to not wearing helmets [14].

Therefore, the 2019 Motor Vehicles (Amendment) Act requires helmets. Ensuring riders wear helmets is essential, as checking manually can be difficult, especially in heavy traffic scenarios. This necessitates an intelligent system that monitors riders and identifies those who do not comply with helmet regulations without human intervention. This paves the way for intelligent transportation systems towards safe and sustainable cities [5].

Hence, this work aims to create an intelligent system based on deep learning to spot MTW riders from real-time video captured through road cameras. The idea is to check whether the riders have helmets, and if not, save their picture along with the number plate. For this purpose, the system uses You Only Look Once version 8 (YOLOv8) [6] to find objects such as the rider, the helmet, and the number plate in each video frame. When it finds a rider without a helmet, the system captures the rider's image and saves the number plate image for the next steps.

The system begins by enhancing the image used to read the license plate. It sharpens the picture, removes noise, and clarifies the image to isolate letters and numbers. A convolutional neural network (CNN) model based on ResNet-18 identifies these characters accurately, having been trained to recognize them effectively [7]. After the characters are detected, the system saves the license plate number and associated traffic violations in a CSV file. This intelligent system simplifies and streamlines traffic monitoring, reducing the need for manual checks. Additionally, it contributes to road safety efforts by automatically identifying violations of helmet regulations [9].

The remainder of this paper is organized as follows. Section II reviews pertinent related works, while Section III describes the proposed methodology. Section IV examines the system architecture and Section V presents the experimental results. Finally, Section VI summarizes the key conclusions and outlines future work.

II. Related Work

Helmet detection and number plate recognition have become increasingly important research areas, especially with the growing demand for intelligent traffic management systems. Many researchers have worked on these tasks separately using classical and deep learning approaches.

A. Helmet Detection

Besides the manual checking, traditional computer vision methods such as color filtering [8] and background subtraction [8] techniques were used in the early stages of helmet detection. However, these methods proved unreliable in real-world environments, where lighting and backgrounds vary constantly. With the advancement of deep learning, more sophisticated models like Faster R-CNN [16], SSD [15], and

especially the YOLO series gained popularity due to their ability to detect objects in real-time [4].

Among these models, YOLOv3 [4] and YOLOv4 [4] were frequently used in previous helmet detection studies because they balanced accuracy and processing speed. Subsequently, YOLOv5 [11] improved detection accuracy and simplified the training process, leading to its widespread adoption in traffic safety applications. The latest version, YOLOv8 [6], performs even better and effectively detects small objects like helmets in heavy traffic scenarios.

B. Number Plate Recognition and OCR

Automatically reading number plates has traditionally relied on Optical Character Recognition (OCR) tools like Tesseract, which perform well under ideal conditions [12]. However, their effectiveness diminishes when the image quality is poor or the text appears distorted or blurry. To address these limitations, researchers have turned to deep learning-based OCR approaches [10]. One standard method involves segmenting and classifying individual characters using CNNs [7]. Another approach employs Convolutional Recurrent Neural Networks (CRNNs) [16], which can recognize entire text sequences in a single pass. A more versatile solution is EasyOCR[1], which integrates a robust text detector with a recognizer and is renowned for its ability to handle various fonts and image qualities.

C. Combined Detection and Recognition Systems

Some recent systems combine both helmet detection and number plate recognition. In many of these, object detection models like YOLOv3 or YOLOv5 are used to detect the rider and the number plate [4], [11], and then an OCR engine like EasyOCR or a CNN-based classifier is used to read the characters [10]. These integrated systems aim to provide a comprehensive solution for real-time traffic surveillance and enforcement. However, not many include fallback options in case one model fails. This is where our work stands out, using YOLOv8 for accurate detection and a custom CNN model for OCR recognition. If this model cannot confidently recognize the characters, the system switches to EasyOCR as a backup. This improves reliability and ensures that even difficult-to-read number plates are not missed.

III. Methodology

This work follows a structured and modular pipeline to detect MTW riders not wearing helmets and extract their vehicle number plates from video input. The entire system is divided into Object Detection and Optical Character Recognition. Each module was trained independently using relevant datasets and evaluated using standard performance metrics.

[1]https://github.com/JaidedAI/EasyOCR

A. Workflow

The system starts by processing video frames from a live camera feed or a pre-recorded video. Each frame is analyzed using a YOLOv8 model to detect riders, helmets, and license plates. If a rider is identified without a helmet, the system captures cropped images of the rider and the corresponding license plate. The license plate is then processed with an OCR system based on ResNet-18 [7]. If the output from the OCR is unclear or incomplete, EasyOCR is utilized as a backup. The final results, which include the recognized license plate number, timestamp, and saved images, are stored in a CSV file.

B. Object Detection

The system utilizes YOLOv8 (shown in Fig. 1), a real-time object detection model recognized for its speed and accuracy in detecting helmets, riders, and license plates. YOLOv8 executes object detection in a single step, making it ideal for live video analysis [6]. This model divides the input image into a grid and predicts bounding boxes and class labels for the objects within each grid cell. Unlike earlier object detection models that separate classification and localization into distinct steps, YOLOv8 integrates both processes into a fast and efficient network [4], [6], [11].

The model takes an input image and returns bounding box coordinates, confidence scores, and object class labels (e.g., helmet, rider, number plate). This one-shot detection mechanism allows the system to process multiple objects in a single frame quickly and accurately.

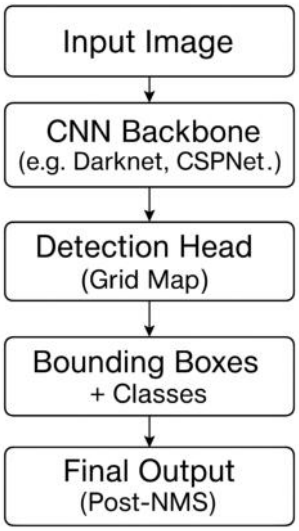

Fig. 1. YOLOv8 working pipeline.

The object detection model was trained using a custom-labeled dataset [17] created explicitly for this work. This dataset includes images of MTW riders captured under various real-world conditions, such as traffic, daylight, shadows, and partial occlusions. The dataset consists of approximately 1,200 images, which are divided into a training set (70%), a validation set (20%), and a test set (10%).

Therefore, each image includes annotations for three object classes: the rider, the helmet, and the number plate. The images were manually annotated using bounding boxes with tools like Roboflow, and the labels were saved in YOLO format. Each image has an associated `.txt` file containing the class label and normalized coordinates of each object in the frame. Data augmentation was applied during training to improve the model's generalization ability. This included random scaling, horizontal flipping, brightness and contrast adjustments, and random rotation.

The input image size for training was set to 416×416 pixels, and the model was trained for 50 epochs with a batch size of 4. The training was done using the YOLOv8n (nano) model, which balances speed and accuracy, making it suitable for real-time applications like helmet and number plate detection in videos. This carefully prepared dataset helped the model learn and detect small and overlapping objects accurately, as is often the case with riders and number plates in crowded traffic scenes.

C. Optical Character Recognition

A CNN-based ResNet-18 architecture, shown in Fig. 2, was used to recognize characters from license plates. ResNet-18 is a convolutional neural network (CNN) consisting of 18 layers. It addresses a common issue in deep networks, where adding more layers can sometimes degrade performance. This problem is referred to as the vanishing gradient problem. ResNet mitigates this issue through a process known as residual learning [7]. Instead of learning everything from scratch in each layer, ResNet-18 enables information to bypass certain layers via skip or shortcut connections. These shortcuts help the network learn faster and more effectively by allowing it to pass important features directly through the network without getting lost in the deeper layers.

In ResNet-18, the input image (in this case, a segmented character from a number plate) is passed through several convolutional layers, batch normalization, and activation functions. At certain points, the input of a layer is added directly to the output of another layer, and this is the residual part. These additions help the network focus on learning only the part of the output that is different from the input, which makes training easier and more stable. We considered ResNet-18 because it is lightweight and fast, making it suitable for real-time systems. It is deep enough to learn complex patterns, and its skip connections allow for efficient training, even with limited data.

After the ResNet-18 model processes the image of each character, it predicts the most likely class, which can be any letter from A to Z or any digit from 0 to 9. These predictions are then combined to reconstruct the full number plate. Once a number plate is cropped, it is preprocessed by converting it to grayscale, enhancing contrast, and removing noise [13]. Then, the characters are segmented using contour detection. Each character is resized and fed into the ResNet-18 model, which predicts the corresponding character label. Finally, all the predicted characters are combined to form the complete number plate string.

If the model fails to identify specific characters confidently or segments are missing, the system returns to EasyOCR, which performs full-plate recognition directly.

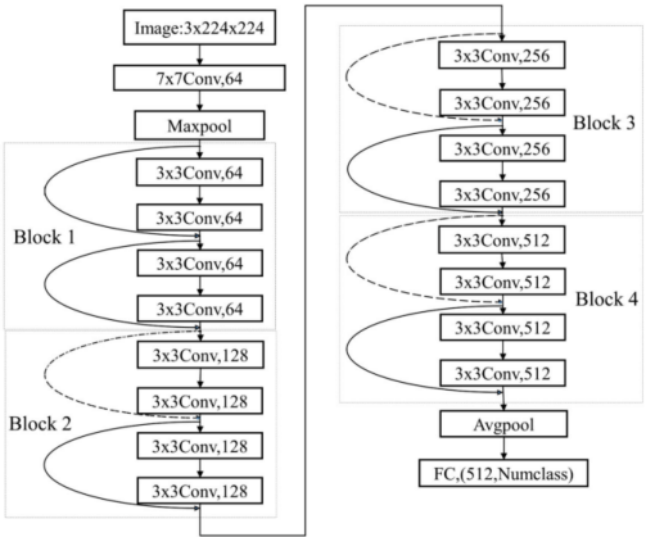

Fig. 2. ResNet-18 architecture.

For the character recognition task, a custom dataset was prepared to train a deep learning model capable of accurately identifying characters from Indian number plates [10] . The dataset includes images of individual characters ranging from digits 0–9 and alphabets A–Z, generated synthetically using different fonts, font sizes, weights (boldness), and styles. This variety was essential to mimic the real-world diversity of license plate typography. The dataset also included variations in lighting, contrast, and slight distortions to enhance realism. Each character image was labeled accordingly and stored in class-specific directories. The images were resized to 224 × 224 pixels to match the input requirements of the CNN model. The model used for OCR was based on the ResNet-18 architecture, which is known for its efficiency and good performance in image classification tasks. The training process involved:

- A total of 10,000+ character images spread across 36 classes (0–9, A–Z).
- Cross-entropy loss was used as the loss function.
- Adam optimizer was applied with a learning rate of 0.001.
- Training was conducted over 10 epochs with a batch size of 64

The model demonstrated smooth convergence throughout the training with increasing accuracy and decreasing loss. The final training accuracy reached approximately 99.6%, showing that the model was well-tuned to recognize characters under a wide variety of conditions. This trained model is critical in extracting plate numbers from the system pipeline's detected and preprocessed number plate regions.

IV. System Architecture

The system architecture shown in Fig. 3 consists of three main parts: object detection, violation processing, and OCR logging. Each module performs its task and passes the output to the next stage.

Input Module - The system takes video as an input source. The video contains footage from traffic cameras or pre-recorded clips of moving bikers. The system also supports live video streams, making real-world deployment practical in real-time monitoring applications.

Object Detection Module - This module uses the YOLOv8 object detection model to identify three objects in each frame, including the motorcycle rider, helmet, and number plate.

YOLOv8 provides high-speed and accurate multi-object detection, which is ideal for real-time traffic monitoring scenarios.

Helmet Violation Check - For every detected rider, the system checks whether a helmet is present near or within the rider's bounding box. If a helmet is not present, a violation is detected.

Cropping and Image Saving - The system crops and saves the rider's and corresponding number plate images when a violation is detected. These are stored in their respective directories for further processing.

Preprocessing the Number Plate - The cropped number plate image is often noisy or blurry, so it is first preprocessed to improve quality. The system converts the image to grayscale, sharpens the edges, removes noise, and adjusts contrast. This helps the OCR model read the characters more accurately.

Character Segmentation and OCR - Next, the cleaned number plate is segmented into individual characters. Each character is passed through a CNN-based model trained to recognize characters (ResNet-18). If CNN-based recognition fails or is incomplete, the system returns to EasyOCR to extract the full plate text.

Output and Logging - Finally, the system stores the violation details in a CSV file, which includes the timestamp, recognized number plate, and paths to saved rider and plate images.

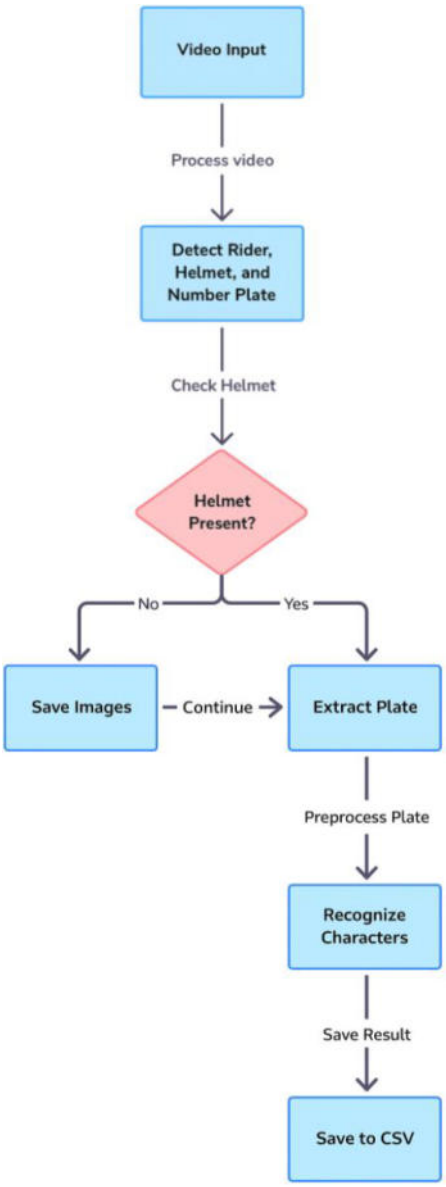

Fig. 3. Proposed system architecture of helmet and number plate detection system.

V. Results Discussion

This section presents and interprets the object detection and OCR module results used in the proposed system. The results demonstrate how accurately the model performs in real-world scenarios, how effectively it recognizes riders without helmets, and how effectively it identifies number plates.

A. Object Detection Results

The YOLOv8 model was trained on a custom dataset consisting of images labeled with three classes: rider, helmet, and number plate. After 50 epochs of training, the object detection model achieved strong detection performance:

- The mean Average Precision (mAP@0.5) converged to around 0.93, showing the model's ability to localize and classify objects of interest accurately.
- The classification loss continued to decrease, indicating the proper convergence of the model.
- The confusion matrix showed that the model could distinguish between riders and helmets effectively, even in dense traffic scenes.

Graphs such as the Precision-Recall curve and Confidence-F1/Precision/Recall curves shown in Fig. 4 demonstrated that the model performed best when the confidence threshold was set between 0.1 and 0.2. The model's precision remained high across all three classes, with the rider class yielding the most consistent performance.

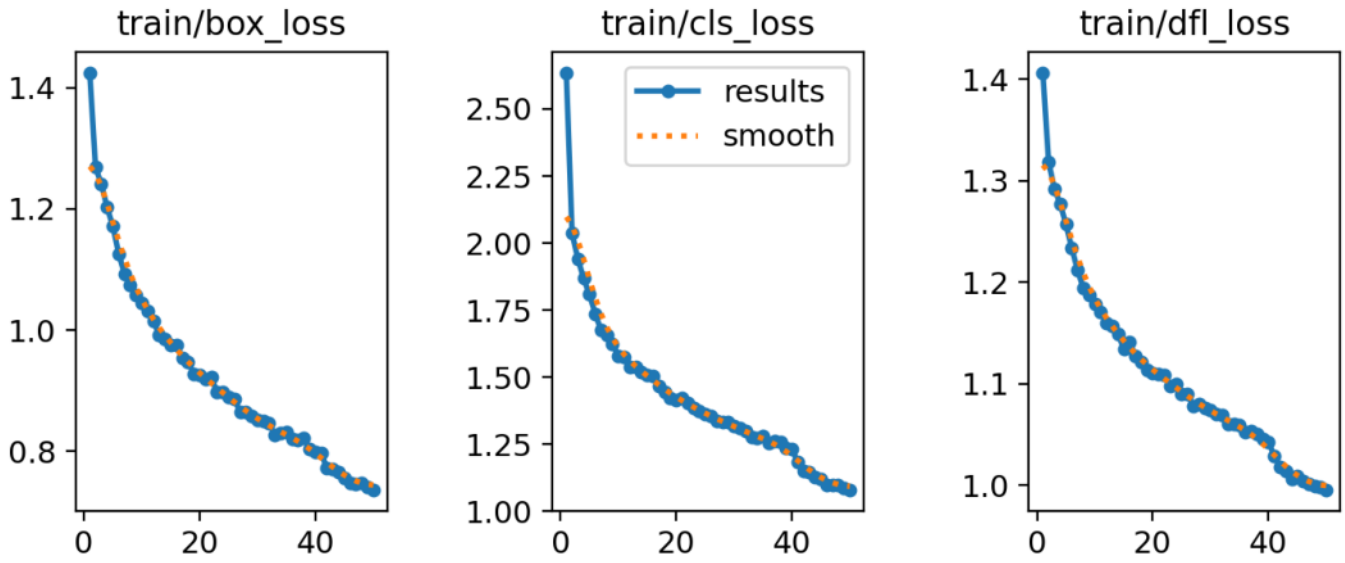

Fig. 4. YOLOv8 training loss curve

B. Visual Output

Whenever all three object types (rider, helmet, and number plate) are detected and classified, a sample frame is extracted and saved to validate the system visually. One such frame is shown in Fig. 5

Fig. 5. Sample output frame showing rider without helmet and number plate

In this frame:

- The rider is detected with a bounding box and confidence score.
- The helmet region is localized, and the system classified it as absent (highlighted in red).
- The number plate is identified, and the image is saved for OCR processing.

This visualization confirms that the detection and classification components are synchronized, even in dynamic road scenes.

C. OCR Model and Character Recognition Results

We used a deep learning model based on the ResNet-18 architecture to recognize the characters on the number plates. This model was trained using a synthetic dataset consisting of all English alphabets (A–Z) and digits (0–9), generated with various font styles, sizes, and boldness to ensure the model could generalize well under real-world conditions.

The model was trained for 10 epochs and achieved an impressive training accuracy of 99.6%, with a steadily decreasing loss, indicating strong learning behavior and minimal overfitting, as shown in Fig. 6.

To ensure high OCR accuracy, we adopted a two-step preprocessing approach before feeding the number plate images to the model:

- **White Plate Extraction:** The white region of the plate was extracted using HSV color thresholding to isolate characters and remove background interference [13].
- **Image Enhancement:** This region was then converted to grayscale, sharpened, denoised, and histogram-equalized to improve character legibility.

We demonstrated this process using three sample images.

1) The raw cropped number plate of the detected vehicle.
2) The result after extracting the white plate region.
3) The fully preprocessed plate, with enhanced clarity for OCR input.

In some cases where the plate image was noisy, blurred, or partially obscured, our custom OCR model could struggle. To address this, we integrated EasyOCR as a fallback mechanism. If the custom ResNet-18 model failed to produce complete or confident predictions, EasyOCR was used to ensure character recognition was still possible. This hybrid approach significantly increased the reliability of the OCR pipeline in varying real-world scenarios.

Table I compares ResNet-18 and EasyOCR in the test set using four metrics: Accuracy, Precision, Recall, and F1 Score. The results show that EasyOCR consistently outperforms ResNet-18 in all metrics. For example, EasyOCR

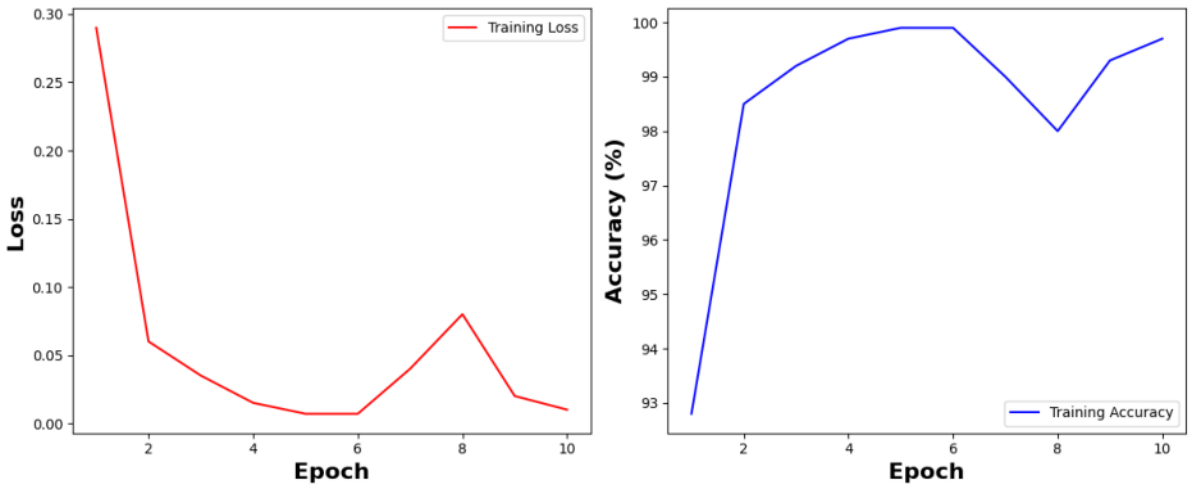

Fig. 6. Curves of training loss (left side) and training accuracy (right side).

TABLE I
EVALUATION METRICS OF RESNET-18 VS EASYOCR ON TEST SET

Model	Accuracy	Precision	Recall	F1 Score
ResNet-18	0.880	0.797	0.866	0.822
EasyOCR	0.953	0.908	0.943	0.922

achieves an accuracy of 95.3%, significantly higher than ResNet-18's 88.0%, indicating a superior overall prediction performance. Its precision of 0.908, compared to 0.797 for ResNet-18, suggests fewer false positives. Similarly, recall is higher for EasyOCR (0.943) than for ResNet-18 (0.866), reflecting a better true positive identification. The F1 Score, which represents the harmonic mean of precision and recall, further confirms this trend: EasyOCR achieves 0.922, outperforming ResNet-18's 0.822. These findings suggest that EasyOCR offers more robust and reliable performance for optical character recognition tasks in the given test scenario, making it a more suitable candidate for real-world deployment where accuracy and consistency are critical.

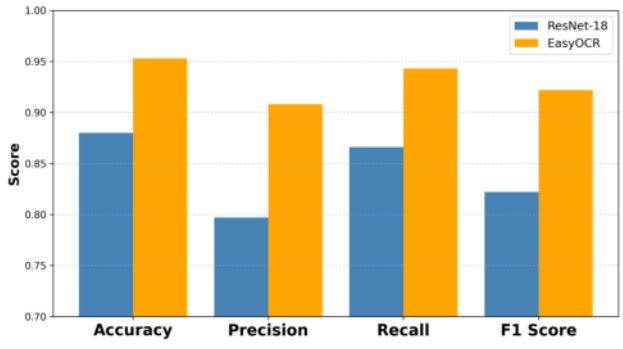

Fig. 7. Comparison of Accuracy, Precision, Recall, and F1 Score between ResNet-18 and EasyOCR.

Figure 7 compares the results of key ResNet-18 and EasyOCR evaluation metrics (represented in Table I) on a bar graph. Overall, EasyOCR performs consistently better in all metrics, and ResNet-18 provides a lightweight competitive solution with faster inference.

To gain a deeper understanding of the limitations of each OCR system, we present a combined confusion matrix in Fig. 8, highlighting the most frequent character-level misclassifications by ResNet18 and EasyOCR. The ResNet-18 matrix reveals notable confusion between visually similar characters such as '0' and 'O,' '8' and 'B,' and '1' and 'I,' with several non-diagonal entries indicating frequent misclassifications.

The findings indicate that ResNet-18 has difficulty distinguishing visually similar characters, particularly in real-world situations where fonts, lighting, and image quality vary. In contrast, EasyOCR demonstrates strong diagonal dominance in its confusion matrix, suggesting a higher rate of correct predictions and significantly less confusion among ambiguous character pairs. This ability reflects greater robustness and aligns with the superior accuracy, precision, and recall metrics noted earlier. Overall, EasyOCR is a more dependable solution for character recognition tasks, making it especially suitable for applications like license plate reading, where accurate and consistent character interpretation is crucial.

D. Final Output Comparison

We used a preprocessing pipeline to extract only the white plate region from the original image. This enhanced image underwent the same OCR process using our trained ResNet-18 model. The character segmentation succeeded, and the model predicted the correct output `MHVAAA0000` as shown in Fig. 9.

VI. CONCLUSION

This paper presents a deep learning-based solution for automatically detecting two-wheeler riders without helmets

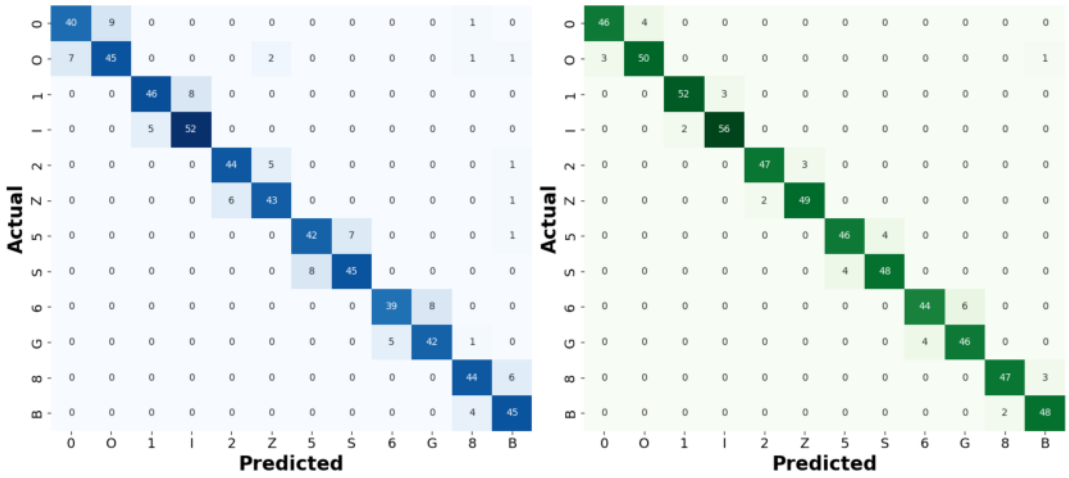

Fig. 8. Character-level confusion matrices (focused on commonly confused pairs) for ResNet-18 and EasyOCR. Left: ResNet-18, Right: EasyOCR.

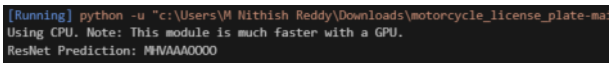

Fig. 9. Final output.

and recognizing their number plates from video footage in real-time. The system uses YOLOv8 to detect riders, helmets, and number plates, and a custom-trained CNN model based on ResNet-18 for character recognition. To increase robustness, EasyOCR is also used as a backup when custom OCR fails, ensuring better reliability under real-world conditions. The entire pipeline saves images of riders without helmets and their corresponding number plates, with the final results stored in a CSV file for easy record-keeping and further processing. The model achieved strong accuracy in both detection and character recognition tasks. The image preprocessing pipeline also effectively cleaned the plate region for better OCR results. This system can assist traffic authorities in monitoring helmet violations and can be integrated into smart city surveillance systems.

In the future, the system can be enhanced by expanding the training data to include more diverse number plate styles and helmet types. Additionally, real-time alert generation, integration with fines or traffic databases, and deployment on edge devices for live traffic monitoring are promising extensions to this work.

REFERENCES

[1] Ministry of Road Transport and Highways (2021) Road Transport Year Book 2017-18 & 2018-19. New Delhi: Government of India, Transport Research Wing. Available at: https://morth.nic.in (Accessed: 25 May 2025).

[2] Patil, M. and Majumdar, B.B., 2022. An investigation on the key determinants influencing electric two-wheeler usage in urban Indian context. Research in transportation business & management, 43, p.100693.

[3] Charran, R.S. and Dubey, R.K., 2022. Two-wheeler vehicle traffic violations detection and automated ticketing for Indian road scenario. IEEE transactions on intelligent transportation systems, 23(11), pp.22002-22007.

[4] Redmon, J. and Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[5] Reddy, R., Almeida, L., Gaitán, M.G., Santos, P.M. and Tovar, E., 2023. Synchronous Management of Mixed Traffic at Signalized Intersections Toward Sustainable Road Transportation. IEEE Access, 11, pp.64928-64940.

[6] Hussain, M., 2024. Yolov1 to v8: Unveiling each variant–a comprehensive review of yolo. IEEE Access, 12, pp.42816-42833.

[7] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[8] Tavares, R. A. (2024). Comparison of Image Preprocessing Techniques for Vehicle License Plate Recognition Using OCR: Performance and Accuracy Evaluation. arXiv preprint arXiv:2410.13622.

[9] Naveen Kumar, K., Judith, J.E., Mohammed Ashik, M., Nithin, V.N. and Sumith John, S., 2023, December. License Plate Recognition of Motorcycle Riders Without Helmet Using Deep Learning. In International Conference on Soft Computing and Pattern Recognition (pp. 167-175). Cham: Springer Nature Switzerland.

[10] Jørgensen, H., 2017. Automatic license plate recognition using deep learning techniques (Master's thesis, NTNU).

[11] Liu, Y., Jiang, B., He, H., Chen, Z. and Xu, Z., 2024. Helmet wearing detection algorithm based on improved YOLOv5. Scientific reports, 14(1), p.8768.

[12] Kaur, E.K. and Banga, V.K., 2013. Number plate recognition using OCR technique. International Journal of Research in Engineering and Technology, 2(09), p.286290.

[13] Suresh Kumar, B., Viswanadha Raju, S. and Uma Maheswari, V., 2023. 1 OpenCV libraries for computer vision. Computer Vision: Applications of Visual AI and Image Processing, 15.

[14] Ministry of Road Transport and Highways, "Road Accidents in India – 2022," Government of India. [Online]. Available: https://morth.nic.in

[15] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016. Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Springer, Part I 14, pp.21-37.

[16] Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580-587.

[17] N. Reddy, "Helmet and Number Plate Detection Dataset," unpublished, 2025.