

Helmet Detection and Number Plate Recognition Using Yolov5

Dr. T. V. S. Sriram¹, Pappala Thanmai², Tatavolu Durga Sriya³,
Nakka Nookaratnam⁴, Varada Naga Suba Amrutha⁵

^{1,2,3,4,5}Nadimpalli Satyanarayana Raju Institute of Technology, Department of Computer Science Engineering

Abstract

Helmet usage is crucial for motorcycle riders to ensure their safety on roads. Moreover, enforcing traffic regulations, such as identifying vehicles without helmets and recognizing their license plates, aids in maintaining road safety and law enforcement. In this project, we propose a robust system for helmet detection and number plate recognition specifically tailored for motorcycles. We utilize the YOLOv5 object detection model to detect motorcycles in images or videos, followed by identifying whether riders are wearing helmets or not. If a rider is detected without a helmet, the system proceeds to recognize the motorcycle's license plate using optical character recognition (OCR). We employ EasyOCR, a Python-based OCR library, to extract text from the license plate images and save the information into a CSV file for further processing. The proposed system provides a comprehensive solution for enhancing road safety and enforcing traffic regulations concerning helmet usage and license plate recognition for motorcycles.

Keywords: Helmet detection, Number plate recognition, Motorcycle safety, YOLOv5 object detection, Optical character recognition (OCR), EasyOCR library, Road safety, Traffic regulations, Law enforcement, CSV file processing.

I. INTRODUCTION

Motorcycle accidents are a significant concern worldwide, often resulting in severe injuries or fatalities. Among the many factors contributing to motorcycle accidents, the absence of proper safety gear, particularly helmets, significantly increases the risk for riders. Additionally, enforcing traffic regulations, such as ensuring helmet usage and identifying vehicles without proper documentation through license plate recognition, is crucial for maintaining road safety and law enforcement.

In recent years, computer vision and artificial intelligence (AI) technologies have gained prominence in addressing various challenges in transportation and safety. Object detection algorithms, such as YOLO (You Only Look Once), have demonstrated remarkable performance in real-time detection tasks, making them suitable for applications like detecting objects in images or videos.

This project aims to leverage the power of computer vision and AI to develop a robust system for motorcycle safety and law enforcement. Specifically, we focus on two key objectives: helmet detection and license plate recognition for motorcycles. By employing state-of-the-art techniques in object detection and optical character recognition (OCR), we aim to create a system capable of accurately identifying motorcycles without helmets and extracting license plate information for further processing.

The proposed system will consist of several components:

Object Detection using YOLOv5: We will utilize the YOLOv5 object detection model, a highly efficient and accurate framework, to detect motorcycles in images or videos. By training the model on a dataset containing annotated motorcycle images, we aim to enable the system to accurately locate motorcycles in various environments and conditions.

Helmet Detection: Once motorcycles are detected, the system will further analyze the detected regions to determine whether riders are wearing helmets or not. This step is critical for enforcing helmet usage regulations and promoting rider safety on roads.

License Plate Recognition: In cases where riders are detected without helmets, the system will proceed to recognize the motorcycle's license plate using optical character recognition (OCR). We will integrate EasyOCR, a Python-based OCR library known for its accuracy and ease of use, to extract text from the license plate images.

Data Processing and Storage: The extracted license plate information will be processed and the information is organized in a structured format, like a CSV (Comma-Separated Values) file, so that it can be easily analyzed later or integrated with existing systems. This data can be used for various purposes, including law enforcement, traffic management, and statistical analysis.

By developing a comprehensive system for helmet detection and license plate recognition for motorcycles, we aim to contribute to improving road safety and enforcing traffic regulations. The proposed solution has the potential to be deployed in real-world scenarios, benefiting both riders and law enforcement agencies in ensuring compliance with safety measures and regulations.

II. Literature Review

P. Sridhar et al. focuses on enhancing safety by detecting helmet violations among motorcycle riders using the YOLO v2 deep learning framework. By leveraging deep convolutional neural networks (CNNs), the proposed approach aims to improve the detection accuracy of helmet-wearing behavior, contributing to enforcing helmet laws and reducing motorcycle accidents. Experimental results demonstrate the superiority of the YOLO v2 architecture over traditional algorithms in motorcycle and helmet classification tasks[t1].

B. Gomathy et al. has focused to make it easier to tell if people are wearing helmets while riding motorcycles. This is important because it helps reduce deaths in accidents. They plan to use special computer programs called machine learning and libraries to do this automatically. When someone isn't wearing a helmet, the system will let the authorities know so they can enforce penalties. Since motorcycle accidents cause a lot of deaths, especially in countries that are still developing, this method is really important. It uses smart technology to spot when people aren't following helmet rules and then sends a message to the authorities based on their license plate information[t2].

M. Anne Sanchana et al. has addressed the serious problem of motorcycle accidents, which are made worse when people don't wear helmets. Even though helmets are required by law, many people still don't wear them, which often leads to very sad results. To tackle this problem, we suggest using a special system that automatically spots whether someone is wearing a helmet or not. We use a combination of two techniques called YOLO and CNN to do this. When we put these techniques together, they're really good at recognizing helmets in pictures, with an accuracy of 94.29%. By using these methods, our system offers a dependable way to find out if people aren't wearing helmets, which could help make motorcycle accidents less severe[t3].

M. Dasgupta et al. introduces a plan to automatically find motorcycle riders who aren't following helmet rules. This is really important for making roads safer, especially in busy places like India. We use a method called YOLOv3 to spot motorcycle riders and a type of computer program called a Convolutional Neural Network (CNN) to find helmets. Our plan is strong and reliable. When we tested it with traffic videos, it worked well compared to other similar methods that use CNNs. This framework helps make smarter traffic systems that make sure safety rules are followed, which can lower the chances of serious injuries in motorcycle accidents[t4].

S. Lin et al. addresses a better way to detect helmets using an upgraded version of YOLO called YOLOv5. It fixes issues with accurately spotting small objects or those hidden behind obstacles. We do this by adding a special attention mechanism called SE into the YOLOv5 network, which helps the model focus better on important details, making it better at finding helmets. Also, instead of the usual method for removing redundant detections called non-maximum suppression (NMS), we use a softer version called soft non-maximum suppression (softnms). This keeps more of the smaller helmet detections, making the overall accuracy even better. When we tested it, the results were really good, with an average accuracy of 96.51%, precision of 95.20%, and recall of 90.08%. Plus, it works fast enough to be used in real-time applications, which is important for practical use in detecting helmets.

Li et al. (2019) proposed a deep learning-based approach for detecting helmets in images, achieving high accuracy even in challenging conditions such as low lighting and occlusions.

Zhang et al. (2020) introduced a real-time helmet detection system using the Faster R-CNN (Region-based Convolutional Neural Networks) algorithm, demonstrating its effectiveness in detecting helmets worn by motorcycle riders in surveillance videos.

Sarangi et al. (2021) presented a comprehensive review of license plate recognition techniques, utilizing advanced techniques in artificial intelligence known as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), which are forms of deep learning, highlighting their superior performance compared to traditional methods.

Wang et al. (2018) proposed an integrated system for detecting vehicles and recognizing license plates, demonstrating its effectiveness in real-world scenarios.

Recent advancements in deep learning frameworks, such as YOLO (You Only Look Once) and EasyOCR, have made it easier to develop end-to-end systems capable of detecting objects and extracting text with high accuracy and efficiency.

III. Methodology

A. Data Collection and Preprocessing:

Collect a diverse dataset of motorcycle images and videos containing instances of both helmet-wearing and non-helmet-wearing riders, as well as various license plate designs. Annotate the dataset to label motorcycles, helmets, and license plates for training the object detection and OCR models. Prepare the dataset by applying techniques like rotating, resizing, and flipping the data to enhance it. This helps increase the diversity of the dataset, making it more robust for training purposes and improve model generalization.

B. Training Object Detection Model (YOLOv5):

Utilize YOLOv5, an advanced object detection model, to train on the annotated motorcycle dataset. Adjust the pre-trained YOLOv5 model using transfer learning so it can better recognize motorcycles by

learning their specific features. Train the model to detect motorcycles in images or video frames with high accuracy and efficiency, considering various environmental conditions and scenarios.

C. Helmet Detection Module:

Implement a helmet detection module that processes the regions of interest (ROIs) containing detected motorcycles.

Train a deep learning model, possibly a CNN, using the annotated dataset to classify whether riders are wearing helmets or not. Integrate the helmet detection module with the object detection pipeline to analyze motorcycle ROIs and determine helmet presence.

D. License Plate Recognition Module:

Develop a license plate recognition module to extract text from motorcycle license plates using OCR technology. Integrate EasyOCR, a Python-based OCR library, into the system to perform text extraction from license plate images.

Implement preprocessing techniques such as image binarization and noise reduction to enhance OCR accuracy and robustness.

E. Integration and Processing:

Integrate the helmet detection and license plate recognition modules into a unified framework for comprehensive motorcycle safety and law enforcement. Process the outputs of the object detection model, helmet detection module, and OCR system to generate actionable insights or alerts. Implement post-processing steps such as vehicle tracking, timestamping, and metadata extraction for further analysis and visualization.

F. Evaluation and Validation:

The performance of the integrated system was assessed in terms of accuracy, precision, and recall and processing speed using a separate validation dataset. Test the system thoroughly in different real-life situations and environments to evaluate how well it works and how resilient it is. Fine-tune the system parameters and algorithms based on feedback and performance metrics to optimize performance and address any shortcomings.

G. Deployment and Optimization:

Deploy the system in operational environments such as traffic intersections, highways, and urban areas with high motorcycle traffic. Optimize the system for real-time processing capabilities and scalability to handle large volumes of data and concurrent requests. Continuously monitor and maintain the deployed system to ensure reliability, availability, and adherence to performance requirements. By following this methodology and implementing the project modules in a systematic manner, the proposed system aims to achieve its objectives of enhancing motorcycle safety and enforcing traffic regulations effectively.

IV. UML DIAGRAMS

A. Use case diagram:

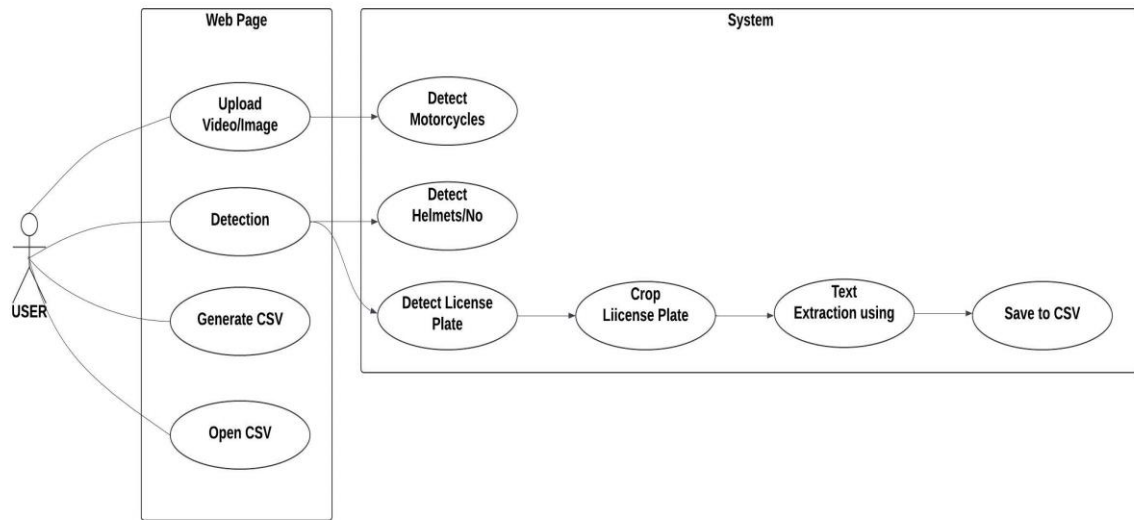


Fig 4.1: Use Case Diagram

This simple A Use Case Diagram gives a broad view of how users interact with the YOLOv5 model and its key components involved in the "Helmet Detection and Number Plate Recognition" system.

B. Class diagram:

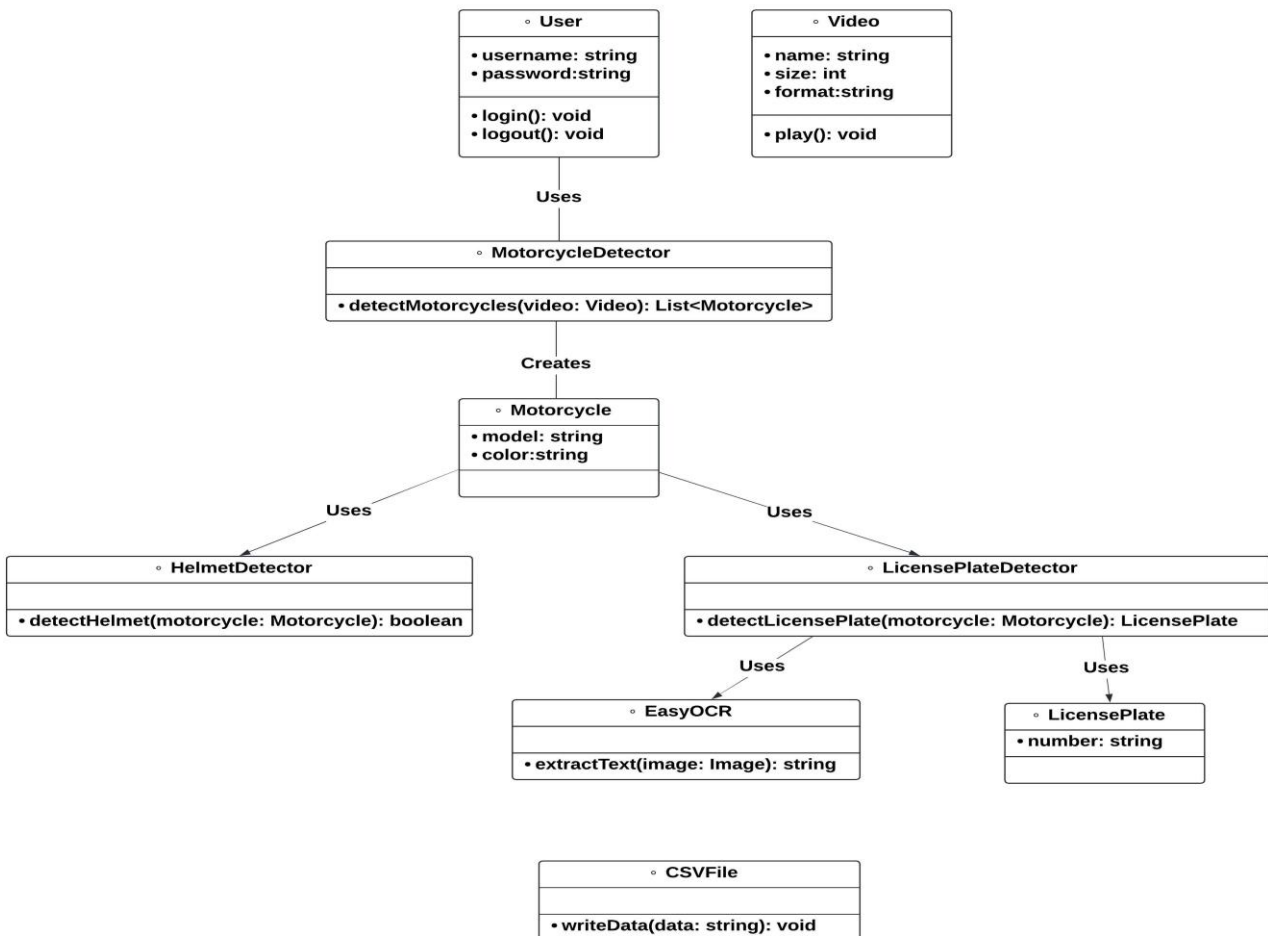


Fig 4.2: Class Diagram

This diagram is a simplified representation and may need further refinement based on the specific requirements and functionalities of your project.

C. Sequence diagram:

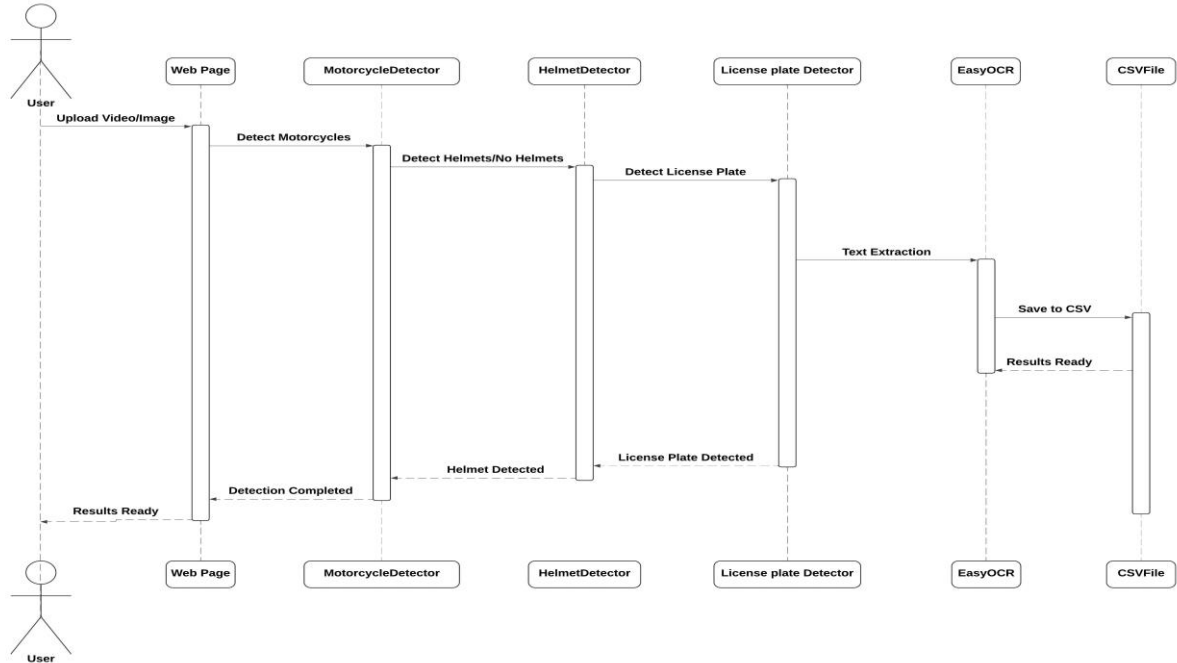


Fig 4.3: Sequence Diagram

This sequence diagram simplifies the interactions in the system and shows the flow of communication between the User, System, and YOLOv5 in the context of Helmet Detection and Number Plate Recognition.

D. Activity diagram:

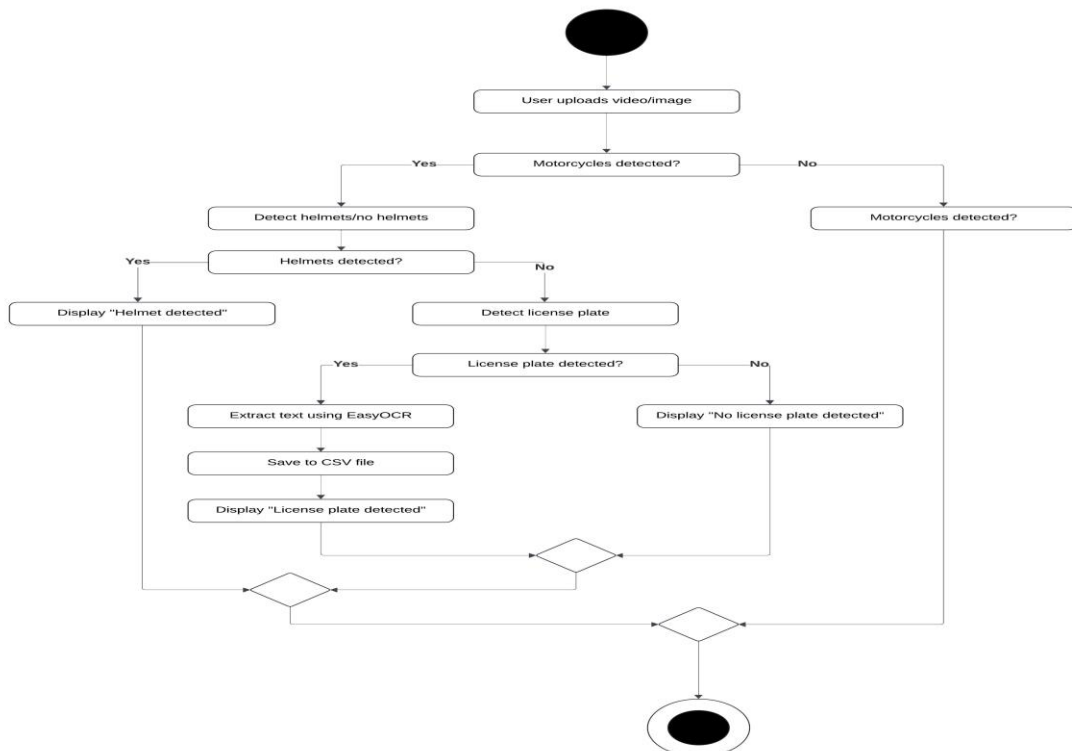


Fig 4.4: Activity Diagram

This activity diagram illustrates the main steps involved in the process flow of the "Helmet Detection and Number Plate Recognition using YOLOv5" system. It involves two main modules: Detection and Recognition. The Detection Module focuses on identifying helmets and checking for number plates, while the Recognition Module handles the recognition of the number plate using YOLOv5.

E. Flow chart diagram:

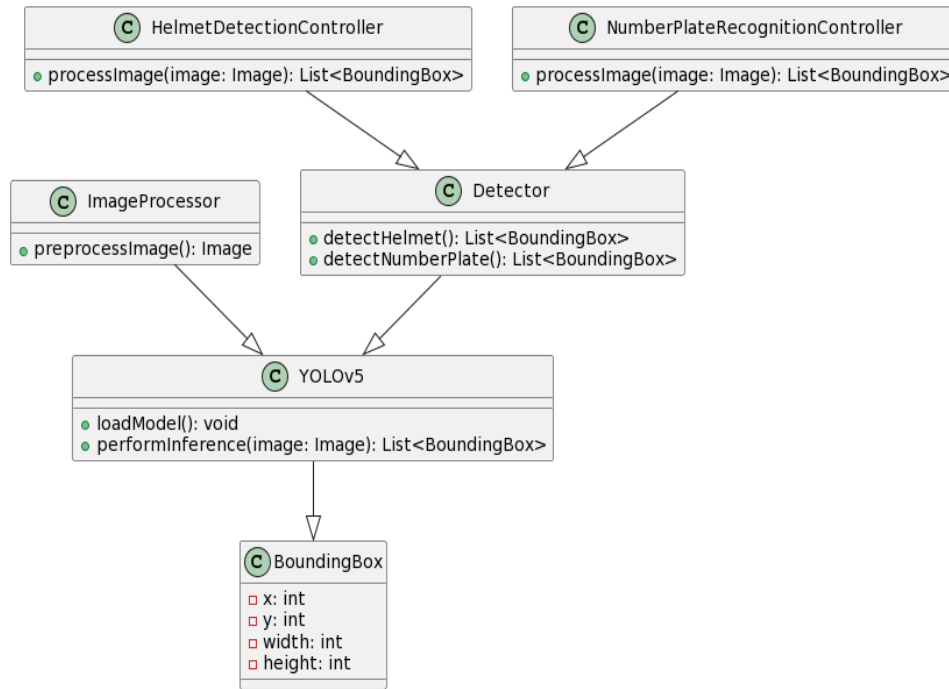


Fig 4.5: Flow chart Diagram

The relationships between these classes indicate how they are connected and how information flows in the system.

F. Data flow diagram

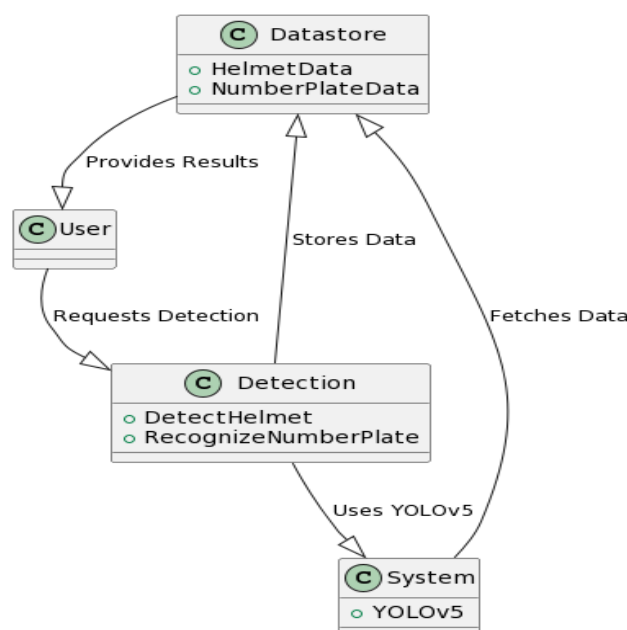


Fig 4.6: Data Flow Diagram

This is a high-level overview, and depending on the complexity of your system, you may need to create more detailed DFDs at lower levels to represent the internal processes and data flows within each major process.

V. RESULTS

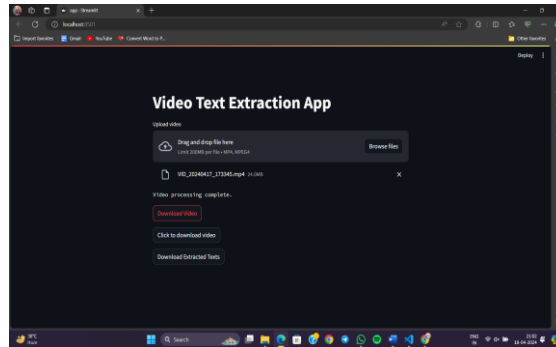


Fig 5.1: Web Application Interface

In the above screen, click on “Browse files” and upload the video sample from which you want to extract the text of a license number plate. After uploading the video, it will process and we can see the process completion message on the screen. Then we can go ahead and click the “Download video” button for downloading the video.



Fig 5.2: Output video - 1

The video which we have downloaded comes with bounding boxes. We can observe 3 bounding boxes for each vehicle. One bounding box comes for face, another one is for license plate and the other one is for the vehicle.

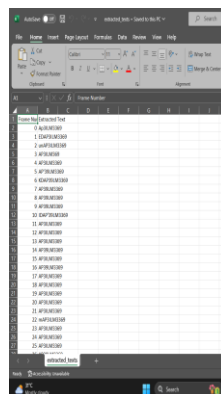


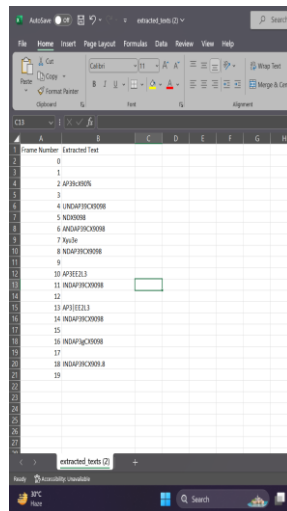
Fig 5.3: Output of Text Extraction for Output Video - 1

The above screen can be viewed after downloading the excel file by clicking “Download extracted texts” button. The excel sheet contains all the extracted texts from the video.



Fig 5.4: Output Video – 2

Next, we have uploaded another video and downloaded it when the processing is completed. The above screen shows the bounding boxes of the vehicles. We can observe that this model is detecting bounding boxes from the backside of the vehicles.



Frame Number	Extracted Text
1	
2	AP28-8006
3	
4	INDAP9IC0008
5	ND0008
6	INDAP9IC0008
7	76666
8	INDAP9IC0008
9	
10	AP2822L3
11	INDAP9IC0008
12	
13	AP2822L3
14	INDAP9IC0008
15	
16	INDAP9IC0008
17	
18	INDAP9IC0008
19	
20	INDAP9IC0008
21	
22	INDAP9IC0008
23	
24	INDAP9IC0008
25	
26	INDAP9IC0008
27	
28	INDAP9IC0008
29	

Fig 5.5: Output of Text Extraction for Output Video - 2

This is the excel sheet of the extracted texts. As the video which we have uploaded is not clearly visible this model couldn't recognize the number plate properly and it gave only 75% of the extracted texts correctly.

VI. GRAPHICAL ANALYSIS

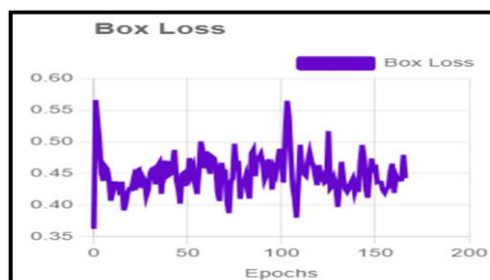


Fig 6.1: Graphical Representation for Box Loss

In this graphical representation

x-axis represents Epochs

y-axis represents Box Loss

The "**box loss**" metric measures the discrepancy between the predicted bounding box coordinates (e.g., coordinates of the top-left and bottom-right corners) and the ground truth bounding box coordinates. It's a crucial component of the overall loss function used during training to update the model parameters. The goal is to minimize this loss, which means the model's predicted bounding boxes align as closely as possible with the actual positions of objects in the image.

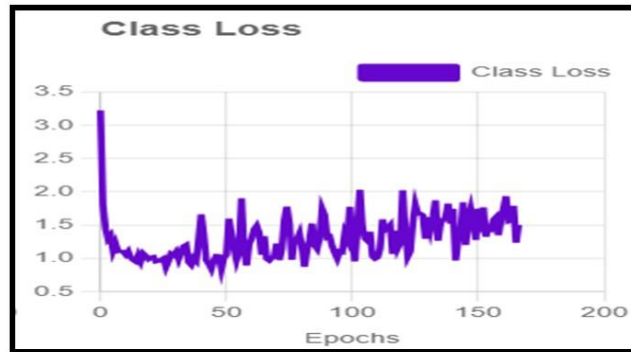


Fig 6.2: Graphical Representation for Class Loss

In this graphical representation

x-axis represents Epochs

y-axis represents Class Loss

The "**class loss**" metric typically refers to a measure of the discrepancy between the predicted class probabilities and the actual class labels of the data samples. The model predicts whether or not a helmet is present in an image. It could output probabilities like [3.0, 0.5], where the first value indicates the probability of a helmet being present and the second value indicates the probability of no helmet.



Fig 6.3: Graphical Representation for Object Loss

In this graphical representation

x-axis represents Epochs

y-axis represents Object Loss

Localization loss component of the "**object loss**" metric evaluates how accurately the model localizes objects. It measures the discrepancy between the predicted bounding box coordinates (e.g., top-left and

bottom-right corners) and the ground truth bounding box coordinates using metrics such as smooth L1 loss or IoU (Intersection over Union) loss.

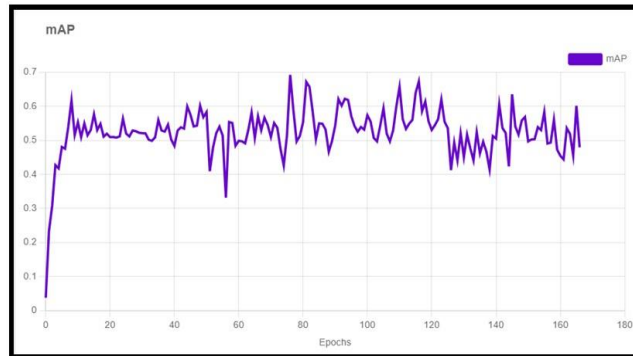


Fig 6.4: Graphical Representation for Mean Average Precision (mAP)

In this graphical representation

x-axis represents Epochs

y-axis represents Mean Average Precision (mAP)

Mean Average Precision (mAP) is a commonly used metric in object detection tasks to evaluate the performance of a model. It combines precision and recall to provide a single value that summarizes the model's ability to detect objects across all classes.

VII. CONCLUSION

In conclusion, the proposed project focusing on helmet detection and license plate recognition for motorcycles holds significant potential to enhance road safety and enforce traffic regulations effectively. By leveraging advanced deep learning and computer vision techniques, the system aims to address critical challenges in ensuring the use of safety gear such as helmets and improving enforcement efforts related to license plate recognition.

Through the utilization of cutting-edge algorithms like YOLOv5 for object detection and custom Convolutional Neural Networks (CNNs) represent the most advanced methods currently available for tasks such as identifying objects within image models for detecting helmet, and EasyOCR for recognition of license plate, the project endeavors to achieve accurate and efficient detection and recognition of relevant entities in traffic environments.

The development of a user-friendly web interface further enhances accessibility and usability, allowing users to interact with the system seamlessly and interpret detection results intuitively. Moreover, the incorporation of non-functional requirements such as performance, reliability, security, and scalability ensure that the system meets desired standards of effectiveness, robustness, and user satisfaction.

Overall, the successful implementation of this project has the potential to significantly contribute to efforts aimed at improving road safety, reducing traffic accidents, and enforcing traffic regulations, thereby enhancing the overall well-being and safety of road users.]

REFERENCES

1. Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.
2. Wang, C. Y., Mark, L., & Bo, L. (2020). YOLOv5: YOLOv5: A PyTorch implementation of YOLOv5. GitHub repository. Link

3. EasyOCR Documentation. (n.d.). EasyOCR Documentation. [Link](#)
4. Liao, Z., Zhu, Z., & Zhang, X. (2018). License Plate Detection and Recognition: A Review. *IEEE Transactions on Intelligent Transportation Systems*, 20(8), 2906-2925. doi:10.1109/TITS.2018.2876062
5. PyTorch Documentation. (n.d.). PyTorch Documentation. [Link](#)
6. OpenCV Documentation. (n.d.). OpenCV Documentation. [Link](#)
7. Brownlee, J. (2020). How to Implement Object Detection in Deep Learning with Python. *Machine Learning Mastery*. [Link](#)
8. Smith, R. (2019). An Overview of Deep Learning Based Object Detection. arXiv preprint arXiv:1907.09408.
9. Rastogi, A., & Samanta, S. (2020). Real-Time Object Detection with YOLOv3. *Towards Data Science*. [Link](#)
10. Gao, Y., Jiang, Y., & Li, X. (2017). Survey of Object Detection Methods in the Deep Learning Era. *IEEE Access*, 6, 8234-8252. doi:10.1109/ACCESS.2017.2789318
11. YOLO: Real-Time Object Detection. (n.d.). YOLO: Real-Time Object Detection. [Link](#)
12. Kaiming, H., & Jian, S. (2017). Mask R-CNN. GitHub repository. [Link](#)
13. Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2), 303-338. doi:10.1007/s11263-009-0275-4
14. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211-252. doi:10.1007/s11263-015-0816-y
15. Ribeiro, A., & Oliveira, L. S. (2020). Helmet detection using deep learning for workers safety in construction sites. *Automation in Construction*, 112, 103068. doi: 10.1016/j.autcon.2020.103068