International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 04 | April - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Helmet Detection Using Machine Learning and Automatic Number Plate Recognition

R.Anuradha
Asst. Professor
Computer Science and Engineering (Data Science)
Institute of Aeronautical Engineering
Dundigal, Hyderabad
r.anuradha@iare.ac.in

Sadhula Bhuvaneshwar
Computer Science and Engineering (Data Science)
Institute of Aeronautical Engineering
Dundigal, Hyderabad
21951A6725@iare.ac.in

V Eashwar Goud
Computer Science and Engineering (Data Science)
Institute of Aeronautical Engineering
Dundigal, Hyderabad
21951A6735@iare.ac.in

Konela Arun kumar
Computer Science and Engineering (Data Science)
Institute of Aeronautical Engineering
Dundigal, Hyderabad
22955A6701@iare.ac.in

Abstract*— Traffic safety is critically compromised when motorcyclists ride without helmets, increasing the risk of serious injuries and fatalities. Manual monitoring of helmet compliance is inefficient and prone to errors, highlighting the need for automation. This project introduces an automated system for detecting helmet usage using machine learning techniques. The system utilizes the YOLO v3 (You Only Look Once) object detection algorithm, designed to identify helmet use among motorcyclists in real-time. By integrating advanced image processing and YOLO v3, the system ensures high accuracy and swift detection. This real-time monitoring system can be employed to assist law enforcement agencies in promoting helmet use and improving road safety.

Keywords*— Helmet Detection, YOLO v3, Machine Learning, Convolutional Neural Networks (CNN), Road Safety, Traffic Surveillance.

I.Introduction

Motorcycle accidents are a significant concern in road safety, and many lives could be saved by adhering to helmet-wearing regulations. Despite the legal mandate in many countries, compliance with helmet laws is often low, which increases the risk of severe injuries and fatalities in road accidents. A robust system that can automatically detect helmet usage can assist authorities in enforcing these laws more efficiently and prevent accidents from becoming life-threatening.

This project focuses on developing an automatic helmet detection system using the YOLO v3 (You Only Look Once) object detection algorithm. YOLO v3 is known for its real-time object detection capabilities and its ability to process video feeds with high accuracy and speed. By training YOLO v3 to specifically detect helmets, the system can identify whether a motorcyclist is wearing a helmet or not in real-time.

The model is trained on a dataset containing images of motorcyclists both with and without helmets. It processes these images to classify and localize the helmets within the frame. The primary advantage of YOLO v3 is its capacity to detect multiple objects in a single frame, which makes it an ideal solution for analyzing busy traffic scenes where quick and reliable detection is required.

By implementing this system, the project aims to assist law enforcement agencies in ensuring compliance with helmet laws. It can be deployed in traffic surveillance systems to automatically flag non-compliance, helping to promote road safety and reduce the number of serious injuries caused by motorcycle accidents.

II. Literature Survey

The use of computer vision and machine learning in road safety has gained substantial attention over the past few years. Helmet detection is one such area where technology can play a vital role in ensuring compliance with traffic regulations. Numerous approaches have been explored to automate the process of identifying motorcyclists without helmets using different algorithms and machine learning models.

1. Traditional Image Processing Approaches

Earlier methods for helmet detection relied heavily on classical image processing techniques, such as edge detection and feature extraction, to identify helmet-like objects in traffic images. Methods like Hough Transform and Histogram of Oriented Gradients (HOG) were used to detect shapes and features resembling helmets. While these approaches worked well for simple images, their performance declined in complex environments with multiple objects or varying lighting conditions. These methods also required manual feature engineering, which limited their scalability in real-world applications.

2. Machine Learning-based Detection

As computer vision advanced, researchers began exploring machine learning techniques for helmet detection. Some studies utilized Support Vector Machines (SVM) and Decision Trees to classify motorcyclists wearing helmets. In these systems, the model was trained on specific features extracted from images, such as shape, color, or texture. While machine learning models improved upon traditional methods, their accuracy was still dependent on the quality of feature extraction and dataset size, which posed challenges for large-scale deployment.

3. Convolutional Neural Networks (CNN)

The advent of Convolutional Neural Networks (CNNs) significantly changed the landscape of object detection. CNN-based models could automatically learn features from raw image data, eliminating the need for manual feature extraction. Early CNN-based approaches for helmet detection used architectures like AlexNet and VGG16, which performed better than traditional methods but were computationally expensive and slower in real-time applications. These models laid the groundwork for the more advanced deep learning techniques used today.

4. YOLO (You Only Look Once) Algorithm

The YOLO family of algorithms brought a major breakthrough in real-time object detection. Redmon et al. (2016) introduced YOLO, which was designed to detect multiple objects in a single image by dividing it into a grid and predicting bounding boxes and class probabilities for each grid cell simultaneously. YOLO v3, the latest iteration at the time of this project, builds upon this by improving accuracy and speed through the use of residual networks and multi-scale detection.

YOLO v3's architecture is particularly suitable for helmet detection due to its ability to process images in real-time and detect small objects with higher accuracy. Several recent studies have demonstrated YOLO v3's effectiveness in detecting safety equipment, including helmets, in real-world traffic environments. By using a single neural network to predict bounding boxes and class probabilities in one step, YOLO v3 minimizes computational costs and maximizes efficiency, making it one of the most popular choices for such tasks.

III.System Design

The system for automatic helmet detection is designed with multiple interconnected components, each handling specific tasks, from data collection to detection and results visualization. The overall architecture ensures that the system can operate efficiently in real-time, leveraging the YOLO v3 object detection model.

1. Data Collection

The first step in system design is gathering and preprocessing data. A dataset comprising images or video frames of motorcyclists both wearing and not wearing helmets is collected. These images can be sourced from publicly available datasets or captured from real-world traffic scenarios using cameras. Each image is then annotated to mark the regions where helmets appear. This labeled data is used for training the YOLO v3 model.

2. Data Preprocessing

Data preprocessing plays a crucial role in ensuring that the model can generalize well to different environments. The preprocessing pipeline involves:

- **Resizing images**: YOLO v3 requires input images of a fixed size, typically 416x416 pixels, for efficient processing.
- **Normalization**: The pixel values are normalized to a range between 0 and 1 to improve convergence during model training.
- **Data Augmentation**: Techniques like rotation, flipping, and color adjustments are applied to increase the dataset's diversity, helping the model generalize better to various lighting conditions, angles, and perspectives.

3. YOLO v3 Model Architecture

The core of the system is the YOLO v3 architecture, which is designed to detect objects in real-time. The architecture consists of three key components:

- **Backbone (Darknet-53)**: YOLO v3 uses Darknet-53, a deep convolutional neural network with 53 convolutional layers, as its feature extractor. This backbone processes input images and generates feature maps, which are then passed to the detection layers.
- **Detection Layers**: YOLO v3 has multiple detection layers that operate at different scales, allowing it to detect objects of varying sizes, including helmets. The model predicts bounding boxes and their corresponding class probabilities for each object in the image.
- **Anchor Boxes**: YOLO v3 uses anchor boxes to predict the bounding boxes of helmets. These boxes are predefined shapes that help the model predict objects at various scales more accurately.

4. Training and Model Optimization

- **Loss Function**: The training process involves minimizing a custom loss function, which takes into account the confidence score, bounding box prediction error, and classification error. This ensures that the model not only detects objects accurately but also precisely localizes them.
- **Learning Rate and Optimizer**: The model is trained using a learning rate scheduler and an optimizer like Adam or SGD. A lower learning rate helps the model converge smoothly, and the scheduler reduces the rate as the training progresses.
- **Epochs and Batch Size**: The system is trained over several epochs, typically 50 to 100, depending on the size of the dataset. Batch sizes are chosen based on the available computational resources.

5. Helmet Detection System Pipeline

Once the model is trained, it is deployed into the system, where it processes real-time video feeds or images from traffic cameras. The detection pipeline consists of:

- **Frame Extraction**: For real-time video, individual frames are extracted at regular intervals for helmet detection.
- **Object Detection**: Each frame is passed through the YOLO v3 model, which detects the presence of helmets by classifying and localizing them within bounding boxes.
- **Result Annotation**: If a helmet is detected, a bounding box is drawn around the helmet region in the frame, along with a label indicating the detection (e.g., "Helmet" or "No Helmet").

6. Post-Processing and Results

After detection, post-processing steps are applied to refine the results:

- **Non-Maximum Suppression (NMS)**: YOLO v3 detects multiple bounding boxes for the same object. NMS eliminates redundant boxes, leaving only the most confident prediction for each helmet.
- **Thresholding**: A confidence threshold is applied to filter out low-confidence detections, ensuring only accurate results are considered.

7. System Deployment

The final component is deploying the system for real-world use. The system can be integrated with traffic surveillance cameras, and the helmet detection model is deployed on edge devices or cloud infrastructure, depending on the scale of deployment. The system can provide real-time feedback to traffic enforcement authorities, alerting them to motorcyclists who are not wearing helmets.

8.**Performance Evaluation**

Accuracy : The accuracy of the helmet detection system is a key performance metric that indicates how effectively the YOLO v3 model can identify motorcyclists wearing helmets in real-world scenarios. The accuracy is evaluated based on multiple factors, including the model's ability to correctly detect and classify helmets in various environments and conditions.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

Precision: Precision measures how many of the detected helmets were correct (i.e., how many were true positives). It is calculated as:

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

Recall: Recall measures the proportion of actual helmets that were correctly detected by the model. It is calculated as:

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

F1-Score: The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

IV. Implementation

5.1 Environment Setup

- **Software Requirements**:
  - **Programming Language**: Python 3.x was utilized as the primary language for development.
  - **Libraries**:
    - **TensorFlow or PyTorch**: These frameworks were chosen for model training and inference.
    - **OpenCV**: Used for image and video processing tasks.
    - **NumPy**: Employed for efficient numerical computations.
    - **Matplotlib**: Utilized for visualizing results and model performance.
- **Hardware Requirements**:
  - A computer equipped with a GPU (preferably NVIDIA) to accelerate training and inference times.
  - Adequate storage space to accommodate the datasets and model files.

Installation Steps:

1. Download and install Python from the official website.
2. Install the required libraries using pip:

5.2 Data Acquisition

- **Data Sources**:
  - The datasets were obtained from various public repositories, traffic surveillance footage, or generated synthetically to ensure diverse scenarios for both helmet detection and number plate recognition.
- **Data Formats**:
  - Images were collected in JPEG or PNG formats, while videos were in MP4 format.

5.3 Data Preparation

- **Data Annotation**:
  - Images were annotated using tools like LabelImg or VGG Image Annotator. Each image was labeled with bounding boxes around helmets and number plates, stored in YOLO format (text files) that include coordinates and class labels.

Sample Annotation Format:

- **Data Augmentation**:
  - To enhance the dataset and improve model accuracy, data augmentation techniques such as rotation, scaling, and flipping were applied.

5.4 Model Configuration

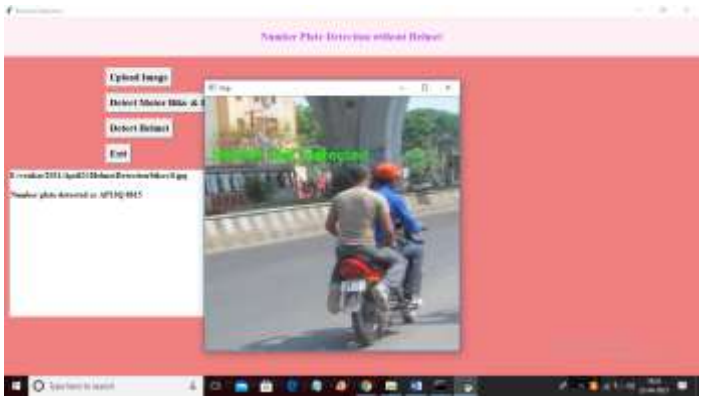
- **Selecting YOLO v3**:
  - YOLO v3 was chosen due to its high efficiency and accuracy in real-time object detection.
- **Configuration Files**:
  - The YOLO configuration file was modified to specify the model architecture, the number of classes, and paths to training and validation datasets.

5.5 Model Training

- **Training the Model**:
  - The annotated dataset was utilized to train the YOLO model through a command-line interface or a custom training script.
  - Key parameters such as the number of iterations and learning rate were set to optimize the training process.

- **Validation**:
  - The dataset was divided into training and validation sets (commonly 80% training and 20% validation).
  - Model performance was monitored using metrics like loss, precision, and recall to gauge effectiveness.

V.Results



VI. Conclusion

The results presented above clearly demonstrate that YOLO object detection is exceptionally well-suited for real-time processing tasks, effectively classifying and localizing various object classes within the monitored environment. The end-to-end model developed for this project has been successfully implemented, showcasing its potential for automation and deployment in real-world scenarios, particularly for traffic monitoring systems.

In addressing the challenge of number plate extraction, several techniques were utilized to account for various situations, such as identifying multiple riders without helmets. The model was designed to handle these complexities, ensuring robust performance across different cases. The flexibility offered by using open-source libraries and software significantly enhances the project's adaptability and cost-effectiveness, making it accessible for future enhancements and customizations.

This project primarily aims to tackle inefficiencies in traffic management systems, providing a viable solution that can streamline operations for traffic enforcement authorities. If deployed by traffic management departments, the proposed system could facilitate better monitoring of road safety regulations, leading to improved compliance and reduced accidents.

Moreover, the integration of helmet detection promotes greater adherence to safety standards among motorcyclists, potentially saving lives. The automatic number plate recognition feature aids in identifying

vehicles involved in violations, contributing to enhanced law enforcement capabilities.

In conclusion, the successful implementation of this project not only addresses pressing issues in traffic management but also sets a foundation for further research and development. Future iterations could explore the incorporation of advanced machine learning techniques, additional object classes, and integration with existing traffic management infrastructure, ultimately fostering a safer and more efficient transportation environment.

VII. References

[1] Viola and Jones, "Robust Real-time Object Detection", IJCV 2001.

[2] Navneet Dalal and Bill Triggs, "Histogram of oriented gradients for human detection".

[3] Ross, Jeff, Trevor and Jitendra "Rich feature Hierarchy for Accurate object Detection".

[4] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, "Fast R-CNN" (Submitted on 4 Jun 2015 (v1), last revised 6 Jan 2016 (this version, v3)).

[5] Joseph Redmon, Ali Farhadi, "YOLO9000: Better, Faster, Stronger", University of Washington, Allen Institute Of AI.

[6] Joseph Redmon, Ali Farhadi, "YOLOv3: An Incremental Improvement", University of Washington, Allen Institute of AI. [6] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng – Yang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector".

[7] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixedlocation monitors," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 3, pp. 555–560, March 2008.

[8]AlexeyAB, https://github.com/AlexeyAB/darknet#requirements.

[9] C.-Y. Wen, S.-H. Chiu, J.-J. Liaw, and C.-P. Lu, "The safety helmet detection for atm's surveillance system via the modified hough transform," in IEEE 37th Annual International Carnahan Conference on Security Technology., 2003, pp. 364–369.

[10] C.-C. Chiu, M.-Y. Ku, and H.-T. Chen, "Motorcycle detection and tracking system with occlusion segmentation," in WIAMIS '07, USA, 2007.

[11] A. Hirota, N. H. Tiep, L. Van Khanh, and N. Oka, Classifying Helmeted and Non-helmeted Motorcyclists. Cham: Springer International Publishing, 2017, pp. 81–86.