YOLO V5: A Novel Approach in Ensuring Safety among Road Users

[1]Sachidananda Tripathy
Computer Science & Information Technology, Siksha 'O' Anusandhan University, Bhubaneswar, India
sachidanandatripathy1234@gmail.com

[4]*Sushree Bibhuprada B. Priyadarshini
Computer Science & Information Technology
Siksha 'O' Anusandhan University, Bhubaneshwar, India
bimalabibhuprada@gmail.com

[2]Malaya Kumar Swain
Computer Science & Information Technology, Siksha 'O' Anusandhan University , Bhubaneshwar, India
swainmalaya07@gmail.com

[5]Smita Rath
Computer Science & Information Technology
Siksha 'O' Anusandhan University, Bhubaneswar, India
smitarath@soa.ac.in

[3] Kunal Kumar
Computer Science & Information Technology, Siksha 'O' Anusandhan University , Bhubaneshwar , India
kunalkumar54345@gmail.com

2024 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC) | 979-8-3503-7018-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ASSIC60049.2024.10508023

***Abstract*—** An important subject in the domain of computer vision frameworks is accurate and effective tracking of concerned objects. In this connection, YOLO v5 stands for "You Only Look Once Version 5," which is the tool that we have employed to identify objects that move in both images and videos in real-time while keeping a constantly low latency. In current research, our objective is to conduct studies to investigate whether users wear helmets at different plots by utilizing YOLO v5 for identifying moving objects. We used YOLO V5s and 200 epochs of Python in Google Colab to conduct extensive experimentation. We found that with the quantity of epochs increased, so did the accuracy values, thereby indicating the effectiveness of the suggested strategy.

Keywords- CNN, Deep learning, Object detection, Vision algorithm.

I. INTRODUCTION

The human population of the country has grown quickly over time, which has led to a rapid increase in the overall number of automobiles. India's population was projected to reach 1.4 billion in 2022. When going short distances, bikes are the preferred mode of transportation for most people in India. The incredibly congested road network and large user base make it very challenging to execute road safety laws and regulations in practice. Because of this, system automation and optimization will be crucial to maintaining traffic safety. Helmet have been shown to cut mortality by 37% among riders and 67% among passengers, according to the most recent data from the CDC USA. For this reason, we decide to make it easier to find the helmets and gather information on the license plate of the passenger car. In the subject of road safety technological advances, there aren't many methods for ensuring that traffic laws are followed.

Nowadays, computers are used to identify penalties like speeding and running red lights. However, these systems still use outdated computer vision technology. In this case, we want to use the much more recent YOLO v5 to speed up the phenomenon and produce improved outcomes in terms of accuracy and efficacy. The most important piece of security gear for motorcycle riders is the helmet, which guards against severe head injuries and death in the event of an accident. Due to the country's lax enforcement of traffic regulations and general lack of knowledge about road safety and the value of wearing a helmet, the majority of riders do not wear them [1-4].

A. Motivation

Even though most countries require riders to wear helmets, many motorcyclists choose not to wear them and risk losing their jobs in the process. Recent advancements in traffic monitoring technology have made it possible to identify and recognize vehicles as well as detect helmets. Computer vision strategies, such as foreground and background image tracking is used to distinguish mobile objects in scene elements and the image identifiers to gather data, have been used in the development of traffic signal systems.

The necessary techniques, like the ML algorithm, are applied to classify the elements. In machine learning (ML), a predictive model operates autonomously by using inputs provided during the training process. Sample data, or "training data," is used to build mathematical formulas and models with the aid of machine learning algorithms and technologies. Thus, by training with a suitable data set, a helmet detection model could be created [5-8].

B. Major Contribution

YOLOv5's exceptional acuity in recognizing small objects is another important feature. The Feature Pyramid Networks (FPNs) and anchor boxes, which enable the proposition to recognize objects at various forms as well as

angles, are used to achieve this. Additionally, Non Maximum Suppression (NMS) and Weighted Box Fusion (WBF), two post-processing techniques that help reduce false positive results and raise object identification accuracy, are greatly improved by YOLOv5.

The remainder sections of the document are arranged as follows: The related literature produced in this field is described in detail in section II. Section III goes over our suggested course of action. The results obtained from the experiment are covered in the subsequent section, and section V concludes with an analysis of the paper.

II. LITERATURE SURVEY

The helmet monitoring model has emerged as a highly effective approach to evaluating safety, particularly in industries with a high user density such as steel plants and construction sites. In previous years, a large number of other specialists have presented object identification methods based on deep learning. The YOLO technique was introduced by Redmon et al. in 2015 and is significantly faster than previous methods. The YOLOv3 object detection mechanism was released by Redmon et al. in 2018, improving both the speed and precision of detection [8,9].

The YOLO storyline of methodologies then progressed to YOLOv5 by 2020. This renews the calls for helmet detection from the consumer safety sector. Using a special architecture called Scaled-YOLOv5, that enables the algorithm to achieve higher accuracy with fewer parameters and faster reasoning times, is one of the most important applications of YOLOv5. In order to improve accuracy, Scaled-YOLOv5 combines smaller and larger models that are trained successively and then integrated during inference [3–7].

Along with improving efficacy and accuracy, YOLOv5 offers a number of new features, including auto hyper-parameter adjustment, improved data reinforcement, and a range of pre-trained algorithms for different use cases. These characteristics make YOLOv5 a well-liked choice for real-time object recognition in a variety of applications, such as autonomous cars, security cameras, and bots. Nevertheless, YOLOv5 offers a novel training method called Self-Supervised Pre-training (SSP), which enables the algorithm to learn from unlabeled data before performing admirably on a labeled dataset. This technique improves YOLOv5's performance with fewer labeled illustrations, making it easier to train and use in practical implications [8–16].

III. PROFERRED METHOD

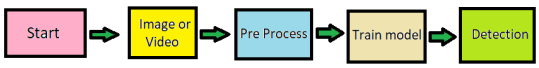

Fig. 1. Block Diagram of Proffered Method

Fig. 2. A Photograph of a Rush Traffic Road having People with helmet (Ref: [9])

Fig. 3. Photograph of People without helmet (Ref: [9])

In order for the YOLO model to be trained, a database of images must be formed and marked with the object. This model is then supplied to the YOLO method for a certain number of cycles, or epochs, to allow the method to gather sufficient data and information. The algorithm is then employed to track objects in real time. In order to properly train the model, we first gather a sufficient number of images of different traffic scenarios and camera angles of the target. Following a physical analysis of each image individually, a coordinate system is created to capture the needed object in every image [15–18].

Our suggested method's block diagram is shown in Fig. 1. We have employed a simple application called "labeling" for the particular task of marking the object across the dataset images. Its sole purpose is to enclose the objects in mathematical shapes (in this case a rectangle) and save the coordinates pertaining to the rectangle in a text file. The files retaining the images as well as their coordinate information, successively, are then used to create a.zip file, which is subsequently utilized in the YOLO training procedure. A scene of a road with heavy traffic is depicted in Fig. 2, and a picture of individuals without helmets is shown in Fig. 3.

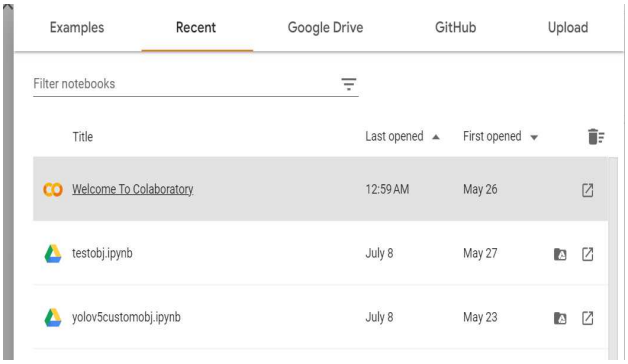

Fig. 4. A scenario of working with Google

With the help of Google Colab, a free online application, users can create and execute Python programs on Jupyter Notebooks (Fig. 4). It offers a cloud-based environment with virtual machine access and free GPU and TPU resources for running machine learning abstract representations and data analysis work. Google Colab makes it easy to share information and collaborate with others by allowing anyone to use and store their notebooks in Google Drive.

It is also possible to install and import Python libraries and to use powerful machine learning frameworks such as TensorFlow, PyTorch, and Scikit-learning. It can also be used to produce live visualizations and other things. All things considered, Google Colab is a powerful and helpful tool for anyone looking to learn and develop Python code online via the cloud.

A git-hub account cloning scenario is shown in Fig. 5. We import the yolov5 file along with all of its dependencies and other requirements next to obtain the result of cloning Fig. 5. Next, we use Google Colab to call Google Drive. After using labelimg to mark the images, a ".zip" data file is created and uploaded to a Google Drive account connected to a Google Colab account. We mount Google Drive and retrieve the uploaded data file location in this snippet.

Here, all we have to do is unzip the data file that is needed to train the Yolov5 approach. Subsequently, the dataset.yaml file gets retrieved, which essentially comprises the class (es) that we are employing to identify or indicate the object(s), while modifying the class names as necessary. A training scenario using data from Google Drive is shown in Fig. 6.

The yolov5s.pt file mentioned in Fig. 6 will be run, and AI will be trained over a range of 0-199 epochs. Additionally, the AI learns more with each epoch, as evidenced by the accuracy value increasing. Our necessary files and folders for the trained portion of YOLOV5 are taken into consideration following the training portion's successful completion. The Yolo v5 models consist of the Yolo v5n, v5s, v5m, v5l, and v5x. We are implementing our code on the Thonny Python Platform. Three libraries are used in our code.

A. cv2

The cv2 library, also called as OpenCV (Open Source Computer Vision Library), is a renowned open-source image processing and computer vision library. It offers an extensive set of functions and algorithms for tasks like picture filtering, object recognition, feature identification, camera calibration, and more. It can also be used for image and video input/output. Developers can use Python in their projects because it is one among the many programming languages for which bindings pertaining to import cv2 are made available. With a large user base and plenty of documentation and training, it is an essential tool for computer vision researchers, developers, and enthusiasts.

B. Torch

The "torch" library, also referred to as "PyTorch," is an effective open-source machine learning framework. For tasks like configuring and training neural networks, utilizing tensors, implementing deep learning techniques, and performing efficient GPU computations, it provides a wide range of functionalities. Users can easily create and test complex deep learning models differentiation features and dynamic computational graph. "PyTorch's" intuitive user interface, extensive documentation, and compatibility with cutting-edge features like networked computing have contributed to its rising prominence in the business and scientific domains. It has become a go-to resource for many machine learning practitioners and academics.

C. Numpy as np

Use the import "numpy" as np command to access the powerful "numpy" library, a useful tool for numerical computation. A large number of analytical operations are ample to efficiently work with large, multidimensional arrays and matrices. If we employ the as np component. We can employ the abbreviation np as an alternative name for the 'numpy' module, making it easier to refer to the functions conjointly with objects of the module in our code. The 'numpy' can be used for a variety of array operations, linear algebraic computations, statistical analysis, and other tasks. It is a vital part of the scientific computing ecosystem and is widely used in fields like data analysis, machine learning, and simulation. The helmet tracking code is shown in Fig. 7.

IV. RESULT DISCUSSION

Employing the Labelling tool, the training process comprises of manually labeling the images of the assigned class objects. As a result, we will receive the class label back. Using a PC with an Intel Core i7-10750H processor running at 2.6 GHz, an NVIDIA GeForce RTX 3050 GPU, and 16 GB of RAM, the research is conducted using Google Colab and a browser. We discovered that accuracy changes rapidly over every epoch. Fig. 8 shows the plots of train/box_loss, train/obj_loss, train/cls_los, etc. The Fig. 9 depicts a situation where hiking accuracy increases as the count of iterations increases. Figures 10 show the detected helmet and the bounding boxes for individual class after the code was run on the Thonny Python Platform.

V. CONCLUSION

The data displayed above makes it evident that YOLOv5 object detection successfully classified and localized every type of object while being well-suited for real-time processing. The proposed end-to-end model was built successfully and has all the parts that need to be automated and deployed for monitoring. This system will be able to afford even greater accuracy than the current prototype once it is outfitted with top-notch machinery. In the near future, more research might be conducted to determine and implement different camera angles for the system. There is just enough of the current dataset left over to build a working prototype model. The project's main objective was to deal with the problem of inadequate traffic control. We are currently working to make this model better. We intend to model for 5000 epochs in various traffic scenarios in the future.

```
!git clone https://github.com/freedomwebtech/yolov5train
!unzip /content/yolov5train/yolov5.zip -d /content/yolov5train
%cd /content/yolov5train/yolov5
!pip3 install -r requirements.txt
```

Fig. 5. Cloning of the github account

```
!python3 /content/yolov5train/yolov5/train.py --img 416 --batch 16 --epochs 500 --data /content/yolov5train/yolov5/dataset.yaml --weights yolov5s.pt
from IPython.display import Image

# Specify the path to the results.png image
image_path = '/content/yolov5train/yolov5/runs/train/exp/results.png'

# Display the image
Image(image_path)

```

Fig. 6. Training of the data from google drive

```
import cv2
import torch
import numpy as np


path ='A:/helmet detection/best.pt'
model = torch.hub.load('ultralytics/yolov5', 'custom',path, force_reload=True)

cap=cv2.VideoCapture('helmet4.mp4')
count=0
while True:
    ret,frame=cap.read ()
    if not ret:
        break
    count += 1
    if count % 3!= 0:
        continue
    frame=cv2.resize (frame, (760,560))
    results=model(frame)
    frame=np.squeeze (results. Render())
    results=model (frame)
    cv2.imshow("FRAME", frame)
    if cv2.waitKey(1)&0xFF==27:
        break
cap.release ()
cv2.destroyAllWindows()
```

Fig. 7. Code of Proffered Helmet Detection Model

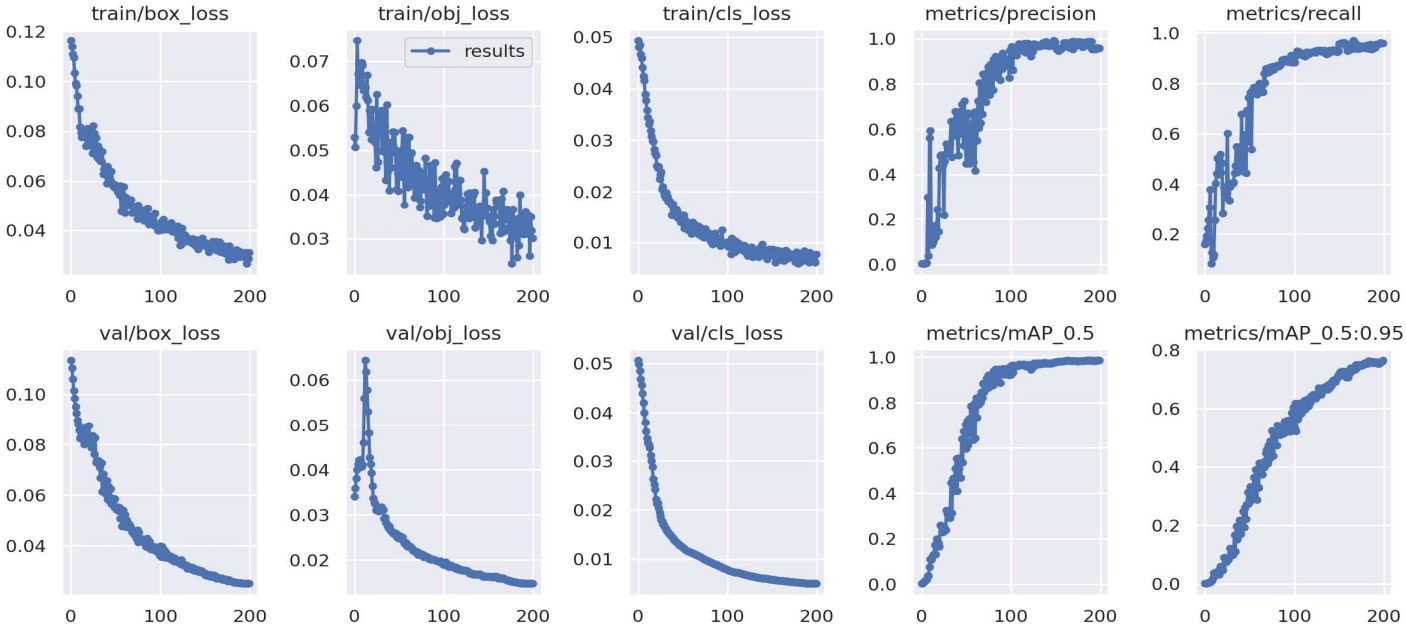

Fig. 8. Graphical Outcomes

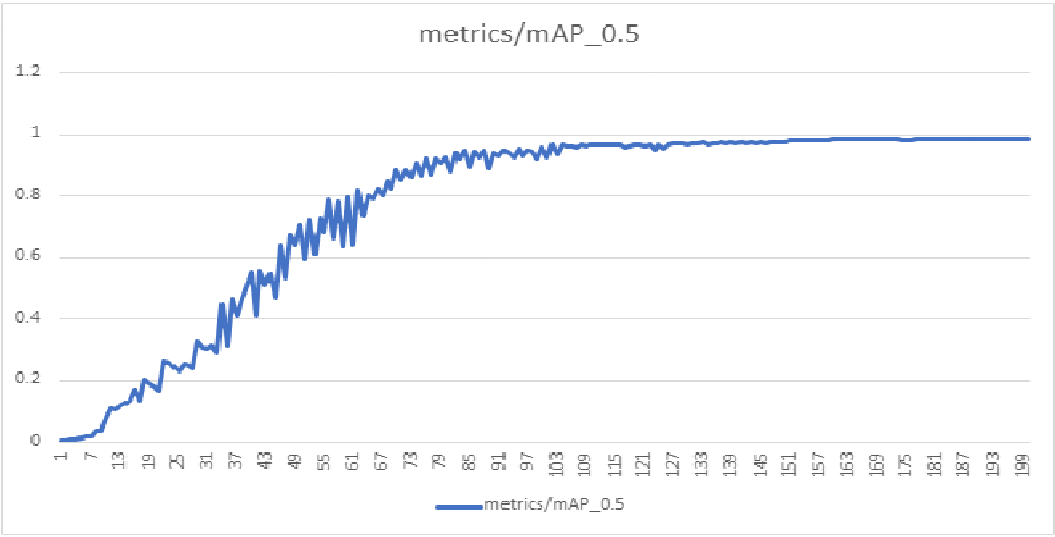

Fig. 9. Hike in Accuracy values with escalation in epochs

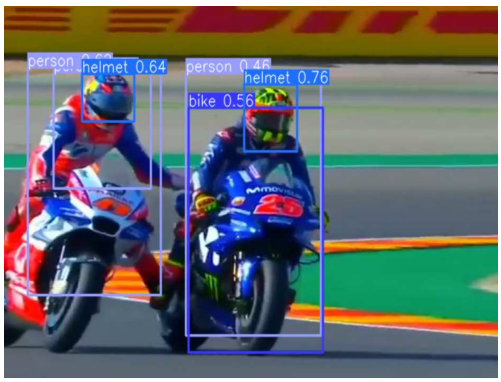

Fig. 10. Helmet Tracking

REFERENCES

[1] Nathan BAi, "Countries With The Highest Moterbike Usage" *Moter riders in Bangkok*

[1] Freedomwebtech,https://github.com/freedomwebtech/yolov5train

[2] https://www.freepik.com/premium-photo/professional-mechanical-engineer-team-working-construction-site_13641616.htm

[3] https://pytorch.org/hub/ultralytics_yolov5/

[4] Alexey Bochkovskiy, Chien-Yao Wang, and HongYuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In CVPR, vol. 1, 2016.

[6] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun. You only look one-level feature. In CVPR, 2021

[7] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In CVPR, 2017.

[8] https://www.worldatlas.com/articles/countries-that-ride-motorbikes.html

[9] A. Krizhevsky, I. Sutskever, G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Communications of the ACM, vol. 60, no. 6, pp. 84-90, 2017.

[10] S. M. Abbas and S. N Singh, "Region-based object detection and classification using faster R-CNN", 4th International Conference on Computational Intelligence & Communication Technology (CICT), IEEE, pp. 1-6, 2018.

[11] C. Kwan, et al., "Real-time and deep learningbased vehicle detection and classification using pixel-wise code exposure measurements", Electronics, vol. 9, no. 6, pp. 10-14, 2020.

[12] V. Bamane, J. Sapkale, A. Pawar, P. G. Chilveri, N. Akhter, N., A. A. B. Raj, "A Review on AI Based Target Classification Advanced Techniques", vol. 10, no. 4, pp. 88-99, 2022.

[13] R. Pérez, et al., "Deep-learning radar object detection and classification for urban automotive scenarios", Kleinheubach Conference, IEEE, pp. 1-4, 2019.

[14] T. Chen, et al., "Road marking detection and classification using machine learning algorithms", IEEE Intelligent Vehicles Symposium (IV) , IEEE, pp. 617-621, 2015.

[15] J. Redmon, et al., "You only look once: Unified, real-time object detection", In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788, 2016.

[16] N. Najva, K.E. Bijoy, "SIFT and tensor-based object detection and classification in videos using deep neural-networks", Procedia Computer Science, pp. 351-358, 2016.

[17] J. Hung, A. Carpenter, "Applying faster RCNN for object detection on malaria images", In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 56-61, 2017.

.