Speed Vision: A Deep Learning Based Approach for Car Speed Estimation

A Report submitted to Praxis Tech School in partial fulfillment of the requirements for the Post Graduate Program in Data Science and Artificial Intelligence at Praxis Business School

By:

Arshi Naaz(A23009)
Bidisha Sadhu(A23012)
Jayeeta Choudhury (A23022)
Nancy Lahiri (A23025)
Siddharth Tiwari (A23039)
Matta Sai Mohan Ranga (A23024)

Under the Guidance & Supervision of

Dr. Subhasis Das Gupta
Assistant Professor
Praxis Tech School
Kolkata, India

Post Graduate Program in Data Science
Praxis Tech School
Praxis Business School (Kolkata Campus)
Kolkata- 700104
April, 2024

Contents

List of Figures

List of Tables

1 Introduction:

In today's transportation landscape, the ongoing issue of upholding road safety continues to be a major priority [1].The ability to accurately measure and monitor the speed of vehicles on roads and highways is essential for ensuring compliance with speed limits, preventing accidents, and optimizing traffic flow. Studies have shown that speeding is a leading cause of traffic fatalities and injuries, emphasizing the importance of effective speed enforcement measures. By deterring speeding behaviour and encouraging compliance with speed limits, speed detection technologies contribute to the prevention of accidents and the mitigation of their consequences. This is where an effective speed monitoring system plays an important role. The need for monitoring vehicle speed dates back to the early days of automotive transportation. Historically, speed limits were enforced through manual observation by law enforcement officers. Traffic management authorities use speed detection systems to monitor traffic flow, identify congestion hotspots, and implement adaptive speed limits. Law enforcement agencies utilize these technologies to enforce speed limits, detect speeding violations, and improve overall road safety. Additionally, speed detection plays a crucial role in automated driving systems, enabling vehicles to adjust their speed dynamically based on road conditions and surrounding traffic. With the advancements in technology, automated methods for speed detection have emerged, offering more accurate and efficient means of enforcement.

Modern vehicle speed detection systems rely on a variety of technologies, including radar, LIDAR, and video-based systems. Radar-based systems use radio waves to measure the speed of vehicles by detecting the Doppler shift in the frequency of reflected signals. Lidar systems employ laser beams to calculate vehicle speed based on the time it takes for light pulses to reflect off a vehicle and return to the sensor. Video-based systems utilize cameras and image processing algorithms to track vehicle movement and estimate speed.

The current project falls into the third category, that is, video-based systems. To effectively address the challenge of speed detection and enhance road safety, the proposed project integrates advanced technologies such as Convolutional Neural Networks (CNN) [1], YOLOv8 [2][3], and OpenCV. The project effectively utilizes OpenCV to break down videos into individual frames, laying the groundwork for further analysis. YOLOv8, known for its object detection [4] capabilities, is then employed to accurately identify vehicles in the video footage, thus improving the precision of speed detection.

Additionally, OpenCV plays a vital role in seamlessly processing and interpreting the data. The main objective goes beyond simply detecting vehicle speeds; it aims to do so in a cost-effective manner, making a significant contribution to enhancing road safety measures. By combining YOLOv8

and OpenCV, the project establishes a strong technological foundation that has the potential to transform speed detection methods and improve overall traffic management strategies.

The project also involves the comparison of YOLOv5, SSD, and YOLOv8 models for their applicability in speed detection and traffic management. YOLOv5 is known for its efficiency and accuracy, while SSD is a popular object detection model. YOLOv8 represents an updated version of the YOLO series. Factors such as detection speed, accuracy, and resource utilization were assessed to understand their performance in real-world scenarios. Each model has unique strengths and considerations, making them suitable for various applications based on specific requirements.

The current study tries to address two objectives. Firstly, it proposes a new method of estimating the speed of a vehicle using the concept of object detection and depth estimation in real-time. Secondly, the project compares different object detection methods to understand the best deep learning model that can be utilized for more accurate speed estimation. The study is primarily application-based in which, there is a good amalgamation of deep learning and statistical pattern recognition.

The remaining portion of the report is organized in the following manner. Section 2 focuses on the details of the objectives of the project where the actual speed estimation problem is further broken down into a machine-learning problem. Section 3 is important as it covers the background studies related to the current work. Section 4 gives full details of the methodology that has been adopted to carry out the project. This section also has a flowchart which can help understand the process end-to-end. Section 5 deals mainly with the results and analyses. Section 6 gives conclusive remarks whereas Section 7 gives ideas on extending the work further in future.

2 Objectives of the Current Project:

Recent advancements in computer vision and deep learning have transformed vehicle speed detection. However, accurately estimating speeds in dynamic traffic remains challenging due to limitations in instance segmentation and depth estimation. This project integrates YOLOv8, Mask R-CNN, and Midas models with OpenCV to improve speed detection accuracy in complex traffic environments. The project tries to address the following objectives:

Utilize YOLOv8 and OpenCV for Vehicle Speed Detection:

YOLOv8 (You Only Look Once) represents a cutting-edge deep learning architecture designed for object detection. Its primary application will involve accurately identifying vehicles within video frames. OpenCV [5], a computer vision library, will play a pivotal role in supporting tasks like video input/output, image preprocessing, and seamless integration with YOLOv8.

Incorporate Midas for Depth Estimation:

The Midas [6] model stands out as a specialized deep learning model crafted for monocular depth estimation, offering depth insights for every pixel within an image. Midas will seamlessly integrate into the pipeline to gauge the depth of objects within the scene, encompassing vehicles. This depth data will elevate the precision and dependability of vehicle speed detection by facilitating more accurate distance computations.

Integrate Mask R-CNN for Precise Instance Segmentation:

Mask R-CNN (Region-based Convolutional Neural Network) [7] is a sophisticated deep learning model specialized in instance segmentation. This entails the precise identification and delineation of individual objects within an image with pixel-level accuracy. Mask R-CNN will be harnessed to accurately segment vehicles within video frames, facilitating the extraction of centroids (center points) for each vehicle. This centroid data plays a vital role in computing distances between vehicles in subsequent frames.

Conduct Comparative Analysis with SSD and YOLOv5:

SSD (Single Shot Detector) [8][9] and YOLOv5 (You Only Look Once version 5) [10] represent alternative deep learning architectures tailored for object detection and speed estimation. These models will be incorporated into the project for comparative analysis, assessing their performance, accuracy, computational efficiency, and robustness in predicting vehicle speeds when compared to YOLOv8 and the integrated Mask R-CNN + Midas pipeline. By leveraging the strengths of YOLOv8, Mask R-CNN, Midas, and OpenCV, this project aims to develop a comprehensive and resilient vehicle speed detection system. The integration of deep learning models for object detection, instance segmentation, depth estimation, and pixel distance calculations will enable precise and dependable speed estimation for moving vehicles.

By combining the strengths of YOLOv8, Mask R-CNN, Midas, and OpenCV, this project aims to develop a comprehensive and robust vehicle speed detection system. The integration of deep

learning models for object detection, instance segmentation, depth estimation, and pixel distance calculations will enable accurate and reliable speed estimation for moving vehicles.

3 Related Work:

Vehicle speed detection is not a new concept. There are various methods available for speed detection of a moving vehicle. The common methods deal with sensors placed on the road at different locations, laser-based method (LIDAR) [11, 12, 13] and Radar-based method employing doppler shift [14, 15, 16]. These methods rely on concepts related to physics and transducers. These methods although quite common in usage, are not free from limitations. For example, the manual count method [17] involves observing two lines created by a model on a frame perpendicular to the vehicle's direction of movement. An individual starts a stopwatch as the vehicle crosses the first line and stops it as soon as the vehicle leaves the second line. The distance between the lines is constant and the speed is found out by simply dividing the distance by the time difference. This method, while intuitive, often lacked precision and required manual intervention. Additionally, interpreting visual cues for speed estimation could be subjective and prone to error.

The method involving sound waves involves the Doppler effect[18] generated by sound waves emitted from moving cars. By analyzing the frequency shift of these waves, it is possible to infer the speed of the vehicle. However, this approach is complex and requires specialized equipment, making it impractical for widespread use. Moreover, environmental factors such as background noise and interference could affect the accuracy of speed measurements. Despite the advancements in these traditional techniques, they were often limited by their complexity, cost, and reliance on specialized equipment. Moreover, the subjective nature of visual observations and the technical challenges associated with sound wave analysis posed significant barriers to widespread adoption.

LIDAR-based methods are quite accurate but they are costly. This is mainly due to the fact that the device deals with laser beams and tries to detect the minute changes in the reflected beams. Moreover, these devices may undergo mechanical wear and tear leading to erroneous results. LIDAR based devices have a narrower view which makes them more selective in nature. Radar based systems, on the other hand, have wider view and when there is less environmental noise, can detect the fastest moving vehicle.

The other method, which is becoming quite popular, is vision-based, that is, relying on computer vision. When it comes to computer vision, deep learning models dominate the area. Particularly, Convolutional Neural Network (CNN) models play a key role. The primary objective of a CNN model is object classification. However, object classification is not enough for speed detection because speed detection requires the detection of a vehicle in a frame. Hence, object localization becomes equally important in such scenarios. Object classification, along with object localization,

leads to object detection in the literature of computer vision [19, 20]. In this context, there are quite a few prominent models available for object detection such as YOLO [21], Fast YOLO,[22], Faster R-CNN [23], Single Shot Multibox Detection (SSD) [24] and AttentionNet [25]. These models are built on pre-trained models such as ResNet50 [26], DenseNet [27], MobileNet [28], VGG16 [29], AlexNet [30] etc which are trained on a large number of images. These models act as the base model or the backbone model for the object detection models. Custom neural network architectures are placed on top of these base models to perform both object classification and object location which is, essentially, the bounding box regression [31, 32].

Another important area of research in the domain of computer vision is depth estimation [33]. Depth estimation is, essentially, measuring the distance of an image pixel from the camera lens. It is usually denoted as a color map. A colour which is having a red shift indicates the level of closeness of the object associated with that pixel and the colour which is having a blue shift indicates that the pixel associated with the object is away from the camera. Depth estimation has other applications also such as volume estimation [34] and 3D representation [35]. In the current project, the same depth estimation was applied to estimate vehicle speed.

This project utilized the existing pre-trained models to estimate vehicle speed by prioritizing simplicity, affordability, and accessibility. Leveraging smartphone cameras, it provided a practical solution that eliminated the need for complex hardware and specialized expertise. This democratization of speed detection not only reduces costs but also enables applications in traffic management, road safety, and smart transportation systems. Additionally, the enhanced accuracy and efficiency of YOLOv8 ensured reliable speed detection, even at higher speeds. Moreover, by incorporating depth estimation techniques, it enhanced the efficacy and accuracy of speed measurements. This integration of depth perception improved performance, particularly in scenarios where visual cues alone may be insufficient.

4 Methodology:

Data Acquisition:

Quality of data plays a very important role in the area of research. For the current study, the data was collected by the researchers. For this study, around 100 data points were collected by carefully curating the videos of moving vehicles. The videos were captured by the hand held mobile phones having 640x480 resolution. For the present study, videos were collected in such a way so that only one car was there in the frame. Each video was either, shot from the front or from the side perspective. The videos varied in duration from 3 seconds to 7 seconds. The researchers were told the speed of the car which were recorded and the subsequent videos were mapped with the respective speeds.

Integrating OpenCV for seamless video input:

Videos are essentially collections of several images presented sequentially, known as frames. For effective video analysis, it's crucial to analyze these frames in a sequential manner to extract meaningful insights. While there are several open-source packages available for this purpose, OpenCV stands out as a highly popular and versatile package for video and image analysis tasks. Its wide range of functionalities includes seamless video input processing, dynamic dimension adjustments, and efficient preprocessing techniques.

Moreover, OpenCV offers seamless integration with other deep learning-based models, expanding the scope of image analytics significantly. This capability allows for the incorporation of advanced models like YOLOv8, Mask R-CNN, and Midas, enhancing the depth and accuracy of the analysis. Hence, for the current study on speed estimation from video datasets, OpenCV was the natural choice, ensuring comprehensive and reliable feature extraction for precise speed estimation.

YOLOv8 Object Detection:

As mentioned earlier, YOLOv8 [36] [37] is quite capable of identifying vehicles at higher speeds. YOLOv8, renowned for its efficiency and accuracy, divides the input image into a grid and predicts bounding boxes and class probabilities directly from each grid cell. By focusing on the area of bounding boxes and timestamps, it was possible to extract crucial information needed for vehicle speed estimation. By utilizing YOLOv8 both these features were easy to extract.

Depth Estimation with MiDaS and Mask R-CNN:

Depth estimation is a separate area of research under the gamut of deep learning. 2D images lack information on the depth. We humans, however, have our understanding of the distance of an object in a 2D image but that estimation is quite crude. Not only that, our understanding has no apparent usage when deep learning based models are to be employed. Hence, depth analysis on a 2D image becomes important and, can be quite helpful in several different situations. There are many research works available on the topic of depth estimation but, in this project, MiDaS [38] was used for estimating depth as the authors had shown good performance metrics of their proposed model. The output of the model was essentially a color map depicting the distance of each pixel from the camera. In this project, vehicle speed was required to be estimated and bounding boxes had portions of background images which could have interfered with the actual distance measurements. Hence, Mask R-CNN model was used along with YOLOv8 to get a more precise distance estimation of the vehicle from the video frames. The objective of the Mask R-CNN was to use distances of only those pixels that were inside the masked region of the vehicle as predicted

by the Mask R-CNN model. The centroid (or average distance) of the mask was considered as the average distance of the vehicle from the camera.

Data Processing and Analysis:

The methodology involved developing code to extract area, time, and depth information from Mask R-CNN [39] [40] outputs and structuring the data into a CSV file format. Linear and non-linear regression analyses were conducted using the extracted data to identify the best-fitting model for speed prediction. This process included exploring various regression techniques to understand the relationship between area, time, depth, and actual speed. Valuable insights into the factors influencing vehicle speeds in dynamic traffic scenarios were obtained through this comprehensive analysis.

Model Evaluation and Selection:

The performance of linear and non-linear regression models was evaluated, with a focus on adjusted R-squared values to identify the most accurate speed prediction model. It was determined that YOLOv8's non-linear regression model yielded the highest adjusted R-squared value, indicating superior predictive capabilities compared to other models. This finding underscores the effectiveness of YOLOv8 in accurately estimating vehicle speeds, particularly in dynamic traffic scenarios where precise predictions are crucial for enhancing road safety measures.

Speed Prediction and Validation:

The beta values obtained from YOLOv8's non-linear regression model were utilized to estimate vehicle speeds in real-world scenarios. Two video datasets captured from front and side angles were employed, allowing for a comprehensive comparison of predicted speeds with known ground truth speeds for validation and accuracy assessment. This approach enabled the evaluation of YOLOv8's performance in diverse traffic scenarios, highlighting its effectiveness in accurately predicting vehicle speeds across different viewing angles and real-world conditions.

SSD and YOLOv5

The same process is being used for SSD and YOLOv5, ensuring consistency and comparability across different object detection models.

Block diagram of the workflow

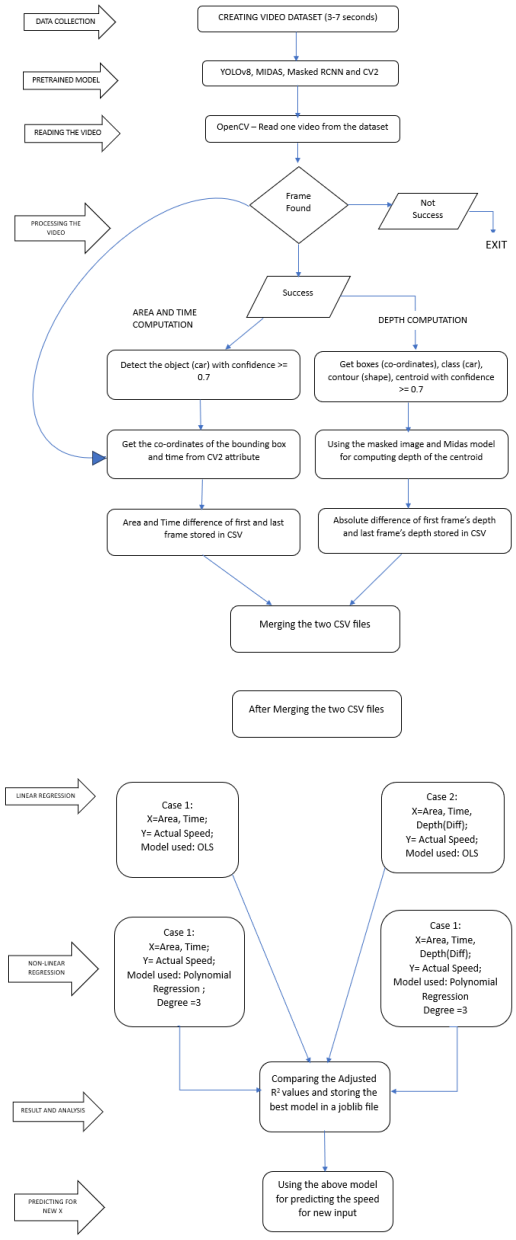

Figure 1: Block Diagram

Comparison among the architecture of YOLO, SSD and MaskRCNN:

YOLO Architecture:

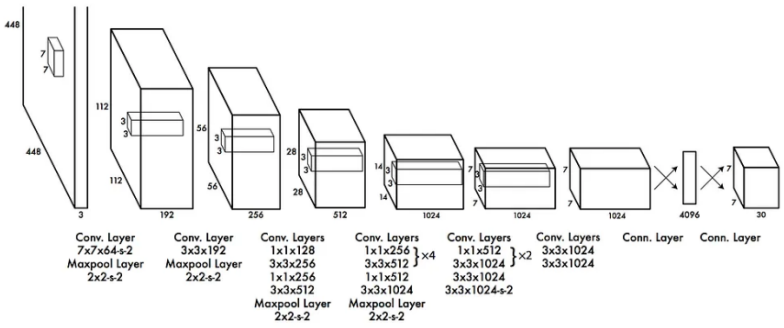

Figure 2: Architecture Diagram of Yolov1

YOLO (You Only Look Once) employs a singular neural network architecture [41] that operates by partitioning the input image into a grid. From this grid, it directly forecasts bounding boxes and class probabilities. The architecture's terminal layers are comprised of fully connected layers. These layers utilize the features extracted by the convolutional layers, constituting the base network, to yield bounding box coordinates and class probabilities for each grid cell. YOLO's predictive process entails regressing from the grid cells to the eventual bounding box coordinates. It anticipates the coordinates (x, y, width, height) of the bounding boxes alongside class probabilities for each bounding box. This methodology enables YOLO to render predictions for all objects within a solitary forward pass through the network, thereby rendering it swift and efficient for real-time applications.

SSD Network:

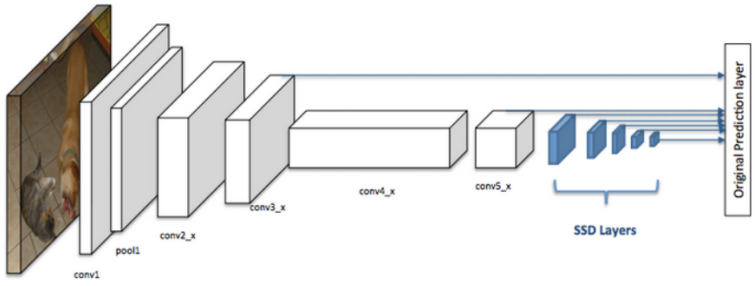

Figure 3: Architecture Diagram of SSD

Similar to YOLO, SSD (Single Shot MultiBox Detector) is a single-shot object detection model, capable of predicting bounding boxes and class probabilities in a single pass through the network. However, SSD employs a distinct strategy in its final prediction layers. Instead of relying on fully connected layers, SSD integrates convolutional layers of diverse sizes. It applies convolutional filters of varying dimensions to feature maps obtained from different layers of the base network. These filters play a pivotal role in predicting bounding box offsets and class probabilities across multiple scales. By leveraging convolutional layers for prediction, SSD adeptly captures objects at various scales and aspect ratios, facilitating more effective detection of objects of different sizes within the image. This approach contributes to SSD's ability to deliver accurate and efficient object detection across a wide range of scenarios and applications.

R-CNN:

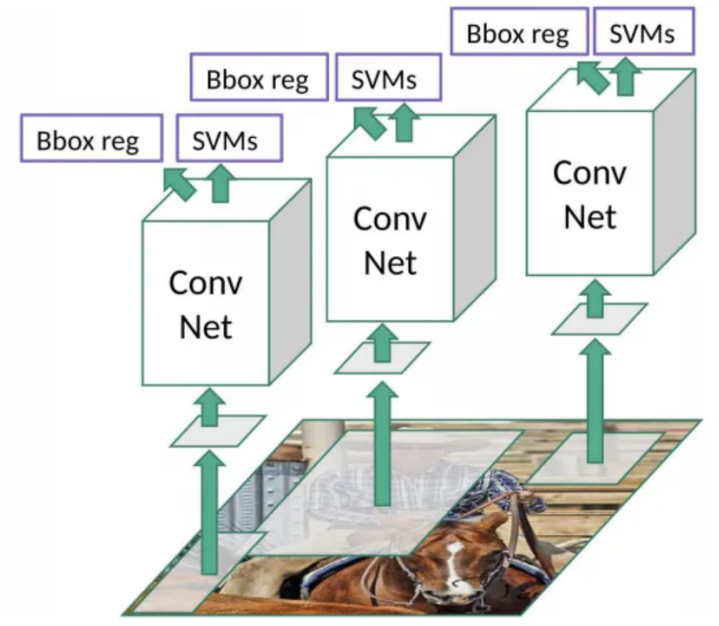

Figure 4: Architecture Diagram of RCNN

The RCNN (Region-based Convolutional Neural Network) architecture is a pioneering approach in object detection, notable for its distinctive methodology. Unlike single-shot detection models such as YOLO and SSD, RCNN operates in a two-step process. Initially, it proposes regions of interest (RoIs) within an image using a selective search algorithm or a similar method. These RoIs represent candidate regions likely to contain objects. In the second step, each proposed RoI is individually processed by a convolutional neural network (CNN), extracting features specific to that region.

These features are then fed into a series of fully connected layers, culminating in the prediction of bounding box coordinates and class probabilities for objects within the RoI. RCNN's approach allows for precise localization and classification of objects within complex scenes, albeit at the cost of increased computational complexity due to the need to process each RoI individually. Despite this drawback, RCNN laid the groundwork for subsequent advancements in object detection, inspiring the development of faster and more efficient models like Fast R-CNN and Faster R-CNN.

Mask R-CNN:

Mask R-CNN performs instance segmentation, predicting pixel-level masks for each detected object instance in addition to bounding boxes and class labels. This enables precise delineation of object boundaries and differentiation between individual instances of the same class, enhancing tasks requiring accurate object localization and segmentation.

Midas:

Depth estimation, also known as depth prediction or depth mapping, refers to the process of inferring the distance of objects in a scene from a given viewpoint. In computer vision, depth estimation is typically performed using images or videos captured by cameras, and the goal is to create a depth map that assigns a depth value to each pixel in the image.

MiDaS is a deep learning architecture that leverages an encoder-decoder structure, multi-scale features, dense attention mechanism, and supervised training to estimate depth from a single image accurately.

The output of the MiDaS (Mixed-scale Dense Attention) model is a depth map. A depth map is a two-dimensional representation where each pixel in the image is assigned a corresponding depth value. The depth value indicates the distance of the corresponding point in the scene from the camera or the observer's viewpoint.

Comparison of Mask R-CNN, YOLO, and SSD:

- Backbone Network:
  - Mask R-CNN: Utilizes a backbone network (e.g., ResNet, VGG) to extract features from the input image.
  - YOLO: Employs a base network, often a variant of Darknet, for feature extraction.
  - SSD: Also uses a backbone network (e.g., VGG, ResNet) for feature extraction.
- Region Proposal:

  - Mask R-CNN: Incorporates a Region Proposal Network (RPN) to generate candidate object bounding boxes.
  - YOLO: Divides the input image into a grid and directly predicts bounding boxes and class probabilities for each grid cell.
  - SSD: Employs default anchor boxes at different scales and aspect ratios to predict object bounding boxes.

- Detection Head:
  - Mask R-CNN: Includes a detection head that predicts bounding box coordinates, object class labels, and pixel-level segmentation masks for each proposed region.
  - YOLO: Uses fully connected layers to predict bounding box coordinates and class probabilities directly from the feature map.
  - SSD: Predicts bounding box coordinates and class scores directly from multiple feature maps with different scales.

- Output:
  - Mask R-CNN: Provides predicted bounding boxes, class labels, and segmentation masks for objects in the image.
  - YOLO: Outputs bounding box coordinates and class probabilities for detected objects.
  - SSD: Outputs predicted bounding boxes and class scores for detected objects.

- Segmentation:
  - Mask R-CNN: Unique to Mask R-CNN, includes a segmentation branch for pixel-level segmentation mask prediction.
  - YOLO: Does not provide pixel-level segmentation.
  - SSD: Also does not provide pixel-level segmentation.

- Speed:
  - Mask R-CNN: Generally slower due to its more complex architecture, especially with the additional segmentation branch.
  - YOLO: Known for its speed and efficiency as it processes the entire image in one pass through the network.
  - SSD: Offers good speed and efficiency due to its single-shot approach.

- Applications:

- Mask R-CNN: Well-suited for tasks requiring precise object segmentation, such as instance segmentation and image editing.
- YOLO: Ideal for real-time object detection applications due to its speed and efficiency.
- SSD: Widely used for various object detection tasks, including pedestrian detection, vehicle detection, and general object detection.[42]

YOLOv1 vs YOLOv5 and YOLOv8

YOLOv1:

YOLOv1 [43] introduced the concept of real-time object detection by dividing the input image into a grid and predicting bounding boxes and class probabilities directly from each grid cell.

Limitations: YOLOv1 had limitations in accuracy, especially in detecting small objects like bikes and scooters, and handling complex scenes with overlapping or closely spaced objects. This could impact the accuracy of speed detection and road safety efforts in dynamic traffic scenarios.

YOLOv5:

YOLOv5 improved upon YOLOv1's speed and generalization capabilities by introducing a streamlined architecture and improved training methodologies.

Advantages: YOLOv5 offers better speed and generalization compared to YOLOv1, making it suitable for real-time object detection tasks. It can detect small objects as well.

YOLOv8:

YOLOv8 incorporates advanced architectural enhancements such as feature pyramid networks (FPN) and cross-stage partial connections (CSP), significantly improving object detection accuracy and robustness.

Advantages for the Project:

- Enhanced Accuracy: YOLOv8's advanced architecture ensures better detection and tracking of vehicles, including small objects like bikes and scooters, in complex traffic scenarios. This is crucial for precise car identification.
- Real-Time Performance: YOLOv8 maintains real-time performance while delivering improved accuracy and robustness, making it suitable for timely car identification tasks.

- Generalization: YOLOv8 benefits from advancements in training methodologies, leading to better generalization and performance on diverse datasets, making it well-suited for accurate car identification in various traffic scenarios.

Overall, while YOLOv5 represents an improvement over YOLOv1 in speed and generalization, YOLOv8 surpasses both YOLOv1 and YOLOv5 in accuracy, robustness, real-time performance, and generalization, making it the superior choice for the project's objectives of accurate speed detection and enhanced road safety.

5 Results and Analysis:

- 90 points are taken where video length is 3-7 sec
- Models Implemented: YOLOv8, YOLOv5, SSD
- Regression Models: Linear(OLS) and Nonlinear(Polynomial degree-3)

Figure 5: Area calculation in front and side view

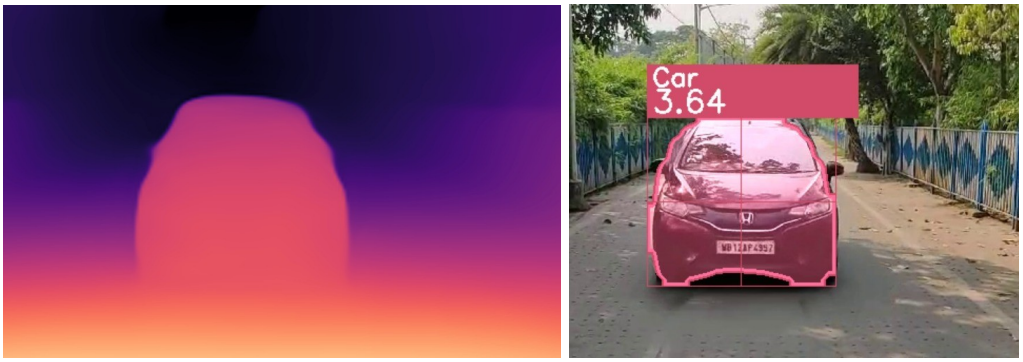

Figure 6: Depth Map and Mask RCNN masked image front view

Figure 7: Depth Map and Mask RCNN masked image side view

Independent Variables:

- Area difference of frames
- Time
- Depth ("diff" variable)

Dependent Variable:

- Actual Speed

Implementation of YOLOv8, YOLOv5, and SSD models for object detection is done. Two values are obtained-Area and time Midas along with MaskRCNN is applied to get the value of the depth(diff) Using the above values linear and nonlinear regression is used Finally, comparison of

the results with and without considering the depth variable is made.

Model Name	**Regression**	**AdjR^2 without diff**	**AdjR^2 with diff**	**R^2 with diff**	**MSE with diff**	**RMSE with diff**
YOLO v8	Linear Regression	0.23	0.50	0.52	12.29	4.90
YOLO v8	Non Linear Regression	0.61	0.74	0.81	5.02	2.24
YOLO v5	Linear Regression	0.15	0.45	0.47	26.82	5.18
YOLO v5	Non Linear Regression	0.15	0.66	0.76	6.24	2.50
SSD	Linear Regression	0.13	0.46	0.48	27.90	5.28
SSD	Non Linear Regression	0.20	0.60	0.72	7.34	2.71

Table 1: Model Results

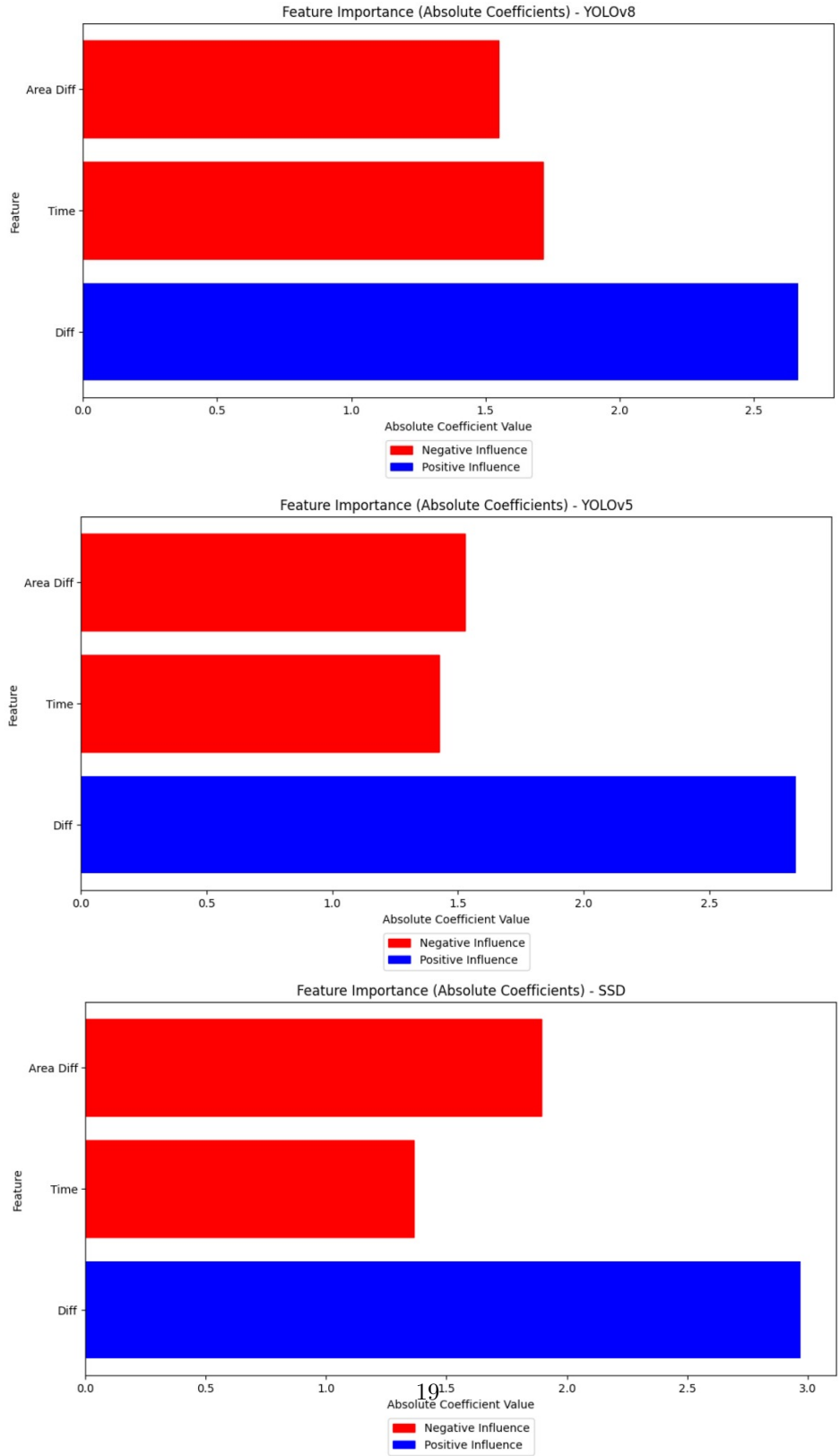

Figure 8: Best features selection in Linear Regression

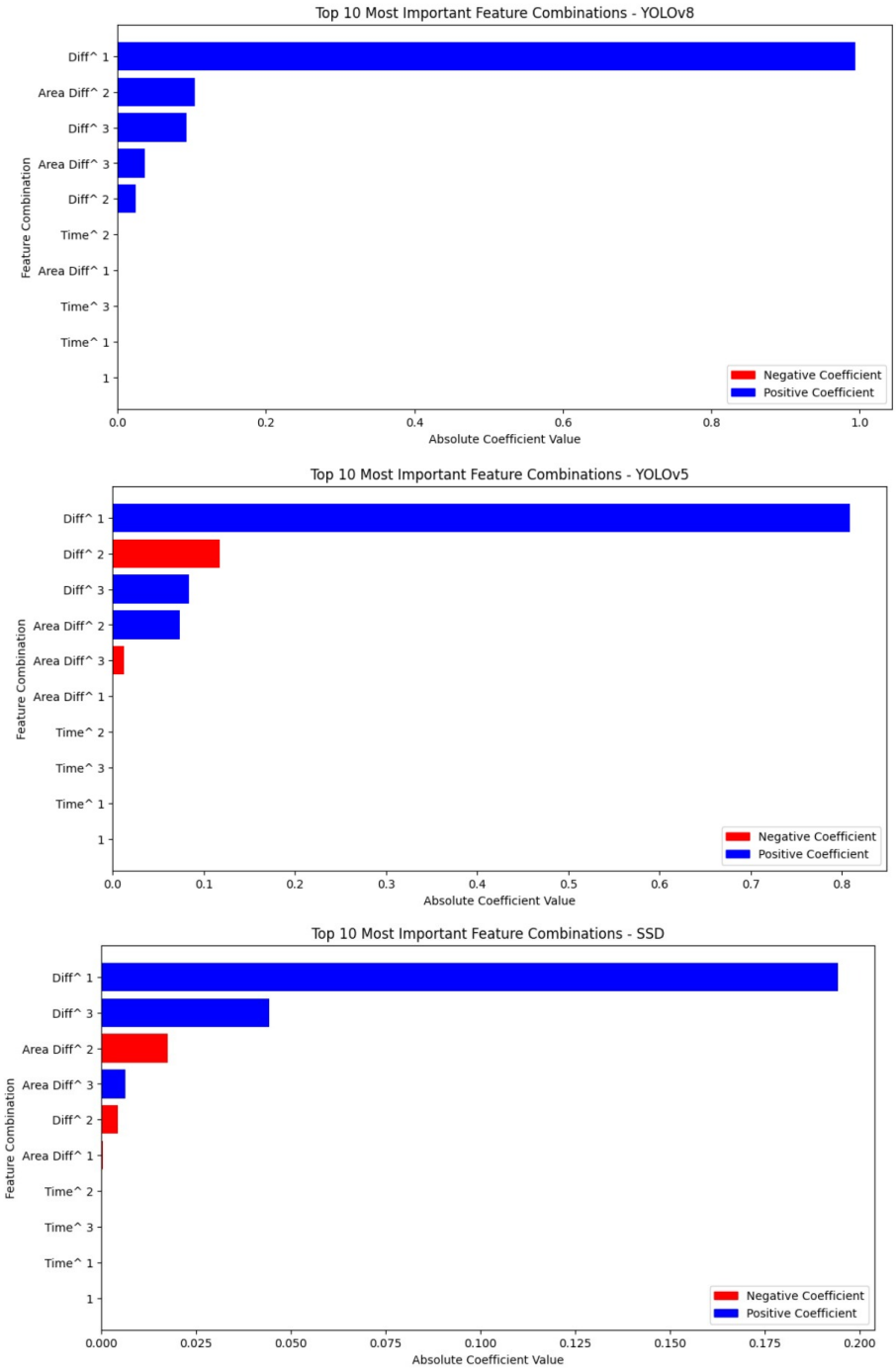

Figure 9: Best features selection in Non Linear Regression

The addition of the "diff" variable (depth) improved the accuracy of speed prediction in all models.

YOLOv8 demonstrated the highest accuracy when depth was considered, followed by YOLOv5 and SSD.

Linear and nonlinear regression models performed better with the inclusion of the depth variable, indicating its importance in speed estimation.

Output video

Figure 10: Output video gif screenshot from front view

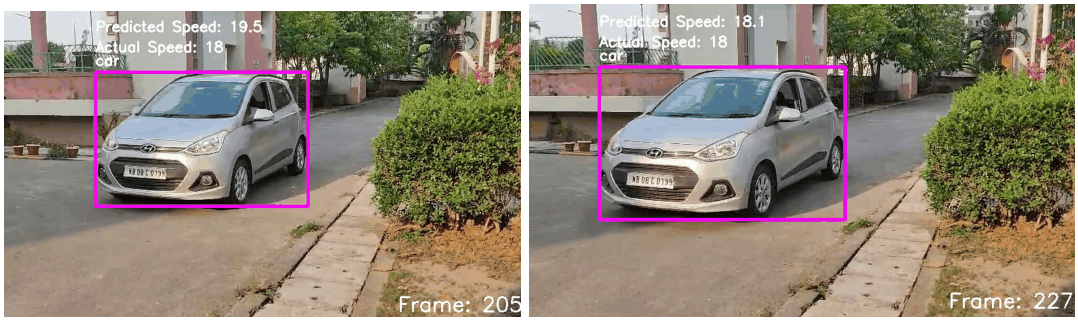

Figure 11: Output video gif screenshot from side view

In the given figure, two frames exhibit predicted data of the same video are taken, one for front view and other side view. As more frames are processed, the difference between the actual and predicted values decreases. This trend persists with additional frames, resulting in a diminishing gap between the predicted and actual values, especially for new data points the model has not encountered before.

6 Conclusion:

The current work presents an alternative method for calculating the speed of a moving vehicle. Despite employing a non-linear regression approach with a smaller dataset, the resulting adj R-squared value is notably high. Furthermore, the calculated speeds closely match the actual values, with a slight margin of error.

This particular work shows a different way of estimating speed. Integrating two deep learning architectures.
Object Detection: Precise vehicle classification, bounding box area, and time calculation.
Depth Estimation: Utilizing Midas and MaskRCNN for getting centroid pixel distance.

Further a Non-linear regression model is used to understand the relationships among features.

Owing to temporal and resource limitations, the work couldn't be compared with existing methods. Nevertheless, the dataset analysis shows that the model functions well. As a result, our model advances our understanding of speed estimation using only video analysis.

In the comparative analysis with YOLOv5 and SSD, the same methodology is followed. The results from the model are utilized to implement regression analysis, starting with linear and non-linear regression using area difference and time. Subsequently, depth is included as an additional variable to enhance the models. The non-linear regression model with depth consistently demonstrates superior predictive capabilities.

Validation through real-world scenarios confirms the accuracy of this approach. The inclusion of depth as an independent variable significantly enhances speed prediction across all models. The structured CSV output facilitates easy interpretation and utilization of speed estimations for further analysis and decision-making processes.

7 Future Direction

Expanding the project's application to encompass heavier vehicles like lorries and smaller ones such as bikes and scooters reflects the commitment to enhancing overall road safety impact across diverse vehicle categories. By developing specialized models and algorithms tailored to each vehicle type, it aims to address specific safety challenges, contributing significantly to reducing road accidents and ensuring safer transportation infrastructure.

Furthermore, implementing advanced algorithms to comprehensively evaluate driver behavior, including speed patterns, lane discipline, and adherence to traffic rules will allow to identify instances of reckless driving accurately. This proactive approach not only enhances road safety but also facilitates targeted interventions and corrective measures to promote responsible driving behavior among motorists.

Moreover, leveraging real-time data collection using handheld devices with 5G connectivity and cloud-based technologies enables efficient data processing and timely decision-making for road safety interventions. This integration ensures a dynamic and robust approach towards enhancing road safety and creating safer road environments for all road users.

References

[1] A. Grents, V. Varkentin, and N. Goryaev, "Determining vehicle speed based on video using convolutional neural network," *Transportation Research Procedia*, vol. 50, pp. 192–200, 01 2020.

[2] A. Vats and D. C. Anastasiu, "Enhancing retail checkout through video inpainting, yolov8 detection, and deepsort tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 5529–5536.

[3] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, 2023.

[4] A. Marode, A. Ambadkar, A. Kale, and T. Mangrudkar, "Car detection using yolo algorithm," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 03, pp. 2582–5208, 05 2021.

[5] "Image Processing Using OpenCV – With Practical Examples," https://www.analyticsvidhya.com/blog/2021/05/image-processing-using-opencv-with-practical-examples/.

[6] "Getting Started with Depth Estimation using MiDaS and Python," https://medium.com/artificialis/getting-started-with-depth-estimation-using-midas-and-python-d0119bfe1159.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," 2018.

[8] J. Cao, C. Song, S. Song, S. Peng, D. Wang, Y. Shao, and F. Xiao, "Front vehicle detection algorithm for smart car based on improved ssd model," *Sensors*, vol. 20, no. 16, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/16/4646

[9] H.-T. Choi, H.-J. Lee, H. Kang, S. Yu, and H.-H. Park, "Ssd-emb: An improved ssd using enhanced feature map block for object detection," *Sensors*, vol. 21, no. 8, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2842

[10] F. N. W. R. W. M. K. B. S. I. Mahendrakar T., Ekblad A., "Performance study of yolov5 and faster r-cnn for autonomous navigation around non-cooperative targets," *IEEE*, vol. 978-1-7281-7436-5/21. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/2301/2301.09056.pdf

[11] J. Zhang, W. Xiao, B. Coifman, and J. P. Mills, "Vehicle tracking and speed estimation from roadside lidar," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 5597–5608, 2020.

[12] M. A. Samuels, S. W. Patterson, J. A. Eppstein, and R. L. Fowler, "Low-cost hand-held lidar system for automotive speed detection and law enforcement," in *Laser Radar VII: Advanced Technology for Applications*, vol. 1633. SPIE, 1992, pp. 147–159.

[13] C. W. Hsu, T. H. Hsu, and K. J. Chang, "Implementation of car-following system using lidar detection," in *2012 12th International Conference on ITS Telecommunications*. IEEE, 2012, pp. 165–169.

[14] P. Misans and M. Terauds, "Cw doppler radar based land vehicle speed measurement algorithm using zero crossing and least squares method," in *2012 13th Biennial Baltic electronics conference*. IEEE, 2012, pp. 161–164.

[15] S.-L. Jeng, W.-H. Chieng, and H.-P. Lu, "Estimating speed using a side-looking single-radar vehicle detector," *IEEE transactions on intelligent transportation systems*, vol. 15, no. 2, pp. 607–614, 2013.

[16] S. Chadwick, W. Maddern, and P. Newman, "Distant vehicle detection using radar and vision," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8311–8317.

[17] M. A. Adnan, N. Sulaiman, N. I. Zainuddin, and T. B. H. T. Besar, "Vehicle speed measurement technique using various speed detection instrumentation," in *2013 IEEE Business Engineering and Industrial Applications Colloquium (BEIAC)*, 2013, pp. 668–672.

[18] H. Rodríguez-Rangel, L. A. Morales-Rosales, R. Imperial-Rojo, M. A. Roman-Garay, G. E. Peralta-Peñuñuri, and M. Lobato-Báez, "Analysis of statistical and artificial intelligence algorithms for real-time speed estimation based on vehicle detection with yolo," *Applied Sciences*, vol. 12, 2022.

[19] Y. Wu, Y. Chen, L. Yuan, Z. Liu, L. Wang, H. Li, and Y. Fu, "Rethinking classification and localization for object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 186–10 195.

[20] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 784–799.

[21] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia computer science*, vol. 199, pp. 1066–1073, 2022.

[22] M. J. Shafiee, B. Chywl, F. Li, and A. Wong, "Fast yolo: A fast you only look once system for real-time embedded object detection in video," *arXiv preprint arXiv:1709.05943*, 2017.

[23] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

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

[25] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, and I. So Kweon, "Attentionnet: Aggregating weak directions for accurate object detection," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2659–2667.

[26] B. Koonce and B. Koonce, "Resnet 50," *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, pp. 63–72, 2021.

[27] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "Densenet: Implementing efficient convnet descriptor pyramids," *arXiv preprint arXiv:1404.1869*, 2014.

[28] D. Sinha and M. El-Sharkawy, "Thin mobilenet: An enhanced mobilenet architecture," in *2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)*. IEEE, 2019, pp. 0280–0285.

[29] H. Qassim, A. Verma, and D. Feinzimer, "Compressed residual-vgg16 cnn model for big data places image recognition," in *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*. IEEE, 2018, pp. 169–175.

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

[31] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 2019, pp. 2888–2897.

[32] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.

[33] Y. Ming, X. Meng, C. Fan, and H. Yu, "Deep learning for monocular depth estimation: A review," *Neurocomputing*, vol. 438, pp. 14–33, 2021.

[34] R. Dalai, N. Dalai, and K. K. Senapati, "An accurate volume estimation on single view object images by deep learning based depth map analysis and 3d reconstruction," *Multimedia Tools and Applications*, vol. 82, no. 18, pp. 28 235–28 258, 2023.

[35] P.-Y. Chen, A. H. Liu, Y.-C. Liu, and Y.-C. F. Wang, "Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation," in *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, 2019, pp. 2624–2632.

[36] M. Talib, A. Al-Noori, and J. Suad, "Yolov8-cab: Improved yolov8 for real-time object detection," *Karbala International Journal of Modern Science*, vol. 10, 01 2024.

[37] "Real-time Object Detection with YOLOv8," https://keylabs.ai/blog/real-time-object-detection-with-yolov8/.

[38] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 3, pp. 1623–1637, 2020.

[39] "Mask R-CNN: Efficient detection of objects in images," https://ai-scholar.tech/en/articles/computer-vision/Mask-R-CNN.

[40] K. H. G. G. P. Dollár, R. Girshick *et al.*, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[41] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, 2023.

[42] "Difference between YOLO and SSD," https://www.geeksforgeeks.org/difference-between-yolo-and-ssd/.

[43] T. Ahmad, Y. Ma, M. Yahya, B. Ahmad, S. Nazir, and A. u. Haq, "Object detection through modified yolo neural network," *Scientific Programming*, vol. 2020, pp. 1–10, 2020.