Object Detection-based Vehicle Classification and Speed Estimation

Submitted in Partial Fulfillment of the Requirements in
CPE313 Data Science Track Elective 3:
Advanced Machine Learning and Deep Learning

Submitted by
Gabrielle E. Dela Cruz
Jens Liam P. Vista

May 23 2025

Object Detection-based Vehicle Speed Estimation with Dynamic Traffic Regulation

Gabrielle E. Dela Cruz
Computer Engineering
Technological Institute of the Philippines
Quezon City
qgedelacruz@tip.edu.ph

Jens Liam P. Vista
Computer Engineering
Technological Institute of the Philippines
Quezon City
qjlpvista@tip.edu.ph

ABSTRACT

Overspeeding is one of the major causes for road fatality, with over 13,000 deaths recorded in 2023, with driving above the speed limit considering various road conditions is a dangerous habit. This project developed a computer-vision powered by deep learning techniques that can classify vehicle type, detect vehicle speed, and determine if a vehicle is overspeeding at the area of effect – providing a technological way of controlling the area by monitoring vehicle speed that varies also with the vehicles type and in accordance with the speed limit set by the government, especially if implementing new speed limits that could be due to changes of road conditions, making it preventive in road fatality. Implementing road monitoring mechanisms will have a big impact on lessening road accidents that lead to road fatality and many other factors too such as heavy traffic, road damages, property damages, etc. The observed performance of the model highlights the system's capability of providing integrity into security and safety of the drivers and also provide opportunity jobs to people. Future work will explore applications and scalability across diverse road conditions, the environment, and many other factors that could make the project more advanced.

Keywords: *deep learning, computer vision, model deployment, classification, speed detection*

I. Introduction

In modern times, the growing number of car owners entails the increased density of urban traffic congestion, thereby increasing the demand for intelligent transportation systems, both of which highlight the need for advanced road vehicle monitoring solutions. To ensure reliability, the real-time accuracy of vehicle speed estimation is a top concern in traffic management, reinforced by imposed speed regulations to also enhance road safety. While widely-used traditional methods such as inductive loop detectors and radar guns operate efficiently, the deployment of these technologies are regarded with high cost and limited scalability. It is these limitations that drive the shift towards computer vision-based approaches concentrating on the conjunction of modern public infrastructures and the integration of deep learning-based solutions.

Recent advances in deep learning-based solutions, particularly its capability to operate object detection methods, enable real-time recognition of vehicles from live video streams. The YOLO (You Only Look Once) model family has proven effectiveness in this domain. More recent iterations in the YOLO lineup and the modification of its architecture brings significant gains in vehicle detection and classification performance in complex urban road conditions, which posits its potential for the deployment of intelligent transportation systems [1].

It must be noted that deep learning is a practical choice for vision-based solutions regarding vehicle speed estimation. Additionally, its versatility to be integrated into networks such as Long Short-Term Memory (LSTM) betters its effectiveness by achieving favorable RMSE scores for different car classes monitored in real-time conditions [2]. Mathematical methods with the use of vanishing geometry point and its combination with 3D bounding boxes to improve the speed estimation accuracy of monocular cameras has been made plausible, putting deep learning-based solutions into various contexts [3].

Moreover, dynamic traffic regulation greatly benefits from accurate vehicle classification. In manning these vehicles of different classes (mainly Sedans and SUVs) in size, weight, and driving behavior, are ways of informing differentiated speed enforcement. These object detection systems can be improved by its extension to perform reliable vehicle type classification for catering specific traffic policies based on vehicle classes [1].

In this study, an investigation regarding a deep learning-based approach in constructing a framework integrating real-time objection detection and classification using the YOLOv11 was conducted alongside the feature of speed estimation. By the categorization of vehicles and its according speed estimation, the system supports dynamic traffic regulation through adaptation to vehicle type and required driving context. In the fusion of these elements, the created system complies to advancements of intelligent traffic monitoring and control that is low-cost, scalable, and adaptive.

II. Related Work

This section relays relevant literature in regards to vehicle detection and speed estimation using deep learning, alongside the dynamic traffic regulation based on vehicle type classification. Prior related works are pertinent for the development of an effective intelligent traffic monitoring system based on speed estimation by vehicle class.

A. Object Detection for Vehicle Recognition

An effective foundation for intelligent vehicle monitoring systems are built by object detection systems. The YOLO (You Only Look Once) model suite is a widely used set for real-time object detection due to its reliability in balancing speed and accuracy. These highly-modifiable kit models were able to have been introduced with a Flip-Mosaic data enhancement algorithm to enhance the accuracy of vehicle detection in areas of varying traffic conditions and reduce false detection rates [4].

More advancements and modifications have been made to tailor YOLO models for urban environments of varying complexities. For instance, the improvisation of YOLOv5 in terms of vehicle detection and accuracy under challenging road and environment conditions were observed by the integration of BiFPN for robust multi-scale feature fusion to boost detection in low-visibility areas which provides significant performance gains over the use of baseline models [5].

B. Deep Learning-based Speed Estimation

In traffic management and enforcement, the accuracy of vehicle speed estimation is vital to relay metrics that accord to traffic policies. The reliance of traditional methods and its drawbacks are often overlooked, usually by its operation using specialized hardware proving its cost and inept scalability, whereas recents studies have explored better vision-based approaches by leveraging the capabilities of deep learning techniques.

With its versatility, any model from the YOLO family, like the YOLOv4, can be combined with other techniques such as DeepSORT for better tracking of vehicle speed along highways, achieving over 90% of accuracy and practical processing speeds for real-time deployment [6]. More granular approaches such as transformation of live video data into a coordinate system ensures the derivation of real-world distances and subsequent speed estimations [7].

C. Dynamic Traffic Regulation by Vehicle Classification

The differentiation between passing vehicle types such as the Sedan and SUV are imperative in implementing dynamic traffic regulations to tailor enforcement strategies, thereby improving traffic flow and safety on the road. The improving iterations of YOLO models continues to blur the challenges of occluded vehicle detection, serving as a clear demonstration of the algorithm's effectiveness in complex scenarios and its potential applications in intelligent transportation systems [8]. Additionally, enhancing vehicle localization accuracy of YOLO models such as YOLOv5 through the addition of attention mechanisms and improved loss functions were observed to also improve the positioning in various road scenes [9].

III. Methods

This section provides the structured procedure of the study that abides by standard academic practices and the incorporation of relevant references to support the methodological choices.

A. Dataset Preparation and Preprocessing

Roboflow Dataset Composition (used Grounding DINO)

SET	Total Images	Sedan	SUV	Augmentation Techniques (in Roboflow):
Training	1960	968	992	- Image resizing to 640x640 pixels
Validation	419	207	212	- Horizontal Flips
Testing	420	207	213	- Brightness/Contrast Shifting - Normalization
Total	**2799**	**1382**	**1417**	

Table 1. Detailed Structure of the Dataset

In this paper, the dataset was initialized by the addition of various images consisting of Sedan and SUVs prepared using the Auto Label feature in Roboflow, leveraging their proprietary Grounding DINO model for object detection. Such approach allows the automatic labeling of the images without the manual labor for labeling, further accelerating the dataset preparation process [10]. Referring to Table 1, a strict number of 2000 gathered images were manually selected from the Stanford Cars dataset, inclusive of the Keras collection of datasets, with the addition of images from external sources (captured photos and sample photos for public use) as the rest. The Data Augmentations that were already applied in the dataset helped increase dataset diversity to improve generalization, using techniques such as random flips, brightness adjustments, and scaling to prevent overfitting to prevent memorization of fixed patterns.

The preference towards the use of Grounding DINO as an object detector is its integration with language and vision modalities, making it possible to detect objects in images simply by textual prompts inputs and their descriptions to make classes such as Sedan and SUV [11]. Several studies have relayed the combination of auto-generated labels made by Grounding DINO with manual annotations and image transformations can enhance accuracy in object detection performance [12].

After the uploading of images to a Roboflow project, the dataset was made and processed with labeling, with the annotated version downloaded locally by preceding installation of Roboflow's Python SDK, a pathway to accessing the dataset online, which is then authenticated via an API key from the test account. This procedure of dataset acquisition follows the official documentation from Roboflow regarding dataset processing.

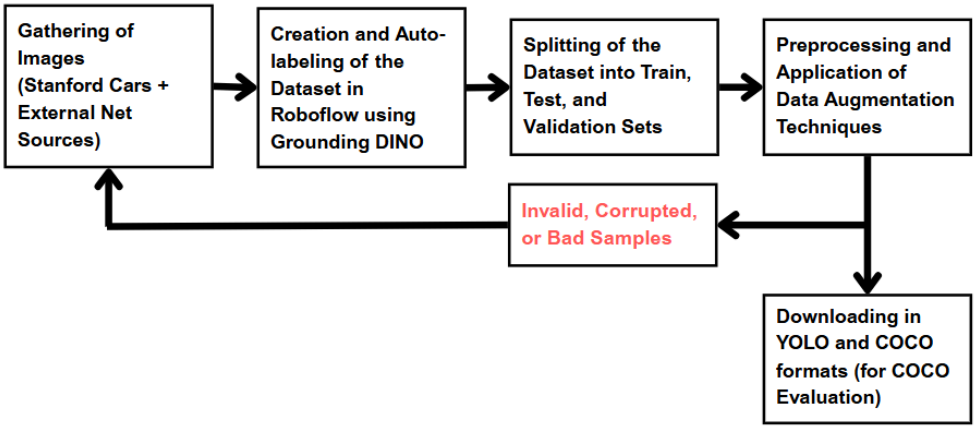

Figure 1. Dataset Creation Process with Roboflow and its Formats (.txt and .JSON)

The resulting dataset is split into 70:20:10 (Test:Val:Train) ratio with the images contained into training, validation, and testing subset folders created automatically by the SDK installation, ensuring a structured approach for model development and evaluation. Referring to the subsequent processes of Figure 1, this procedure was observed as the foundation for training our object detection model in the latter phases of the study.

B. Deployment and Training of Object Detection Models

In this paper, the deployment of three models with their distinctive features were considered to distinguish their strengths and weaknesses, with the best-performing model to be deployed in Streamlit, namely: YOLOv11, Faster R-CNN with ResNet50, and SSDLite with MobileNetV3 Large in the classification of vehicle types Sedan and SUV. Each of these models were trained on the custom-labeled dataset from Roboflow.

Within the YOLO family of models, the YOLOv11 is the latest iteration that introduces more technicalities that brings more architectural elements such as the C3k2 block, Convolutional block with Parallel Spatial Attention (C2PSA), and Spatial Pyramid Pooling - Fast (SPPF) which collectively improve object detection accuracy and improves feature extraction. Studies have highlighted the improved performance of YOLOv11 in various applications compared to previous iterations in the series, outperforming the older models in accuracy and computation efficiency [13]. Additionally, in projects operational for months, the robustness of YOLOv11 shows in its effectiveness in multi-category detection and adaptability in diverse scenarios [14]. For training, the Ultralytics CLI was used with epochs at 3 (suitable for lower-end devices) with image sizing set to 640 pixels, streamlining the process by handling the model optimization, dataset parsing and its augmentation.

For balancing computational efficiency and accuracy, the deployment of Faster R-CNN with ResNet50 was considered, a lightweight backbone model optimized with its NetAdapt algorithm and the use of neural architecture search which is ideal for mobile applications [15]. The customization of Faster R-CNN involved the replacement of its classification head in handling three classes – the background, sedan, and SUV. The training for this model was conducted over 10 epochs with a batch size of 4 to minimize resource taxing, using Stochastic Gradient Descent (SGD), with a learning rate of 0.005, for a momentum of 0.9, and weight decay of 0.0005. Additionally, a custom evaluation function was created to calculate for batch-wise accuracy by Intersection over Union (IoU) thresholds, a practice observed in applications of vehicle detection [4]. This is supported by Region Proposal Network (RPN) of Faster R-CNN and backbone adaptability widely recognized for producing results of strong accuracy for lightweight applications.

Considering resource-constrained devices requiring best suitability, SSDLite with MobileNetV3 Large relays a strong point in choice of model deployment. One of the variants of the Single Shot MultiBox Detector (SSD), the structure of SSDLite entails reduced model complexity by the utilization of linear bottlenecks and inverted residuals (also exists in MobileNetV2) for high-speed detection at low cost [16]. This model was trained with a similar structure to Faster R-CNN with consistency in data transformations and dataset preparation classes. The classification head of the model was adjusted to behave with the same three classes (the background, sedan, and SUV), involving loss computation and backpropagation cycles.

By the testing of these models, their respective strengths were determined: YOLOv11 exhibits the better performance but heavily penalizes system resources, Faster R-CNN with lesser accuracy and robustness but with consideration to system resources, and SSDLite maximizes on computational efficiency but with the least accuracy. The comprehensive training and evaluation of these models will help in the creation of systems suitable to their applications with scalability and adaptability for real-world intelligent traffic monitoring devices.

C. Speed-Detection and Vehicle Classification

The methodology of this research focuses on integrating real-time object detection, object tracking, spatial filtering, and temporal analysis to build a system capable of monitoring vehicle speeds and detecting speed violations based on vehicle classification. The goal is to develop an intelligent surveillance system that accurately classifies vehicle types, computes their speed over a known distance, and identifies those exceeding pre-defined speed limits for their category.

The process begins with the initialization of a custom-trained object detection model built upon the YOLOv8 architecture. This model, saved as best.pt, was the optimal version obtained during the training and deployment stages. It has been fine-tuned to identify specific types of vehicles, particularly SUVs and Sedans, using class IDs defined within the model. A confidence threshold of 0.05 was applied to filter out low-confidence detections, ensuring that only high-probability predictions are retained for further analysis. Once initialized, the model is used to process each frame of a video input, detecting vehicles in real-time.

To track these detected objects across multiple frames, the Simple Online and Realtime Tracking (SORT) algorithm is employed. The SORT tracker maintains the identity of objects using a combination of bounding box positions and predictive motion models. For this study, the SORT parameters were specifically set with max_age equal to 30, min_hits equal to 3, and an iou_threshold of 0.3. These values were carefully selected to balance track persistence and accuracy, allowing the system to retain identities over temporary occlusions or missed detections while minimizing ID switching and false matches. Each vehicle is assigned a unique ID that remains consistent throughout its presence in the frame, which is essential for accurate timing and speed calculations.

To ensure that only relevant vehicles are processed, the area of interest within each frame is defined using a polygonal region which is the green box in the figure 2. This polygon, constructed using specific coordinates, delineates the traffic lane or segment under observation. The Supervision library's PolygonZone module is utilized to determine whether detected objects fall within this region. By filtering detections based on their spatial presence inside the polygon, the system avoids unnecessary computation and misclassification caused by objects outside the intended monitoring zone, such as parked vehicles, those in adjacent lanes, or non-vehicle entities.

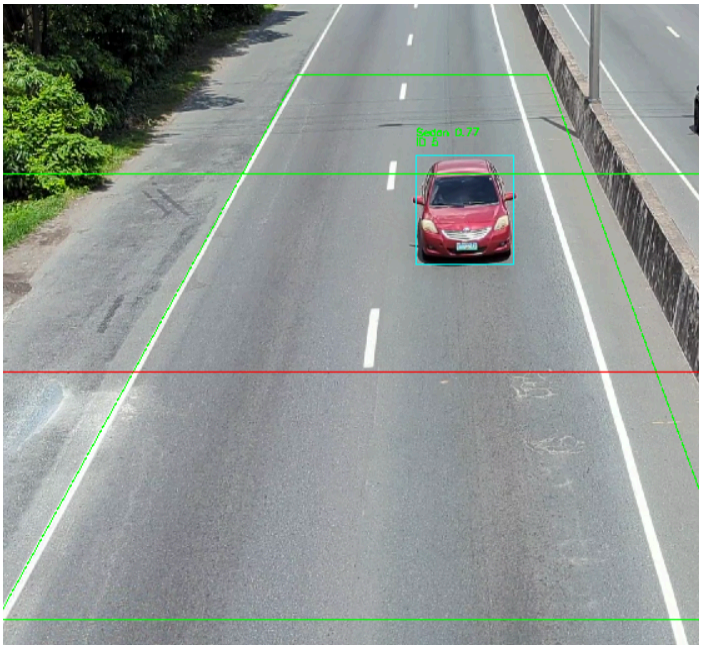

Figure 2. Speed Detection Monitoring Mechanism using YOLOv11

The key aspect of speed computation relies on temporal analysis using two reference lines, defined horizontally across the video frame at known Y-coordinates. These lines, referred

to as Point A and Point B, represent a fixed physical distance of 100 meters in the real world. As a vehicle moves through the defined polygonal zone, the system captures the exact timestamps at which the vehicle's bounding box crosses Points A and B. This time difference is used to calculate the speed using the equation Speed = Distance / Time. To convert this value into kilometers per hour, a factor of 3.6 is applied to the result. This process requires accurate temporal logging and ID tracking to ensure that each measurement corresponds precisely to the same object.

Vehicle IDs are remapped starting from 1 to simplify annotation and improve human readability during result analysis. Timing data and computed speeds are stored using dictionaries that map each tracking ID to its respective entry and exit times at Points A and B. These speed values are continually updated and referenced throughout the vehicle's presence in the frame.

Speed violation detection is another critical component of the system. Each vehicle type can be associated with a speed threshold, and if the computed speed exceeds this threshold, the system flags the vehicle as violating the limit. In such cases, a visual warning appears on the video frame, indicating "OVERSPEEDING," and the bounding box surrounding the vehicle changes color to red. This visual feedback mechanism serves both as an alerting system and as a means of post-event documentation for traffic authorities.

For clarity and interpretability, all analytical overlays are rendered directly onto the video frames. This includes bounding boxes around detected vehicles, textual annotations showing class labels and confidence scores, speed values next to their corresponding vehicles, visual indicators of Points A and B, and the polygonal boundary of the monitored area. These annotated frames are then written to a new output video file, which serves as the final output for visualization and further review.

Through this methodology, the system achieves real-time vehicle classification and speed estimation while maintaining robust identity tracking. The combination of deep learning, tracking algorithms, spatial and temporal filtering, and visual annotation results in a practical and scalable framework for traffic monitoring and road safety enforcement.

D. Deployment of the Model on Streamlit

After the training and saving the best run of the model, the deployment on Streamlit is planned. In preparation, a Github Repository for each component will serve as the foundation for the Streamlit web application, including necessary files such as the main Python source file, instruction text file that details the needed installation of modules and dependencies, and the saved best run of the YOLOv11 model. The preparation has been processed by uploading these files to the Github repositories and redirecting Streamlit to look for app.py in main branch commits and the files that follow.

IV. Results and Discussion

This section provides the outline of the performance evaluation of three deployed models, with emphasis on the deployed application model – YOLOv11, then Faster R-CNN with ResNet50, and SSDLite with MobileNetV3. Their convergent and complementary strengths in object detection accuracy, robustness to classification of vehicle types, and their operation under various resource constraints were deeply examined with COCO-style evaluation by having a copy of image annotations in JSON format.

A. Object Detection Model Performance Evaluation

The YOLOv11 model scaled at its Nano version was trained with 3 epochs at 640x640 with the specification of the model structure, backbone and detection head provided by the .yaml file from Roboflow, the annotations in .txt format in folders. With the YOLOv11 training results, the performance metrics were simplified thanks to Ultralytics's implementations, highlighting the steady convergence with box loss declining from the ranges of 0.325 to 0.313 and the classification loss around 0.79, mirroring the performance of YOLOv11's modules in loss reduction and high mAP yield on benchmarks [17]. Furthermore, it has been observed that loss trends and inference speed net of 3.8 images/second is a strong indication of trade-off between precision and throughput aligning towards Ultralytic's real-time detection claims.

YOLOv11 Model Evaluation Metrics

	Instances	True Positives	Precision	Recall	mAP@0.50	mAP@0.5:0.95
All	420	419	0.925	0.915	0.971	0.908
Sedan	212	212	0.895	0.923	0.969	0.896
SUV	207	207	0.954	0.907	0.974	0.920

Table 2. Performance Metrics of the YOLOv11 Nano Model

Referring to Table 2, the model main evaluation table (not using COCO) for YOLOv11 demonstrates its outstanding performance in detecting and classifying vehicles into classes sedan and SUV by having competitive results in its lightweight design. With an overall precision score of 92.50% and Recall of 91.50%, it balances object detection accuracy with minimal false positives or negatives effectively. With the mean Average Precision (mAP) of 97.1% at IoU 0.5 and 90.8% when it's at a stricter range of IoU from 0.5-0.95, the model achieves high localization accuracy, further reflecting the model's robustness in the alignment of tighter bounding boxes and detection, supported by class-specific performances with the Sedan class achieving a high precision of 95.4% and the SUV class with a strong Recall of 92.3%.

These printed metrics underline a notable performance for a Nano variant of a YOLO iteration which is a set of models usually optimized for real-time inference for lower-end devices and slower computation power. This is made possible with the original YOLO framework aimed as a fast alternative to region-based detectors the likes of Faster R-CNN, thereby processing images in one pass without sacrificing considerable accuracy. Furthermore, this is indicative of the strong performance the YOLOv11 offers that even lightweight models outperform heavier and more complex ones like SSD and Faster R-CNN if tuned and accustomed properly to the dataset. Thus, the deployment of this model is a practical choice for vehicle detections in applications that involve autonomous driving.

Faster R-CNN ResNet50 vs SSDLite MobileNetV3 vs YOLOv11 (COCO) Performance Comparison

Metrics	Faster R-CNN (ResNet50)	SSDLite (MobileNetV3)	YOLOv11
AP@ [0.50:0.95] (mAP)	0.936	0.883	0.908
AP@0.50	0.984	0.981	0.971
AP@0.75	0.983	0.948	0.960
AP (Large Objects)	0.936	0.883	0.900
AR (Large Objects)	0.957	0.909	0.949
AR@1 (Recall, maxDets=1)	0.957	0.907	0.938
AR@10 (Recall, maxDets=10)	0.957	0.908	0.949
AR@100 (Recall, maxDets=100)	0.957	0.909	0.949
COCO-style mAP	0.9356	0.8826	0.900

Table 3. Performance Metrics for Faster R-CNN, SSDLite, YOLOv11

In reference to Table 3, following the YOLOV11 model is the configuration of Faster R-CNN with ResNet50 as its backbone model, achieving an impressive 93.56% Mean Average Precision (mAP) with the Average Precision (AP) at IoU 0.5 score of 0.984, identical to IoU 0.75 at 0.983, indicative of high precision values in these restrictive thresholds, a testament to superb

localization accuracy and robust bounding box refinement. Moreover, its reliability in diverse test conditions is observed at how it retains AP and AR 0.936 and 0.957, respectively.

Despite the architecture of SSDLite paired with MobileNetV3 being lightweight yet with robust object detection performance, yielding an mAP of 88.26%, AP at IoU 0.5 score of 0.981 and AP at IoU 0.75 score of 0.948, slightly below that of Faster R-CNN with ResNet50 performance but notable due to it being a single-shot detector optimized for speed. Onwards, the Average Recall remained above 90% at maxDets=100, realistic of the model performance towards larger vehicle targets, mirroring MobileNet-based SSDLite architecture in real-time traffic on embedded systems that have recorded comparable AP values at ultra-low latency [19]. The architecture of SSDLite making use of MobileNetV3's inverted residuals and linear bottlenecks enables it to perform efficient inference without sacrificing considerable accuracy [16].

Computational Time Differences among YOLOv11, Faster R-CNN, and SSDLite				
Model | Workload	**Workload A**	**Workload B**	**Workload C**	**Sample Video (4K, 24 FPS)**
YOLOv11 Nano	15 secs	17 secs	19 secs	1 minute, 49 secs
Faster R-CNN with ResNet50	3 secs	4 secs	4 secs	3 minutes, 30 secs
SSDLite with MobileNetV3	1 secs	2 secs	3 secs	2 minutes, 42 secs

Table 4. Testing Inference Speeds of the Models under variable workloads

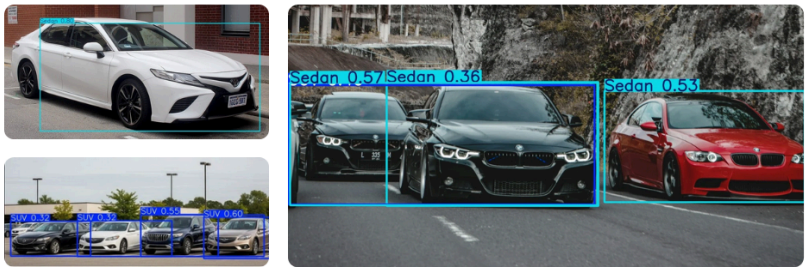

Figure 3. Sampled Images for Testing (Workloads A, B, C)

To test the real-world use practicality of the trained models, the computational time to process the inference of individual images or video frames were determined. In Table 4, the trained models were used to conduct image inferences under three different workloads: Workload A, Workload B, and Workload C, along with a sample 4K video running at 24 frames per second. The different types of Workload images involved more challenging tasks – from Workload A having one car, Workload B having three cars, and Workload C having four cars to be identified. Furthermore, the sample video subject is described to be a well-lit township environment in a road intersection with lots of moving traffic and people, proving to be challenges in the detection and correct classification of objects within the video frames. It must be noted that the computational times for these models are relative to system performance; thereby, a system with faster processing speed will conduct inference faster than lower-end systems – as such, individual testings vary. The goal of this testing is to measure the inferencing speed of models to better understand and deduce their underlying architectures and its mechanisms. Finally, the sampled images are displayed in Figure 3, with these gathered samples subject under free public use.

From the tallied results in Table 4, while the deployed YOLOv11 lagged behind the other models, it significantly showed faster video inferencing performance, crucial for the live monitoring of traffic situations. This demonstrates the efficiency of image inferencing by video frames with the YOLOv11 performance where the latter models struggled despite having faster individual image inferencing in the conducted tests. Since the SSDLite architecture specializes in edge devices, paired with the MobileNetV3 backbone model suitable for mobile applications, it showed promising results in following the video inferencing speed of YOLOv11. By comparison in these models, it is proven that the Faster R-CNN setup may be the most accurate but takes longer time, where individual testing still varies system by system.

V. Conclusion and Recommendations

The proposed design of the object-detection and speed estimation was successfully identifying vehicles and classifying them based on their vehicle type and also detecting the vehicle speed. Furthermore, It is successfully able to alert if the vehicle is overspeeding making it preventive in road accidents. The design can be improved by adding more machine learning techniques such as reading plate number and adding adjustment to speed limitation that varies on the road condition or weather. The design can generate opportunity and jobs to people such as people who will be alerted if someone is overspeeding and tell to the one that will accost the violator and many more.

VI. Acknowledgements

The authors would like to express their sincere gratitude to their course instructor, Engr. Roman R. Richard, whose guidance and strict supervision with his given constructive feedback were a key part in making his contribution invaluable. His expertise in the domain of advanced machine learning and deep learning has allowed students of prospering intellect be shaped and directed to a path towards dedicated learning that makes in-depth research suchlike this. With this, we also extend our thanks to the learning opportunities provided in his course CPE 313 - Advanced Machine Learning and Deep Learning which paved the foundation for the conceptualization, implementation, and realization of this study.

VII. References

[1] Hao, C., Li, Z., Wang, H., & Sun, L. (2023). Improved YOLOv5 Model for Vehicle Detection in Complex Road Environments. Electronics, 12(6), 1323. https://doi.org/10.3390/electronics12061323

[2] Khanal, R., Aryal, S., & Pathak, R. (2024). Vehicle Speed Prediction Using YOLO and LSTM. Kathmandu University Journal of Engineering and Management, 3(1). https://doi.org/10.3126/kjem.v3i1.74702

[3] Miao, J., Liu, X., & Yuan, Y. (2024). Monocular Vehicle Speed Estimation via Vanishing Point Geometry and Temporal Tracking. arXiv preprint arXiv:2505.01203. https://arxiv.org/abs/2505.01203

[4] Zhang, Y., Guo, Z., Wu, J., Tian, Y., Tang, H., & Guo, X. (2022). Real-Time Vehicle Detection Based on Improved YOLOv5. MDPI - Sustainability. https://doi.org/10.3390/su141912274

[5] Liu, S. (2025). Vehicle detection in different traffic scenarios based on YOLOv5. Proc. SPIE 13521, International Conference on Computer Vision and Image Processing (CVIP 2024), 135210V (17 January 2025). https://doi.org/10.1117/12.3057990

[6] Nguyen, P.H., & Duy, M.B. (2023). An algorithm using YOLOv4 and DeepSORT for tracking vehicle speed on highway. Indonesian Journal of Electrical Engineering and Informatics (IJEEI). https://section.iaesonline.com/index.php/IJEEI/article/view/3448

[7] Bell, D., Xiao, W., & James, P. (2020). ACCURATE VEHICLE SPEED ESTIMATION FROM MONOCULAR CAMERA FOOTAGE Accurate Vehicle Speed Estimation From Monocular Camera Footage. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. https://doi.org/10.5194/isprs-annals-V-2-2020-419-2020

[8] Zhou, L. (2024). Multi-scenario Vehicle Detection Based on the YOLOv5 Algorithm. Transactions on Computer Science and Intelligent Systems Research, 6, pp. 103–110. https://doi.org/10.62051/mbgg8t65

[9] Zhang, Y., Gong, Y., & Chen, X. (2024). Research on YOLOv5 Vehicle Detection and Positioning System Based on Binocular Vision. World Electric Vehicle Journal. 2024; 15(2):62. https://doi.org/10.3390/wevj15020062

[10] Witt, J. (2024). Launch: Auto Label Images with Roboflow. Roboflow. https://blog.roboflow.com/launch-auto-label/?utm_source=chatgpt.com

[11] Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Jiang, Q., Li, C., Yang, J., Su, H., Zhu, J., & Zhang, L. (2023). Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection. arXiv:2303.05499. https://doi.org/10.48550/arXiv.2303.05499

[12] Son, J., & Jung, H. Teacher–Student Model Using Grounding DINO and You Only Look Once for Multi-Sensor-Based Object Detection. Appl. Sci. 2024, 14, 2232. https://doi.org/10.3390/app14062232

[13] Bento, J., Paixao, T., & Alvarez, A.B. (2025). Performance Evaluation of YOLOv8, YOLOv9, YOLOv10, and YOLOv11 for Stamp Detection in Scanned Documents. Appl. Sci. 2025, 15(6), 3154; https://doi.org/10.3390/app15063154

[14] Tian, Z., Yang, F., Yang, L., Wu, Y., Chen, J., & Qiang, P. (2025). An Optimized YOLOv11 Framework for the Efficient Multi-Category Defect Detection of Concrete Surface. Sensors 2025, 25(5), 1291; https://doi.org/10.3390/s25051291

[15] Howard, A., Sandler, M., Chu, G., Chen, Liang-Chieh, Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Quoc, & Adam, H. (2019). Searching for MobileNetV3. arXiv:1905.02244. https://doi.org/10.48550/arXiv.1905.02244

[16] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, Liang-Chieh (2018).MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381. https://doi.org/10.48550/arXiv.1801.04381

[17] Khanam, R., & Hussain, M. (2024). YOLOv11: An Overview of the Key Architectural Enhancements. arXiv:2410.17725. https://doi.org/10.48550/arXiv.2410.17725

[18] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017. https://ieeexplore.ieee.org/document/7485869/authors#authors

[19] Liu, T., Zhu, Y., & Yuan, F. (2022). Underwater Accompanying Robot Based on SSDLite Gesture Recognition. Appl. Sci. 2022, 12(18), 9131; https://doi.org/10.3390/app12189131

Evaluation Metrics

Student Outcome 7							
Criteria	**Ratings**						**Pts**
T.I.P. SO 7.1 Acquire and apply new knowledge from outside sources threshold: 4.2 pts	**6 pts** **[Excellent] Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently and applies knowledge learned into practice**	**5 pts** **[Good] Educational interests and pursuits exist and flourish outside classroom requirements,knowledge and/or experiences are pursued independently**	**4 pts** **[Satisfactory] Look beyond classroom requirements, showing interest in pursuing knowledge independently**	**3 pts** **[Unsatisfactory] Begins to look beyond classroom requirements, showing interest in pursuing knowledge independently**	**2 pts** **[Poor] Relies on classroom instruction only**	**1 pts** **[Very Poor] No initiative or interest in acquiring new knowledge**	6 pts
T.I.P. SO 7.2 Learn independently threshold: 4.2 pts	**6 pts** **[Excellent] Completes an assigned task independently and practices continuous improvement**	**5 pts** **[Good] Completes an assigned task without supervision or guidance**	**4 pts** **[Satisfactory] Requires minimal guidance to complete an assigned task**	**3 pts** **[Unsatisfactory] Requires detailed or step-by-step instructions to complete a task**	**2 pts** **[Poor] Shows little interest to complete a task independently**	**1 pts** **[Very Poor] No interest to complete a task independently**	6 pts
T.I.P. SO 7.3 Critical thinking in the broadest context of technological change threshold: 4.2 pts	**6 pts** **[Excellent] Synthesizes and integrates information from a variety of sources; formulates a clear and precise perspective; draws appropriate conclusions**	**5 pts** **[Good] Evaluate information from a variety of sources; formulates a clear and precise perspective.**	**4 pts** **[Satisfactory] Analyze information from a variety of sources; formulates a clear and precise perspective.**	**3 pts** **[Unsatisfactory] Apply the gathered information to formulate the problem**	**2 pts** **[Poor] Gather and summarized the information from a variety of sources but failed to formulate the problem**	**1 pts** **[Very Poor] Gather information from a variety of sources**	6 pts
T.I.P. SO 7.4 Creativity and adaptability to new and emerging technologies threshold: 4.2 pts	**6 pts** **[Excellent] Ideas are combined in original and creative ways in line with the new and emerging technology trends to solve a problem or address an issue.**	**5 pts** **[Good] Ideas are creative and adapt the new knowledge to solve a problem or address an issue**	**4 pts** **[Satisfactory] Ideas are creative in solving a problem, or address an issue**	**3 pts** **[Unsatisfactory] Shows some creative ways to solve the problem**	**2 pts** **[Poor] Shows initiative and attempt to develop creative ideas to solve the problem**	**1 pts** **[Very Poor] Ideas are copied or restated from the sources consulted**	6 pts
							Total Points: 24

Evaluated by:

_____

Engr. Roman M. Richard
Course Instructor