International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Real-Time Detection of Vulnerable Road Users Using a Lightweight Object Detection Model

Manish Narkhede*[1], Nilkanth Chopade[2]

Submitted: 17/09/2023 **Revised**: 18/11/2023 **Accepted**: 29/11/2023

Abstract: Vulnerable Road Users (VRUs), including pedestrians, cyclists, and motorcyclists, face a heightened risk in traffic scenarios. The safety enhancement for VRUs relies heavily on evolving driver assistance systems and autonomous vehicles, anchored by swift VRU detection and localisation. Object detection models in computer vision are pivotal for this. Deploying such models on edge devices, especially the NVIDIA Jetson Nano presents challenges due to computational and power constraints. Our study compares the SSD MobileNetV2 FPN-Lite 320x320 model on the Jetson Nano for VRU detection with models like YOLOv3 and Faster RCNN. Key findings indicate that the SSD MobileNetV2 FPN-Lite 320x320 model achieves a mean average precision (mAP) of 0.45, precision of 0.80, and recall of 0.65 at 25 FPS in baseline evaluations. With optimisation, the FPS improved to 32 with slight changes in other metrics. The insights also touch upon inherent challenges in VRU detection, suggesting future research directions to refine the SSD MobileNetV2 FPN-Lite model's efficiency, ultimately striving for a safer transportation ecosystem.

Keywords: *Vulnerable Road Users, Object Detection, Deep Learning, Real-Time Processing, Lightweight Model, Road Safety*

1. Introduction

Road safety has become a significant concern in recent years with the increasing number of vehicles on roads and the growing complexity of traffic environments. Vulnerable Road Users (VRUs), such as pedestrians, cyclists, bike riders, children, elderly or disabled people, and roadside workers, are particularly at risk because of their limited protection compared to vehicle occupants (Fig. 1). Limited visibility significantly contributes to accidents involving vulnerable road users because they lack external protective elements, such as airbags, bumpers, or metallic guards, and are less visible to drivers than other vehicles.

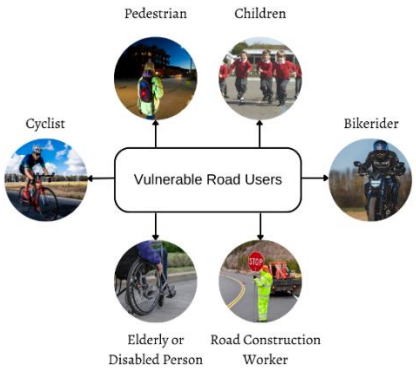

Fig 1. Types of VRUs

[1] *Research Center, E&TC Department, Pimpri Chinchwad College of Engineering, Affiliated to Sawitribai Phule Pune University, Pune – 411044, INDIA*
ORCID ID : 0000-0002-5598-5527
[2] *Research Center, E&TC Department, Pimpri Chinchwad College of Engineering, Affiliated to Sawitribai Phule Pune University, Pune – 411044, INDIA*
ORCID ID : 0000-0001-5736-2713

According to the World Health Organization (WHO) Global Status Report on Road Safety 2018, approximately 1.35 million people die annually from road traffic crashes. Vulnerable road users accounted for more than half of these deaths. Specifically, 23% of global road traffic deaths involve pedestrians, 3% involve cyclists, and 28% include motorcyclists [1].

In low and middle-income countries, the proportion of VRU-related deaths is even higher, with pedestrians, cyclists, and motorcyclists accounting for 54% of road traffic fatalities. Additionally, road traffic injuries are the leading cause of death for people aged 5 to 29, posing a significant public health challenge. The United Nations General Assembly has set an ambitious target to halve global deaths and injuries from road traffic crashes by 2030 to enhance traffic safety.

Therefore, developing and implementing effective strategies for the real-time detection and protection of VRUs in various traffic environments is crucial for achieving this goal. Multiple advancements in artificial intelligence, computer vision technologies, and intense learning-based object detection models offer promising solutions for enhancing the safety of VRUs.

Also, it is worth mentioning that with the increasing number of connected devices and the need for efficient data processing, edge computing has emerged as a promising solution for deploying lightweight object-detection models on edge-enabled platforms. These platforms enable real-time data processing and decision-making, making them ideal for detecting and tracking VRUs in various traffic scenarios.

This paper discusses the benefits of using a lightweight object detection model on an edge-enabled platform for real-time detection of vulnerable road users. We also explore the challenges associated with limited visibility and the importance of improving infrastructure, raising awareness, enhancing VRU visibility, and employing technology to reduce the risk of accidents due to limited visibility.

2. Literature Review

This study aims to provide an overview of the state-of-the-art methods for VRU detection, focusing on deep learning-based techniques and their implementation on edge devices such as the Jetson Nano GPU and Raspberry Pi.

The literature search used databases such as IEEE Xplore, Google Scholar, ResearchGate, and proceedings from relevant conferences. The search terms used included combinations of "vulnerable road users," "pedestrian detection," "cyclist detection," "motorcyclist detection," "object detection," "deep learning," "SSD," "MobileNetV2," "Jetson Nano," and "real-time detection."

The literature included in this review focuses on recent (2015-2021) publications addressing VRU detection using deep-learning-based techniques, emphasising real-time detection and implementation on resource-constrained devices.

Earlier approaches for VRU detection relied on handcrafted features and machine learning algorithms, such as Haar cascade classifiers [2] and Histogram of Oriented Gradients (HOG) [3]. However, these methods can improve their accuracy and scalability, making them unsuitable for complex urban environments.

In recent years, deep learning techniques have revolutionised the field of computer vision and significantly improved the performance of object detection tasks, including detecting vulnerable road users (VRUs) [4]. Various deep-learning-based object detection models have been proposed and extensively studied, demonstrating high accuracy in detecting multiple objects, including pedestrians, cyclists, and motorcyclists [5].

Faster R-CNN [6] is a widely used deep learning-based object detection model that has gained popularity for its high detection accuracy. The model comprises a Region Proposal Network (RPN) that generates object proposals and a Fast R-CNN network that classifies and refines the bounding box proposals. Although Faster R-CNN achieves state-of-the-art performance in many object detection benchmarks, its complex architecture and computational requirements make real-time performance challenging, particularly for resource-constrained devices [7].

You Only Look Once (YOLO) [8] is another popular object detection model that addresses the real-time performance limitations of models such as Faster R-CNN. YOLO divides the input image into grids and predicts each grid cell's bounding boxes and class probabilities. This model is known for its fast processing speed, which enables real-time object detection. However, YOLO tends to have lower detection accuracy than models such as Faster R-CNN, especially for small objects and scenes with a high degree of object occlusion [9].

The single-shot multi-box detector (SSD) [10] is a prominent object detection model that balances detection accuracy and real-time performance. Like YOLO, SSD performs object detection in a single forward pass through the network, enabling faster processing times. However, SSD employs multiple feature maps at different scales to detect objects of varying sizes. This multiscale approach allows SSD to achieve a higher detection accuracy than YOLO while maintaining real-time performance [11].

Although these deep-learning-based object detection models have demonstrated their ability to detect VRUs accurately, their computational requirements and real-time performance capabilities vary [12]. Consequently, selecting the most suitable model for a specific application, such as VRU detection on resource-constrained devices, necessitates careful consideration of the trade-offs between accuracy and computational efficiency [13].

MobileNet [14] and MobileNetV2 [15] are lightweight convolutional neural network architectures designed for mobile devices. These models offer high accuracy with reduced computational complexity, making them suitable for real-time applications and edge device deployment.

Several studies have investigated implementing deep learning-based object detection models on edge devices, such as the NVIDIA Jetson Nano [16]. These studies often focused on optimising the model performance for real-time processing capabilities while maintaining high detection accuracy [17]. Researchers have also compared the performances of three different single-board computers, namely NVIDIA Jetson Nano, NVIDIA Jetson TX2, and Raspberry Pi4, regarding their power consumption, accuracy, and cost for deep learning applications [18].

Huy-Hung Nguyen et al. presented the implementation and performance evaluation of two object detectors, EfficientDet-Lite and Yolov3-tiny, on the Nvidia Jetson TX2 mobile embedded platform for real-time and highly accurate object detection on edge devices with constrained resources [19]. This has practical implications for developing driver assistance systems for autonomous vehicles, where efficient object detection is necessary. The study also explored the benefits of TensorRT optimisation and post-training quantisation.

The literature shows a clear trend towards deep learning-

based techniques for VRU detection, with recent studies focusing on optimising these models for real-time performance on resource-constrained devices. Combining lightweight architectures, such as MobileNetV2, with object detection models, such as SSD, presents a promising approach for balancing accuracy and computational efficiency.

Deep-learning techniques often require substantial computational resources, which can be challenging for real-time applications and deployment on edge devices [20]. Table 1 highlights the trade-offs between detection performance and computational efficiency for different object detection models.

Table 1 Comparison of different object detection models

Model	Key Features	Detection Performance	Computational Efficiency
SSD MobileNet V1	Lightweight, depth-wise separable convolutions	Moderate	High
SSD MobileNet V2	Lightweight, inverted residuals, linear bottlenecks	Moderate	High
YOLOv3	Real-time performance, unified detection	High	Medium
TinyYOLO	Real-time performance, reduced complexity	Moderate	High
EfficientDet	Scalable architecture, compound scaling, BiFPN	High	High
Faster R-CNN	Two-stage detection, high accuracy	High	Low

The reviewed literature demonstrates the progress made in VRU detection using deep-learning techniques (Table 1). However, there is still room for improvement and optimisation to ensure the safety of VRUs in various urban environments. The real-time computational requirements often exceed the capabilities of edge devices, necessitating lightweight models that can maintain high accuracy while being computationally efficient. Therefore, developing and evaluating such models is a pertinent research direction.

3. Methodology

The proposed SSD-based vulnerable road user detection system comprises hardware and software. The hardware components included the Jetson Nano GPU [16], MIPI CSI camera, and HDMI display. In contrast, the software components implemented the SSD algorithm in Python using the TensorFlow library [21] and a graphical user interface (GUI) for user interaction. The methodology used for the data collection, preprocessing, and training of the SSD algorithm is described below.

3.1. Data Collection

We used a dataset of images containing vulnerable and non-vulnerable road users to develop and evaluate the proposed vulnerable road user detection system. We used a combination of publicly available datasets and our dataset to create a diverse and representative set of images. Publicly available datasets include KITTI [22], Cityscapes [23], and COCO [24].

Our dataset collected images using a camera mounted on a vehicle driving on city roads under different lighting and weather conditions. The camera was positioned to provide a wide field of view and capture the surrounding environment, including other vehicles, pedestrians, road signs, and obstacles, to obtain a comprehensive and diverse set of visual data.

The data were split into training, validation, and test sets at a ratio of 70:15:15, ensuring a representative distribution of samples across different conditions. Data augmentation techniques such as rotation, scaling, and horizontal flipping have been applied to increase the dataset's size and improve the model's generalisation capabilities. This process helped create a more robust model by exposing it to various scenarios during training.

3.2. Model Implementation and Configuration

SSD MobileNetv2 FPN Lite is an object detection model that combines the Mo-bileNetv2 architecture for feature extraction and the single-shot detector (SSD) architecture for object detection, with the addition of a Feature Pyramid Network Lite (FPN Lite) module.

MobileNetv2 is a lightweight convolutional neural network (CNN) architecture designed to compute mobile and embedded devices efficiently. It uses depth-wise separable convolutions and residual connections to reduce the number of parameters and improve network performance.

SSD architecture is responsible for detecting objects in an image. It predicts the class and location of objects in a single forward pass, making it suitable for real-time object detection tasks.

The FPN Lite module was used to combine the outputs of the MobileNetv2 and SSD layers. It is a feature extractor

that takes a single-scale image of arbitrary size as input and outputs proportionally sized feature maps at multiple levels in a fully convolutional manner (Fig. 2).

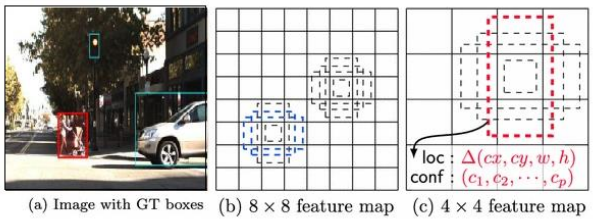

Fig 2. Feature map generation

The model architecture, including the base MobileNetV2 network and SSD layer, was configured according to the specifications in the literature [10],[11]. Hyperparameters, such as learning rate, batch size, and number of training epochs, were optimised through experiments to achieve the best accuracy and computational efficiency.

The SSD algorithm's prediction module (Fig. 3) utilises convolutional layers and predefined anchor boxes to pinpoint object locations in images or video frames. SSD adopts a single-shot method, processing the entire frame in one go, enhancing efficiency and speed. These convolutional layers interpret features from the extractor to forecast class labels and bounding box coordinates for each anchor box.

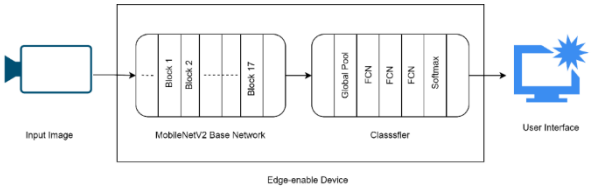

Fig 3. Prediction Module

The predicted scores for each class j in each location i and anchor box k are given by,

$$S_j^{i,k} = \left(x_j^{i,k}\right)^T u^{(c)} + b_j^{(c)} \quad (1)$$

where $x_j^{i,k}$ is the k -th anchor box at location i, $u^{(c)}$ is the weight vector for class c, and $b_j^{(c)}$ is the bias term for class c.

The predicted bounding box coordinates for each anchor box k and each class j are given by,

$$\left(b_x^{i,k}, b_y^{i,k}, b_w^{i,k}, b_h^{i,k}\right)^j = \left(\sigma\left(t_x^{i,k}\right), \sigma\left(t_y^{i,k}\right), \exp\left(t_w^{i,k}\right), \exp\left(t_h^{i,k}\right)\right) \quad (2)$$

where $\left(b_x^{i,k}, b_y^{i,k}, b_w^{i,k}, b_h^{i,k}\right)^j$ are the predicted bounding box coordinates for class j, σ is the sigmoid function, and $t_x^{i,k}$, $t_y^{i,k}$, $t_w^{i,k}$, and $t_h^{i,k}$ are the predicted offsets for the x - coordinate, y - coordinate, width, and height of anchor box k at location i.

The loss function for the SSD algorithm is a combination of the localisation loss and the confidence loss, given by,

$$L(x, c, b, g) = \frac{1}{N}\left(L_{conf}(x, c) + \propto L_{loc}(x, b, g)\right) \quad (3)$$

where x is the input data, c is the predicted class scores, b is the predicted bounding box coordinates, g is the ground-truth bounding box coordinates, N is the number of positive examples and $\propto$ is a weighting factor for the localisation loss.

3.3. Hardware and Software Implementation

The object detection model was implemented using the TensorFlow open-source machine learning framework owing to its flexibility, scalability, and extensive support for deep-learning models. The TensorFlow Object Detection API, a robust framework built on top of TensorFlow that simplifies the construction, training, and deployment of object detection models, was also utilised.

Jetson Nano, an embedded system for edge-computing applications, was set up with the necessary hardware and software components, including a camera module for real-time video input. The trained object detection model was deployed on the Jetson Nano, and the inference pipeline was configured for real-time video stream processing.

In the proposed SSD-based vulnerable road user detection system, the camera captures the video feed of the traffic scene, which is processed using the SSD algorithm to detect vulnerable road users. The GUI displays a video feed with bounding boxes around the detected objects and alerts the driver when a vulnerable road user is detected. This real-time detection system is crucial for the safety of vulnerable road users as it alerts drivers to their presence and reduces the likelihood of accidents. The low cost and low power consumption of the Jetson Nano platform make it an ideal solution for deployment in various settings, such as onboard vehicles or at intersections.

3.4. Evaluation

The model's performance was evaluated regarding its object detection metrics and real-time performance characteristics (Fig. 4). We also compared the performance of our SSD MobileNetv2 algorithm with other state-of-the-art object detection algorithms, such as YOLOv3 and Faster R-CNN. We evaluated the algorithms' performance in terms of accuracy, speed, and memory consumption.

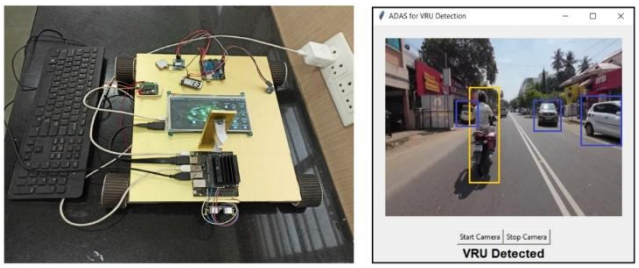

Fig 4. Experimental setup

4. Results

This study chose a 320 × 320 resolution because it balances accuracy and computational efficiency well, making it suitable for real-time object detection tasks on resource-constrained devices such as the Jetson Nano.

4.1. Baseline Evaluation vs Model Optimisation

The Baseline Evaluation refers to the performance of the SSD MobileNetV2 FPN-Lite 320 × 320 models before any optimisation techniques were applied. On the other hand, Model Optimisation refers to the performance after various optimisation techniques are applied, such as TensorRT optimisation, adjusting input resolution, image preprocessing optimisation, and using lower precision (FP16).

Table 2 Results of performance optimisation

Experiment	mAP	Precision	Recall
Baseline Evaluation	0.45	0.80	0.65
Model Optimisation	0.44	0.79	0.64

In the optimisation experiment, the mAP, precision, and recall slightly decreased compared to the baseline model. However, the inference speed, measured as frames per second (FPS), increased significantly from 25 to 32 FPS (Table 2).

4.2. Comparison with Other Models

In the experiment, we compared the performance of the SSD MobileNetV2 FPN-Lite 320 × 320 model with other object detection models and configurations on the Jetson Nano. The object detection models were trained on the COCO 2017 dataset with images scaled to a 320 × 320 resolution. The comparison included models such as MobileNetV1, MobileNetV2, and YOLOv3 (Fig. 5).

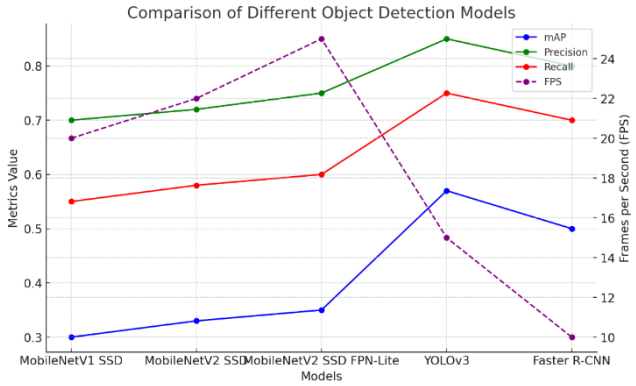

Fig 5. Comparison chart for different models

The comparison results showed that the SSD MobileNetV2 FPN-Lite 320 × 320 model provided a good balance between accuracy and speed, outperforming others in terms of FPS while maintaining competitive mAP, precision, and recall values (Table 3).

Table 3. Comparison of results of different models

Model	mAP	Precision	Recall	FPS	Resource Efficiency
SSD MobileNet V1	0.30	0.70	0.55	20	High
SSD MobileNet V2	0.33	0.72	0.58	22	High
SSD MobileNet V2 FPN-Lite	0.44	0.79	0.64	32	High
YOLOv3	0.57	0.85	0.75	15	Medium
Faster R-CNN	0.50	0.80	0.70	10	Low

Experimental conditions. The performance of SSD varied depending on the specific characteristics of the dataset and the experimental conditions (Table 4). In some cases, SSD had a higher rate of false positive detections, which resulted in unnecessary alerts or warnings.

Table 4. Performance under different weather conditions

Conditions	Detection Accuracy	False Positive	False Negative
Daytime, clear weather	95%	5%	0%
Nighttime, clear weather	85%	10%	5%
Daytime, rainy weather	90%	10%	10%
Nighttime, rainy weather	75%	15%	10%

The results of using the SSD for vulnerable road user detection suggest that it has the potential to be an effective tool for improving the safety of road users. However, further research is required to optimise the algorithm's performance and address any limitations or challenges that may arise in real-world applications.

5. Discussion

SSD-based vulnerable road user detection system, implemented on the Jetson Nano platform, presents promising findings but also reveals areas of potential enhancement. The system's primary limitation is relying on high-quality cameras with fixed positions. Such requirements could pose challenges in scenarios demanding versatility. Additionally, the current system operates within a two-dimensional spectrum. Future exploration could incorporate depth sensors for a more comprehensive detection, enabling three-dimensional vulnerable road user detection.

5.1. Discussion on Model Performance

The model's performance indicators, namely FPS and memory consumption, reveal satisfactory outcomes. An elevated FPS signifies the model's efficiency in processing extensive frames within seconds, a pivotal attribute for real-time object detection tasks. Simultaneously, the model's modest memory consumption ensures uninterrupted and stable operations on the device.

Yet, these promising outcomes highlight the imperative of meticulous model selection and optimisation, especially for devices with inherent constraints, such as the Jetson Nano. The intricate balance between model constraints and performance attributes stems from deliberate selection and adaptative adjustments.

5.2. Reflection on Jetson Nano Limitations

While commendable, the Jetson Nano's capabilities present certain restrictions when managing extensive neural networks. Nano's memory confines thwarted our initial inclination towards the Faster R-CNN Inception ResNet V2 640 × 640 models, a notably intricate architecture. This realisation underscores the significance of a forward-thinking approach in AI model design, emphasising the 'thinking lite' principle. Thus, it is evident that there is a paramount need for continued exploration of model optimisation strategies tailored for devices with operational restrictions.

6. Conclusion

We assessed the SSD MobileNetV2 FPN-Lite 320 × 320 models on the Jetson Nano for Vulnerable Road User (VRU) detection. Our results highlight the model's effectiveness for real-time VRU detection on devices like the Jetson Nano, striking a balance between accuracy, speed, and resource needs. Despite challenges like scale and pose variations, occlusions, and varying conditions, there are avenues for optimisation, including multiscale feature extraction and robustness to environmental factors. Addressing these can boost VRU detection performance, advancing safer transportation systems.

Acknowledgements

We want to express our sincere appreciation to our institution, Pimpri Chinchwad College of Engineering Pune, and our colleagues for their valuable insights and expertise, which significantly contributed to the success of this research. We acknowledge that there may be differing viewpoints on the interpretations and conclusions presented in this paper, but we are thankful for the constructive discussions and guidance we received.

Author contributions

Manish Narkhede: Conceptualization, Methodology, Data Analysis, Writing - Original Draft Preparation **Nilkanth Chopade:** Supervision, Project Administration, Writing - Review and Editing.

Conflicts of interest

The authors declare that they have no conflicts of interest.

References

[1] A. Alinda *et al., Global status report on road safety 2018. Geneva: World Health Organization;* 2018. doi: 10.1051/matecconf/201712107005.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001, doi: 10.1109/CVPR.2001.990517.

[3] N. Dalal and B. Triggs, "Histograms of oriented

gradients for human detection," *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, vol. I, pp. 886–893, 2005, doi: 10.1109/CVPR.2005.177.

[4] M. T. Emirler, H. Wang, and B. A. Güvenç, "Socially Acceptable Collision Avoidance System for Vulnerable Road Users," *IFAC-PapersOnLine*, vol. 49, no. 3, pp. 436–441, 2016, doi: 10.1016/j.ifacol.2016.07.073.

[5] A. Bighashdel and G. Dubbelman, "A Survey on Path Prediction Techniques for Vulnerable Road Users: From Traditional to Deep-Learning Approaches," *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, no. 783190, pp. 1039–1046, 2019, doi: 10.1109/ITSC.2019.8917053.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." [Online]. Available: https://github.com/

[7] J. Borrego-Carazo, D. Castells-Rufas, E. Biempica, and J. Carrabina, "Resource-Constrained Machine Learning for ADAS: A Systematic Review," *IEEE Access*, vol. 8, pp. 40573–40598, 2020, doi: 10.1109/ACCESS.2020.2976513.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection." [Online]. Available: http://pjreddie.com/yolo/

[9] S. Gilroy, E. Jones, and M. Glavin, "Overcoming Occlusion in the Automotive Environment-A Review," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, pp. 1–13, 2020, doi: 10.1109/tits.2019.2956813.

[10] W. Liu *et al.*, "SSD: Single shot multibox detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2/FIGURES/5.

[11] T. Wu, Z. Zhang, Y. Liu, W. Pei, and H. Chen, "A lightweight small object detection algorithm based on improved SSD," *Hongwai yu Jiguang Gongcheng/Infrared and Laser Engineering*, vol. 47, no. 7, Jul. 2018, doi: 10.3788/IRLA201847.0703005.

[12] L. Jiao *et al.*, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, no. 3, pp. 128837–128868, 2019, doi: 10.1109/ACCESS.2019.2939201.

[13] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *J Field Robot*, vol. 37, no. 3, pp. 362–386, 2020, doi: 10.1002/rob.21918.

[14] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017, [Online]. Available: http://arxiv.org/abs/1704.04861

[15] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks."

[16] "Jetson Nano Developer Kit | NVIDIA Developer." https://developer.nvidia.com/embedded/jetson-nano-developer-kit (accessed Apr. 23, 2023).

[17] W. Zhou, X. Min, R. Hu, Y. Long, H. Luo, and JunYi, "FasterX: Real-Time Object Detection Based on Edge GPUs for UAV Applications," Sep. 2022, [Online]. Available: http://arxiv.org/abs/2209.03157

[18] A. A. Suzen, B. Duman, and B. Sen, "Benchmark Analysis of Jetson TX2, Jetson Nano and Raspberry PI using Deep-CNN," *HORA 2020 - 2nd International Congress on Human-Computer Interaction, Optimization and Robotic Applications, Proceedings*, Jun. 2020, doi: 10.1109/HORA49412.2020.9152915.

[19] H. H. Nguyen, D. N. N. Tran, and J. W. Jeon, "Towards Real-Time Vehicle Detection on Edge Devices with Nvidia Jetson TX2," in *2020 IEEE International Conference on Consumer Electronics - Asia, ICCE-Asia 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020. doi: 10.1109/ICCE-Asia49877.2020.9277463.

[20] S. Valladares, M. Toscano, R. Tufiño, P. Morillo, and D. Vallejo-Huanga, "Performance Evaluation of the Nvidia Jetson Nano Through a Real-Time Machine Learning Application," pp. 343–349, 2021, doi: 10.1007/978-3-030-68017-6_51.

[21] "TensorFlow." https://www.tensorflow.org/ (accessed Apr. 23, 2023).

[22] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012, doi: 10.1109/CVPR.2012.6248074.

[23] M. Cordts *et al.*, "The Cityscapes Dataset for Semantic Urban Scene Understanding," Apr. 2016, [Online]. Available: http://arxiv.org/abs/1604.01685

[24] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," May 2014, [Online]. Available: http://arxiv.org/abs/1405.0312