Comparative Analysis of YOLO-Based Algorithms for Vehicle Detection in Aerial Imagery

Amin Dustali[1], Mahdi Hasanlou[1,*], Seyed Majid Azimi[2]

[1] School of Surveying and Geospatial Engineering, College of Engineering, University of Tehran, Tehran, Iran
{amin.doustali; hasanlou}@ut.ac.ir
[2] Remote Sensing Technology Institute, German Aerospace Center (DLR), Oberpfaffenhofen, Germany
seyedmajid.azimi@dlr.de

Keywords: YOLO, Vehicle Detection, Real-time object detection, Edge Computing, Deep Learning, Aerial Imagery.

Abstract

Real-time object detection has become an essential tool in applications such as traffic surveillance, autonomous vehicles, and industrial monitoring. Among various algorithms, the You Only Look Once (YOLO) series has garnered significant attention for its balance between speed and accuracy. Since its introduction in 2016, YOLO has seen significant advancements and it has been widely adopted due to its ability to provide fast and accurate real-time detection. Over the years, different versions, including YOLO-v1 to YOLO-v11, have introduced improvements in both accuracy and speed. This paper presents a comparative analysis of four recent versions of YOLO-v8-n, YOLO-v9-t, YOLO-v10-n, and YOLO-v11-n focusing on evaluating their detection accuracy and speed in aerial imagery using the EAGLE dataset. Each version incorporates specific advancements aimed at improving performance under different conditions. The study examines the models using a standardized dataset of aerial images with varying illumination and weather conditions. Key performance metrics, such as inference time and Average Precision (AP), are used to evaluate how each model performs in the vehicle detection task in challenging environments. The results provide valuable insights into the suitability of these YOLO models for real-world applications, particularly in dynamic urban environments and areas where traditional camera systems may be less effective. This study aims to identify the fastest and most accurate YOLO model for vehicle detection in aerial imagery using embedded GPU board of Nvidia Jetson AGX Xavier, contributing to the performance enhancement in real-time surveillance and monitoring systems.

1. Introduction

Object detection models have evolved significantly over the years, and they can be broadly categorized into two main groups: one-stage models and two-stage models. One-stage models, such as YOLO (Redmon et al., 2016) (real-time object detection with versions, ranging from YOLO-v1 to YOLO-v11), SSD (Liu et al., 2016) (a small and popular model that uses multi-scale techniques for better accuracy in detecting small objects), RetinaNet (Lin et al., 2017b) (an optimized model for handling imbalanced datasets with a new loss function and Feature Pyramid Network (FPN)(Lin et al., 2017a)), and LADet (Zhou et al., 2019) (a lightweight and adaptable model designed for multi-scale object detection), quickly extract image features and provide accurate results. These models are highly efficient for real-time object detection, particularly when the detection process needs to be completed in a single pass (Vijayakumar and Vairavasundaram, 2024). On the other hand, two-stage models include RCNN (Girshick et al., 2015) (which involves region extraction, feature extraction with Convolutional Neural Network (CNN), and classification using Support Vector Machines (SVM)), SPP (He et al., 2015) (an improved version of RCNN using a pyramid structure for better multi-scale feature extraction), Fast RCNN (Girshick et al., 2015) (which improves accuracy and speed by using a Region proposal Network (RPN) and refining bounding boxes), and Faster RCNN (Ren et al., 2015) (which increases detection speed using anchor boxes and the sliding window technique). Mask RCNN (He et al., 2017), built upon Faster RCNN, not only detects objects but also performs instance segmentation by predicting pixel-level masks for objects within predefined Region of Interest Network (ROI). In recent years Vision Transformers have shown exceptional performance and have exceeded the performance of CNN-based algorithms by far. Algorithms such as (Carion et al., 2020) are based on transformer-based model. Traditionally, two-stage or transformer-based algorithms have had better accuracy than YOLO-based algorithms at the cost of slower inference time. Recently, there have been works to close this gap by making DETR-based algorithms as fast as or even faster YOLO-based algorithms (Peng et al., 2024), (Zhao et al., 2024).

Object detection algorithms, particularly the YOLO family, have seen significant advancements since its introduction. YOLO has gained widespread use in various applications, such as traffic surveillance, security systems, and autonomous vehicles, due to its high speed and suitable accuracy for the real-time object detection. Since the release of YOLO-v1, which is a single-stage model for object detection, several advanced versions, including YOLO-v2, YOLO-v3, and more recently YOLO-v11, have introduced new features that improve the algorithm's accuracy, speed, and flexibility.

YOLO-v1 (Redmon et al., 2016) is developed in 2016 by Joseph Redmon and his colleagues, and it is the first real-time object detection system using a single-stage approach. This model predicts both bounding boxes and class probabilities in one pass, offering higher speed and accuracy compared to traditional methods like Faster RCNN .Then, YOLO-v2 (Redmon and Farhadi, 2017) is released in the same year and renamed YOLO9000. This version improved speed and accuracy by using Darknet-19 (a faster architecture than VGGNet) and allowed the detection of over 9000 object categories. New features like anchor boxes and batch normalization are added to

* Corresponding author

YOLO-v2 to improve detection accuracy for objects of various sizes. In 2018, YOLO-v3 (Redmon and Farhadi, 2018) is introduced, adding skip connections and FPN to address the vanishing gradient problem and improve object detection at different scales. This version also used anchor box clustering to improve bounding box prediction accuracy. . Two years later, in 2020, YOLO-v4 (Bochkovskiy et al., 2020) is released, focusing on improving both speed and accuracy. This version incorporated features like Bag of Specials (BoS), Bag of Freebies (BoF), and Self-adversarial Training (SAT) to increase the model's robustness to input variations. It also used CSPDarknet-53 (Wang and Wang, 2021) as its CNN backbone, which is more efficient in feature extraction. In the same year, YOLO-v5 (Zhang et al., 2022) is introduced using PyTorch instead of Darknet. This version leveraged features like Autoanchor and Genetic Evolution (GE) for anchor box optimization, as well as augmentation techniques like mosaic and copy-paste to improve accuracy. YOLO-v5 includes five different versions (YOLO-v5n, s, m, l, x), each varying in depth and width of the convolutional layers. YOLO-v6 (Li et al., 2022) is released in September 2022, focusing on industrial applications with high-speed and high-accuracy detection across various hardware. This version uses anchor-free techniques and features like Rep-PAN and EfficientRep for backbone and neck optimization. YOLO-v6 also utilizes Varifocal Loss (Zhang et al., 2021) and Distribution Focal Loss to improve detection accuracy. Finally, YOLO-v7 (Wang et al., 2023) is released in 2022, aiming to increase detection speed and accuracy by using E-ELAN architecture for better gradient flow and model scaling to support various model sizes. YOLO-v7 also introduced RepConv for convolution optimization and a dual-head architecture for improved training and prediction.

This study compares the performance of four recent YOLO versions: YOLO-v8-n, YOLO-v9-t, YOLO-v10-n, and YOLO-v11-n. Each version has its own specific features and advancements that make it suitable for different applications. Specifically, YOLO-v8 (Terven et al., 2023) is one of the successful versions of this model, which improves feature extraction and increases object detection accuracy by utilizing an advanced architecture in the Backbone and Neck sections. Additionally, the use of the Anchor-free head in YOLO-v8 increases processing speed and provides significant improvements in both accuracy and efficiency. Another standout feature of YOLO-v8 is its optimal balance between accuracy and speed, making it ideal for real-time applications. Figure 1 illustrates the architecture of YOLO-v8. Furthermore, the variety of pretrained models tailored for different needs makes this version more flexible, allowing users to select a model suited to their specific task. In YOLO-v9-t (Ambali Parambil et al., 2024) shown in Figure 2. new technologies like PGI (Programmable Gradient Information) and GELAN (Generalized Efficient Layer Aggregation Network) are proposed, resulting in significant improvements in performance and accuracy. PGI is used to preserve essential data in the deeper layers of the network, ensuring that information is retained throughout the learning process, which enhances model performance. This feature is particularly effective in lightweight models, which use fewer parameters. Alongside that, GELAN optimizes the use of parameters and computational efficiency, making it a strategic advancement in the YOLO-v9 architecture. This version also addresses information loss challenges in deep neural networks and, by introducing inverse functions, helps the network fully preserve information Next, YOLO-v10 (Mao et al., 2024) as illustrated in Figure 3 adds new features like non-maximum suppression (NMS)-free

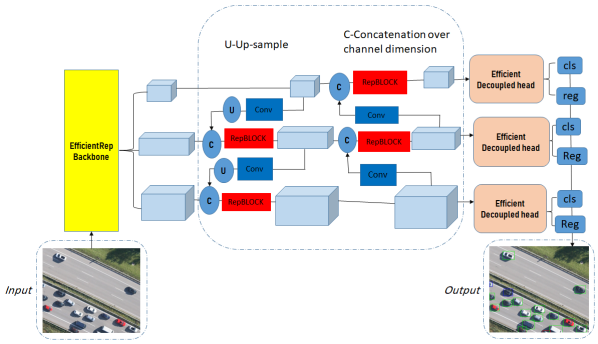

Figure 1. The architecture of YOLO-v8 with RepBLOCKs.

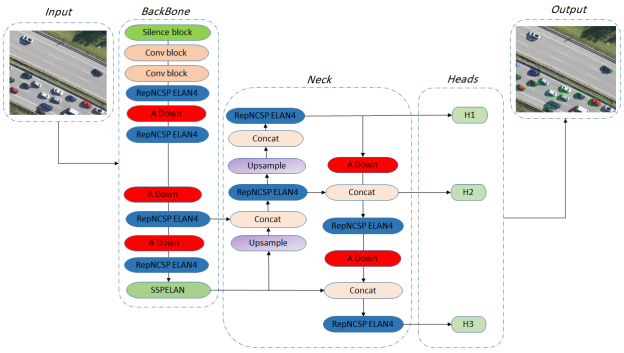

Figure 2. YOLO-v9 architecture introduces PGI (Programmable Gradient Information) and GELAN (Generalized Efficient Layer Aggregation Network), enhancing performance and accuracy.

training and Dual-Assignment Learning. This model helps reduce inference latency and optimizes processing speed by combining one-to-many and one-to-one strategies during training, improving the quality of predictions. Additionally, Large Kernel Convolutions and Partial Self-Attention Modules improve the model's performance without significantly increasing computational cost. Finally, YOLO-v11 (Sapkota et al., 2024) with the architecture shown in Figure 4, with its enhanced Backbone and Neck structures, improves the ability to extract features and increases object detection accuracy. This version also reduces the number of parameters by 22% compared to YOLO-v8m, achieving higher accuracy on the COCO dataset (Lin et al., 2014), making it a more computationally efficient model. YOLO-v11 is capable of running in various environments, including edge devices, cloud platforms, and systems supporting NVIDIA GPUs, offering flexibility to use the model in different scenarios. Additionally, YOLO-v11 supports a wide range of computer vision tasks, including object detection, instance segmentation, pose estimation, and object detection with Oriented Bounding Boxes (OBB) This study is conducted using the EAGLE Dataset, and the goal is to compare the accuracy and speed of these four models in vehicle detection within aerial images. Using a standardized dataset, this research evaluates the performance of each version under various conditions and aims to select the lightest model with the highest accuracy in the vehicle detection task using the EAGLE (Azimi et al., 2020) dataset of aerial images on Nvidia Jetson AGX Xavier.

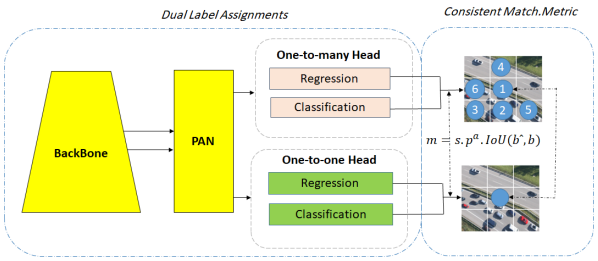

Figure 3. YOLO-v10 architecture with NMS-free training and dual-assignment learning reduces inference latency and improves prediction quality.

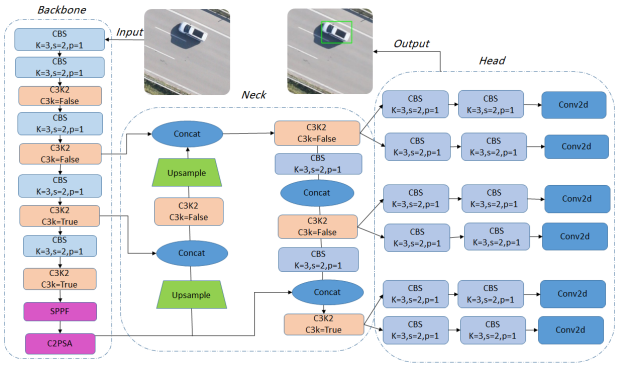

Figure 4. YOLO-v11 architecture, featuring enhanced Backbone and Neck structures, improves feature extraction and detection accuracy.

2. Data and Methods

2.1 Data Preprocessing of EAGLE dataset

The (Azimi et al., 2020) dataset is comprised of aerial images with two different sizes such as 5616×3744px which are acquired at different time of day/year, illumination, weather, camera angle and altitudes between 300m and 3000m, leading to a range of , as known as spatial resolution, from 5to 45per each pixel. We combine both classes of large-vehicle and small-vehicle in this dataset to handle this task as a binary object detection task.

Before training the models, the aerial images from the EAGLE dataset are cropped into 1024×1024px tiles and resized during training and inference to 416×416px in order to ensure uniformity across all input images. Data augmentation techniques such as random rotations, flipping are applied to increase the robustness of the models. The dataset is split into training (23,001 tiles) and validation (7,682 tiles) sets. Additionally, the data is augmented to prevent overfitting and improve model generalization.

2.2 Model Training

We fine-tune four different versions of YOLO — YOLO-v8-n, YOLO-v9-t, YOLO-v10-n, and YOLO-v11-n—on the EAGLE dataset. The models are trained for 10 epochs with a batch size of 8. The AdamW optimizer with a learning rate of 0.001 and momentum of 0.9 is used for model optimization. Pre-trained weights are utilized as the starting point for each model to speed up convergence and improve performance. The training is conducted using PyTorch.

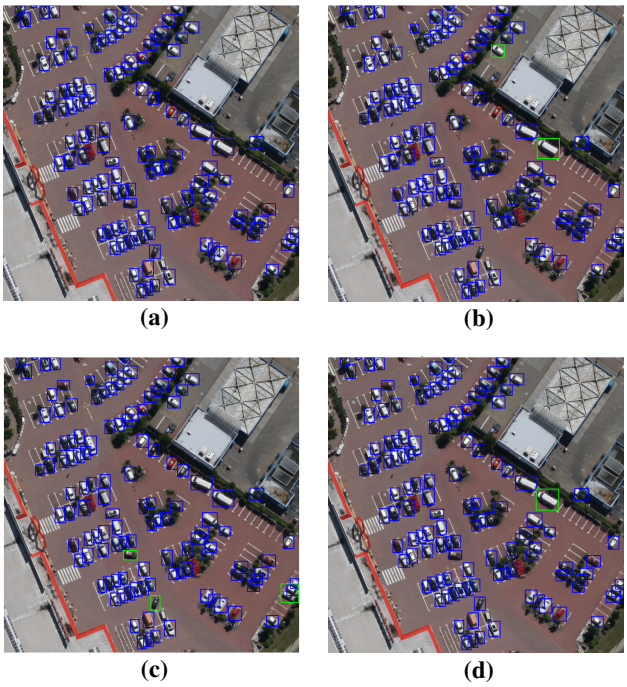

Figure 5. Comparison of vehicle detection results for YOLO models with pre-trained models on MS-COCO dataset, tested on the EAGLE dataset: (a) YOLO-v8-n, (b) YOLO-v9-t, (c) YOLO-v10-n, (d) YOLO-v11-n. Red bounding boxes: false positives, blue: false negatives, green: true positives.

2.3 Evaluation Metrics

The models are evaluated using key performance metrics including inference time and AP. Inference time is measured to assess the real-time performance of the models. AP is used to evaluate the detection accuracy, particularly in complex scenarios with varying illumination and weather conditions based on Table 1. Additionally, F1-Score is calculated to combine precision and recall, particularly in the cases of class imbalance.

	Predicted Positive	**Predicted Negative**
Actual Positive	**TP**	**FN**
Actual Negative	**FP**	**TN**

Table 1. Confusion matrix illustrating the performance of the model, showing true positive, true negative, false positive, and false negative values across different classes.

Precision (P) and Recall (R) rates are calculated using the equations:

$$P = \frac{TP}{TP + FP} \tag{1}$$

$$R = \frac{TP}{TP + FN} \tag{2}$$

where TP stands for the True Positives, FN for the False Negatives and FP for the False Positives. These metrics depend on the confidence threshold required to count as a detection and can be plotted against each other for every confidence threshold, the so-called Precision-Recall curve. The area under this curve for

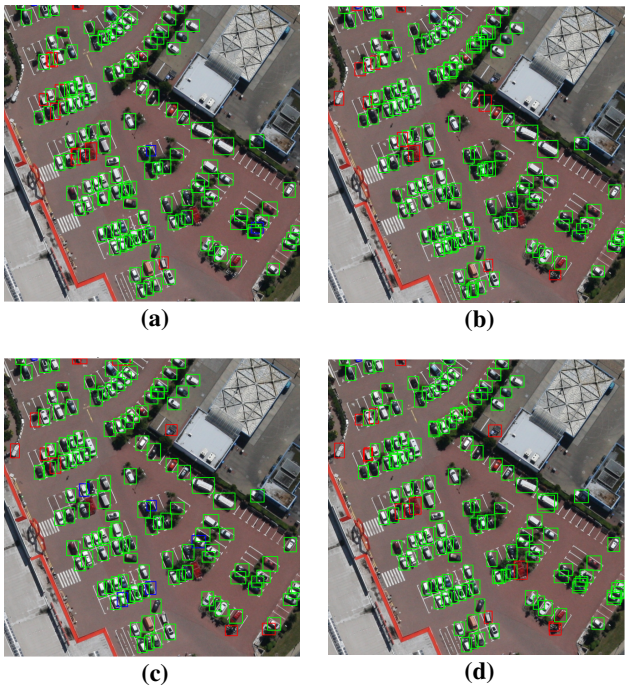

Figure 6. Comparison of fine-tuned vehicle detection results for YOLO models fine-tuned and tested using the EAGLE dataset: (a) YOLO-v8-n, (b) YOLO-v9-t, (c) YOLO-v10-n, (d) YOLO-v11-n. Red bounding boxes: false positives, blue: false negatives, green: true positives.

Models	Inference Time (s)	AP	F1-Score
YOLO-v8-n	0.77	0.00	0.00
YOLO-v9-t	1.17	0.18	0.03
YOLO-v10-n	0.92	0.35	0.05
YOLO-v11-n	1.17	0.10	0.01

Table 2. Comparison of the performance among YOLO-v8-n, YOLO-v9-t, YOLO-v10-n and YOLO-v11-n with pret-trained models on the MS-COCO dataset.

each class is then calculated as AP.

$$AP = \int_0^1 P(R)dR \tag{3}$$

F1 score is also calculated using the equation of:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

2.4 Environment and Tools

The models are trained using PyTorch with GPU for acceleration.

3. Results

The performance of the four YOLO models — YOLO-v8-n, YOLO-v9-t, YOLO-v10-n, and YOLO-v11-n — is summarized in Table 2 when using the pretrained models on MS-COCO dataset and the results of the algorithms finetuned on EAGLE dataset are provided in Table 3. The inference on test images of EAGLE has been carried out on Jetson AGX Xavier using the maximum performance mode with overclocking. The qualitative results are shown in Figure 5 for the pre-trained model of MS-COCO and in Figure 6.

Models	Inference Time (s)	AP	F1-Score
YOLO-v8-n	4.14	0.63	0.94
YOLO-v9-t	4.53	0.62	0.96
YOLO-v10-n	4.00	0.61	0.92
YOLO-v11-n	4.15	0.63	0.95

Table 3. Comparison of the performance among YOLO-v8-n, YOLO-v9-t, YOLO-v10-n and YOLO-v11-n after having fine-tuned on the EAGLE dataset on Jetson AGX Xavier.

YOLO-v11-n demonstrates the best performance in comparison with the other version on the AP factor, showing superior detection results. This is likely due to its use of synthetic data generation for training and its reduced number of parameters, which allows it to handle complex environments with limited real-world data. YOLO-v11-n also excels in terms of the F1-Score, indicating a balanced performance between precision and recall, particularly in complex detection tasks. YOLO-v10-n achieves the fastest inference time, making it the most suitable model for real-time applications. This can be attributed to its NMS-free training and spatial-channel decoupled downsampling, which optimize the model’s speed.

4. Discussion

The results indicate notable differences in how each version of YOLO handles the vehicle detection task in aerial imagery, offering valuable insights for selecting the most appropriate model based on the use case. Qualitative results also reveals that how much finetuning can improve the performance of the model and what is the generalization of YOLO algorithms trained in MS-COCO on different image modalities. For instance, Figure 5 clearly shows that YOLO algorithms have a very poor performance on the vehicle detection in aerial imagery when trained only on the MS-COCO dataset, while after being fine-tuned, Figure 6 indicates the high jump in the performance.

4.1 Inference Time

One of the most critical factors in real-time applications, such as traffic monitoring or autonomous vehicles, is the speed of the model. The results reveals that YOLO-v10-n outperforms the other models in terms of inference time. Its ability to eliminate redundant bounding boxes through NMS-free training and optimize inference speed via the spatial-channel decoupled downsampling contributes to its faster processing times, making it an ideal candidate for real-time applications. This feature is especially important for scenarios where low-latency performance is crucial. In contrast, YOLO-v9-t exhibits the slowest inference time, which could limit its suitability for real-time applications that require rapid decision-making. Although this version showed improvements in feature retention through its PGI and GELAN modules, which helps in the complex image recognition task, the trade-off in speed may be a disadvantage for some use cases.

4.2 Accuracy and Precision

In terms of detection accuracy, YOLO-v11-n emerges as the best-performing model, achieving the highest AP. This is attributed to its utilization of synthetic data generation during

training, which allowed the model to better handle complex, real-world scenarios with limited available data. By reducing the model's parameters by 22%, YOLO-v11-n also improves its precision in detecting objects in cluttered or challenging environments, such as areas with occlusions or unusual angles. This makes YOLO-v11-n particularly suitable for applications where high accuracy is prioritized over speed, such as disaster management or infrastructure monitoring in hard-to-reach areas. On the other hand, YOLO-v8-n shows solid performance, but lags behind in comparison to YOLO-v11-n in terms of AP. The novel loss function in YOLO-v8-n like Varifocal Loss and CIoU Loss plays a significant role in its ability to handle complex detection tasks, but it could not match the level of precision seen in YOLO-v11-n.

4.3 F1-Score and Balance Between Precision and Recall

The F1-Score —a metric that combines precision and recall— is highest in YOLO-v11-n, indicating that it provides a good balance between true positive and false negative rates, even in challenging environments. This makes YOLO-v11-n the most well-rounded model, excelling both in detection accuracy and in maintaining a good balance between precision and recall. In contrast, YOLO-v9-t, despite its slower inference time, demonstrates solid F1-Score results, showing that its ability to retain important features across layers helps mitigate the challenges of imbalanced classes. However, the slower inference time limits its real-time applicability.

4.4 Implications for Real-World Applications

While YOLO-v10-n excels in speed, making it ideal for real-time use cases, YOLO-v11-n proves to be the most accurate and reliable model for tasks where precision is more important than processing speed. For instance, in aerial imagery used for disaster management or search and rescue, where the environment can be complex and unpredictable, YOLO-v11-n is the preferable choice, due to its ability to detect objects accurately even in challenging conditions. Conversely, if the application demands fast decision-making and real-time monitoring, such as in traffic surveillance or autonomous driving, YOLO-v10-n would be the most appropriate model. Its speed, coupled with relatively high accuracy, allows it to process aerial images quickly while maintaining a reasonable level of precision.

4.5 Limitations and Future Work

Despite the impressive results of YOLO-v11-n, one limitation is its reliance on synthetic data generation, which might not fully replicate the diversity and complexity of real-world scenarios. Future work could focus on improving the model's ability to handle diverse environmental factors, such as varying weather conditions, different light intensities, and dynamic objects like moving vehicles or pedestrians.

Moreover, the real-time inference of YOLO-10-n could be further optimized by reducing its computational complexity without sacrificing accuracy. Techniques such as knowledge distillation or model pruning could be explored to improve the efficiency of these models for edge devices.

In addition with the rise of new object detection algorithms based on vision transformers and the closing gap between the performance of two difference between DETR and YOLO-based algorithms, future works could be on how to leverage the advantages of each design architecture for different use cases whether speed, accuracy or a speed-accuracy trade-off is required on the application side defined by the end user.

5. Conclusion

In conclusion, the study demonstrates that each YOLO version has its unique strengths, making them suitable for different use cases in edge computing using embedded processors. Given the limited amount of computing power as well as memory on embedded devices, efficient vehicle detection algorithms are required. YOLO-v10-n stands out for its speed, making it ideal for real-time applications, while YOLO-v11-n excels in accuracy, making it the best choice for complex, data-scarce environments. Understanding the strengths and weaknesses of these models helps in selecting the most appropriate version based on the specific requirements of the task, whether it's real-time object detection or high-precision vehicle identification in challenging conditions.

References

Ambali Parambil, M., Ali, L., Swavaf, M., Bouktif, S., Gochoo, M., Aljassmi, H., Alnajjar, F., 2024. Navigating the YOLO Landscape: A Comparative Study of Object Detection Models for Emotion Recognition. *IEEE Access*, 12, 12.

Azimi, S. M., Bahmanyar, R., Henry, C., Kurz, F., 2020. Eagle: Large-scale vehicle detection dataset in real-world scenarios using aerial imagery. In: 2020 25th International Conference on Pattern Recognition (ICPR), pages 6920–6927.

Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M., 2020. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-end object detection with transformers. *European conference on computer vision*, Springer, 213–229.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2015. Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans. Patt. Analy. Machine Intel, 38(1):142–158.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, 2961–2969.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Patt. Analy. Machine Intel, 37(9):1904–1916.

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Wei, X., 2022. Yolov6: A single-stage object detection framework for industrial applications. arXiv preprint arXiv:2209.02976.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017a. Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2117–2125.

Lin, T. Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017b. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, 2980-2988.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., 2014. Microsoft coco: Common objects in context. In: Computer Vision – ECCV 2014, pages 740–755, Cham: Springer International Publishing. Editors: Fleet, David; Pajdla, Tomas; Schiele, Bernt; Tuytelaars, Tinne; ISBN: 978-3-319-10602-1.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., Berg, A. C., 2016. Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14 (pp. 21-37). Springer International Publishing.

Mao, M., Lee, A., Hong, M., 2024. Efficient Fabric Classification and Object Detection Using YOLOv10. *Electronics*, 13, 13.

Peng, Y., Li, H., Wu, P., Zhang, Y., Sun, X., Wu, F., 2024. D-FINE: Redefine Regression Task in DETRs as Fine-grained Distribution Refinement. *arXiv preprint arXiv:2410.13842*.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 779–788.

Redmon, J., Farhadi, A., 2017. Yolo9000: Better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7263–7271.

Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, 28.

Sapkota, R., Meng, Z., Karkee, M., 2024. Synthetic Meets Authentic: Leveraging LLM Generated Datasets for YOLO11 and YOLOv10-Based Apple Detection Through Machine Vision Sensors. *Smart Agricultural Technology*, 9, 9.

Terven, J., Córdova-Esparza, D.-M., Romero-González, J.-A., 2023. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, 5(4), 1680–1716.

Vijayakumar, A., Vairavasundaram, S., 2024. Yolo-based object detection models: A review and its applications. Multimedia Tools and Applications.

Wang, C. Y., Bochkovskiy, A., Liao, H. Y. M., 2023. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7464–7475.

Wang, W., Wang, Y., 2021. Underwater target detection system based on yolo v4. In: 2021 2nd International Conference on Artificial Intelligence and Information Systems, Chongqing, China, 5 pages.

Zhang, H., Wang, Y., Dayoub, F., Sunderhauf, N., 2021. Varifocalnet: An iou-aware dense object detector. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8514–8523.

Zhang, Y., Guo, Z., Wu, J., Tian, Y., Tang, H., Guo, X., 2022. Real-time vehicle detection based on improved yolo v5. Sustainability, Volume 14, Number 19, Article Number 12274.

Zhao, Y., Lv, W., Xu, S., Wei, J., Wang, G., Dang, Q., Liu, Y., Chen, J., 2024. Detrs beat yolos on real-time object detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16965–16974.

Zhou, J., Tian, Y., Li, W., Wang, R., Luan, Z., Qian, D., 2019. Ladet: A light-weight and adaptive network for multi-scale object detection. In: Asian Conference on Machine Learning, 912-923. PMLR.