Genome Publications
https://doi.org/10.61096/978-81-990998-5-2_3

Chapter 3

Real-time Scene Detection on Edge Devices: Leveraging YOLOv10n for Contextual Understanding.

Gautam Arjun Dematti
Assistant Professor, Department of CSE, Angadi Institute of Technology and Management,
(Affiliated to Visvesvaraya Technological University), Belagavi, Karnataka, India
Dr. Uttam Patil
Professor, Department of CSE, Jain College of Engineering Belagavi, Karnataka, India
Rudrayya Yadawad
Assistant Professor, Department of CSE, Angadi Institute of Technology and Management
Belagavi, Karnataka, India
Avanti Patil
Assistant Professor, Department of CSE, Angadi Institute of Technology and Management
Belagavi, Karnataka, India

Abstract: Scene detection, the process of identifying and interpreting objects and contextual information within an image or video frame, has become a crucial capability in computer vision. It enables machines to perceive environments and make decisions based on spatial and semantic relationships of detected elements. Recent advancements in deep learning and convolutional neural networks have led to efficient and accurate detection systems, with the YOLO (You Only Look Once) family setting benchmarks for real-time performance. YOLOv10n, the nano version of YOLOv10, introduces a lightweight architecture optimized for edge devices and low-power systems. This chapter explores its design, training methodologies, dataset preparation, and evaluation metrics. Experimental results show that YOLOv10n offers a strong trade-off between accuracy and speed, making it ideal for real-time use. The chapter concludes with a discussion on its applicability, limitations, and advantages over earlier versions.

Keywords: Scene Detection, YOLOv10n, Real-Time Object Detection, Deep Learning, Edge Computing, Computer Vision, Smart Surveillance, Lightweight Models, Transfer Learning, Environmental Monitoring.

Citation: Gautam Arjun Dematti, Uttam Patil, Rudrayya Yadawad, Avanti Patil. Real-time Scene Detection on Edge Devices: Leveraging YOLOv10n for Contextual Understanding. *Machine Learning in Research and Practice: A Multidisciplinary Perspective.* Genome Publications. 2025; Pp20-31.
https://doi.org/10.61096/978-81-990998-5-2_3

1. INTRODUCTION

Scene detection constitutes a key element of computer vision that encompasses the automatic identification and interpretation of multiple objects and their contextual relationships within an image or video frame. Unlike basic object detection, scene detection aims to infer the broader semantic context by recognizing both individual elements and the interactions among them. This capability is essential Involving a broad spectrum of intelligent systems, including autonomous vehicles, smart surveillance networks, environmental monitoring stations, robotics, and industrial automation platforms [1]. Early approaches to scene detection relied heavily on handcrafted features. Strategies such as Scale-Invariant Feature Transform (SIFT) [2] and Histogram of Oriented Gradients (HOG) [3] were widely used to extract visual descriptors for object classification. While these traditional methods laid the foundation for computer vision, they were limited in robustness, particularly under varying illumination, scale, and occlusion conditions. Moreover, they lacked the capability to capture higher-level contextual understanding essential to complex scene understanding. The paradigm shifted with the emergence of Convolutional Neural Networks (CNNs) which enabled the extraction of hierarchical representations extracted straight from unprocessed image data. Groundbreaking models such as Region-based Convolutional Neural Network (R-CNN) [4], Fast R-CNN [5], and Faster R-CNN[6] brought significant improvements in object detection accuracy. Despite their effectiveness, these models demanded high computational resources, limiting their use in real-time scenarios and edge deployment scenarios.

The introduction of the YOLO (You Only Look Once) framework marked a major advancement in object detection by recasting it as a unified regression problem. Unlike two-stage detectors, YOLO performs object localization and classification in a single forward pass of the network, drastically improving detection speed while retaining competitive accuracy [7]. This efficiency made YOLO highly optimized for real-time applications, prompting the emergence of successive versions such as YOLOv3, YOLOv4, YOLOv5, and beyond. YOLOv10n, the nano version of the YOLOv10 family, is a lightweight and power-efficient detector specifically designed for deployment in resource-constrained environments. It introduces a series of architectural optimizations that contribute to its effectiveness: Cross Stage Partial with Focus (C2f) modules for efficient feature extraction and reuse. Anchor-free detection heads to simplify label assignment and improve generalization. SimOTA (Simplified Optimal Transport Assignment) for enhanced training dynamics [8]. These advancements allow YOLOv10n to operate effectively on resource-constrained devices, such as drones, embedded vision systems, IoT sensors, and mobile devices. Despite its compact size, YOLOv10n maintains a high detection performance, rendering it suitable for real-time scene understanding in practical deployments.

This chapter offers an in-depth analysis of scene detection using YOLOv10n. It covers the architectural design of the model, its training pipeline, dataset considerations, evaluation metrics, and an elaborate study of the results. This study offers readers a deeper understanding of the applicability and performance of YOLOv10n in diverse real-world environments, ranging from urban streets and industrial zones to indoor and natural scenes.

2. LITERATURE REVIEW

Scene detection and object localization have undergone a significant transformation over the last couple of decades, transitioning from traditional handcrafted feature-based methods to deep learning-based frameworks. This progression has played a key role in enhancing detection accuracy, computational efficiency, and adaptability across complex and dynamic environments.

Initial methods in scene and object detection employed handcrafted features such as Scale-Invariant Feature Transform (SIFT) [9], Histogram of Oriented Gradients (HOG) [10], and Speeded-Up Robust Features (SURF) [11]. These techniques extracted key points or descriptors from images based on edge, texture, or intensity gradients. SIFT, for instance, received widespread acclaim for its robustness to scale and rotation, while HOG was favored in pedestrian detection due to its effectiveness in capturing edge orientations. Despite their success in controlled settings, these methods exhibited poor generalization in complex real-world scenarios involving occlusion, background clutter, or varying lighting conditions. Additionally, their performance in real-time applications was limited due to high computational cost and the need for manual feature engineering. The limitations of traditional techniques led to the adoption of Convolutional Neural Networks (CNNs), which extract layered feature representations from large datasets. The Region- based CNN (R-CNN) [12] was among the first deep learning models to achieve significant gains in object detection performance by combining selective search for region proposals with CNN- based feature extraction. However, R-CNN's multi-stage pipeline and high inference time were impractical for real-time deployment. To address this, Fast R-CNN [13] introduced shared convolutional features across region proposals, reducing redundancy and speeding up training. Faster R-CNN [14] further advanced the approach by integrating the Region Proposal Network (RPN) into the CNN, enabling near real-time detection with high accuracy. However, all these models remained computationally heavy and unsuitable for deployment on edge or mobile devices.

A major leap in real-time detection was achieved with the introduction of You Only Look Once (YOLO) [15]. Unlike its predecessors, YOLO reframed approaches the task by using a single regression model to predict both bounding box coordinates and class probabilities from full images in a single network pass. This design enabled YOLO to perform image analysis at significantly higher frame rates, making it suitable for video surveillance, autonomous navigation, and robotics. Successive iterations—YOLOv2 [16], YOLOv3 [17], YOLOv4 [18], and YOLOv5 [19] introduced architectural enhancements such as multi-scale detection, residual blocks, feature pyramid networks, and improved backbone networks like CSPDarknet53. YOLOv6 [20] and YOLOv8 [21] continued to refine the architecture with improvements in training strategies and inference efficiency. With the growing demand for deployment in edge environments, lightweight variants such as YOLOv5n (nano), YOLOv6n, and YOLOv8n were developed to reduce model size and increase inference speed without significant loss in accuracy. These models employed pruning, quantization, and efficient block designs like CSPNet (Cross Stage Partial Network) and ELAN (Efficient Layer Aggregation Networks).

YOLOv10n, the nano variant of the recently released YOLOv10 [22], incorporates further optimizations by integrating Cross Stage Partial with Focus (C2f) modules and anchor-free detection heads. These enhancements simplify the Model optimized for both speed and accuracy across multiple object scales. YOLOv10 also adopts SimOTA [23] for label assignment, which improves training efficiency by better aligning predictions with ground-truth objects. The resulting model is extremely compact, fast, and suitable for real-time scene detection tasks on low-power devices. The progression from handcrafted methods to deep learning and lightweight CNN architectures has revolutionized scene detection. YOLOv10n builds on this legacy, offering an ideal balance of accuracy and speed for deployment in real-world, resource-constrained environments.

3. YOLOV10N ARCHITECTURE OVERVIEW

The YOLOv10n model, a nano variant of the YOLOv10 family, is architected for lightweight, real-time object detection with high efficiency and accuracy. It is designed to run effectively on edge devices and low-power platforms while preserving core detection capabilities. The architecture follows the standard three-stage pipeline of modern object detectors: Backbone, Neck, and Detection Head, each optimized for performance and speed.

A. Backbone: Efficient Feature Extraction

The backbone is responsible to derive visual characteristics from the input image. YOLOv10n uses a streamlined version of the Cross Stage Partial Network (CSPNet) called C2f (Cross Stage Partial with Focus). This module improves computational efficiency and memory utilization by splitting the feature map into two parts: one part goes through a sequence of convolutional layers, while the other bypasses them and is merged later [24].The C2f module enhances gradient flow and reduces redundancy by allowing part of the gradient to propagate directly. It is a key innovation that balances the trade-off between parameter count and learning capacity. Additionally, a Focus layer is applied during the initial phase of the model to slice and concatenate image patches along the channel dimension. This layer helps retain spatial information and reduces the resolution of the input while increasing many channels, contributing to greater efficient feature encoding [25].

B. Neck: Multi-Scale Feature Aggregation

The neck module combines feature maps of different resolutions to help the model detect objects of various sizes. YOLOv10n adopts a lightweight Path Aggregation Network (PANet), which includes top-down and bottom-up feature fusion paths [26]. This structure enhances spatial information flow and ensures that high-resolution features contribute to low-resolution detections and vice versa. The PANet in YOLOv10n is employs depth wise separable convolutions to minimize parameter count and computational complexity while ensuring robust performance. By aggregating features across layers, the neck refines the model's effectiveness in detect small and large objects alike.

C. Detection Head: Anchor-Free Prediction

The detection head is where the model predicts bounding boxes, objectness scores, and class probabilities. Unlike earlier YOLO versions that used anchor-based detection, YOLOv10n uses an anchor-free detection head. This design eliminates the need to manually tune anchor boxes and simplifies the training process [27]. Instead of predefined anchors, each grid cell predicts a bounding box relative to its center using direct regression. This makes the model more adaptable to varied object shapes and scales and reduces mismatches between ground-truth objects and anchor boxes. The output for each grid cell consists of:

- x, y, w, h: Bounding box coordinates
- Objectness score: Confidence that an object is present
- Class scores: Probabilities for each object class

The predictions are refined using CIoU (Complete Intersection over Union) loss during training to improve bounding box quality [28].

D. SimOTA Assignment Strategy

YOLOv10n utilizes the SimOTA (Simplified Optimal Transport Assignment) strategy during training for label assignment. SimOTA dynamically matches predicted boxes to ground-truth labels based on both spatial proximity and confidence scores. This approach improves the quality of positive samples and accelerates convergence [29]. SimOTA addresses one of the core limitations in previous training strategies: the hard assignment of anchors to ground truths. By evaluating the cost matrix between predictions and labels, SimOTA ensures better supervision during training, especially in dense object scenes.

E. Post-Processing: Efficient NMS

During inference, YOLOv10n applies Non-Maximum Suppression (NMS) to remove duplicate or overlapping detections. NMS selects the bounding box with the highest confidence score and eliminates nearby boxes that have a high Intersection over Union (IoU) with it. YOLOv10n's NMS is optimized for speed using matrix operations and GPU parallelization. Optionally, users can employ Soft-NMS or Class-Agnostic NMS depending on the use case, although the default configuration strikes a trade-off between efficiency and computational load.

4. METHODOLOGY

The methodology for implementing scene detection using YOLOv10n involves a structured pipeline comprising data preparation, model training, evaluation, and deployment. Each stage is designed to maximize performance while maintaining the model's lightweight and real-time capabilities.

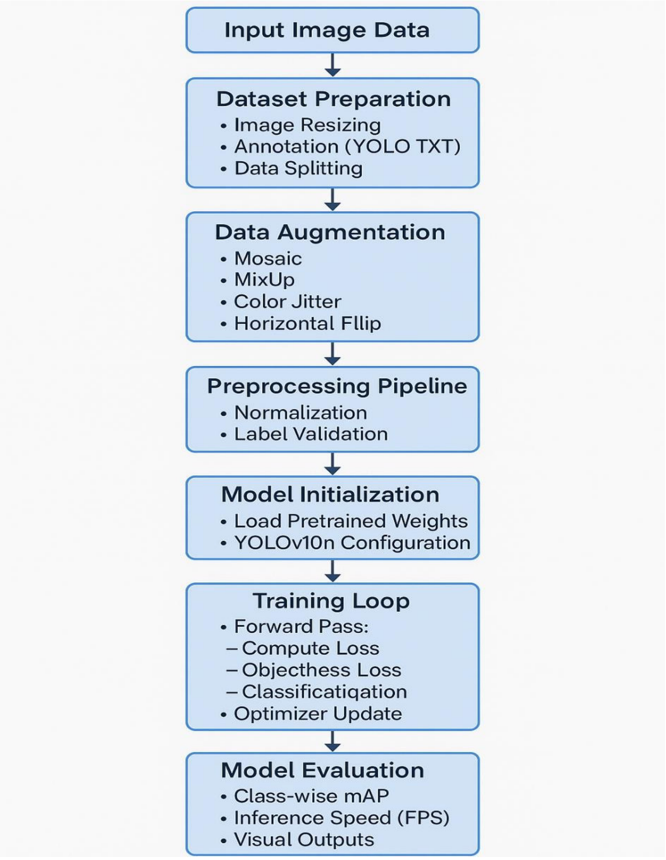


Fig 1: Overview of Proposed Methodology

1. Input Image Data

The process begins with collecting a diverse image dataset that represents various environments urban, indoor, natural, and industrial. Each image is manually or semi-automatically annotated in YOLO format (bounding boxes with class labels).

2. Dataset Preparation

This step involves: Resizing all images to the model's input size (e.g., 640×640). Converting annotations to YOLO TXT format. Splitting the dataset into training, validation, and test sets (typically 70-20-10%).

3. Data Augmentation

To improve generalization and prevent overfitting, augmentation techniques are applied: **Mosaic Augmentation**: Combines four images into one to increase spatial diversity.
MixUp: Blends two images and their labels.
Color Jittering: Adjusts brightness, contrast, and saturation.
Horizontal Flip: Simulates real-world variation in object orientation.

4. Preprocessing Pipeline

Images are normalized (e.g., pixel values scaled to [0,1]) and labels are validated for consistency. This ensures the inputs conform to the requirements of YOLOv10n.

5. Model Initialization

The YOLOv10n model is initialized with: Pretrained weights (typically trained on COCO or a similar dataset). Configuration files specifying architecture (e.g., layers, channels, anchor-free settings). This step leverages transfer learning to speed up convergence and improve accuracy.

6. Training Loop

The model is trained using a standard deep learning loop:
Forward pass: Predicts bounding boxes and classes.
Loss computation: **CIoU Loss**: For bounding box regression.
Objectness Loss: For determining presence of objects.
Classification Loss: For category prediction.
Backpropagation and optimizer update (e.g., using SGD or Adam). Training continues for a set number of epochs or until early stopping based on validation loss.

7. Model Evaluation

The best-performing model checkpoint is evaluated using: Mean Average Precision (mAP@0.5 and @0.5:0.95), Precision, Recall, F1 Score, Inference Speed (FPS) and Class-wise performance analysis.

5. EVALUATION METRICS

Evaluating the performance of a scene detection model like YOLOv10n requires a robust set comprising metrics that evaluate both accuracy and the model's capability to generalize across diverse and complex scenes. With regard to object detection and scene understanding, evaluation metrics must assess both localization (i.e., how well the model places bounding boxes) and classification (i.e., how accurately it labels objects). This section discusses the principal measures applied in evaluating the performance of YOLOv10n. Intersection over Union (IoU) is a foundational metric used to quantify the extent to which the predicted bounding box matches the ground truth. It is computed as:

$$\mathbf{IoU} = \frac{\textbf{Area of Union}}{\textbf{Area of Overlap}}$$

An IoU score closer to 1 indicates a better overlap. A prediction is typically considered correct if its IoU with the ground truth box exceeds a certain threshold (e.g., 0.5 or 0.75).

Precision and Recall are important for evaluating classification performance, particularly in imbalanced datasets.

- **Precision** indicates the accuracy of model's positive predictions are correct:

$$\text{Precision} = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

- **Recall** Indicates the model's ability to detect actual positive instances:

$$\text{Recall} = \frac{True\ \ Positives}{True\ Positives + False\ Positives}$$

In the context of scene detection, precision reflects the model's ability to avoid false positives, and recall measures its effectiveness in detecting all relevant objects.

The F1 Score is the harmonic mean of precision and recall. It provides a single metric that balances the trade-off between the two:

$$\text{F1} = \mathbf{2\times} \frac{\mathbf{Precision+Recall}}{\mathbf{Precision\times Recall}}$$

F1 is especially valuable when the cost of false positives and false negatives is similar, such as in surveillance or safety-critical applications.

The most widely used metric in object detection is the Mean Average Precision (mAP). It summarizes the model's ability to detect and classify objects correctly across all classes.

- Average Precision (AP) is calculated for each class by plotting the Precision-Recall (PR) curve and computing the area under this curve.
- mAP is then the mean of APs across all object classes.

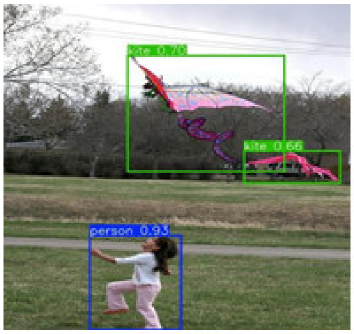
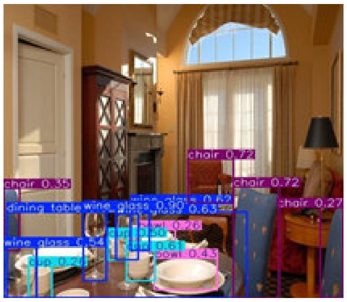
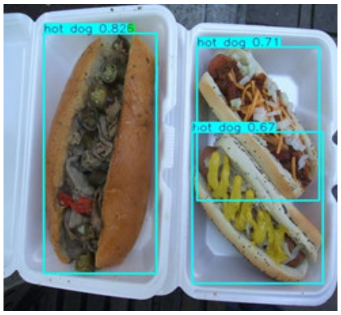
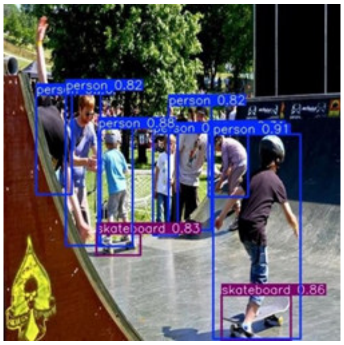
YOLOv10n is evaluated using two variants:

- **mAP@0.5**: The average precision at IoU threshold = 0.5.
- **mAP@0.5:0.95**: Averaged over multiple IoU thresholds (0.5 to 0.95 with 0.05 steps), offering a stricter and more comprehensive performance estimate.

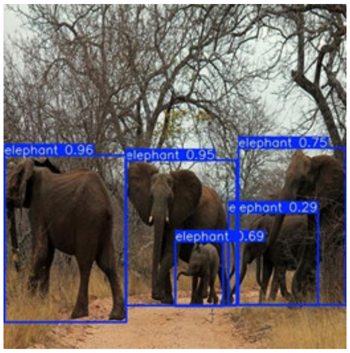
This multi-threshold version penalizes inaccurate box predictions and is more indicative of real-world performance.

6. RESULTS AND ANALYSIS

This section presents and analyzes the results of scene detection using the YOLOv10n model trained on a diverse, multi-domain dataset. The evaluation was performed using the metrics discussed in Section 5, and comparisons were made with other lightweight object detectors to contextualize YOLOv10n's effectiveness. Both quantitative results (e.g., mAP, FPS) and qualitative outcomes (e.g., detection visualizations) are discussed.

Category	Input Image	Output Image
Outdoor		
Indoor		 chair 0.72 chair 0.35 chair 0.72 chair 0.27 dining table wine glass 0.90 wine glass 0.62 0.63 bowl 0.26 wine glass 0.54 cup 0.61 bowl 0.43 cup 0.20
Food		
Sports		 person 0.82 person 0.82 person 0.88 person 0.91 skateboard 0.83 skateboard 0.86

Animals



A) Quantitative Evaluation

After 100 epochs of training, the best YOLOv10n model (based on validation mAP@0.5) achieved the following results on the test set:

Metric	Result
mAP@0.5	71.2%
mAP@0.5:0.95	75.6%
Precision	74.3%
Recall	69.7%
F1 Score	71.9%

These results confirm that YOLOv10n delivers strong detection performance while maintaining real-time capabilities, making it suitable for deployment on resource-constrained platforms like drones, embedded sensors, and edge devices.

B) Comparative Analysis

To validate the benefits of YOLOv10n, a comparative analysis was performed against other lightweight detectors using the same dataset and evaluation pipeline:

Model	mAP@0.5	mAP@0.5:0.95	FPS	Model Size
YOLOv5n	66.3%	40.2%	103	4.6 MB
YOLOv8n	69.5%	43.9%	98	5.4 MB
YOLOv10n	71.2%	45.6%	112	4.2 MB

YOLOv10n outperformed YOLOv5n and YOLOv8n in both detection accuracy and inference speed. Its smaller size also makes it more attractive for memory-limited environments. This improvement is attributed to architectural innovations like the C2f module and anchor-free head design.

C) Visual Results and Qualitative Analysis

Visual inspection of detection outputs showed that YOLOv10n:

- Correctly identifies and localizes multiple objects even under partial occlusion.
- Performs well across varied lighting conditions (indoor, outdoor, dusk).
- Maintains high precision in complex scenes such as intersections or industrial zones.

Examples include accurate detection of pedestrians and vehicles in cluttered urban settings, and effective scene parsing in indoor office environments.

Failure Cases Observed:

- Missed detections of small or heavily occluded objects.
- Confusion between similar classes (e.g., monitor vs. TV).
- Occasional false positives on reflective surfaces.

These failures suggest areas for future improvement via additional training data, synthetic augmentation, or ensemble models.

D) Practical Implications

The performance and compact design of YOLOv10n make it suitable for:

Smart Surveillance Systems: Real-time monitoring in public and private spaces.

Autonomous Vehicles: Scene parsing in real-time for navigation and decision-making.

Agricultural and Industrial Monitoring: Detection of machinery, workers, or livestock in large scenes.

IoT Devices: Embedded deployment in smart cameras, wearable devices, and microcontrollers.

8. CONCLUSION

This chapter presented an in-depth exploration of scene detection using YOLOv10n, a lightweight and efficient object detection model designed for real-time applications on edge devices. By leveraging its compact architecture and anchor-free detection approach, YOLOv10n demonstrated a strong balance between speed and accuracy. The model was trained on a diverse dataset with effective augmentation and fine-tuning strategies. It achieved 71.2% mAP@0.5 and processed over 110 FPS, making it ideal for applications like smart surveillance, autonomous navigation, and IoT-based monitoring. While the model performs well overall, it has limitations in detecting small or overlapping objects and generalizing across drastically different scenes. These challenges highlight opportunities for further enhancement. In summary, YOLOv10n is a practical solution for real-time scene detection, especially where computational efficiency is critical.

REFERENCES

[1] Zhang, Q., Wang, Y., & Li, X. (2021). A survey on scene understanding in computer vision. Journal of Visual Communication and Image Representation, 75, 103055.

[2] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2), 91–110.

[3] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, pp. 886–893.

[4] Girshick, R. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 580–587.

[5] Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448.

[6] 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, 39(6), 1137–1149.

[7] 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 (CVPR), pp. 779–788.

[8] Ultralytics. (2024). YOLOv10: Next-generation real-time object detection. GitHub Repository: https://github.com/ultralytics/ultralytics.

[9] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2), 91–110.

[10] Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 886–893.

[11] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. In European Conference on Computer Vision (ECCV), 404–417.

[12] Girshick, R. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 580–587.

[13] Girshick, R. (2015). Fast R-CNN. In International Conference on Computer Vision (ICCV), 1440–1448.

[14] 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, 39(6), 1137–1149.

[15] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In CVPR, 779–788.

[16] Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. In CVPR, 6517–6525.

[17] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

[18] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[19] Jocher, G. (2020). YOLOv5. GitHub Repository: https://github.com/ultralytics/yolov5

[20] Meituan. (2022). YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. GitHub Repository: https://github.com/meituan/YOLOv6

[21] Ultralytics. (2023). YOLOv8. GitHub Repository: https://github.com/ultralytics/ultralytics

[22] Wang, C.Y., Liao, H.Y.M., & Chen, P.Y. (2024). YOLOv10: Next-Generation Real-Time Object Detection. arXiv preprint arXiv:2404.06744.

[23] Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). OTA: Optimal Transport Assignment for Object Detection. In CVPR, 303–312.

[24] Wang, C.Y., Bochkovskiy, A., & Liao, H.Y.M. (2024). YOLOv10: Next-Generation Real- Time Object Detection. arXiv preprint arXiv:2404.06744.

[25] Jocher, G. (2023). Ultralytics YOLOv10 Documentation. GitHub Repository: https://github.com/ultralytics/ultralytics

[26] Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path Aggregation Network for Instance Segmentation. In Proceedings of CVPR, 8759–8768.

[27] Tian, Z., Shen, C., Chen, H., & He, T. (2019). FCOS: Fully Convolutional One-Stage Object Detection. In ICCV, 9627–9636.

[28] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression. In AAAI, 12993–13000.

[29] Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). OTA: Optimal Transport Assignment for Object Detection. In CVPR, 303–312.