Anusandhanvallari
Vol 2025, No.1
September 2025
ISSN 2229-3388

A Comparative Survey of YOLO Variants in Object Detectors

Mr. Jayasheelan Palanisamy[1], Dr. A. Somasundaram[2]

[1]Assistant Professor, Department of Data Science

Sri Krishna Adithya college of arts and Science College Coimbatore, Tamil Nadu, India

sheelan.jsp@gmail.com

[2]Department of Computer Applications

Sri Krishna Arts and Science College

Coimbatore, Tamil Nadu, India.

somasundaram@gmail.com

ABSTRACT

As a single-stage object detection framework, YOLO (You Only Look Once) has gained prominence for its remarkable balance between speed and accuracy across diverse detection tasks. This research article provides a comprehensive survey of the YOLO family of algorithms, tracing their evolution from the earliest version to the most recent advancements. The review offers an in-depth analysis of the performance and defining characteristics of each iteration, with particular emphasis on YOLO's applications in real-time detection, especially on embedded systems. Special attention is given to recent developments in model compression and optimization techniques aimed at addressing the challenges posed by the large size of YOLO models, thereby enabling deployment on resource-constrained devices. Practical implementation examples are also discussed to illustrate its applicability in real-world scenarios. Finally, the study highlights potential research directions, including novel architectural innovations and advanced training strategies, that may further enhance the YOLO framework. Overall, this review serves as a valuable reference for researchers and practitioners seeking insights into the evolving landscape of YOLO-based object detection.

KEYWORDS

YOLO, object detection, deep learning, application, compressing.

1. Introduction

Object detection is a fundamental task in computer vision that involves locating objects of interest within an image and accurately classifying each detected target. In recent years, it has become one of the most popular and widely studied tasks in computer vision due to its broad range of applications and rapid technological progress.

Traditional object detection algorithms rely on sliding windows or image segmentation techniques to generate a large number of candidate regions. Features such as Histograms of Oriented Gradients (HOG) [1] are then extracted from these regions and passed to classifiers like Support Vector Machines (SVM) [2] for categorization. However, these approaches require producing numerous candidate boxes, which significantly reduces efficiency. Consequently, their speed and accuracy fail to meet the demands of real-world applications, creating a bottleneck for traditional methods.

The breakthrough came with the introduction of deep convolutional neural networks (CNNs), beginning with the release of AlexNet in 2012 [3]. Deep learning opened new avenues for object detection, leading to a wide range of models that leverage CNNs to enhance both accuracy and efficiency. These deep learning-based object detection methods can broadly be divided into two categories: region-based and regression-based. Region-based systems, also known as two-stage detectors, first generate candidate regions and then classify them using CNNs—for example, the R-CNN framework [4]. In contrast, regression-based systems, known as one-stage detectors, bypass candidate region generation and directly perform object detection through regression analysis.

Comparative studies reveal that two-stage detectors generally achieve higher accuracy but suffer from slower inference speeds, while one-stage detectors strike a balance with superior efficiency and real-time performance. Among one-stage detectors, the most influential and widely adopted framework is YOLO (You Only Look Once) [5].

Thanks to its impressive performance, YOLO has undergone continuous improvements and updates, leading to the latest version, YOLOv8 [6], along with other notable variants such as YOLOR [7] and YOLOX [8]. Shao et al. [9] provide a detailed discussion of YOLO algorithms, though their work focuses only on YOLOv5 [10] and excludes newer developments. Similarly, Terven et al. [11] present a summary of YOLO but in a less intuitive manner.

To address these gaps, this article offers a comprehensive overview of YOLO models from YOLOv1 through YOLOv8, as well as YOLOR and YOLOX, to equip researchers with a deeper understanding of the YOLO family and guide them in selecting models best suited to their needs.

The remainder of this paper is organized as follows: Section 2 presents the architectures of the YOLO family, summarizing them in terms of structure, input size, AP, AP50, FPS, and parameter counts. Section 3 discusses the improvements and applications of YOLO across various domains. Section 4 introduces model compression techniques and reviews the use of compressed YOLO models. Finally, Section 5 concludes with an overview of YOLO's development trends and potential future research directions.

2. Overview on key features of YOLO families

The initial YOLO was presented by Joseph Redmon et al. in CVPR 2016. The YOLO means "You Only Look Once," which glances at an image like a human and offhandedly knows what object is in the image, what they are doing, and where they are. Unlike traditional object detection models based on deep learning, YOLO had excellent accuracy and speed as a regression-based object detection algorithm. YOLO is an advanced one-stage object detection framework that has evolved over the years and spawned several versions. This section introduces the differences between YOLOv1-YOLOv8, YOLOX, and YOLOR. YOLO algorithm forgoes the traditional sliding window technique, and it divides the input image into $S \times S$ grids, each of which predicts B bounding boxes of the same class and the confidence of each grid for C different classes. Each bounding box predicts five values: (x,y,w,h,c), representing the bounding box's position, size, and confidence, respectively. Each grid predicts $(B \times 5 + C)$ values, after which Non-Maximum Suppression (NMS) is used to remove duplicate detections.

2.1 YOLOv1

Accuracy—comprising both classification accuracy and localization accuracy—along with detection speed, are critical metrics for evaluating the performance of object detection models [12]. Unlike traditional methods that generate candidate boxes and classify them using CNNs, YOLO directly regresses bounding box coordinates and class probabilities at multiple positions in the image, thereby enabling efficient end-to-end object detection.

In YOLOv1 [5], the input image is resized to 448×448 and processed using features extracted by a CNN. The architecture employs a backbone network similar to GoogLeNet [13], consisting of 24 convolutional layers and two fully connected layers, with 1×1 convolutions applied to reduce feature map dimensions. The model is pre-trained on ImageNet [14] and fine-tuned on the PASCAL VOC dataset for validation. YOLOv1 divides the input image into a 7×7 grid, with each grid cell predicting two bounding boxes, resulting in 7×7×2 = 98 candidate boxes and allowing the detection of up to 49 objects per image.

Despite its efficiency, YOLOv1 struggles with detecting small objects and densely packed targets. On the PASCAL VOC 2007 benchmark, it achieved an average precision (AP) of 63.4%.

2.2 YOLOv2

YOLOv2 [15], inspired by the VGG architecture [16], introduced several significant improvements over YOLOv1. It employed a new backbone, Darknet-19, consisting of 19 convolutional layers and five max-pooling layers. Unlike YOLOv1, which relied on fully connected layers for bounding box prediction—often leading to a loss of spatial information and reduced localization accuracy—YOLOv2 adopted anchor boxes from Faster R-CNN [17] to improve bounding box regression. Additionally, the integration of batch normalization enhanced convergence and overall model stability. Robustness was further achieved by training with variable input image sizes, allowing the model to adapt to objects at different scales.

A notable contribution of YOLOv2 was its ability to extend detection to object categories with limited labeled samples. By jointly training on multiple datasets, including ImageNet for classification and MS COCO for detection, the authors introduced the WordTree hierarchical classification method, enabling real-time detection of over 9,000 object categories. This extended version became widely known as YOLO9000.On the PASCAL VOC 2007 benchmark, YOLOv2 achieved an average precision (AP) of 78.6%, representing a 15.2% improvement over YOLOv1.

2.3 YOLOv3

YOLOv3 [18] introduced significant architectural enhancements by adopting Darknet-53 as its backbone. The design incorporated residual connections inspired by ResNet [19], allowing the network to be deepened without encountering gradient explosion issues, thereby improving convergence stability. To enhance detection across different object sizes, YOLOv3 employed a strategy similar to the Feature Pyramid Network (FPN) [20], performing multi-scale predictions by generating bounding boxes at three different scales. For anchor box design, k-means clustering was used to determine bounding box priors more effectively. Unlike YOLOv2, YOLOv3 utilized three anchor boxes per scale, targeting large, medium, and small objects. By fusing feature maps across three scales and replacing the softmax function with a logistic classifier, YOLOv3 enabled multi-label classification, further improving flexibility in object detection.

These improvements significantly enhanced performance on small object detection while maintaining high efficiency. In fact, YOLOv3 achieved detection speeds 3–4 times faster than earlier YOLO models when bounding box precision was relaxed, with comparable detection accuracy.

At the time of its release, the benchmark for evaluating detection algorithms had shifted from PASCAL VOC to the more challenging MS COCO dataset. Consequently, YOLOv3 and later versions were assessed using COCO metrics. On the MS COCO test-dev 2017 dataset, YOLOv3 achieved an average precision (AP) of 31.0% and AP50 of 55.3% at 20 FPS, demonstrating its balance between speed and accuracy.

2.4 YOLOv4

YOLOv4 [21] introduced a range of optimization strategies categorized into two groups: bag-of-freebies and bag-of-specials. The bag-of-freebies consists of techniques that enhance training without affecting inference speed, while the bag-of-specials includes modules that slightly impact inference time but provide significant performance gains.

A key innovation in YOLOv4 is the adoption of Cross Stage Partial (CSP) networks [22] and the Mish activation function [23] in the backbone. CSPNet addresses the problem of redundant gradient information in deep networks by partitioning feature maps and integrating gradients more effectively, thereby reducing the number of parameters and FLOPs while maintaining accuracy. These optimizations allow YOLOv4 to achieve higher detection performance compared to earlier YOLO versions, while also being efficient enough to train on a single GPU.

2.5 YOLOv5

YOLOv5 [10] shares a structural similarity with YOLOv4 but introduces flexibility through model scaling across different channel sizes. It provides five model variants—YOLOv5-N (Nano), S (Small), M (Medium), L (Large), and X (Extra Large)—ranging from lightweight models suitable for resource-constrained environments to larger models optimized for higher accuracy. Unlike earlier versions, YOLOv5 was developed in PyTorch, making it easier to deploy across diverse hardware platforms compared to YOLOv4.

Although no official research paper has been published for YOLOv5, it has undergone continuous updates, with the latest iterations extending capabilities beyond detection to include image classification and instance segmentation, while also offering faster training.

On the MS COCO test-dev 2017 benchmark, YOLOv5x6 achieved an AP of 55.8% with an input size of 1536×1536. With a smaller input size of 640×640, YOLOv5x reached an AP of 50.7%, achieving speeds of over 200 FPS on an NVIDIA Tesla V100, highlighting its balance of accuracy and real-time performance.

2.6 YOLOvR

In 2021, the YOLOv4 research team introduced You Only Learn One Representation (YOLOR) [7], a novel framework designed to enhance adaptability by integrating implicit knowledge into neural networks. Traditional CNN-based models primarily rely on explicit features learned during training, which often limits their ability to generalize to new or unseen tasks. YOLOR addresses this limitation by unifying explicit knowledge (learned directly from data) with implicit knowledge (derived subconsciously, similar to human intuition). This design equips the model with a learning mechanism that more closely resembles human cognitive processes, allowing it to generalize beyond the scope of training data.

YOLOR's unified representation enables its application across multiple computer vision tasks, including object detection and image segmentation. On the MS COCO test-dev 2017 benchmark, YOLOR-D6 achieved an AP of 55.4% and an AP50 of 73.3% at 30 FPS on an NVIDIA Tesla V100. These results demonstrate that incorporating implicit knowledge into the network improves overall task performance, validating YOLOR as a significant step forward in object detection research.

2.8. YOLOv6

The Meituan Vision AI Department released YOLOv6[31] in 2022. The overall structure refers to YOLOv4 but uses a more advanced mechanism. Firstly, they use a new backbone efficiency developed based on RepVGG[32]. Secondly, the neck of YOLOv6 adopts the PANet[25], and RepPAN is obtained after the neck is enhanced. YOLOv6's architecture is similar to YOLOX. YOLOv6 also uses Efficient Decoupled Head and anchor-free technology. In addition, YOLOv6 also uses a self-distillation strategy to reduce the cost of reasoning. Similar to YOLOv5, YOLOv6 also provides multiple versions to facilitate model quantification and hardware deployment. It is also suitable for application in the industrial field.

Evaluated on the MS COCO dataset test-dev 2017, YOLOv6-L achieves an AP of 52.5% and *AP*50 of 70.0% on a NVIDIA Tesla T4 in the same environment with TensorRT.

2.7 YOLOv7

The research team of YOLOv4 and YOLOR propose YOLOv7[33] 2022. YOLOv7 outperforms all known object detectors in speed and accuracy, ranging from 5 FPS to 160 FPS. Like YOLOv4, YOLOv7 is trained from scratch on the MS COCO dataset.

The main changes in the architecture of YOLOv7 are Extended efficient layer aggregation networks (E-ELAN) by improved ELAN and Model scaling for concatenation-based models. If more computational blocks are stacked indefinitely, it may destroy the stable state of the network. ELAN[34] enables the deeper network to learn and converge efficiently by controlling the shortest and longest gradient path. E-ELAN proposed by YOLOv7 further takes expand, shuffles, and merges cardinality to achieve the ability to continuously enhance the learning ability of the network without destroying the original gradient path.

Unlike architectures such as ResNet, applying model scaling to a concatenation-based architecture results in a change in the ratio of input and output channels, which leads to the model's inefficiency for hardware. YOLOv7 proposed a composite scaling approach that maintains the characteristics of the model at the time of initial design and maintains the optimal architecture. The composite scaling method proposed by YOLOv7 maintains the model's properties at the initial design. It preserves the optimal structure by scaling the width factor on the transition layer with the same amount of variation.

Evaluated on the MS COCO dataset test-dev 2017, when input image size is 640×640, YOLOv7 achieves AP of 51.4% and *AP*50 of 69.7% at 161 FPS by NVIDIA Tesla V100.

2.8 YOLOv8

The YOLOv5 team releases YOLOv8[6] in January 2023, and an official article still needs to be published. The main improvement as follows:

Backbone: The backbone of YOLOv8 is CSPDraknet-53. As YOLOv5, YOLOv8's C3 module is replaced by the C2f module with richer gradient flow, and different channel numbers are adjusted for different scale models to achieve further lightweight. Moreover, YOLOv8 still uses the SPPF module used in YOLOv5.

Head: The Head part has two significant improvements compared with YOLOv5. Firstly, it is replaced with the current mainstream Decoupled-Head, which separates the classification and detection. Secondly, it is also changed from Anchor-Based to Anchor-Free.

Loss: YOLOv8 abandons the previous IOU matching or unilateral ratio distribution method

Table 1

Summary of YOLO families architecture

Model	Backbone	Neck	Anchor
YOLOv1	GoogLeNet,VGG-16	2×fully connected layers	No
YOLOv2	Darknet-19	fully connected layers	**Yes**
YOLOv3	Darknet-53	FPN	**Yes**
YOLOv4	CSPDarknet-53	SPP,PANet	**Yes**
YOLOv5	CSPDarknet-53	SPPF,CSP-PAN	**Yes**
YOLOR	CSPDarknet-53	FPN,SPP	**Yes**
YOLOv6	EfficientRep	Rep-PAN	No
YOLOv7	Extended-ELAN	SPPCSPC	No
YOLOv8	Darknet-53	SPP,PAN	No

but takes the Task-Aligned Assigner positive and negative sample matching method. Furthermore, YOLOv8 introduced Distribution Focal Loss (DFL)[35].

Data augmentation can improve model performance, but introducing mosaic augmentation in training may have adverse effects. YOLOv8 turns off the Mosaic augmentation in the last ten epochs, improving accuracy. To meet the needs of different scenarios, YOLOv8 provides different size models of N / S / M / L / X scales.
Evaluated on MS COCO dataset test-dev 2017, YOLOv8X achieves an AP of 53.9% with 283 FPS on NVIDIA Tesla A100 and TensorRT.

Summary
The architecture of the YOLO families is shown in Table 2, including the backbone, neck, and anchor. YOLOv2 first proposes darknet-19 as the backbone, and YOLOv3 deepened the 19 convolutional layers to 53 layers. Almost all subsequent YOLO models use Darknet-53 or an improved version of darknet-53 as the backbone architecture. The initial YOLO does not use anchors, which were used in YOLOv2, and improves the prediction accuracy until YOLOX takes an anchor-free approach and performs well. Since then, subsequent versions of YOLO dropped the use of anchors.
Table2 shows the performance, size, FPS, parameters, and GPU used for training mainstream YOLO versions. Among them, YOLOv1 and YOLOv2 were tested on PASCAL VOC2007. When YOLOv3 was released, the benchmark for object detection had changed from PASCAL VOC to Microsoft COCO. Therefore, the performance of subsequent versions is tested on Microsoft COCO. In addition, the YOLOv6 and YOLOv8x models are quantized by TensorRT. From the parameters in Table2, YOLO increasingly favors lighter models. In YOLOv8, the model YOLOv8v8m is only 3.7% less than the AP_{50} of YOLOv8x, but the parameters are only about 38% of that of YOLOv8x, which is 25.9M. In many versions of YOLO, researchers can choose models according to their needs to find a balance between accuracy and speed.

Table 2
Summary of YOLO families main indexes

Model	Size	AP(%)	AP_{50}(%)	GPU	Params(M)	FPS
YOLOv1	448	63.4	-	TitanX	-	-
YOLOv2	448	78.6	-	TitanX	-	-
YOLOv3	416	31.0	55.3	-	-	34.5
YOLOv4	416	41.2	62.8	Tesla V100	64.4	96
YOLOv5x	640	50.7	68.9	Tesla V100	86.7	208.3
YOLOR-D6	1280	**55.4**	**73.3**	Tesla V100	152.0	30
YOLOv6	640	52.5	70.0	Tesla T4	58.5	121
YOLOv7	640	51.4	69.7	Tesla V100	**36.9**	161
YOLOv8x	640	-	53.9	Tesla V100	68.2	**283.3**

3. Compression methods for YOLO models
The YOLO model has become more complex in pursuit of better performance. Model compression has become the focus of research to facilitate deployment on edge devices. This section introduces several techniques for

model compression, such as model pruning, parameter quantization, knowledge distillation, and lightweight model design[47].

Model Pruning: Model pruning is achieved by searching for redundant layers/channels in the model and deleting them with little or no impact on performance. By pruning the YOLOv3tiny network, Shi et al.[48] reduce the network computation by 68.7% Wu et al.[49] propose a real-time apple flower detection using the channel-pruned YOLOv4 model. The number of parameters is reduced by 96.74%.

Parameter Quantization: Parameter quantization converts floating-point calculations to low-bit-rate integer calculations, such as converting float32 to int8 or int4. We quantize the YOLOv4 model on 8-bit through TensorRT and deploy it to the Jetson Nano, and it achieves the purpose of real-time detection of dirty eggs[50]. Wang et al.[51] deploy the pruned YOLOv3 detection model on the FPGA, and the model size is reduced by 80% with little change in accuracy.

Knowledge Distillation: The teacher network is a complex pre-trained network, and the student network is a simple small network. By transferring knowledge, the student network that is more suitable for reasoning can be obtained through the teacher network. Chen et al.[52] propose a lightweight ship detector by knowledge distillation. Xing et al.[53] use ResNet101 as the teacher network and DD-YOLO as the student network, reducing the model complexity to 61.4%, which is more suitable for mobile deployment.

Lightweight Model Design: Lightweight DNN model design refers to the redesign based on the existing deep neural network structure to reduce the number of parameters and computational complexity. Liu et al.[54] replace the backbone of YOLOv3 with ShuffleNet to realize real-time vehicle detection. Liu et al.[55] uses the backbone of YOLOv4 with Mobilenetv3, which improves the accuracy of an extensive pedestrian detection network.

Conclusion

This paper summarizes the models of the YOLO series and provides a detailed analysis of the critical feature of each model.Afterward, the application of YOLO in different scenarios was introduced. Finally, the method of YOLO model compression was briefly described, and an illustration of the application of YOLO in model compression.

Although the YOLO series is the leader in the speed-accuracy balance in the field of target detection, its main work is for the computer side. In further work, how to make YOLO lighter and faster is worth pondering, especially embedded devices such as Nvidia Jetson Nano and Raspberry Pi. Moreover, it will become a trend to carry DNN models on FPGA to make the model run more efficiently. In addition, the technology combining AI and IoT will also bring more convenience to human life. Finally, since the release of YOLOv4, integrating various advanced algorithms has become an essential way to develop the YOLO algorithm. With the development of the YOLO framework, YOLO is more versatile and powerful and will be applied in a broader range of fields.

References

[1] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), volume 1, San Diego, CA, USA, 2005, pp. 886–893. doi:10.1109/CVPR.2005.177.

[2] N. Cristianini, J. Shawe-Taylor, An introduction to support vector machines and other kernel-based learning methods, Cambridge university press, 2000.

[3] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Communications of the ACM 60 (2017).

[4] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 580–587. doi:10.1109/CVPR.2014.81.

[5] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779–788. doi:10.1109/CVPR.2016.91.

[6] Glenn Jocher and Ayush Chaurasia and Jing Qiu, Yolo by ultralytics, 2023. URL: https://github.com/ultralytics/ultralytics.

[7] C.-Y. Wang, I.-H. Yeh, H.-Y. M. Liao, You only learn one representation: Unified network for multiple tasks, arXiv preprint arXiv:2105.04206 (2021).

[8] Z. Ge, S. Liu, F. Wang, Z. Li, J. Sun, Yolox: Exceeding yolo series in 2021, arXiv preprint arXiv:2107.08430 (2021).

[9] Y. Xiao, Z. Tian, J. Yu, Y. Zhang, S. Liu, S. Du, X. Lan, A survey of deep learning-based object detection, Multimedia Tools and Applications 79 (2020). doi:10.1007/s11042-020-08976-6.

[10] Glenn Jocher, Yolov5 by ultralytics, 2020. URL: https://github.com/ultralytics/yolov5.

[11] J. Terven, D. Cordova-Esparza, A comprehensive review of yolo: From yolov1 to yolov8 and beyond, arXiv preprint arXiv:2304.00501 (2023).

[12] Z. Zou, K. Chen, Z. Shi, Y. Guo, J. Ye, Object detection in 20 years: A survey, Proceedings of the IEEE 111 (2023). doi:10.1109/JPROC.2023.3238524.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke,A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015, pp. 1–9. doi:10.1109/CVPR.2015.7298594.

[14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International Journal of Computer Vision 115 (2015). doi:10.1007/s11263-015-0816-y.

[15] J. Redmon, A. Farhadi, Yolo9000: Better, faster, stronger, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 6517–6525.doi:10.1109/CVPR.2017.690.

[16] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[17] R. Girshick, Fast r-cnn, in: 2015 IEEE International Conference on Computer Vision(ICCV), Santiago, Chile, 2015, pp. 1440–1448. doi:10.1109/ICCV.2015.169.

[18] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, CoRR abs/1804.02767 (2018). URL: http://arxiv.org/abs/1804.02767. arXiv:1804.02767.

[19] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[20] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 936–944. doi:10.1109/CVPR.2017.106.

[21] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, Yolov4: Optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934 (2020).

[22] C.-Y. Wang, H.-Y. Mark Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, I.-H. Yeh, Cspnet: A new backbone that can enhance learning capability of cnn, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern

Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 1571–1580. doi:10.1109/CVPRW50498.2020.00203.

[23] D. Misra, Mish: A self regularized non-monotonic activation function, arXiv preprint arXiv:1908.08681 (2019).

[24] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 37 (2015). doi:10.1109/TPAMI.2015.2389824.

[25] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, Path aggregation network for instance segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 8759–8768. doi:10.1109/CVPR.2018.00913.

[26] S. Woo, J. Park, J.-Y. Lee, I. S. Kweon, Cbam: Convolutional block attention module, in: Computer Vision – ECCV 2018, 2018, pp. 3–19. doi:10.1109/CVPR.2018.00913.

[27] G. Song, Y. Liu, X. Wang, Revisiting the sibling head in object detector, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 11560–11569. doi:10.1109/CVPR42600.2020.01158.

[28] Y. Wu, Y. Chen, L. Yuan, Z. Liu, L. Wang, H. Li, Y. Fu, Rethinking classification and localization for object detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 10183–10192. doi:10.1109/ CVPR42600.2020.01020.

[29] Z. Tian, C. Shen, H. Chen, T. He, Fcos: Fully convolutional one-stage object detection, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 9626–9635. doi:10.1109/ICCV.2019.00972.

[30] Z. Ge, S. Liu, Z. Li, O. Yoshie, J. Sun, Ota: Optimal transport assignment for object detection, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR), Nashville, TN, USA, 2021, pp. 303–312. doi:10.1109/CVPR46437.2021.00037.

[31] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, et al., Yolov6: A single-stage object detection framework for industrial applications, arXiv preprint arXiv:2209.02976 (2022).

[32] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, J. Sun, Repvgg: Making vgg-style convnets great again, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2021, pp. 13728–13737. doi:10.1109/CVPR46437.2021.01352.

[33] C.-Y. Wang, A. Bochkovskiy, H.-Y. M. Liao, Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, arXiv preprint arXiv:2207.02696 (2022).

[34] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Designing network design strategies through gradient path analysis, arXiv preprint arXiv:2211.04800 (2022).

[35] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, J. Yang, Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, Advances in Neural Information Processing Systems 33 (2020).

[36] X. Yue, H. Li, M. Shimizu, S. Kawamura, L. Meng, Yolo-gd: A deep learning-based object detection algorithm for empty-dish recycling robots, Machines 10 (2022). doi:10.3390/ machines10050294.

[37] X. Yue, H. Li, L. Meng, An ultralightweight object detection network for empty-dish recycling robots, IEEE Transactions on Instrumentation and Measurement 72 (2023).doi:10.1109/TIM.2023.3241078.

[38] Y. Ge, X. Yue, L. Meng, A high-efficiency dirty-egg detection system based on yolov4 and tensorrt, in: 2022 International Conference on Advanced Mechatronic Systems (ICAMechS), Toyama, Japan, 2022, pp. 59–63.

[39] D. He, Z. Zou, Y. Chen, B. Liu, J. Miao, Rail transit obstacle detection based on improved cnn, IEEE Transactions on Instrumentation and Measurement 70 (2021). doi:10.1109/TIM.2021.3116315.

[40] Z. Li, W. Xie, L. Zhang, S. Lu, L. Xie, H. Su, W. Du, W. Hou, Toward efficient safety helmet detection based on yolov5 with hierarchical positive sample selection and box density filtering, IEEE Transactions on Instrumentation and Measurement 71 (2022).doi:10.1109/TIM.2022.3169564.

[41] T. Morioka, C. Aravinda, L. Meng, An ai-based android application for ancient documents text recognition, in: Proceedings of the 2021 International Symposium on Advanced Technologies and Applications in the Internet of Things, Virtual, Kusatsu, Japan, 2021, pp. 91–98.

[42] Y. Liu, M. Reynolds, D. Huynh, G. Hassan, Study of accurate and fast estimation method of vehicle length based on yolos, in: 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS), Dalian, China, 2020, pp. 118–121. doi:10. 1109/ICAIIS49377.2020.9194930.

[43] X. Yue, H. Li, L. Meng, Ai-based prevention embedded system against covid-19 in daily life, Procedia Computer Science 202 (2022).

[44] Z. Zhuang, G. Liu, W. Ding, A. N. J. Raj, S. Qiu, J. Guo, Y. Yuan, Cardiac vfm visualization and analysis based on yolo deep learning model and modified 2d continuity equation, Computerized Medical Imaging and Graphics 82 (2020).

[45] G. Liu, J. Xing, J. Xiong, Spatial pyramid block for oracle bone inscription detection, in: Proceedings of the 2020 9th International Conference on Software and Computer Applications, New York, NY, USA, 2020,, p. 133–140. doi:10.1145/3384544.3384561.

[46] Y. Fujikawa, H. Li, X. Yue, C. Aravinda, G. A. Prabhu, L. Meng, Recognition of oracle bone inscriptions by using two deep learning models, International Journal of Digital Humanities (2022). doi:10.1007/s42803-022-00044-9.

[47] Z. Li, H. Li, L. Meng, Model compression for deep neural networks: A survey, Computers12 (2023). doi:10.3390/computers12030060.

[48] R. Shi, T. Li, Y. Yamaguchi, An attribution-based pruning method for real-time mango detection with yolo network, Computers and Electronics in Agriculture 169 (2020). doi:10.1007/s11263-014-0733-5.

[49] D. Wu, S. Lv, M. Jiang, H. Song, Using channel pruning-based yolo v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments, Computers and Electronics in Agriculture 178 (2022). doi:10.1016/j.compag.2020.105742.

[50] X. Wang, X. Yue, H. Li, L. Meng, A high-efficiency dirty-egg detection system based on yolov4 and tensorrt, in: 2021 International Conference on Advanced Mechatronic Systems (ICAMechS), Tokyo, Japan, 2021, pp. 75–80. doi:10.1109/ICAMechS54019.2021.9661509.

[51] Z. Wang, H. Li, X. Yue, L. Meng, Design and acceleration of field programmable gate array-based deep learning for empty-dish recycling robots, Applied Sciences 12 (2022).doi:10.3390/app12147337.

[52] S. Chen, R. Zhan, W. Wang, J. Zhang, Learning slimming sar ship object detector through network pruning and knowledge distillation, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 14 (2021). doi:10.1109/JSTARS.2020.3041783.

[53] Z. Xing, X. Chen, F. Pang, Dd-yolo: An object detection method combining knowledge distillation and differentiable architecture search, IET Computer Vision 16 (2022). doi:10. 1049/cvi2.12097.

[54] J. Liu, R. Zhang, Vehicle detection and ranging using two different focal length cameras, Journal of Sensors 14 (2020). doi:10.1155/2020/4372847.

[55] L. Liu, C. Ke, H. Lin, H. Xu, et al., Research on pedestrian detection algorithm based on mobilenet-yolo, Computational intelligence and neuroscience 2022 (2022). doi:10.1155/ 2022/8924027.