Proceedings of the 6th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/71/20241642

A review of the development of YOLO object detection algorithm

Junbiao Liang

South China Normal University, Guangzhou, 510000, China

ljb18823128392@163.com

Abstract. The You Only Look Once (YOLO) algorithm series, as the forefront of object detection technology, has evolved from YOLOv1 to YOLOv10, consistently enhancing detection speed and accuracy. Through literature review and data analysis. This paper mainly discusses the development processes of the YOLO algorithm series, focuses on the changes and innovations in network structure, training strategies, and performance optimization. By introducing techniques, such as CSPNet, Anchor-free, data augmentation, and multi-scale training, the YOLO algorithm has progressively found a better balance between detection speed and accuracy, demonstrating excellent performance in processing real-time images and handling high-complexity scenarios. Furthermore, this paper also addresses some challenges faced by the YOLO algorithms and potential future research directions, such as the lower accuracy in detecting small targets and reduced robustness in complex scenarios. Through analysis, potential optimization directions for the future include further refining network structure and employing more efficient training methods to enhance algorithm efficiency. This paper intends to offer a thorough performance evaluation of the YOLO series algorithms, identify potential areas for future improvements to advance YOLO technology.

Keywords: YOLO algorithm, object detection, computer vision.

1. Introduction

Since the introduction of YOLOv1 [1] in 2015, the YOLO series algorithms have drawn considerable attention due to their novel approach of converting object detection into an end-to-end [2] regression problem. This concept starkly contrasts with two-stage detection methods [3], streamlining the detection process and markedly increasing detection speed, thereby making YOLO ideal for real-time detection applications like video surveillance and autonomous driving.

As the continuous technological advancements, the YOLO series algorithms have undergone numerous iterations, evolving from YOLOv1 to YOLOv10, with each version incorporating enhancements and modifications in network architecture, training strategies, and performance optimization. These updates have not only improved the accuracy and speed of the algorithms but also expanded their applicability across various application requirements and hardware conditions.

This review meticulously traces the development of YOLO from YOLOv1 to YOLOv10, systematically analyzing the critical enhancements in each version through data analysis and literature review. Additionally, it examines the principal challenges algorithms face and explores potential future research directions, aiming to offer readers a thorough technical reference and forecasts of future trends

DOI: 10.54254/2755-2721/71/20241642

for the YOLO series algorithms. Through this comprehensive analysis, people can better grasp research trends, optimize current algorithms, and develop more efficient solutions moving forward.

2. Object detection

Object detection [4], as a core technology in the field of computer vision, is tasked with identifying and locating objects from complex visual data. The identification of objects involves feature extraction and classification, while the location of objects provides the computer with positional information about the objects. The development of object detection can be divided into two stages: the traditional object detection stage (before 2014) and the deep learning-based [5] object detection stage (after 2014). Traditional object detection algorithms relied on manually designed feature extractors (such as Haar features [6], SIFT features [7], etc.) combined with sliding window-based classifiers to achieve object detection. However, early algorithms exhibited poor performance and robustness. Subsequently, deep neural networks were applied to object detection, resulting in improved performance.

Deep learning-based object detection is typically categorized into two types: two-stage detection methods and one-stage detection methods. Two-stage algorithms mainly involve two steps: first, potential candidate regions are extracted from the image, and then these candidate regions are subjected to detailed classification and boundary box refinement to obtain the final detection results. Representative algorithms include R-CNN [8], Fast R-CNN [9], and Mask R-CNN [10]. In contrast, one-stage algorithms directly predict the categories and locations of objects in the image. This approach generally offers faster detection speed, although it may exhibit slightly lower accuracy compared to two-stage algorithms. Representative algorithms include SSD [11], RetinaNet [12], and You Only Look Once (YOLO).

The YOLO series of algorithms, as one of the faster one-stage object detection algorithms, differs from previous methods by treating object detection as a single regression problem, directly obtaining bounding box coordinates and class probabilities from image pixels. This significantly enhances its real-time detection and processing capabilities, making YOLO particularly suitable for applications requiring rapid response and instant analysis, such as autonomous driving, video surveillance, and robotic vision.

3. Evolution from YOLOv1 to YOLOv10

3.1. YOLOv1

YOLOv1 [13] drew inspiration from the GoogLeNet [14] architecture, utilizing 24 convolutional layers to extract image features, followed by 2 fully connected layers to predict bounding boxes and classes. Additionally, it employed 1×1 convolutions for dimensionality reduction combined with 3×3 convolutions, replacing the Inception modules of GoogLeNet. This approach reduced computational load while enhancing the model's non-linear capabilities.

YOLOv1 used the Darknet framework for all its training. Initially, it was pre-trained on the ImageNet1000 [15] dataset. Subsequently, four convolutional layers and two fully connected layers were added, and further training was conducted on the Pascal VOC 2007 [16] and 2012 datasets. Throughout the network's training process, all layers utilized the leaky ReLU [18] activation function except for the last layer, which used the ReLU [17] function. This not only addressed the zero-gradient problem for negative inputs but also enhanced the model's non-linear capabilities.

The design of such a network significantly improved YOLOv1's performance in object detection. Compared to two-stage algorithms, YOLOv1 offered faster detection rates and higher real-time performance. However, its accuracy was still lower compared to advanced algorithms like R-CNN and Fast R-CNN. Additionally, YOLOv1 performed poorly in detecting small clustered objects and objects of uncommon sizes, with occasional instances of missed detections.

3.2. YOLOv2

YOLOv2 [19] made several improvements over YOLOv1 to address issues such as inaccurate localization, low detection accuracy, and low recall rate. Compared to its predecessor, YOLOv2 used

DOI: 10.54254/2755-2721/71/20241642

the Darknet-19 network as its framework (similar to the VGG [20] network), consisting of 19 convolutional layers and 5 max-pooling layers. YOLOv2 achieved higher detection accuracy and faster speed, capable of real-time prediction of over 9000 different object categories.

Regarding training strategies, YOLOv2 first employed Batch Normalization across all YOLO layers to normalize the model and enhance its generalization capability. Additionally, while YOLOv1 required input data of size 224×224 during pre-training and resized inputs to 448×448 for detection, which caused performance loss, YOLOv2 solved this issue. The authors used a High Resolution Classifier method, where the input data size was already 448×448 during pre-training, thus avoiding accuracy loss due to size conversion. Furthermore, YOLOv2 introduced several improvements, such as the anchor box mechanism, K-means clustering for preset anchors, and direct prediction of bounding box center positions, making bounding box detection more efficient and enhancing model performance. YOLOv2 also combined features from different resolutions through the passthrough layer and feature map concatenation, improving small object detection. Finally, multi-scale training was implemented to enhance the model's robustness and adaptability to images of different sizes.

Although YOLOv2 showed an improvement in accuracy compared to YOLOv1, it still lagged behind two-stage algorithms in terms of precision, and detection of small objects remains an area for improvement.

3.3. YOLOv3

YOLOv3 [21] introduced several minor improvements based on YOLOv2. In terms of network structure, the authors were inspired by the FPN [22] method and incorporated residual networks and upsampling modules into Darknet-19, allowing the network to extract deeper features and reducing gradient descent issues. Additionally, YOLOv3 replaced max-pooling layers with convolutional layers with a stride of 2, reducing information loss during the feature extraction process.

For bounding box prediction, YOLOv3 utilized logistic regression to predict the confidence of each bounding box, optimizing the problem of accuracy degradation due to overlapping detection boxes during multi-object detection. In terms of label classification, it adopted a multi-label classification approach, using logistic classifiers for classification, which supports multiple categories for a single object. YOLOv3 also used multi-scale feature maps to predict objects of different sizes.

Overall, YOLOv3 performed relatively well, with improvements in detection accuracy and speed compared to YOLOv2, and enhanced performance in detecting small objects. However, it struggled more with detecting medium- and large-sized objects.

3.4. YOLOv4

YOLOv4 [23] introduced many changes based on YOLOv3, resulting in significant improvements in both accuracy and speed. In terms of network structure, YOLOv4 utilized CSPDarknet53 as the backbone, which effectively reduced computational complexity and improved the model's generalization capability. Additionally, SPP blocks [24] and PAN structures [25] were incorporated into the network, enabling it to handle different input sizes and enhancing feature extraction and fusion capabilities.

Moreover, YOLOv4 employed two different approaches, 'bag of freebies' and 'bag of specials', to improve model performance. The former involves training phase techniques and strategies such as data augmentation, mixed precision training, label smoothing, and hard example mining, which enhance model performance without increasing inference time and computational complexity. The latter involves adding specific modules and operations, such as introducing SPP, PAN, CSP [26] structures, and the Mish activation function [27], which, although increasing inference time and computational complexity, significantly boost the model's detection capabilities and accuracy.

Building upon the foundation of YOLOv3, YOLOv4 introduced numerous strategies to enhance model accuracy, enabling it to surpass most mainstream object detectors of its time in both accuracy and speed, making it suitable for real-time detection tasks. However, YOLOv4 still has certain shortcomings

DOI: 10.54254/2755-2721/71/20241642

in small object detection, computational resource requirements, training complexity, data requirements, and applicability to specific scenarios.

3.5. YOLOv5

YOLOv5 [28] offers faster detection speed and a smaller model compared to YOLOv4, with improvements in both network architecture and training strategies. In the YOLOv5 network structure, the authors used the Focus module before the image enters the backbone, enhancing the model's computational efficiency through slicing. In later versions, the Focus module was replaced with 6×6 convolutions to enable more efficient GPU operation. Additionally, the SPPF structure replaced the SPP structure, significantly improving computational speed. Moreover, data augmentation methods such as Mosaic, Copy-paste, and Random affine [29] were adopted to enhance the model's detection accuracy.

In terms of training strategies, YOLOv5 enhanced the model's generalization and robustness through multi-scale training and random resizing of input sizes. During the early stages of training, warmup training was used with a small learning rate for pre-training, and the cosine learning rate decay strategy was employed to optimize the training process. Additionally, YOLOv5 introduced EMA to smooth weight updates and used mixed precision training to reduce memory usage and accelerate training speed. The combination of these optimization methods enabled YOLOv5 to maintain efficient training while significantly improving detection performance.

YOLOv5 is divided into four versions based on different depth and width: YOLOv5S, YOLOv5M, YOLOv5L, and YOLOv5X. Among these, YOLOv5S is the fastest version, suitable for real-time long-term detection scenarios. Although YOLOv5 offers very fast detection speeds, its shallower network results in some performance degradation in detection accuracy.

3.6. YOLOX

YOLOX [30] builds upon the optimizations and improvements of YOLOv3 and YOLOv5, significantly enhancing the overall detection performance of the model. Unlike previous versions of YOLO, YOLOX introduces a decoupled head structure, which separates the classification and regression tasks, allowing the model to focus more on the current task and improve convergence speed and accuracy. Specifically, within the FPN, it includes a 1×1 convolutional layer to reduce channel dimensions, followed by two parallel branches to handle classification and regression tasks separately.

YOLOX adopts an anchor-free approach, eliminating predefined anchor boxes. By directly predicting the offset of the target center relative to the grid and the height and width of the target bounding box, it simplifies the prediction process, reduces the time consumption from hyperparameters and anchor box clustering, and removes restrictions on the detection area. To optimize the sample matching process, YOLOX uses the SimOTA [31] strategy, dynamically allocating the number of positive samples for different targets, avoiding extra parameter optimization, and improving detection speed and accuracy. Additionally, YOLOX improves bounding box localization accuracy by introducing the GIoU regression loss function [32] and enhances the model's robustness and efficiency by combining Mosaic and MixUp data augmentation and regularization techniques.

YOLOX is a faster and smaller model, balancing detection accuracy and speed, making it suitable for deployment on various devices. It outperforms previous best results on the COCO dataset [33]. However, its complex model structure and training process result in slower runtime on devices, and there is still room for improvement in small object detection.

3.7. YOLOv6

YOLOv6 [34] incorporates contemporary advanced network design, training strategies, and testing techniques into the algorithm. In terms of network structure, it introduces EfficientRep as the backbone, using different modules to construct the network depending on the model size, effectively balancing computational cost and accuracy. For the neck, YOLOv6 adopts an improved PAN structure as the foundation, and additionally uses RepBlock (suitable for small models) or CSPStackRep Block (used for large models) to replace the CSPBlock used in YOLOv5. The head part employs a hybrid-channel

DOI: 10.54254/2755-2721/71/20241642

strategy to build a more efficient decoupled head, further reducing computational cost and inference delay.

Regarding training strategies, YOLOv6 uses an anchor-free detection method and the Task Alignment Learning (TAL) algorithm [35] for dynamic allocation of positive samples, significantly accelerating the training speed. It also adopts VFL as the classification loss function and SIoU as the boundary regression loss function for supervised learning, speeding up network convergence and further improving network accuracy.

On the COCO dataset, YOLOv6 surpasses other models of the same scale in terms of both accuracy and speed. YOLOv6 is also dedicated to industrial scenarios, offering eight different model sizes to meet various application needs, greatly simplifying the deployment process.

3.8. YOLOv7

YOLOv7 [36] focuses on optimizing model structure and training processes. It introduces Extended Efficient Layer Aggregation Networks (E-ELAN), which use group convolution to expand the channels and cardinality of computational blocks, significantly reducing the model's parameters and computational load without substantial performance loss. Additionally, it proposes model scaling for concatenation-based models, considering both depth factor scaling of computational modules and width factor scaling of transition layers, thus reducing the drop in hardware utilization during the scaling process. Inspired by YOLOv6's RepConv, YOLOv7 employs RepConv without identity connections (RepConvN) to replan the reparameterized convolutional structure, enhancing detection accuracy.

In terms of label assignment strategy, YOLOv7 introduces Coarse for Auxiliary and Fine for Lead Loss, using fine labels generated by the Lead head to guide the learning of both the Lead head and the Auxiliary head. This enables the Lead head to more accurately perform object localization and classification, while the Auxiliary head enhances the training process of the Lead head, increasing the model's stability and robustness.

YOLOv7 surpasses the speed and accuracy of mainstream object detectors of its time. On the COCO dataset, YOLOv7 achieved the highest accuracy of 56.8% at an image processing speed of 30 FPS on a GPU V100.

3.9. YOLOv8

YOLOv8 [37] mainly draws on the advantages of models such as YOLOv5, YOLOv6, and YOLOX, and optimizes based on them. It provides a brand new state-of-the-art (SOTA) model and offers different scales of models based on scaling factors to support various tasks such as image classification, object detection, and image segmentation, and is also compatible with different hardware platforms.

YOLOv8 adopts the ELAN [38] concept from YOLOv7, replacing the original C3 structure with a more gradient-rich C2f structure in the YOLOv5 architecture. It also employs the Decouple-head and Anchor-free strategies, separating classification and regression tasks and removing the model's dependency on prior boxes, leading to faster convergence and higher detection rates. Additionally, YOLOv8 uses VFL Loss as the classification loss function and DFL loss and CIoU Loss as regression loss functions, improving detection performance. For label assignment strategy, YOLOv8 utilizes Task Alignment Learning (TAL) to match the most relevant samples and incorporate them into the loss function to enhance the model's robustness.

3.10. YOLOv9

YOLOv9 [39] addresses the shortcomings of traditional deep learning methods: in conventional deep networks, as input data passes through layers for feature extraction, significant data loss occurs, which is detrimental to maintaining image detail and integrity during the detection process.

To tackle this issue, YOLOv9 introduces the concept of Programmable Gradient Information (PGI), which mainly consists of three parts: the main branch, the Auxiliary Reversible Branch, and Multi-level Auxiliary Information. For the main branch, PGI only uses the main branch during inference, avoiding additional inference costs. To address the extra inference time caused by the reversible structure, the

DOI: 10.54254/2755-2721/71/20241642

Auxiliary Reversible Branch acts as an auxiliary reversible branch, using an auxiliary supervision mechanism to generate gradients to update the original information, thereby retaining important information while reducing inference time. To minimize loss during deep supervision, Multi-level Auxiliary Information introduces the concept of integrated networks, combining gradients returned by different prediction heads to assist in the training of the main branch. Additionally, YOLOv9 features a new network architecture, GELAN, which combines CSPNet and ELAN, balancing lightweight design, inference speed, and accuracy, while generalizing ELAN so that its convolutional layers can be replaced by any computational blocks.

The YOLOv9 model is lighter and more efficient. Compared to YOLOv8, YOLOv9 achieves better detection results on the COCO dataset, with a 49% reduction in parameters and a 43% reduction in computational load, yet still achieving a 0.6% improvement in accuracy.

3.11. YOLOv10

Previous versions of YOLO heavily relied on NMS technology for post-processing, significantly slowing down the model's inference efficiency. To address this issue, YOLOv10 [40] introduced Consistent Dual Assignments for NMS-free Training. This technique employs both one-to-one and one-to-many detection heads during training, but only uses the one-to-one detection head during inference. This approach retains the deep supervision benefits of one-to-many assignments and increases inference speed, allowing the model to be deployed end-to-end without additional inference costs. Additionally, YOLOv10 improves upon the shortcomings of previous YOLO versions, focusing on two areas: Efficiency driven model design and Accuracy driven model design.

In the Efficiency driven model design, methods such as Lightweight classification head, Spatial-channel decoupled downsampling, and Rank-guided block design reduce the computational cost and improve detection efficiency. In the Accuracy driven model design, techniques like Large-kernel convolution and Partial self-attention (PSA) reduce inference overhead, attempting to enhance model performance with minimal computational cost.

YOLOv10 comes in five versions: N, S, M, L, and X, to meet the application needs of different scenarios. On the COCO dataset, YOLOv10 aims to achieve state-of-the-art performance across various model sizes and attain low-latency end-to-end deployment compared to other YOLO versions.

4. Conclusion

This paper reviews the evolution of the YOLO algorithm from YOLOv1 to YOLOv10, thoroughly discussing the key improvements in network architecture, training strategies, and performance optimization across different versions. From the foundational infrastructure of YOLOv1 to the advanced technologies and architectural optimizations of YOLOv10, the YOLO series has continually enhanced detection speed and accuracy through innovation and the integration of advanced technologies, driving the development of real-time object detection.

By introducing techniques such as CSPNet, Anchor-free, data augmentation, and multi-scale training, the YOLO algorithm has progressively found a better balance between detection speed and accuracy, demonstrating excellent performance in processing real-time images and handling high-complexity scenarios. Furthermore, to cater to devices with varying computational capabilities, YOLOv5 and later versions offer models in multiple sizes, allowing flexible deployment of the YOLO algorithm across a range of devices to meet diverse application needs.

However, despite significant advancements, the YOLO series algorithms still face challenges, such as lower accuracy in small object detection, instability in performance in complex backgrounds, and weaker robustness under extreme environmental conditions. Future research can explore more effective feature fusion techniques, improved loss functions, and deeper network architecture optimizations to further enhance the performance and applicability of the YOLO algorithm. Overall, the YOLO series has become a vital tool in the field of object detection, and future efforts will continue to push this algorithm series towards higher performance, making it suitable for an even broader range of scenarios.

DOI: 10.54254/2755-2721/71/20241642

References

[1] Jiang, P., Ergu, D., Liu, F., Cai, Y. and Ma, B. (2022) A Review of Yolo Algorithm Developments. Procedia Computer Science, 199:1066–1073.

[2] Al-batat, R., Angelopoulou, A., Premkumar, S., Hemanth, J. and Kapetanios, E. (2022) An End-to-End Automated License Plate Recognition System Using YOLO Based Vehicle and License Plate Detection with Vehicle Classification. SENSORS, 22:9477.

[3] Bai, T. (2020) Analysis on Two-Stage Object Detection Based on Convolutional Neural Networkorks. 2020 INTERNATIONAL CONFERENCE ON BIG DATA & ARTIFICIAL INTELLIGENCE & SOFTWARE ENGINEERING (ICBASE 2020):321–325.

[4] Zou, Z., Chen, K., Shi, Z., Guo, Y. and Ye, J. (2023) Object Detection in 20 Years: A Survey. PROCEEDINGS OF THE IEEE, 111:257–276.

[5] Talaei Khoei, T., Ould Slimane, H. and Kaabouch, N. (2023) Deep Learning: Systematic Review, Models, Challenges, and Research Directions. NEURAL COMPUTING & APPLICATIONS, 35:23103–23124.

[6] Agaian, S., Tourshan, K. and Noonan, J.P. (2003) Parameterisation of Slant-Haar Transforms. IEE PROCEEDINGS-VISION IMAGE AND SIGNAL PROCESSING, 150:306–311.

[7] Ng, P.C. and Henikoff, S. (2003) SIFT: Predicting Amino Acid Changes That Affect Protein Function. NUCLEIC ACIDS RESEARCH, 31:3812–3814.

[8] Girshick, R., Donahue, J., Darrell, T. and Malik, J. (2014) Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR) :580–587.

[9] Girshick, R. (2015) Fast R-CNN. 2015 IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION (ICCV):1440–1448.

[10] He, K., Gkioxari, G., Dollar, P. and Girshick, R. (2020) Mask R-CNN. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, 42:386–397.

[11] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A.C. (2016) SSD: Single Shot MultiBox Detector. COMPUTER VISION - ECCV 2016,pp.21–37.

[12] Lin, T.Y., Goyal, P., Girshick, R., He, K. and Dollar, P. (2017) Focal Loss for Dense Object Detection. 2017 IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION (ICCV), pp.2999–3007.

[13] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016) You Only Look Once: Unified, Real-Time Object Detection.

[14] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going Deeper with Convolutions. 2015 IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR), pp.1–9.

[15] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Li, F.-F. (2009) ImageNet: A Large-Scale Hierarchical Image Database. CVPR: 2009 IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, VOLS 1-4, pp.248–255.

[16] Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J. and Zisserman, A. (2010) The Pascal Visual Object Classes (VOC) Challenge. INTERNATIONAL JOURNAL OF COMPUTER VISION, 88:303–338.

[17] Glorot, X., Bordes, A. and Bengio, Y. (2011) Deep Sparse Rectifier Neural Networks. Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp.315–323.

[18] Maas, A.L., Hannun, A.Y. and Ng, A.Y. Rectifier Nonlinearities Improve Neural Network Acoustic Models.

[19] Redmon, J. and Farhadi, A. (2017) YOLO9000: Better, Faster, Stronger. 30TH IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR 2017), pp.6517–6525.

[20] Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition.

[21] Redmon, J. and Farhadi, A. YOLOv3: An Incremental Improvement. http://arxiv.org/abs/1804.02767
[22] Lin, T.-Y., Dollar, P., Girshick, R., He, K., Hariharan, B. and Belongie, S. (2017) Feature Pyramid Networks for Object Detection. 30TH IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR 2017), pp.936–944.
[23] Bochkovskiy, A., Wang, C.Y. and Liao, H.Y.M. (2020) YOLOv4: Optimal Speed and Accuracy of Object Detection. http://arxiv.org/abs/2004.10934
[24] He, K., Zhang, X., Ren, S. and Sun, J. (2014) Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. COMPUTER VISION - ECCV 2014, PT III, pp.346–361.
[25] Liu, S., Qi, L., Qin, H., Shi, J. and Jia, J. (2018) Path Aggregation Network for Instance Segmentation. http://arxiv.org/abs/1803.01534
[26] Wang, C.Y., Liao, H.Y.M., Wu, Y.H., Chen, P.Y., Hsieh, J.W. and Yeh, I.H. (2020) CSPNet: A New Backbone That Can Enhance Learning Capability of CNN. 2020 IEEE/CVF CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION WORKSHOPS (CVPRW 2020), pp.1571–1580.
[27] Dubey, S.R., Singh, S.K. and Chaudhuri, B.B. (2022) Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark. NEUROCOMPUTING, 503:92–108.
[28] NELSON, J., SOLAWETZ, J. (2020) YOLOv5 Is Here: State-of-the-Art Object Detection at 140 FPS. Roboflow Blog. https://blog.roboflow.com/yolov5-is-here/
[29] Shorten, C. and Khoshgoftaar, T.M. (2019) A Survey on Image Data Augmentation for Deep Learning. JOURNAL OF BIG DATA, 6:60.
[30] Ge, Z., Liu, S., Wang, F., Li, Z. and Sun, J. (2021) YOLOX: Exceeding YOLO Series in 2021. http://arxiv.org/abs/2107.08430
[31] Ge, Z., Liu, S., Li, Z., Yoshie, O. and Sun, J. (2021) OTA: Optimal Transport Assignment for Object Detection. http://arxiv.org/abs/2103.14259
[32] Terven, J., Cordova-Esparza, D.M., Ramirez-Pedraza, A. and Chavez-Urbiola, E.A. (2023) Loss Functions and Metrics in Deep Learning. http://arxiv.org/abs/2307.02694
[33] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollar, P. and Zitnick, C.L. (2014) Microsoft COCO: Common Objects in Context. COMPUTER VISION - ECCV 2014, PT V, pp.740–755.
[34] Li, C., Li, L., Jiang, H., Weng, K., et al. (2022) YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. http://arxiv.org/abs/2209.02976
[35] Feng, C., Zhong, Y., Gao, Y., Scott, M.R. and Huang, W. (2021) TOOD: Task-Aligned One-Stage Object Detection. http://arxiv.org/abs/2108.07755
[36] Wang, C.Y., Bochkovskiy, A. and Liao, H.Y.M. (2022) YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. http://arxiv.org/abs/2207.02696
[37] Jocher, G., Chaurasia, A. and Qiu, J. (2023) Ultralytics YOLO. https://github.com/ultralytics/ultralytics
[38] Zhang, X., Zeng, H., Guo, S. and Zhang, L. (2022) Efficient Long-Range Attention Network for Image Super-Resolution. http://arxiv.org/abs/2203.06697
[39] Wang, C.Y., Yeh, I.H. and Liao, H.Y.M. (2024) YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. http://arxiv.org/abs/2403.13616
[40] Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J. and Ding, G. (2024) YOLOv10: Real-Time End-to-End Object Detection. http://arxiv.org/abs/2405.14458