Proceedings of the 3rd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/119/2025.21591

Evolution of YOLO: A Comparative Analysis of YOLOv5, YOLOv8, and YOLOv10

Jianing Zhang[1,a,*]

[1]*Computer Science and Technology, Nanjing University, Nanjing City, Jiangsu Province, 210023, China*
a. 221220129@smail.nju.edu.cn
**corresponding author*

Abstract: This paper presents a systematic comparative analysis of three versions of the YOLO (You Only Look Once) target detection algorithm - YOLOv5, YOLOv8 and YOLOv10. Through experiments on the VOC2012 dataset (converted to COCO format), this paper evaluates the versions in terms of multiple dimensions such as detection performance, inference speed and model complexity. The experimental results show that the detection accuracy and robustness significantly improve with version iteration and the mAP of v8 v10 is improved by 6.69% and 9.12% relative to v5, However, the number of model parameters increases by 68.98% and 48.66, and the FLOPS increases by 94.08% and 91.51%, respectively, which leads to an increased demand for computational resources and a slight decrease in inference speed compared to the old version, especially in practical application scenarios with limited resources.This paper not only demonstrates the continuous progress of the network structure and training strategy, but also explores the balance between performance and efficiency in real-time target detection, which provides references and insights for the future development of related technologies.

Keywords: YOLO, Target detection, Comparative analysis, Performance evaluation, Model complexity

1. Introduction

Target detection is one of the core tasks in computer vision, aiming to recognize and localize target objects in images or videos. This technology plays an important role in many practical applications, such as automatic driving, intelligent surveillance, medical image analysis, etc., and is an important force driving the development of artificial intelligence. With the continuous advancement of technology, target detection has experienced a leapfrog development from manual features to automatic feature extraction from early traditional methods to today's deep learning models, which has greatly improved its performance ability in complex scenes.

The development of target detection techniques can be traced back to the 1990s. Early conventional methods relied heavily on manually designed feature extractors and classifiers. Typical methods searched image regions one by one by sliding window technique, combined with feature extraction algorithms to generate target descriptions, which were then classified by classifiers. Among them, DPM (Deformable Part-based Model) is a representative method in this stage. However,

DOI: 10.54254/2755-2721/119/2025.21591

such methods have the limitations of high computational complexity, slow detection speed, and insufficient generalization ability in complex scenes.

Since 2012, the introduction of Convolutional Neural Networks (CNNs) has revolutionized the field of target detection.The R-CNN [1] family of algorithms proposed in 2014 combined region proposal with deep learning models for the first time, enabling end-to-end learning from feature extraction to target classification. Subsequently, Fast R-CNN and Faster R-CNN [2]further improved the detection efficiency by introducing region proposal network (RPN) and shared convolutional computation. In the same period, single-stage detectors such as YOLO (You Only Look Once) [3] and SSD (Single Shot MultiBox Detector) [4] transformed the target detection task into a regression problem, which maintains a high accuracy rate while significantly improved detection speed, laying the foundation for real-time applications. In recent years, Transformer [5] based detection methods (e.g., DETR) [6] have realized a more end-to-end detection process by introducing the attention mechanism; the rise of lightweight models (e.g., YOLO-NAS and NanoDet) has further extends the application of target detection in edge devices.

As a representative algorithm in the field of real-time target detection, YOLO (You Only Look Once) has received widespread attention for its efficient design and excellent performance. The core idea of YOLO is to transform the task of target detection into a regression problem, and simultaneously accomplish target classification and bounding box prediction in a single network inference. Compared with traditional two-stage detection methods (e.g., Faster R-CNN), YOLO exhibits significant advantages: it completes target classification and localization in one go without the need for additional region proposing process, which greatly improves the detection efficiency; it realizes real-time detection capability with a lightweight design, and the latest version, such as YOLOv8, further optimizes the accuracy and speed; it avoids the localization problem of the traditional methods through global feature extraction, which makes it more efficient and faster than the traditional methods. YOLOv8 is further optimized in terms of accuracy and speed, and avoids the localization problem of traditional methods through global feature extraction, which makes it more accurate in target recognition in complex scenes. However, YOLO also has some limitations, such as insufficient detection of small objects, large localization error in complex scenes, and limited adaptability to special targets. To address these problems, researchers have continuously optimized the performance of the algorithm by means of multi-scale feature fusion, improved non-maximum suppression methods (e.g., Soft-NMS), and adaptive anchor frame design.

As soon as the current development of YOLO has made significant progress, there is a lack of comprehensive paper reviews for its latest versions. Therefore, in this paper, we will systematically analyze the latest development of YOLO series algorithms, focusing on the architectural improvement and performance enhancement of YOLOv5, YOLOv8, and YOLOv10, and make a detailed comparison of their advantages and disadvantages in different application scenarios. By summarizing the improvement ideas and application performance of these algorithms, we hope to provide new ideas for future research on target detection technology.

2. Related Work

YOLOv5 [7] is a lightweight target detection algorithm from the Ultralytics team. Compared to YOLOv4, it introduces a more efficient network architecture and is implemented closer to the needs of industrial applications. Its key features include:

Lightweight design: Use smaller model parameters to make it suitable for resource-limited environments.

Data Enhancement Technique: Mosaic data enhancement method is introduced to improve the generalization ability of the model.

DOI: 10.54254/2755-2721/119/2025.21591

Supports multiple resolution inputs:Improves the balance between inspection accuracy and speed by automatically adjusting the input size.

Although YOLOv5 is not from the original YOLO author team, its efficiency and flexibility make it widely used in industry.

YOLOv8 [8] is another important upgrade version of YOLO series, which further optimizes the model structure based on YOLOv5. The main improvement points include:

Improved feature extraction network: deeper network architecture and multi-scale feature fusion techniques are used to enhance the detection of small targets.

More flexible training framework: supports more hyper-parameter configurations, facilitating users to optimize the model on different datasets.

Improved balance of speed and accuracy: better performance on COCO dataset than YOLOv5.

The goal of YOLOv8 is to further improve the robustness and adaptability of the model to perform better in complex scenarios.

YOLOv10 [9] is the latest version of the YOLO series, featuring significant innovations in both network design and optimization methods. Its key features include:

Introduction of a new Transformer mechanism: the model's ability to model long-range dependencies is improved by integrating Transformer into target detection.

More efficient training strategy: by improving the optimization algorithm, it significantly reduces the training time while ensuring the high accuracy of the model.

Adapt to more complex scenarios: YOLOv10 is especially optimized for challenging tasks such as multi-targeting and high occlusion.

Although YOLOv10 is not yet widely available in industry, its performance under laboratory conditions has demonstrated its potential for future target detection research.

3. Methodology

3.1. Introduction to the VOC2012 Dataset

VOC2012 [10] is one of the widely used standard datasets in the field of target detection, which is mainly used for evaluating the performance of models in target detection, image segmentation and classification tasks. Released by the PASCAL VOC project, this dataset, with its rich labels and diverse scenarios, has become an important benchmark for research on target detection algorithms.

Overview of data sets - The VOC2012 dataset contains 20 categories of object labels, including humans, animals, transportation, and household items, covering common targets in daily life. There are a total of 17,125 images in the dataset.

Each image may contain one or more targets, and the diversity and complexity of the data presents a challenge to the generalization ability of the model.

Characteristics of the dataset - Diverse Scenarios: The dataset contains a variety of complex scenarios, such as occlusion, light variations, and multi-target distribution, which simulate the challenges of real applications.

Comprehensive annotation: Bounding Box, target category labeling, and pixel-level segmentation masks are provided, providing rich resources for target detection and semantic segmentation algorithm research.

Generality:VOC2012 is widely used as a performance benchmark for target detection algorithms and is an indispensable tool in model comparison.

Importance of dataset - The importance of VOC2012 is not only reflected in its widespread use, but also provides standardized evaluation metrics (e.g., mAP and Precision-Recall curves) for target detection studies. By testing on the VOC2012 dataset, the performance of the model on small-scale

DOI: 10.54254/2755-2721/119/2025.21591

and high-diversity data can be effectively evaluated, providing a basis for further improvement of the algorithm.

Dataset processing - The YOLO algorithm uses COCO format dataset for training, therefore, in this experiment, we first converted the VOC2012 dataset to COCO format to ensure compatibility with the YOLO model. The dataset is divided in the following proportions: 80% for the training set (train), 10% for the validation set (val), and 10% for the test set (test). This division ensures that the training set contains enough samples to train the model, while the validation and test sets provide criteria for evaluating the generalization ability of the model.

3.2. Model Configurations

In this experiment, we chose YOLOv5, YOLOv8 and YOLOv10 for comparison training. The training of all models is set to 100 rounds (epochs), and the batch size of each batch is uniformly 16. In addition, we maintain consistent hyper-parameter settings for the training configuration of each model to ensure a fair comparison. All models use COCO-format datasets, and data enhancement strategies, such as horizontal flipping, scaling, and color adjustment, are used during the training process to improve the generalization ability of the models. The training environment is a single GPU setup, implemented using the PyTorch framework, and model optimization and tuning are performed based on predefined profiles.

3.3. Experimental Setup

Hardware Configuration - As shown in Table 1Hardware Configuration Table, this experiment relies on a high-performance hardware platform, including the RTX 4090 GPU with 24GB of video memory, a powerful multi-threaded CPU 16 vCPU Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz and 120GB of high-capacity memory to ensure that the experiment can be completed in an efficient, stable environment. The GPU's computational power directly affects the training speed of the deep neural network, while the CPU is responsible for efficiently handling data loading and multi-threaded tasks, and the sufficient memory capacity provides a guarantee for pre-processing and caching of large-scale data.

Table 1: Hardware Configuration.

Name	Version	Effect
GPU	RTX 4090(24GB graphics memory)	Supports large-scale image processing and deep neural network training
CPU	16 vCPU Intel(R) Xeon(R)Platinum 8352V CPU @ 2.10GHz	Efficient handling of multi-threaded tasks and ensure smooth data loading process
RAM	120GB	Ensure that there is enough memory for efficient data caching and preprocessing during the training process.

Software Configuration - As shown in Table 2Software Configuration Table, a series of efficient and flexible software tools such as PyTorch, cuda, numpy, etc. were used in the experiment to support the training and validation of the model.

DOI: 10.54254/2755-2721/119/2025.21591

Table 2: Software Configuration

Name	Version	Effect
Python	3.12	Support for YOLO models and other related deep learning tool libraries
PyTorch	2.3.0	Supports GPU-accelerated training with flexible neural network definition and optimization
Cuda	12.1	Supports efficient matrix operations and massively parallel computation using GPUs
Torchvision	0.11.1	For processing image and video data, as well as model-related functions
Numpy	1.21.2	Perform matrix manipulation and data processing
Pandas	1.3.3	For data analysis and processing
Matplotlib	3.4.3	For visualizing loss and accuracy changes during training
Opencv-python	4.5.3	For image processing and enhancement.

Hyperparameter Setting - The setting of hyperparameters plays a crucial role in the training of deep learning models, which directly affects the convergence speed and final performance of the model.As shown in Table 3 hyperparameter setting table: this experiment sets the basic parameters of training, including the number of training rounds (epochs), the number of training samples in each batch (batch size), the size of the input image (imgsz), and the learning rate (learning rate). Among them, learning rate is one of the most important parameters in the optimization process, which affects the step size of gradient descent. Momentum (momentum) is used to accelerate the gradient update and avoid local oscillations, while weight decay (weight decay) helps to prevent overfitting. The proper selection of these hyperparameters ensures that the model can complete the training task efficiently and stably.

Table 3: Hyperparameter Setting.

Name	Value
Epochs	100
Batch_size	16
Imgsz	640
Learning rate	0.01
Momentum	0.9370
Weight_decay	0.0005

3.4. Evaluation Metrics

F1 Score - Definition: F1 Score is the reconciled average of Precision and Recall. F1 Score takes both Precision and Recall into account, and in the case of unbalanced data or when False Positives and False Negatives need to be considered in combination, F1 Score is able to provide a more comprehensive performance evaluation.

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{1}$$

mAP@0.5 - Definition:mAP refers to mean Average Precision, which is used to comprehensively evaluate the model's detection performance across all categories. For each category, the Average Precision (AP) of the category is calculated first, and then the APs of all categories are averaged to

DOI: 10.54254/2755-2721/119/2025.21591

get mAP. 0.5 means that the IoU (Intersection over Union) threshold of 0.5 is used to calculate the AP.

$$IoU=\frac{|B_p \cap B_{gt}|}{|B_p \cup B_{gt}|} \tag{2}$$

$$AP=\sum_n (r_n - r_{n-1})\ p_{interp}(r_n) \tag{3}$$

where r_n is the first n Recall value, and $p_{interp}(r_n)$ is the interpolation accuracy at Recall value r_n.

$$mAP=\frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4}$$

FPS - Definition: FPS indicates the number of frames that the system can process (or display) in one second. It is an important indicator for evaluating real-time performance. Higher values mean faster processing.

$$FPS=\frac{N}{T} \tag{5}$$

where N is the total number of frames and T is the total time.

FLOPS - Definition: FLOPS (Floating Point Operations Per Second) refers to the number of floating point operations performed per second and is used to measure the computational performance of a computing device or model.

$$FLOPS=\frac{N}{T} \tag{6}$$

where N is the total number of floating point operations and T is the total time.

4. Experiments and Results

4.1. Detection Performance

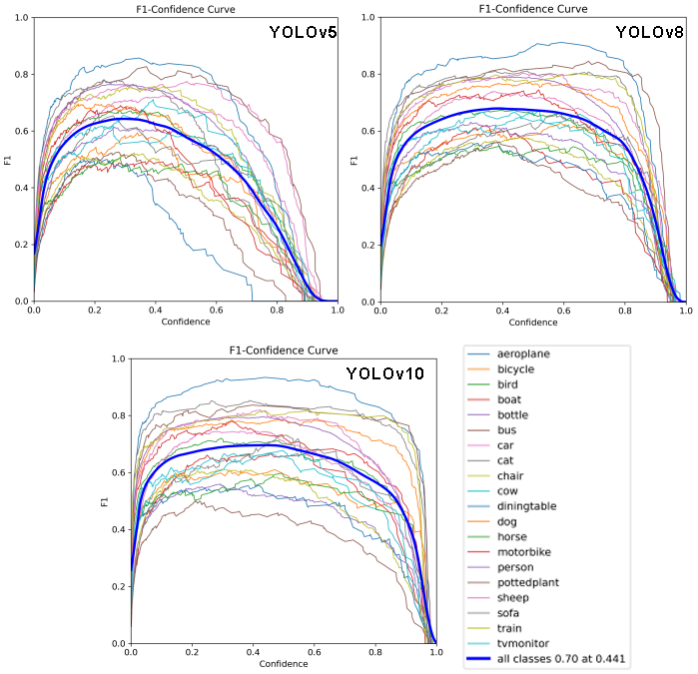

Figure 1: YOLOv5, 8, 10 F1-confidence curve

DOI: 10.54254/2755-2721/119/2025.21591

As shown in Figure 1 F1-confidence curve, the yolov5, v8, v10 F1 score great values gradually increase. It can be analyzed and interpreted from the following aspects:

Improvement in Combined Performance: The extremely large increase in F1 Score indicates that the new version of the model performs better in terms of the combination of Precision and Recall when it reaches the best equilibrium point.

False Positives and False Negatives Reduction: The large enhancement indicates that both False Positives and False Negatives are reduced and the model is more reliable in critical tasks such as autonomous driving or security surveillance.

4.2. Accuracy Metrics

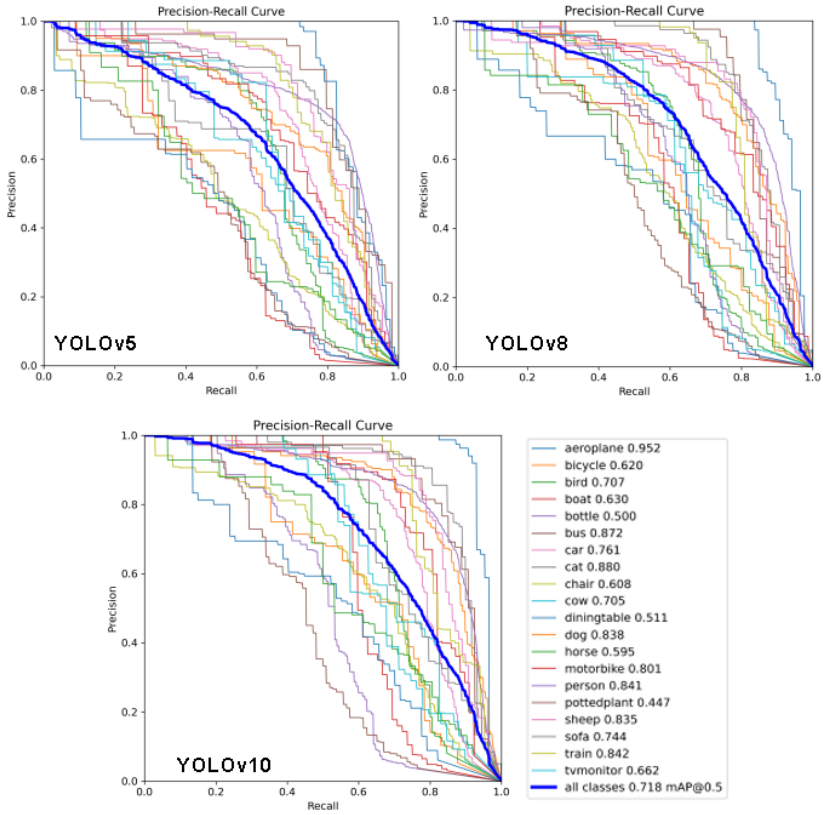

Figure 2: YOLOv5, 8, 10 PR curve

As shown in Figure 2 PR curve,the gradual incremental increase in the value of mAP@0.5 can be analyzed and interpreted in the following ways:

Meaning of incremental increase in value of mAP@0.5 - Illustrating the continuous improvement in model performance: mAP@0.5 is a core metric in the target detection task, indicating the average accuracy of the model at an IoU threshold of 0.5. The gradual increment indicates that the model's ability to detect targets has been optimized in each generation.

Improved balance of False Positives and False Negatives: False Positives and False Negatives are gradually reduced as the model is updated.

Reasons of incremental increase in value of mAP@0.5 - Model Structure and Improvements: The new version employs a deeper and stronger feature extraction network that captures a richer set of target features, resulting in a higher overlap between the detection frame and the true frame.

Training Strategies and Data Processing: Richer or more appropriate data augmentation methods allow the model to see a greater variety of targets and scenarios, improving the robustness of the

DOI: 10.54254/2755-2721/119/2025.21591

model under different conditions and thus improving the mAP. loss function and optimization algorithms are improved to help the model learn a more accurate frame localization.

Anchor Design and Matching Strategy: The new version has optimized the design of the ANCHOR box and the matching strategy, which makes the pre-checked box match the actual target better and improves the mAP@0.5.

Other Optimizations The optimization of the non-maximal suppression (NMS) strategy improves the final mAP@0.5 by better filtering redundant frames and retaining high quality predictions.

4.3. Inference speed

Table 3: Inference speed

Model	FPS
V5	46.55
V8	39.59
V10	33.16

As shown in Table 4Training Speed Table: the inference speeds (FPS) of YOLOv5, YOLOv8 and YOLOv10 are 46.55, 39.59 and 33.16, respectively. From the results, it can be seen that the inference speeds gradually decrease as the model versions are updated. This may be due to the fact that YOLOv10 introduces more complex network structures or additional optimization strategies to improve the detection accuracy at the expense of some inference speed. Nevertheless, YOLOv10 still maintains high real-time performance and can meet the needs of most practical application scenarios.

4.4. Model complexity

Table 4: Model complexity

Model	FLOPS	Parameters
V5	36.51GFLOPS	1.87M
V8	70.86GFLOPS	3.16M
V10	69.92GFLOPS	2.78M

As shown in the table 5Model Complexity Table, the model complexity (measured in FLOPS) of YOLOv5, YOLOv8 and YOLOv10 are 36.51GFLOPS, 70.86GFLOPS and 69.92GFLOPS, with parameter counts of 1.87M, 3.16M and 2.78M, respectively. As can be seen from the results. YOLOv8 and YOLOv10 are significantly more complex than YOLOv5, which may be due to the introduction of deeper network structures, more parameters or more complex feature extraction modules. Although the FLOPS of YOLOv10 is slightly lower than that of YOLOv8, it strikes a better balance between accuracy and speed, indicating that YOLOv10 has made some progress in optimizing the efficiency of the model.

4.5. Performance in target categories

As shown in Fig. 2,v8 v10 there is a significant improvement in the recognition ability for small objects and fewer sample categories.

Improved small object recognition: detection accuracy is improved in high-density scenes or small target detection tasks

Improvement in less-sample category recognition: in datasets with long-tailed distributions (e.g., medical image categorization or rare species detection), the improvement in less-sample categories means that the model is more adaptable to cold categories.

DOI: 10.54254/2755-2721/119/2025.21591

5. Conclusion

This paper presents a comprehensive comparative analysis of the latest three versions of the YOLO family of algorithms - YOLOv5, YOLOv8 and YOLOv10. Through a series of experiments on the VOC2012 dataset (converted to COCO format), this paper evaluates and compares the versions in detail in terms of multiple dimensions such as detection performance, accuracy metrics, inference speed, and model complexity. The main conclusions are as follows:

Firstly, as the algorithm version evolves, key metrics such as F1 score and mAP@0.5 are gradually improved, indicating that the new version has significantly improved the accuracy and robustness of target detection as well as the ability to detect small objects and fewer sample categories. Secondly, in terms of inference speed, YOLOv5 has the highest FPS, while YOLOv10, despite its lower inference speed, performs better in detection accuracy and complex scene adaptability, reflecting a compromise between accuracy and real-time performance. Last, the comparison of model complexity shows that YOLOv8 and YOLOv10 adopt a deeper and more complex network structure, and despite the increase in the number of parameters and FLOPS, this design effectively improves the detection results, especially showing advantages in the complex background and small target detection tasks.

Overall, this study reveals that the YOLO series of algorithms inevitably trade-off between computational complexity and inference speed while continuously pursuing high accuracy.YOLOv10 achieves a better balance between accuracy and speed by introducing the Transformer mechanism and improving the training strategy, which provides a valuable reference for the development of future target detection algorithms. Future work can further explore the lightweight design, accelerated inference, and applicability in more scenarios to promote the popularization and development of real-time target detection technology in practical applications.

References

[1] *Girshick, R.B., Donahue, J., Darrell, T., & Malik, J. (2013). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 580-587.*

[2] *Redmon, J., Divvala, S.K., Girshick, R.B., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 779-788.*

[3] *Ren, Shaoqing et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." IEEE transactions on pattern analysis and machine intelligence vol. 39,6 (2017): 1137-1149.*

[4] *Liu, W. et al. (2016). SSD: Single Shot MultiBox Detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, 14, 21-37*

[5] *Vaswani, Ashish et al. "Attention is All you Need." Neural Information Processing Systems (2017).*

[6] *Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. Springer. p 213–229.*

[7] *Glenn Jocher. YOLOv5 by Ultralytics, May 2020.*

[8] *Glenn Jocher, Ayush Chaurasia, and Jing Qiu. YOLO by Ultralytics, January 2023.*

[9] *Wang, Ao et al. "YOLOv10: Real-Time End-to-End Object Detection." ArXiv abs/2405.14458 (2024): n. pag*

[10] *Mark. Everingham, Luc. Van Gool, Chris Williams, JJohn. Winn, and AAndrew. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results.*