YOLOv10: Real-Time End-to-End Object Detection

YOLOv10：实时端到端目标检测

Ao Wang [1] Hui Chen [2*] Lihao Liu [1] Kai Chen [1]

王奥 [1] 陈辉 [2*] 刘立昊 [1] 陈凯 [1]

Zijia Lin [1] Jungong Han [3] Guiguang Ding [1*]

林子佳 [1] 韩俊功 [3] 丁贵广 [1*]

[1] School of Software,Tsinghua University [2] BNRist,Tsinghua University

[1] 清华大学软件学院 [2] 清华大学北京信息科学与技术国家研究中心

[3] Department of Automation,Tsinghua University

[3] 清华大学自动化系

wa22@mails.tsinghua.edu.cn huichen@mail.tsinghua.edu.cn linzijia07@tsinghua.org.cn
{louisliu2048,chenkai2010.9,jungonghan77}@gmail.com dinggg@tsinghua.edu.cn

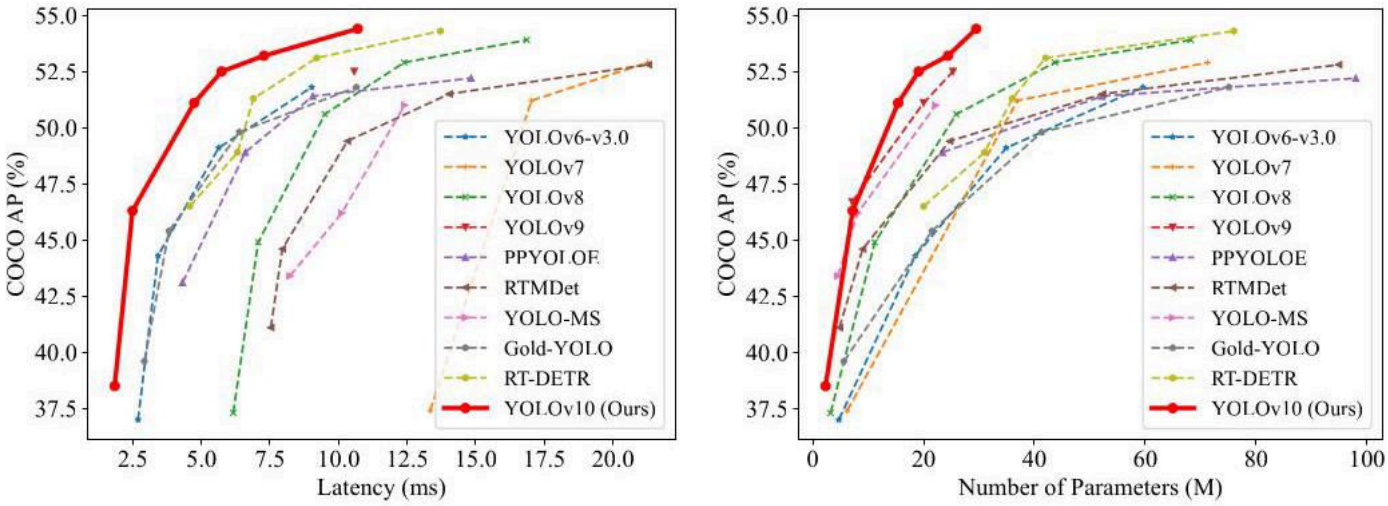

Figure 1: Comparisons with others in terms of latency-accuracy (left) and size-accuracy (right) trade-offs. We measure the end-to-end latency using the official pre-trained models.

图1：在延迟-准确率（左）和模型大小-准确率（右）权衡方面与其他方法的对比。我们使用官方预训练模型测量端到端延迟。

Abstract

摘要

Over the past years, YOLOs have emerged as the predominant paradigm in the field of real-time object detection owing to their effective balance between computational cost and detection performance. Researchers have explored the architectural designs, optimization

objectives, data augmentation strategies, and others for YO-LOs, achieving notable progress. However, the reliance on the non-maximum suppression (NMS) for post-processing hampers the end-to-end deployment of YOLOs and adversely impacts the inference latency. Besides, the design of various components in YOLOs lacks the comprehensive and thorough inspection, resulting in noticeable computational redundancy and limiting the model's capability. It renders the suboptimal efficiency, along with considerable potential for performance improvements. In this work, we aim to further advance the performance-efficiency boundary of YOLOs from both the post-processing and the model architecture. To this end, we first present the consistent dual assignments for NMS-free training of YOLOs, which brings the competitive performance and low inference latency simultaneously. Moreover, we introduce the holistic efficiency-accuracy driven model design strategy for YOLOs. We comprehensively optimize various components of YOLOs from both the efficiency and accuracy perspectives, which greatly reduces the computational overhead and enhances the capability. The outcome of our effort is a new generation of YOLO series for real-time end-to-end object detection, dubbed YOLOv10. Extensive experiments show that YOLOv10 achieves the state-of-the-art performance and efficiency across various model scales. For example,our YOLOv10-S is $1.8\times$ faster than RT-DETR-R18 under the similar AP on COCO,meanwhile enjoying $2.8\times$ smaller number of parameters and FLOPs. Compared with YOLOv9-C, YOLOv10-B has 46% less latency and 25% fewer parameters for the same performance. Code and models are available at https://github.com/THU-MIG/yolov10

过去数年间，YOLO系列因其在计算成本与检测性能间的有效平衡，已成为实时目标检测领域的主导范式。研究者们从网络架构设计、优化目标、数据增强策略等方面对YOLO进行探索，取得了显著进展。然而，依赖非极大值抑制（NMS）的后处理阻碍了YOLO的端到端部署，并对推理延迟产生负面影响。此外，YOLO各组件设计缺乏全面彻底的检验，导致明显计算冗余并限制模型能力，使得效率未达最优且存在巨大改进潜力。本工作旨在从后处理和模型架构两方面推进YOLO的性能-效率边界。为此，我们首先提出用于无NMS训练的YOLO一致性双重分配策略，同步实现优越性能与低推理延迟。进而提出面向YOLO的全方位效率-精度驱动模型设计策略，从效率与精度双重视角全面优化各组件，显著降低计算开销并增强模型能力。最终产出新一代实时端到端目标检测YOLO系列——YOLOv10。大量实验表明，YOLOv10在不同模型规模下均实现最优性能与效率。例如，在COCO数据集相似AP下，YOLOv10-S较RT-DETR-R18快$1.8\times$，同时具有$2.8\times$更少参数量和FLOPs；与YOLOv9-C相比，YOLOv10-B在同等性能下延迟降低46%、参数减少25%。代码与模型详见https://github.com/THU-MIG/yolov10

*Corresponding author.

*通讯作者

1 Introduction

1 引言

Real-time object detection has always been a focal point of research in the area of computer vision, which aims to accurately predict the categories and positions of objects in an image under low latency. It is widely adopted in various practical applications, including autonomous driving [3], robot navigation [12], and object tracking [72], etc. In recent years, researchers have concentrated on devising CNN-based object detectors to achieve real-time detection [19, 23, 48, 49, 50, 57, 13]. Among them, YOLOs have gained increasing popularity due to their adept balance between performance and efficiency [2, 20, 29, 20, 21, 65, 60, 70, 8, 71, 17, 29]. The detection pipeline of YOLOs consists of two parts: the model forward process and the NMS post-processing. However, both of them still have deficiencies, resulting in suboptimal accuracy-latency boundaries.

实时目标检测始终是计算机视觉领域的研究焦点，其目标是在低延迟下准确预测图像中物体的类别与位置。该技术被广泛应用于自动驾驶[3]、机器人导航[12]、目标跟踪[72]等实际场景。近年来，研究者致力于设计基于CNN的实时检测器[19,23,48-50,57,13]，其中YOLO系列因卓越的效能平衡特性广受欢迎[2,20,29,20,21,65,60,70,8,71,17,29]。YOLO检测流程包含模型前向过程与NMS后处理两部分，但二者均存在缺陷，导致现有准确率-延迟边界未达最优。

Specifically, YOLOs usually employ one-to-many label assignment strategy during training, whereby one ground-truth object corresponds to multiple positive samples. Despite yielding superior performance, this approach necessitates NMS to select the best positive prediction during inference. This slows down the inference speed and renders the performance sensitive to the hyperparameters of NMS, thereby preventing YOLOs from achieving optimal end-to-end deployment [78]. One line to tackle this issue is to adopt the recently introduced end-to-end DETR architectures [4, 81, 73, 30, 36, 42, 67]. For example, RT-DETR [78] presents an efficient hybrid encoder and uncertainty-minimal query selection, propelling DETRs into the realm of real-time applications. Nevertheless, when considering only the forward process of model during deployment, the efficiency of the DETRs still has room for improvements compared with YOLOs. Another line is to explore end-to-end detection for CNN-based detectors, which typically leverages one-to-one assignment strategies to suppress the redundant predictions [6, 55, 66, 80, 17]. However, they usually introduce additional inference overhead or achieve suboptimal performance for YOLOs.

具体而言，YOLO系列模型在训练阶段通常采用一对多的标签分配策略，即每个真实目标对应多个正样本。虽然这种方法能带来卓越的性能，但在推理阶段需要依赖非极大值抑制(NMS)来筛选最佳预测结果。这不仅降低了推理速度，还使得模型性能对NMS的超参数设置敏感，从而阻碍了YOLO实现最优的端到端部署[78]。解决该问题的研究方向之一是采用新兴的端到端DETR架构[4,81,73,30,36,42,67]。例如RT-DETR[78]通过提出高效混合编码器和不确定性最小化查询选择机制，将DETR系列推向了实时应用领域。然而仅从模型前向计算效率来看，DETR相比YOLO仍有提升空间。另一研究方向是探索基于CNN的端到端检测器，这类方法通常采用一对一标签分配策略来抑制冗余预测[6,55,66,80,17]，但往往会引入额外计算开销或导致YOLO性能次优。

Furthermore, the model architecture design remains a fundamental challenge for YOLOs, which exhibits an important impact on the accuracy and speed [50, 17, 71, 8]. To achieve more efficient and effective model architectures, researchers have explored different design strategies. Various primary computational units are presented for the backbone to enhance the feature extraction ability, including DarkNet [48, 49, 50], CSPNet [2], EfficientRep [29] and ELAN [62, 64], etc. For the neck, PAN [37], BiC [29], GD [60] and RepGFPN [71], etc., are explored to enhance the multi-scale feature fusion. Besides, model scaling strategies [62, 61] and re-parameterization [11, 29] techniques are also investigated. While these efforts have achieved notable advancements, a comprehensive inspection for various components in YOLOs from both the efficiency and accuracy perspectives is still lacking. As a result, there still exists considerable computational redundancy within YOLOs, leading to inefficient parameter utilization and suboptimal efficiency. Besides, the resulting constrained model capability also leads to inferior performance, leaving ample room for accuracy improvements.

此外，模型架构设计始终是YOLO面临的核心挑战，其显著影响检测精度与速度[50,17,71,8]。为构建更高效的模型架构，研究者探索了多种设计策略：在骨干网络方面，DarkNet[48,49,50]、CSPNet[2]、EfficientRep[29]和ELAN[62,64]等基础计算单元被提出以增强特征提取能力；在特征融合模块，PAN[37]、BiC[29]、GD[60]和RepGFPN[71]等结构被用于优化多尺度特征融合；此外还研究了模型缩放策略[62,61]和重参数化技术[11,29]。尽管这些工作取得显著进展，但从效率与精度双重视角对YOLO各组件进行系统分析的研究仍然匮乏，导致模型存在大量计算冗余、参数利用率低下等问题。受限的模型能力也制约了检测精度，存在较大优化空间。

In this work, we aim to address these issues and further advance the accuracy-speed boundaries of YOLOs. We target both the post-processing and the model architecture throughout the detection pipeline. To this end, we first tackle the problem of redundant predictions in the post-processing by presenting a consistent dual assignments strategy for NMS-free YOLOs with the dual label assignments and consistent matching metric. It allows the model to enjoy rich and harmonious supervision during training while eliminating the need for NMS during inference, leading to competitive performance with high efficiency. Secondly, we propose the holistic efficiency-accuracy driven model design strategy for the model architecture by performing the comprehensive inspection for various components in YOLOs. For efficiency, we propose the lightweight classification head, spatial-channel decoupled downsampling, and rank-guided block design, to reduce the manifested computational redundancy and achieve more efficient architecture. For accuracy, we explore the large-kernel convolution and present the effective partial self-attention module to enhance the model capability, harnessing the potential for performance improvements under low cost. Based on these approaches, we succeed in achieving a new family of real-time end-to-end detectors with different model scales,i.e.,YOLOv10-N / S / M / B / L / X. Extensive experiments on standard benchmarks for object detection, i.e., COCO [35], demonstrate that our YOLOv10 can significantly outperform previous state-of-the-art models in terms of computation-accuracy trade-offs across various model scales. As shown in Fig. 1,our YOLOv10-S / X are $1.8\times$ / $1.3\times$ faster than RT-DETR-R18 / R101, respectively, under the similar performance. Compared with YOLOv9-C, YOLOv10-B achieves a 46% reduction in latency with the same performance. Moreover, YOLOv10 exhibits highly efficient parameter utilization. Our YOLOv10-L / X outperforms YOLOv8-L / X by 0.3 AP and $0.5\mathrm{AP}$,with $1.8\times$ and $2.3\times$ smaller number of parameters,respectively. YOLOv10-M achieves the similar AP compared with YOLOv9-M / YOLO-MS, with 23% / 31% fewer parameters, respectively We hope that our work can inspire further studies and advancements in the field.

本研究旨在解决上述问题，突破YOLO系列精度-速度的边界。我们从后处理流程和模型架构两个维度进行创新：首先针对后处理冗余问题，提出基于双重标签分配和一致性匹配度量的双路径分配策略，使模型在训练阶段获得充分监督的同时，完全摆脱推理时对NMS的依赖，实现高效优质的检测性能。其次提出全栈式效率-精度驱动的模型设计策略，通过系统分析YOLO各组件：在效率方面，设计轻量化分类头、空间-通道解耦下采样和秩导向模块结构，显著降低计算冗余；在精度方面，探索大核卷积和高效局部自注意力模块，以较低成本充分释放模型潜力。基于这些方法，我们成功构建了不同规模的端到端实时检测器系列YOLOv10-N/S/M/B/L/X。COCO[35]基准测试表明，YOLOv10在所有模型规模下均显著超越现有最优模型。如图1所示，在同等精度下，YOLOv10-S/X分别比RT-DETR-R18/R101快$1.8\times$ / $1.3\times$；相比YOLOv9-C，YOLOv10-B在保持性能同时延迟降低46%。参数效率方面，YOLOv10-L/X以$1.8\times$和$2.3\times$更少的参数量，分别超越YOLOv8-L/X达0.3AP和$0.5\mathrm{AP}$；YOLOv10-M在参数量减少23%/31%的情况下，性能与YOLOv9-M/YOLO-MS持平。期待本工作能为领域研究带来新启示。

2 Related Work

2 相关工作

Real-time object detectors. Real-time object detection aims to classify and locate objects under low latency, which is crucial for real-world applications. Over the past years, substantial efforts have been directed towards developing efficient detectors [19, 57, 48, 34, 79, 75, 32, 31, 41]. Particularly, the YOLO series [48, 49, 50, 2, 20, 29, 62, 21, 65] stand out as the mainstream ones. YOLOv1, YOLOv2, and YOLOv3 identify the typical detection architecture consisting of three parts, i.e., backbone, neck, and head [48, 49, 50]. YOLOv4 [2] and YOLOv5 [20] introduce the CSPNet [63] design to replace DarkNet [47], coupled with data augmentation strategies, enhanced PAN, and a greater variety of model scales, etc. YOLOv6 [29] presents BiC and SimCSPSPPF for neck and backbone, respectively, with anchor-aided training and self-distillation strategy. YOLOv7 [62] introduces E-ELAN for rich gradient flow path and explores several trainable bag-of-freebies methods. YOLOv8 [21] presents C2f building block for effective feature extraction and fusion. Gold-YOLO [60] provides the advanced GD mechanism to boost the multi-scale feature fusion capability. YOLOv9 [65] proposes GELAN to improve the architecture and PGI to augment the training process.

实时目标检测器。实时目标检测旨在低延迟条件下对物体进行分类与定位，这对实际应用至关重要。近年来，研究者们投入大量精力开发高效检测器[19, 57, 48, 34, 79, 75, 32, 31, 41]。其中YOLO系列[48, 49, 50, 2, 20, 29, 62, 21, 65]尤为突出成为主流方案。YOLOv1/v2/v3确立了由骨干网络、颈部结构和检测头三部分组成的经典架构[48,49,50]。YOLOv4[2]和YOLOv5[20]采用CSPNet[63]设计替代DarkNet[47]，结合数据增强策略、改进的PAN结构及多样化模型尺度等。YOLOv6[29]分别为颈部和骨干网络引入BiC与SimCSPSPPF模块，配合锚框辅助训练和自蒸馏策略。YOLOv7[62]提出

E-ELAN结构增强梯度流路径，并探索多种可训练免费技巧。YOLOv8[21]设计C2f模块实现高效特征提取与融合。Gold-YOLO[60]通过先进GD机制提升多尺度特征融合能力。YOLOv9[65]提出GELAN改进架构，采用PGI增强训练过程。

End-to-end object detectors. End-to-end object detection has emerged as a paradigm shift from traditional pipelines, offering streamlined architectures [53]. DETR [4] introduces the transformer architecture and adopts Hungarian loss to achieve one-to-one matching prediction, thereby eliminating hand-crafted components and post-processing. Since then, various DETR variants have been proposed to enhance its performance and efficiency [42, 67, 56, 30, 36, 28, 5, 77, 82]. Deformable-DETR [81] leverages multi-scale deformable attention module to accelerate the convergence speed. DINO [73] integrates contrastive denoising, mix query selection, and look forward twice scheme into DETRs RT-DETR [78] further designs the efficient hybrid encoder and proposes the uncertainty-minimal query selection to improve both the accuracy and latency. Another line to achieve end-to-end object detection is based CNN detectors. Learnable NMS [24] and relation networks [26] present another network to remove duplicated predictions for detectors. OneNet [55] and DeFCN [66] propose one-to-one matching strategies to enable end-to-end object detection with fully convolutional networks. $\mathrm{FCOS}_{\mathrm{pss}}$ [80] introduces a positive sample selector to choose the optimal sample for prediction.

端到端目标检测器。端到端检测技术颠覆了传统流程，提供更简洁的架构[53]。DETR[4]首次引入Transformer结构，采用匈牙利损失实现一对一匹配预测，从而消除人工设计组件和后处理环节。此后研究者提出多种改进方案[42, 67, 56, 30, 36, 28, 5, 77, 82]：Deformable-DETR[81]利用可变形注意力模块加速收敛；DINO[73]融合对比去噪、混合查询选择与前瞻机制；RT-DETR[78]设计高效混合编码器并提出最小不确定性查询选择策略。另一类端到端检测基于CNN实现：Learnable NMS[24]和关系网络[26]通过附加网络消除重复预测；OneNet[55]与DeFCN[66]提出全卷积网络的一对一匹配策略。$\mathrm{FCOS}_{\mathrm{pss}}$ [80]则引入正样本选择器优化预测样本。

3 Methodology

3 方法论

3.1 Consistent Dual Assignments for NMS-free Training

3.1 无NMS训练的双重一致性分配

During training, YOLOs [21, 65, 29, 70] usually leverage TAL [15] to allocate multiple positive samples for each instance. The adoption of one-to-many assignment yields plentiful supervisory signals, facilitating the optimization and achieving superior performance. However, it necessitates YOLOs to rely on the NMS post-processing, which causes the suboptimal inference efficiency for deployment. While previous works [55, 66, 80, 6] explore one-to-one matching to suppress the redundant predictions, they usually introduce additional inference overhead or yield suboptimal performance. In this work, we present a NMS-free training strategy for YOLOs with dual label assignments and consistent matching metric, achieving both high efficiency and competitive performance.

训练阶段，YOLO系列[21,65,29,70]通常采用TAL[15]为每个实例分配多个正样本。这种一对多分配策略能提供丰富监督信号，促进模型优化并获得优异性能，但迫使检测器依赖NMS后处理，影响部署效率。虽然前人工作[55,66,80,6]探索一对一匹配来抑制冗余预测，但往往引入额外计算开销或导致性能下降。本研究提出具有双重标签分配和一致性匹配度量的无NMS训练策略，兼顾高效性与竞争力。

Dual label assignments. Unlike one-to-many assignment, one-to-one matching assigns only one prediction to each ground truth, avoiding the NMS post-processing. However, it leads to weak supervision, which causes suboptimal accuracy and convergence speed [82]. Fortunately, this deficiency can be compensated for by the one-to-many assignment [6]. To achieve this, we introduce

双重标签分配。与一对多分配不同，一对一匹配每个真值仅对应单个预测，从而避免NMS后处理。但这会导致监督信号薄弱，影响精度与收敛速度[82]。幸运的是，该缺陷可通过一对多分配[6]弥补。为此我们引入

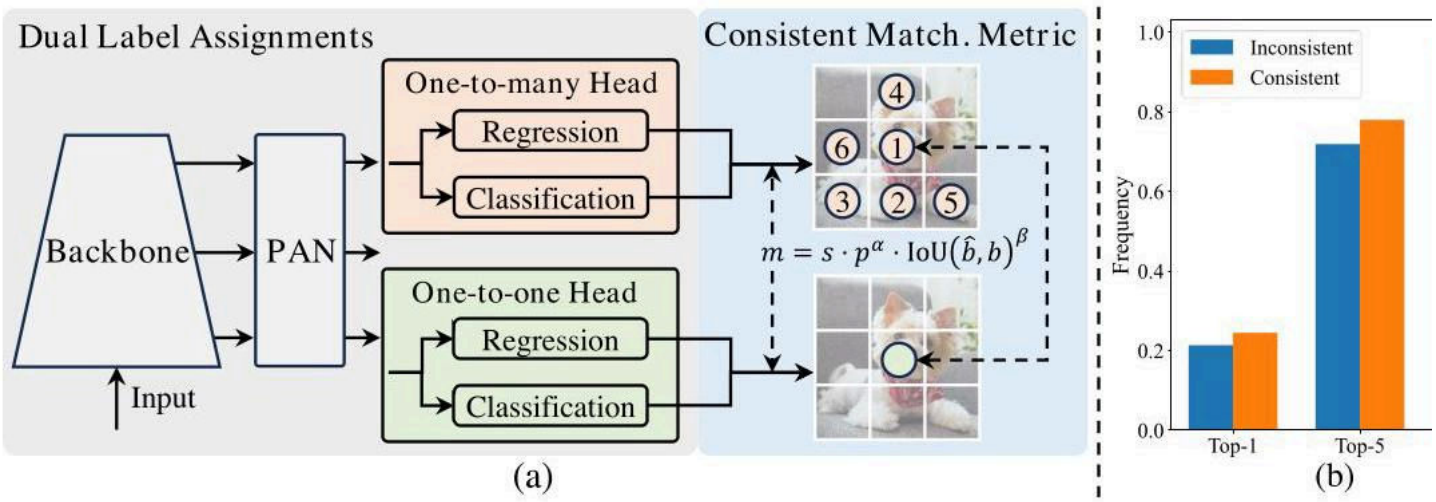

Figure 2: (a) Consistent dual assignments for NMS-free training. (b) Frequency of one-to-one assignments in Top-1/5/10 of one-to-many results for YOLOv8-S which employs $\alpha_{o2m} = 0.5$ and $\beta_{o2m} = 6$ by default [21]. For consistency, $\alpha_{o2o} = 0.5; \beta_{o2o} = 6$. For inconsistency, $\alpha_{o2o} = 0.5; \beta_{o2o} = 2$.

图2：(a)无NMS训练的双重一致性分配。(b)YOLOv8-S模型在默认使用$\alpha_{o2m} = 0.5$和$\beta_{o2m} = 6$时[21]，一对多结果中Top-1/5/10的一对一分配频率。一致性情形用$\alpha_{o2o} = 0.5; \beta_{o2o} = 6$表示，非一致性情形用$\alpha_{o2o} = 0.5; \beta_{o2o} = 2$表示。

dual label assignments for YOLOs to combine the best of both strategies. Specifically, as shown in Fig. 2(a), we incorporate another one-to-one head for YOLOs. It retains the identical structure and adopts the same optimization objectives as the original one-to-many branch but leverages the one-to-one matching to obtain label assignments. During training, two heads are jointly optimized with the model, allowing the backbone and neck to enjoy the rich supervision provided by the one-to-many assignment. During inference, we discard the one-to-many head and utilize the one-to-one head to make predictions. This enables YOLOs for the end-to-end deployment without incurring any additional inference cost. Besides, in the one-to-one matching, we adopt the top one selection, which achieves the same performance as Hungarian matching [4] with less extra training time.

为YOLO设计双标签分配策略以融合两种方案优势。具体如图2(a)所示，我们为YOLO引入另一个一对一检测头。该头部保持与原始一对多分支完全相同的结构和优化目标，但采用一对一匹配机制获取标签分配。训练时，两个检测头与模型联合优化，使主干网络和颈部能受益于一对多分配提供的丰富监督信号。推理时则丢弃一对多头部，仅使用一对一头部进行预测，从而实现无需额外推理成本的端到端部署。此外，在一对一匹配中采用Top1选择策略，该方式在保持与匈牙利匹配[4]相同性能的同时减少了额外训练时间。

Consistent matching metric. During assignments, both one-to-one and one-to-many approaches leverage a metric to quantitatively assess the level of concordance between predictions and instances. To achieve prediction aware matching for both branches, we employ a uniform matching metric, i.e.,

一致性匹配度量。在分配过程中，一对一和一对多方法均采用量化评估预测与实例匹配程度的指标。为实现两个分支的预测感知匹配，我们使用统一的匹配度量标准：

$$(1) m_{(\alpha,\beta)} = s \cdot p^{\alpha} \cdot \mathrm{IoU}\left(\hat{b}, b\right)^{\beta},$$

where p is the classification score, $\hat{b}$ and b denote the bounding box of prediction and instance, respectively. s represents the spatial prior indicating whether the anchor point of prediction is within the instance [21,65,29,70]. α and β are two important hyperparameters that balance the impact of the semantic prediction task and the location regression task. We denote the one-to-many and one-to-one metrics as $m_{o2m} = m\left(\alpha_{o2m}, \beta_{o2m}\right)$ and $m_{o2o} = m\left(\alpha_{o2o}, \beta_{o2o}\right)$,respectively. These metrics influence the label assignments and supervision information for the two heads.

其中p表示分类得分，$\hat{b}$和b分别代表预测框和实例框。s是空间先验指标，用于判断预测锚点是否位于实例内部[21,65,29,70]。α与β是两个重要超参数，用于平衡语义预测任务和位置回归任务的影响。我们将一对多和一对一度量分别记为$m_{o2m} = m\left(\alpha_{o2m}, \beta_{o2m}\right)$和$m_{o2o} = m\left(\alpha_{o2o}, \beta_{o2o}\right)$，这些度量直接影响两个头部的标签分配和监督信息。

In dual label assignments, the one-to-many branch provides much richer supervisory signals than one-to-one branch. Intuitively, if we can harmonize the supervision of the one-to-one head with that of one-to-many head, we can optimize the one-to-one head towards the direction of one-to-many head's optimization. As a result, the one-to-one head can provide improved quality of samples during inference, leading to better performance. To this end, we first analyze the supervision gap between the two heads. Due to the randomness during training, we initiate our examination in the beginning with two heads initialized with the same values and producing the same predictions, i.e., one-to-one head and one-to-many head generate the same p and IoU for each prediction-instance pair. We note that the regression targets of two branches do not conflict, as matched predictions share the same targets and unmatched predictions are ignored. The supervision gap thus lies in the different classification targets. Given an instance,we denote its largest IoU with predictions as u^* ,and the largest one-to-many and one-to-one matching scores as m^*_{o2m} and m^*_{o2o} ,respectively. Suppose that one-to-many branch yields the positive samples Ω and one-to-one branch selects i -th prediction with the metric $m_{o2o,i} = m^*_{o2o}$,we can then derive the classification target $t_{o2m,j} = u^* \cdot \frac{m_{o2m,j}}{m^*_{o2m}} \leq u^*$ for $j \in \Omega$ and $t_{o2o,i} = u^* \cdot \frac{m_{o2o,i}}{m^*_{o2o}} = u^*$ for task aligned loss as in [21, 65, 29, 70, 15]. The supervision gap between two branches can thus be derived by the 1-Wasserstein distance [46] of different classification objectives, i.e.,

双标签分配中，一对多分支提供的监督信号远丰富于一对一分支。直观上，若能协调一对一头部与一对多头部的监督信号，就能将一对一头部朝一对多头部的优化方向调整。这将提升推理阶段一对一头部提供的样本质量，从而获得更好性能。为此，我们首先分析两个头部间的监督差距。由于训练随机性，我们假设两个头部初始值相同且产生相同预测，即对每个预测-实例对生成相同的p和IoU值。注意到两个分支的回归目标不存在冲突——匹配预测共享相同目标，未匹配预测则被忽略。因此监督差距主要体现在分类目标差异上。给定实例时，记其与预测的最大IoU为u^*，最大一对多和一对一匹配分数分别为m^*_{o2m}和m^*_{o2o}。假设一对多分支产生正样本Ω，一对一分支通过度量$m_{o2o,i} = m^*_{o2o}$选择第i个预测，则可推导出$j \in \Omega$的分类目标$t_{o2m,j} = u^* \cdot \frac{m_{o2m,j}}{m^*_{o2m}} \leq u^*$及任务对齐损失$t_{o2o,i} = u^* \cdot \frac{m_{o2o,i}}{m^*_{o2o}} = u^*$（参照[21,65,29,70,15]）。两分支间的监督差距可通过不同分类目标的1-Wasserstein距离[46]量化表示为：

$$(2)\frac{A}{t_{o2o,i}} \Gamma(i \in \Omega) t_{o2m,i} + \sum_{k \in \Omega \setminus \{i\}} t_{o2m,k},$$

We can observe that the gap decreases as $t_{o2m,i}$ increases,i.e., i ranks higher within Ω . It reaches the minimum when $t_{o2m,i} = u^*$,i.e., i is the best positive sample in Ω ,as shown in Fig. 2 (a). To achieve this,we present the consistent matching metric,i.e., $\alpha_{o2o} = r \cdot \alpha_{o2m}$ and $\beta_{o2o} = r \cdot \beta_{o2m}$,which implies $m_{o2o} = m^r_{o2m}$. Therefore,the best positive sample for one-to-many head is also the best for one-to-one head. Consequently, both heads can be optimized consistently and harmoniously. For simplicity, we take $r = 1$,by default,i.e., $\alpha_{o2o} = \alpha_{o2m}$ and $\beta_{o2o} = \beta_{o2m}$. To verify the improved supervision alignment, we count the number of one-to-one matching pairs within the top- $1 / 5 / 10$ of the one-to-many results after training. As shown in Fig. 2(b), the alignment is improved under the consistent matching metric. For a more comprehensive understanding of the mathematical proof, please refer to the appendix.

我们可以观察到，随着$t_{o2m,i}$增大，间隙逐渐缩小，即i在Ω中的排序上升。当$t_{o2m,i} = u^*$时达到最小值，如图2(a)所示，此时i成为Ω中的最佳正样本。为此，我们提出一致性匹配度量（即$\alpha_{o2o} = r \cdot \alpha_{o2m}$和$\beta_{o2o} = r \cdot \beta_{o2m}$），这意味着$m_{o2o} = m^r_{o2m}$。因此，一对多头部的最佳正样本同样适用于一对一头部。这使得两个头部能够协调一致地进行优化。默认情况下我们采用$r = 1$（即$\alpha_{o2o} = \alpha_{o2m}$和$\beta_{o2o} = \beta_{o2m}$）以简化流程。为验证监督对齐的改进效果，我们统计了训练后一对多结果中top-$1 / 5 / 10$范围内一对一匹配对的数量。如图2(b)所示，在一致性匹配度量下对齐效果得到提升。更完整的数学证明详见附录。

Discussion with other counter-parts. Similarly, previous works [28, 5, 77, 54, 6, 82, 45] explore the different assignments to accelerate the training convergence and improve the performance for different networks. For example, H-DETR [28], Group-DETR [5], and MS-DETR [77] introduce one-to-many matching in conjunction with the original one-to-one matching by hybrid or multiple group label assignments, to improve upon DETR. Differently, to achieve the one-to-many matching, they usually introduce extra queries or repeat ground truths for bipartite matching, or select top several queries from the matching scores, while we adopt the prediction aware assignment that incorporates the spatial prior. Besides, LRANet [54] employs the dense assignment and sparse assignment branches for training, which all belong to the one-to-many assignment, while we adopt the one-to-many and one-to-one branches. DEYO [45, 43, 44] investigates the step-by-step training with one-to-many matching in the first stage for convolutional encoder and one-to-one matching in the second stage for

transformer decoder, while we avoid the transformer decoder for end-to-end inference. Compared with works [6, 80] which incorporate dual assignments for CNN-based detectors, we further analyze the supervision gap between the two heads and present the consistent matching metric for YOLOs to reduce the theoretical supervision gap. It improves performance through better supervision alignment and eliminates the need for hyper-parameter tuning.

与其他方案的对比讨论。类似地，先前研究[28,5,77,54,6,82,45]通过不同分配策略加速训练收敛并提升各类网络性能。例如H-DETR[28]、Group-DETR[5]和MS-DETR[77]通过混合或多组标签分配，在原始一对一匹配基础上引入一对多匹配以改进DETR。不同的是，为实现一对多匹配，这些方法通常需要额外查询或重复真实标注进行二分图匹配，或根据匹配分数选取top查询，而我们采用融合空间先验的预测感知分配。此外，LRANet[54]使用稠密分配和稀疏分配分支进行训练（均属一对多分配），而我们采用一对多与一对一分支。DEYO[45,43,44]研究分阶段训练方案：第一阶段对卷积编码器使用一对多匹配，第二阶段对Transformer解码器使用一对一匹配，而我们避免在端到端推理中使用Transformer解码器。相比基于CNN检测器的双分配方案[6,80]，我们进一步分析两个头部间的监督差距，并为YOLO提出一致性匹配度量以减小理论监督差距。该方法通过优化监督对齐提升性能，同时无需超参数调优。

3.2 Holistic Efficiency-Accuracy Driven Model Design

3.2 效率-精度协同驱动的整体模型设计

In addition to the post-processing, the model architectures of YOLOs also pose great challenges to the efficiency-accuracy trade-offs [50, 8, 29]. Although previous works explore various design strategies, the comprehensive inspection for various components in YOLOs is still lacking. Consequently, the model architecture exhibits non-negligible computational redundancy and constrained capability, which impedes its potential for achieving high efficiency and performance. Here, we aim to holistically perform model designs for YOLOs from both efficiency and accuracy perspectives.

除后处理环节外，YOLO的模型架构也对效率-精度平衡提出重大挑战[50,8,29]。尽管先前研究探索了多种设计策略，但对YOLO各组件仍缺乏系统审视。这导致模型架构存在不可忽视的计算冗余和性能局限，制约其高效能潜力。本研究旨在从效率与精度双重视角对YOLO进行整体模型设计。

Efficiency driven model design. The components in YOLO consist of the stem, downsampling layers, stages with basic building blocks, and the head. The stem incurs few computational cost and we thus perform efficiency driven model design for other three parts.

效率导向的模型设计。YOLO组件包含stem层、下采样层、基础构建块构成的stage层以及检测头。由于stem层计算成本较低，我们主要针对其余三部分进行效率驱动设计。

(1) Lightweight classification head. The classification and regression heads usually share the same architecture in YOLOs. However, they exhibit notable disparities in computational overhead. For example,the FLOPs and parameter count of the classification head ($5.95\mathrm{G}\ /\ 1.51\mathrm{M}$) are $2.5\times$ and $2.4\times$ those of the regression head (2.34G/0.64M) in YOLOv8-S, respectively. However, after analyzing the impact of classification error and the regression error (seeing Tab. 6), we find that the regression head undertakes more significance for the performance of YOLOs. Consequently, we can reduce the overhead of classification head without worrying about hurting the performance greatly. Therefore, we simply adopt a lightweight architecture for the classification head, which consists of two depthwise separable convolutions [25,9] with the kernel size of 3×3 followed by a 1×1 convolution.

(1) 轻量化分类头。YOLO系列中分类头与回归头通常采用相同架构，但两者计算开销存在显著差异。例如在YOLOv8-S模型中，分类头($5.95\mathrm{G}\ /\ 1.51\mathrm{M}$)的FLOPs和参数量分别为$2.5\times$和$2.4\times$，而回归头为2.34G/0.64M。通过分析分类误差与回归误差的影响（见表6），我们发现回归头对模型性能更为关键。因此可大幅精简分类头结构而不致显著影响性能。最终采用由两个3×3核深度可分离卷积[25,9]和1×1卷积组成的轻量架构。

(2) Spatial-channel decoupled downsampling. YOLOs typically leverage regular 3×3 standard convolutions with stride of 2,achieving spatial downsampling (from $H\times W$ to $\frac{H}{2}\times\frac{W}{2}$) and channel transformation (from C to $2C$) simultaneously. This introduces non-negligible computational cost of $O\left(\frac{9}{2}HWC^2\right)$ and parameter count of $O\left(18C^2\right)$. Instead,we propose to decouple the spatial reduction and channel increase operations, enabling more efficient downsampling. Specifically, we firstly leverage the pointwise convolution to modulate the channel dimension and then utilize the depthwise convolution to perform spatial downsampling. This reduces the computational cost to $O\left(2HWC^2+\ \frac{9}{2}HWC\right)$ and the parameter count to $O\left(2C^2+18C\right)$. Meanwhile,it maximizes information retention during downsampling, leading to competitive performance with latency reduction.

(2) 空间-通道解耦下采样。传统YOLO使用步长为2的3×3标准卷积同时实现空间降维（$H\times W\rightarrow\frac{H}{2}\times\frac{W}{2}$）和通道转换（$C\rightarrow2C$），这会带来$O\left(\frac{9}{2}HWC^2\right)$计算量和$O\left(18C^2\right)$参数。我们提出将空间压缩与通道扩展操作解耦：先通过逐点卷积调整通道维度，再用深度卷积执行空间下采样。该方法将计算量降至$O\left(2HWC^2+\ \frac{9}{2}HWC\right)$，参数量减至$O\left(2C^2+18C\right)$，在保留更多信息的同时实现更高效的下采样。

(3) Rank-guided block design. YOLOs usually employ the same basic building block for all stages [29, 65, e.g., the bottleneck block in YOLOv8 [21]. To thoroughly examine such homogeneous design for YOLOs,we utilize the intrinsic rank [33,16] to analyze the redundancy [2] of each stage. Specifically, we calculate the numerical rank of the last convolution in the last basic block in each stage, which counts the number of singular values larger than a threshold. Fig. 3 (a) presents the results of

(3) 秩引导块设计。现有YOLO模型各阶段通常采用相同基础模块[29,65]，如YOLOv8[21]的瓶颈块。为评估这种同构设计的合理性，我们利用内在秩[33,16]分析各阶段冗余度[2]。具体通过计算每阶段末位基础块中最后卷积层的数值秩（奇异值超过阈值的数量），结果如

[2] A lower rank implies greater redundancy,while a higher rank signifies more condensed information.

[2]秩值越低表明冗余度越高，反之则信息密度越大。

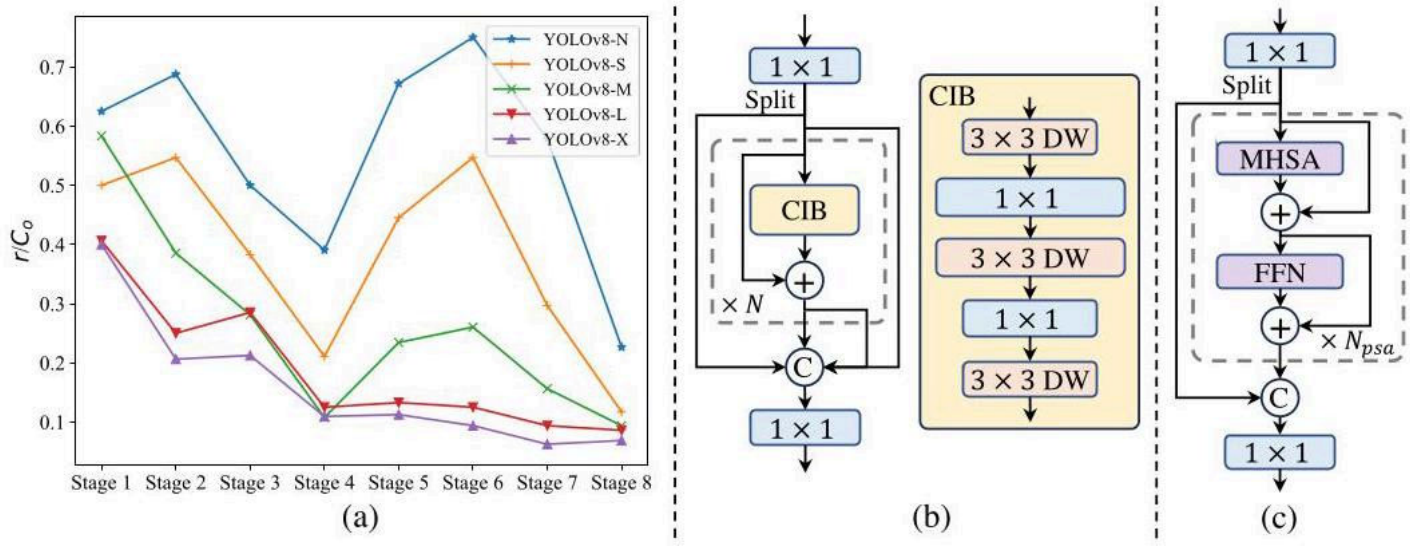

Figure 3: (a) The intrinsic ranks across stages and models in YOLOv8. The stage in the backbone and neck is numbered in the order of model forward process. The numerical rank r is normalized to r / C_o for y-axis and its threshold is set to $\lambda_{\max}$ / 2 ,by default,where C_o denotes the number of output channels and $\lambda_{\max}$ is the largest singular value. It can be observed that deep stages and large models exhibit lower intrinsic rank values. (b) The compact inverted block (CIB). (c) The partial self-attention module (PSA).

图3：(a)YOLOv8各阶段/模型的内在秩分布。主干网与颈部分阶段按前向顺序编号。数值秩r经r / C_o归一化（纵轴），默认阈值设为$\lambda_{\max}$ / 2（其中C_o为输出通道数，$\lambda_{\max}$为最大奇异值）。可见深层阶段与大模型呈现更低内在秩。(b)紧凑倒置块(CIB)结构。(c)局部自注意力模块(PSA)。

YOLOv8, indicating that deep stages and large models are prone to exhibit more redundancy. This observation suggests that simply applying the same block design for all stages is suboptimal for the best capacity-efficiency trade-off. To tackle this, we propose a rank-guided block design scheme which aims to decrease the complexity of stages that are shown to be redundant using compact architecture design. We first present a compact inverted block (CIB) structure, which adopts the cheap depthwise convolutions for spatial mixing and cost-effective pointwise convolutions for channel mixing, as shown in Fig. 3(b). It can serve as the efficient basic building block, e.g., embedded in the ELAN structure [64, 21] (Fig. 3 (b)). Then, we advocate a rank-guided block allocation strategy to achieve the best efficiency while maintaining competitive capacity. Specifically, given a model, we sort its all stages based on their intrinsic ranks in ascending order. We further inspect the performance variation of replacing the basic block in the leading stage with CIB. If there is no performance degradation compared with the given model, we proceed with the replacement of the next stage and halt the process otherwise. Consequently, we can implement adaptive compact block designs across stages and model scales, achieving higher efficiency without compromising performance. Due to the page limit, we provide the details of the algorithm in the appendix.

YOLOv8（You Only Look Once version 8）的研究表明，深层阶段和大模型易呈现更多冗余。这一发现提示，简单地为所有阶段采用相同的模块设计难以实现最佳容量-效率平衡。为此，我们提出基于秩指导的模块设计方案，通过紧凑架构设计降低已识别冗余阶段的复杂度。首先，我们设计了一种紧凑倒置模块（CIB）结构（图3b），其采用低成本深度卷积进行空间混合，高效点卷积进行通道混合，可作为基础构建单元（例如嵌入ELAN结构[64,21]）。继而提出秩指导的模块分配策略：对模型各阶段按本征秩升序排列，逐步将领先阶段的基础模块替换为CIB，若性能无衰减则继续替换下一阶段，否则终止流程。由此实现跨阶段、跨模型规模的自适应紧凑设计，在保持性能的同时提升效率。因篇幅限制，算法细节详见附录。

Accuracy driven model design. We further explore the large-kernel convolution and self-attention for accuracy driven design, aiming to boost the performance under minimal cost.

精度导向模型设计。我们进一步探索大核卷积与自注意力机制，旨在最小成本下提升性能。

(1) Large-kernel convolution. Employing large-kernel depthwise convolution is an effective way to enlarge the receptive field and enhance the model's capability [10, 40, 39]. However, simply leveraging them in all stages may introduce contamination in shallow features used for detecting small objects, while also introducing significant I/O overhead and latency in high-resolution stages [8]. Therefore, we propose to leverage the large-kernel depthwise convolutions in CIB within the deep stages. Specifically,we increase the kernel size of the second 3×3 depthwise convolution in the CIB to 7×7 ,following [39]. Additionally,we employ the structural reparameterization technique [11,10,59] to bring another 3×3 depthwise convolution branch to alleviate the optimization issue without inference overhead. Furthermore, as the model size increases, its receptive field naturally expands, with the benefit of using large-kernel convolutions diminishing. Therefore, we only adopt large-kernel convolution for small model scales.

（1）大核卷积。采用大核深度卷积是扩大感受野、增强模型能力的有效手段[10,40,39]。但全阶段使用会污染小目标检测所需的浅层特征，并导致高分辨率阶段产生显著I/O开销与延迟[8]。因此，我们仅在深层阶段的CIB中应用大核深度卷积：将第二个3×3深度卷积核尺寸增至7×7[39]，并引入结构重参数化技术[11,10,59]添加3×3深度卷积分支以缓解优化问题（不增加推理开销）。随着模型规模扩大，其感受野自然扩展，大核卷积收益递减，故仅在小规模模型中使用。

(2) Partial self-attention (PSA). Self-attention [58] is widely employed in various visual tasks due to its remarkable global modeling capability [38, 14, 76]. However, it exhibits high computational complexity and memory footprint. To address this, in light of the prevalent attention head redundancy [69], we present an efficient partial self-attention (PSA) module design, as shown in Fig. 3 (c). Specifically,we evenly partition the features across channels into two parts after the 1×1 convolution. We only feed one part into the N_{PSA} blocks comprised of multi-head self-attention module (MHSA) and feed-forward network (FFN). Two parts are then concatenated and fused by a 1×1 convolution. Besides, we follow [22] to assign the dimensions of the query and key to half of that of the value in MHSA and replace the LayerNorm [1] with BatchNorm [27] for fast inference. Furthermore, PSA is only placed after the Stage 4 with the lowest resolution, avoiding the excessive overhead from the

（2）局部自注意力（PSA）。自注意力机制[58]凭借卓越的全局建模能力[38,14,76]广泛应用于视觉任务，但存在高计算复杂度与内存占用。鉴于注意力头冗余普遍存在[69]，我们设计高效局部自注意力模块（图3c）：1×1卷积后将特征通道均分为二，仅将一部分输入含多头自注意力（MHSA）和前馈网络（FFN）的N_{PSA}模块，最后拼接并通过1×1卷积融合。参照[22]，我们将MHSA中查询/键维度设为值维度的一半，并用批归一化[27]替代层归一化[1]以加速推理。PSA仅置于分辨率最低的第四阶段后，避免高分辨率带来的过大开销。

Table 1: Comparisons with state-of-the-arts. Latency is measured using official pre-trained models. Latency ^{f} denotes the latency in the forward process of model without post-processing. † means the results of YOLOv10 with the original one-to-many training using NMS. All results below are without the additional advanced training techniques like knowledge distillation or PGI for fair comparisons.

表1：与前沿方法对比。延迟测试基于官方预训练模型，^{f}表示无后处理的模型前向过程延迟，†为采用原始一对多训练及NMS的YOLOv10结果。为公平比较，以下结果均未使用知识蒸馏/PGI等进阶训练技术。

Model	#Param.(M)	FLOPs(G)	AP^{val} (%)	Latency(ms)	Latency ^{f} (ms)
YOLOv6-3.0-N [29	4.7	11.4	37.0	2.69	1.76
Gold-YOLO-N 60	5.6	12.1	39.6	2.92	1.82
YOLOv8-N [21	3.2	8.7	37.3	6.16	1.77
YOLOv10-N (Ours)	2.3	6.7	$38.5 / 39.5^{\dagger}$	1.84	1.79
YOLOv6-3.0-S [29	18.5	45.3	44.3	3.42	2.35
Gold-YOLO-S [60	21.5	46.0	45.4	3.82	2.73
YOLO-MS-XS [8]	4.5	17.4	43.4	8.23	2.80
YOLO-MS-S [8	8.1	31.2	46.2	10.12	4.83
YOLOv8-S [21	11.2	28.6	44.9	7.07	2.33
YOLOv9-S 65	7.1	26.4	46.7	-	-
RT-DETR-R18 [78]	20.0	60.0	46.5	4.58	4.49
YOLOv10-S (Ours)	7.2	21.6	46.3 / 46.8	2.49	2.39
YOLOv6-3.0-M [29	34.9	85.8	49.1	5.63	4.56
Gold-YOLO-M [60]	41.3	87.5	49.8	6.38	5.45
YOLO-MS [8]	22.2	80.2	51.0	12.41	7.30

YOLOv8-M [21	25.9	78.9	50.6	9.50	5.09
YOLOv9-M [65]	20.0	76.3	51.1	-	-
RT-DETR-R34 [78]	31.0	92.0	48.9	6.32	6.21
RT-DETR-R50m [78]	36.0	100.0	51.3	6.90	6.84
YOLOv10-M (Ours)	15.4	59.1	$51.1 / 51.3^{\dagger}$	4.74	4.63
YOLOv6-3.0-L [29	59.6	150.7	51.8	9.02	7.90
Gold-YOLO-L [60]	75.1	151.7	51.8	10.65	9.78
YOLOv9-C [65]	25.3	102.1	52.5	10.57	6.13
YOLOv10-B (Ours)	19.1	92.0	$52.5 / 52.7^{\dagger}$	5.74	5.67
YOLOv8-L [21	43.7	165.2	52.9	12.39	8.06
RT-DETR-R50 [78]	42.0	136.0	53.1	9.20	9.07
YOLOv10-L (Ours)	24.4	120.3	$53.2 / 53.4^{\dagger}$	7.28	7.21
YOLOv8-X [21]	68.2	257.8	53.9	16.86	12.83
RT-DETR-R101 [78]	76.0	259.0	54.3	13.71	13.58
YOLOv10-X (Ours)	29.5	160.4	$54.4 / 54.4^{\dagger}$	10.70	10.60

模型	参数量(百万)	计算量(十亿次)	AP^{val} (%)	延迟(毫秒)	延迟 ^{f} (毫秒)
YOLOv6-3.0-N [29	4.7	11.4	37.0	2.69	1.76
Gold-YOLO-N 60	5.6	12.1	39.6	2.92	1.82
YOLOv8-N [21	3.2	8.7	37.3	6.16	1.77
YOLOv10-N (我们的)	2.3	6.7	$38.5 / 39.5^{\dagger}$	1.84	1.79
YOLOv6-3.0-S [29	18.5	45.3	44.3	3.42	2.35
Gold-YOLO-S [60	21.5	46.0	45.4	3.82	2.73
YOLO-MS-XS [8]	4.5	17.4	43.4	8.23	2.80
YOLO-MS-S [8	8.1	31.2	46.2	10.12	4.83
YOLOv8-S [21	11.2	28.6	44.9	7.07	2.33
YOLOv9-S 65	7.1	26.4	46.7	-	-
RT-DETR-R18 [78]	20.0	60.0	46.5	4.58	4.49
YOLOv10-S (我们的)	7.2	21.6	46.3 / 46.8	2.49	2.39
YOLOv6-3.0-M [29	34.9	85.8	49.1	5.63	4.56
Gold-YOLO-M [60]	41.3	87.5	49.8	6.38	5.45
YOLO-MS [8]	22.2	80.2	51.0	12.41	7.30
YOLOv8-M [21	25.9	78.9	50.6	9.50	5.09
YOLOv9-M [65]	20.0	76.3	51.1	-	-
RT-DETR-R34 [78]	31.0	92.0	48.9	6.32	6.21
RT-DETR-R50m [78]	36.0	100.0	51.3	6.90	6.84
YOLOv10-M (我们的)	15.4	59.1	$51.1 / 51.3^{\dagger}$	4.74	4.63
YOLOv6-3.0-L [29	59.6	150.7	51.8	9.02	7.90
Gold-YOLO-L [60]	75.1	151.7	51.8	10.65	9.78
YOLOv9-C [65]	25.3	102.1	52.5	10.57	6.13
YOLOv10-B (我们的)	19.1	92.0	$52.5 / 52.7^{\dagger}$	5.74	5.67
YOLOv8-L [21	43.7	165.2	52.9	12.39	8.06
RT-DETR-R50 [78]	42.0	136.0	53.1	9.20	9.07
YOLOv10-L (我们的)	24.4	120.3	$53.2 / 53.4^{\dagger}$	7.28	7.21
YOLOv8-X [21]	68.2	257.8	53.9	16.86	12.83
RT-DETR-R101 [78]	76.0	259.0	54.3	13.71	13.58
YOLOv10-X (我们的)	29.5	160.4	$54.4 / 54.4^{\dagger}$	10.70	10.60

quadratic computational complexity of self-attention. In this way, the global representation learning ability can be incorporated into YOLOs with low computational costs, which well enhances the model's capability and leads to improved performance.

自注意力机制的二次计算复杂度。通过这种方式，全局表征学习能力能够以较低计算成本融入YOLO系列模型，显著增强模型性能并提升检测效果。

4 Experiments

4 实验

4.1 Implementation Details

4.1 实现细节

We select YOLOv8 [21] as our baseline model, due to its commendable latency-accuracy balance and its availability in various model sizes. We employ the consistent dual assignments for NMS-free training and perform holistic efficiency-accuracy driven model design based on it, which brings our YOLOv10 models. YOLOv10 has the same variants as YOLOv8, i.e., N/S/M/L/X. Besides, we derive a new variant YOLOv10-B, by simply increasing the width scale factor of YOLOv10-M. We verify the proposed detector on COCO [35] under the same training-from-scratch setting [21, 65, 62]. Moreover, the latencies of all models are tested on T4 GPU with TensorRT FP16, following [78].

我们选择YOLOv8[21]作为基线模型，因其在延迟与精度间具有出色平衡性且提供多种尺寸版本。采用无NMS训练的双标签分配策略，并基于此进行全局效率-精度驱动的模型设计，最终得到YOLOv10系列。该系列包含与YOLOv8相同的N/S/M/L/X变体，另通过增大YOLOv10-M的宽度比例因子新增YOLOv10-B变体。所有模型均在COCO数据集[35]上采用从头训练设置[21,65,62]进行验证，延迟测试则遵循[78]在T4 GPU搭配TensorRT FP16环境下完成。

4.2 Comparison with state-of-the-arts

4.2 与前沿方法对比

As shown in Tab. 1, our YOLOv10 achieves the state-of-the-art performance and end-to-end latency across various model scales. We first compare YOLOv10 with our baseline models, i.e., YOLOv8. On N / S / M / L / X five variants, our YOLOv10 achieves 1.2% / 1.4% / 0.5% / 0.3% / 0.5% AP improvements, with 28% / 36% / 41% / 44% / 57% fewer parameters, 23% / 24% / 25% / 27% / 38% less calculations, and 70% / 65% / 50% / 41% / 37% lower latencies. Compared with other YOLOs,

如表1所示，YOLOv10在不同模型规模下均实现最优性能与端到端延迟。相较于基线模型YOLOv8，在N/S/M/L/X五个变体上分别获得1.2%/1.4%/0.5%/0.3%/0.5%的AP提升，参数量减少28%/36%/41%/44%/57%，计算量降低23%/24%/25%/27%/38%，延迟下降70%/65%/50%/41%/37%。与其他YOLO模型相比，

Table 2: Ablation study with YOLOv10-S and YOLOv10-M on COCO.

表2：基于COCO数据集的YOLOv10-S与YOLOv10-M消融实验

#Model	NMS-free.	Efficiency.		#Param.(M)	FLOPs(G)	AP^{val} (%)	Latency(ms)
1 2 YOLOv10-S 3 4				11.2	28.6	44.9	7.07
	✓			11.2	28.6	44.3	2.44
	✓	✓		6.2	20.8	44.5	2.31
	✓	✓	✓	7.2	21.6	46.3	2.49
5 YOLOv10-M				25.9	78.9	50.6	9.50
	✓			25.9	78.9	50.3	5.22
	✓	✓		14.1	58.1	50.4	4.57
8	✓	✓	✓	15.4	59.1	51.1	4.74

#模型	无需非极大值抑制	效率		#参数量(百万)	浮点运算量(十亿次)	AP^{val} (%)	延迟(毫秒)
1 2 YOLOv10-S 3 4				11.2	28.6	44.9	7.07
	✓			11.2	28.6	44.3	2.44
	✓	✓		6.2	20.8	44.5	2.31
	✓	✓	✓	7.2	21.6	46.3	2.49

5 YOLOv10-M				25.9	78.9	50.6	9.50
	✓			25.9	78.9	50.3	5.22
	✓	✓		14.1	58.1	50.4	4.57
8	✓	✓	✓	15.4	59.1	51.1	4.74

Table 3: Dual assign. Table 4: Matching metric.

表3：双重分配。表4：匹配度量。

APLatency		α_{o2o}	β o2o	AP^{val}	α_{o2o}	β o2o	AP^{val}
✓44.9	7.07	0.5	2.0	42.7	0.25	3.0	44.3
✓43.4	2.44	0.5	4.0	44.2	0.25	6.0	43.5
✓✓44.3	2.44	0.5	6.0	44.3	1.0	6.0	43.9
		0.5	8.0	44.0	1.0	12.0	44.3

AP延迟		α_{o2o}	β 线上到线下	AP^{val}	α_{o2o}	β 线上到线下	AP^{val}
✓44.9	7.07	0.5	2.0	42.7	0.25	3.0	44.3
✓43.4	2.44	0.5	4.0	44.2	0.25	6.0	43.5
✓✓44.3	2.44	0.5	6.0	44.3	1.0	6.0	43.9
		0.5	8.0	44.0	1.0	12.0	44.3

Table 5: Efficiency. for YOLOv10-S/M.

表5：YOLOv10-S/M模型的效率指标

#Model	#Param	FLOPs	$\mathrm{AP}^{\mathrm{val}}$	Latency
1 base.	11.2/25.9	28.6/78.9	44.3/50.3	2.44/5.22
2 +cls.	9.9/23.2	23.5/67.7	44.2/50.2	2.39/5.07
3 +downs.	8.0/19.7	22.2/65.0	44.4/50.4	2.36/4.97
+block.	6.2/14.1	20.8/58.1	44.5/50.4	2.31/4.57

#模型	#参数	浮点运算次数	$\mathrm{AP}^{\mathrm{val}}$	延迟
1 基础	11.2/25.9	28.6/78.9	44.3/50.3	2.44/5.22
2 +分类	9.9/23.2	23.5/67.7	44.2/50.2	2.39/5.07
3 +降采样	8.0/19.7	22.2/65.0	44.4/50.4	2.36/4.97
+块	6.2/14.1	20.8/58.1	44.5/50.4	2.31/4.57

YOLOv10 also exhibits superior trade-offs between accuracy and computational cost. Specifically, for lightweight and small models, YOLOv10-N / S outperforms YOLOv6-3.0-N / S by 1.5 AP and 2.0 AP,with 51% / 61% fewer parameters and 41% / 52% less computations,respectively. For medium models, compared with YOLOv9-C / YOLO-MS, YOLOv10-B / M enjoys the 46% / 62% latency reduction under the same or better performance, respectively. For large models, compared with Gold-YOLO-L, our YOLOv10-L shows 68% fewer parameters and 32% lower latency, along with a significant improvement of 1.4% AP. Furthermore, compared with RT-DETR, YOLOv10 obtains significant performance and latency improvements. Notably,YOLOv10-S / X achieves $1.8\times$ and $1.3\times$ faster inference speed than RT-DETR-R18 / R101, respectively, under the similar performance. These results well demonstrate the superiority of YOLOv10 as the real-time end-to-end detector.

YOLOv10在精度与计算成本间展现出更优的平衡。具体而言，轻量级和小型模型中，YOLOv10-N/S分别以51% / 61%更少参数和41% / 52%更低计算量，超越YOLOv6-3.0-N/S达1.5 AP和2.0 AP。中型模型方面，在同等或更优性能下，YOLOv10-B/M相较YOLOv9-C/YOLO-MS分别实现46%/62%的延迟降低。大型模型中，YOLOv10-L相比Gold-YOLO-L参数减少68%，延迟降低32%，同时AP显著提升1.4%。与RT-DETR相比，YOLOv10在性能和延迟上均有显著提升。值得注意的是，在相近性能下，YOLOv10-S/X的推理速度分别比RT-DETR-R18/R101快$1.8\times$和$1.3\times$。这些结果充分证明了YOLOv10作为实时端到端检测器的优越性。

We also compare YOLOv10 with other YOLOs using the original one-to-many training approach. We consider the performance and the latency of model forward process (Latency ^{f}) in this situation, following [62, 21, 60]. As shown in Tab. 1, YOLOv10 also exhibits the state-of-the-art performance and efficiency across different model scales, indicating the effectiveness of our architectural designs.

我们同样采用原始一对多训练方法将YOLOv10与其他YOLO模型对比。参照[62,21,60]的研究范式，我们评估了模型前向过程的性能与延迟(Latency^f)。如表1所示，YOLOv10在不同模型规模下均展现出最先进的性能与效率，印证了我们架构设计的有效性。

4.3 Model Analyses

4.3 模型分析

Ablation study. We present the ablation results based on YOLOv10-S and YOLOv10-M in Tab. 2 It can be observed that our NMS-free training with consistent dual assignments significantly reduces the end-to-end latency of YOLOv10-S by 4.63ms, while maintaining competitive performance of 44.3% AP. Moreover,our efficiency driven model design leads to the reduction of 11.8 M parameters and 20.8 GFlOPs,with a considerable latency reduction of 0.65 ms for YOLOv10-M,well showing its effectiveness. Furthermore, our accuracy driven model design achieves the notable improvements of 1.8 AP and 0.7 AP for YOLOv10-S and YOLOv10-M,alone with only 0.18ms and 0.17ms latency overhead, respectively, which well demonstrates its superiority.

消融实验。基于YOLOv10-S和YOLOv10-M的消融结果展示于表2。可见，我们采用一致性双分配的无NMS训练使YOLOv10-S端到端延迟显著降低4.63ms，同时保持44.3% AP的竞争力。效率驱动的模型设计使YOLOv10-M参数量减少11.8 M，计算量降低20.8 GFLOPS，延迟显著下降0.65 ms，充分验证其有效性。精度驱动的设计则使YOLOv10-S/M分别提升1.8 AP和0.7 AP，仅增加0.18ms/0.17ms延迟，彰显其优越性。

Analyses for NMS-free training.

无NMS训练分析

- Dual label assignments. We present dual label assignments for NMS-free YOLOs, which can bring both rich supervision of one-to-many (o2m) branch during training and high efficiency of one-to-one (o2o) branch during inference. We verify its benefit based on YOLOv8-S, i.e., #1 in Tab. 2, Specifically, we introduce baselines for training with only o2m branch and only o2o branch, respectively. As shown in Tab. 3, our dual label assignments achieve the best AP-latency trade-off.

- 双标签分配。我们为无NMS的YOLO模型设计双标签分配策略，既能保留训练阶段一对多(o2m)分支的丰富监督，又具备推理阶段一对一(o2o)分支的高效特性。基于YOLOv8-S(即表2#1)的验证实验显示：如表3所示，单独使用o2m或o2o分支的基线模型中，我们的双标签分配实现了最佳AP-延迟平衡。

- Consistent matching metric. We introduce consistent matching metric to make the one-to-one head more harmonious with the one-to-many head. We verify its benefit based on YOLOv8-S, i.e., #1 in Tab. 2,under different α_{o2o} and β_{o2o} . As shown in Tab. 4,the proposed consistent matching metric, i.e., $\alpha_{o2o} = r \cdot \alpha_{o2m}$ and $\beta_{o2o} = r \cdot \beta_{o2m}$,can achieve the optimal performance,where $\alpha_{o2m} = 0.5$ and $\beta_{o2m} = 6.0$ in the one-to-many head [21]. Such an improvement can be attributed to the reduction of the supervision gap (Eq. (2)), which provides improved supervision alignment between two branches. Moreover, the proposed consistent matching metric eliminates the need for exhaustive hyper-parameter tuning, which is appealing in practical scenarios.

- 一致性匹配度量。我们引入一致性匹配度量以使一对一头部与一对多头部更协调。基于YOLOv8-S(表2#1)在不同α_{o2o}和β_{o2o}条件下的实验表明：如表4所示，提出的一致性匹配度量(即$\alpha_{o2o} = r \cdot \alpha_{o2m}$和$\beta_{o2o} = r \cdot \beta_{o2m}$)能实现最优性能，其中一对多头部采用$\alpha_{o2m} = 0.5$和$\beta_{o2m} = 6.0$[21]。这种改进源于监督差距的减小(公式(2))，促进了两分支间的监督对齐。该度量还免除了繁琐的超参数调优，在实际应用中极具吸引力。

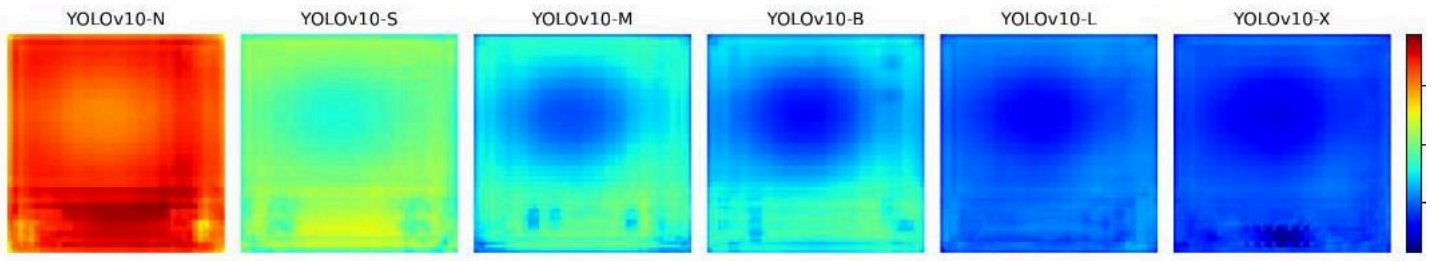

Figure 4: The average cosine similarity of each anchor point's extracted features with all others.

图4：各锚点提取特征与所有其他锚点的平均余弦相似度

Table 6: cls. results. Table 7: Results of d.s.

表6：分类结果 表7：检测结果

Table 8: Results of CIB.

表8：CIB测试结果

Model	AP^{val}	Latency
IRB	43.7	2.30
IRB-DW	44.2	2.30
ours	44.5	2.31

模型	AP^{val}	延迟
机构审查委员会(IRB)	43.7	2.30
机构审查委员会-数据仓库(IRB-DW)	44.2	2.30
我们的方案	44.5	2.31

Table 9: Rank-guided.

表9：排名引导式(Rank-guided)

Stages with CIB APAP^{val}
empty44.4
844.5
8,4,44.5
8,4,744.3

采用CIB APAP^{val} 的施工阶段
空置44.4
844.5
8,4,44.5
8,4,744.3

	base.	+cls.
AP^{val}	44.3	44.2
$AP^{val}_{w/oc}$	59.9	59.9
$AP^{val}_{w/or}$	64.5	64.2

	基础。	+分类。
AP^{val}	44.3	44.2
$AP^{val}_{w/oc}$	59.9	59.9

$AP^{val}_{w/or}$	64.5	64.2

Model	AP^{val}	Latency
base.	43.7	2.33
ours	44.4	2.36

模型	AP^{val}	延迟
基础	43.7	2.33
我们的	44.4	2.36

- Performance gap compared with one-to-many training. Although achieving superior end-to-end performance under NMS-free training, we observe that there still exists the performance gap compared with the original one-to-many training using NMS, as shown in Tab. 3 and Tab. 1 Besides, we note that the gap diminishes as the model size increases. Therefore, we reasonably concludes that such a gap can be attributed to the limitations in the model capability. Notably, unlike the original one-to-many training using NMS, the NMS-free training necessitates more discriminative features for one-to-one matching. In the case of the YOLOv10-N model, its limited capacity results in extracted features that lack sufficient discriminability, leading to a more notable performance gap of 1.0% AP. In contrast,the YOLOv10-X model,which possesses stronger capability and more discriminative features, shows no performance gap between two training strategies. In Fig. 4, we visualize the average cosine similarity of each anchor point's extracted features with those of all other anchor points on the COCO val set. We observe that as the model size increases, the feature similarity between anchor points exhibits a downward trend, which benefits the one-to-one matching. Based on this insight, we will explore approaches to further reduce the gap and achieve higher end-to-end performance in the future work.

- 与一对多训练的性能差距。尽管在无需NMS的训练下取得了优异的端到端性能，但我们观察到与原始使用NMS的一对多训练相比仍存在性能差距（如表3和表1所示）。此外，我们注意到该差距随着模型规模增大而缩小。因此，我们合理推断这种差距可归因于模型能力的局限性。值得注意的是，与原始使用NMS的一对多训练不同，无需NMS的训练需要更具判别性的特征来实现一对一匹配。以YOLOv10-N模型为例，其有限容量导致提取的特征缺乏足够区分度，从而产生1.0% AP的显著性能差距。相比之下，具备更强能力和更判别性特征的YOLOv10-X模型在两种训练策略间未显示性能差距。图4中，我们可视化了COCO验证集上各锚点提取特征与其他所有锚点特征的平均余弦相似度。观察到随着模型规模增大，锚点间特征相似度呈下降趋势，这有利于一对一匹配。基于此发现，我们将在未来工作中探索进一步缩小差距、提升端到端性能的方法。

Analyses for efficiency driven model design. We conduct experiments to gradually incorporate the efficiency driven design elements based on YOLOv10-S/M. Our baseline is the YOLOv10-S/M model without efficiency-accuracy driven model design, i.e., #2/#6 in Tab. 2. As shown in Tab. 5, each design component, including lightweight classification head, spatial-channel decoupled downsampling, and rank-guided block design, contributes to the reduction of parameters count, FLOPs, and latency. Importantly, these improvements are achieved while maintaining competitive performance.

效率驱动模型设计的分析。我们基于YOLOv10-S/M逐步引入效率驱动设计元素进行实验。基线是未采用效率-精度驱动设计的YOLOv10-S/M模型（即表2中的#2/#6）。如表5所示，每个设计组件（包括轻量级分类头、空间-通道解耦下采样和秩导向块设计）都有助于减少参数量、FLOPs和延迟。重要的是，这些改进是在保持竞争力的性能前提下实现的。

- Lightweight classification head. We analyze the impact of category and localization errors of predictions on the performance, based on the YOLOv10-S of #1 and #2 in Tab. 5, like [7]. Specifically, we match the predictions to the instances by the one-to-one assignment. Then, we substitute the predicted category score with instance labels, resulting in $AP^{val}_{w/oc}$ with no classification errors. Similarly, we replace the predicted locations with those of instances, yielding $AP^{val}_{w/or}$ with no regression errors. As shown in Tab. 6, $AP^{val}_{w/or}$ is much higher than $AP^{val}_{w/oc}$, revealing that eliminating the regression errors achieves greater improvement. The performance bottleneck thus lies more in the regression task. Therefore, adopting the lightweight classification head can allow higher efficiency without compromising the performance.

- 轻量级分类头。我们基于表5中#1和#2的YOLOv10-S模型，参照文献[7]分析预测的类别误差与定位误差对性能的影响。具体而言，通过一对一分配将预测与实例匹配后，用实例标签替换预测类别分数得到$AP^{val}_{w/oc}$（无分类误差）；用实例位置替换预测位置得到$AP^{val}_{w/or}$（无回归误差）。如表6所示，$AP^{val}_{w/or}$显著高于$AP^{val}_{w/oc}$，表明消除回归误差能带来更大提升。因此性能瓶颈更多在于回归任务，采用轻量级分类头可在不影响性能的前提下提高效率。

- Spatial-channel decoupled downsampling. We decouple the downsampling operations for efficiency, where the channel dimensions are first increased by pointwise convolution (PW) and the resolution is then reduced by depthwise convolution (DW) for maximal information

retention. We compare it with the baseline way of spatial reduction by DW followed by channel modulation by PW, based on the YOLOv10-S of #3 in Tab. 5. As shown in Tab. 7, our downsampling strategy achieves the 0.7% AP improvement by enjoying less information loss during downsampling.

- 空间-通道解耦下采样。为提升效率，我们将下采样操作解耦：先通过逐点卷积（PW）增加通道维度，再通过深度卷积（DW）降低分辨率以最大化信息保留。基于表5中#3的YOLOv10-S模型，与DW空间缩减后PW通道调制的基线方法对比。如表7所示，我们的下采样策略因减少信息损失而实现0.7% AP提升。

- Compact inverted block (CIB). We introduce CIB as the compact basic building block. We verify its effectiveness based on the YOLOv10-S of #4 in the Tab. 5, Specifically, we introduce the inverted residual block [51] (IRB) as the baseline,which achieves the suboptimal 43.7% AP, as shown in Tab. 8. We then append a 3×3 depthwise convolution (DW) after it,denoted as `"IRB-DW"`,which

- 紧凑倒置块（CIB）。我们引入CIB作为紧凑基础构建模块。基于表5中#4的YOLOv10-S模型验证其有效性：以倒置残差块[51]（IRB）为基线时获得次优的43.7% AP（表8所示）；在其后追加3×3深度卷积（DW）形成`"IRB-DW"`结构后

Table 10: Accuracy. for S/M. Table 11: L.k. results. Table 12: L.k. usage. Table 13: PSA results. brings 0.5% AP improvement. Compared with "IRB-DW", our CIB further achieves 0.3% AP improvement by prepending another DW with minimal overhead, indicating its superiority.

表10：S/M精度。表11：L.k.结果。表12：L.k.使用情况。表13：PSA结果。带来0.5% AP提升。与"IRB-DW"相比，我们的CIB通过前置另一个低开销DW卷积进一步实现0.3% AP提升，证明了其优越性。

#Model	AP^{val}	Latency
1base.	44.5/50.4	2.31/4.57
2+L.k.	44.9/-	2.34/-
3+PSA	46.3/51.1	2.49/4.74

#模型	AP^{val}	延迟
1基准	44.5/50.4	2.31/4.57
2+链路损耗	44.9/-	2.34/-
3+相控阵	46.3/51.1	2.49/4.74

Model		Latency
k.s.=5	44.7	2.32
k.s.=7	44.9	2.34
k.s.=9	44.9	2.37
w/o rep. 44.8		2.34

模型		延迟
k.s.=5	44.7	2.32
k.s.=7	44.9	2.34
k.s.=9	44.9	2.37
无重复 44.8		2.34

	w/o L.k.	w/L.k.
N	36.3	36.6
S	44.5	44.9
M	50.4	50.4

	无L.k.	含L.k.
N	36.3	36.6
S	44.5	44.9

M	50.4	50.4

Model	AP^{val}	Latency
PSA	46.3	2.49
Trans.	46.0	2.54
$N_{PSA} = 1$	46.3	2.49
$N_{PSA} = 2$	46.5	2.59

模型	AP^{val}	延迟
PSA	46.3	2.49
翻译	46.0	2.54
$N_{PSA} = 1$	46.3	2.49
$N_{PSA} = 2$	46.5	2.59

- Rank-guided block design. We introduce the rank-guided block design to adaptively integrate compact block design for improving the model efficiency. We verify its benefit based on the YOLOv10-S of #3 in the Tab. 5. The stages sorted in ascending order based on the intrinsic ranks are Stage 8-4-7-3-5-1-6-2, like in Fig. 3 (a). As shown in Tab. 9, when gradually replacing the bottleneck block in each stage with the efficient CIB, we observe the performance degradation starting from Stage 7. In the Stage 8 and 4 with lower intrinsic ranks and more redundancy, we can thus adopt the efficient block design without compromising the performance. These results indicate that rank-guided block design can serve as an effective strategy for higher model efficiency.

- 秩引导块设计。我们提出秩引导块设计方法，通过自适应整合紧凑块结构来提升模型效率。基于表5中#3的YOLOv10-S验证其优势。如图3(a)所示，按本征秩升序排列的层级顺序为Stage 8-4-7-3-5-1-6-2。表9显示，当逐步用高效CIB模块替换各阶段瓶颈块时，从Stage 7开始出现性能下降。在本征秩较低、冗余度较高的Stage 8和4中，可采用高效块设计而不影响性能。这些结果表明秩引导块设计是提升模型效率的有效策略。

Analyses for accuracy driven model design. We present the results of gradually integrating the accuracy driven design elements based on YOLOv10-S/M. Our baseline is the YOLOv10-S/M model after incorporating efficiency driven design, i.e., #3/#7 in Tab. 2 As shown in Tab. 10, the adoption of large-kernel convolution and PSA module leads to the considerable performance improvements of 0.4% AP and 1.4% AP for YOLOv10-S under minimal latency increase of 0.03ms and 0.15ms, respectively. Note that large-kernel convolution is not employed for YOLOv10-M (see Tab. 12).

精度驱动模型设计分析。基于YOLOv10-S/M逐步整合精度驱动设计元素的结果显示：以表2中#3/#7（整合效率驱动设计后的模型）为基线时，如表10所示，采用大核卷积和PSA模块使YOLOv10-S在仅增加0.03ms/0.15ms延迟的情况下，AP分别提升0.4%和1.4%。需注意YOLOv10-M未采用大核卷积（见表12）。

- Large-kernel convolution. We first investigate the effect of different kernel sizes based on the YOLOv10-S of #2 in Tab. 10 As shown in Tab. 11, the performance improves as the kernel size increases and stagnates around the kernel size of 7×7 ,indicating the benefit of large perception field. Besides, removing the reparameterization branch during training achieves 0.1% AP degradation, showing its effectiveness for optimization. Moreover, we inspect the benefit of large-kernel convolution across model scales based on YOLOv10-N / S / M. As shown in Tab. 12, it brings no improvements for large models, i.e., YOLOv10-M, due to its inherent extensive receptive field. We thus only adopt large-kernel convolutions for small models,i.e.,YOLOv10-N / S.

- 大核卷积。基于表10中#2的YOLOv10-S研究不同核尺寸效果：表11显示性能随核尺寸增大而提升，在7×7附近趋于稳定，证实大感受野的优势。移除训练时的重参数化分支会导致AP下降0.1%，说明其优化有效性。进一步在不同规模模型（YOLOv10-N/S/M）上验证发现（表12），大核卷积对本身具有广阔感受野的大模型（如YOLOv10-M）无增益，故仅在小模型（YOLOv10-N/S）中采用。

- Partial self-attention (PSA). We introduce PSA to enhance the performance by incorporating the global modeling ability under minimal cost. We first verify its effectiveness based on the YOLOv10- S of #3 in Tab. 10, Specifically, we introduce the transformer block, i.e., MHSA followed by FFN, as the baseline, denoted as `"Trans."`. As shown in Tab. 13, compared with it, PSA brings 0.3% AP improvement with 0.05 ms latency reduction. The performance enhancement may be attributed to the alleviation of optimization problem [68, 10] in self-attention, by mitigating the redundancy in attention heads. Moreover,we investigate the impact of different N_{PSA} . As shown in Tab. 13, increasing N_{PSA} to 2 obtains 0.2% AP improvement but with 0.1 ms latency overhead. Therefore, we set N_{PSA} to 1,by default,to enhance the model capability while maintaining high efficiency.

- 局部自注意力（PSA）。我们提出PSA模块以最小代价引入全局建模能力。基于表10中#3的YOLOv10-S验证：以Transformer块（MHSA+FFN，标记为`"Trans."`）为基线时，如表13所示，PSA在降低0.05 ms延迟的同时提升AP 0.3%，其性能增益可能源于通过减少注意力头冗余缓解了自注意力的优化问题[68,10]。不同N_{PSA}值实验表明（表13），当N_{PSA}=2时获得0.2%AP提升但需付出0.1 ms延迟代价，故默认设为1以平衡性能与效率。

5 Conclusion

5 结论

In this paper, we target both the post-processing and model architecture throughout the detection pipeline of YOLOs. For the post-processing, we propose the consistent dual assignments for NMS-free training, achieving efficient end-to-end detection. For the model architecture, we introduce the holistic efficiency-accuracy driven model design strategy, improving the performance-efficiency tradeoffs. These bring our YOLOv10, a new real-time end-to-end object detector. Extensive experiments show that YOLOv10 achieves the state-of-the-art performance and latency compared with other advanced detectors, well demonstrating its superiority.

本文针对YOLO检测流程中的后处理和模型架构进行优化：提出无NMS训练的一致性双分配策略实现高效端到端检测；引入全栈效率-精度驱动的模型设计方法改善性能-效率权衡。由此推出的YOLOv10成为新型实时端到端目标检测器。大量实验表明，相比先进检测器，YOLOv10在性能和延迟方面均达到最先进水平，充分证明其优越性。

6 Acknowledgments

6 致谢

This work was supported by National Natural Science Foundation of China (Nos. 61925107, 62271281) and Beijing Natural Science Foundation (No. L223023).

本研究由国家自然科学基金（61925107、62271281）和北京市自然科学基金（L223023）资助。

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020.

[3] Daniel Bogdoll, Maximilian Nitsche, and J Marius Zöllner. Anomaly detection in autonomous driving: A survey. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 4488-4499, 2022.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In European conference on computer vision, pages 213-229. Springer, 2020.

[5] Qiang Chen, Xiaokang Chen, Jian Wang, Shan Zhang, Kun Yao, Haocheng Feng, Junyu Han, Errui Ding, Gang Zeng, and Jingdong Wang. Group detr: Fast detr training with group-wise one-to-many assignment. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 6633-6642, 2023.

[6] Yiqun Chen, Qiang Chen, Qinghao Hu, and Jian Cheng. Date: Dual assignment for end-to-end fully convolutional object detection. arXiv preprint arXiv:2211.13859, 2022.

[7] Yiqun Chen, Qiang Chen, Peize Sun, Shoufa Chen, Jingdong Wang, and Jian Cheng. Enhancing your trained detrs with box refinement. arXiv preprint arXiv:2307.11828, 2023.

[8] Yuming Chen, Xinbin Yuan, Ruiqi Wu, Jiabao Wang, Qibin Hou, and Ming-Ming Cheng. Yolo-ms: rethinking multi-scale representation learning for real-time object detection. arXiv preprint arXiv:2308.05480, 2023.

[9] François Chollet. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251-1258, 2017.

[10] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11963-11975, 2022.

[11] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 13733-13742, 2021.

[12] Douglas Henke Dos Reis, Daniel Welfer, Marco Antonio De Souza Leite Cuadros, and Daniel Fernando Tello Gamarra. Mobile robot navigation using an object recognition software with rgbd images and the yolo algorithm. Applied Artificial Intelligence, 33(14):1290-1305, 2019.

[13] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In Proceedings of the IEEE/CVF international conference on computer vision, pages 6569-6578, 2019.

[14] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12873-12883, 2021.

[15] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 3490-3499. IEEE Computer Society, 2021.

[16] Ruili Feng, Kecheng Zheng, Yukun Huang, Deli Zhao, Michael Jordan, and Zheng-Jun Zha. Rank diminishing in deep neural networks. Advances in Neural Information Processing Systems, 35:33054-33065, 2022.

[17] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. arXiv preprint arXiv:2107.08430, 2021.

[18] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2918-2928, 2021.

[19] Ross Girshick. Fast r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 1440-1448, 2015.

[20] Jocher Glenn. Yolov5 release v7.0. https://github.com/ultralytics/yolov5/tree/ v7.0,2022.

[21] Jocher Glenn. Yolov8. In th tps://github. com/ultralytics/ultralytics/tree/ main,2023.

[22] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. In Proceedings of the IEEE/CVF international conference on computer vision, pages 12259- 12269, 2021.

[23] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 2961-2969, 2017.

[24] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. Learning non-maximum suppression. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4507-4515, 2017.

[25] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[26] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3588-3597, 2018.

[27] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, pages 448-456. pmlr, 2015.

[28] Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. Detrs with hybrid matching. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 19702-19712, 2023.

[29] Chuyi Li, Lulu Li, Yifei Geng, Hongliang Jiang, Meng Cheng, Bo Zhang, Zaidan Ke, Xiaoming Xu, and Xiangxiang Chu. Yolov6 v3.0: A full-scale reloading. arXiv preprint arXiv:2301.05586, 2023.

[30] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 13619-13627, 2022.

[31] Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11632-11641, 2021.

[32] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. Advances in Neural Information Processing Systems, 33:21002-21012, 2020.

[33] Ming Lin, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Neural architecture design for gpu-efficient networks. arXiv preprint arXiv:2006.14090, 2020.

[34] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pages 2980-2988, 2017.

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision-ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, pages 740-755. Springer, 2014.

[36] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. arXiv preprint arXiv:2201.12329, 2022.

[37] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8759-8768, 2018.

[38] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF international conference on computer vision, pages 10012-10022, 2021.

[39] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11976-11986, 2022.

[40] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. Advances in neural information processing systems, 29, 2016.

[41] Chengqi Lyu, Wenwei Zhang, Haian Huang, Yue Zhou, Yudong Wang, Yanyi Liu, Shilong Zhang, and Kai Chen. Rtmdet: An empirical study of designing real-time object detectors. arXiv preprint arXiv:2212.07784, 2022.

[42] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In Proceedings of the IEEE/CVF international conference on computer vision, pages 3651-3660, 2021.

[43] Haodong Ouyang. Deyov2: Rank feature with greedy matching for end-to-end object detection. arXiv preprint arXiv:2306.09165, 2023.

[44] Haodong Ouyang. Deyov3: Detr with yolo for real-time object detection. arXiv preprint arXiv:2309.11851, 2023.

[45] Haodong Ouyang. Deyo: Detr with yolo for end-to-end object detection. arXiv preprint arXiv:2402.16370, 2024.

[46] Victor M Panaretos and Yoav Zemel. Statistical aspects of wasserstein distances. Annual review of statistics and its application, 6:405-431, 2019.

[47] Joseph Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/ darknet/, 2013-2016.

[48] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[49] Joseph Redmon and Ali Farhadi. Yolo90000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[50] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[51] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4510-4520, 2018.

[52] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In Proceedings of the IEEE/CVF international conference on computer vision, pages 8430-8439, 2019.

[53] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end people detection in crowded scenes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2325-2333, 2016.

[54] Yuchen Su, Zhineng Chen, Zhiwen Shao, Yuning Du, Zhilong Ji, Jinfeng Bai, Yong Zhou, and Yu-Gang Jiang. Lranet: Towards accurate and efficient scene text detection with low-rank approximation network. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 4979-4987, 2024.

[55] Peize Sun, Yi Jiang, Enze Xie, Wenqi Shao, Zehuan Yuan, Changhu Wang, and Ping Luo. What makes for end-to-end object detection? In International Conference on Machine Learning, pages 9934-9944. PMLR, 2021.

[56] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 14454-14463, 2021.

[57] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: A simple and strong anchor-free object detector. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(4):1922-1933, 2020.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[59] Ao Wang, Hui Chen, Zijia Lin, Hengjun Pu, and Guiguang Ding. Repvit: Revisiting mobile cnn from vit perspective. arXiv preprint arXiv:2307.09283, 2023.

[60] Chengcheng Wang, Wei He, Ying Nie, Jianyuan Guo, Chuanjian Liu, Yunhe Wang, and Kai Han. Gold-yolo: Efficient object detector via gather-and-distribute mechanism. Advances in Neural Information Processing Systems, 36, 2024.

[61] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. In Proceedings of the IEEE/cvf conference on computer vision and pattern recognition, pages 13029-13038, 2021.

[62] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7464-7475, 2023.

[63] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pages 390-391, 2020.

[64] Chien-Yao Wang, Hong-Yuan Mark Liao, and I-Hau Yeh. Designing network design strategies through gradient path analysis. arXiv preprint arXiv:2211.04800, 2022.

[65] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. arXiv preprint arXiv:2402.13616, 2024.

[66] Jianfeng Wang, Lin Song, Zeming Li, Hongbin Sun, Jian Sun, and Nanning Zheng. End-to-end object detection with fully convolutional network. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 15849-15858, 2021.

[67] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. In Proceedings of the AAAI conference on artificial intelligence, volume 36, pages 2567-2575, 2022.

[68] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In Proceedings of the IEEE/CVF international conference on computer vision, pages 22-31, 2021.

[69] Haiyang Xu, Zhichao Zhou, Dongliang He, Fu Li, and Jingdong Wang. Vision transformer with attention map hallucination and ffn compaction. arXiv preprint arXiv:2306.10875, 2023.

[70] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, et al. Pp-yoloe: An evolved version of yolo. arXiv preprint arXiv:2203.16250, 2022.

[71] Xianzhe Xu, Yiqi Jiang, Weihua Chen, Yilun Huang, Yuan Zhang, and Xiuyu Sun. Damo-yolo: A report on real-time object detection design. arXiv preprint arXiv:2211.15444, 2022.

[72] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In European Conference on Computer Vision, pages 659-675. Springer, 2022.

[73] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. arXiv preprint arXiv:2203.03605, 2022.

[74] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412, 2017.

[75] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 9759-9768, 2020.

[76] Wenqiang Zhang, Zilong Huang, Guozhong Luo, Tao Chen, Xinggang Wang, Wenyu Liu, Gang Yu, and Chunhua Shen. Topformer: Token pyramid transformer for mobile semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12083-12093, 2022.

[77] Chuyang Zhao, Yifan Sun, Wenhao Wang, Qiang Chen, Errui Ding, Yi Yang, and Jingdong Wang. Ms-detr: Efficient detr training with mixed supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 17027-17036, 2024.

[78] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. Detrs beat yolos on real-time object detection. arXiv preprint arXiv:2304.08069, 2023.

[79] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 12993-13000, 2020.

[80] Qiang Zhou and Chaohui Yu. Object detection made simpler by eliminating heuristic nms. IEEE Transactions on Multimedia, 2023.

[81] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020.

[82] Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training. In Proceedings of the IEEE/CVF international conference on computer vision, pages 6748-6758, 2023.

A Appendix

A 附录

A.1 Implementation Details

A.1 实现细节

Following [21, 62, 65], all YOLOv10 models are trained from scratch using the SGD optimizer for 500 epochs. The SGD momentum and weight decay are set to 0.937 and 5×10^{-4} ,respectively. The initial learning rate is 1×10^{-2} and it decays linearly to 1×10^{-4} . For data augmentation,we adopt the Mosaic [2, 20], Mixup [74] and copy-paste augmentation [18], etc., like [21, 65]. Tab. 14] presents the detailed hyper-parameters. All models are trained on 8 NVIDIA 3090 GPUs. Besides, we increase the width scale factor of YOLOv10-M to 1.0 to obtain YOLOv10-B. For PSA, we employ it after the SPPF module [21] and adopt the expansion factor of 2 for FFN. For CIB, we also adopt the expansion ratio of 2 for the inverted bottleneck block structure. Following [65, 62], we report the standard mean average precision (AP) across different object scales and IoU thresholds on the COCO dataset [35].

参照[21,62,65]的方法，所有YOLOv10模型均采用SGD优化器从头开始训练500个周期。SGD动量（momentum）和权重衰减（weight decay）分别设置为0.937和5×10^{-4}。初始学习率为1×10^{-2}，并线性衰减至1×10^{-4}。数据增强方面，我们采用Mosaic[2,20]、Mixup[74]和复制粘贴增强[18]等技术，与[21,65]类似。表14列出了详细的超参数。所有模型均在8块NVIDIA 3090 GPU上完成训练。此外，我们将YOLOv10-M的宽度缩放因子增至1.0以获得YOLOv10-B。对于PSA模块，我们将其部署在SPPF模块[21]之后，并为FFN采用2倍的扩展因子。CIB模块同样采用倒残差块结构，扩展比设为2。根据[65,62]的规范，我们在COCO数据集[35]上报告了不同目标尺度和IoU阈值下的标准平均精度（AP）。

Moreover, we follow [78] to establish the end-to-end speed benchmark. Since the execution time of NMS is affected by the input, we thus measure the latency on the COCO val set with the batch size of 1, like [78]. We adopt the same NMS hyperparameters used by the detectors during their validation. The TensorRT efficientNMSPlugin is appended for post-processing and the I/O overhead is omitted. We report the average latency across all images.

此外，我们参照[78]建立端到端速度基准测试。由于NMS执行时间受输入影响，因此我们如[78]所述，在batch size为1的COCO验证集上测量延迟。检测器验证阶段使用的NMS超参数保持不变，后处理环节添加TensorRT efficientNMSPlugin插件并忽略I/O开销。最终报告所有图像的平均延迟。

Table 14: Hyper-parameters of YOLOv10.

表14：YOLOv10超参数配置

hyper-parameter	YOLOv10-N/S/M/B/L/X

epochs	500
optimizer	SGD
momentum	0.937
weight decay	5×10^{-4}
warm-up epochs	3
warm-up momentum	0.8
warm-up bias learning rate	0.1
initial learning rate	10^{-2}
final learning rate	10^{-4}
learning rate schedule	linear decay
box loss gain	7.5
class loss gain	0.5
DFL loss gain	1.5
HSV saturation augmentation	0.7
HSV value augmentation	0.4
HSV hue augmentation	0.015
translation augmentation	0.1
scale augmentation	0.5/0.5/0.9/0.9/0.9/0.9
mosaic augmentation	1.0
Mixup augmentation	0.0/0.0/0.1/0.1/0.15/0.15
copy-paste augmentation	0.0/0.0/0.1/0.1/0.3/0.3
close mosaic epochs	10

超参数(hyper-parameter)	YOLOv10-N/S/M/B/L/X
训练轮次(epochs)	500
优化器(optimizer)	随机梯度下降(SGD)
动量(momentum)	0.937
权重衰减(weight decay)	5×10^{-4}
预热轮次(warm-up epochs)	3
预热动量(warm-up momentum)	0.8
预热偏置学习率(warm-up bias learning rate)	0.1
初始学习率(initial learning rate)	10^{-2}
最终学习率(final learning rate)	10^{-4}
学习率调度(learning rate schedule)	线性衰减(linear decay)
边界框损失增益(box loss gain)	7.5
分类损失增益(class loss gain)	0.5
DFL损失增益(DFL loss gain)	1.5
HSV饱和度增强(HSV saturation augmentation)	0.7
HSV明度增强(HSV value augmentation)	0.4
HSV色相增强(HSV hue augmentation)	0.015
平移增强(translation augmentation)	0.1
缩放增强(scale augmentation)	0.5/0.5/0.9/0.9/0.9/0.9
马赛克增强(mosaic augmentation)	1.0
混合增强(Mixup augmentation)	0.0/0.0/0.1/0.1/0.15/0.15
复制粘贴增强(copy-paste augmentation)	0.0/0.0/0.1/0.1/0.3/0.3
关闭马赛克轮次(close mosaic epochs)	10

A.2 Details of Consistent Matching Metric

A.2 一致性匹配度量细节

We provide the detailed derivation of consistent matching metric here.

此处我们提供一致性匹配度量的详细推导过程。

As mentioned in the paper,we suppose that the one-to-many positive samples is Ω and the one-toone branch selects i -th prediction. We can then leverage the normalized metric [15] to obtain the classification target for task alignment learning [21,15,65,29,70],i.e., $t_{o2m,j} = u^* \cdot \frac{m_{o2m,j}}{m^*_{o2m}} \leq u^*$ for $j \in \Omega$ and $t_{o2o,i} = u^* \cdot \frac{m_{o2o,i}}{m^*_{o2o}} = u^*$. We can thus derive the supervision gap between two branches by the 1-Wasserstein distance [46] of the different classification targets, i.e.,

如论文所述，假设一对多正样本为Ω且一对一分支选择第i个预测。我们可利用归一化度量[15]获得任务对齐学习[21,15,65,29,70]的分类目标，即$t_{o2m,j} = u^* \cdot \frac{m_{o2m,j}}{m^*_{o2m}} \leq u^*$对应$j \in \Omega$和$t_{o2o,i} = u^* \cdot \frac{m_{o2o,i}}{m^*_{o2o}} = u^*$。进而通过不同分类目标的1-Wasserstein距离[46]推导两分支间的监督差距：

$$A = \left| \left(1 - t_{o2o,i}\right) - \left(1 - I\left(i \in \Omega\right) t_{o2m,i}\right) \right| + \sum_{k \in \Omega \setminus \{i\}} \left| 1 - \left(1 - t_{o2m,k}\right) \right|$$

$$\overset{(3)}{=} \left| t_{o2o,i} - I\left(i \in \Omega\right) t_{o2m,i} \right| + \sum_{k \in \Omega \setminus \{i\}} t_{o2m,k}$$

$$= t_{o2o,i} - I\left(i \in \Omega\right) t_{o2m,i} + \sum_{k \in \Omega \setminus \{i\}} t_{o2m,k},$$

where $I\left(\cdot\right)$ is the indicator function. We denote the classification targets of the predictions in Ω as $\left\{\hat{t}_1, \hat{t}_2, \ \ldots, \hat{t}_{|\Omega|}\right\}$ in descending order,with $\hat{t}_1 \geq \hat{t}_2 \geq \ \ldots \ \geq \hat{t}_{|\Omega|}$. We can then replace $t_{o2o,i}$ with u^* and obtain:

其中$I\left(\cdot\right)$为指示函数。将Ω中预测的分类目标按降序记为$\left\{\hat{t}_1, \hat{t}_2, \ \ldots, \hat{t}_{|\Omega|}\right\}$，其中$\hat{t}_1 \geq \hat{t}_2 \geq \ \ldots \ \geq \hat{t}_{|\Omega|}$。将$t_{o2o,i}$替换为$u^*$可得：

$$A = u^* - I\left(i \in \Omega\right) t_{o2m,i} + \sum_{k \in \Omega \setminus \{i\}} t_{o2m,k}$$

$$\overset{(4)}{=} u^* + \sum_{k \in \Omega} t_{o2m,k} - 2 I\left(i \in \Omega\right) t_{o2m,i}$$

$$= u^* + \sum_{k=1}^{|\Omega|} \hat{t}_k - 2 \cdot I\left(i \in \Omega\right) t_{o2m,i}$$

We further discuss the supervision gap in two scenarios, i.e.,

我们分两种情形讨论监督差距：

1. Supposing $i \notin \Omega$,we can obtain:

1. 假设$i \notin \Omega$，可得：

$$\overline{A} \overline{u^*} + \sum_{k=1}^{|\Omega|} \hat{t}_k \quad (5)$$

2. Supposing $i \in \Omega$,we denote $t_{o2m,i} = \hat{t}_n$ and obtain:

2. 假设$i \in \Omega$，记$t_{o2m,i} = \hat{t}_n$可得：

$$\overline{A} \overline{u^*} + \sum_{k=1}^{|\Omega|} \hat{t}_k \overline{2} \hat{t}_n \quad (6)$$

Due to $\hat{t}_n \geq 0$,the second case can lead to smaller supervision gap. Besides,we can observe that A decreases as $\hat{t}_n$ increases,indicating that n decreases and the ranking of i within Ω improves. Due to $\hat{t}_n \leq \hat{t}_1, A$ thus achieves the minimum when $\hat{t}_n = \hat{t}_1$,i.e., i is the best positive sample in Ω with $m_{o2m,i} = m^*_{o2m}$ and $t_{o2m,i} = u^* \cdot \frac{m_{o2m,i}}{m^*_{o2m}} = u^*$.

由于$\hat{t}_n \geq 0$，第二种情形会产生较小监督差距。此外可观察到A随$\hat{t}_n$增大而减小，表明n降低且i在Ω中的排序提升。因$\hat{t}_n \leq \hat{t}_1, A$，当$\hat{t}_n = \hat{t}_1$时取得最小值，即$i$在$\Omega$中成为最佳正样本，此时$m_{o2m,i} = m^*_{o2m}$且$t_{o2m,i} = u^* \cdot \frac{m_{o2m,i}}{m^*_{o2m}} = u^*$。

Furthermore, we prove that we can achieve the minimized supervision gap by the consistent matching metric. We suppose $\alpha_{o2m} > 0$ and $\beta_{o2m} > 0$,which are common in [21,65,29,15,70]. Similarly, we assume $\alpha_{o2o} > 0$ and $\beta_{o2o} > 0$. We can obtain $r_1 = \frac{\alpha_{o2o}}{\alpha_{o2m}} > 0$ and $r_2 = \frac{\beta_{o2o}}{\beta_{o2m}} > 0$,and then derive m_{o2o} by

进一步证明通过一致性匹配度量可实现最小化监督差距。假设$\alpha_{o2m} > 0$与$\beta_{o2m} > 0$（该假设常见于[21,65,29,15,70]）。类似地设$\alpha_{o2o} > 0$与$\beta_{o2o} > 0$，可得$r_1 = \frac{\alpha_{o2o}}{\alpha_{o2m}} > 0$和$r_2 = \frac{\beta_{o2o}}{\beta_{o2m}} > 0$，进而通过下式推导$m_{o2o}$：

$$m_{o2o} = s \cdot p^{\alpha_{o2o}} \cdot \mathrm{IoU}\left(\hat{b}, b\right)^{\beta_{o2o}}$$

$$\overline{s} p^{r_1 \cdot \alpha_{o2m}} \mathrm{IoU}\left(\hat{b}, b\right)^{r_2 \cdot \beta_{o2m}} \quad (7)$$

$$= s \cdot \left(p^{\alpha_{o2m}} \cdot \mathrm{IoU}\left(\hat{b}, b\right)^{\beta_{o2m}}\right)^{r_1} \cdot \mathrm{IoU}\left(\hat{b}, b\right)^{(r_2 - r_1) \cdot \beta_{o2m}}$$

$$= m_{o2m}^{r_1} \cdot \mathrm{IoU}\left(\hat{b}, b\right)^{\left(r_2 - r_1\right) \cdot \beta_{o2m}}$$

To achieve $m_{o2m,\,i} = m_{o2m}^{*}$ and $m_{o2o,\,i} = m_{o2o}^{*}$,we can make m_{o2o} monotonically increase with m_{o2m} by assigning $\left(r_2 - r_1\right) = 0$,i.e.,

为实现$m_{o2m,\,i} = m_{o2m}^{*}$和$m_{o2o,\,i} = m_{o2o}^{*}$，可通过分配$\left(r_2 - r_1\right) = 0$使$m_{o2o}$随$m_{o2m}$单调递增，即：

$$\begin{aligned}(8)\quad m_{o2o} &= m_{o2m}^{r_1} \\ &\quad \mathrm{IoU}\left(\hat{b}, b\right)^{0 \cdot \beta_{o2m}} \\ &= m_{o2m}^{r_1}\end{aligned}$$

Supposing $r_1 = r_2 = r$,we can thus derive the consistent matching metric,i.e., $\alpha_{o2o} = r \cdot \alpha_{o2m}$ and $\beta_{o2o} = r \cdot \beta_{o2m}$. By simply taking $r = 1$,we obtain $\alpha_{o2o} = \alpha_{o2m}$ and $\beta_{o2o} = \beta_{o2m}$.

假设$r_1 = r_2 = r$，我们可由此推导出一致的匹配度量，即$\alpha_{o2o} = r \cdot \alpha_{o2m}$与$\beta_{o2o} = r \cdot \beta_{o2m}$。仅需取$r = 1$，即可获得$\alpha_{o2o} = \alpha_{o2m}$和$\beta_{o2o} = \beta_{o2m}$。

A.3 Details of Rank-Guided Block Design

A.3 秩引导块设计细节

We present the details of the algorithm of rank-guided block design in Algo. 1 Besides, to calculate the numerical rank of the convolution,we reshape its weight to the shape of $\left(C_o, K^2 \times C_i\right)$,where C_o and C_i denote the number of output and input channels,and K means the kernel size,respectively.

我们在算法1中详述秩引导块设计算法。此外，为计算卷积的数值秩，将其权重重塑为$\left(C_o, K^2 \times C_i\right)$形状，其中$C_o$和$C_i$分别表示输出与输入通道数，$K$代表卷积核尺寸。

A.4 Training Cost Analyses

A.4 训练成本分析

In addition to the inference efficiency analyses, we also investigate the training cost of our YOLOv10 models. We compare with other YOLO variants and measure the training throughput on 8 NVIDIA 3090 GPUs using the official codebases. Tab. 15 presents the comparison results based on the medium model scale. We observe that despite having 500 training epochs, YOLOv10 achieves a high training throughput, making its training cost affordable. We also note that the one-to-many head in the NMS-free training will introduce the extra overhead for YOLOv10. To investigate this, we measure the training cost of YOLOv10 with only the one-to-one head, which is denoted as "YOLOv10-o2o" As shown in Tab. 15, YOLOv10-M results in a small increase in the training time over "YOLOv10- M-o2o", about 18s each epoch, which is affordable. To fairly verify the benefit of the one-to-many head in NMS-free training, we also adopt longer 550 training epochs for "YOLOv10-M-o2o", which leads to a similar training time (29.3 vs. 29.1 hours) but still yields inferior performance (48.9% vs. 51.1% AP) compared with YOLOv10-M.

除推理效率分析外，我们还研究了YOLOv10模型的训练成本。通过与其他YOLO变体对比，使用8块NVIDIA 3090 GPU和官方代码库测量训练吞吐量。表15展示了中规模模型的对比结果。尽管需500个训练周期，YOLOv10仍保持较高训练吞吐量，使训练成本可控。值得注意的是，无NMS训练中的一对多头会带来额外开销。为验证此点，我们测量了仅含一对一头的"YOLOv10-o2o"训练成本。如表15所示，YOLOv10-M较"YOLOv10-M-o2o"每周期仅增加约18秒训练时间，尚属可接受范围。为公平验证一对多头在无NMS训练中的优势，我们对"YOLOv10-M-o2o"采用550个训练周期，虽总训练时间相近（29.3 vs. 29.1小时），但性能仍逊于YOLOv10-M（48.9% vs. 51.1% AP）。

Algorithm 1: Rank-guided block design

算法1：秩引导块设计

Input: Intrinsic ranks R for all stages S ; Original Network Θ ; CIB θ_{cib} ;

输入：各阶段S的本征秩R；原始网络Θ；CIB模块θ_{cib}

Output: New network Θ^* with CIB for certain stages.

输出：含特定阶段CIB的新网络Θ^*

$t \leftarrow 0$;

$\Theta_0 \leftarrow \Theta; \Theta^* \leftarrow \Theta_0$;

$ap_0 \leftarrow \mathrm{AP}\left(\mathrm{T}\left(\Theta_0\right)\right)$; / / T:training the network; AP:evaluating the AP performance.

$ap_0 \leftarrow \mathrm{AP}\left(\mathrm{T}\left(\Theta_0\right)\right)$; / / T：训练网络；AP：评估AP性能

while $S \neq \emptyset$ do

当$S \neq \emptyset$时

$s_t \leftarrow \mathrm{argmin}_{s \in S} R$;

$\Theta_{t+1} \leftarrow \mathrm{Replace}\left(\Theta_t, \theta_{cib}, s_t\right)$; / / Replace the block in Stage s_t of Θ_t with CIB θ_{cib} .

$\Theta_{t+1} \leftarrow \mathrm{Replace}\left(\Theta_t, \theta_{cib}, s_t\right)$; / / 将Θ_t的第s_t阶段块替换为CIBθ_{cib}

$ap_{t+1} \leftarrow \mathrm{AP}\left(\mathrm{T}\left(\Theta_{t+1}\right)\right)$;

if $ap_{t+1} \geq ap_0$ then

若$ap_{t+1} \geq ap_0$成立

$\Theta^* \leftarrow \Theta_{t+1}; S \leftarrow S \setminus \{s_t\}$;

else

否则

return Θ^* ;

返回Θ^*

end

结束

end

结束

return Θ^* ;

返回 Θ^*；

Table 15: Training cost analyses on 8 NVIDIA 3090 GPUs.

表15：基于8块NVIDIA 3090显卡的训练成本分析。

Model	Epoch	Speed (epoch/hour)	Time (hour)
YOLOv6-3.0-M	300	7.2	41.7
YOLOv8-M	500	18.3	27.3
YOLOv9-M	500	12.3	40.7
Gold-YOLO-M	300	4.7	63.8
YOLO-MS	300	7.1	42.3
YOLOv10-M-o2o	500	18.8	26.7
YOLOv10-M	500	17.2	29.1

模型	训练轮次	速度（轮次/小时）	时间（小时）
YOLOv6-3.0-M	300	7.2	41.7
YOLOv8-M	500	18.3	27.3
YOLOv9-M	500	12.3	40.7
Gold-YOLO-M	300	4.7	63.8
YOLO-MS	300	7.1	42.3
YOLOv10-M-o2o	500	18.8	26.7
YOLOv10-M	500	17.2	29.1

Table 16: Latency with NMS.

表16：采用非极大值抑制(NMS)的延迟情况。

Model	Latency
YOLOv10-N	6.19ms
YOLOv10-S	7.15ms
YOLOv10-M	9.03ms
YOLOv10-B	10.04ms
YOLOv10-L	11.52ms
YOLOv10-X	14.67ms

模型	延迟
YOLOv10-N	6.19毫秒
YOLOv10-S	7.15毫秒
YOLOv10-M	9.03毫秒
YOLOv10-B	10.04毫秒
YOLOv10-L	11.52毫秒
YOLOv10-X	14.67毫秒

A.5 More Results on COCO

A.5 COCO数据集上的更多结果

We measure the latency of YOLOv10 with the original one-to-many training using NMS and report the results on COCO in Tab. 16. Besides, we report the detailed performance of YOLOv10, including AP_{50}^{val} and AP_{75}^{val} at different IoU thresholds,as well as $\mathrm{AP}_{\mathrm{small}}^{val}$, $\mathrm{AP}_{\mathrm{medium}}^{val}$,and

AP^{val}_{large} across different scales, in Tab. 17. We also present the comparisons with more lightweight detectors, including DAMO-YOLO [71], YOLOv7 [62], and DEYO [45], in Tab. 18. It shows that our YOLOv10 also achieves superior performance and efficiency trade-offs. Additionally, in experiments, we follow previous works [21, 65] to train the models for 500 epochs. We also conduct experiments to train the models for 300 epochs and present the comparison results with YOLOv6 [29], Gold-YOLO [60], and YOLO-MS [8] which adopt 300 epochs, in Tab. 19. We observe that our YOLOv10 also exhibits better performance and inference latency. We also note that despite trained for 500 epochs, YOLOv10 has less training cost compared with these models as presented in Tab. 15

我们测量了采用原始一对多训练方式和NMS的YOLOv10的延迟，并在表16中报告了COCO数据集上的结果。此外，我们在表17中详细列出了YOLOv10的性能指标，包括不同IoU阈值下的AP^{val}_{50}和AP^{val}_{75}，以及不同尺度下的AP^{val}_{small}，AP^{val}_{medium}和AP^{val}_{large}。表18展示了与更轻量级检测器（包括DAMO-YOLO[71]、YOLOv7[62]和DEYO[45]）的对比结果，表明我们的YOLOv10在性能与效率的平衡上同样表现出色。实验中我们遵循前人工作[21,65]采用500轮训练周期，同时补充了300轮训练的对比实验，在表19中与同样采用300轮训练的YOLOv6[29]、Gold-YOLO[60]和YOLO-MS[8]进行对比。值得注意的是，尽管采用500轮训练，如表15所示，YOLOv10的训练成本仍低于这些对比模型。

A.6 Inference Efficiency Comparison on CPU

A.6 CPU端推理效率对比

We present the speed comparison results of YOLOv10 and others on CPU (Intel Xeon Skylake, IBRS) using OpenVINO in Fig. 5. We observe that YOLOv10 also shows state-of-the-art trade-offs in terms of performance and efficiency.

图5展示了YOLOv10与其他模型在CPU（Intel Xeon Skylake, IBRS）上使用OpenVINO的推理速度对比，我们的模型在性能与效率平衡方面继续保持领先优势。

A.7 More Analyses for Holistic Efficiency-Accuracy Driven Model Design

A.7 效率-精度协同驱动模型设计的深入分析

We note that reducing the latency of YOLOv10-S (#2 in Tab. 2) is particularly challenging due to its small model scale. However, as shown in Tab. 2, our efficiency driven model design still achieves a 5.3% reduction in latency without compromising performance. This provides substantial support for the further accuracy driven model design. YOLOv10-S achieves a better latency-accuracy trade-off with our holistic efficiency-accuracy driven model design,showing a 2.0% AP improvement with only 0.05 m s latency overhead. Besides,for YOLOv10-M (#6 in Tab. 2),which has a larger model scale and more redundancy, our efficiency driven model design results in a considerable 12.5% latency reduction, as shown in Tab. 2. When combined with accuracy driven model design, we observe a notable 0.8% AP improvement for YOLOv10-M,along with a favorable latency reduction of 0.48ms. These results well demonstrate the effectiveness of our design strategy across different model scales.

由于模型规模较小，降低YOLOv10-S（表2中#2）的延迟具有特殊挑战性。但如表2所示，我们的效率驱动设计仍实现了5.3%的延迟降低且性能无损，这为后续精度驱动设计奠定了坚实基础。通过效率-精度协同设计，YOLOv10-S在仅增加0.05 m秒延迟的情况下获得2.0%AP提升。对于规模更大、冗余度更高的YOLOv10-M（表2中#6），效率驱动设计带来12.5%的显著延迟降低，结合精度驱动设计后更实现0.8%AP提升与0.48ms延迟降低的双重优化。这些结果充分验证了我们设计策略在不同模型规模上的有效性。

Table 17: Detailed performance of YOLOv10 on COCO.

表17：YOLOv10在COCO数据集上的详细性能指标

Model	AP^{val}（%）	AP^{val}_{50}（%）	AP^{val}_{75}（%）	AP^{val}_{small}（%）	AP^{val}_{medium}（%）	AP^{val}_{large}（%）
YOLOv10-N	38.5	53.8	41.7	18.9	42.4	54.6
YOLOv10-S	46.3	63.0	50.4	26.8	51.0	63.8
YOLOv10-M	51.1	68.1	55.8	33.8	56.5	67.0
YOLOv10-B	52.5	69.6	57.2	35.1	57.8	68.5
YOLOv10-L	53.2	70.1	58.1	35.8	58.5	69.4
YOLOv10-X	54.4	71.3	59.3	37.0	59.8	70.9

模型	AP^{val}（%）	AP^{val}_{50}（%）	AP^{val}_{75}（%）	AP^{val}_{small}（%）	AP^{val}_{medium}（%）	AP^{val}_{large}（%）
YOLOv10-N	38.5	53.8	41.7	18.9	42.4	54.6
YOLOv10-S	46.3	63.0	50.4	26.8	51.0	63.8

YOLOv10-M	51.1	68.1	55.8	33.8	56.5	67.0
YOLOv10-B	52.5	69.6	57.2	35.1	57.8	68.5
YOLOv10-L	53.2	70.1	58.1	35.8	58.5	69.4
YOLOv10-X	54.4	71.3	59.3	37.0	59.8	70.9

Table 18: Comparisons with more lightweight detectors.

表18：与更轻量级检测器的对比

Model	#Param.(M)	FLOPs(G)	AP^{val} (%)	Latency(ms)
DEYO-tiny 45	4.0	8.0	37.6	2.01
YOLOv10-N	2.3	6.7	38.5	1.84
DAMO-YOLO-T [71]	8.5	18.1	42.0	2.21
DAMO-YOLO-S [71]	16.3	37.8	46.0	3.18
DEYO-S 45	14.0	26.0	45.8	3.34
YOLOv10-S	7.2	21.6	46.3	2.49
DAMO-YOLO-M [71]	28.2	61.8	49.2	4.57
DAMO-YOLO-L [71]	42.1	97.3	50.8	6.48
DEYO-M 45	33.0	78.0	50.7	7.14
YOLOv10-M	15.4	59.1	51.1	4.74
YOLOv7 62	36.9	104.7	51.2	17.03
YOLOv10-B	19.1	92.0	52.5	5.74
YOLOv7-X [62]	71.3	189.9	52.9	21.45
DEYO-L [45]	51.0	155.0	52.7	10.00
YOLOv10-L	24.4	120.3	53.2	7.28
DEYO-X 45	78.0	242.0	53.7	15.38
YOLOv10-X	29.5	160.4	54.4	10.70

模型	参数量(百万)	浮点运算量(十亿次)	AP^{val} (%)	延迟(毫秒)
DEYO-tiny 45	4.0	8.0	37.6	2.01
YOLOv10-N	2.3	6.7	38.5	1.84
DAMO-YOLO-T [71]	8.5	18.1	42.0	2.21
DAMO-YOLO-S [71]	16.3	37.8	46.0	3.18
DEYO-S 45	14.0	26.0	45.8	3.34
YOLOv10-S	7.2	21.6	46.3	2.49
DAMO-YOLO-M [71]	28.2	61.8	49.2	4.57
DAMO-YOLO-L [71]	42.1	97.3	50.8	6.48
DEYO-M 45	33.0	78.0	50.7	7.14
YOLOv10-M	15.4	59.1	51.1	4.74
YOLOv7 62	36.9	104.7	51.2	17.03
YOLOv10-B	19.1	92.0	52.5	5.74
YOLOv7-X [62]	71.3	189.9	52.9	21.45
DEYO-L [45]	51.0	155.0	52.7	10.00
YOLOv10-L	24.4	120.3	53.2	7.28
DEYO-X 45	78.0	242.0	53.7	15.38
YOLOv10-X	29.5	160.4	54.4	10.70

Table 19: Performance comparisons under 300 training epochs.

表19：300训练周期下的性能对比。

Model	#Param.(M)	FLOPs(G)	AP^{val} (%)	Latency(ms)
YOLOv6-3.0-N [29]	4.7	11.4	37.0	2.69
Gold-YOLO-N [60]	5.6	12.1	39.6	2.92
YOLOv10-N (Ours)	2.3	6.7	37.7	1.84
YOLOv6-3.0-S [29	18.5	45.3	44.3	3.42
Gold-YOLO-S [60	21.5	46.0	45.4	3.82
YOLO-MS-XS [8]	4.5	17.4	43.4	8.23
YOLOv10-S (Ours)	7.2	21.6	45.6	2.49
YOLOv6-3.0-M [29	34.9	85.8	49.1	5.63
Gold-YOLO-M [60]	41.3	87.5	49.8	6.38
YOLOv10-M (Ours)	15.4	59.1	50.3	4.74
YOLOv6-3.0-L [29]	59.6	150.7	51.8	9.02
Gold-YOLO-L [60	75.1	151.7	51.8	10.65
YOLO-MS [8]	22.2	80.2	51.0	12.41
YOLOv10-B (Ours)	19.1	92.0	51.6	5.74
YOLOv10-L (Ours)	24.4	120.3	52.4	7.28
YOLOv10-X (Ours)	29.5	160.4	53.6	10.70

模型	参数量(百万)	计算量(十亿次)	AP^{val} (%)	延迟(毫秒)
YOLOv6-3.0-N [29]	4.7	11.4	37.0	2.69
Gold-YOLO-N [60]	5.6	12.1	39.6	2.92
YOLOv10-N (本模型)	2.3	6.7	37.7	1.84
YOLOv6-3.0-S [29	18.5	45.3	44.3	3.42
Gold-YOLO-S [60	21.5	46.0	45.4	3.82
YOLO-MS-XS [8]	4.5	17.4	43.4	8.23
YOLOv10-S (本模型)	7.2	21.6	45.6	2.49
YOLOv6-3.0-M [29	34.9	85.8	49.1	5.63
Gold-YOLO-M [60]	41.3	87.5	49.8	6.38
YOLOv10-M (本模型)	15.4	59.1	50.3	4.74
YOLOv6-3.0-L [29]	59.6	150.7	51.8	9.02
Gold-YOLO-L [60	75.1	151.7	51.8	10.65
YOLO-MS [8]	22.2	80.2	51.0	12.41
YOLOv10-B (本模型)	19.1	92.0	51.6	5.74
YOLOv10-L (本模型)	24.4	120.3	52.4	7.28
YOLOv10-X (本模型)	29.5	160.4	53.6	10.70

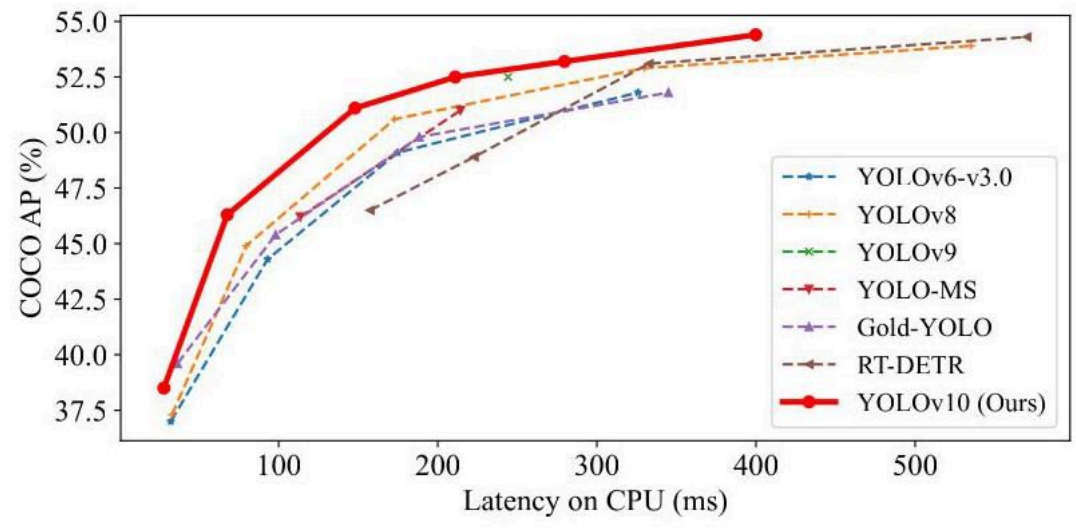

Figure 5: Performance and efficiency comparisons on CPU.

图5：CPU上的性能与效率对比。

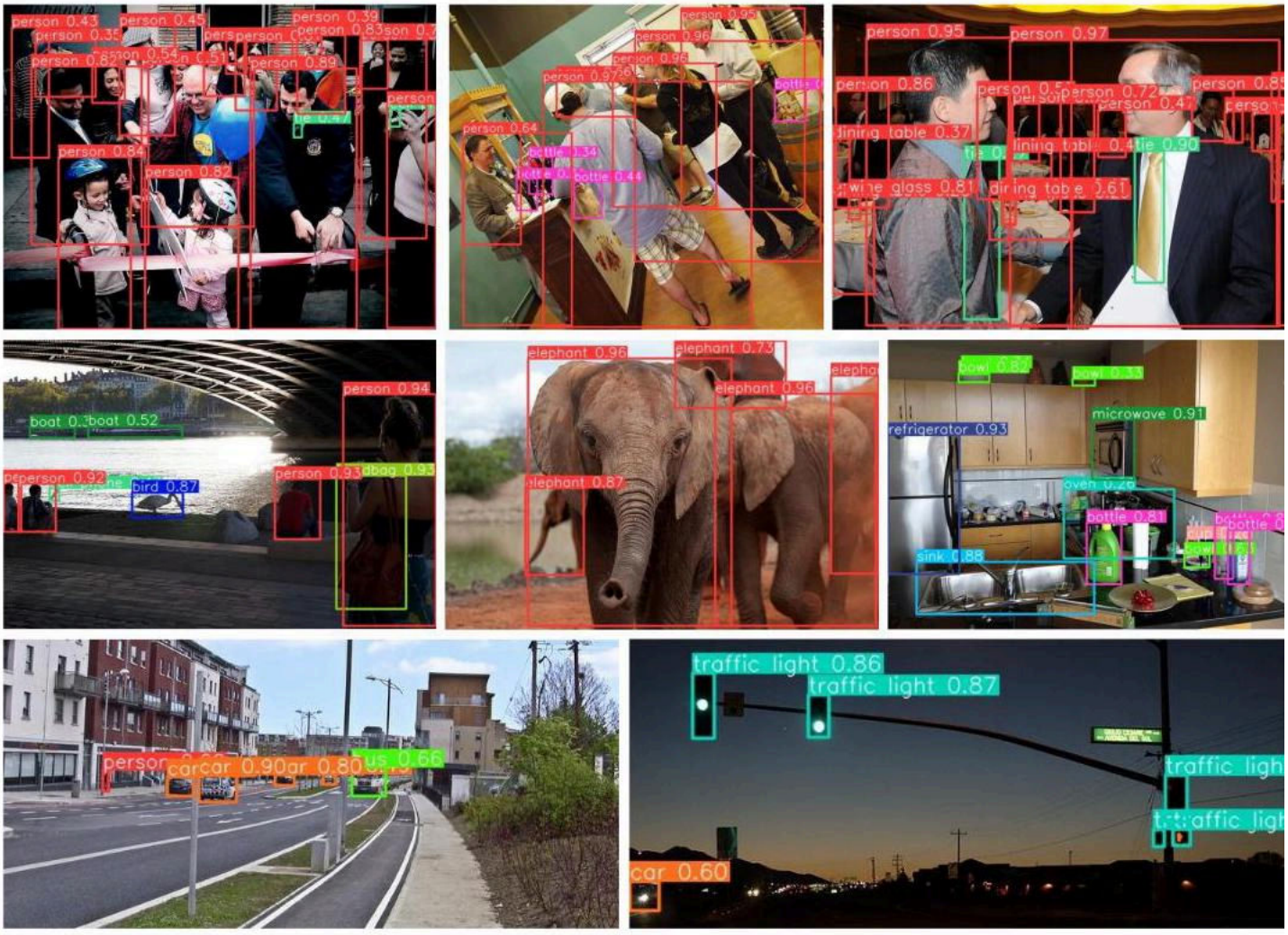

Figure 6: Visualization results under complex and challenging scenarios.

图6：复杂挑战场景下的可视化结果。

A.8 Visualization Results

A.8 可视化结果

Fig. 6 presents the visualization results of our YOLOv10 in the complex and challenging scenarios. It can be observed that YOLOv10 can achieve precise detection under various difficult conditions, such as low light, rotation, etc. It also demonstrates a strong capability in detecting diverse and densely packed objects, such as bottle, cup, and person. These results indicate its superior performance.

图6展示了YOLOv10在复杂挑战场景中的可视化结果。可见该模型在低光照、旋转等多种困难条件下仍能实现精准检测，同时对瓶罐、杯具、行人等密集多样目标展现出强大识别能力，这些结果印证了其卓越性能。

A.9 Contribution, Limitation, and Broader Impact

A.9 贡献、局限性与广泛影响

Contribution. In summary, our contributions are three folds as follows:

贡献。总结而言，我们的贡献主要体现在以下三方面：

1. We present a novel consistent dual assignments strategy for NMS-free YOLOs. A dual label assignments way is designed to provide rich supervision by one-to-many branch during training and high efficiency by one-to-one branch during inference. Besides, to ensure the harmonious supervision between two branches, we innovatively propose the consistent matching metric, which can well reduce the theoretical supervision gap and lead to improved performance.

1. 提出面向无NMS的YOLO系列的新型双标签分配策略：通过训练阶段一对多分支的丰富监督与推理阶段一对一分支的高效执行实现双重优化。创新性设计的一致性匹配度量方法，有效缩小理论监督差距，提升模型性能。

2. We propose a holistic efficiency-accuracy driven model design strategy for the model architecture of YOLOs. We present novel lightweight classification head, spatial-channel decoupled down-sampling, and rank-guided block design, which greatly reduce the computational redundancy and achieve high efficiency. We further introduce the large-kernel convolution and innovative partial self-attention module, which effectively enhance the performance under low cost.

2. 构建效率-精度协同驱动的YOLO架构设计范式：开发轻量化分类头、空间-通道解耦下采样及秩导向模块设计，显著降低计算冗余；引入大核卷积与创新性局部自注意力模块，以低成本实现性能跃升。

3. Based on the above approaches, we introduce YOLOv10, a new real-time end-to-end object detector. Extensive experiments demonstrate that our YOLOv10 achieves the state-of-the-art performance and efficiency trade-offs compared with other advanced detectors.

3. 基于上述方法推出新一代实时端到端检测器YOLOv10。大量实验表明，相较先进检测器，YOLOv10实现了最优的性能-效率平衡。

Limitation. Due to the limited computational resources, we do not investigate the pretraining of YOLOv10 on large-scale datasets, e.g., Objects365 [52]. Besides, although we can achieve competitive end-to-end performance using the one-to-one head under NMS-free training, there still exists a performance gap compared with the original one-to-many training using NMS, especially noticeable in small models. For example, in YOLOv10-N and YOLOv10-S, the performance of one-to-many training with NMS outperforms that of NMS-free training by 1.0% AP and 0.5% AP, respectively. We will explore ways to further reduce the gap and achieve higher performance for YOLOv10 in the future work.

局限性。受计算资源限制，未开展大规模数据集（如Objects365[52]）预训练研究。尽管无NMS训练下通过一对一检测头可获得竞争力表现，但较传统NMS辅助的一对多训练仍存在性能差距（YOLOv10-N/S模型分别有1.0%/0.5% AP差距）。后续将着力缩小该差距。

Broader impact. The YOLOs can be widely applied in various real-world applications, including medical image analyses and autonomous driving, etc. We hope that our YOLOv10 can assist in these fields and improve the efficiency. However, we acknowledge the potential for malicious use of our models. We will make every effort to prevent this.

广泛影响。YOLO系列可广泛应用于医疗影像分析、自动驾驶等领域。我们希望YOLOv10能提升相关领域效率，同时警惕模型被恶意使用的可能性，并将全力防范此类情况。