YOLO-MS：对实时目标检测中多尺度表征学习的重新思考

陈宇铭，袁信彬，王家宝，武睿祺，李翔，侯淇彬， 程明明

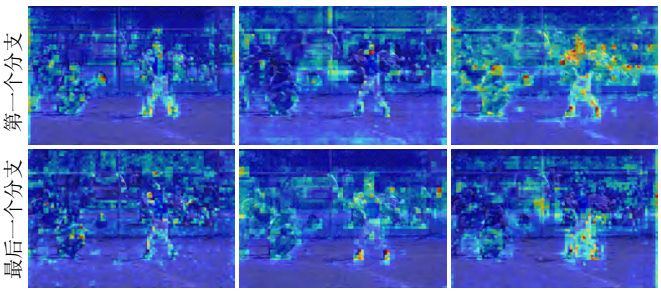
摘要—我们旨在为目标检测社区提供一种高效且性能优越的目标检测器，称为 YOLO-MS。其核心设计基于一系列关于基本块的多分支特征及不同卷积核大小对不同尺度目标检测性能影响的研究。我们最终提出了一种新方法，能够显著增强实时目标检测器的多尺度特征表示能力。为了验证我们工作的有效性，我们在 MS COCO 数据集上从零开始训练 YOLO-MS，不依赖于任何其他大规模数据集（如 ImageNet）或预训练权重。无需花哨的技巧，YOLO-MS 便可超越当前的 SOTA（最先进的）实时目标检测方法，包括 YOLO-v7、RTMDet 和 YOLO-v8。以 YOLO-MS 的 XS 版本为例，其在 MS COCO 上的 AP（平均精确度）得分为 42%+，高 RTMDet 的同等大小模型约 2%。此外，我们的工作还可以作为即插即用模块用于其他 YOLO 模型。例如，我们的方法能够显著提升 YOLOv8-N 的 APs、API 和 AP，从 18%+、52%+ 和 37%+ 分别提高到 20%+、55%+ 和 40%+，同时参数量和 MACs（乘加累积操作数）更少。代码和预训练模型已公开发布在 https://github.com/FishAndWasabi/YOLO-MS.
我们还提供了 Jittor 版本，详见 https://github.com/NK-JittorCV/nk-yolo.

Index Terms—目标检测；实时目标检测；多尺度表征学习

1 引言

以 YOLO 系列 [53], [54], [55], [1], [65], [28], [16], [34], [66], [48], [73], [29], [80], [64], [69], [63] 为代表的实时目标检测器在工业领域得到了广泛的应用，特别是在无人机和机器人等边缘设备上。不同于以往的目标检测器 [82], [75], [56], [3], [61], [7]，实时目标检测器旨在寻求速度与精度之间的最佳平衡。为此，研究者们提出了众多研究以设计高效且强大的实时目标检测架构 [55], [68], [67], [66]。从第一代 DarkNet [55] 到 CSPNet [68]，再到近期扩展的 ELAN [66]，实时目标检测器的架构发生了显著变化，同时性能也得到了快速提升。

尽管这些工作取得了可观的性能，但对于实时目标检测器而言，识别多尺度目标仍是一项基本挑战。现有的实时目标检测器主要通过改进宏观结构来增强多尺度特征表示。典型方法包括 FPN [41]、PAFPN [45] 和聚集-分发（Gather-Distribute）机制 [64]，它们主要通过修改模型颈部（neck）来改善不同尺度特征的聚合。尽管这些方法取得了成功，但它们忽略了在基本构建块中学习多尺度特征表示的重要性。近期 YOLO 系列的进



模型	YOLOv7	RTMDet	Res2Net
参数量	6.2M	4.9M	4.9M
MACs	6.9G	8.1G	8.1G
APs	19.9	20.7	21.3
$\mathcal{D}$	1.6×1e-3	2.1×1e-3	2.9×1e-3

图 1. 不同 YOLO 模型的分支特征多样性比较。简单起见，仅展示了两个分支的特征可视化结果，这足以证明我们方法在增强特征多样性方面的有效性。在表格中，$\mathcal{D}$ 是一项用于衡量检测器分支间特征多样性的直观指标[1]。

- 所有作者均来自南开大学计算机学院媒体计算实验室，中国天津（通讯作者：侯淇彬）。
- 本研究受到了国家自然科学基金（*NO. 62225604*，*No. 62276145*）和中央高校基本科研业务费（南开大学，*070-63223049*）的支持。计算受到了南开大学高性能计算中心（*NKSC*）的支持。

1. 假设给定数据集中包含 N_d 张图片，且模型中包含 N_m 个块。设 $\mathbf{f}_i(d,m)$ 为第 d 张图像中第 m 个块的第 i 个分支的特征，其中 $\mathbf{f} \in R^L$，L 为 $\mathbf{f}$ 的维度。则 $\mathcal{D} = \frac{1}{N_m N_d N_b} \sum_{d=1}^{N_d} \sum_{m=1}^{N_m} \sum_{i=1}^{N_b-1} \sum_{j=i+1}^{N_b} \frac{\|\mathbf{f}_i(d,m)-\mathbf{f}_j(d,m)\|_2}{L}$，其中 $\|\cdot\|_2$ 为欧几里得范数，N_b 为基本块的分支数。$\mathcal{D}$ 越大表明分支之间特征的多样性越高。

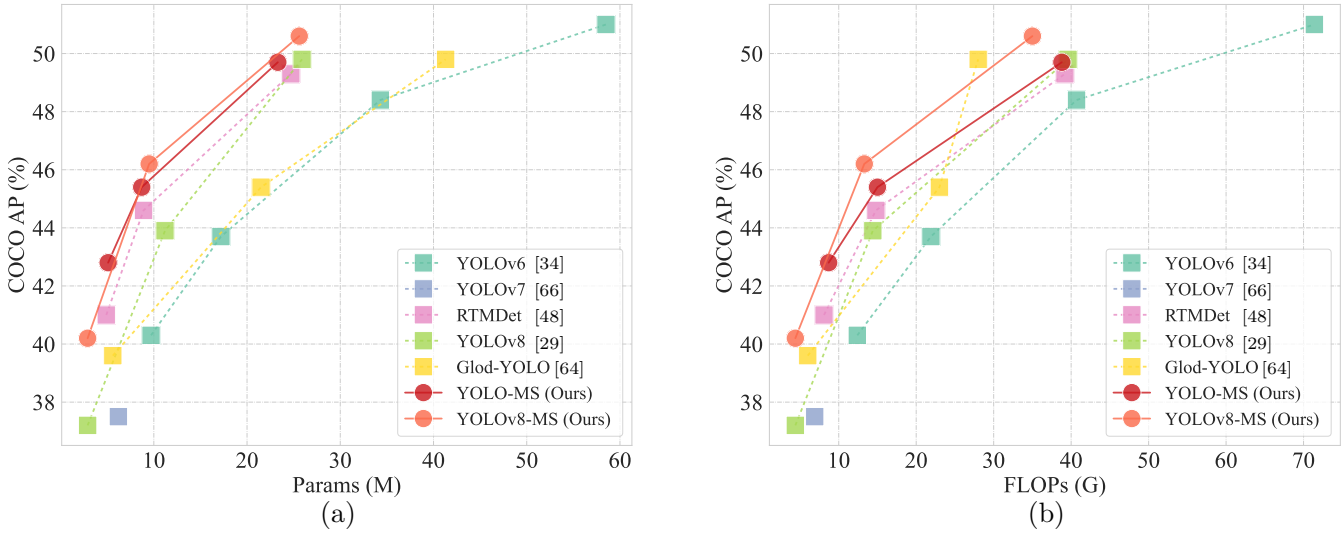


图 2. 与其他 SOTA（最先进）实时目标检测器在 MS COCO 数据集 [43] 上的比较。(a) AP（平均精度）性能 v.s. 参数量。(b) AP 性能 v.s. 计算量（MACs）。计算 MACs 时使用的输入大小为 640×640。我们的 YOLO-MS 在性能与计算量之间实现了最佳平衡。

展 [66], [68] 引入了多分支结构，其可以视为一种隐式多尺度特征学习方法。然而，图 1 中的可视化和统计分析表明，其分支间特征趋于同质化，这反映出可能缺乏多尺度信息的多样性。针对上述问题，另一种解决方案是 Res2Net [14] 的块结构。尽管 Res2Net 采用了层次化多分支结构，相较上述方法可以丰富分支间特征的多样性，但其同时也引入了与目标无关的污染性空间信息（如图 1 所示），这可能会损害检测性能。

在本文中，我们重新思考了如何在多分支构建块中编码多尺度特征表示。我们认为，对于每个构建块，除了像 [14] 那样编码多尺度特征外，动态聚合来自不同分支的不同粒度信息同样至关重要。在 FPN [41] 中，每个特征级别的检测头负责检测尺寸在一定范围内的目标，例如边界框短边介于 32 到 64 之间的目标。为此，我们设计了一种全局查询学习（**G**lobal **Q**uery **L**earning, GQL）方法，并提出了一种新型构建块，命名为 MS-Block。如图 3 所示，我们维护了一个全局查询，用于存储每个分支跨阶段的空间表示。GQL 使得每个块能够根据输入和阶段位置动态平衡各分支的影响。

此外，与以往实时目标检测器在所有块中使用同构卷积核大小的做法不同，我们提出了异构核大小选择（**H**eterogeneous **K**ernel Size **S**election, HKS）协议。该协议在不同 MS-Block 中随着网络的深入逐步增加卷积核大小。具体而言，我们在浅层使用小卷积核，以更高效地处理高分辨率特征；而在深层采用大卷积核，以捕获高级语义信息，从而更好地识别大目标。

基于上述设计原则，我们提出了一种实时目标检测器，称为 **YOLO-MS**。为了评估 YOLO-MS 的性能，我们在 MS COCO [43] 数据集上进行了全面的实验。我们还与其他 SOTA 方法进行了定量比较，以展示本方法的优越性能。如图 2 所示，YOLO-MS 在计算量与性能的平衡方面优于其他近期的实时目标检测器。具体而言，我们的 YOLO-MS-XS 在 MS COCO 上取得了 42.8% 的 AP 分数，仅需要 5.1M 可学习参数和 8.7G MACs。此外，YOLO-MS-S 和 YOLO-MS 分别达到 45.4% 和 52.1% 的 AP 分数，具有 8.7M 和 50.8M 的可学习参数，超越了基线方法 RTMDet [48]。另外，我们的工作还可以用作即插即用模块，为其他 YOLO 模型带来提升。例如，我们的方法可将 YOLOv8-n 的 AP 从 37%+ 提高到 40%+，同时减少参数量和 MACs。

2 相关工作

2.1 实时目标检测

目标检测任务旨在检测特定场景中的目标。尽管多阶段检测器 [19], [18], [56], [41], [24], [2], [70], [51], [4] 和端到端检测器 [82], [3], [50], [35], [44], [75] 取得了卓越的性能，但它们的复杂结构往往使得它们的性能无法达到实时水平，而实时性是目标检测在实际中应用的先决条件。为了在速度和精度之间达到最佳平衡，研究者们付出了大量努力以在网络架构和训练技术上开发出高效的检测器。与传统的两阶段检测器不同，大多数实时目标检测网络采用单阶段框架 [42], [61], [81], [77], [37], [36]。其中，YOLO 系列是最突出的代表。

架构设计是 YOLO 开发过程中的核心关注点，因为它是影响模型性能的关键因素。从 YOLO 家族的起

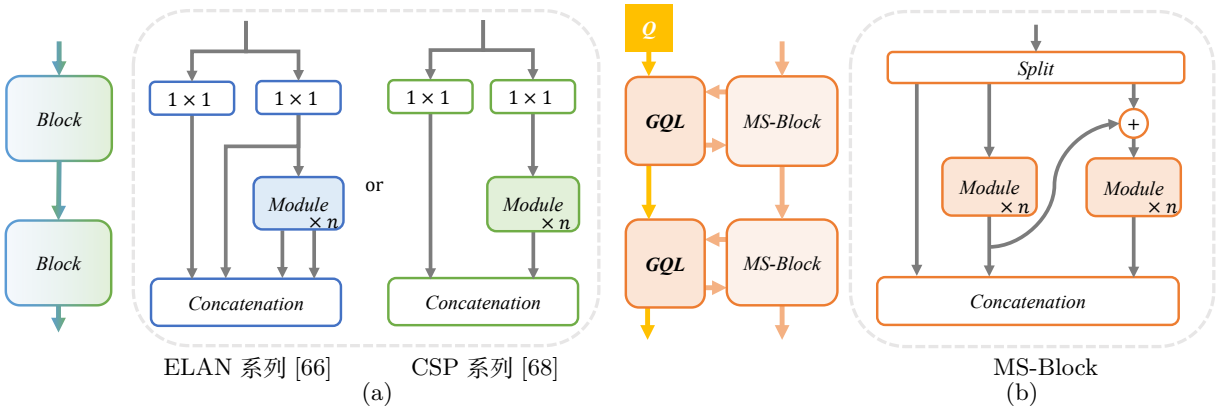


图 3. (a) 以往 YOLO 模型中广泛使用的构建块架构，例如 ELAN [68] 系列或 CSP [68] 系列。(b) 我们提出的 MS-Block 架构。其中，n 表示模块的数量（我们采用了反向瓶颈模块 [58]）。Q 表示在 GQL 中使用的全局查询。

源——YOLOv1 [53] 开始，网络架构经历了巨大的变化。YOLOv4 [1], [65] 使用跨阶段部分连接（CSPNet）[68] 改进 DarkNet 从而提升了性能，并催生了许多基于此的 YOLO 变体，例如 YOLOv5 [28]、RTMDet [48] 和 YOLOv8 [29]。具体来说，RTMDet [48] 首次将大核卷积（5×5）引入网络，以提高 CSP 块的特征提取能力。YOLOv6 [34] 和 PPYOLOE [73] 研究了重参数化技术，以在不增加推理成本的情况下获得更高的精度。YOLOv7 [66] 提出了扩展高效层聚合网络（E-ELAN），通过控制最短和最长梯度路径，该网络能够有效地学习和收敛。RT-DETR [80] 构建了第一个基于 Transformer 的模型，以规避推理过程中非极大值抑制（NMS）的负面影响。与上述方法不同的是，本文并未引入新的训练或优化技术，而是专注于通过学习更具表现力的多尺度特征表示来改进实时目标检测器。

2.2 多尺度特征的表征学习

计算机视觉中的多尺度特征表征学习已有很长的研究历史 [71], [6], [78], [15], [74], [20], [26]。强大的多尺度特征表示能力能够有效提升模型性能，这已在多个任务中得到验证 [41], [45], [25], [79], [73]，其中包括实时目标检测 [55], [1], [28], [34], [48]。

实时目标检测中的多尺度特征学习。许多实时目标检测器通过整合颈部不同特征级别的特征来提取多尺度特征 [41], [45]。例如，YOLOv3 [55] 及其后续 YOLO 系列分别引入了 FPN [41]、PAFPN [45] 和聚集-分发机制 [64]，以增强多尺度特征的融合能力。SPP（空间金字塔池化）[25] 模块则被广泛用于扩大编码器的感受野。此外，多尺度数据增强 [55] 也被广泛用作提升模型多尺度能力的有效训练技巧。然而，主流的基本构建块在关注如何提升检测效率或引入新的训练技术的同时，往往忽略了多尺度特征表征学习的重要性，尤其是 CSP [68] 块和 ELAN [66], [67] 块。相比之下，我们的方法专注于基本块的设计，并分析了分支间特征如何影响多尺度特征的表征能力。

大核卷积。我们的工作还涉及使用不同核大小的卷积，特别是大核卷积。近年来，大核卷积以深度卷积的形式重新受到了关注 [9]。大核卷积提供了更宽的感受野，这可以成为一种构建强大多尺度特征表示的有效技术 [21], [20], [26], [46], [39], [38]。在实时目标检测领域，RTMDet [48] 首次尝试在网络中引入大核卷积，但由于速度限制，其卷积核大小仅达到 5×5。在不同阶段采用同构构建块的设计限制了大核卷积的应用。本文基于一些经验发现提出了 HKS 协议，HKS 协议在不同阶段采用核大小不同的卷积层，利用大核卷积实现了速度与精度之间的良好平衡。

3 方法

作为现代目标检测器的关键方面，多尺度特征表示的学习对检测性能具有显著影响 [41], [45]。在本节中，为了构建具有稳健多尺度能力的实时目标检测器，我们提出了 MS-Block 和 HKS 协议。我们的 MS-Block 包含一个增强的分层多分支结构以提升分支间特征的多样性，并结合 GQL 提供的跨阶段引导，减少了有害的空间信息并增强了多尺度表示能力。我们引入 HKS 协议，以高效且有效地在实时目标检测器中整合大核卷积，从而提升多尺度能力。

3.1 对基本构建块中多尺度特征学习的重新思考

如第 1 节所述，在两种流行的块——CSP [68] 和 ELAN [66] 中，不同分支的特征之间存在冗余，可能会削弱多尺

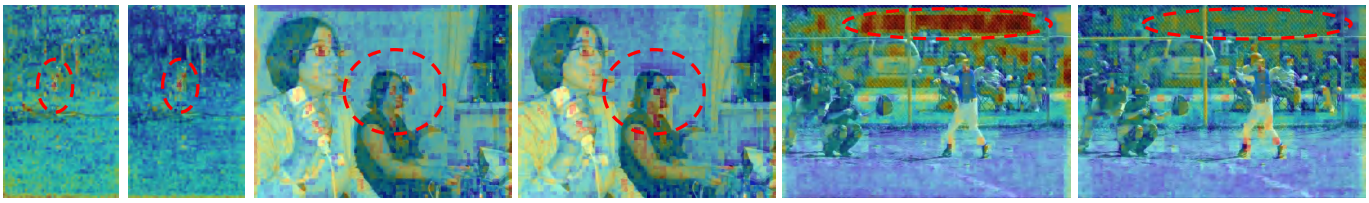


图 4. 使用和不使用 GQL 方法的特征图的可视化。我们展示了每个分支不同图像的示例，以进一步突出 GQL 在提升特征提取定位精度方面的有效性。右侧部分展示了使用 GQL 的方法。

度表示能力。为弥补这一不足，我们重新审视了以往实时目标检测器的基本块设计，并在我们的 MS-Block 中采用了分层多分支结构 [14]，而不是 CSP 和 ELAN。如图 3 所示，与 CSP 中的双分支结构和 ELAN 中的类并行多分支结构相比，分层多分支结构使基本块的每个分支都具备不同的感受野。这种分层多分支结构是 Res2Net [14] 的关键，可以增加构建块中不同尺度的信息丰富性。

图 3(b) 的右侧部分展示了 MS-Block 多尺度块的细节。它被用于检测器的主干（backbone）和颈部。设输入特征为 $\mathbf{Z} \in R^{H\times W\times C}$，其中 C、H 和 W 分别表示 $\mathbf{Z}$ 的通道数、宽度和高度。我们首先使用一个 1×1 卷积将 $\mathbf{Z}$ 变换为 $\mathbf{X} \in R^{H\times W\times NC}$。然后，我们将 $\mathbf{X}$ 划分为 N 个不同的组，记作 $\{\mathbf{X}_i\}$，其中 $i \in 1, 2, 3, ..., N$。为了在性能和速度之间取得更好的平衡，我们选取 N 为 3。除了 $\mathbf{X}_1$ 之外，每组都会通过一个带有具有反向瓶颈模块（IBM，Inverted Bottleneck Module）[58] 的分支，以获得输出 $\mathbf{Y}_i$。这与 Res2Net 中每个分支都使用标准卷积的做法不同。IBM 的另一大优点是使用了深度卷积，这使得使用大核卷积成为可能。需要注意的是，在小尺度模型中，我们采用了简化的反向瓶颈模块（SIBM），省略了最后的 1×1 层以获得更快的速度。$\mathbf{Y}_i$ 的数学表示如下：

$$\mathbf{Y}_i = \begin{cases} \mathbf{X}_i, & i = 1 \\ F_I(\mathbf{Y}_{i-1} + \mathbf{X}_i), & i > 1 \end{cases} \tag{1}$$

其中，F_I 表示 (S)IBM 模块。最后，我们连接所有分支，并应用一个 1×1 卷积来实现不同分支间的交互。每个分支均编码不同尺度的特征。当网络深入时，该 1×1 卷积还用于调整通道数。

3.2 全局查询学习

尽管在构建块中引入上述分层结构可以使目标检测器捕获丰富的多尺度特征，但它忽略了显式建模不同尺度目标的多尺度特征的重要性。Transformer 中的注意力机制已被证明是一种建立给定查询 $\mathbf{Q}$ 与键 $\mathbf{K}$ 之间成对关系的有效技术 [62], [10], [22], [23], [30]。受此启发，在本小节中，我们提出了一种名为**全局查询学习**（GQL, Global Query Learning）的高效策略，旨在为基本块提供跨阶段的指导。如图 3(b) 所示，我们维护了一个轻量且可学习的全局查询，记为 $\mathbf{Q} \in R^{N\times c^2}$，它在不同阶段间传递，将全局空间信息转换为每个块的输入，以增加分支间的多样性。而 N 和 c 分别代表分支的数量和 $\mathbf{Q}$ 的大小。它使得当前块的空间信息 $\mathbf{K}$ 能够索引权重，这一权重用于根据每个分支所在的阶段及其前一阶段的信息安排各分支的影响力。需要注意的是，我们仅在主干网络的第 2、3、4 阶段使用 GQL，以在计算成本和效果之间取得更好的平衡。形式上，如第 3.1 节所定义，给定 $\mathbf{Y} \in R^{H\times W\times NC}$，我们注意力模块的输出 $\mathbf{Y}' \in R^{H\times W\times NC}$ 可表示为

$$\mathbf{K} = \text{Linear}(\text{GAP}(\mathbf{Y})), \tag{2}$$

$$\mathbf{Y}' = S(\mathbf{Q} \times \mathbf{K}^T) \odot F_{ms}(\mathbf{Y}), \tag{3}$$

其中，S 代表 Sigmoid 函数，GAP 是全局平均池化（Global Average Pooling）操作，$\odot$ 表示逐元素乘法，Linear 代表用于提取空间信息的线性层。F_{ms} 代表多尺度块，旨在增强分支间的特征多样性，如图 3(b) 所示。使用 GAP 的全局查询 $\mathbf{Q}$ 是轻量级的，这使得我们提出的 YOLO-MS 的计算成本几乎可以忽略不计。在图 4 中，我们展示了一些案例的可视化证据，以证明 GQL 能够降低污染性空间信息的负面影响。第 4.2 节中提供了 GQL 的进一步分析。

3.3 异构核大小选择协议

除了设计构建块，我们还从宏观架构的角度深入研究了卷积的使用。尽管我们使用的构建块在多尺度能力上带来了很大提升，但它们并未充分探索不同核大小卷积的作用，尤其是大核卷积。大卷积核已被证明在基于 CNN 的视觉识别任务模型中是有效的 [26], [46], [9]，但被实时目标检测器忽视了。将大核卷积引入实时目标检测器的主要障碍是计算开销，尤其是在底层阶段。如表 1 所示，对高分辨率特征应用大核卷积计算成本较高，因此在低分辨率特征上采用大核卷积能够大大降低计算开销。

此外，以往的实时目标检测器大多在不同的编码器阶段采用同构卷积（即具有相同核大小的卷积），但我们认为这并不是提取多尺度语义信息的最佳选择。在金字塔结构中，从检测器浅层阶段提取的高分辨率特征通常用于捕获细粒度语义，以检测小目标。相反，来自更深层阶段的低分辨率特征则用于捕获高级语义，以检测大目标。如果我们在所有阶段都统一采用小核卷积，那么深层阶段的有效感受野（*ERF*）将会受限，从而影响对大目标的检测性能。在每个阶段引入大核卷积可以帮助缓解这一限制。尽管如此，由于它们具有较大的 ERF，并编码更广泛的面积，这增加了在小目标外包含污染信息的可能性。

基于上述分析，我们提出了**异构核大小选择**（HKS）协议。HKS 利用不同阶段的异构卷积来捕获更丰富的多尺度特征。具体而言，我们从较低阶段到较高阶段逐步增加核大小。由于核大小通常为奇数，我们从 3×3 开始，采用步长 2 递增核大小。根据此协议，卷积核大小从最浅到最深依次为 3、5、7、9。与以往工作 [49], [26] 不同，我们还将这些设置扩展到了 PAFPN 和头部（head）部分。此外，我们在第 4.3 节中进行了深入分析，以说明引入 HKS 协议的原因及其有效性。我们的 HKS 协议能够在不对浅层阶段产生其他任何影响的情况下，扩大深层阶段的感受野。它能够提取细粒度和粗粒度的语义信息，增强多尺度特征表示能力。不仅有助于编码更丰富的多尺度特征，而且确保了推理的高效。在实际实验中，我们经验性地发现，采用 HKS 协议的 YOLO-MS 其推理速度几乎与仅使用 3×3 卷积的情况相同。

表 1
网络中不同阶段不同核大小卷积的 FPS（$\times 10^3$）。灰色表示 YOLO-MS 在各个阶段使用的核大小。所有模型的基准计算环境相同。

阶段	输入大小	# 通道数	3×3	5×5	7×7	9×9
#1	320×320	160	**2.72**	1.38	0.93	0.65
#2	160×160	320	5.53	**2.78**	1.86	1.31
#3	80×80	640	10.46	5.52	**3.65**	2.65
#4	40×40	1280	14.25	10.73	7.21	**5.21**

3.4 架构

我们模型的主干由四个阶段组成，每个阶段后均采用一个步长为 2 的 3×3 卷积进行下采样。我们采用当前的 SOTA 实时检测器 RTMDet [48] 作为基线。在编码器中，我们使用 SiLU [11] 作为激活函数，并采用 BN [27] 进行归一化处理。与 [48] 一样，我们在第三阶段后添加了一个 SPP 块 [25]。我们借鉴 [1], [68] 的方法，使用 PAFPN 作为颈部在编码器上构建特征金字塔 [41], [45]，用于融合主干网络不同阶段提取的多尺度特征。颈部使用的基本构建块同样为我们的 MS-Block，HKS 在颈部和头部中均有类似用途。此外，为了在速度与精度之间取得更好的平衡，我们将主干的多级特征的通道数减半。我们提出了 YOLO-MS 的三种变体，分别为 YOLO-MS-XS、YOLO-MS-S 和 YOLO-MS。不同规模版本 YOLO-MS 的详细配置列于 表 2。在其他方面，我们与 [48] 保持一致。

4 实验

4.1 实验设置

实现细节。我们的实现基于 MMDetection [5] 框架和 PyTorch [52]。所有实验均在配备 8 张 NVIDIA 3090 GPU 的机器上进行，每张 GPU 的批量大小为 32 以保证公平性。具体而言，由于硬件限制，大版本模型的批量大小为每张 GPU 16。所有规模的 YOLO-MS 模型均从零开始训练 300 轮次，不依赖于其他大规模数据集（如 ImageNet [8]）或预训练权重。所有实验的输入大小均为 640×640。训练过程中，我们采用 AdamW 优化器 [31]，动量参数为 0.9，权重衰减参数为 0.05。对于偏置项和归一化参数，权重衰减参数设为 0。学习率设置包括起始因子为 1×10^{-5} 的 1000 次迭代预热，以及初始值为 1×10^{-4} 的平滑余弦退火调度。我们还引入了衰减因子为 0.9998 的指数移动平均（EMA）[32] 以提升模型性能。所有实验均从零开始训练 300 轮次。此外，我们在分类和边界框回归中分别使用 Focal 损失 [42] 和 DIoU（距离交并比）损失 [57]。标签分配基于 SimOTA [16]，该方法在匹配过程中使用包含分类成本、区域先验成本和回归成本的成本函数。更多细节见 Lyu 等人的研究 [48]。针对数据增强，我们在前 280 轮次中使用缓存 Mosaic [48] 和 MixUP [76]，在最后 20 轮次中使用 LSJ [17]。为了比较的公平，在评估过程中，我们确保后处理方法与先前的做法一致 [28], [34], [66], [48]。在后处理阶段，在执行非极大值抑制（NMS）之前，我们会过滤掉置信度低于 0.001 的边界框，并选取最高的 300 个框进行评估。

数据集。我们遵循大多数先前工作的标准做法，在广泛使用的 MS COCO [43] 基准上评估了所提出的检测器。具体而言，我们使用包含 $115K$ 张图像的 *train2017* 集进行训练，并使用包含 $5K$ 张图像的 *val2017* 集进行验证。评估采用标准的 COCO 评价指标，即平均精度

表 2

YOLO-MS 的简要配置。“模块类型”表示基本块中的模块类型。“放大因子”表示用于缩放通道维度的系数。基本通道设置为 $\{32, 64, 128, 512, 256\}$。“模块数量”是编码器中基本块的数量。YOLO-MS 三个分支中的每一个都包含两个 MS-Block。“通道扩展比率”表示 MS-Block 的 IBM 模块内通道扩展过程的比率。c 代表 IBM 输入的通道维度。基于 RTMDet，我们实现了参数量分别为 4.54M、8.13M 和 22.17M 的三个变体。

模型	模块类型	放大因子	模块名称	通道扩展比率	延迟	参数量	MACs
RTMDet-XS	CSPNext 模块	0.375	$1+1+1+1$	-	6.5ms	4.9M	8.1G
YOLO-MS-XS	SIBM + SIBM	1.050	$2+2+2+2$	$c \to 2c$	7.1ms	5.1M	8.7G
RTMDet-S	CSPNext 模块	0.500	$1+2+2+1$	-	7.3ms	8.9M	14.8G
YOLO-MS-S	SIBM + SIBM	1.375	$2+2+2+2$	$c \to 2c$	7.3ms	8.7M	15.0G
RTMDet-M	CSPNext 模块	0.750	$2+4+4+2$	-	10.0ms	24.7M	39.3G
YOLO-MS	SIBM + IBM	2.175	$2+2+2+2$	$c \to 2c$	10.5ms	23.3M	38.8G

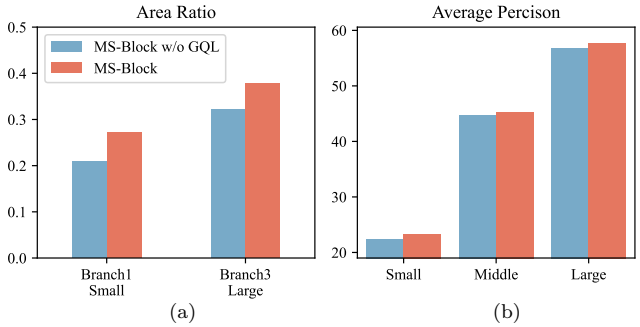


图 5. (a) 对于分别对应小目标和大目标的分支，其在 GT（Ground Truth，真实标注）框内高特征激活值（>0.5）的面积占比。(b) 不同尺度目标的平均精度（AP）对比。**红色表示使用 GQL 的方法。**

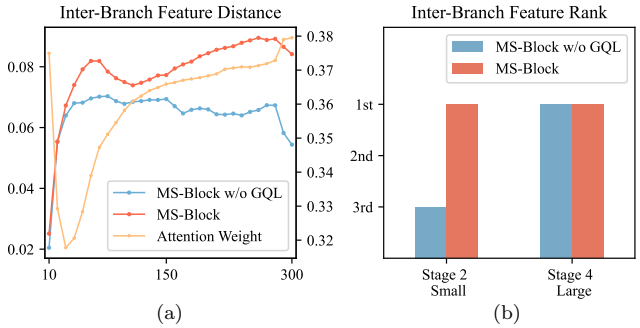


图 6. (a) 不同阶段中，目标分支与其他分支之间的平均特征距离的趋势。(b) 对于分别对应小目标和大目标的阶段，其目标分支特征分布的排名。**红色表示使用 GQL 的方法。黄线表示目标分支注意力权重的变化趋势。**

（AP）作为主要的挑战指标。此外，我们还报告了在 IoU 阈值 0.5 和 0.75 之间的 mAP，以及针对小型、中型和大型目标的 AP 作为支持指标。

基准设置。遵循先前的工作，我们使用一张 NVIDIA 3090 GPU 以全精度浮点格式（FP32）测量所有模型的每秒帧数（FPS）。测试过程中，我们执行的推理不包含 NMS 后处理步骤。用于推理过程的批量大小设为 1。此外，MACs 计算基于 640×640 的输入大小，使用 MMDetection 框架 [5] 进行计算。

4.2 GQL 的分析

为了验证我们 GQL 的有效性，我们在本小节中进行了一系列分析。在图 5(a) 中，我们针对第 2 阶段中感受野最小的第一个分支和第 4 阶段中感受野最大的最后一个分支，分别计算了在 GT 框内小型和大型目标高激活特征（值 > 0.5）的面积比。图 5(a) 显示，在那些与提取特征图时的阶段和分支对应尺度的目标的区域内，特征图具有更强的激活值。这意味着 GQL 能够从特征学习的角度使模型更准确地定位不同尺度的目标。在图 5(b) 中，我们展示了不同目标尺度上的平均精度。它直观地表明，我们的 GQL 在多尺度性能上带来了明显的提升。

此外，在图 6(a) 中，我们可视化了训练过程中 GQL 的分支间特征距离以及注意力权重的变化趋势。为简单起见，我们使用目标分支与其他分支之间的平均 L_1 距离来展示该变化。如第 3.2 节中所定义，GQL 中的每个值分别表示 MS-Block 内每个分支的权重。为了方便，我们仅可视化目标分支对应权重值的变化趋势，该值代表目标分支在该块中的影响力。目标分支的选择基于其感受野和所在阶段，例如，在第 2 阶段左侧分支为目标分支，对应于小型目标。我们取所有阶段的平均结果进行比较。还可以明显看出，目标分支的注意力权重与特征距离的变化趋势一致，这进一步证明了 GQL 的有效性。在图 6(b) 中，我们比较了不同阶段各分支之间特征图的平均激活值，并得到了排名值。使用 GQL 时，感受野较小的左侧分支在浅层阶段表现出较高的特征激活值，反之亦然。这表明 GQL 能够自适应地调整各分支在相应阶段的效果，以最大化其影响力。

4.3 HKS 协议的分析

先前的研究 [47], [9] 引入了 ERF（有效感受野）的概念，作为理解深度卷积神经网络（CNNs）行为的度量。ERF 衡量输入空间中受特征表示影响的有效面积。在本小节

中，我们进一步利用 ERF 的概念来研究 HKS 的有效性。我们遵循 RepLKNet [9] 中提出的方法，通过聚合贡献分数矩阵来衡量 ERF，记为 $\mathbf{A} \in R^{H\times W}$。假设输入为 $\mathbf{I} \in R^{H\times W\times C}$，输出特征图为 $\mathbf{F} \in R^{H'\times W'\times C'}$，那么 $\mathbf{A}$ 的计算过程可以数学地描述为 [9]：

$$\mathbf{P} = \max\left(\sum_{c}^{C'} \frac{\partial \mathbf{F}(\frac{H'}{2}, \frac{W'}{2}, c)}{\partial \mathbf{I}}, 0\right), \tag{4}$$

$$\mathbf{A} = \log_{10}\left(\sum_{c}^{C} \mathbf{P}(:,:,c) + 1\right). \tag{5}$$

首先，我们计算编码器中第 2、3、4 阶段的 $\mathbf{A}$。随后，将每个阶段的 $\mathbf{A}$ 归一化到 $[0,1]$。假设存在一个阈值 θ，值高于 θ 的面积比例，即高贡献面积，可以表示为 $h(\theta)$：

$$h(\theta) = \frac{1}{|\mathbf{A}|} \sum_{a\in\mathbf{A}} \mathbb{1}[a > \theta], \tag{6}$$

其中，$\mathbb{1}$ 代表指示函数。为了直观地展示各模型之间的差异，我们在 θ 取值从 0.5 到 0.9（步长为 0.05）的情况下计算 h，并取其平均值 $\bar{h}$ 作为 ERF 的度量标准。

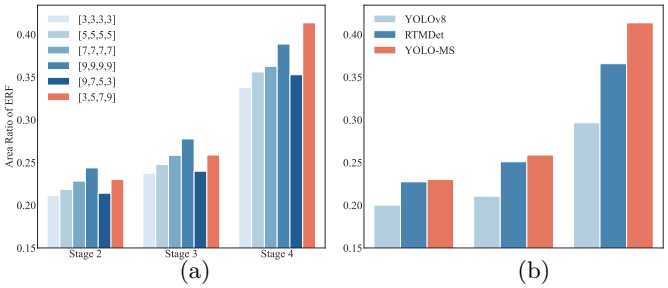


图 7. 有效感受野的统计分析。(a) 不同核大小设置的比较。(b) 不同实时检测器的比较。k_i 表示第 i 阶段的核大小。红色表示使用 HKS 的方法。

视觉比较结果如图 7 所示。为了简化表示，我们采用格式 $[k_1, k_2, k_3, k_4]$，其中 k_i 代表第 i 阶段的核大小。如图 7(a) 所示，随着核大小的增加，ERF 的面积在所有阶段也随之增大，这支持了核大小与感受野之间的正相关性。此外，在浅层阶段，ERF 的面积比大多数其他设置都要小，而在深层阶段则相反。这一观察结果表明，该协议能在不影响浅层阶段的情况下，有效扩大深层阶段的感受野。在图 7(b) 中，我们可以观察到 HKS 在深层阶段达到了最佳的 ERF，这使得我们能够更好地检测大目标。

4.4 消融实验

所提方法的消融实验。为了研究我们所提方法的影响，我们通过在基线模型 RTMDet [48] 上逐步增加组件直到变成 YOLO-MS 进行了消融实验。结果报告在表 3 中。值得注意的是，所提出的使用 GQL 和 HKS 的 MS-Block，使模型在各方面都得到了提升。具体而言，与 RTMDet 相比，我们的方法使 AP 显著提高了 +1.8%。

表 3
对所提方法的消融实验。所有模型均为 tiny 版本，并从零开始训练。基线为不带预训练权重的 RTMDet-T。所有模型均在相同的计算环境下进行基准测试。

MS-Block	+GQL	+HKS	AP	AP_s	AP_m	AP_l	FPS	参数量	MACs
RTMDet-T			40.3	20.9	44.8	57.4	154	4.9M	8.1G
Res2Net			40.0	21.3	44.8	55.2	170	4.5M	8.2G
✓			41.0	22.4	45.2	56.9	159	4.2M	8.6G
		✓	41.3	22.7	45.3	57.4	158	4.2M	8.6G
✓	✓		41.5	23.3	45.7	57.7	150	4.2M	8.6G
✓	✓	✓	42.8	23.1	46.8	60.1	141	5.1M	8.7G

表 4
对 MS-Block 分支数量的消融实验。N_b 代表 MS-Block 中的分支数量。基线为不使用 HKS 协议的 YOLO-MS-XS。所有模型均在相同的计算环境下进行基准测试。

N_b	AP	AP_s	AP_m	AP_l	FPS	参数量	MACs
2	39.0	20.9	43.5	54.2	184	3.5M	7.6G
3	41.5	23.3	45.7	57.7	150	4.2M	8.6G
4	43.5	24.0	47.3	60.3	137	5.1M	9.7G

分支数量的消融实验。我们的 MS-Block 通过多个分支对输入特征进行划分和传播。然而，增加分支数量还会增加 IBM，并减少每个分支中的通道数量。为了研究分支数量（记为 N_b）的影响，这里我们进行了消融实验，结果详见表 4。为了取得性能与速度间的更好平衡，我们在所有后续实验中默认设置 $N_b = 3$。如果速度不是限制因素，也推荐使用 $N_b = 4$。

表 5
全局查询空间维度的消融实验。D_s 代表 GQL 中全局共享查询的空间维度。基线为不使用 HKS 协议且从零开始训练的 YOLO-MS-XS。所有模型均在相同的计算环境下进行基准测试。

D_s	AP	AP_s	AP_m	AP_l	FPS	参数量	MACs
2^2	41.0	22.2	45.3	57.0	153	4.2M	8.6G
3^2	41.2	22.4	45.5	57.4	152	4.2M	8.6G
4^2	41.5	23.3	45.7	57.7	150	4.2M	8.6G
5^2	41.4	22.2	45.6	57.6	145	4.2M	8.6G
6^2	41.4	22.7	46.0	57.4	142	4.2M	8.6G

查询空间维度的消融实验我们的 GQL 维护了一个全局查询，从而为自适应增强分支特征提供跨阶段信息。我们进行了消融实验，以研究查询空间维度的影响。值得注意的是，直接增加分支的尺寸并不总能带来性能提升。结果总结在表 5 中。具体而言，当 $D_s = 4$ 时，YOLO-MS 达到了 41.5% AP 的最佳性能。考虑到性能与速度

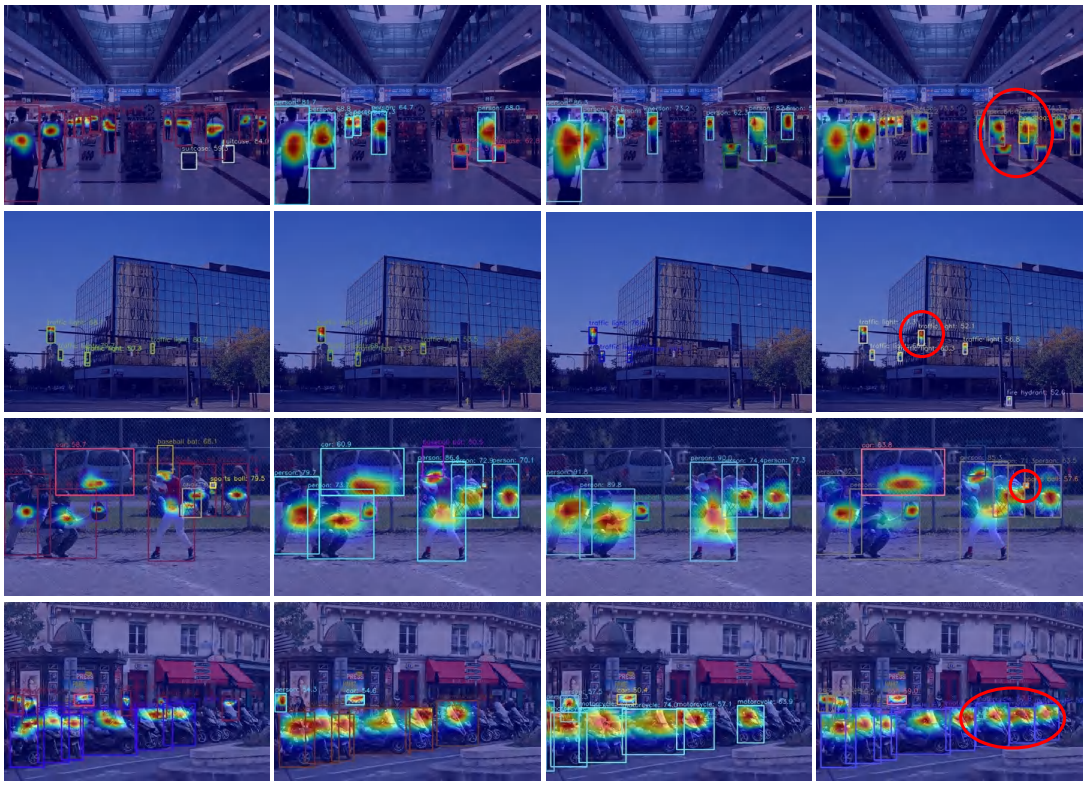


图 8. 通过 Grad-CAM [59] 与 SOTA 模型进行视觉对比。我们的方法能够更好地定位不同尺度的目标。

的权衡，我们在所有后续实验中选择 4^2 为全局查询的空间维度。

表 6
对 HKS 不同核大小设置的比较。k_i 表示第 i 阶段的核大小。“Neck-HKS”和“Head-HKS”分别表示在 PAFPN 和 Head 模块中使用 HKS 协议。基线为 YOLO-MS-XS。所有模型均从零开始训练且在相同的计算环境下进行基准测试。

$[k_1, k_2, k_3, k_4]$	AP	AP_s	AP_m	AP_l	参数量	MACs
$[3, 3, 3, 3]$	41.0	22.4	45.2	56.9	4.2M	8.6G
$[5, 5, 5, 5]$	41.7	22.7	46.2	57.7	4.3M	8.7G
$[7, 7, 7, 7]$	41.8	23.4	46.2	58.4	4.3M	8.9G
$[9, 9, 9, 9]$	41.8	22.3	46.4	57.8	4.4M	9.1G
$[11, 11, 11, 11]$	41.9	22.7	46.8	57.7	4.5M	9.6G
$[5, 7, 9, 11]$	41.9	22.7	46.8	57.7	4.5M	9.6G
$[3, 7, 11, 15]$	41.9	22.7	46.8	57.7	4.5M	9.6G
$[9, 7, 5, 3]$	41.2	22.0	45.2	58.2	4.3M	9.0G
$[3, 5, 7, 9]$	41.9	22.6	46.2	58.9	4.2M	8.6G
+ Neck-HKS	42.3	22.0	46.1	59.8	4.4M	8.7G
+ Neck-HKS + Head-HKS	42.8	23.1	46.8	60.1	5.1M	8.7G

不同核设置的消融实验。我们使用不同的核大小设置进行定量比较，以评估 HKS 的有效性。我们探索了 3、5、7、9 和 11 的同构核大小设置以及 HKS 的倒置版本，即 $[9, 7, 5, 3]$。如表 6 所示，简单地增大核大小可以提升性能，但会带来更高的计算成本，从而影响推理效率。此外，卷积核在不同阶段的排列顺序也起着关键作用。具体而言，当在浅层阶段使用大核卷积、在深层阶段使用小核卷积时，相比于 HKS，性能下降了 0.7% AP。这一结果表明，与浅层阶段不同，深层阶段需要更大的感受野来有效捕获粗粒度信息。考虑到计算成本，我们的 HKS 由于计算开销最小而脱颖而出。这表明，通过有策略地在合适位置放置不同大小的卷积，我们可以最大限度地提高这些卷积的利用效率。

表 7
和不同核大小设置的比较。“我们的模型”指 YOLO-MS-XS。“TTA”指的是测试时增强。

模型	分辨率	AP	AP_s	AP_m	AP_l
RTMDet-tiny	320 × 320	30.0	8.3	36.0	54.0
我们的模型		32.7	10.5	35.3	56.8
RTMDet-tiny	640 × 640	41.0	20.7	45.3	58.0
我们的模型		42.8	23.1	46.8	60.1
RTMDet-tiny	1280 × 1280	35.2	29.2	45.8	36.4
我们的模型		37.9	31.0	46.7	37.5
RTMDet-tiny	TTA	41.9	28.1	47.1	55.5
我们的模型		45.2	31.0	48.4	60.6

图像分辨率分析。这里我们通过实验研究了图像分辨率与多尺度构建块设计之间的相关性。在推理过程中，我们采用测试时增强（Test Time Augmentation, TTA）对图

表 8
与 SOTA 实时目标检测器的比较。“†”表示使用预训练模型进行训练。所有模型均在相同的计算环境下进行基线测试。评估 ATSS、GFocalv2 和 TOOD 时的输入大小为 1333×800。评估其他检测器时则为 640×640。所有模型的 MACs 计算均基于 640×640 的输入大小。延迟计算时不包括 NMS。YOLOv10 的推理采用一对一训练。其他检测器的性能结果参考自 MMDetection [5] 和官方仓库。需要注意的是，EMA（指数移动平均）作为一种常见的模型优化技术，被用于所有 YOLO 模型和 RT-DETR 的训练。

模型	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	延迟	参数量	MACs
ATSS-R50 † [77]	39.4	57.6	42.8	23.6	42.9	50.3	-	32.3M	82.0G
GFocalv2-R50 † [36]	40.2	58.4	43.3	23.3	44.0	52.2	-	32.4M	83.3G
TOOD-R50 † [12]	42.4	59.7	46.2	25.4	45.5	55.7	-	32.2M	80.4G
YOLOv5-N [28]	28.0	45.9	29.4	14.0	31.8	36.6	4.8ms	1.9M	2.3G
YOLOv5-S [28]	37.7	57.1	41.0	21.7	42.5	48.8	5.2ms	7.2M	8.3G
YOLOv5-M [28]	45.3	64.1	49.4	28.4	50.8	57.7	7.1ms	21.2M	24.5G
YOLOv5-L [28]	48.8	67.4	53.3	33.5	54.0	61.8	9.4ms	46.6M	54.6G
YOLOv6-N [34]	36.2	51.6	39.2	16.8	40.2	52.6	7.9ms	4.3M	5.5G
YOLOv6-T [34]	40.3	57.4	43.9	21.2	45.6	57.5	8.0ms	9.7M	12.4G
YOLOv6-S [34]	43.7	60.8	47.0	23.6	48.7	59.8	8.5ms	17.2M	21.9G
YOLOv6-M [34]	48.4	65.7	52.7	30.0	54.1	64.5	14.2ms	34.3M	40.7G
YOLOv6-L [34]	51.0	68.4	55.2	33.5	56.2	67.3	14.3ms	58.5M	71.4G
YOLOv7-T [66]	37.5	55.8	40.2	19.9	41.1	50.8	4.9ms	6.2M	6.9G
YOLOv7-L [66]	50.9	69.3	55.3	34.7	55.1	66.6	10.4ms	36.9M	52.4G
RTMDet-T† [48]	41.0	57.4	44.4	20.7	45.3	58.0	6.5ms	4.9M	8.1G
RTMDet-S† [48]	44.6	61.7	48.3	24.2	49.2	61.8	7.3ms	8.9M	14.8G
RTMDet-M [48]	49.3	66.9	53.9	30.5	53.6	66.1	10.0ms	24.7M	39.3G
Gold-YOLO-N [64]	39.6	55.7	–	19.7	44.1	57.0	-	5.6M	6.0G
Gold-YOLO-S [64]	45.4	62.5	–	25.3	50.2	62.6	-	21.5M	23.0G
Gold-YOLO-M [64]	49.8	67.0	–	32.3	55.3	66.3	-	41.3M	43.0G
YOLOv8-N [29]	37.2	52.7	40.3	18.9	40.5	52.5	5.3ms	3.2M	4.4G
YOLOv8-S [29]	43.9	60.8	47.6	25.3	48.7	59.5	5.8ms	11.2M	14.4G
YOLOv8-M [29]	49.8	66.9	54.2	32.6	54.9	65.9	8.3ms	25.9M	39.6G
YOLOv9-T [69]	37.5	52.3	40.6	18.4	41.7	52.9	10.3ms	2.1M	4.1G
YOLOv9-S [69]	46.8	63.4	50.7	26.6	56.0	64.5	10.5ms	7.1M	13.4G
YOLOv10-N [63]	38.5	53.8	41.7	19.0	42.3	54.7	6.3ms	2.8M	4.3G
YOLOv10-S [63]	46.2	62.9	50.2	26.8	51.0	63.6	6.9ms	8.1M	12.4G
YOLOv10-M [63]	51.0	68.0	55.7	33.7	56.3	66.9	8.8ms	16.5M	31.5G
RT-DETR-R18 † [80]	46.5	63.8	50.4	28.4	49.8	63.0	9.5ms	20.0M	30.0G
RT-DETR-R34 † [80]	48.9	66.8	52.8	30.9	52.3	66.3	12.3ms	31.0M	45.1G
YOLO-MS-XS	42.8	60.0	46.7	23.1	46.8	60.1	7.1ms	5.1M	8.7G
YOLO-MS-S	45.4	62.8	49.5	25.9	49.6	62.4	7.3ms	8.7M	15.0G
YOLO-MS	49.7	67.2	54.0	32.8	53.8	65.6	10.5ms	23.3M	38.8G
YOLOv8-MS-N	40.2	56.5	43.3	20.9	44.1	55.5	6.5ms	2.9M	4.4G
YOLOv8-MS-S	46.2	63.3	50.1	27.0	51.0	62.7	6.9ms	9.5M	13.3G
YOLOv8-MS-M	50.6	67.8	55.1	33.6	55.8	65.7	9.3ms	25.9M	35.2G
YOLOv9-MS-T	38.5	53.7	41.9	19.3	42.8	52.8	6.1ms	2.0M	4.2G
YOLOv10-MS-S	46.8	63.6	51.1	27.7	51.3	62.5	7.7ms	7.1M	11.5G

像进行多尺度变换（320×320、640×640 和 1280×1280）。此外，我们还单独使用这些分辨率进行评估。需要注意的是，训练时使用的图像分辨率为 640×640。结果见表 7。实验结果呈现出一致的趋势：随着图像分辨率的增加，AP_s（小目标的 AP）也随之提升。然而，在低分辨率图像上，我们可以获得更高的 AP_l（大目标的 AP）。这也验证了 HKS 协议的有效性。

4.5 和 SOTA 的比较

可视化比较。为了评估图像的哪些部分吸引了检测器的注意力，我们使用 Grad-CAM [59] 生成类别响应图。我

表 9
应用于其他 YOLO 版本。"†" 表示使用预训练模型进行训练。所有模型均在相同的计算环境下进行基准测试。YOLOv10 的推理采用一对一训练方式。

模型	AP	AP_s	AP_l	延迟	参数量	MACs
RTMDet-T† [48]	41.0	20.7	58.0	6.5ms	4.9M	8.1G
YOLO-MS-XS	42.8	23.1	60.1	7.1ms	5.1M	8.7G
RTMDet-S† [48]	44.6	24.2	61.8	7.3ms	8.9M	14.8G
YOLO-MS-S	45.4	25.9	62.4	7.3ms	8.7M	15.0G
RTMDet-M [48]	49.3	30.5	66.1	10.0ms	24.7M	39.3G
YOLO-MS	49.7	32.8	65.6	10.5ms	23.3M	38.8G
YOLOv8-N [29]	37.2	18.9	52.5	5.3ms	3.2M	4.4G
YOLOv8-MS-N	40.2	20.9	55.5	6.5ms	2.9M	4.4G
YOLOv8-S [29]	43.9	25.3	59.5	5.8ms	11.2M	14.4G
YOLOv8-MS-S	46.2	27.0	62.7	6.9ms	9.5M	13.3G
YOLOv8-M [29]	49.8	32.6	65.9	8.3ms	25.9M	39.6G
YOLOv8-MS-M	50.6	33.6	65.7	9.3ms	25.9M	35.2G
YOLOv9-T [69]	37.2	18.4	52.9	10.3ms	2.1M	4.1G
YOLOv9-MS-T	38.5	19.3	52.8	6.1ms	2.0M	4.2G
YOLOv10-S [63]	46.2	26.8	63.6	6.9ms	8.1M	12.4G
YOLOv10-MS-S	46.8	27.7	62.5	7.7ms	7.1M	11.5G

们对从 YOLOV7-T [34]、RTMDet-T [48]、YOLOV8-N [66] 和 YOLO-MS-XS 颈部生成的类别响应图进行了可视化。我们还从 MS COCO 数据集 [43] 中选择了不同大小的典型图像，包括小型、中型和大型目标。可视化结果如图 8 所示。YOLOV7-T、RTMDet-T 和 YOLOV8-N 无法有效检测小型、密集的目标，例如摩托车群和人群，并且会忽略部分目标。相反，YOLO-MS-XS 对类别响应图中的所有目标均表现出较强的响应，表明其具有卓越的多尺度特征表示能力。此外，这进一步证明，我们的检测器在不同尺度目标和包含不同密度目标的图像上均具有优异的检测性能。

定量比较。在本节中，我们将 YOLO-MS 与当前最先进的目标检测器进行对比，并将结果展示在表 8 中。很明显，YOLO-MS 在速度与精度之间实现了出色的平衡。YOLO-MS-XS 达到了 42.8% AP，与第二名的小型检测器 RTMDet [48] 相比，在 RTMDet 了使用 ImageNet [8] 预训练模型的情况下提高了 1.8% AP。YOLO-MS-S 取得了 45.4% AP，相较于 RTMDet 提高了 0.8% AP，同时推理速度更快。此外，YOLO-MS 的检测性能为 49.7% AP，在参数量和计算复杂度相近的情况下优于基线模型。综上所述，YOLO-MS 证明了其能作为有潜力的实时目标检测基线，提供强大的多尺度特征表示能力。

在其他 YOLO 上的应用。我们提出的方法可以作为即插即用模块用于其他 YOLO 模型。为了证明我们方法的泛化能力，我们将其应用于 RTMDet、YOLOv8 [29]、YOLOv9 [69] 和 YOLOv10 [63]。MS COCO 上的结果列在表 9 中。在使用我们的方法后，所有尺度基线模型的 AP 得分均有所提升，同时参数量和 MACs 更少。具体而言，我们的方法将 YOLOv8-N、YOLOv8-S 和 YOLOv8-M 的 AP 分别从 37.2%、43.9% 和 49.8% 提高至 40.2%、46.2% 和 50.6%。此外，我们的方法提升了不同尺度目标的 AP，这表明其在增强多尺度能力上的有效性。

4.6 在其他任务上的应用

在本小节中，我们将 YOLO-MS 扩展至目标检测的三种典型子任务：实例分割、任意方向目标检测和拥挤场景目标检测。我们分别使用了 COCO [43]、DOTA-v1.0 [72] 和 CrowdHuman [60] 数据集。结果如表 10、表 11 和表 12 所示。结果表明，我们的 YOLO-MS 表现优于强基线模型，证明了 YOLO-MS 在不同应用环境下的稳健性。

实例分割。实例分割是目标检测的扩展任务，旨在在像素水平上勾画出图像中的每个实例。我们将在 YOLO-MS 应用于 MS COCO [43] 上的实例分割任务中，结果如表 10 所示。在相同的训练设置下，YOLO-MS 取得了显著提升，超越了基线模型。具体而言，我们的 YOLO-MS 的分割 AP 从 40.5% 提升至 42.8%，提高了 +2.3%。

表 10
YOLO-MS 在实例分割任务上的定量结果。结果报告基于 MS-COCO [43] 验证集。"(LB)" 代表 [28] 提出的 LetterBox 大小变换。† 表示使用预训练模型的模型。提出的 YOLO-MS 的结果用灰色标注。最佳结果用**加粗**标注。

模型	输入大小	边界框 AP	分割 AP	分割 AP_s	参数量	MACs
YOLOv5-N	640(LB)	27.6	23.4	-	2.0M	3.6G
YOLOv5-S	640(LB)	37.6	31.7	-	7.6M	13.2G
RTMDet-T†	640 × 640	40.5	35.4	13.1	5.6M	11.8G
YOLO-MS-XS	640 × 640	**42.3**	**36.6**	**15.6**	5.1M	12.9G

任意方向目标检测。任意方向目标检测旨在检测任意方向的目标。我们在 DOTA-v1.0 [72] 集上比较了 YOLO-MS 和基线模型，结果如表 11 所示。在与 RTMDet-R 相同的训练设置下，我们的 YOLO-MS-R 在 tiny 规模版本上超越了基线，且在 small 规模版本上达到了与基线相同的性能。此外，在其他规模版本上，YOLO-MS-R 也相较基线有显著提升。考虑到遥感场景中的目标通常较小，在 DOTA v1.0 上较 RTMDet-R 的进步表明 YOLO-MS-R 增强了多尺度能力。

拥挤场景的目标检测。在拥挤场景中检测目标也是计算机视觉中的一项重要课题。我们评估了 YOLO-MS 在

表 11
YOLO-MS 在任意方向目标检测上的定量结果。结果报告基于 DOTA-v1.0 [72] 验证集。† 表示使用预训练模型的模型。所提出的 YOLO-MS 的结果用灰色标记。最优结果用黑体标记。评估和计算 MACs 的输入大小为 1024 × 1024。

模型	AP	AP_{50}	AP_{75}	参数量	MACs
RTMDet-R-T	58.3	85.2	66.1	4.9M	21G
RTMDet-R-S	62.0	88.1	70.6	8.9M	39G
RTMDet-R-M	64.4	88.9	74.9	24.7M	100G
RTMDet-R-L	66.3	89.4	76.9	52.3M	205G
YOLO-MS-R-XS	61.8	88.0	70.3	4.4M	22G
YOLO-MS-R-S	63.8	88.7	73.6	7.4M	38G
YOLO-MS-R	66.9	89.9	77.8	20.0M	99G
YOLO-MS-R-L	**68.6**	**90.6**	**80.7**	42.7M	206G

CrowdHuman [60] 数据集上的性能，结果如表 12 所示。与先前的工作一致，我们使用平均精度（AP）、mMR 和 JI 进行评估。mMR 表示每幅图像假阳性结果的平均对数漏报率，范围从 10^{-2} 到 1。mMR 对假阳性率敏感，值越低表明性能越好。另一方面，Jaccard 指数（JI，Jaccard Index）衡量了预测结果与 GT 的重叠程度，通常用于衡量检测器在拥挤目标检测场景中的计数能力，值越高意味着性能更越优。在与基线相同的训练设置下，YOLO-MS 取得了显著提升，AP 提高了 1.2%。考虑到 CrowdHuman 数据集的目标密度高且挑战性大，这些改进进一步证明了我们的 YOLO-MS 在处理拥挤场景方面性能优越。

表 12
YOLO-MS 在拥挤场景的定量结果。结果报告基于 CrowdHuman [60] 验证集。† 表示使用预训练模型的模型。所提出的 YOLO-MS 的结果用灰色标记。最优结果用黑体标记。评估和计算 MACs 的输入大小为 640 × 640。

模型	AP	mMR ↓	JI	参数量	MACs
RTMDet-T†	85.8	47.2	76.0	4.9M	8.0G
YOLO-MS-XS	**87.0**	**46.3**	**78.1**	5.1M	8.6G

水下场景目标检测。RUDO [13] 数据集包含 14,000 张高分辨率水下图像、74,903 个标注目标以及 10 个常见水生类别。如表 13 所示，我们的 YOLO-MS 取得了 SOTA 性能，并在相同训练设置下相较基线提升了 0.3% 的 AP。考虑到 RUDO 数据集中水下场景富有挑战性，这些改进突显了 YOLO-MS 在处理不同条件时的优越性能。

雾天场景的目标检测。RTTS [33] 数据集包含 4,322 张雾天图像，分为 5 类：自行车、公交车、汽车、摩托车和人。如表 14 所示，我们的 YOLO-MS 达到了 SOTA 性能，相较于在相同训练策略下的基线，AP 提高了 0.5%。考虑到 RTTS 数据集中雾天场景具有挑战性，这些提升进一步表明我们的 YOLO-MS 能够在处理不同的天气条件方面表现出色。

表 13
YOLO-MS 在水下场景的定量结果。结果报告基于 RUDO [13] 验证集。† 表示使用预训练模型的模型。所提出的 YOLO-MS 的结果用灰色标记。最优结果用黑体标记。计算 MACs 的输入大小为 640 × 640。

模型	输入大小	AP	参数量	MACs
Cascade RCNN-R50 [2]	1333 × 800	55.3	77.3M	1709G
ATSS-R50 [77]	1333 × 800	55.7	32.3M	82.0G
TOOD-R50 [12]	1333 × 800	57.4	32.2M	80.4G
RTMDet-T†	640 × 640	63.6	4.9M	8.1G
YOLO-MS-XS	640 × 640	**63.9**	5.1M	8.6G

表 14
YOLO-MS 在雾天场景的定量结果。结果报告基于 RTTS [33] 验证集。† 表示使用预训练模型的模型。所提出的 YOLO-MS 的结果用灰色标记。最优结果用黑体标记。计算 MACs 的输入大小为 640 × 640。

模型	输入大小	AP	参数量	MACs
Cascade RCNN-R50 [2]	1333 × 800	50.8	77.3M	1709G
ATSS-R50 [77]	1333 × 800	48.2	32.3M	82.0G
TOOD-R50 [12]	1333 × 800	50.8	32.2M	80.4G
RTMDet-T†	640 × 640	59.2	4.9M	8.1G
YOLO-MS-XS	640 × 640	**59.7**	5.1M	8.6G

不同条件下的 RAW 图像目标检测。AODRaw [40] 数据集包含 7,785 张高分辨率真实 RAW 图像，其中标注了 135,601 个实例跨越 62 个类别。它包含在九种不同光照和天气条件下拍摄的各种室内外场景。如表 14 所示，我们的 YOLO-MS 达到了 SOTA 性能，相较于在相同训练策略下的基线，AP 有 1.1% 的显著提升。考虑到 AODRaw 数据集的多样化场景及 RAW 图像输入，这些提升进一步表明，我们的 YOLO-MS 不仅适用于 RGB 图像，还能在处理不同的室内外天气条件方面表现出色。

5 结论与展望

本文提出了一种高性能且计算成本合理的实时目标检测器。为实现这一目标，我们研究了特征分布和不同核大小卷积的影响，并构建了一个能够强大提取多尺度特征表示的编码器。实验研究表明，我们所提出的 MS-Block 结合 GQL 和 HKS 协议，显著提升了检测器的速度-精度权衡性能，超越了其他实时检测器。我们希望本研究能够为目标检测领域带来新的见解。

参考文献

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. *arXiv:2004.10934*.

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 6154–6162, June 2018.

表 15

YOLO-MS 在不同条件下 RAW 图像上的定量结果。结果报告基于 AODRaw [40] 验证集。† 表示使用预训练模型的模型。所提出的 YOLO-MS 的结果用**灰色**标记。最优结果用**黑体**标记。评估的输入大小为 1280×1280，而计算 MACs 的输入大小为 640×640。

模型	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	参数量	MACs
YOLOX-T [16]	16.4	32.1	14.9	6.8	23.2	29.4	5.1M	7.6G
YOLOv6-N [34]	18.0	30.0	18.0	7.6	24.4	32.8	4.3M	5.5G
YOLOv8-N [29]	18.9	32.0	18.8	8.9	26.5	33.2	3.0M	4.4G
RTMDet-T† [48]	24.3	40.0	24.7	11.5	34.0	40.6	4.9M	8.1G
YOLO-MS-XS	**25.4**	**40.6**	**26.1**	**11.8**	**34.4**	**42.7**	5.1M	8.7G

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Eur. Conf. Comput. Vis.*, page 213–229, November 2020.

[4] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4969–4978, June 2019.

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark, 2019. *arXiv:1906.07155*.

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. *arXiv:1706.05587*.

[7] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, volume 29, 2016.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 248–255, June 2009.

[9] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31Œ31: Revisiting large kernel design in cnns. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 11953–11965, June 2022.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. Learn. Represent.*, 2021.

[11] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, November 2018.

[12] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R. Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *Int. Conf. Comput. Vis.*, pages 3490–3499, October 2021.

[13] Chenping Fu, Risheng Liu, Xin Fan, Puyang Chen, Hao Fu, Wanqi Yuan, Ming Zhu, and Zhongxuan Luo. Rethinking general underwater object detection: Datasets, challenges, and solutions. *Neurocomputing*, 517:243–256, January 2023.

[14] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(2):652–662, February 2021.

[15] Shanghua Gao, Zhong-Yu Li, Qi Han, Ming-Ming Cheng, and Liang Wang. Rf-next: Efficient receptive field search for convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(3):1–19, June 2022.

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

[17] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2918–2928, June 2021.

[18] Ross Girshick. Fast r-cnn. In *Int. Conf. Comput. Vis.*, pages 1440–1448, December 2015.

[19] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 580–587, June 2014.

[20] Meng-Hao Guo, Cheng-Ze Lu, Qibin Hou, Zhengning Liu, Ming-Ming Cheng, and Shi-min Hu. Segnext: Rethinking convolutional attention design for semantic segmentation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Adv. Neural Inform. Process. Syst.*, volume 35, pages 1140–1156, 2022.

[21] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shi-Min Hu. Visual attention network. *Comput. Visual Media*, 9(4):733–752, July 2023.

[22] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R. Martin, Ming-Ming Cheng, and Shi-Min Hu. Attention mechanisms in computer vision: A survey. *Comput. Visual Media*, 8(3):331–368, March 2022.

[23] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, Zhaohui Yang, Yiman Zhang, and Dacheng Tao. A survey on vision transformer. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(1):87–110, January 2023.

[24] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Int. Conf. Comput. Vis.*, pages 2980–2988, October 2017.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1904–1916, September 2015.

[26] Qibin Hou, Cheng-Ze Lu, Ming-Ming Cheng, and Jiashi Feng. Conv2former: A simple transformer-style convnet for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):8274–8283, December 2024.

[27] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, volume 37, pages 448–456, July 2015.

[28] Glenn Jocher. Yolov5. https://github.com/ultralytics/yolov5, 2020.

[29] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Yolov8. https://github.com/ultralytics/ultralytics/, 2023.

[30] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM Comput. Surv.*, 54(10s):1–41, January 2022.

[31] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. *arXiv:1412.6980*.

[32] Frank Klinker. Exponential moving average versus moving exponential average. *Math. Semesterber.*, 58(1):97–107, December 2010.

[33] Boyi Li, Wenqi Ren, Dengpan Fu, Dacheng Tao, Dan Feng, Wenjun Zeng, and Zhangyang Wang. Benchmarking single-image dehazing and beyond. *IEEE Trans. Image Process.*, 28(1):492–505, January 2019.

[34] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications, 2022. *arXiv:2209.02976*.

[35] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M. Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13619–13627, June 2022.

[36] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11632–11641, June 2021.

[37] 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. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Adv. Neural Inform. Process. Syst.*, volume 33, pages 21002–21012, 2020.

[38] Yuxuan Li, Qibin Hou, Zhaohui Zheng, Ming-Ming Cheng, Jian Yang, and Xiang Li. Large selective kernel network for remote sensing object detection. In *Int. Conf. Comput. Vis.*, pages 16748–16759, October 2023.

[39] Yuxuan Li, Xiang Li, and Jian Yang. Spatial group-wise enhance: Enhancing semantic feature learning in cnn. In *Asian Conf. on Comp. Vis.*, pages 687–702, December 2022.

[40] Zhong-Yu Li, Xin Jin, Boyuan Sun, Chun-Le Guo, and Ming-Ming Cheng. Towards raw object detection in diverse conditions, 2024. *arXiv:2411.15678*.

[41] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 936–944, July 2017.

[42] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(2):318–327, February 2020.

[43] 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 *Eur. Conf. Comput. Vis.*, page 740–755, November 2014.

[44] 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. In *Int. Conf. Learn. Represent.*, 2022.

[45] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8759–8768, June 2018.

[46] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11966–11976, June 2022.

[47] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, 2016.

[48] 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, 2022. *arXiv:2212.07784*.

[49] Muhammad Maaz, Abdelrahman Shaker, Hisham Cholakkal, Salman Khan, Syed Waqas Zamir, Rao Muhammad Anwer, and Fahad Shahbaz Khan. Edgenext: efficiently amalgamated cnn-transformer architecture for mobile vision applications. In *Eur. Conf. Comput. Vis.*, page 3–20, February 2023.

[50] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Int. Conf. Comput. Vis.*, pages 3651–3660, October 2021.

[51] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 821–830, June 2019.

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, volume 32, 2019.

[53] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 779–788, June 2016.

[54] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 6517–6525, July 2017.

[55] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018. *arXiv:1804.02767*.

[56] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, volume 28, 2015.

[57] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 658–666, June 2019.

[58] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 4510–4520, June 2018.

[59] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Int. Conf. Comput. Vis.*, page 618–626, October 2017.

[60] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd, 2018. *arXiv:1805.00123*.

[61] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Int. Conf. Comput. Vis.*, pages 9626–9635, October 2019.

[62] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Adv. Neural Inform. Process. Syst.*, volume 30, 2017.

[63] Ao Wang, Hui Chen, Lihao Liu, Kai Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Yolov10: Real-time end-to-end object detection. In *Adv. Neural Inform. Process. Syst.*, 2024.

[64] 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. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Adv. Neural Inform. Process. Syst.*, volume 36, pages 51094–51112, 2023.

[65] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13024–13033, June 2021.

[66] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, page 7464–7475, June 2023.

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

[68] 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 *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, pages 1571–1580, June 2020.

[69] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. In *Eur. Conf. Comput. Vis.*, page 1–21, October 2024.

[70] Jiaqi Wang, Kai Chen, Shuo Yang, Chen Change Loy, and Dahua Lin. Region proposal by guided anchoring. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2960–2969, June 2019.

[71] Andrew Witkin. Scale-space filtering: A new approach to multi-scale description. In *ICASSP*, volume 9, pages 150–153, January 1984.

[72] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Dota: A large-scale dataset for object detection in aerial images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3974–3983, June 2018.

[73] 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, 2022. *arXiv:2203.16250*.

[74] Bowen Yin, Xuying Zhang, Deng-Ping Fan, Shaohui Jiao, Ming-Ming Cheng, Luc Van Gool, and Qibin Hou. Camoformer: Masked separable attention for camouflaged object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):10362–10374, December 2024.

[75] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. In *Int. Conf. Learn. Represent.*, 2023.

[76] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Int. Conf. Learn. Represent.*, 2018.

[77] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9756–9765, June 2020.

[78] Xuying Zhang, Xiaoshuai Sun, Yunpeng Luo, Jiayi Ji, Yiyi Zhou, Yongjian Wu, Feiyue Huang, and Rongrong Ji. Rstnet: Captioning with adaptive attention on visual and non-visual words. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 15465–15474, June 2021.

[79] Xuying Zhang, Bowen Yin, Zheng Lin, Qibin Hou, Deng-Ping Fan, and Ming-Ming Cheng. Referring camouflaged object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, page 1–14, early access, Jan. 21, 2025. doi:10.1109/tpami.2025.3532440.

[80] 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. In *IEEE Conf. Comput. Vis. Pattern Recog.*, page 16965–16974. IEEE, June 2024.

[81] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. *AAAI Conf. on Artif. Intel.*, 34(07):12993–13000, April 2020.

[82] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable {detr}: Deformable transformers for end-to-end object detection. In *Int. Conf. Learn. Represent.*, 2021.

陈宇铭 于 2022 年在兰州大学获得计算机科学学士学位。他目前是南开大学媒体计算实验室的博士研究生，师从程明明教授和侯淇彬教授。他的研究兴趣包括视觉理解、目标检测和知识蒸馏。

袁信彬 于 2023 年在西北农林科技大学获得信息管理与信息系统学士学位。他目前是南开大学媒体计算实验室的硕士研究生，师从程明明教授和侯淇彬教授。他的研究兴趣包括目标检测和遥感。

王家宝 于 2022 年在西北工业大学获得自动化硕士学位。他目前是南开大学媒体计算实验室的博士研究生，师从程明明教授和侯淇彬教授。他的研究兴趣涵盖计算机视觉和机器学习的多个主题，例如 2D/3D/定向目标检测以及自动驾驶中的 3D 感知。

武睿祺 于 2022 年获得武汉理工大学计算机与人工智能学院学士学位。他目前是南开大学媒体计算实验室的博士研究生，师从程明明教授和郭春乐教授。他的研究兴趣包括图像/视频生成和图像处理。

李翔 是南开大学计算机学院副教授。2020 年，他在南京理工大学（中国江苏）获得博士学位。他的研究兴趣包括 CNN/Transformer 主干网络、目标检测、知识蒸馏和自监督学习。他在 TPAMI、CVPR、NeurIPS 等顶级期刊和会议上发表了 30 余篇论文。

侯淇彬 在南开大学计算机学院获得博士学位。随后，他在新加坡国立大学担任研究员。目前，他是南开大学计算机学院的副教授。他在 T-PAMI、CVPR、ICCV、NeurIPS 等顶级会议/期刊上发表了 40 余篇论文。他的研究兴趣包括深度学习、图像处理和计算机视觉。

程明明 于 2012 年在清华大学获得博士学位。随后，他在牛津大学师从 Philip Torr 教授进行了 2 年的博士后研究。他现在是南开大学的教授，领导媒体计算实验室。他的研究兴趣包括计算机图形学、计算机视觉和图像处理。他获得的科研奖项包括国家杰出青年科学基金和 ACM 中国新星奖。他担任 IEEE TPAMI 和 IEEE TIP 的编委会成员。