计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
2025,34(7):140−151 [doi: 10.15888/j.cnki.csa.009882] [CSTR: 32024.14.csa.009882]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

融合剪枝和知识蒸馏的水下生物检测①

吴明轩[1], 李远禄[1,2], 王键翔[1]

[1](南京信息工程大学 自动化学院, 南京 210044)
[2](江苏省大气环境与装备技术协同创新中心, 南京 210044)
通信作者: 吴明轩, E-mail: 202212490059@nuist.edu.cn

摘　要: 针对捕鱼打捞、海底勘探等行业存在的现有水下设备存储和计算资源有限, 检测模型体积庞大, 难以在终端设备高效运行的问题, 提出一种融合剪枝和知识蒸馏的轻量级水下生物检测算法, 首先设计 C2f_GSConv 结构来替换原有 YOLOv8n 颈部网络中的 C2f 模块, 减少模型整体的计算复杂度, 优化模型结构; 其次引用 *MPDIoU* 来替换 CIoU 作为新的损失函数, 加快回归边界框收敛速度, 提升检测性能; 然后利用 LAMP 剪枝算法对模型进行裁剪, 去除冗余的通道信息和卷积核, 进一步地减少参数量和计算量, 压缩模型体积; 最后通过知识蒸馏来恢复模型的检测精度, 减少剪枝带来的不必要的精度损失. 实验结果表明, 在 URPC 数据集上, 改进后的模型相较于基准模型 YOLOv8n, *mAP*50 提升了 1.8%, 参数量减少了 62%, 计算量减少了 56%, FPS 提高了 186 f/s. 通过在嵌入式开发板上进行部署验证, 结果同样具备良好的性能, 因此能够满足水下低配置设备的应用部署.

关键词: 水下目标检测; YOLOv8n; 轻量化; 模型剪枝; 知识蒸馏

引用格式: 吴明轩,李远禄,王键翔.融合剪枝和知识蒸馏的水下生物检测.计算机系统应用,2025,34(7):140–151. http://www.c-s-a.org.cn/1003-3254/9882.html

Underwater Biological Detection Integrating Pruning and Knowledge Distillation

WU Ming-Xuan[1], LI Yuan-Lu[1,2], WANG Jian-Xiang[1]

[1](School of Automation, Nanjing University of Information Science & Technology, Nanjing 210044, China)
[2](Jiangsu Collaborative Innovation Center for Atmospheric Environment and Equipment Technology, Nanjing 210044, China)

Abstract: To address the issues of limited storage and computing resources in existing underwater equipment used in industries such as fishing and seabed exploration, as well as the large size of detection models, which are challenging to operate efficiently on terminal devices, a lightweight underwater biological detection algorithm combining pruning and knowledge distillation is proposed. First, the C2f_GSConv structure is designed to replace the C2f module in the neck of the YOLOv8n network, reducing the overall computational complexity and optimizing the model structure. Second, *MPDIoU* is introduced to replace CIoU as a new loss function, accelerating the convergence of the regression bounding box and improving detection performance. The LAMP pruning algorithm is then applied to trim the model by removing redundant channel information and convolutional kernels, further reducing the number of parameters and computational complexity, thus compressing the model size. Finally, knowledge distillation is employed to restore the model's detection accuracy and reduce the precision loss caused by pruning. Experimental results show that, on the URPC dataset, the improved model outperforms the benchmark YOLOv8n model with a 1.8% increase in *mAP*50, a 62% reduction in parameters, a 56% reduction in computational cost, and a 186 f/s increase. The results also demonstrate excellent performance upon deployment and verification on an embedded development board, confirming its suitability for

① 基金项目: 国家自然科学基金 (61671010)
收稿时间: 2024-12-09; 修改时间: 2025-01-02; 采用时间: 2025-01-21; csa 在线出版时间: 2025-05-27
CNKI 网络首发时间: 2025-05-28

application in low-configuration underwater equipment.

Key words: underwater target detection; YOLOv8n; lightweight; model pruning; knowledge distillation

海洋是生命的摇篮, 不仅孕育了生命, 还蕴藏着丰富的物质资源, 因此对海洋资源的合理利用和开发对整个社会的可持续发展起着战略性作用[1]. 其中水下生物目标检测技术是一项关键技术, 广泛应用于海底探矿, 渔业捕捞, 水产养殖等行业. 这些行业的发展大都依赖于水下嵌入式设备的运作, 而水下设备存储和计算资源受限, 使得大型网络难以在海洋中部署. 传统的检测算法往往参数量和计算量较大, 无法满足实时性检测的需求. 因此研究轻量级的水下目标检测算法具有深远而重大的意义.

近年来, 随着人工智能技术的飞速发展, 基于深度学习的目标检测算法逐渐成为主流, 大大提高了检测结果的鲁棒性, 其可以分为两阶段算法和单阶段算法. 两阶段算法主要以 R-CNN[2]和 Fast R-CNN[3]等算法为代表, 虽有着不错的检测精度, 但检测速度慢、实时性差且模型尺寸偏大, 不适合用于水下目标检测任务. 与两阶段检测算法相比, 单阶段检测算法能够直接在特征图中对目标进行分类和定位[4], 减轻计算复杂度的同时提升检测速度. 其中以 YOLO[5]系列算法为代表的单阶段检测算法以其结构灵活、高效快捷、低硬件要求等特点, 被广泛应用于计算机视觉任务中, 能够很好地满足水下目标检测轻量化的需求.

目前国内外科研人员对水下目标检测开展了一系列研究, 其中水下 YOLO 模型轻量化改进方面取得了不错的研究成果. Chen 等人[6]提出基于改进 YOLOv4 的水下目标识别算法, 采用反卷积模块替换传统上采样模块, 并引入深度可分离卷积优化特征提取过程, 不仅实现了识别精度与速度的同步优化, 同时降低了对计算资源的依赖, 提高了水下目标检测性能; 赵瑞金等人[7]提出了一种通道空间深度感知的轻量化水下目标检测网络 CSDP-L-YOLO, 通过线性操作和混洗结构生成特征映射, 减少冗余特征的融合和计算, 以减少模型参数量和计算量; 许朝龙等人[8]设计了一种轻量化网络检测算法 YOLOv8-FESF, 采用 Slim-neck 作为颈部结构压缩模型规模, 并重新设计检测头, 利用参数共享思想合并特征提取模块, 从而轻量化模型, 提高检测速度.

除了上述对网络模型的轻量化改进研究之外, 更多学者开始聚焦在轻量化方法本身. Li 等人[9]提出通过对训练后的网络修剪不重要的通道来完成模型压缩, 然后微调网络, 可以恢复原本性能. 该剪枝方法属于结构化剪枝范畴, 由于其在保持模型原始架构不变的同时, 能够在一般硬件平台实现高效推理, 因此近年来备受研究者关注. Changpinyo 等人[10]在研究中提出了网络稀疏化, 虽能获得一个较小的网络, 但也导致了网络性能的损失. 为进一步挖掘结构化剪枝的可能性, 研究人员采用基于稀疏正则化的结构化剪枝方法进行稀疏训练. Wen 等人[11]提出一种结构化稀疏性学习 (SSL) 方法来规定通道比例因子以及层深度等. Liu 等人[12]提出一种简单有效的 Network Slimming 剪枝算法, 利用批归一化 BN (batch normalization) 层中的通道缩放因子 γ 作为通道重要性度量指标, 并对 γ 参数进行 L1 正则化约束, 实现特征通道的稀疏化学习, 再将 γ 值小于预设阈值的通道进行裁剪, 实现模型压缩与加速.

模型剪枝的方法虽然可以大幅减少模型参数, 但剪枝后的模型往往会损失一部分精度. 为了避免由剪枝和量化带来的这一问题, Hinton 等人[13]提出了知识蒸馏的策略并将其应用于深度学习图像分类任务中. 知识蒸馏旨在使用规模较大、更复杂的教师模型将知识传输给规模较小、更简单的学生模型, 使学生模型的检测性能无限接近于教师模型, 最终得到一个性能相对较高且易于部署的网络模型. Ba 等人[14]研究证明, 学生模型的浅层网络依旧可以学习教师模型深层网络中的复杂知识. 而 Romero 等人[15]提出 FitNets, 对学生网络和教师网络的中间特征层进行连接, 使得学生网络能够获取教师网络的中间层信息, 从而得到深度更深, 效果更好的学生模型. FitNets 的引入很容易导致网络训练过程中过拟合现象的发生, 为此 Zagoruyko 等人[16]引入注意力特征模块来让学生网络去学习教师网络中间特征层中激活值更高的部分, 规范学生网络的学习. Czarnecki 等人[17]从数学角度出发, 将目标函数的导数与神经网络的训练相融合, 使得学生与教师之间的导数差异达到最小值. 2017 年, Chen 等人[18]首次将知识蒸馏技术应用于目标检测任务中, 开创性地在模型骨干网络中引入自适应层, 有效解决了师生模型

间的特征维度匹配问题. 知识蒸馏方法将教师模型的知识转移到学生模型中, 使得学生模型与教师模型达到一样的网络性能. 但如何选择合适的网络结构作为教师模型, 来帮助学生模型更高效地学习知识, 是目前需要进一步研究的问题.

综上, 上述轻量化研究改进方法都只是单独应用在模型中, 虽然展现出一定的效果, 但都存在各自的局限性. 本研究针对水下检测模型体积庞大, 参数量和计算量大, 难以应用在移动设备等问题, 选择 YOLOv8n 网络作为基准模型并对其结构进行轻量化改进, 融合模型剪枝和知识蒸馏的方法保证轻量化处理的有效性. 主要的处理工作有以下几点.

(1) 在网络结构方面, 设计 C2f_GSConv 模块来替换原始 YOLOv8n 颈部网络中的 C2f 模块, 减少计算冗余的同时加快网络特征提取速度.

(2) 通过引入 *MPDIoU* 损失函数来更好地计算损失, 使得边界框回归更加准确. 在不增加计算量的同时, 进一步提升模型的检测效果.

(3) 利用 LAMP 剪枝对结构改进后的 YOLOv8n 模型进行通道剪枝操作, 以获得更为轻量化的网络模型. 通过实验测试确定合适的剪枝比例, 并对剪枝后的模型进行微调, 实现精度与压缩程度间的平衡.

(4) 通过知识蒸馏来弥补因剪枝操作而损失的一部分精度, 进一步提升检测效率和准确率.

1 YOLOv8 算法原理

YOLOv8 是最新的 SOTA (state of the art) 模型, 是由开发团队 Ultralytics 在 YOLOv5 基础上进一步提出的, 其网络结构如图 1 所示, 主要分成 4 个部分: 输入端 (Input)、骨干网络 (Backbone)、颈部网络 (Neck) 和预测头 (Head). 其中, 输入端选用 Mosaic 数据增强方法, 有效提高了预测精度和模型性能. 在骨干网络层, YOLOv8 使用梯度流更丰富的 C2f 结构替换 YOLOv5 中的 C3 模块, 并对不同尺度的模型调整了不同的通道数, 轻量化了整体网络结构. 在颈部网络中, 采用特征金字塔网络 (FPN) 和路径聚合网络 (PAN) 相结合的双重特征金字塔结构, 实现了多尺度的特征融合 (FPN-PAN), 进一步提高特征提取能力. 在头部网络中, 从原先的耦合头变成了解耦头, 将分类和检测头分离, 并将 YOLOv5 中的有锚节点 (anchor-based) 替换成了无锚节点 (anchor-free), 这样可以直接预测目标的中心点和宽高比例, 从而进一步地提升模型的检测速度和精度.

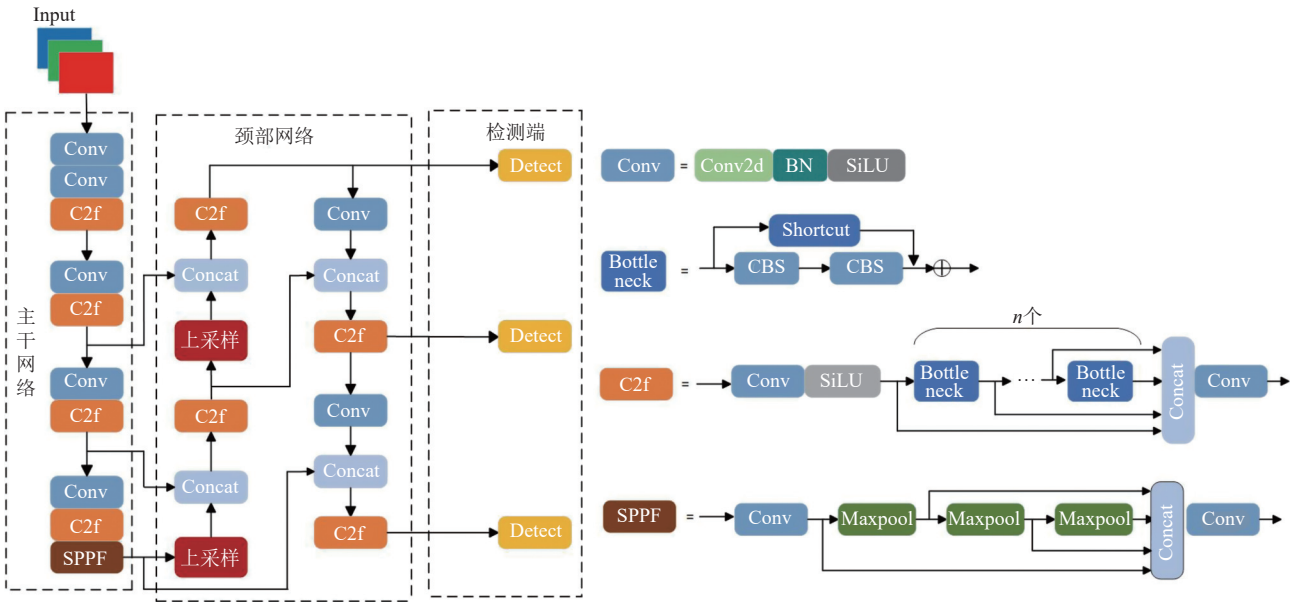


图 1 YOLOv8 网络结构图

2 YOLOv8 算法轻量化改进

YOLOv8 根据模型大小和深度的不同, 分成了 n、s、m、l、x 这 5 种尺寸, 适用于各种应用环境和计算资源. 其中 YOLOv8n 模型能在较快推理速度的同时, 保持目标设备的准确度, 并且不会产生较大的参数量. 因此为了更好地满足实时检测的需求, 保证检测速度,

本文选取复杂度较低的 YOLOv8n 作为基础模型.

2.1 设计 C2f_GSConv 结构替换 C2f 模块

由于水下采集的图像中背景错综复杂, 干扰因素多, 导致提取的特征信息在神经网络中存在大量的冗余, 影响整个网络的适应能力和鲁棒性. 卷积模块在计算机视觉任务中一直发挥着高效作用. 通过加入卷积模块可以减少模型参数, 为构建轻量化模型奠定基础.

鉴于 YOLOv8 网络的 C2f 结构中的 Bottleneck 模块存在大量的信息叠加, 导致结构特征冗余, 运行速度慢, 影响模型检测效率. 因此引入一种轻量级卷积模块 GSConv[19]来替换原有的 Bottleneck 模块, 设计 C2f_GSConv 模块替换原来的 C2f 模块, 通过自适应地调整卷积核大小来降低模型复杂度, 提升模型性能. GSConv 卷积模块结构如图 2 所示.

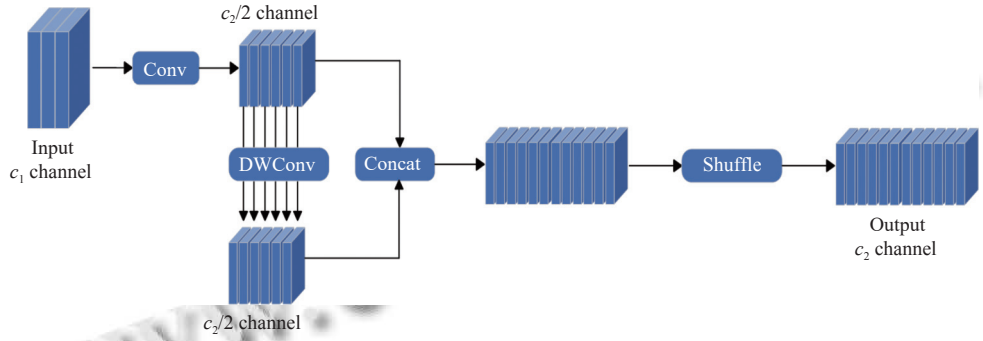


图 2 GSConv 模块结构

GSConv 结构主要由普通卷积 Conv、深度可分离卷积 DWConv[20]、Concat 层和 Shuffle 模块构成. 首先对通道数为 c_1 的输入特征图进行普通卷积下采样操作, 生成通道数为 $c_2/2$ 的特征向量, 然后对该向量进行深度可分离卷积, 得到一个通道数同样为 $c_2/2$ 的特征向量, 将两个特征向量通过 Concat 层拼接起来, 最后通过 Shuffle 操作将所有通道混洗重排, 完成通道信息的相互融合, 得到一个全新的通道数为 c_2 的特征图. 通过该卷积计算方法得到的输出特征图, 不仅能以更少的参数量达到普通卷积的效果, 且增强了信息的特征表达能力. 虽然 GSConv 结构可以减少计算成本, 但如果大量使用在模型网络层中, 会导致网络层加深, 模型推理时间增加, 因此本文仅在 YOLOv8 颈部网络的 C2f 结构中引入 GSConv 模块, 以减轻颈部网络复杂度.

采用 GSConv 卷积模块设计 C2f_GSConv 模块来替换 C2f 模块, 使得模型更加轻便, 大大减少了训练过程中推理的计算成本, 同时保持了足够的精度, 便于在一些水下边缘设备中进行部署. C2f_GSConv 结构如图 3 所示.

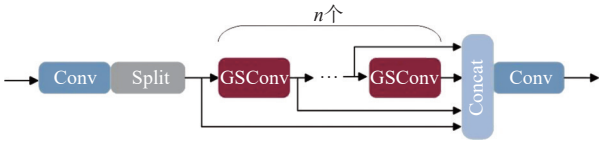


图 3 C2f_GSConv 模块结构图

2.2 优化损失函数

损失函数是边界框回归的重要组成部分, 用来衡量模型预测值与真实值间的相似性, 原始 YOLOv8 模型采用 CIoU 作为边界回归损失函数. 其计算公式为:

$$L_{\mathrm{CIoU}} = 1 - IoU + \frac{\rho^2(B_{\mathrm{gt}}, B_{\mathrm{pred}})}{c^2} + \alpha v \tag{1}$$

$$V = \frac{4}{\pi^2}\left(\arctan\frac{w^{\mathrm{gt}}}{h^{\mathrm{gt}}} - \arctan\frac{w}{h}\right)^2 \tag{2}$$

$$\alpha = \frac{v}{1 - IoU + v} \tag{3}$$

其中, ρ 表示真实框中心点到预测框中心点的长度, B_{gt} 和 B_{pred} 分别表示真实框和预测框的中心点位置, c 表示真实框和预测框重叠部分形成的最小外接矩形框对角线的长度, α 为惩罚函数, 用来平衡参数. v 是修正因子, 用来衡量纵横比的相似性. w^{gt} 和 h^{gt} 分别表示真实框的宽度和高度, w 和 h 分别表示预测框的宽度和高度. IoU 表示真实框和预测框交集与并集的比值. 从式 (2) 可知, 当真实框和预测框的宽高比例设置相同时, CIoU 中的 v 会变成 0, 失去修正作用, 无法精准捕捉边界框形状的差异性, 导致损失函数失去作用. 此外在实际检测场景中, 一旦数据集中存在质量较低的数据样本时, CIoU 边界框回归的准确性会明显降低, 从而影响模型的收敛效果和检测精度.

因此, 考虑到 CIoU 损失函数的局限性, 本文引用

了一种基于最小点距离的新型边界框回归损失函数 *MPDIoU*[21], 能够最小化预测框和真实框左上角和右下角间的距离, 有效提高边界框回归效率. 其计算公式为:

$$d_1^2 = (x_1^{\text{pred}} - x_1^{\text{gt}})^2 + (y_1^{\text{pred}} - y_1^{\text{gt}})^2 \tag{4}$$

$$d_2^2 = (x_2^{\text{pred}} - x_2^{\text{gt}})^2 + (y_2^{\text{pred}} - y_2^{\text{gt}})^2 \tag{5}$$

$$MPDIoU = IoU - \frac{d_1^2}{w^2+h^2} - \frac{d_2^2}{w^2+h^2} \tag{6}$$

$$L_{MPDIoU} = 1 - MPDIoU \tag{7}$$

其中, $(x_1^{\text{pred}}, y_1^{\text{pred}})$、$(x_1^{\text{gt}}, y_1^{\text{gt}})$分别代表预测框和真实框的左上角坐标位置; $(x_2^{\text{pred}}, y_2^{\text{pred}})$、$(x_2^{\text{gt}}, y_2^{\text{gt}})$分别代表预测框和真实框的右下角坐标位置; d_1表示预测框和真实框左上角的欧氏距离; d_2表示预测框和真实框右下角的欧氏距离; w 和 h 分别表示输入图像的宽和高.

相较于 CIoU, *MPDIoU* 通过考虑边界框本身的形状、位置和中心点距离等因素来计算损失, 从而简化计算过程, 提高了模型的回归精度和收敛速度. 使用 *MPDIoU* 可以很好地适应样本差异大而导致的检测失真情况, 进一步增强模型的鲁棒性.

2.3 LAMP 通道剪枝

模型剪枝是神经网络轻量化最常用的一种方式, 通过对原有网络进行特定的裁剪, 剪去不重要的通道和连接, 移除冗余部分, 大幅度减少模型参数量和计算量的同时, 将模型精度保持在原有水平. 本文采用基于层自适应幅度的通道剪枝方法 LAMP[22]对改进后的 YOLOv8n 网络进行修剪, 进一步完成轻量化设计. LAMP 提出了一种新的计算权重的全局剪枝方法, 不需要复杂的运算和超参数的调试, 只需依靠 LAMP 分数和剪枝压缩率的设置即可完成剪枝任务, 且在大多数网络中展现出更优异的效果. 其具体计算公式如式 (8) 所示:

$$Score(u;W) = \frac{(W[u])^2}{\sum_{v \geqslant u}(W[v])^2} \tag{8}$$

$$(W[u])^2 > (W[v])^2 \rightarrow Score(u;W) > Score(v;W) \tag{9}$$

其中, W 为权重, u 和 v 代表权重的索引映射, $W[u]$和 $W[v]$代表索引 u 和 v 对应的权重项, 权重依据指定的索引映射进行升序排列, 即当 $u<v$ 时, $W[u] \leqslant W[v]$. 由式 (8) 可以推导出权重值与 LAMP 评分呈正相关, 权重越大, 对应的 LAMP 评分也越高, 而那些评分较低的权重项就会作为不重要的连接而被剪掉. LAMP 评分机制能够有效评估网络各层中连接权重的重要性, 通过迭代移除评分最低的权重连接, 直到满足预设的剪枝需求.

LAMP 分数剪枝的具体操作流程为: 1) 对预训练完成的 YOLOv8n 模型权重进行参数初始化操作. 2) 逐层计算各连接权重的 L2 范数平方值, 并通过层内归一化处理获得标准化的 LAMP 评分. 3) 根据预设的全局剪枝率和各连接的 LAMP 评分分布, 采用阈值判定法对低重要性连接进行选择性剪除. 4) 通过微调训练对剪枝后的模型进行优化, 以补偿剪枝过程带来的精度损失. LAMP 剪枝过程如图 4 所示.

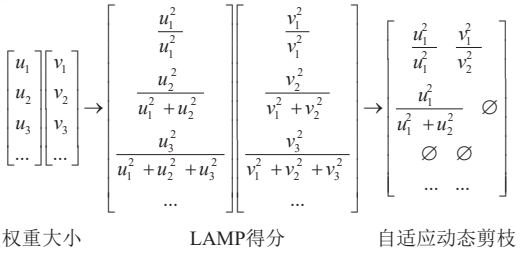


图 4 LAMP 剪枝过程

LAMP 剪枝方法根据每个连接的 LAMP 得分, 自适应地进行动态剪枝, 其优势在于每一层都会至少保留一个权重, 也就至少保留一个存活连接, 有效地避免了因盲目剪枝而导致的层崩溃现象发生. 通过应用 LAMP 算法对 YOLOv8n 模型进行通道剪枝, 不仅能够有效降低模型复杂度, 还能够降低功耗并提高推理速度, 使其更适合在资源受限的水下嵌入式设备上进行部署.

2.4 知识蒸馏

剪枝后的模型不可避免地会造成精度下降, 尽管通过微调训练可以使模型重新适应结构, 恢复一定的性能, 但是效果有限. 为了获得更好的轻量化模型性能, 本文采用知识蒸馏方法对剪枝后的模型继续训练, 尽可能地消除剪枝带来的负面影响. 知识蒸馏的核心思想在于将性能更好、更复杂的教师模型中学习到的暗知识转移到简单的学生模型中, 以此来指导学生模型获得和教师模型同样的性能. 因此结合本文实际情况, 选择未剪枝的 YOLOv8n 改进模型作为教师网络, 而剪枝后的模型作为学生网络进行知识蒸馏. 其训练过程如图 5 所示.

在知识传递过程中, 目标输出通常分为软标签和硬标签两部分, 软标签是使用教师网络来训练学生网络时的预测输出目标值, 硬标签是使用原始标签来训练学生网络时的输出目标值, 具体公式如下:

$$q_i = \frac{\exp(Z_i/T)}{\sum_j \exp(Z_j/T)} \tag{10}$$

其中, q_i 为经过 *Softmax* 后的函数输出; Z_i 为类别 i 的输出概率; Z_j 为类别 j 的输出概率. T 为温度变量, 当 T=1 时, 该函数为原始 *Softmax* 函数; 当 T 趋向于 0 时, *Softmax* 输出为硬目标; 当 T 趋向于无穷时, *Softmax* 输出则变为软目标. T 值越大, *Softmax* 的输出概率分布熵也越大, 负标签获取的信息也越多.

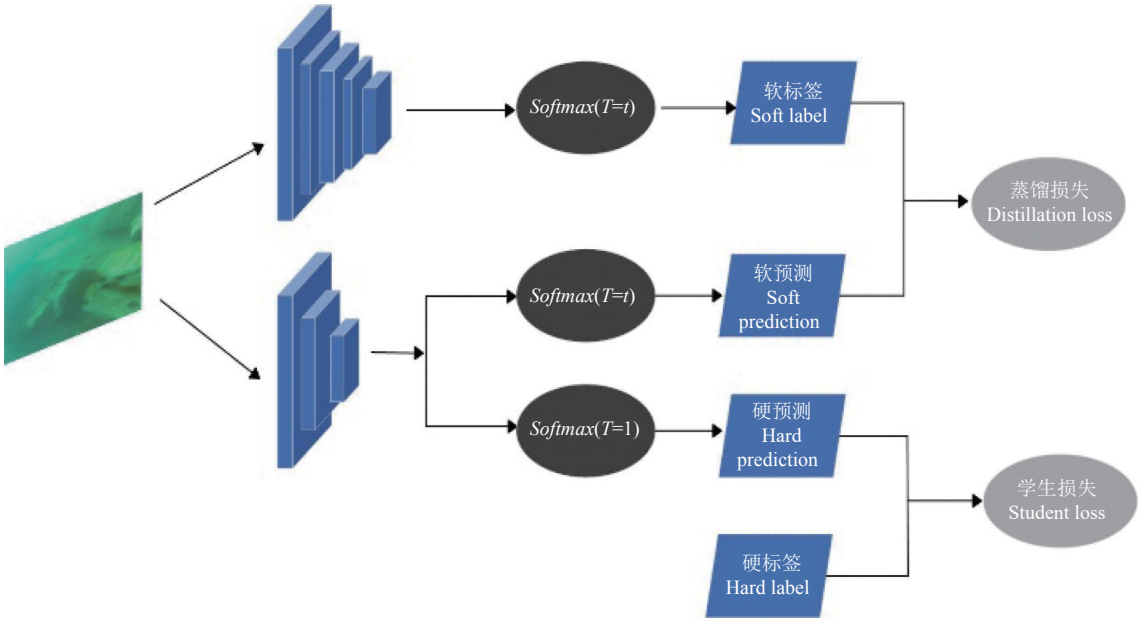


图 5 知识蒸馏过程

在图 5 训练新模型的过程中, 通过引入温度参数 T 来调整 *Softmax* 输出信息, 使用较高的 T 让 *Softmax* 输出经过升温后产生足够软的标签, 由此得到与原模型 *Softmax* 输出近似的新模型. 当训练结束后, 重新使用正常的温度 T 来进行预测, 其中原始的 *Softmax* 输出向量为硬预测, 调整过后的 *Softmax* 输出向量为软预测. 结合硬标签标注的真实值构造学生损失函数和蒸馏损失函数, 两部分损失函数相加得到总体损失函数, 其计算公式如下:

$$L_{\text{loss}} = (1-\alpha)L_{\text{student}} + \alpha T^2 L_{\text{distill}} \tag{11}$$

其中, L_{loss} 为总体损失函数的值; L_{student} 为学生损失函数的值, 代表硬预测值和硬标签值之间的差异; L_{distill} 为蒸馏损失函数的值, 代表软预测值和软标签值之间的差异; α 为平衡系数, 用来控制两种损失函数在训练过程中的参数, 从而确保了教师模型知识的有效迁移, 使得检测精度能够恢复到剪枝前的效果, 提升了学生模型检测性能.

3 实验结果与分析

3.1 实验环境及参数配置

本实验的目标检测模型训练及测试均在 Windows 11 操作系统上进行, CPU 环境为 Core i7-12650H, 内存 16 GB, GPU 环境为 NVIDIA GeForce RTX 4060, 显存为 8 GB. 采用 PyTorch 1.13 深度学习框架, Python 3.8 作为编程语言, CUDA 版本为 11.6, 编译软件为 PyCharm. 除此之外设置其他一些网络参数: 输入图片大小为 640×640, 训练轮数为 100 轮, 初始学习率为 0.01, batchsize 为 16, 并采用 YOLOv8n 作为基础网络模型. 其他超参数设置如表 1 所示.

表 1 实验超参数设置

参数	数值
优化器	SGD
权重衰减系数	0.000 5
动量	0.937
*IoU*训练阈值	0.2
分类损失系数	0.5
Mosaic增强	1.0
*IoU*损失系数	0.05

3.2 实验数据集

为了更好地验证改进后模型的有效性与实用性, 本实验采用水下机器人抓取比赛 URPC 数据集来进行模型测试与训练. 该数据集包含丰富的水下物种合集, 拥有 7600 幅水下光学图像, 为水下目标检测任务提供了可靠的科学依据. 其中主要分为 4 种常见的海洋生物目标, 分别为“扇贝 (scallop)”“海胆 (echinus)”“海星 (starfish)”和“海参 (holothurian)”. 经过筛选, 去除无目标图像和海草相关样本, 最后保留了 6 625 张有效图像. 并按 8:2 的比例划分训练集和测试集, 其中训练集有 5300 张图像, 测试集有 1325 张图像.

3.3 评价指标

本文采用目标检测常用评估指标来衡量网络性能, 分别为准确率 (Precision, *P*)、召回率 (Recall, *R*)、参数量 (Params)、计算量 (FLOPs)、平均精度均值 (*mAP*) 及模型每秒检测图像帧数 (FPS). 具体计算公式为:

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$AP = \int_0^1 P(R)\mathrm{d}R \tag{14}$$

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i \tag{15}$$

其中, *TP* 表示正确检测目标的数量, *FP* 表示错误检测目标的数量, *FN* 表示未检测到目标的数量. *AP* 是平均精度, 衡量模型在单个类别上的识别效果. 以精确率值为纵轴, 召回率值为横轴所构成的 *P*-*R* 曲线与横纵轴围起来的面积就是 *AP* 的值. *mAP* 是对所有类别的 *AP* 值取平均之后得到的结果, 值越高说明模型的准确性越好. *mAP*50 代表在 *IoU* 阈值设为 0.5 时的平均精度, 而 *mAP*50:95 代表 *IoU* 阈值取值从 0.5–0.95, 步长为 0.05 时的平均精度. FPS 用来衡量模型的运算速度, 当 FPS 越大表示检测速度越快.

3.4 不同剪枝率下的模型性能对比实验

为研究不同剪枝率对模型性能的影响, 确定合适的剪枝比例. 本文对改进后的 YOLOv8n 模型设置不同的剪枝比例进行剪枝并设计了一组对比实验, 得到的实验结果如表 2 所示. 结果表明, 随着剪枝率的增大, 模型的参数量和计算量随之相应减少, 然而受到剪枝影响, 模型的精度也呈现下降趋势. 但是当剪枝率小于 0.4 时, 经过微调后的模型 *mAP*50 值均大于未剪枝的模型, 而当剪枝率大于 0.4 时, 模型 *mAP*50 值较未剪枝的模型出现了不同程度的下降, 其中剪枝率为 0.5 时, 模型的准确率提升 0.2%, 召回率提升 0.8%, 精度下降 0.2%, 并未导致模型整体性能出现明显下滑, 但其参数量下降 58%, 计算量下降 53%, 大幅缩减了模型复杂度, 兼顾了轻量化与精度的需求, 因此最终选择 50% 剪枝率下的模型进行后续的知识蒸馏操作.

表 2 不同剪枝率对比实验结果

剪枝率	Params (M)	FLOPs (G)	*P* (%)	*R* (%)	*mAP*50 (%)	FPS (f/s)
0	2.81	7.65	80.2	80.4	84.0	396
0.1	2.53	7.12	80.5	81.9	86.3	423
0.2	2.17	6.5	80.3	82.4	85.6	466
0.3	1.91	5.65	80.4	80.6	85.1	512
0.4	1.63	4.94	81.1	79.9	84.3	523
0.5	**1.18**	**3.58**	**80.4**	**81.2**	**83.8**	**567**
0.6	1.03	3.02	79.6	80.1	82.5	590
0.7	0.86	2.63	78.8	77.6	81.3	634
0.8	0.73	2.18	76.7	75.5	78.2	660
0.9	0.59	1.96	76.2	70.3	75.4	693

3.5 消融实验

3.5.1 损失函数消融实验

为了验证本文所引入的损失函数的高效性, 将 *MPDIoU* 与其他几种常见的损失函数进行对比, 包括 CIoU[23]、GIoU[24]、DIoU[25]、EIoU[26]和 SIoU[27]. 在相同实验环境下, 设置一组消融对比实验, 得到的实验结果如表 3 所示, 然后根据实验结果将训练过程中的损失函数曲线进行可视化对比, 如图 6 所示. 根据表 3 数据, 使用 *MPDIoU* 损失函数时获得了最高的 *mAP*50 和 *mAP*50:95 值, 相较基准模型分别提升了 0.9% 和 1.1%, 说明使用 *MPDIoU* 损失函数的 YOLOv8n 模型在边界框定位精度方面相较其他损失函数表现得更好, 更适配于本文提出的水下目标检测任务. 同时由图 6 可知, *MPDIoU* 的最终收敛值远小于其他损失函数的最终收敛值, 说明 *MPDIoU* 收敛性能更好. 综上, 证明了引入 *MPDIoU* 损失函数的卓越性.

表 3 损失函数对比实验结果 (%)

损失函数	*P*	R	*mAP*50	*mAP*50:95
CIoU[23] (baseline)	80.0	80.1	83.3	46.5
GIoU[24]	79.9	80.3	83.1	46.1
DIoU[25]	80.4	79.8	82.4	45.8
EIoU[26]	80.1	**80.5**	83.8	46.9
SIoU[27]	**80.7**	79.6	84.0	47.2
MPDIoU	80.2	80.4	**84.2**	**47.6**

3.5.2 改进过程消融实验

为了进一步地验证本文各改进策略的有效性, 以 YOLOv8n 作为基准模型, 通过采取逐步增加改进步骤的方法进行多组消融实验, 得到的实验结果如表 4 所示.

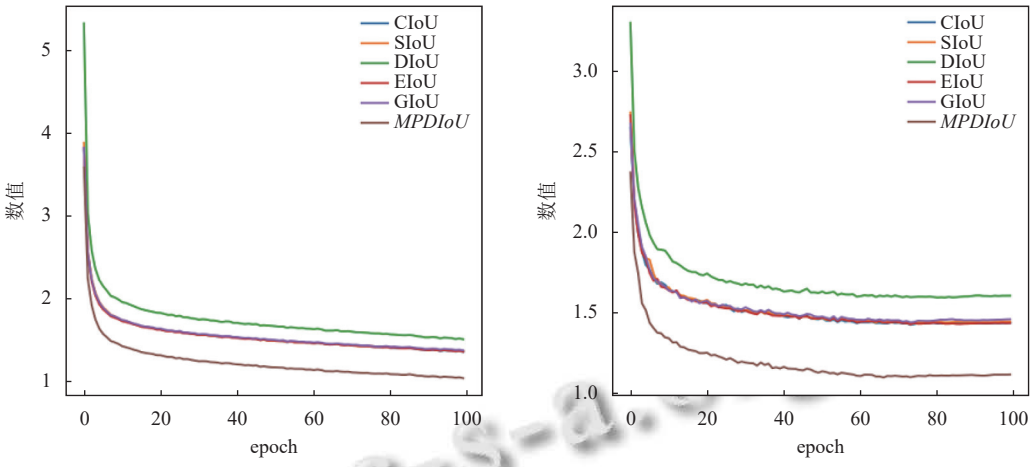


图 6 损失函数对比图

表 4 改进过程消融实验结果

模型	GSConv	*MPDIoU*	LAMP 剪枝	知识蒸馏	Params (M)	FLOPs (G)	*mAP*50 (%)	FPS (f/s)
1	×	×	×	×	3.1	8.1	83.3	381
2	√	×	×	×	2.81	7.65	82.8	396
3	√	√	×	×	2.81	7.65	84.0	396
4	√	√	√	×	1.18	3.58	83.8	567
5	√	√	√	√	**1.18**	**3.58**	**85.1**	**567**

模型 1 为原始 YOLOv8n 模型, 没做任何改进. 模型 2 为引入 GSConv 模块之后的模型, 根据表 4 可知, 模型参数量和计算量分别下降 0.29M 和 0.45G, 证实了所选卷积模块的轻量性. 但检测精度也因此受到了影响, *mAP*50 下降了 0.5 个百分点. 所以模型 3 在模型 2 基础上, 引入了 *MPDIoU* 损失函数, 在参数量和计算量保持不变的情况下, 将 *mAP*50 提高至 84%, 相较基准模型也提高了 0.7%, 进一步提升了模型的检测精度. 模型 4 对改进后的 YOLOv8n 模型进行 LAMP 剪枝操作, 通过设置 50% 的剪枝比例得到参数量仅为 1.18M, 计算量仅为 3.58G 的新模型, 对比剪枝前后, 参数量下降 1.63M, 计算量下降 4.07G, FPS 提升 171 f/s, 且经过微调, 模型的精度仅有 0.2% 的损失. 模型 5 对剪枝后的模型进行知识蒸馏, 实验结果表明, 经过蒸馏后的模型精度提升了 1.3 个百分点, 较基准模型提高了 1.8 个百分点. 综合来看, 本文所提改进算法不仅满足轻量化网络的要求, 而且提升了模型检测性能, 降低了在移动设备上部署的难度.

3.6 不同算法对比实验

为验证本文所提改进算法的优越性, 首先, 与目前主流目标检测算法进行对比实验, 选取代表性网络 Faster R-CNN[28]、SSD[29]、YOLOv5s[30]、YOLOv7[31]、YOLOv7-tiny、YOLOv8s 及 YOLOv9s[32]作为参照网络; 其次, 选取在一些在文献中已经改进过的轻量化水下目标检测算法与本文算法再次进行对比实验, 包括 YOLOv7-PSS[33] 、CSDP-L-YOLO[7]、YOLOv8-FESF[8] 等算法. 得到的实验结果如表 5 所示.

表 5 不同算法对比实验结果

模型	Params (M)	FLOPs (G)	*mAP*50 (%)	FPS (f/s)
Faster R-CNN	113.4	412.4	66.4	32
SSD	24.1	215	75.4	51
YOLOv4	27.6	68.4	79.3	77
YOLOv5s	7	16	79.3	77
YOLOv7	36.5	103.2	79.8	62
YOLOv7-tiny	6	13.2	77.6	213
YOLOv8s	11.2	28.6	84.5	256
YOLOv8n	3.1	8.1	83.3	381
YOLOv9s	7.1	26.4	84.4	189
CSDP-L-YOLO	5.64	13.9	79.8	261
YOLOv7-PSS	29	82.9	**87.3**	67
YOLOv8-FESF	1.8	3.7	84.2	550
Ours (未剪枝)	2.81	7.65	84	396
Ours (50%剪枝)	1.18	3.58	83.8	567
Ours (剪枝+蒸馏)	**1.18**	**3.58**	85.1	**567**

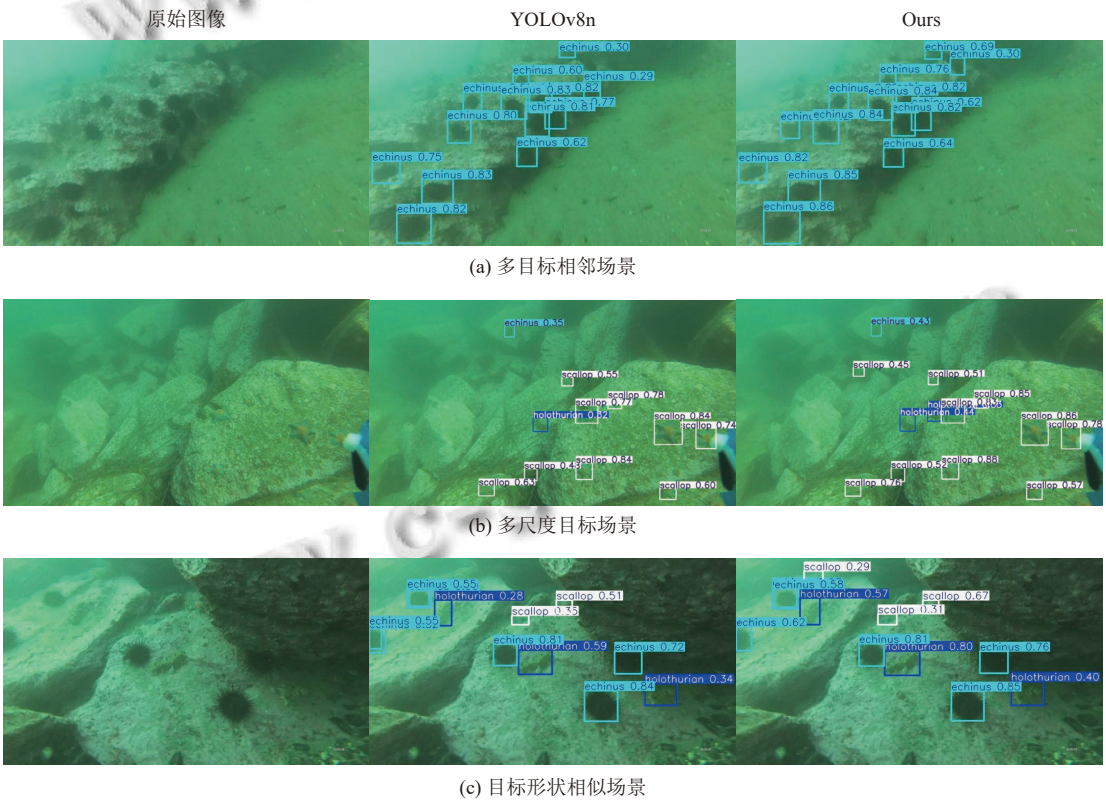
从表 5 可以看出, 主流目标检测算法普遍结构比较复杂, 参数量和计算量较大, 没有达到轻量化的要求.

而 YOLOv7 的轻量化算法 YOLOv7-tiny, 其参数量为 6M, 计算量为 13.2G, 该模型仍不够轻便. 此外精度更高、模型稍复杂的 YOLOv8s, 也仅比 YOLOv8n 高出 1.2% 的精度, 但参数量和计算量却接近 YOLOv8n 的 4 倍. 因此本文选择 YOLOv8n 模型作为基准模型是非常合理的. 轻量化算法方面, 可以发现 CSDP-L-YOLO 的轻量化效果并不明显. 而对于 YOLOv7-PSS, 可以看到 *mAP*50 达到了 87.3%, 为表 4 中的最高值, 说明此算法在检测精度方面非常出色, 但其参数量和计算量却高达 29M 和 82.9G, 较 YOLOv8n 算法多出近 9 倍, 且 FPS 也只有 67 f/s, 远没有达到轻量化指标. 最后一种算法 YOLOv8-FESF 的参数量和计算量仅为 1.8M 和 3.7G, 精度达到 84.2%, 较基准模型确实有显著效果, 但是对比本文经过剪枝和蒸馏后的算法模型, 参数量多了 0.62M, 计算量多了 0.12G, 精度也不如本文算法. 综上所述, 本文算法是一种更全面, 性能更优的算法, 在检测精度、速度以及轻量化方面均达到了平衡, 未来可以应用于小型存储空间设备, 具有一定的发展前景.

3.7 可视化分析

3.7.1 检测图对比

为了更加直观地反映本文改进检测算法的提升效果, 选取几组在不同场景下的原始 YOLOv8n 算法与本文改进算法的检测图进行可视化对比, 如图 7 所示. 其中左侧为水下原始图像, 中间为 YOLOv8n 算法的检测效果, 右侧为本文改进算法的检测效果. 根据对比图可以清晰地看到, 无论在哪种特定场景下, 本文改进后的算法都可以准确无误地检测出水下生物的类别, 且检测精度要明显优于原始 YOLOv8n 算法, 证明了本文改进算法能在保证轻量化的同时具有较高的准确性.



(a) 多目标相邻场景

(b) 多尺度目标场景

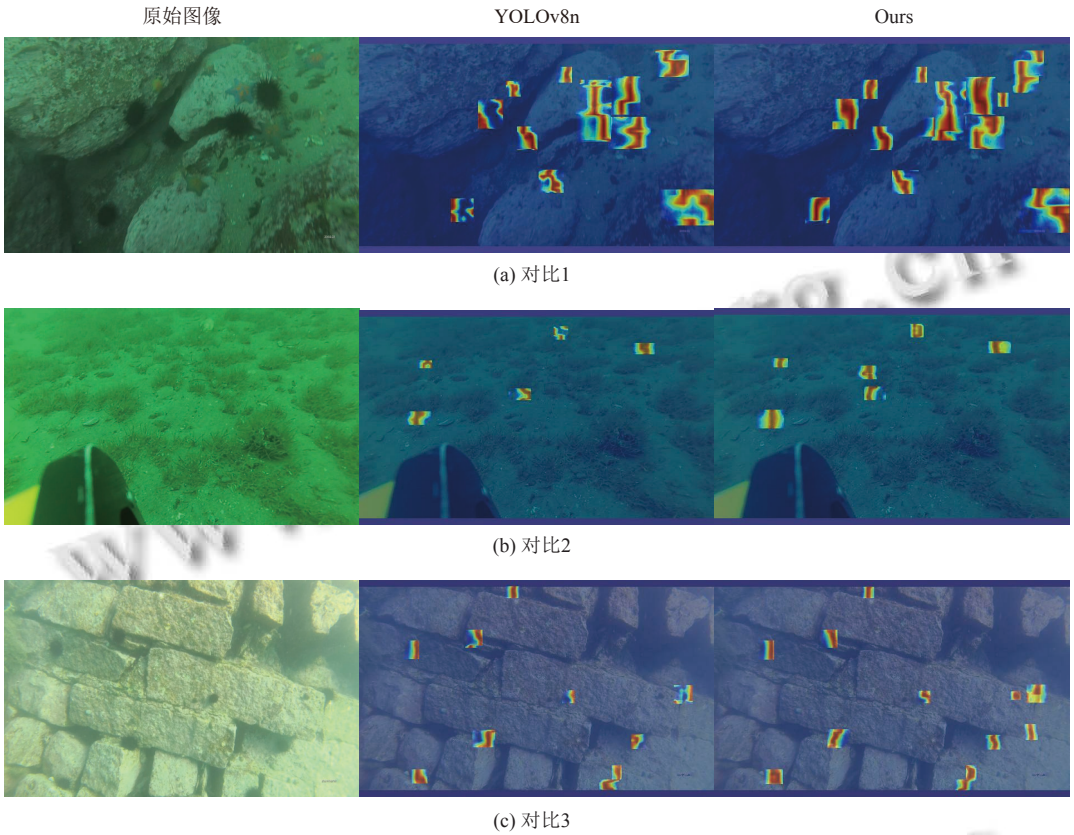
(c) 目标形状相似场景

图 7 检测效果对比图

3.7.2 热力图对比

为了进一步分析模型的检测性能, 本文在测试集中选取了 3 类图像进行热力图可视化. 对基准模型 YOLOv8n 和本文改进后的算法分别生成热力图, 并进行分析对比, 如图 8 所示, 其中左侧为原始图像, 中间为基准模型, 右侧为改进模型. 热力图能够简洁明了地反映模型重点关注的特征图区域. 一般而言, 特征图中梯度较高的区域, 数据密集程度越高, 在热力图中以较

暖的红色表示. 相反梯度较低的区域则以较冷的蓝色表示. 模型的重点关注区域颜色越深, 代表模型的检测性能越好. 从图 8 可以看出, 经过结构改进、剪枝以及蒸馏后的 YOLOv8n 模型对于要检测的特征区域更为敏感, 关注度更高, 因此可以更加准确地检测出待检目标图像的信息, 提升模型的检测精度和整体性能.



(a) 对比1

(b) 对比2

(c) 对比3

图 8 热力对比图

3.8 嵌入式设备部署和应用

为进一步检验改进模型在资源受限的嵌入式平台中的性能表现, 本文在嵌入式开发板 Jetson Nano B01 上进行模型部署和验证实验, Jetson Nano B01 是由 NVIDIA 公司研发的一款小型人工智能计算平台, 可以应用于深度学习算法的部署, 常作为一些边缘计算场景的终端设备, 其实物图如图 9 所示.

图 9 Jeston Nano B01 实物图

该开发板搭载 128 核 NVIDIA Maxwell 架构的 GPU 系统以及 4 核 ARM Cortex A57 的 CPU 系统, 支持多种深度学习框架, 包括 PyTorch 和 TensorFlow 等. 同时包含了 4 个 USB 3.0 接口、1 个 USB 2.0 接口、1 个 Micro-B、1 个 HDMI 输出以及 GPIO 针脚, 用以连接各种外接设备. 其具体配置参数如表 6 所示.

表 6 Jetson Nano B01 配置参数

配置	参数
CPU	64位 4核 ARM Cortex A57 @ 1.43 GHz
GPU	128核 NVIDIA Maxwell
内存	4 GB 64位 LPDDR4
接口	4×USB 3.0、USB 2.0、Micro-B、GPIO
输出端口	HDMI
存储	MicroSD

根据表 6 可知, Jetson Nano B01 的内存仅为 4 GB, 而本文第 3.1 节中所采用的 GPU 平台 RTX 4060 显存达到了 8 GB, 此外嵌入式计算卡的算力与 GPU 设备

相比也存在不小的差距. 以本文使用的嵌入式平台为例, Jetson Nano B01 算力仅有 472 FLOPs, 而用于模型训练的 RTX 4060 显卡算力高达 25 500 FLOPs, 两者浮点算力相差约 54 倍之多. 因此将原始 YOLOv8n 模型和改进后的模型分别部署到 Jetson Nano B01 开发板上并完成测试, 模拟在水下低配置低算力移动设备中的应用, 实验数据集依然采用 URPC, batchsize 设定为 1, 置信度阈值设为 0.5, 评价指标主要为 CPU 下的推理 FPS, 得到的实验结果如表 7 所示.

表 7 算法嵌入式设备部署实验结果

模型	Params (M)	*mAP*50 (%)	FPS (f/s)
YOLOv8n	3.1	82.8%	11
Ours (未剪枝)	2.81	83.6	28
Ours (剪枝+蒸馏)	1.18	84.8	58

从实验结果可以看出, YOLOv8n 模型在 Jetson Nano B01 平台上检测速度相当之慢, FPS 仅为 11 f/s, 难以保证检测的实时性. 而本文的改进模型在未剪枝的情况下将 FPS 提升至 28 f/s, 检测速度得到了明显改善, 在经过剪枝和知识蒸馏后 FPS 达到了 58 f/s, 相比 YOLOv8n 提高了 47 f/s, 并且远远超出 FPS≥30 f/s 的实时性标准, 同时在精度方面也仍然保持着更高的准确率.

综上来看, 本文提出的轻量化模型在资源受限的嵌入式设备中同样更具优势, 基本满足对于水下目标检测任务的实时性要求, 未来可以应用在一些水下低配置设备中.

4 结论与展望

本文针对水下检测效率差, 模型复杂, 难以在小型设备上部署的问题. 以 YOLOv8n 为基准模型, 从轻量化角度出发, 首先对其进行结构上的改进, 设计 C2f_GSConv 模块来替换颈部网络中的 C2f 模块, 降低模型的复杂性; 其次引入新的损失函数 *MPDIoU* 提升模型的检测精度; 最后融合模型剪枝和知识蒸馏的方法, 确保改进模型的有效性. 实验结果表明, 相较于 YOLOv8n, 改进后的算法在参数量和计算量上都有大幅度的下降, 并且提高了模型的实时检测速度, 满足了轻量化的需求. 而在检测精度方面, 改进后的算法也有 1.8% 的提升, 对比其他几种算法, 本文模型的性能最为均衡高效. 在后续的研究中, 为了进一步对水下模型进行轻量化改进, 会继续尝试在模型结构层面进行不一样的改进, 优化模型结构. 此外, 还会采用低秩分解等方法更好地对模型进行压缩, 降低模型计算开销. 并在水下嵌入式设备上部署以进行实地检测, 验证方法的可行性.

参考文献

1 林森, 赵颖. 水下光学图像中目标探测关键技术研究综述. 激光与光电子学进展, 2020, 57(6): 060002.

2 Xie XX, Cheng G, Wang JB, *et al*. Oriented R-CNN for object detection. Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision. Montreal: IEEE, 2021. 3500–3509.

3 Liu Y, Jing ZB. Power system relay protection based on Faster R-CNN algorithm. International Journal of Information Technology and Web Engineering (IJITWE), 2023, 18(1): 1–15.

4 Zhao ZQ, Zheng P, Xu ST, *et al*. Object detection with deep learning: A review. IEEE Transactions on Neural Networks and Learning Systems, 2019, 30(11): 3212–3232. [doi: 10.1109/TNNLS.2018.2876865]

5 Redmon J, Divvala S, Girshick R, *et al*. You only look once: Unified, real-time object detection. Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 779–788.

6 Chen LY, Zheng MC, Duan SQ, *et al*. Underwater target recognition based on improved YOLOv4 neural network. Electronics, 2021, 10(14): 1634. [doi: 10.3390/electronics10141634]

7 赵瑞金, 李海涛, 陆光豪. 通道空间深度感知的轻量化水下目标检测. 计算机测量与控制, 2024, 32(9): 86–93. [doi: 10.16526/j.cnki.11-4762/tp.2024.09.013]

8 许朝龙, 解志斌, 宋科宁. 基于轻量化网络的水下目标检测算法. 无线电工程, 2025, 55(2): 264–270.

9 Li H, Kadav A, Durdanovic I, *et al*. Pruning filters for efficient ConvNets. arXiv:1608.08710v3, 2017.

10 Changpinyo S, Sandler M, Zhmoginov A. The power of sparsity in convolutional neural networks. arXiv:1702.06257, 2017.

11 Wen W, Wu CP, Wang YD, *et al*. Learning structured sparsity in deep neural networks. Proceedings of the 30th International Conference on Neural Information Processing Systems. Barcelona: Curran Associates Inc., 2016. 2082–2090.

12 Liu Z, Li JG, Shen ZQ, *et al*. Learning efficient convolutional networks through network slimming. Proceedings of the 2017 IEEE International Conference on Computer Vision. Venice: IEEE, 2017. 2755–2763.

13 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a

neural network. arXiv:1503.02531, 2015.
14 Ba LJ, Caruana R. Do deep nets really need to be deep? Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal: MIT Press, 2014. 2654–2662.
15 Romero A, Ballas N, Kahou SE, *et al*. FitNets: Hints for thin deep nets. arXiv:1412.6550, 2015.
16 Zagoruyko S, Komodakis N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. Proceedings of the 5th International Conference on Learning Representations. Toulon: OpenReview.net, 2017.
17 Czarnecki WM, Osindero S, Jaderberg M, *et al*. Sobolev training for neural networks. Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach: NIPS, 2017. 4281–4290.
18 Chen GB, Choi W, Yu X, *et al*. Learning efficient object detection models with knowledge distillation. Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 742–751.
19 Li HL, Li J, Wei HB, *et al*. Slim-neck by GSConv: A better design paradigm of detector architectures for autonomous vehicles. arXiv:2206.02424v1, 2024.
20 Wang ZJ, He XW, Li Y, *et al*. EmbedFormer: Embedded depth-wise convolution layer for token mixing. Sensors, 2022, 22(24): 9854. [doi: 10.3390/s22249854]
21 Ma SL, Xu Y. MPDIoU: A loss for efficient and accurate bounding box regression. arXiv:2307.07662, 2023.
22 Lee J, Park S, Mo S, *et al*. Layer-adaptive sparsity for the magnitude-based pruning. Proceedings of the 9th International Conference on Learning Representations. OpenReview.net, 2021.
23 Du S, Zhang B, Zhang P, *et al*. An improved bounding box regression loss function based on CIoU loss for multi-scale object detection. Proceedings of the 2nd IEEE International Conference on Pattern Recognition and Machine Learning (PRML). Chengdu: IEEE, 2021. 92–98.
24 Rezatofighi H, Tsoi N, Gwak J, *et al*. Generalized intersection over union: A metric and a loss for bounding box regression. Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach: IEEE, 2019. 658–666.
25 Zheng Z, Wang P, Liu W, *et al*. Distance-IoU loss: Faster and better learning for bounding box regression. Proceedings of the 34th AAAI Conference on Artificial Intelligence. New York: AAAI, 2020. 12993–13000.
26 Zhang YF, Ren WQ, Zhang Z, *et al*. Focal and efficient IoU loss for accurate bounding box regression. Neurocomputing, 2022, 506: 146–157. [doi: 10.1016/j.neucom.2022.07.042]
27 Xiang N, Gong ZH, Xu Y, *et al*. Material-aware path aggregation network and shape decoupled SIoU for X-ray contraband detection. Electronics, 2023, 12(5): 1179. [doi: 10.3390/electronics12051179]
28 Ren S, He K, Girshick R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016, 39(6): 1137–1149.
29 Zhai SP, Shang DR, Wang SH, *et al*. DF-SSD: An improved SSD object detection algorithm based on densenet and feature fusion. IEEE Access, 2020, 8: 24344–24357. [doi: 10.1109/ACCESS.2020.2971026]
30 Zhou SX, Yang D, Zhang ZY, *et al*. Enhancing autonomous pavement crack detection: Optimizing YOLOv5s algorithm with advanced deep learning techniques. Measurement, 2025, 240: 115603. [doi: 10.1016/j.measurement.2024.115603]
31 Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Vancouver: IEEE, 2023. 7464–7475.
32 Wang CY, Yeh IH, Liao HYM. YOLOv9: Learning what you want to learn using programmable gradient information. Proceedings of the 18th European Conference on Computer Vision. Milan: Springer, 2024: 1–21.
33 唐鲁婷, 黄洪琼. 基于 YOLOv7 的轻量化水下目标检测算法. 电光与控制, 2024, 31(9): 92–97. [doi: 10.3969/j.issn.1671-637X.2024.09.016]

(校对责编: 王欣欣)