计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2023,32(11):120−130 [doi: 10.15888/j.cnki.csa.009279]

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

基于 MCA-YOLOv5s 的轻量化地铁站内行人检测①

孙同庆[1], 刘光杰[1], 唐　喆[1], 李佑文[2]

[1](南京信息工程大学 电子与信息工程学院, 南京 210044)
[2](南京国电南自轨道交通工程有限公司, 南京 210032)
通信作者: 刘光杰, E-mail: everglow_sun@163.com

摘　要: 随着智慧车站和云计算的迅速发展, 地铁站内大规模视频监控系统行人检测的部署愈发重要, 在客流监测、乘客引导和行为警示等方面发挥着人力不能及的重要作用. 在实际工程应用中, 受到计算资源有限以及多尺度多角度遮挡的困难样本带来错漏检的不利影响, 为此提出一种轻量化行人检测算法 MCA-YOLOv5s. 首先使用 MobileNetv3 代替 YOLOv5 主干网络, 实现网络模型轻量化处理, 并用 PConv 代替 MobileNetv3 网络中的 DWConv, 减少冗余计算和内存访问; 其次在特征融合阶段的 C3 模块中融入坐标注意力机制, 使模型更加关注行人的位置信息; 同时将损失函数 CIoU 替换为 Alpha IoU 以增加 High Loss 目标的权重和边界框的回归精度; 最后通过 FPGM 剪枝压缩改进后的网络模型, 提升模型加载和运行速度. 将改进后的模型部署在华为 Atlas 300 AI 加速卡中, 对地铁站内行人进行检测, 其平均精度达到 94.1%, 检测速度为 104.1 fps. 实际工程实践表明, 改进后的算法检测速度提升 71.8%, 节省了站内硬件部署资源, 更满足地铁大客流下的行人监测和管理的工程实际需求.

关键词: 行人检测; MCA-YOLOv5s; 轻量化; 注意力机制; 剪枝; 模型部署

引用格式: 孙同庆,刘光杰,唐喆,李佑文.基于 MCA-YOLOv5s 的轻量化地铁站内行人检测.计算机系统应用,2023,32(11):120–130. http://www.c-s-a.org.cn/1003-3254/9279.html

Lightweight Subway Pedestrian Detection Based on MCA-YOLOv5s

SUN Tong-Qing[1], LIU Guang-Jie[1], TANG Zhe[1], LI You-Wen[2]

[1](School of Electronic and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China)
[2](Nanjing Guodian Nanzi Rail Transit Engineering Co. Ltd., Nanjing 210032, China)

Abstract: With the rapid development of smart stations and cloud computing, the deployment of large-scale video surveillance systems for pedestrian detection in subway stations is becoming more and more important, which plays an important role in passenger flow monitoring, passenger guidance, and behavior warning. In practical engineering applications, a lightweight pedestrian detection algorithm MCA-YOLOv5s is proposed due to the adverse effects of limited computing resources and difficult samples caused by multi-scale and multi-angle occlusion. Firstly, MobileNetv3 replaces the YOLOv5 backbone network to achieve lightweight network model processing, and PConv replaces DWConv in the MobileNetv3 network to reduce redundant computation and memory access. Secondly, the coordinate attention mechanism is incorporated in the C3 module of the feature fusion stage to make the model pay more attention to pedestrian position information. At the same time, the loss function CIoU is replaced by Alpha IoU to increase the weight of the High Loss target and the regression accuracy of the bounding box. Finally, the improved network model is compressed by FPGM pruning to improve the loading and running speed of the model. The improved model is deployed

① 基金项目: 国家自然科学基金 (U21B2003); 江苏省产业前瞻与关键核心技术竞争项目 (BE2022075)
收稿时间: 2023-04-18; 修改时间: 2023-05-17; 采用时间: 2023-05-23; csa 在线出版时间: 2023-08-09
CNKI 网络首发时间: 2023-08-10

in Huawei Atlas 300 AI accelerator to detect pedestrians in subway stations. The average accuracy is 94.1%, and the detection speed is 104.1 fps. The actual engineering practice shows that the detection speed of the improved algorithm is increased by 71.8%, saving the hardware deployment resources in the station and meeting the actual engineering needs of pedestrian monitoring and management in subway stations with large passenger flow.

Key words: pedestrian detection; MCA-YOLOv5s; lightweight; attention mechanism; pruning; model deployment

如今地铁已经成为城市交通出行不可或缺的方式, 智慧车站的发展满足了人们对地铁智能化服务的需求. 随着近年来云计算、大数据、人工智能等技术的不断演进, 智慧车站中的视频监控系统逐渐以深度学习目标检测体系为架构, 不仅能在保障地铁安全上发挥越来越重要的作用, 还能改善地铁运营效率, 提升地铁服务质量, 增强地铁应急能力. 智能视频监控系统可以实时检测、分析和处理所监控的图像, 实现地铁人流量监测、行人异常行为分析、安全隐患警示等快速的响应. 地铁站内大规模视频监控系统的计算资源有限, 在成本约束下, 选择性能优越的国产算能卡部署目标检测网络更符合工程需求.

目标检测技术是为了解决目标视频或者图片中待检测物体的定位和分类问题, 它的性能好坏会直接影响到计算机视觉研究的后续进程. 随着神经网络在目标检测领域的迅速发展, 以深度学习为基础的目标检测算法成为主流, 其主要分为 two-stage 检测算法和 one-stage 检测算法两类. Two-stage 算法将图像候选区域和卷积神经网络进行融合, 使用 CNN 提前在输入图像的生成区域中创造一个目标分类器, 然后进行分类和特征提取, 常见的算法有 R-CNN[1]、Fast R-CNN[2]、Faster R-CNN 等. One-stage 算法主要包括 SSD[3]、YOLO 系列[4–7], 不需要生成候选框, 直接对初始的目标进行检测, 加快了图像检测速度, 满足大量图像检测的需求. 虽然当前很多目标检测算法精度很高, 但是部署在视频监控系统中并不能满足快速检测的需求, 而且行人之间遮挡较严重, 算法的误检率和漏检率也比较高.

针对密集场景下遮挡和多尺度行人检测精度低的问题, Zhang 等人提出一种跨通道的注意机制[8], 在 Faster R-CNN 架构中增加注意网解决不同的遮挡情况, 将身体不同部位与 CNN 通道进行关联, 提高网络对行人目标的关注度. 王明吉等人提出一种改进 YOLOv3 的行人检测方法[9], 通过搭建新一层的特征流在网络颈部进行特征融合, 增强网络的特征信息, 但该网络结构复杂度较高. 邓杰等人提出 Crowd-YOLO 算法[10], 将行人可见框和全身框进行结合并在空间注意力上增加频域通道注意力机制, 但在拥挤人群场景中, 该算法会生成特别多的锚框, 正负样本比例失衡.

单阶段检测算法中的 YOLOv5 网络模型综合性能较优异, 其网络深度和宽度可以自行调节. 根据参数量由小到大, 可以分为 YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x 这 4 种不同结构. 考虑到地铁站内大规模监控系统中的算能卡算力有限, 本文选择 YOLOv5s 为基础算法进行改进, 提出一种轻量化行人检测算法 MCA-YOLOv5s, 并采用 FPGM 剪枝进一步压缩网络模型, 然后部署在 Atlas 300 AI 加速卡中进行测试. MCA-YOLOv5s 首先分析采集到的地铁行人图像, 然后采用轻量级模块 MobileNetv3[11] 重构 YOLOv5s 的主干网络, 减少模型的体积和参数量, 实现网络模型轻量化处理, 并用 PConv[12] 代替深度可分离卷积中的 DWConv, 减少冗余计算和内存访问, 提高网络的计算速度. 针对行人目标多尺度问题, 将注意力模块 CA (coordinate attention)[13] 融入模型结构中的特征融合阶段的 C3 模块中, 使模型更加关注行人的位置信息, 提高对目标位置的定位能力, 同时弥补轻量化处理带来的精度损失. 最后将损失函数 CIoU 替换为 Alpha IoU[14] 以增加 High Loss 目标的权重和边界框的回归精度, 优化模型整体性能. 为了进一步提高算法部署在 Atlas 300 AI 加速卡上的检测速度, 对优化的网络模型进行剪枝, 压缩模型大小. 考虑到地铁场景内行人身体部位相互遮挡带来的不利影响, 本文选择遮挡范围较小的头部作为行人的检测目标. 实际工程实践表明, 改进后的算法相比于原始网络模型, 部署到加速卡设备中拥有更快的实时检测性能, 而且检测精确率也很高.

1 YOLOv5 网络介绍

本文所提出的算法在 YOLOv5s-6.0 版本的基础上

进行改进, 网络结构分为输入端、Backbone、Neck 和 Head. 输入端采用自适应图片缩放技术和 Mosaic 数据增强以及 K-means 算法处理输入的图像. Backbone 部分特征图首先经过第一层的卷积层 (Conv), 接着通过 4 层 C3 模块生成不同尺寸的特征图, 最后使用空间金字塔池化结构 (SPPF) 融合不同感受野的特征图. Neck 部分采用 FPN+PAN[15] 结合的路径聚合网络架构, 加强网络特征的融合能力. Head 检测层分别解码预测 3 种不同尺寸的特征图, 使用 NMS (non-maximum suppression) 非极大值抑制算法获取目标最优预测框, 输出预测框和类别位置信息.

2 改进 YOLOv5s 检测算法

复杂网络模型通常具有较大的参数量, 部署到设备中将面临占用空间大和检测速度慢的问题, 难以满足地铁大规模监控系统低延迟和快速响应的需求, 同时地铁站内设有多路监控视频设备, 需要考虑到工程成本的实际需求. 而且采用头部作为行人的检测目标虽然可以解决身体部位的遮挡问题, 但是行人头部依然存在多尺度、多角度、穿戴物遮挡等困难检测样本, 当前算法依然存在误检和漏检的问题. 为优化地铁站内大规模视频监控场景下的行人检测, 对 YOLOv5s 网络结构进行改进, 如图 1 所示, 具体方法为: (1) 为提高模型检测速度, 使用 MobileNetv3 网络替换主干网络, 并使用 PConv 替换 MobileNetv3 中的 DWConv, 减少网络内存的访问, 降低计算延迟. (2) 为了增强网络各层的特征融合能力, 在特征融合模块的 C3 层中融入 CA 注意力模块, 使模型更加关注目标的位置信息. (3) 为提高 High Loss 目标的权重和边界框的回归精度, 损失函数使用 Alpha IoU 替换 CIoU. (4) 为进一步压缩网络模型, 提高算法部署到设备中的推理速度, 使用 FPGM 剪枝去除不重要的卷积核和冗余的通道数.

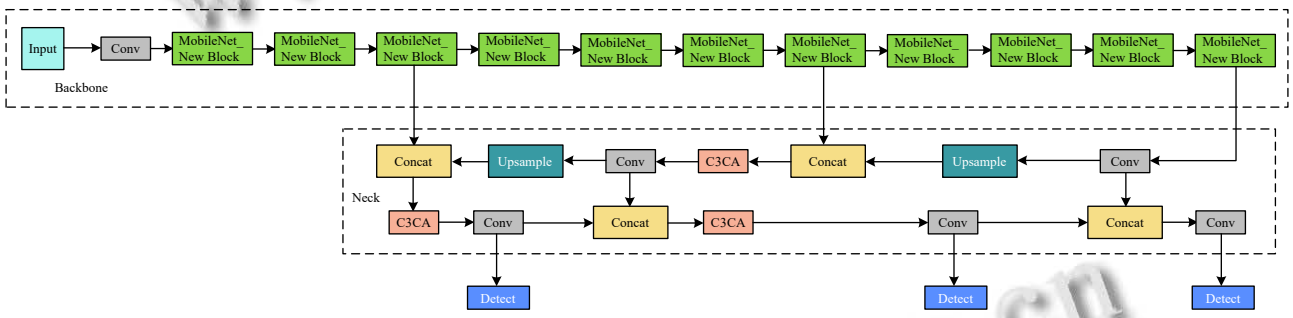


图 1 改进的 YOLOv5s 网络结构

2.1 MobileNetv3

MoblieNetv3 采用 MoblieNetv1[16] 和 MoblieNetv2[17] 中提出的深度可分离卷积和逆残差结构, 在此基础上更新 Block, 加入 SE (squeeze and excitation)[18] 模块, 利用 H-swish 代替 swish 激活函数, 进一步地提高了计算速度和模型性能.

MoblieNetv3 网络中的 Block 网络结构如图 2 所示, 主要包括了通道可分离卷积和 SE 通道注意力机制以及残差网络结构.

其核心是使用深度可分离卷积代替传统卷积层, 将传统卷积层拆分成逐通道卷积 (DWConv) 和逐点卷积 (PWConv). 逐通道卷积用于空间滤波, 将卷积核变为单通道, 每个卷积核处理一个通道. 逐点卷积用于特征生成, 不仅可以改变特征图的维度, 还可以在逐通道卷积生成的特征图通道上进行融合. 在逐通道卷积中, 每个卷积核的深度都为 1, 输出特征矩阵与输入特征矩阵深度相等. 逐点卷积则相当于卷积核大小为 1 的普通卷积, 一般与逐通道卷积搭配使用, 放在逐通道卷积后用来改变或者自定义特征矩阵的深度, 极大地减少了模型参数数量和计算量. 逐通道卷积和逐点卷积组合如图 3 所示.

MobileNetv3 通过 NAS 搜索全局网络结构, 分为 Large 和 Small 两种版本. 主要的不同在于经过卷积升维后的通道数量以及网络中的 Block 使用次数. 本文采用 MobileNetv3-Small 模型进行实验.

2.2 MobileNetv3 的改进

为了减少逐通道卷积中的冗余计算和内存访问的数量, 使用 PConv (partial convolution) 替换 DWConv, 更好的平衡检测延迟 (*Latency*) 和浮点运算 (*FLOPs*) 之间的联系, 它们之间的关系公式如下:

$$Latency = \frac{FLOPs}{FLOPS} \tag{1}$$

其中, *FLOPS* 表示每秒浮点运算的缩写, 度量有效的计算速度. PConv 可以缓和网络进行 *FLOPs* 时, 内存访问频繁造成 *FLOPS* 减小的副作用, 在降低 *FLOPs* 的同时优化 *FLOPS*, 尽可能多地使用设备的计算能力, 实现更好的低延迟效果. PConv 的工作原理如图 4 所示.

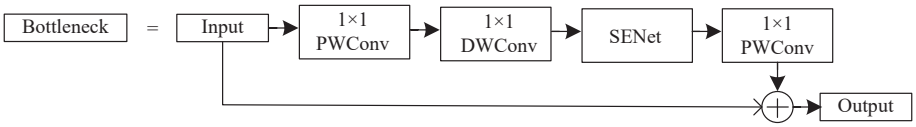


图 2 Block 网络结构

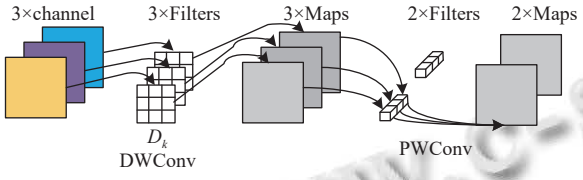


图 3 逐通道卷积和逐点卷积组合

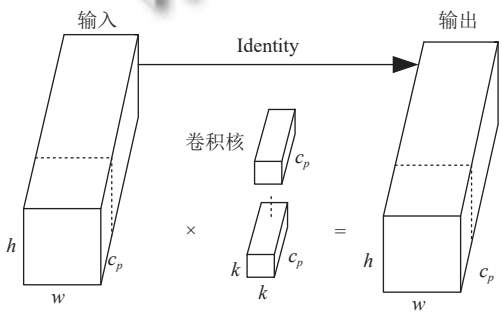


图 4 PConv 工作原理

在 PConv 结构中, 只需要使用部分输入图像的通道与标准卷积结合, 进行特征的提取, 其余通道保持不变. 如果内存访问是连续或者规则的, 使用第 1 个或最后一个连续的通道作为计算代表与整个特征图进行融合. PConv 的内存访问数量为:

$$h \times w \times 2c_p + k^2 \times c_p^2 \approx h \times w \times 2c_p \tag{2}$$

其中, h 和 w 分别为输入矩阵的宽高, c_p 是常规卷积作用的通道数, k 为卷积核的大小. 在实际实现过程中, c_p 一般设置为常规矩阵的 1/4, 其余通道数不参与计算. 而 DWConv 在降低 *FLOPs* 的同时, 会增大通道数来弥补精度的下降, 一般通道数会增大为常规卷积的 6 倍. 因此, PConv 相比与 DWConv 能够极大地减小内存访问的数量和计算冗余. PConv 层中没有简单地删除剩余的通道, 而是接着使用 PWConv 进行剩余通道特征的进一步提取. PWConv 可以提取所有通道特征信息流, 充分完整的捕获所有通道的特征信息. PConv 与 PWConv 组合成新的结构 New Block. 改进后的 MobileNetv3 网络结构如表 1 所示.

表 1 改进 MobileNetv3 网络结构

Input	Operator	Exp. size	SE	AF	Stide
$640^2 \times 3$	conv2d, 3×3	—	—	HS	2
$320^2 \times 8$	New Block, 3×3	16	√	RE	2
$160^2 \times 8$	New Block, 3×3	72	—	RE	2
$80^2 \times 16$	New Block, 3×3	88	—	RE	1
$80^2 \times 16$	New Block, 5×5	96	√	HS	2
$40^2 \times 24$	New Block, 5×5	240	√	HS	1
$40^2 \times 24$	New Block, 5×5	240	√	HS	1
$40^2 \times 24$	New Block, 5×5	120	√	HS	1
$40^2 \times 24$	New Block, 5×5	144	√	HS	1
$40^2 \times 24$	New Block, 5×5	288	√	HS	2
$20^2 \times 48$	New Block, 5×5	576	√	HS	1
$20^2 \times 48$	New Block, 5×5	576	√	HS	1

对于输入网络的图像, 统一调整为 640×640 的尺寸, 其中 New Block 表示将深度可分离卷积中的 DWConv 替换成 PConv, 网络中一共包含 11 个 New Block. HS 为 hard-swish 激活函数, RE 为 ReLU 激活函数.

2.3 融合 CA 注意力模块

CA 注意力机制相比其他注意力机制, 如 SENet、ECA[19]、CBAM[20], 同时考虑了通道维度和空间维度的信息, 并把位置信息嵌入到通道注意力当中, 有效地解决了空间维度存在的长距离依赖的问题, 而且还避免了大量的计算, 适合嵌入到轻量化网络当中. CA 注意力机制的具体流程如图 5 所示.

CA 注意力机制首先进行信息嵌入操作. 针对输入特征图的每一个通道信息, 利用尺寸为 (H, 1) 和 (1, W) 的卷积核对每个通道的水平方向和垂直方向进行全局平均池化操作, 聚合两个空间方向的特征, 输出具有方

向感知的特征图. 其中水平方向得到 $H\times1\times C$ 的信息特征图的公式为:

$$Z_c^h(h)=\frac{1}{W}\sum_{0\leqslant i\leqslant w}x_c(h,i),\ Z_c^h\in R^{C\times H\times1} \tag{3}$$

垂直方向得到 $1\times W\times C$ 的信息特征图公式为:

$$Z_c^w(w)=\frac{1}{H}\sum_{0\leqslant i\leqslant H}x_c(j,w),\ Z_c^w\in R^{C\times1\times W} \tag{4}$$

接着进行 CA 注意力生成操作. 沿着空间维度对生成的特征图Z_c^h和Z_c^w进行级联操作, 把水平方向和垂直方向的特征级联为全局特征. 再使用 1×1 的卷积和激活函数进行 F_1 变换. 然后在空间维度使用分片操作得到两个单独的注意力张量 g^h 和 g^w, 使用两个 1×1 的卷积将张量的通道数变换为和输入相同的通道数. F_1 变换和 g^h 以及 g^w 分别表示为:

$$f=\delta\left(F_1\left(\left[Z^h,Z^w\right]\right)\right),\ f\in R^{\frac{C}{r}\times1\times(H+W)} \tag{5}$$

$$g^h=\sigma\left(F_h\left(f^h\right)\right) \tag{6}$$

$$g^w=\sigma\left(F_w\left(f^w\right)\right) \tag{7}$$

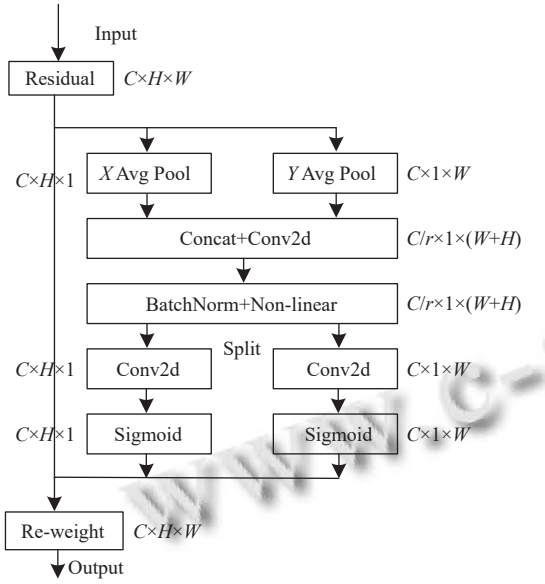


图 5 CA 注意力机制流程

最后使用广播变换把 g^h 和 g^w 拓展到 $C\times H\times W$ 维度, 对特征图进行矫正, 得到注意力特征. CA 注意力机制输出的最终表达式为:

$$y_c=x_c\times g^h\times g^w \tag{8}$$

CA 注意力机制融入到 C3 模块当中只会引入少量的参数, 而且不会增加网络的总层数, 运用在本文数据集中可以提高整个网络的性能, 具体实现如图 6 所示. 在 C3 模块主通道的 CBS 卷积之后接入 CA 注意力模块组合成 C3CA 模块. 其中, CBS 为带有 BN 和激活函数 SiLU 的卷积核大小为 1×1 的卷积, Bottleneck 为两个卷积核大小分别为 1×1 和 3×3 的 CBS 卷积融合而成.

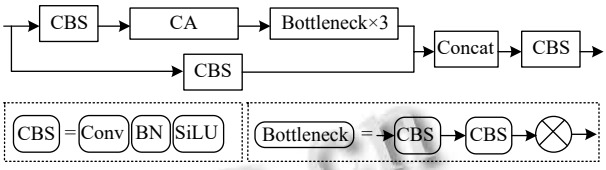


图 6 CA 融合 C3 模块

2.4 IoU Loss 改进

GIoU 损失函数考虑到真实框和预测框不相交的情况下, 梯度恒为 0 无法反向传播的问题, 提出了使用最小外接矩阵包含真实框和预测框[21]. 但是在对行人进行检测时, 经常会出现真实框与预测框完全重叠在一起的情况, 这会导致损失收敛速度变慢. CIoU 损失函数进一步地优化了检测框, 同时考虑到预测框与真实框的重叠面积、中心点聚集、长宽比之间的差异[22], 从而使得目标框的发散减少, 回归更加稳定, 同时加速损失的收敛速度. CIoU 的定义如下:

$$L_{\text{loss}}=1-IoU+\frac{\rho^2\left(b,b_{\text{gt}}\right)}{c^2}+\gamma u \tag{9}$$

$$u=\frac{4}{\pi^2}\left(\arctan\frac{w^{\text{gt}}}{h^{\text{gt}}}-\arctan\frac{w}{h}\right)^2 \tag{10}$$

其中, L_{loss} 为 CIoU 的损失; IoU、ρ、b_{gt} 和 b 以及 c 分别为真实框与预测框的交并比、中心点之间的距离、中心点以及并集部分对角线的长度; γ 为调节因子; u 是长宽比的相似系数, w 和 h 分别为预测框的高度与宽度, w^{gt} 和 h^{gt} 分别为真实框的高度和宽度.

但是该损失函数对每个目标的 IoU 都分配相同的权重, 在训练过程中梯度和损失无法自适应, 对于某些 High IoU 目标回归精度较差. 因此本文使用 Alpha IoU 对 CIoU 做进一步的优化, 自适应地调节 High IoU 目标的权重. 运用 Box-Cox 变换, Alpha IoU Loss 可表示为:

$$L_{\alpha\text{-}IoU}=\frac{1-IoU^{\alpha}}{\alpha},\ \alpha>0 \tag{11}$$

其中, 当 α 趋近于 0 时, $L_{\alpha\text{-}IoU}=-\log(IoU)$, 当 α 不趋近于 0 时, $L_{\alpha\text{-}IoU}=1-IoU^{\alpha}$.

将惩罚项加在上述公式中, 可以归纳出 CIoU 的改进损失函数 α-*CIoU*, 即:

$$L_{\alpha\text{-}CIoU}=1-IoU^{\alpha}+\frac{\rho^{2\alpha}\left(b,b_{\mathrm{gt}}\right)}{c^{2\alpha}}+(\gamma u)^{\alpha} \tag{12}$$

此时, 该函数可以概括出现有的一些其他的 *IoU* 损失类型. 当 α 大于 1 时, 分配给 High IoU 目标更多的损失权重, 有利于模型更加关注 High IoU 目标, 提高检测性能和边界框的回归精度. 同时模型训练更加灵活, 能够实现计算不同情况下的目标框回归精度, 在轻量化网络模型的结构中能发挥更好的作用. 经过实验对比, α 参数取 3 的情况下模型对 High IoU 目标的检测效果最好.

2.5 基于滤波器剪枝的 MCA-YOLOv5s

FPGM (filter pruning via geometric median)[23] 提出了一种新的剪枝方法, 通过计算几何中值的卷积神经网络过滤器剪枝. 突破范数准则不能总是满足范数偏差很大以及最小范数很小这两个要求的局限性, 实现“相对不重要”到“可替代性”的转变. FPGM 剪枝过程如图 7 所示, 通过修剪带有冗余信息的卷积核来压缩 CNN 模型, 具体为修剪同一层卷积核中几何中值附近的卷积核, 生成深度减小的特征图. 其相比基于范数准则的剪枝方法保留了更多的特征信息, 并且能够结合网络模型正常的训练过程, 免去额外的微调.

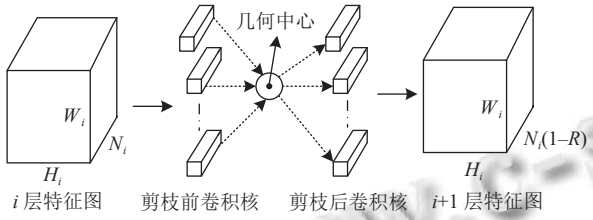


图 7 FPGM 剪枝过程

FPGM 结合 YOLOv5s 训练过程具体为: (1) YOLOv5s 网络根据梯度更新网络参数, 定义剪枝率参数 R, 以迭代的方式进行剪枝操作. (2) 在所有卷积层中, 计算当前层的几何中心点, 然后计算其他卷积核到几何中心点的欧式距离. (3) 对欧式距离进行排序, 裁剪掉后 $R\times N$ 个卷积核, N 为当前层的卷积核总数. (4) 模型进行下一轮的训练, 裁剪掉的卷积核权重修改为 0. (5) 模型的每轮训练不断地迭代 (2)–(4) 这 3 步. (6) 训练完成后开始调整网络结构, 去除参数为 0 的卷积核和卷积核中的冗余通道以及 BN 层参数的冗余数值, 得到最终剪枝后的模型.

3 实验结果及分析

3.1 实验环境

本实验的基本参数以及训练的深度学习环境如表 2 所示, 使用 PyTorch 深度学习框架部署网络模型.

表 2 实验环境配置表

名称	配置
操作系统	Windows 11
开发环境	CUDA 11.4
CPU	AMD Ryzen 7-5800H
显卡	NVIDIA GeForce RTX 3060 Laptop 6 GB
内存	16 GB
部署设备	Atlas 300 AI加速卡

3.2 实验数据集

本文采用的数据集来自地铁站内监控视频的抽帧图片. 现场捕获地铁站内乘客运动的视频, 使用 Python 导入 OpenCV 包对视频进行读取, 并使用 5 fps 的帧率进行抽帧, 通过对不同时间段视频的采集, 建立了丰富的站内乘客数据, 提供训练模型多样化的样本数据. 本文共 8 738 张图片, 其中包含密集和稀疏的行人场景以及多尺度的行人. 对获取的图片使用 LabelImg 软件进行手动标注, 获取 XML 格式的标签, 然后计算归一化的坐标以及归一化的宽高, 将标注文件转换为 YOLOv5s 算法适配的 txt 格式. 标注目标为行人的头部, 标签为“head”类, 以此来消除行人身体部分遮挡带来的影响. 实验按照 8:2 的比例划分图片, 训练集 6 166 张, 验证机 2 622 张, 一共包含 41 925 个标签.

3.3 训练参数及评价指标

将网络的输入图片大小统一裁剪为 640×640 的尺寸, 设置 batchsize 为 4, 优化器选择随机梯度下降法来更新网络参数, 学习率设置为 0.01, 使用余弦函数动态调整, Dropout 率设置为 0.05, 在训练集上最大训练次数 epoch 设置为 200, 前 3 个 epoch 用作热身训练, *IoU* 训练时的置信度阈值设置为 0.45.

为了更好地衡量地铁场景内行人检测算法的精确度和实时性能, 本研究设置的模型检测性能评价指标有: 精确率 (precision, P)、召回率 (recall, R)、平均精度 (average precision, AP)、浮点计算量 (giga floating-point operation per second, GFLOPs)、每秒传输帧数 (frames per second, FPS)、参数量 (Parameters).

精确率 P 定义为在所有的检测目标中预测正确的

概率, 计算公式为:

$$P=\frac{TP}{TP+FP} \tag{13}$$

召回率 R 定义为实际存在的图片中预测正确的概率, 计算公式为:

$$R=\frac{TP}{TP+FN} \tag{14}$$

平均精度 AP 定义为不同召回率下的平均精确度, 计算公式为:

$$AP=\int_0^1 p(R)dR \tag{15}$$

其中, TP (true positives) 表示正样本中被正确检测的例子; FP (false positives) 表示负样本被预测为正样本的例子; FN (false negatives) 表示正样本被错误预测为负样本的例子. AP 表示 $P(R)$ (precision-recall) 学习的模型检测性能的好坏.

GFLOPs 表示浮点运算次数, 可以用来衡量网络模型的复杂度. FPS 表示模型的检测速度, 即检测图片的数量和检测时间的比例. Parameters 表示模型中包含参数的数量.

3.4 实验结果分析

共进行 6 组对比实验和 1 组消融实验, 第 1 组 (见表 3) 为不同轻量化网络模型比较; 第 2 组 (见图 8) 为不同剪枝率下的模型性能比较; 第 3 组 (见图 9 和图 10) 为 YOLOv5s 和 MCA-YOLOv5s 的损失函数收敛性比较; 第 4 组 (见表 4) 为 YOLOv5s 加入不同改进模块的消融实验; 第 5 组 (见图 11) 为 MCA-YOLOv5s 与 YOLOv5s 的性能比较; 第 6 组 (见表 5) 为 MCA-YOLOv5s 与其他主流方法的比较; 第 7 组 (见图 12–图 14) 为改进算法的检测效果图比较.

表 3 轻量化效果对比

轻量化网络	*FLOPs* (G)	*P* (%)	*R* (%)	*AP* (%)	FPS
GhostNet	7.9	**94.9**	91.8	94.4	66.8
MobileNetv3	**1.9**	93.8	93.5	94.3	74.6
FasterNet	11.8	93.5	**94.2**	94.6	76.1
MobileNetv3 (PConv)	4.3	94.1	93.7	**94.8**	**78.7**

首先对改进后的 MobileNetv3 与原始的 MobileNetv3、FasterNet 和 GhostNet[24] 轻量化网络模型做对比试验, 在实验中均加入 CA 注意力模块和 Alpha IoU 损失, 结果如表 3 所示. FasterNet 虽然 *FLOPs* 较高, 但是其结构中的 PConv 相比于 DWConv 极大地降低了内存访问数量, 对空间特征的提取更为有效, 因此检测平均精度和速度都高于 GhostNet 和 MobileNetv3. 改进的 MobileNetv3 模型与 FasterNet 相比较, 检测精度和速度都有提高, 轻量化效果更加优异.

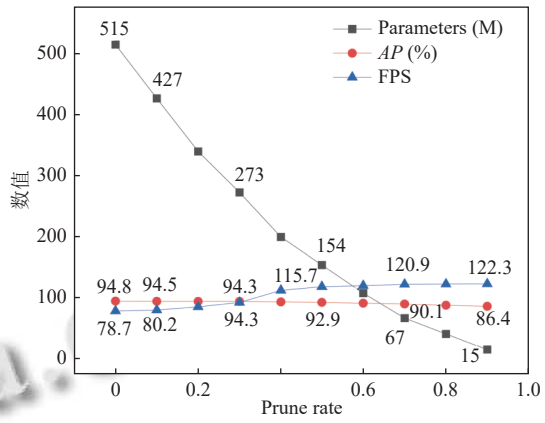


图 8 不同剪枝率下网络模型参数的变化

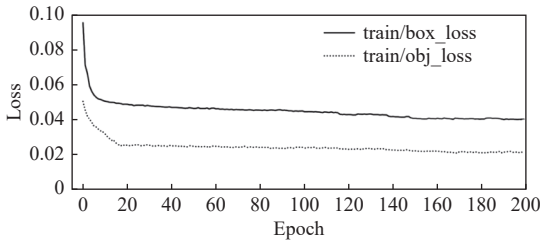


图 9 YOLOv5s 损失函数收敛曲线

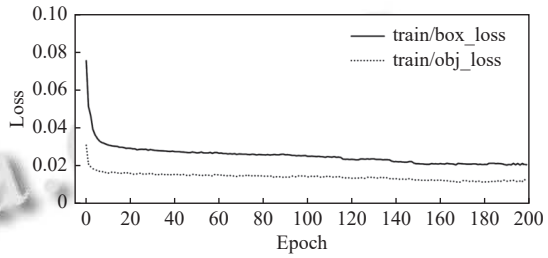


图 10 MCA-YOLOv5s 损失函数收敛曲线

表 4 MCA-YOLOv5s 的消融实验

算法	MobileNetv3	PConv	CA	Alpha IoU	剪枝 (%)	*FLOPs* (G)	*AP* (%)	FPS
YOLOv5s	—	—	—	—	—	15.8	94.6	53.2
A	√	—	—	—	—	1.9	92.5	75.6
B	√	√	—	—	—	4.3	93.1	81.8
C	√	√	√	—	—	3.9	94.7	78.2
D	√	√	√	√	—	3.9	**94.8**	78.7
E	√	√	√	√	40	1.4	94.2	**102.6**

对使用不同剪枝率的 MCA-YOLOv5s 模型的性能进行对比验证, 剪枝结果如图 8 所示. 图中展示了不同剪枝率下网络的 AP、FPS、Parameters 的指标变化. 剪枝率为 0 表示 MCA-YOLOv5s 网络模型. 本文的骨

干网络已经进行了轻量化处理, 为了维持模型的检测精度, 此处主要对特征融合阶段的多余通道进行剪枝. 从实验结果可以看出, 网络模型的参数量随着剪枝率的增加线性下降. 模型性能方面, 当剪枝率达到 0.4 后, 检测速度的增加变得缓慢, 检测精度开始加速降低, 当剪枝率为 0.9 时 *AP* 仅有 86.4%, 原因是因为剪枝后的网络结构如果太浅会严重影响精度. 在保证检测精度的情况下, 本文实验选择使用 0.4 的剪枝率.

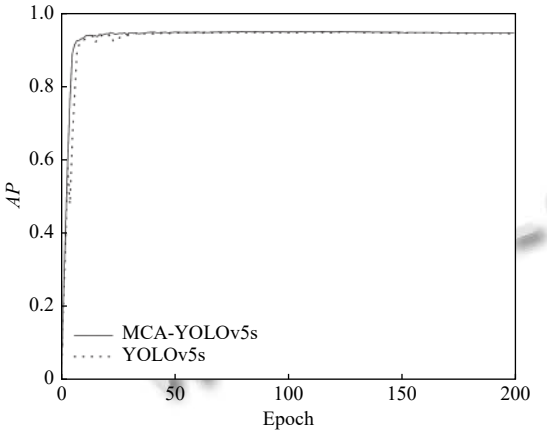


图 11 YOLOv5s 与改进 YOLOv5s 的 *AP* 对比

表 5 当前主流方法的对比试验

算法	*P* (%)	*R* (%)	*AP* (%)	FPS
Faster R-CNN	64.5	89.4	78.3	10.9
SSD	88.2	81.1	56.6	30.4
YOLOv4	92.5	80.4	80.2	18.6
YOLOXs	87.4	92.8	93.9	48.3
YOLOv7	93.7	**93.9**	**95.2**	42.1
YOLOv5s	93.5	92.8	94.6	53.2
MCA-YOLOv5s	**94.1**	93.7	94.8	**78.7**

图 12 稀疏行人场景

图 13 密集行人场景

对比使用 GIoU 损失函数的 YOLOv5s 与加入 Alpha IoU 损失函数的 MCA-YOLOv5s 的损失函数收敛情况, 得到定位损失 box_loss 与置信度损失 obj_loss 的收敛曲线, 如图 9 和图 10 所示.

通过对比图 9 和图 10 可以发现, MCA-YOLOv5s 的定位损失函数达到稳定状态时损失值稳定在 0.021 附近, 相比于 YOLOv5s 有明显的下降. 这是由于采用 Alpha IoU 损失函数可以有效降低边界框的损失值, 而且使得模型的置信度损失也有所下降, 提高网络整体的收敛效果.

为验证改进模型的性能, 本文进行消融实验. 实验结果如表 4 所示, 采用 *AP* 和 FPS 以及 GFLOPs 这 3 种指标评价模型. 首先在 YOLOv5s-6.0 的基础上使用改进的 MobileNetv3 主干网络, 测试轻量化模型的性能, 然后在轻量化模型的基础上依次加入 CA 注意力机制和 Alpha IoU 损失函数观察检测效果, 最后采用 40% 剪枝率的 FPGM 剪枝对改进后的网络模型进行压缩, 测试去除冗余通道后模型检测速度和精度的变化, 其中√表示加入此模块.

由实验结果可知: 在模型主干替换为 MobileNetv3 后模型的计算量大幅减少, 检测速度加快, 但带来了精度上的小幅降低; 把 MobileNetv3 中的 DWConv 替换为 PConv 后, 虽然参数量有所上升, 但是其降低了检测延迟, 因此检测速度有所加快, 行人目标的检测 *AP* 也有提升; 在此基础上加入 CA 注意力模块, 在小幅降低检测速度的同时, 行人目标检测 *AP* 提高了 1.6%, 弥补

了轻量化结构导致的部分精度损失, 然后再将 CIoU 损失函数替换为 Alpha IoU, 行人目标检测的精度和速度都得到了提升. 改进后的算法在保证了检测精度的同时, 大大减小了模型的计算量和复杂度, 加快了检测速度. 更进一步的使用 40% 剪枝率的剪枝压缩改进后的网络模型, 在 *AP* 下降了 0.6% 的情况下, FPS 增加 30.4%, 达到 102.6, 更加适合后续的算法模型部署.

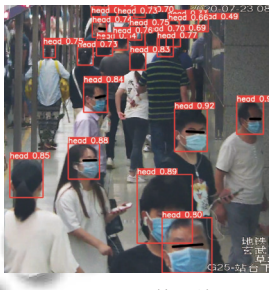
最后对比在相同数据集下 MCA-YOLOv5s 与原始 YOLOv5s 模型的性能, *AP* 对比图如图 11 所示, 改进的 YOLOv5s 算法在检测速度加快 48% 的情况下, 检测精度保持与原始 YOLOv5s 相近的水平, 因此综合性能更加符合地铁内大规模监控系统的实时检测需求.

为了验证本文 MCA-YOLOv5s 算法的效果与性能, 将算法与其他当前主流算法, 如 YOLOv5s、YOLOv7、YOLOv4、Faster R-CNN、SSD、YOLOXs 在相同的场景下进行对比实验, 结果如表 5 所示.

对比表 5 中的实验结果可知, Faster R-CNN 作为传统的 two-stage 算法, 在本文的数据集进行验证, 检测精度和速度都难以满足检测需求. YOLOv7 算法拥有最高的召回率和平均精度, 但是检测速度较慢, 难以部署在计算资源受限的设备平台. YOLOv4、SSD 算法以及 YOLOv5 算法同样为 one-stage 算法, 它们的算法平均准确率、精确率和召回率以及检测速度都比改进的 YOLOv5s 算法要低, 主要是因为改进 YOLOv5s 算法在特征融合阶段添加了注意力模块, 使得模型训练过程中更加关注重要的特征信息, 一定程度上增加了模型的检测效果. 同时它们的损失函数计算没有考虑到 High IoU 目标, 因此造成模型的性能和边界框回归精度的降低, 而改进的 YOLOv5s 算法引入 Alpha IoU 损失弥补了这一缺点, 同时改进的算法还使用了轻量化主干, 大大地降低了模型计算量, 在小幅降低检测精度的同时极大地提高了模型的检测速度. 综上所述, 本文提出的改进算法综合性能优于其他算法, 适合部署在地铁内大规模监控系统设备当中.

此外, 本文将改进后的算法 MCA-YOLOv5s 和原始的 YOLOv5s 算法在相同的地铁场景下针对不同密度和尺度的行人做对比试验, 结果如图 12–图 14 所示. 实验结果显示本文提出的改进算法在显著提高行人检测速度的同时, 在密集场景中对于遮挡的小目标检测仍具有较高的检测框置信度. 虽然剪枝后的模型相比于 YOLOv5s 检测精度有略微的下降, 但是在一定程度上改善了行人误检和漏检的问题, 并且在稀疏和多尺度行人场景中, 检测效果与 YOLOv5s 算法同样优异.

(a) 原始图像



(b) YOLOv5s 检测结果

(c) MCA-YOLOv5s 检测结果

(d) MCA-YOLOv5s (剪枝) 检测结果

图 14 多尺度行人场景

4 Atlas 300 AI 加速卡算法部署

4.1 软硬件融合

为验证本文算法在地铁大规模监控系统中的实用性和有效性, 将改进的 YOLOv5s 网络模型部署到华为 Atlas 300 AI 加速卡中进行测试. 该加速卡面向边缘侧和数据中心服务器场景, 包括低功耗的海思 Ascend 310 处理器实现快速高效的模型推理和图像识别及处理、2 个全新的达芬奇架构负责矩阵运算、ARM64 架构的鲲鹏 920CPU 负责部分算子调度和实现, 通过 Camera 模块融合外接的摄像头和加速模块, 在半精度下最高能实现 32 TOPS 的计算能力, 单卡最高能提供 64 TOPS INT8 的计算性能, 业界能够实现最高 64 路高清视频的实时分析能力, 且加速模块中集成了丰富的计算单元, 在程序运行时提供强大的神经网络计算能力, 实现高性能和低功耗的算法实现, 完全能够满足大规模视频监控场景设备部署的内存容量大、带宽高、低延迟的需求.

Atlas 300 AI 加速卡内有 4 个独立操作系统的 NPU 芯片, 实现地铁站内行人检测过程的加速推理. 业

务软件根据运行环境分为服务器侧和 NPU 侧, 在模型推理前处理阶段利用集成的 4 块 NPU 芯片提供算力, NPU 通过使 CNN 结构与权重结合成一体进而加速推理, 其他过程在服务器侧运行. 将深度学习开发框架训练生成的模型部署到 Atlas 300 AI 加速卡需要先转化为晟腾 AI 处理器支持的.om 离线模型, 转化主要包括 3 个步骤: 首先将 PyTorch 训练生成的.pth 权重文件转化为.onnx 文件, 并获取其中的网络结构和权值, 然后使用统一的中间图重新表达网络结构, 并基于结构中的权重对算子和模型进行编译. 最后采用序列化操作将.om 模型保存在外部文件当中, 离线模型生成步骤如图 15 所示.

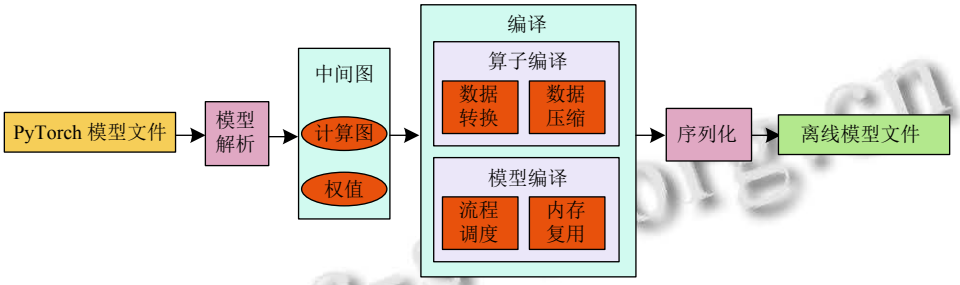


图 15 离线模型生成步骤

4.2 性能测试验证

在 Atlas 300 AI 加速卡上分别移植不同的离线模型, 并在晟腾 AI 处理器上输入相同的地铁内行人数据集, 观察不同网络模型的性能, 对比实验结果如表 6 所示.

表 6 Atlas 300 AI 加速卡运行结果对比

算法	*P* (%)	*R* (%)	*AP* (%)	FPS
YOLOv5s	93.6	92.1	**94.5**	60.6
MCA-YOLOv5s	**93.9**	**93.5**	94.3	80.9
MCA-YOLOv5s (剪枝)	93.1	91.9	94.1	**104.1**

由表 6 可知, YOLOv5s 算法检测平均精度最高, 但是因为其网络结构较深, 每秒传输帧数比其他网络慢很多. MCA-YOLOv5s 算法在检测精度下降 0.2% 的情况下, 检测速度提高 33.5%. 经过剪枝后, 进一步测试最终网络模型的检测效果, 实验结果显示召回率为 91.9%, 平均精度为 94.1%, FPS 达到 104.1, 相比于原始 YOLOv5s 网络模型, 在检测精度只降低了 0.4% 的情况下, 加快了 71.8% 的检测速度.

图 16 为经过剪枝的 MCA-YOLOv5s 和 YOLOv5s 模型分别部署在 Atlas 300 AI 加速卡中的检测效果, 可以看出两个模型的检测精度和检测框的定位几乎没有区别. 在地铁大规模监控系统设备部署的成本约束下, 改进后的网络模型部署在 Atlas 300 AI 加速卡后能够同时检测更多路视频, 而且检测精度也完全符合需求, 更加适合实际的工程应用.

5 结论与展望

本文针对地铁场景内大规模监控系统设备计算资源有限和行人多尺度、多角度、遮挡等问题, 提出了一种轻量化检测算法 MCA-YOLOv5s, 在保证高检测精度的前提下, 拥有更快的检测速度. 实际工程实践表明, 将改进后的模型剪枝后部署在 Atlas 300 AI 加速卡上, 相比于 YOLOv5s 算法, 检测速度提升 71.8%, 达到 104.1 fps, 而且检测性能几乎没有下降, 其能够同时实现地铁站内更多路视频的实时检测, 降低了实际工程中成本的投入, 而且更易于实现检测后行人追踪、行为警示、客流引导等进一步功能的实现.

(a) YOLOv5s 检测结果 (b) 剪枝 MCA-YOLOv5s 检测结果

图 16 Atlas 300 AI 加速卡检测效果对比

目前模型对严重遮挡的小尺寸行人的误检率和漏检率还有待进一步优化. 之后的研究在维持当前检测速度的前提下, 继续优化算法的网络结构, 降低行人检测的误检率和漏检率.

参考文献

1 Girshick R, Donahue J, Darrell T, *et al*. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus: IEEE, 2014. 580–587. [doi: 10.1109/CVPR.2014.81]

2 Ren SQ, He KM, Girshick R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137–1149. [doi: 10.1109/TPAMI.2016.2577031]

3 Liu W, Anguelov D, Erhan D, *et al*. SSD: Single shot multibox detector. Proceedings of the 14th European Conference on Computer Vision. Amsterdam: Springer, 2016. 21–37. [doi: 10.1007/978-3-319-46448-0_2]

4 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.

5 Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu: IEEE, 2017. 6517–6525.

6 Redmon J, Farhadi A. YOLOv3: An incremental improvement. arXiv:1804.02767, 2018.

7 Bochkovskiy A, Wang CY, Liao HYM. YOLOv4: Optimal speed and accuracy of object detection. arXiv:2004.10934, 2020.

8 Zhang SS, Yang J, Schiele B. Occluded pedestrian detection through guided attention in CNNs. Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018. 6995–7003.

9 王明吉, 张政, 倪子颜, 等. 基于 YOLOv3 的视频场景行人检测研究. 通信技术, 2021, 54(6): 1378–1383. [doi: 10.3969/j.issn.1002-0802.2021.06.014]

10 邓杰, 万旺根. 基于改进 YOLOv3 的密集行人检测. 电子测量技术, 2021, 44(11): 90–95. [doi: 10.19651/j.cnki.emt.2106129]

11 Howard A, Sandler M, Chen B, *et al*. Searching for MobileNetV3. Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul: IEEE, 2019. 1314–1324.

12 Chen JR, Kao SH, He H, *et al*. Run, don't walk: Chasing higher flops for faster neural networks. arXiv:2303.03667, 2023.

13 Hou QB, Zhou DQ, Feng JS. Coordinate attention for efficient mobile network design. Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Nashville: IEEE, 2021. 13708–13717.

14 He JB, Erfani S, Ma XJ, *et al*. Alpha-IoU: A family of power intersection over union losses for bounding box regression. Proceedings of the 35th Conference on Neural Information Processing Systems. 2021. 20230–20242.

15 Liu S, Qi L, Qin HF, *et al*. Path aggregation network for instance segmentation. Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018. 8759–8768. [doi: 10.1109/CVPR.2018.00913]

16 Howard AG, Zhu ML, Chen B, *et al*. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861, 2017.

17 Sandler M, Howard A, Zhu ML, *et al*. MobileNetv2: Inverted residuals and linear bottlenecks. Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018. 4510–4520.

18 Hu J, Shen L, Sun G. Squeeze-and-excitation networks. Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018. 7132–7141.

19 Wang QL, Wu BG, Zhu PF, *et al*. ECA-Net: Efficient channel attention for deep convolutional neural networks. Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle: IEEE, 2020. 11531–11539.

20 Woo S, Park J, Lee JY, *et al*. CBAM: Convolutional block attention module. Proceedings of the 15th European Conference on Computer Vision. Munich: Springer, 2018. 3–19. [doi: 10.1007/978-3-030-01234-2_1]

21 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.

22 Zheng ZH, Wang P, Ren DW, *et al*. Enhancing geometric factors in model learning and inference for object detection and instance segmentation. IEEE Transactions on Cybernetics, 2022, 52(8): 8574–8586. [doi: 10.1109/TCYB.2021.3095305]

23 He Y, Liu P, Wang ZW, *et al*. Filter pruning via geometric median for deep convolutional neural networks acceleration. Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach: IEEE, 2019. 4335–4344.

24 Han K, Wang YH, Tian Q, *et al*. GhostNet: More features from cheap operations. Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020. 1577–1586.

(校对责编: 牛欣悦)