汕頭大學
SHANTOU UNIVERSITY

硕 士 学 位 论 文

题　　目　基于深度学习的轻量化小目标检测算法研究

英文题目　Research of Lightweight Small Object Detection Algorithm Based on Deep Learning

姓　　名　胡星晨　　学　　号　111809020

所在学院　工学院　　导师姓名　范衡

专　　业　信息与通信工程

入学日期　2018 年 9 月　　答辩日期　2021 年 6 月

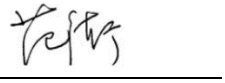
学位论文原创性声明

本论文是我个人在导师指导下进行的工作研究及取得的研究成果。论文中除了特别加以标注和致谢的地方外，不包含其他人或其它机构已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在论文中以明确方式标明。本人完全意识到本声明的法律责任由本人承担。

作者签名：  日期：2021 年 5 月 26 日

学位论文使用授权声明

本人授权汕头大学保存本学位论文的电子和纸质文档，允许论文被查阅和借阅；学校可将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它复制手段保存和汇编论文；学校可以向国家有关部门或机构送交论文并授权其保存、借阅或上网公布本学位论文的全部或部分内容。对于保密的论文，按照保密的有关规定和程序处理。

作者签名：  导师签名： 

日期：2021 年 5 月 26 日 日期：2021 年 5 月 26 日

硕 士 学 位 论 文

论文中文题目： 基于深度学习的轻量化小目标检测算法研究

论文英文题名： Research of Lightweight Small Object Detection Algorithm Based on Deep Learning

指 导 教 师 ： 范衡

申 请 人 ： 胡星晨

论文答辩委员会成员

主席：	陈耀文	教授	(汕头大学)
委员：	杨金耀	教授级高级工程师	(汕头市超声仪器研究所有限公司)
	庄哲民	教授	(汕头大学)
	熊智	教授	(汕头大学)
	邢阳辉	副教授	(汕头大学)

摘要

随着深度学习技术的发展，深度学习的研究和应用成功解决了很多学术和工程问题。在目标检测任务中，从两阶段到单阶段，再到无锚框的检测算法，平均精度不断提高。但是也存在两个问题，一是小目标的平均精度较低；二是随着平均精度的增加，运算量和参数量也越来越大，导致模型尺寸较大，检测速度缓慢。

这些问题限制了目标检测算法在移动机器人、自动驾驶汽车、智能手机等移动平台与嵌入式设备上的落地应用。

为了解决这些问题，推动目标检测算法的落地应用。本文提出一种面向小目标检测的轻量化多尺度的目标检测新算法 LMDet-S（Lightweight Multiscale Detector for Small Object）。该算法主要分为两个部分，一部分是轻量化的特征融合网络 Lite-PAN，另一部分是轻量化的检测头 LAF-Head。

为验证算法性能，本文进行了大量实验。首先，在标准数据集 COCO 上的实验表明，与前沿算法 YOLOv4-tiny 相比，在相同分辨率的情况下，本文算法浮点运算量下降了 42%，参数量减少了 34%，同时平均精度提高了 8.6%，小目标检测精度提高了 2%。

然后，本文在小目标数据集 TinyPerson 上进行了应用，在“图像切分”与“尺度匹配”策略的基础上，本文算法在单尺度训练、单模型的情况下取得了较好的检测效果。

最后，在 vivo-iQOO-Z1x 手机上进行了部署实验。经实机测试，本文以 ShuffleNetV2 为骨干网络的算法速度可达到 30FPS，满足实时的标准。

关键词：深度学习；小目标检测；轻量化神经网络

Abstract

With the development of deep learning technology, the research and application of deep learning have successfully solved many academic and engineering problems. In the object detection task, from two-stage to one-stage, and then to anchor-free detectors, the average precision is constantly improved. However, there are still two drawbacks. Firstly, the average precision of small object is low. Secondly, with the increase of the average precision, the number of operations and parameters are also increasing, result in a larger model size and a slower detection speed.

These problems limit the application of object detection algorithm in mobile robots, self- driving vehicles, smart phones and other mobile platforms or embedded devices.

To solve these problems and promote the application of object detectors. In this paper, a new algorithm named LMDet-S (Lightweight Multi-scale Detector for Small Object) was proposed. The algorithm is mainly divided into two parts, one is the Lightweight feature fusion network Lite-PAN, the other is the Lightweight detection Head LAF-Head.

To verify the performance of the algorithm, many experiments are carried out. Firstly, the experiments on the standard dataset COCO show that, compared with the SOTA algorithm YOLOv4-Tiny, the FLOPs of the proposed algorithm are reduced by 42% and the params is reduced by 34% at the same resolution. Meanwhile, the mAP of the proposed algorithm is improved by 8.6%, and the APs is improved by 2%.

Then we applied our algorithm on Tiny Benchmark such as TinyPerson. Based on the “image cutting” and “scale match” strategies, we use single scale training and single model, our algorithm achieves better performance.

Finally, the deployment experiment is conducted on a vivo-iQOO-Z1x mobile phone. The real machine test shows that the speed of our algorithm based on ShuffleNetV2 backbone can achieve 30FPS and satisfy the real-time standard.

Keywords: Deep Learning, Small Object Detection, Lightweight Neural Network

目　录

第 1 章 绪论

1.1 研究背景与意义

1.1.1 小目标检测算法的研究背景与意义

目标检测（Object Detection）就是在图像中找到感兴趣的目标，并给出它们的类别，以及它们在图像中的具体位置。目标检测是计算机视觉领域的重要研究方向之一，是智能机器人，自动驾驶汽车，智能安防等实际应用的重要支撑技术。随着智能科技的发展，目标检测技术在这些领域的应用取得了不错的成效。但是，在实际场景中存在着许多难以检测的目标，比如自动驾驶场景下远景的行人和车辆、安防监控中距离摄像头较远的行人，以及无人机航拍图像中的众多小目标等。这些情况中的目标相对尺寸较小，像素点少，分辨率较低，对于这些小目标，现阶段的目标检测算法检测效果较差。

表 1–1 常用目标检测算法在 COCO 数据集上的平均精度（单位：%）

检测算法	骨干网络	分辨率	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv2[1]	DarkNet-19	288	21.6	44.0	19.2	**5.0**	22.4	35.5
YOLOv3[2]	DarkNet-53	320	28.2	51.5	29.7	**11.9**	30.6	43.4
SSD[3]	VGG-16	300	25.1	43.1	25.8	**6.6**	25.9	41.4

表 1–1 列举了常用的目标检测算法在 COCO[4]数据集上的表现。COCO 数据集是由微软公司标注的大型的自然场景图像数据集，其中小目标占比 41.43%，是目前学术界采用的标准数据集。COCO 数据集中将像素面积小于 32×32 的物体称为小目标，将像素面积大于 32×32 小于 96×96 的物体称为中目标，将像素面积大于 96×96 的物体称为大目标，它们的检测平均准确率分别用 AP_S，AP_M 和 AP_L 来表示。从表 1-1 中可以发现在该数据集上大中型目标的检测效果良好，但是小目标的检测效果不理想，远不能应用于实际生产生活中。

提高小目标的检测精度成为目标检测中一个亟待解决的问题。小目标检测能力的提升可以让自动驾驶汽车更早的看到远处的行人和路标，提高行车安全性，可以使机器视觉检测瑕疵更准确，提升产品质量等。因此，小目标检测在理论研究和实际应用方面都具有重要意义。

1.1.2 神经网络轻量化的研究背景与意义

深度学习技术自 2012 年以来飞速发展，成功解决了很多学术和工程问题，在图像识别、自然语言处理等领域部分算法甚至超越了人类的学习能力。虽然深度学习算法有良好的精度，但深度神经网络计算复杂度高，模型参数量大（如表 1–2 所示），一般在服务器端部署，限制了其在计算能力以及存储资源有限的移动设备上的部署。

表 1–2 常用目标检测算法的参数量与浮点运算量

网络名称	骨干网络	输入分辨率	参数量	FLOPs
YOLOv3[2]	Darknet-53	608*608	62.5M	65.9B
RetinaNet[5]	ResNet50-FPN	640*640	34M	97B
Mask R-CNN[6]	ResNet101-FPN	800*800	44.4M	149B

让深度学习技术从云端扩展到边缘和终端设备，在各行业落地应用是目前学术界和工业界研究的热点，被称为边缘人工智能，在工业制造、商业零售、健康医疗、智慧教育和智慧社区等领域有着广泛的需求。

对于目标检测而言，在移动机器人、自动驾驶汽车、智能手机以及工业检测等移动设备和场景中有着很大的需求。通过神经网络轻量化技术，设计出低功耗，低存储以及低延迟的神经网络模型，可以让深度学习模型在更多的边缘设备上得到部署，有利于解决云端部署存在的网络延迟、安全性与隐私性等问题。

针对深度学习落地部署的需求，为提升小目标检测的效果，促进目标检测算法在边缘端落地应用，本课题应运而生。本文设计的轻量化的小目标检测算法，在保证检测性能的同时，降低了浮点运算量，减小了模型尺寸，将基于深度学习的网络模型应用于资源受限的移动嵌入式设备。本课题不仅在理论研究方面具有重要意义，而且在现实场景中也有着广阔的应用空间。

1.2 国内外研究现状

1.2.1 目标检测算法研究现状

由于各类物体具有不同的形状、姿态和尺寸，再加上成像时的光照、遮挡等干扰，目标检测一直是计算机视觉领域最具挑战性的问题之一。在深度学习出现以前的目标检测算法统称为传统的目标检测算法，传统的目标检测算法主要步骤如图 1–1 所示，包括区域选择、特征提取和分类器设计等步骤。

图 1–1 基于传统方法的目标检测算法流程图

对于静态图像，候选区域选择一般采用滑动窗口的方法，设置不同尺度、不同长宽比的窗口在图像上滑动，遍历目标在图像上所有可能出现的位置，依次得到候选区域。但这种方法计算量很大，为降低计算复杂度 2013 年 Uijlings 等人提出了选择性搜索[7]（Selective Search）算法，通过对区域相似度的计算，以及相似合并的方法找到候选区域，这种方法在基于深度学习的目标检测算法 Fast R-CNN[8]中仍有使用。对于动态图像，候选区域选择一般采用背景差法[9]和光流法[10]。

特征提取对提高目标检测的精度起到了重要的作用。传统方法中，一般根据颜色、纹理、几何形状和梯度，结合具体任务或数据集来手工设计特征。比较典型的有，用于行人检测的方向梯度直方图特征[11]（Histogram of Oriented Gradient，HOG），用于人脸检测的 Haar 特征[12]等。

分类器的作用是对提取到的特征进一步整理，判断提取到的特征是否和想要检测的目标的特征一致或者类似。在传统的目标检测算法中，常用的分类器有 AdaBoost[13]和 SVM 等。

经过上述步骤得到的预测框，很可能出现多个重叠度较高的预测框分配给一个真值框的情况。针对这种情况，2006 年 Neubeck 等人提出了非极大值抑制（Non-Maximum Suppression，NMS）算法[14]。采用 NMS 进行后处理可以去掉置信度较低的预测框。为了避免将相邻两个目标对应的预测框误认为同属于一个目标的情况，2017 年 Bodla 等人又提出了 soft-NMS[15]的方法。NMS 的方法一直沿用至今，现在的基于深度学习的目标检测算法中也有用到。

传统的目标检测算法中，手工设计的特征泛化性不强，鲁棒性较差，不能很好的应对复杂的检测任务。2012 年 AlexNet[16]在 ImageNet 数据集[17]上的分类任务中取得了巨大成功。自此深度卷积神经网络在计算机视觉领域被广泛应用，并逐渐取代了传统手工设计特征的方法。目前在目标检测任务中，基于深度学习的算法，已经成为主流的算法，包括两阶段的 R-CNN[18]系列[6, 8, 18, 20, 22, 23]和单阶段的 YOLO 系列[1,2,24,25,53]，SSD 系列[3,33]，以及无锚框的目标检测算法等，如图 1–2 所示。

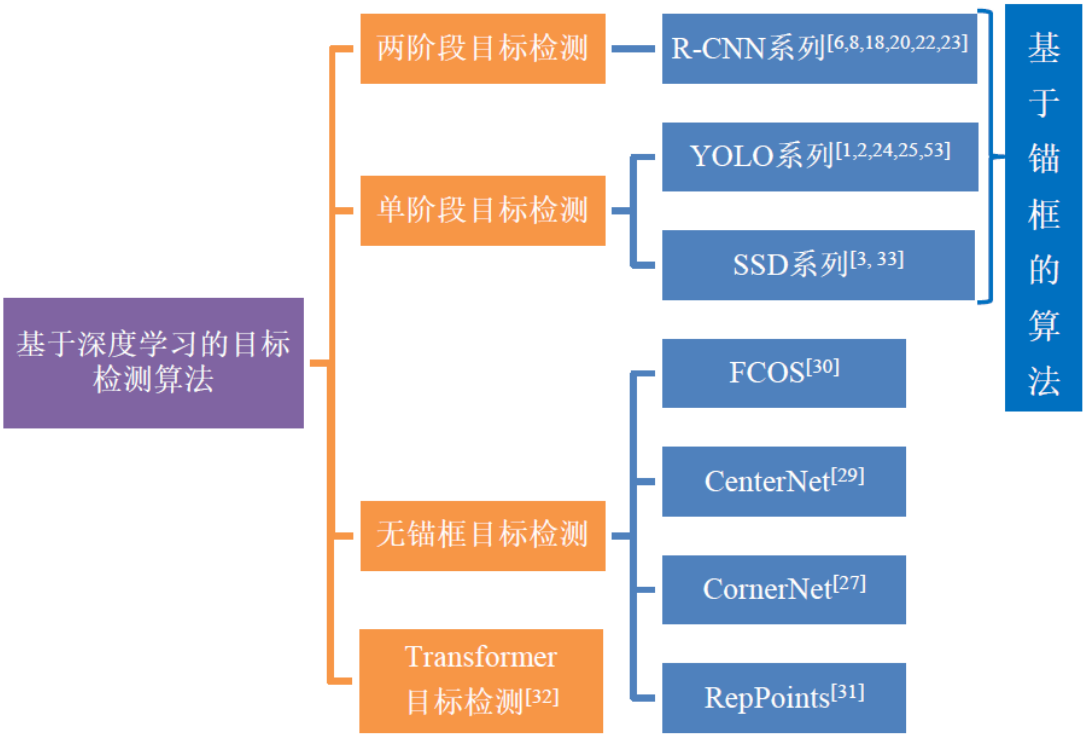


图 1–2 基于深度学习的目标检测算法总结

(1) 两阶段目标检测算法

2014 年 Girshick 等人提出的 R-CNN 算法[18]第一次将深度学习引入到目标检测任务中。该算法首先使用“选择性搜索”提取候选框，然后将每一个候选框都缩放到相同的尺寸，再利用在 ImageNet 上预训练过的 AlexNet 网络提取每一个候选框的特征，最后采用 SVM 分类器对这些特征进行分类。该方法步骤繁多，并且不是端到端的训练，因此许多学者对其进行了改进。

在 R-CNN 的基础上，何凯明等提出了 SPPNet[19]，在卷积层和全连接层之间添加了一个空间金字塔池化（Spatial Pyramid Pooling，SPP）层。这个多分辨率的池化层，可以接受不同尺寸的输入特征图，将特征图归一化为相同长度的输出向量，再传入到全连接层，确保了全连接层的兼容性。与 R-CNN 先提取候选框，再进行缩放，最后输入到卷积神经网络不同，SPPNet 只需对原图进行一次卷积操作，得到特征图，然后找到每一个候选框在特征图上对应的区域，将这个区域作为每个候选框的卷积特征输入到后面的网络中，进行后续的计算。

借鉴 SPP-Net 的思路，采用共享特征图的思想，2015 年 Girshick 等人提出了 Fast R-CNN[8]，使用 ROI Pooling 层提取不同尺寸的候选区域的特征，同时加入了候选区域映射的功能，使得网络能够进行反向传播，解决了 SPP-Net 整体训练的问题。另外，该网络在分类任务中使用 Softmax 取代了 SVM 进行多分类，在回归任务中使用了

Smooth L1 Loss 对候选框进一步调整位置，并且回归任务和分类任务并列执行，实现了多任务学习，大大提高了检测速度。但是该网络仍不能实现端到端的训练。

2015 年任少卿等人提出了 Faster R-CNN[20]，采用区域候选网络（Region Proposal Network，RPN）替代了选择性搜索的方法进行候选区域的选取，并引入了锚框（Anchor）的机制，至此整个目标检测网络全部由卷积神经网络实现，Faster R-CNN 成为第一个端到端的目标检测网络。

2017 年，Tsung-Yi Lin 等人又在 Faster R-CNN 基础上提出了特征金字塔网络[21]（Feature Pyramid Network，FPN），替换了原来的 RPN，将语义信息丰富的高层特征和位置信息丰富的低层特征进行融合，生成不同尺度的特征图，然后在不同尺度的特征图上预测相应尺度的目标，该方法对于检测不同尺度的目标效果良好。

2018 年 Zhaowei Cai 等人提出了 Cascade R-CNN[22]。这是一个多阶段的 R-CNN 模型，在不同阶段，采用不同的 IoU 阈值来确定正负样本，通过级联的检测网络优化检测结果。Libra R-CNN[23]是由浙江大学和香港中文大学联合提出目标检测模型。针对候选区域选取、特征融合、多任务损失函数收敛中的不平衡问题，提出三个改进方法，分别是平衡 IoU 采样, 平衡特征金字塔，以及平衡 L1 损失。不用对网络进行较大改造，在不需要增加计算成本的前提下，mAP 提高了 2 个百分点。

(2) 单阶段目标检测算法

2016 年 Redmon 等提出了 YOLO[24]，将图像划分为若干个网格，利用回归的思想，若一个目标真值框的中心点落在某个网格上，那么该网格就负责检测这个目标。该方法不用生成候选区域，提高了预测速度，但定位精度较低。之后的 YOLOv2[1]引入了 Faster R-CNN 中的锚框机制，在每个网格周围生成若干个不同大小、不同宽高比的锚框，提高了检测的精度。2018 年，YOLOv3[2]借鉴了 ResNet 的残差结构设计了 DarkNet53 骨干网络，并采用了多尺度特征融合与多尺度预测的方法进一步提高了精准度。2020 年 4 月，在 YOLO 系列原作者 Redmon 宣布退出计算机视觉领域以后，Bochkovskiy 继承了 YOLO 系列的思想和理念，提出了 YOLOv4[25]，并得到了原作者的认可。YOLOv4 主要对 YOLOv3 进行了一系列优化，比如采用了 Mosaic 数据增强，设计了 CSP-Darknet-53 骨干网络，以及采用了 Mish 激活函数等。同年 6 月，粒子物理与人工智能初创公司 Ultralytics 提出了 YOLOv5[26]，这是 YOLO 家族中，第一个采用 PyTorch 而非 Darknet 框架的原生版本。YOLOv5 包含 YOLOv5-s，YOLOv5-m，YOLOv5-l，YOLOv5-x 四个版本，多种网络结构使其使用起来更加灵活，可以根据不同的项目需求，选择不同的版本。

2016 年，在 YOLO 诞生之后不久，另一个单阶段的目标检测框架 SSD[3] 被 Liu Wei 等人提出，并且检测精度更高，平均精度超过了 Faster R-CNN。SSD 以 VGG-16

为骨干网络，首先将 VGG-16 中的全连接层 fc6 和 fc7 换成了卷积层，去掉了 fc8，并增加了 conv8, conv9, conv10, conv11 这 4 个额外的卷积层；然后把 pool5 中步长为 2 的 2× 2的卷积核，改为了步长为 1 的 3× 3卷积核；最后提取 conv4_3, conv7，conv8_2，conv9_2，conv10_2，conv11_2 这 6 个卷积层输出的特征图。在这个 6 个特征图中的每个特征图上设置不同尺寸的先验框（default box），在高层特征图（比如 conv11_2）上设置较大的先验框，预测较大的物体；在低层特征图（比如 conv4_3）上设置较小的先验框，预测较小的物体。有效利用不同卷积层的感受野，实现了分层多尺度目标的检测。

SSD 虽说用到了不同卷积层的特征，但只在不同卷积层上单独预测，没有进行特征融合。对此，RetinaNet[5]基于特征金字塔网络，对来自不同层的特征图进行特征融合，并提出了 Focal Loss，解决了正负样本不均衡的问题。

(3) 无锚框的目标检测算法

以上介绍的两阶段和单阶段目标检测算法，除了第一代 YOLO 算法以外，其余都是基于锚框（Anchor）的算法。这种算法在每个网格上生成多个不同尺度、不同宽高比的锚框，然后再在锚框的基础上进行位置的微调以生成预测框。由于锚框是通过对数据集统计聚类得到的，采用锚框可以使模型更容易收敛。但是，这类方法也存在一些问题，比如鲁棒性与泛化性较差，检测结果对超参数敏感，锚框数量众多且计算复杂，以及正负样本不均衡等。

为了解决这些问题，出现了无锚框（Anchor-Free）的目标检测方法。2018 年 Law 等提出了 CornerNet[27]，检测目标框左上角的点和右下角的点，将它们组合起来形成检测框，并引入了角点池化（Corner Pooling），帮助网络更好的定位角点。之所以选择角点，是因为角点相对于中心点更容易训练。比如左上角点只与检测框的两条边相关，而中心点与四条边都相关。不过这种方法需要更复杂的后处理来对属于同一实例的角点对进行分组。CornerNet 及其后续的变体 CornerNet-Lite[28]均属于基于关键点的检测算法。2019 年 Duan 等提出了 CenterNet[29]，只把中心点作为正样本，通过高斯分布生成监督信息，直接检测物体中心点和尺寸。这种方法中，每个物体只有一个正样本点，不需要 NMS，有利于加速模型预测。2019 年 Tian 等提出了 FCOS[30]，使用全卷积网络的结构直接对特征图的每个单元到原图四边的距离进行回归，和语义分割类似把原图每一个单元都作为训练样本，并采用多层级预测的方式在不同层级预测不同尺度的目标，处理重叠物体框产生的模糊性问题。因为 CenterNet 和 FCOS 都是直接检测物体的中心区域和边界信息，所以这类算法被称为基于中心的检测算法（Center-Based）。

上述方法都是单阶段的无锚框检测算法，也可以对两阶段的目标检测算法进行无锚框的设计，比如 RepPoints[31]。

2020 年脸书人工智能研究院（Facebook AI）提出了一种将目标检测视为直接集预测问题的新方法 DETR[32]，将 Transformer 机制引入了目标检测，大大简化了检测流程，有效地消除了许多需要手工设计的组件，如锚框生成和非极大值抑制程序。但是，DETR 在检测小目标上性能较差。因为现代的目标检测器通常都利用了多尺度的特征，从而可以在高分辨率的特征图中检测小目标。但是对于 DETR 来说，高分辨率的特征图将带来不可接受的计算复杂度和内存复杂度。

1.2.2 小目标检测研究现状

由于小目标分辨率低，边缘模糊，携带信息较少，特征提取困难，传统的目标检测算法中，针对小目标检测的算法几乎没有。传统的目标检测算法主要是处理广义的目标检测问题，常采用图像金字塔的方式处理多尺度目标,在金字塔的每一层用固定输入分辨率的分类器在该层滑动，以求在金字塔的底部检测出小目标。比如，基于 HOG 特征加 SVM 分类器的 DPM[33]目标检测框架。

由于小目标本身包含的特征信息比较少，最初的深度卷积神经网络对小目标的学习能力也比较差。为了增强深度神经网络对小目标特征的提取能力，研究人员采用了多尺度特征融合的方法，提升网络对小目标的特征表达能力，并在浅层预测小目标。比如，特征金字塔网络[21]采用上采样的方式将具有语义信息的深层特征和具有细节信息的浅层特征相融合，对不同尺度的目标在不同分辨率的特征层进行独立预测。2017 年 Fu 等提出的 DSSD[34]采用转置卷积对不同尺度的特征图进行融合。2019 年 Zhao 等提出了 M2Det[35]，采用多级特征金字塔网络进行特征融合，更好的解决了目标检测中尺度变化带来的问题。

上述方法在网络结构上进行了创新提升了小目标的检测性能，还有一些工作在训练机制和数据增强方面进行了创新，比如多尺度训练（Multi-Scale Training，MST），困难样本挖掘等。2018 年 Singh 等提出了 SNIP[36]算法，该算法采用一种新的图像金字塔尺度归一化训练方案，在图像金字塔的相同尺度上训练和测试检测器，将不同大小目标的梯度根据图像尺度的变化有选择性地反向传播。2019 年该团队又提出了 SNIPER[37]算法，在图像金字塔中，以适当的比例选择目标周围的上下文区域（Chips）来训练检测器。这种多尺度的训练方法，使得不同尺度的物体都能得到充分的训练，提高了检测器对小目标的检测能力。Shrivastava 等人提出了一种在线困难样本挖掘方法[38]，可以显著提高小目标检测的性能。Kisantal 等采用数据增强[39]的方法增加小目标的特征数量，提升了小目标的检测效果。

此外还有基于超分辨率[40]（Super Resolution，SR）和注意力机制[41]的方法提升了小目标的检测精度，但是这些方法网络结构复杂，不符合本文轻量化的设计原则，这里不再赘述。

1.2.3 轻量化神经网络研究现状

目前，对卷积神经网络的轻量化已经成为了本领域中学术界和工业界的一个研究热点。神经网络的轻量化分为剪枝（Pruning）、量化（Quantization）、蒸馏（Distillation）和采用轻量化网络结构等方法。

2016 年韩松等提出了深度压缩[42]（Deep Compression）的概念，通过剪枝、共享权重和霍夫曼编码实现了对网络模型的压缩，为后续的模型压缩方法定下了基本思路。2015 年 Hinton 提出了知识蒸馏[43]（Knowledge Distillation）方法，并引入了教师网络和学生网络的概念，通过复杂、但性能优越的教师网络输出的软标签指导精简、低复杂度的学生网络训练。通过大型网络蒸馏出的小型网络，不仅计算代价小，而且精度可以和大型网络媲美。

除了上述模型压缩的方法，我们还可以通过调整网络的内部结构来进行轻量化网络的设计。2017 年 Google 的研究人员提出了 MobileNet[44]，利用深度可分离卷积，减少了计算量，构建了面向移动平台的轻量化模型。2018 年该团队又提出了 MobileNetV2[45]，借鉴 ResNet 中残差连接的思想，提出了基于倒置残差的线性瓶颈结构，进一步提升了模型性能。2018 年旷视的研究人员提出了 ShuffleNet[46]，利用分组卷积和通道重组（Channel Shuffle）的方法减少了参数量，并且使各通道之间信息可以交换，在较 VGG-16 仅增加 0.6%错误率的情况下，将网络计算量减少了 30 倍左右。该团队后续又提出了 ShuffleNetV2[47]，指出 使用 FLOPs 作为计算复杂度的唯一度量是不够的，还要考虑内存访问成本、算法并行度以及硬件平台等因素。

随着神经网络架构搜索（Neural Architecture Search，NAS）技术的发展，采用 NAS 技术设计轻量级的神经网络结构成为了可能。2019 年 Tan 等提出了一种资源约束的移动神经网络架构搜索方法 Mnas[48]，在多个视觉任务中优于最先进的移动卷积神经网络。同年 Wu 等提出了 FBNet[49]，基于可微神经网络架构搜索的方法设计了硬件感知的高效卷积神经网络，超过了手工设计和自动生成的最先进模型。基于 NAS 的 MobileNetV3[50]和 EfficientNet[51]也在这一年相继诞生。

针对目标检测任务，从两阶段算法到单阶段算法，不仅在检测精度上有了很大的提升，模型的计算量和参数量也大大降低。但即便如此，在机器人、无人机等移动嵌入式设备上部署高效的目标检测算法，仍然存在着许多挑战。

2017 年 Huang 等在文献[52]中就目标检测中速度、精度和内存的权衡问题进行了分析，为目标检测网络结构的设计提供了指南。2019 年普林斯顿大学的 Law 等人

提出了CornerNet-Lite[28]，由CornerNet-Saccade和CornerNet-Squeeze两种CornerNet[27]的高效变体的组合而成。前者使用了一种注意力机制，消除了对图像中所有像素进行详尽处理的需要，后者引入了一种新的紧凑的骨干架构。2020 年 Tan 等基于 EfficientNet 提出了 EfficientDet[53]，采用一种加权的双向特征金字塔网络（Bi-FPN），实现了简单快速的多尺度特征融合，提出了一种复合缩放方法，对所有骨干网络的分辨率、深度和宽度进行均匀缩放，实现了高效可扩展的目标检测。同年 11 月，Wang 等人提出了 Scaled-YOLOv4[54]系列算法，从大型网络到小型网络对 YOLOv4 进行了扩展，覆盖了高精度版本和快速版本，其中包括 YOLOv4-tiny 这一轻量化的目标检测算法。

1.3 本文的主要工作与创新

本文的主要工作是提出并实现了基于深度学习的轻量化多尺度小目标检测算法 LMDet-S（Lightweight Multiscale Detector for Small Object），兼顾了精度、速度和模型体积，以及小目标检测的能力。还在专门的小目标数据集 TinyPerson 上进行了应用，在“图像切分”与“尺度匹配”策略的基础上，本文的算法在单尺度训练、单模型的情况下取得了较好的检测效果。最后，在 vivo-iQOO-Z1x 手机上进行了部署实验。经实机测试，本文以 ShuffleNetV2 为骨干网络的算法速度可达到 30FPS，满足实时的要求。

本文的主要创新体现在轻量化的颈部网络和检测头部分，具体的创新点有 3 点。

(1) 为了更好的满足深度学习落地的需要，面向移动平台与嵌入式设备的部署，本文提出了一个轻量化的、实用的目标检测框架 LMDet-S，并在移动终端上进行了实机部署，达到了 30FPS 的检测速度，在移动端实现了实时的目标检测。

(2) 基于多尺度特征融合的思想及轻量化的设计原则，设计了轻量化的多尺度特征融合网络 Lite-PAN。相比于现有的特征融合网络如 FPN，PANet 等，该网络更加轻量化，便于在移动设备上部署。并且该网络具有双向的融合路径，很好的平衡了分类与定位任务。

(3) 由于基于锚框的算法鲁棒性与泛化性较差，检测结果对超参数敏感，并且锚框数量众多且计算复杂。本文提出了无锚框的基于中心回归的轻量化检测头部网络 LAF-Head（Lite Anchor-Free Head）。主要包括分类与质量估计分支的合并，以及多层预测结构的轻量化。在标准数据集 COCO 上的实验表明，与前沿算法 YOLOv4-tiny 相比，在相同输入分辨率的情况下，本文算法浮点运算量下降了 42%，参数量减少了 34%，同时平均精度提高了 8.6%，小目标检测精度提高了 2%。

1.4 论文的组织结构

第 1 章首先介绍了小目标检测算法和神经网络轻量化的研究背景与意义，并对它们的国内外研究现状进行了综述。

第 2 章首先研究了深度学习的基础理论与目标检测的基本原理，然后分析了目标检测算法常用的评价指标，最后介绍了常用的标准数据集。

第 3 章首先对轻量级骨干网络的基础模块与结构进行了分析，然后对常用的轻量级骨干网络进行了性能对比分析，最后在支持骨干网络可插拔的基础上选择了适合本文的骨干网络。

第 4 章对轻量化的多尺度特征融合网络进行了研究，提出了轻量化的多尺度特征融合网络 Lite-PAN，并进行了相关实验，验证了该结构的有效性。

第 5 章对目标检测的预测方法进行了研究，提出了基于中心的轻量化无锚框检测头 LAF-Head，并进行了相关实验，最后进行了综合实验。

第 6 章对本文的工作进行了总结与展望。

第 2 章 深度学习与目标检测算法的基础理论

2.1 引言

深度学习的概念最早由 Hinton 等人提出[16]，目前是机器学习领域研究的热点。利用深度神经网络，深度学习可以从大量的数据中提取深层次的结构特征，处理更为复杂的任务。通常认为更深层次的网络可以提取到更多的特征，丰富的特征有利于进一步提升网络分类和检测的性能，因此深度神经网络的层数不断加深，相继诞生了 8 层的 AlexNet[16]，16 层的 VGG-16[54]，22 层的采用 Inception 结构的 GoogLeNet[56]，以及 101 层甚至更深的 ResNet[57]等。

目标检测在生产生活中都有着极其广泛的应用，是近些年来计算机视觉领域的热门研究方向之一。深度学习的发展也推动了目标检测技术的进步，基于深度学习的目标检测算法相对传统的目标检测算法具有更好的特征提取结构和特征分类器，检测精度取得了很大的提升。

2.2 深度学习基础理论

2.2.1 从感知机到卷积神经网络

人工神经网络（Artifial Neural Networks，ANN），简称神经网络，是一种模拟生物神经网络结构和功能的数学模型。

单层人工神经网络又被称为感知机（Perceptron），其结构如图 2–1 所示。图中 $(x_1,x_2,\ldots,x_n)$ 表示神经元的输入，可以记为一个 n 维向量 X，$(w_1,w_2,\ldots,w_n)$ 表示神经元的权重，记为向量 W，y 表示神经元的输出，b 表示偏置项，$f(\cdot)$ 表示激活函数。将图 2–1 表示成公式为：

$$y=f(WX+b) \tag{2-1}$$

感知机学习的过程就是根据网络输出值和真实值的偏差，调整更新权重，使网络输出值不断接近真实值的过程。

将单层感知机推广后得到多层感知机（Multi-Layer Perceptron，MLP）。多层感知机含有多个神经元层，其中第一层称为输入层，中间部分称为隐藏层，最后一层称为输出层。多层感知机一般通过反向传播（Back Propagation，BP）算法，对神经网络的权值进行训练。

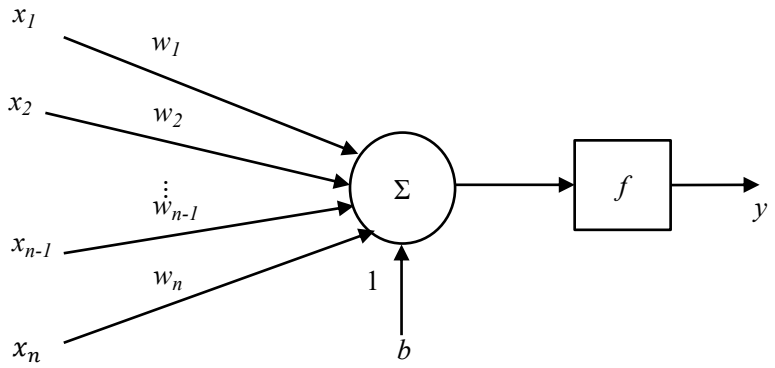


图 2–1 感知机模型

基于感知机模型的全连接神经网络在处理图像信息时，忽略了输入图像的拓扑结构，不具备旋转、伸缩和平移不变性。

为了让神经网络更好的表达视觉信息，科学家们进行了深入研究，1962 年 Hubel 和 Wiesel 在对猫大脑中视觉系统的研究中，发现了视觉皮层的感受野（Receptive Fields）机制[58]；1980 年福岛邦彦提出了 Neocognitron[59]网络，显现了卷积层、池化层的雏形；1998 年 Lecun 提出了 LeNet[60]，卷积神经网络（Convolutional Neural Network，CNN）诞生；2012 年 Hinton 提出了 AlexNet，奠定了深度卷积神经网络在当今计算机视觉领域的地位。卷积神经网络主要由卷积层、激活层和池化层等基本结构组成，如图 2–2 所示。

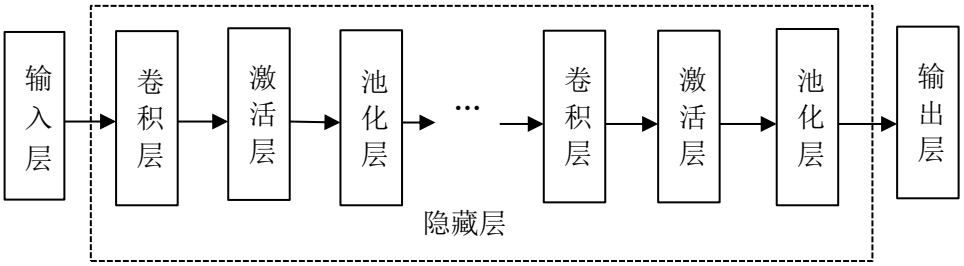


图 2–2 卷积神经网络模型

卷积（Convolution）是信号处理领域的基础运算，应用于图像信号处理的通常是二维卷积，如公式（2–2）所示。

$$S(i,j)=(I*K)(i,j)=\sum_{i=0}^{m}\sum_{j=0}^{n}I(i+m,j+n)K(m,n) \tag{2-2}$$

公式（2–2）中 I 表示输入的二维图像或特征图，K 表示卷积核（Kernel），* 表示卷积操作。

图 2–3 展示了二维卷积运算的过程，其中的卷积核就是卷积神经网络要学习的内容。一个卷积层可以有多个不同的卷积核，对应多个输出通道，可以实现不同特征的组合。

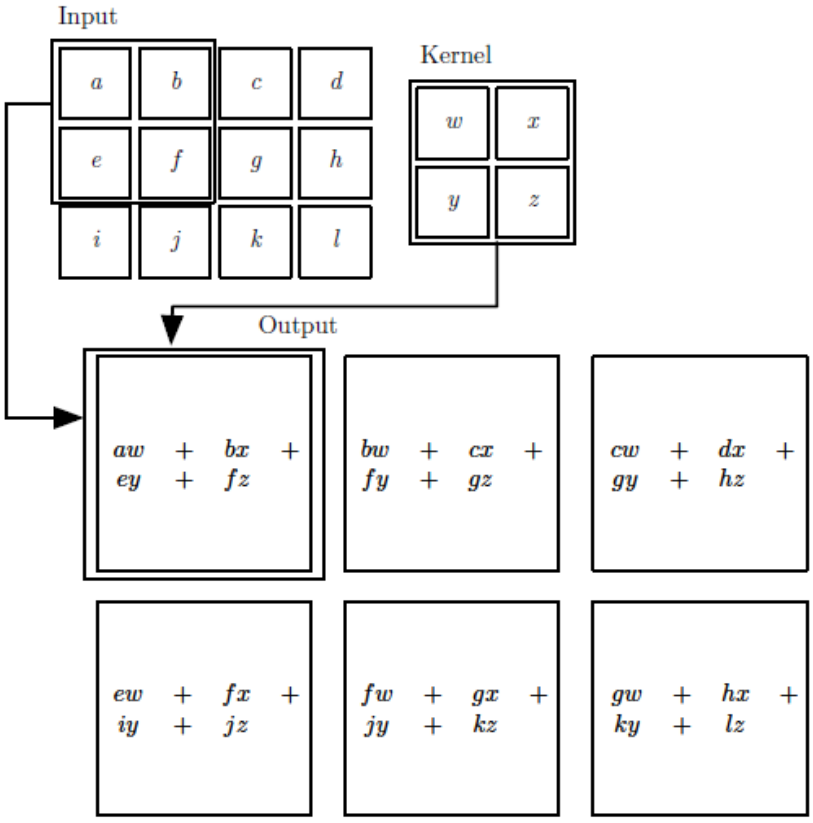


图 2–3 二维卷积运算过程

卷积的参数还包括填充（Padding）和步长，填充是为了保证输出特征图和输入特征图大小一致，以及更好的提取边缘信息。常见的填充方法有补零填充、边界复制填充、镜像填充和块填充。步长是指卷积核在输入特征图上每次卷积后移动的距离，通过调节步长大小可以对输出特征图的尺寸进行缩放。记填充数量为 P，步长为 S，卷积核大小为 K，假设输入特征图的尺寸为 $W_i \times H_i \times D_i$，卷积核个数为 N，输出特征图的尺寸为 $W_o \times H_o \times D_o$，则输出特征图大小的计算公式为：

$$\begin{cases} W_o=(W_i-K+2P)/S+1 \\ H_o=(H_i-K+2P)/S+1 \\ D_o=N \end{cases} \tag{2-3}$$

通过卷积运算获得了图像的特征，但如果把所有特征直接送到分类器，计算量会非常大，而且容易发生过拟合（Over-fitting）。

因此，人们引入池化（Pooling）操作，降低特征维度，减少后续网络的计算量，防止过拟合。池化一般分为两种，一种是最大池化（Max Pooling），选择池化区域的

最大值作为输出；另一种是平均池化（Mean Pooling），选择池化区域的平均值作为输出。因为采用了求最大值或平均值这种统计聚合的方法，所以池化后的特征鲁棒性更强。

激活层主要是为了增强神经网络的非线性表达能力，常见的非线性激活函数有：

(1) Sigmoid 函数

Sigmoid 函数的表达式如公式（2–4）所示，它可以将输入的实数映射到 0 到 1 之间。因为 Sigmoid 函数的输出不是零均值的，因此在下一层的输入也是非零均值的，这将导致后续的网络训练时间增加。

$$f(x)=\frac{1}{1+e^{-x}} \tag{2-4}$$

Sigmoid 函数的导数如公式（2–5）所示，从中可以看出，采用 Sigmoid 函数可能在梯度反向传播时，出现梯度消失的问题。

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x}=\frac{e^{x}}{(1+e^{x})^{2}} \tag{2-5}$$

(2) tanh 函数

tanh 函数即双曲正切函数，其表达式如公式（2–6）所示，它的输出取值范围为(-1, 1)，是零均值的输出，因此收敛速度比 Sigmoid 快。

$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{2-6}$$

tanh 函数的导数如公式（2–7）所示，虽然没有解决梯度消失的问题，但实际应用效果已经有了很大提升。

$$\frac{\mathrm{d}tanh(x)}{\mathrm{d}x}=1-tanh^{2}(x) \tag{2-7}$$

(3) ReLU 函数

ReLU 函数（The Rectified Linear Unit）的表达式如公式（2–8）所示，它的输出不是零均值的。

$$f(x)=\begin{cases}0 & x\le 0\\ x & x>0\end{cases} \tag{2-8}$$

ReLU 函数的导数如公式（2–9）所示，可见 ReLU 函数及其导数都非常简单，因此运行速度很快，自从 AlexNet 将 ReLU 引入深度学习以来，它在深度神经网络中得到了广泛应用。

$$\frac{\mathrm{d}ReLU(x)}{\mathrm{d}x}=\begin{cases}0 & x\le 0\\ 1 & x>0\end{cases} \tag{2-9}$$

由公式（2–8）和公式（2–9）可以看出，在坐标轴的负半轴上，ReLU 函数及其导数的输出均为 0，这意味着网络在训练时无法更新权重，这个问题被形象的称为神经元“死亡”问题。为了解决这个问题出现了 LeakyReLU[61]和 PReLU[62]等针对 ReLU 函数负半轴进行改进的变体。在坐标轴的非负半轴上，ReLU 函数的输出为 $[0,\infty)$，为了减少模型量化后精度的损失，可采用“范围受限”的激活函数，比如 ReLU6，将输出范围限制在 $[0,6]$。

2.2.2 卷积算子

上节对标准卷积运算与网络的基本结构进行了分析，随着深度学习的发展，为了适应更多的深度学习任务，提高卷积神经网络的性能，在标准卷积的基础上又提出了许多变体，比如膨胀卷积、转置卷积、深度可分离卷积和可变形卷积等。在本文提出的目标检测算法中对这些新的卷积算子也有应用，因此本节对这些卷积算子进行简要介绍。

(1) 膨胀卷积

膨胀卷积[63]（Dilated Convolution）也称空洞卷积，由 Yu 等人在 2016 年提出，最初应用于语义分割任务。膨胀卷积的计算过程如公式（2–10）所示。

$$S(i,j)=(I*K)(i,j)=\sum_{i=0}^{m}\sum_{j=0}^{n}I(i+r\cdot m,j+r\cdot n)K(m,n) \tag{2-10}$$

式中，r 为膨胀率。当 r=1 时，公式（2–10）相当于标准卷积。当 r>1 时，空洞卷积相当于在卷积核的相邻元素间填充了 r-1个 0，在不损失特征图分辨率的情况下，增大了卷积的感受野（Receptive Field），如图 2–4 所示。

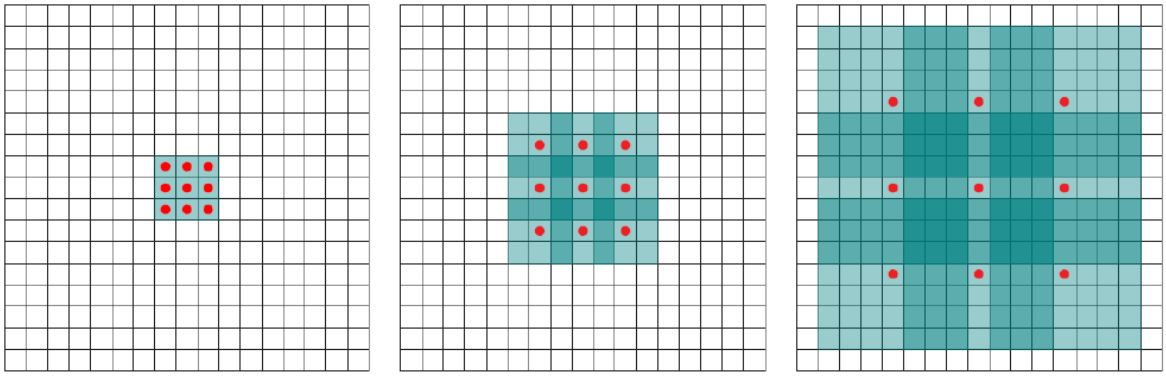


图 2–4 膨胀卷积示意图，(a) 膨胀率为 1 的卷积，感受野大小为 3×3；(b) 膨胀率为 2 的卷积，感受野大小为 7×7；(c) 膨胀率为 4 的卷积，感受野大小为 15×15

从图 2–4 中我们可以发现，一个尺寸为 $k\times k$ 的膨胀卷积核相当于一个 $k_s\times k_s$ 大小的标准卷积核，如公式（2–11）所示。

$$k_s=k+(r-1)\cdot(k+1) \tag{2-11}$$

(2) 转置卷积

转置卷积（Transposed Convolution）与卷积下采样的过程相反，是一个上采样的过程，因此也做叫反卷积或者去卷积（Deconvolution），如图 2–5 所示。

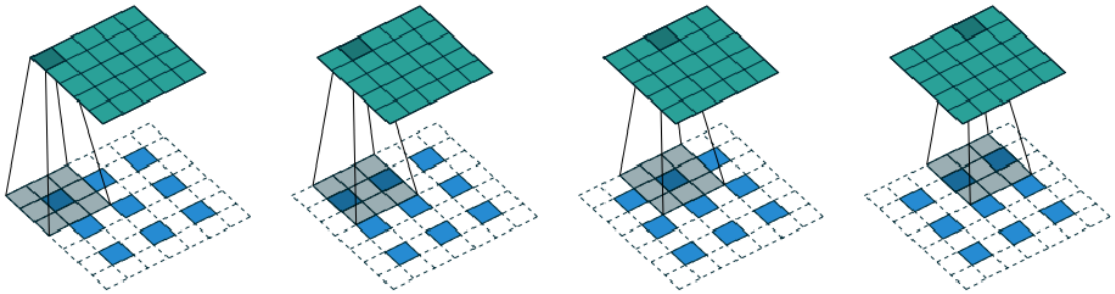


图 2–5 将一个3×3的特征图经过转置卷积上采样成一个5×5的特征图

在目标检测等视觉任务中，常用于不同尺度特征图分辨率的匹配，可以将小尺寸的特征图经过转置卷积上采样到和大尺度特征图相同的分辨率，从而实现不同尺度特征度的融合。需要注意的是，和标准卷积一样，转置卷积的卷积核在训练过程中也是需要学习的。

(3) 深度可分离卷积

深度可分离卷积（Depthwise Separable Convolution）将标准卷积分解成逐层卷积（Depthwise Convolution）和逐点卷积（Pointwise Convolution）两个步骤。

逐层卷积对于输入特征图的每一个输入通道使用一个卷积核进行卷积，然后逐点卷积通过1×1的卷积核对特征图的所有通道进行卷积，将逐层卷积生成的特征组合起来生成新的特征。深度可分离卷积通过将标准卷积解耦成生成特征和组合特征这两个部分，减少了运算量，使得网络更加轻量化。

本文提出的 LAF-Head 中就采用了深度可分离卷积，减少了运算量。3.3.1 节对深度可分离卷积的计算过程以及运算量进行了详细分析。

(4) 可变形卷积

可变形卷积[64]（Deformable Convolution，DCN）由微软亚洲研究院在 2017 年提出。在标准卷积中，卷积核在在输入图像或特征图的预定义矩形网格上进行操作，而在可变形卷积中，每个网格都可以通过一个可学习的偏移量进行移动，卷积作用在这些可移动的网格上，如图 2–6 所示。

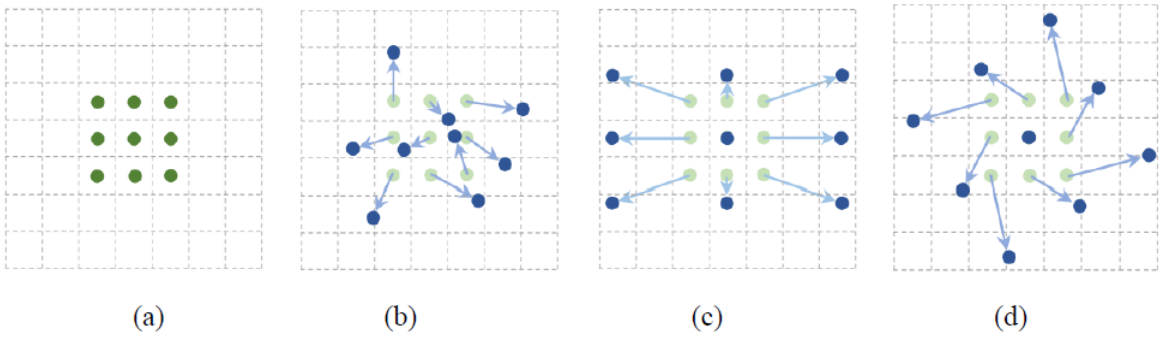


图 2–6 3×3标准卷积和可变形卷积的采样位置说明，(a) 标准卷积的规则采样网格，(b) 带有增强偏移量（浅色箭头）的变形采样点（外围深色点），(c) (d)是 (b) 的特殊情况，表明可变形卷积可以涵盖尺度、宽高比和旋转的各种变换

这样可以很好的解决待检测的对象在图像中的变形或遮挡问题，只要增加很少的计算量，就可以在检测或分割任务上得到性能的提升。

2.2.3 损失函数与优化方法

损失函数反映了模型预测值和真实值之间的差异。网络会根据损失函数的值通过反向传播（Back Propagation，BP）算法对网络的所有层的权重进行更新，使损失函数的值最小化，来指导网络模型的学习。

不同的深度学习任务有不同的损失函数，目标检测任务可以分为一个分类任务与一个回归任务。因此本节主要介绍用于分类和回归的损失函数。

(1) 分类损失函数

交叉熵（Cross Entropy）函数是分类任务中常用的损失函数，如公式（2–12）所示，$p(x)$ 表示真实值的概率分布，$q(x)$ 表示网络预测值的概率分布，x 表示网络的输入。

$$H(p,q)=-\sum_{x}p(x)\,log\,(q(x)) \tag{2-12}$$

对于二分类问题，交叉熵损失函数如公式（2–13）所示。式中 y_i 表示样本 i 的标签，正样本为 1，负样本为 0。

$$L=\frac{1}{N}\sum_{i}L_i=\frac{1}{N}\sum_{i}-\left[y_i\cdot log(p_i)+(1-y_i)\cdot log(1-p_i)\right] \tag{2-13}$$

对于多分类问题，交叉熵损失函数如公式（2–14）所示。式中 M 表示类别的数量，y_{ic} 是指示变量（0 或 1），如果类别 c 与第 i 个样本的类别相同就是 1，否则是 0。p_{ic} 表示对于第 i 个观测样本属于类别 c 的预测概率。

$$L=\frac{1}{N}\sum_{i}L_i=\frac{1}{N}\sum_{i}-\sum_{c=1}^{M}y_{ic}\,log(p_{ic}) \tag{2-14}$$

(2) 回归损失函数

常见的回归损失函数包括 Smooth L1、均方误差和 IoU 损失函数等。

1) SmoothL1 损失函数

在目标检测中，在进行边框回归时一般采用 SmoothL1 损失，其表达式如公式（2–15）所示。式中 x 表示网络预测值与真实值的偏差。

$$SmoothL_1(x)=\begin{cases}0.5x^2 & if\ |x|<1\\ |x|-0.5 & otherwise\end{cases} \tag{2-15}$$

公式（2–16）表示的是 Smooth L1 函数的导数，可以看出，当预测值与监督信息差别过大时，梯度值不至于过大；当预测值与监督信息差别很小时，梯度值足够小。

$$\frac{\mathrm{d}SmoothL_1(x)}{\mathrm{d}x}=\begin{cases}x & |x|<1\\ -1 & x<-1\\ 1 & x>1\end{cases} \tag{2-16}$$

2) 均方误差损失函数

均方误差损失（Mean Square Error，MSE）也叫 L2 损失，如公式（2–17）所示。

$$L_{MSE}=\frac{1}{N}\sum_{i=1}^{N}\left(y_i-p_i\right)^2 \tag{2-17}$$

3) IoU 损失函数[65]

IoU（Intersection over Union）又称交并比，表示预测框和真实框（Ground Truth）之间的重合度，如图 2–7 所示，图中 A 表示预测框，G 表示真实框。

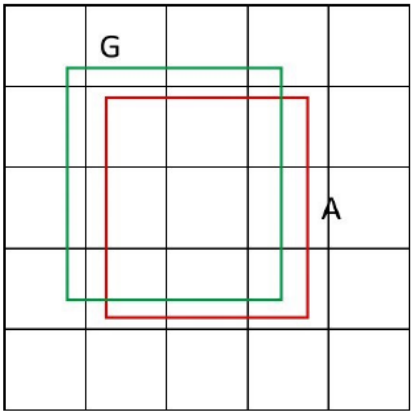


图 2–7 IoU 示意图

IoU 的计算方法如公式（2–18）所示。

$$IoU=\frac{A\cap G}{A\cup G} \tag{2-18}$$

公式（2–19）是 IoU 损失的计算公式，IoU 损失将矩形框看作一个整体来计算损失，相较于将矩形框的每一个点分别计算损失的 L2 损失效果明显提升。

$$IoUloss=-ln(IoU) \tag{2-19}$$

深度神经网络的训练可以看作是其损失函数的最优化问题。深度学习模型通常由随机梯度下降（Stochastic Gradient Descent，SGD）算法进行训练。随机梯度下降算法有许多变体，包括基于动量的梯度下降算法和基于自适应学习率的梯度下降算法。

随机梯度下降算法就是每次从训练集中选取一个或者一部分（mini-batch）样本来计算损失函数的梯度，并进行参数的更新，如公式（2–20）所示。

$$\theta=\theta-\eta\nabla_{\theta}J(\theta) \tag{2-20}$$

式中，θ 表示神经网络的参数，$J(\theta)$ 表示损失函数，η 代表学习率。如果学习率设置的过小，网络很可能会陷入局部最优解，也会耗费较长的时间进行网络的训练；如果学习率设置的过大，损失函数可能会产生震荡，训练将无法收敛。因此在训练神经网络模型时，一般设置变化的学习率，随着训练时间的推移降低学习率，以使神经网络模型更好的收敛。

为了减少由于随机采样或者噪声带来的梯度振荡，解决随机梯度下降有时会很慢的问题，1964 年 Polyak 提出了动量（Momentum）算法[66]。根据物理学中“物体下一时刻的速度将由物体当前的速度及其所受的力共同决定”的思想，动量算法引入了一个新的变量 v，表示之前梯度计算值的累加，使得之前的梯度也能影响到当前参数的更新，如公式（2–21）与公式（2–22）所示。

$$v_{n+1}=\gamma\cdot v_{n}+\eta\cdot\nabla_{\theta}J(\theta) \tag{2-21}$$

$$\theta_{n+1}=\theta_{n}-v_{n+1} \tag{2-22}$$

式中，γ 为衰减项，决定了之前梯度衰减的程度，γ 越大，之前梯度对当前参数的影响也就越大。Momentum 算法可以加速神经网络的收敛，之后又出现了它的变体 Nesterov 加速梯度下降（Nesterov Accelerated Gradient，NAG）算法[67]。NAG 算法可以计算损失函数对于参数未来近似值的梯度，从而更加高效的求解。

在神经网络训练的过程中，有的参数可能已经到了仅需微调的阶段，但有的参数仍需较大幅度的调整，也就是说同一个更新速率不一定适合所有参数。

针对这一情况，研究人员提出了自适应学习速率的优化算法，包括 AdaGrad[68]，AdaDelta[69]，RMSProp[70]，Adam[71]，和 AdamW[72]等。其中 Adam（Adaptive Moment Estimation）算法是带有动量项的自适应学习速率优化算法，在大型数据集的训练中应用较多。

2.3 目标检测原理与相关概念

2.3.1 目标检测原理

目标检测任务分为目标分类和目标定位两个子任务。目标分类就是输出目标的类别信息；目标定位就是输出目标的位置信息，一般用矩形框的方式表示出来，包含矩形框左上角或者中心位置的坐标和矩形框的宽高。一个现代的目标检测器通常由三个部分组成，一部分是在 ImageNet 上预训练的骨干网络，另一部分是用来增强特征的颈部网络，最后一部分是用来预测目标类别和边界框的检测头部网络，如图 2–8 所示。

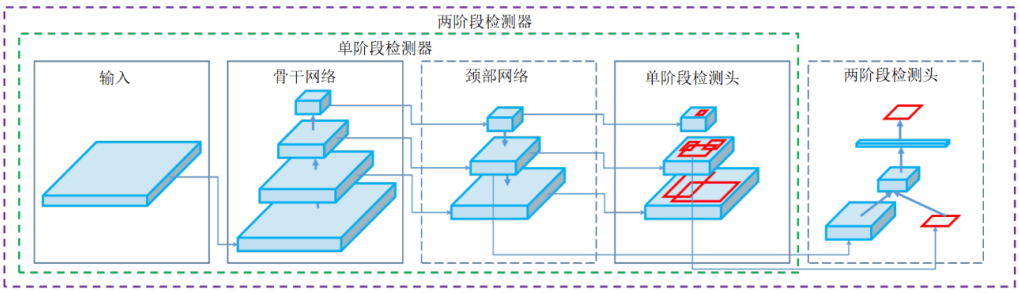


图 2–8 现代目标检测器组成

骨干网络可以分为重量级的网络和轻量级的网络，前者主要运行在大型的 GPU 平台上，可以是 VGG[54]，ResNet[57]，ResNeXt[73]或 DenseNet[74]等；后者主要运行在 CPU 或嵌入式平台上，可以是 MobileNet， ShuffleNet 或 SqueezeNet 等。

表 2–1 列举了常见的骨干网络的网络结构和网络层数。除了上述网络，还有一些研究人员开发了目标检测的专用骨干网络，包括 DetNet[75]，DetNAS[76]等。

表 2–1 常见的骨干网络分析

网络名称	网络结构	网络层数
VGG-16[54]	直筒型	16
ResNet101[57]	残差型	101
MobileNetV2[45]	倒置残差型	54
Hourglass-104[27]	沙漏型	104
FBNet[49]	NAS 自动设计	26

近年来发展起来的目标检测算法往往在骨干网络和检测头之间加入一些层，这些层通常用于提取不同阶段的特征图，业界一般称之为检测器的颈部。颈部网络通常由多条自顶向下和自底向上的网络通路组成。具备这种机制的网络包括，FPN[21]、PANet[77]、Bi-FPN[53]和 NAS-FPN[78]等。

根据检测头部网络的不同，可以把目标检测器分为两类单阶段目标检测器和两阶段目标检测器。同时，随着无锚框检测思想的流行，两阶段目标检测器和单阶段目标检测器都可以做成无锚框检测器。常见的目标检测器如 1.2.1 节所述。

2.3.2 目标检测的评价指标

在目标检测的任务中，精确率（Precision）和召回率（Recall）是最主要的两个评价指标。精确率表示的是所有被检测为正例的样本中，真实值也为正例的比例，也叫查准率。召回率表示的是，所有的正样本中，被正确检测到的正样本占的比例，也叫查全率。二者如公式（2–23）所示。

$$\begin{cases} Precision = \dfrac{TP}{TP+FP} \\ Recall = \dfrac{TP}{TP+FN} \end{cases} \tag{2-23}$$

式中 TP 表示真阳性（True Positive），FP 表示假阳性（Full Positive），FN 表示假阴性（False Positive），TN 表示真阴性（True Neagative），如图 2–9 的混淆矩阵（Confusion Matrix）所示。

混淆矩阵		真实值	
		Positive	Negative
预测值	Positive	TP	FP
	Negative	FN	TN

图 2–9 混淆矩阵

精确率与召回率数据的取值范围都在 0~1 之间，以召回率为 X 轴、以精确率为 Y 轴可以绘制 P-R（Precision-Recall）曲线，如图 2–10 所示。

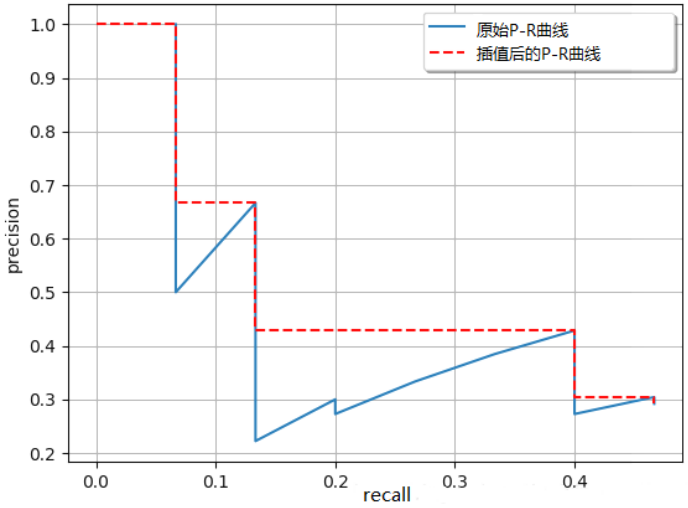


图 2–10 P-R 曲线的插值

为了平衡精确率和召回率两个指标的关系，在目标检测任务中常用平均精度（Average Precision，AP）作为检测结果的评价指标。AP 的计算方法为 P-R 曲线下覆盖的面积，如公式（2–24）所示。

$$AP=\int_0^1 p(r)dr \tag{2-24}$$

由于积分计算相对困难，因此引入插值法计算，如公式（2–25）所示。

$$AP=\sum_{k=1}^{N} \max_{\tilde{k}\geq k} P(\tilde{k})\Delta r(k) \tag{2-25}$$

式中 k 表示的插值点数，VOC 数据集中是 11 点插值，COCO 数据集中是 101 点插值[4]。

AP 表示的是单类别的平均精度，对于多类别则采用 mAP（Mean Average Precision），也即所有类别下得到的 AP 求平均，如公式（2–26）所示。式中 C 表示类别，N 表示总的类别数目。

$$mAP=\frac{\sum_{C=1}^{N} AP}{N} \tag{2-26}$$

上述的 AP，Recall 以及 mAP 都是针对目标分类而言的，但在目标检测中一幅图像中可能有多个目标，这时就需要从定位方面来考虑。一般使用 IoU 指标来评估预测框和真实框的重合率，从而判断该目标是否定位正确。通过设定适当的 IoU 阈值，对预测过程中产生的大量的框进行筛选，当预测得到的矩形框和真值框之间的 IoU 大于阈值且类别正确，那么就说明该矩形框有效。

2.3.3 常用数据集

用于目标检测的常用大型标准数据集包括 COCO 和 PASCAL VOC 两个数据集。这两个数据集包含较多的小目标数据，适合本课题的研究。目前，专门针对小目标检测的大型数据集目前还很少见，由中国科学院大学标注的 TinyPerson[79]数据集是一个新兴的弱小目标数据集，主要用于弱小人体目标的检测。

(1) COCO

COCO（Microsoft Common Objects in Context）数据集[4]由微软出资标注，起源于 2014 年，包括 COCO-2014 和 COCO-2017 两个子数据集，主要用于目标检测、语义分割、人体关键点检测和图片标题生成等任务。其中 COCO2014 包含 80 个目标类，82783 张训练图像，40504（约 40K）张验证图像，886266 个标注框。也可将从验证集中取出 35504（约 35K）张放到训练集中，重新组成 118287 张图像的训练集，和 5036（约 5K）张图像的迷你验证集。图 2–11 为 COCO-2014 数据集的数据示例。

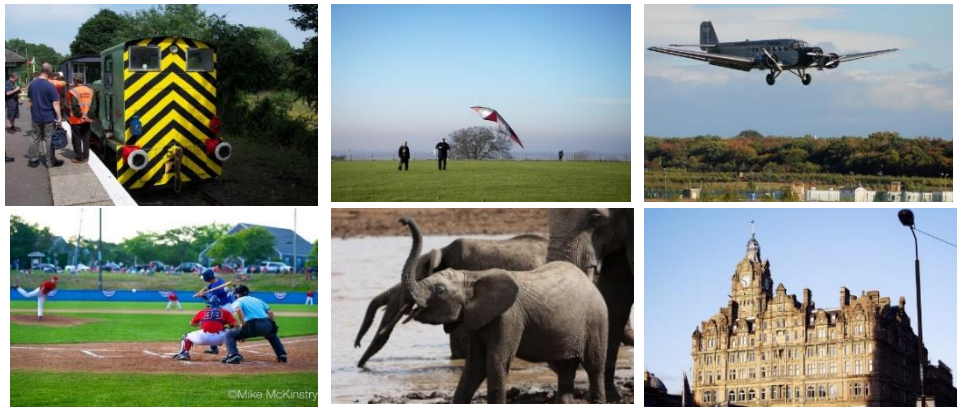


图 2–11 COCO2014 数据集数据示例

COCO 数据集中将像素面积小于 32*32 的物体称为小目标，将像素面积大于 32*32 小于 96*96 的物体称为中目标，将像素面积大于 96*96 的物体称为大目标，如表 2–2 所示。

表 2–2 MS-COCO 数据集中对不同尺寸目标的定义

目标类别	最小像素值	最大像素值
小目标	0×0	32×32
中目标	32×32	96×96
大目标	96×96	∞×∞

COCO 数据集提供了不同尺寸目标的评价指标，AP_S, AP_M和AP_L，分别表示小目标、中目标，以及大目标的 AP。同理定义了不同尺寸平均召回率的评价指标 AR_S，AR_M 和 AR_L。

COCO 数据集中含有约 41%的小目标，34%的中目标，和 24%的大目标，如表 2–3 所示。

表 2–3 COCO 数据集中小目标数量和像素统计

目标类别	数量统计	像素所占比例
小目标	41.43%	15.3%
中目标	34.32%	34.2%
大目标	24.24%	50.5%

从表中可以发现 COCO 数据集中小目标数量较多，但是相比于大中目标，小目标所占的像素又很少，这说明了目标的相对尺寸小，也就是小目标在整幅图像中所占的像素比例少，而不是图像尺寸较小导致的目标绝对尺寸较小。因此，也可以采用相对尺寸的定义表示小目标如公式（2–27）所示。

$$Relative\ Scale=\sqrt{\frac{width_{gt}*height_{gt}}{width_{image}*height_{image}}}<3\% \tag{2-27}$$

在 COCO 数据集中，采用了更加严格的 mAP 计算方法，从 0.5 到 0.95，依次间隔 0.05，选取一系列 IoU 阈值，计算 mAP，然后再取平均值，作为最后的 mAP。COCO 风格的 mAP 如公式（2–28）所示，业界习惯上也简称为 AP。

$$mAP_{\mathrm{COCO}}=\frac{\sum_{IoU=0.5}^{0.95} mAP}{10}(IoU=0.5:0.05:0.95) \tag{2-28}$$

业界常用 AP_{50} 表示 IoU=0.5 时的 AP，也就是下文所述的 PASCAL VOC 中的度量方法，用 AP_{75} 表示 IoU=0.75 时的 AP，也就是更严格的度量方法。对于轻量化模型，往往采用 AP_{50} 这种比较宽容的评价指标。

(2) PASCAL VOC

PASCAL VOC[82]（The PASCAL Visual Object Classes Challenge）曾经是计算机视觉领域中一个世界级的挑战赛，促进了计算机视觉中目标检测和语义分割等任务的发展，催生了大量杰出的算法模型。虽然该比赛已在 2012 年停止举办，但比赛官方发布的 PASCAL VOC 数据集，至今仍被广大研究者使用。

PASCAL VOC 数据集共有 20 个类，包含了生活中常见的物体。其中鸟、瓶子、盆栽植物等属于尺寸较小的物体。目前较为常用的是 PASCAL VOC 2007 和 PASCAL VOC 2012 两种数据集。其中 PASCAL VOC 2007 包含训练集 5011 张，验证集 4952 张，PASCAL VOC 2012 包含训练集 5717 张，测试集 5823 张。PASCAL VOC2012 数据集数据示例如图 2–12 所示。

图 2–12 PASCAL VOC2012 数据集数据示例

(3) TinyPerson

TinyPerson[79]是以快速海上救援和海岸周边防御为背景的航拍图像数据集，由中国科学院大学视觉实验室标注，是领域内第一个用于弱小人体目标感知的数据集，包含 1610 图像，其中训练集 794 张，验证集 816 张，共 72651 个目标。TinyPerson 数据集数据示例如图 2–13 所示。

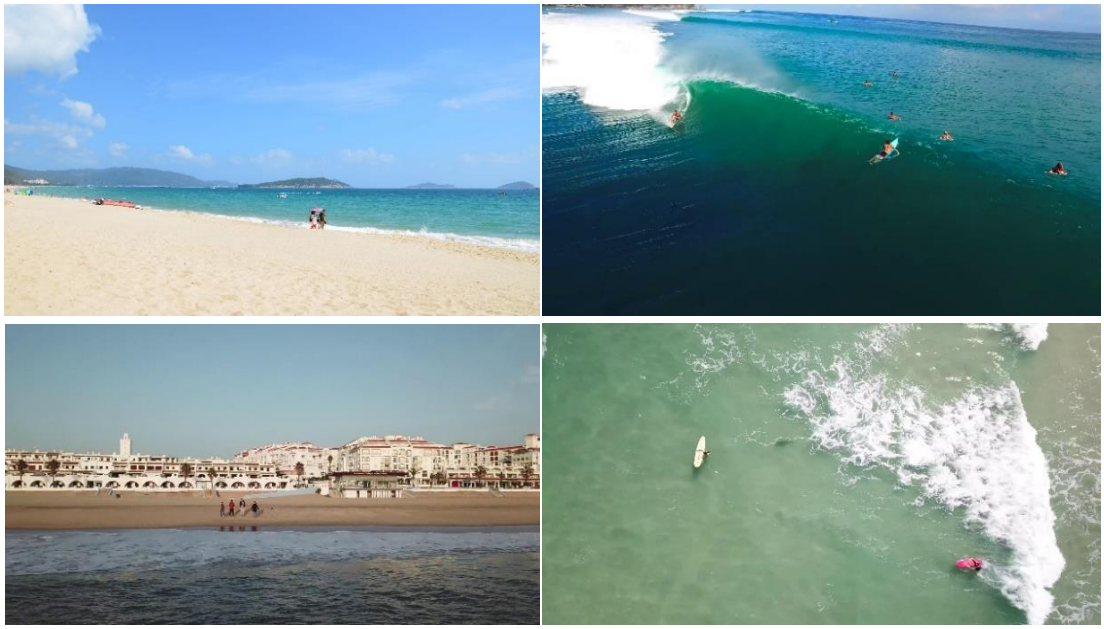


图 2–13 TinyPerson 数据集数据示例

图 2–14 TinyPerson 数据集标注示例“sea person”，“earth person”，“uncertain sea person”，“uncertain earth person”和“ignore region”分别用红色，绿色，蓝色，黄色，和紫色的矩形表示，这些区域被放大并显示在右侧

TinyPerson 数据集标注示例如图 2–14 所示。该数据集一共有两个类别，海洋人（sea person）和陆地人（earth person）。标注中含有 iscrowd，ignore，uncertain，logo，in_dense_image 等参数选项。极小的目标尺寸是该数据集的关键特征，也是主要挑战。

TinyPerson 数据集中目标的尺寸范围被分为 2 个区间，tiny[2, 20] 和 small[20, 32]。其中 tiny[2, 20] 又被分为 3 个子区间，分别为 tiny1[2, 8]，tiny2[8, 12]，tiny3[12, 20]。其中 small[20, 32] 的评价指标和本文中的小目标标准接近，因此本文主要关注 small[20, 32] 的性能指标。TinyPerson 中也采用 AP 进行性能评估，因为在许多微小人检测的应用中，比如沉船搜救，更多的是在于寻找人而不是精确定位，所以 IoU 阈值设置为 0.5 或 0.25。

本文主要在 COCO 数据集上进行实验，部分测试图像选自 VOC 数据集，最后在 TinyPerson 数据集上进行了小目标检测的验证性实验。

2.4 本章小结

本章首先研究了深度学习的基础理论，对深度卷积神经网络的基本结构，优化方法进行了深入探讨；然后研究了目标检测的基础原理，并分析了目标检测算法常用的评价指标；最后介绍了常用的标准数据集，并对这些数据集的样本数量、目标类别以及度量方法等进行了分析。

第 3 章 轻量化骨干网络的研究

3.1 引言

骨干网络处于整个目标检测网络的前端，起到特征提取的作用。骨干网络轻量化的好坏直接影响到后续网络的性能。目前，已经有很多成熟的轻量级骨干网络模型，如 MobileNet[44], ShuffleNet[46], EfficientNet[51]以及 FBNet[49]等，也诞生了一系列神经网络轻量化的方法，如剪枝、量化和蒸馏等。目标检测中的骨干网络一般采用在 ImageNet 上预训练的网络模型。本文提出的 LMDet-S 采用了模块化设计，只需修改配置文件即可更换不同的骨干网络模型，支持 ShuffleNetV2 和 EfficientNet-Lite 作为骨干网络。

3.2 神经网络轻量化的方法与评价指标

实现深度神经网络高精度和高效之间的最优平衡是近年来一个活跃的研究领域。常用的神经网络轻量化方法包括网络剪枝、参数量化、知识蒸馏，以及设计轻量化卷积结构等，它们的优缺点对比分析如表 3–1 所示。

表 3–1 模型轻量化方法优缺点对比

轻量化方法	优点	缺点
网络剪枝（Pruning）	去除冗余参数；降低网络的复杂度；避免网络过拟合	底层的硬件和计算库没有较好的支持，难以获得实质的性能提升
参数量化（Quantization）	减小模型尺寸；降低的内存和缓存；提高计算速度；降低功耗	精度损失；最终加速效果取决于硬件本身架构
知识蒸馏（Distillation）	蒸馏得到的小型网络的精度可以媲美大型网络	对于超参数敏感，不同的超参设置可能导致截然不同的结果

因为本文的算法面向移动平台与嵌入式设备，需要考虑算法的落地应用，结合表 3–1 的分析本文主要采用了设计轻量化卷积结构的方法。

评价神经网络轻量化的常用指标，可分为理论分析指标和实机测试指标两种类型。其中，理论分析层面常用的指标有浮点运算量（Floating Point Operations，FLOPs）和模型参数量。浮点运算量是指模型推断时需要的浮点运算次数，通常是以乘加次数（Multiply-Adds，MAdds）表示，间接表征了运算速度。FLOPs 的常用单位有 BFLOPs 和 MFLOPs。

$$1\ \text{BFLOPs}=10^{9}\ \text{FLOPs} \tag{3-1}$$

$$1\ \text{MFLOPs}=10^{6}\ \text{FLOPs} \tag{3-2}$$

FLOPs 用来衡量算法的时间复杂度，而参数量（Params）用来衡量算法的空间复杂度。模型参数量是指模型含有多少参数，直接决定模型文件的大小，也影响推理时内存的占用量。

实机测试层面的指标主要有帧率（Frames Per Second，FPS）和每帧的推理时间（Inference Time）。FPS 是推理时间的倒数。这两个参数与硬件有关，在不同的 GPU 或者 CPU 上表现不同，在比较 FPS 和推理时间时应说明硬件平台。

3.3 轻量化骨干网络的基础模块与结构分析

轻量化的特征提取网络从 MobileNet, ShuffleNet 到基于 NAS 的 MnasNet, MobileNetV3 以及 EfficientNet 等都采用了深度可分离卷积和基于倒置残差的线性瓶颈结构。本节对这两个基本模块（Building Block）的原理进行分析，借鉴其中的设计思想，为设计本文轻量化算法框架的后续检测网络提供指南。

3.3.1 深度可分离卷积

卷积神经网络中，卷积运算占据了整个卷积神经网络的大部分时间，直接影响整个卷积神经网络的复杂度。

以往的卷积神经网络模型主要在服务器上运行，计算量比较大。在服务器上运行的标准卷积模型如图 3–1 所示。其特点主要有：

(1) 有 N 个卷积核，也即 N 个输出通道；

(2) 每个卷积核的长宽都是 D_K；

(3) 输入有 M 个通道，也即卷积核的深度为 M。

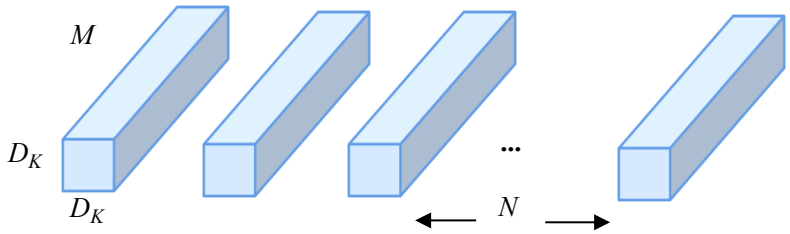


图 3–1 标准卷积结构图，N 个 $D_K \times D_K$ 卷积核对 M 个通道进行卷积

设输出特征图的尺寸为 $D_F \cdot D_F$，则图 3–1 中的标准卷积结构计算量为：

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \tag{3-3}$$

为了减少计算量，满足轻量化的设计要求，深度可分离卷积（Depthwise Separable Convolution），将一个标准的卷积结构改为：

(1) 一个逐层卷积，也叫深度卷积（Depth-wise），如图 3–2 所示；

(2) 一个逐点（Point-wise）卷积，也就是 1×1 卷积，如图 3–3 所示。

图 3–2 逐层卷积核结构图，每个通道和一个不同的 $D_K \times D_K$ 卷积核进行卷积

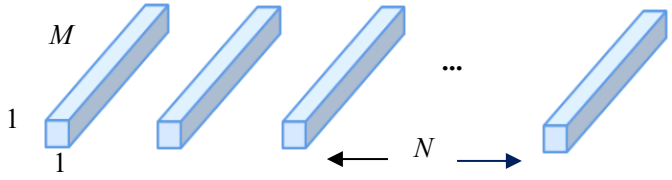


图 3–3 逐点卷积核结构图，分别对 M 个通道进行逐点卷积计算

深度可分离卷积的计算量为：

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F \tag{3-4}$$

深度可分离卷积的计算量和标准卷积的计算量的比值为：

$$\frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} = \frac{1}{N} + \frac{1}{D_K^2} \approx \frac{1}{D_K^2} \tag{3-5}$$

输出通道数N一般远大于卷积核尺寸 D_K，因此深度可分离卷积的计算量近似为标准卷积的 $1/D_K^2$。现代卷积核尺寸一般为 3×3，所以深度可分离卷积的计算量近似为标准卷积的 1/9，大幅减少了运算量。

本文第五章提出的轻量化无锚框检测头 LAF-Head 中也采用了深度可分离卷积作为基础模块，降低了模型计算量。

3.3.2 基于倒置残差的线性瓶颈层

ResNet[57]设计了残差模块（Residual Block）作为网络的基本组成部分，这种残差连接的思想，在一定程度上缓解了因为深度带来的梯度消失问题，让深度卷积神经网络的层数可以做得更深，提高了网络的性能，促进了深度卷积神经网络的发展。MobileNetV2[45]借鉴了 ResNet 网络结构的设计思想，与原始的残差模块先降维后升

维不同，MobileNetV2 中的残差模块采用了先升维后降维的结构，称为倒置残差模块（Inverted Residual Block），并采用深度可分离卷积替换了标准卷积。两者的结构对比如图 3–4 所示。

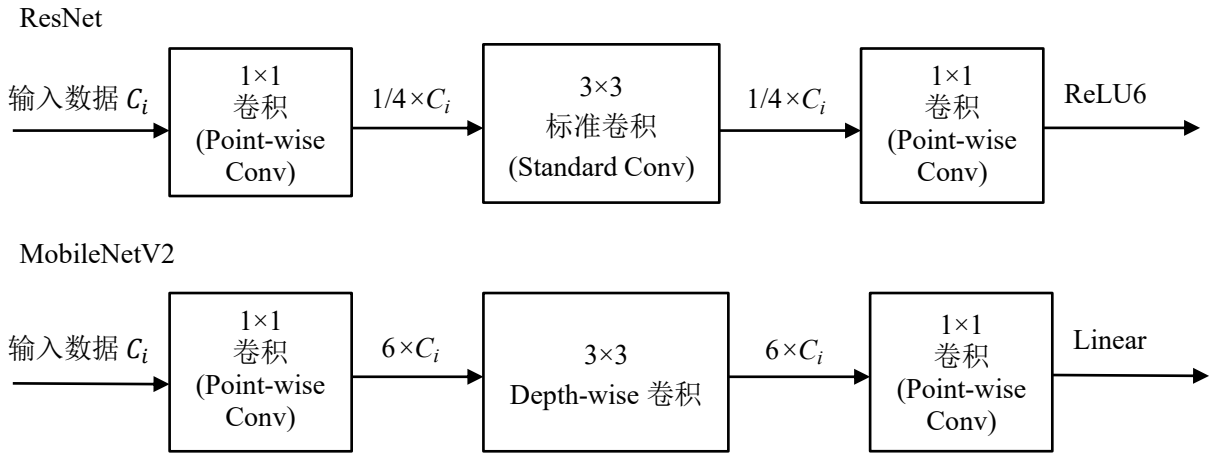


图 3–4 残差模块与倒置残差模块对比

残差模块的结构是一个两边大中间小的沙漏形，改进后的倒置残差模块的结构则是两边小中间大的纺锤形。中间升维扩张后通道数量增加，可以获得更多的特征，对识别精度有较好的提升。

MobileNetV2 残差模块的输出采用了线性输出。因为激活函数在高维空间能有效增加非线性，而在低维空间时则会破坏特征，所以去掉了降维之后的激活函数。改进后的瓶颈层称为线性瓶颈层（Linear Bottleneck）。

线性瓶颈结构详细的内部组成如表 3–2 所示，其中 s 表示卷积核步长，t 表示线性瓶颈层内部的升维倍数，k 和 k' 分别为输入通道数和输出通道数。

表 3–2 线性瓶颈层内部组成

输入	算子	输出
$h\times w\times k$	1×1 conv2d, ReLU6	$h\times w\times(tk)$
$h\times w\times tk$	3×3 dwise s=s, ReLU6	$\frac{h}{s}\times\frac{w}{s}\times(tk)$
$\frac{h}{s}\times\frac{w}{s}\times tk$	Linear 1×1, conv2d	$\frac{h}{s}\times\frac{w}{s}\times k'$

3.4 轻量化骨干网络的性能分析与选择

2018 年旷视的研究人员提出了 ShuffleNetV2[47]，在 MobileNet 的基础上利用通道重组（Channel Shuffle）的方法减少了参数量，并且使各通道之间信息可以交换，

最后直接在目标平台上进行了性能度量。2019 年谷歌研究院的研究人员采用 NAS 的方法设计了 EfficientNet[51]系列模型，2020 年 3 月该小组又面向移动平台进行了优化设计，推出了 EfficientNet-Lite。

ShuffleNetV2 和 EfficientNet 的主要性能分析如表 3–3 所示，Top1 指标是指在 ImageNet 数据集上的准确率，是评价骨干网络性能的通用指标。

表 3–3 轻量级骨干网络性能分析表

网络名称	BFLOPs	ImageNet Top-1(%)
EfficientNet-B1[51]	0.70	78.8
EfficientNet-Lite0	0.407	75.1
EfficientNet-Lite1[90]	0.631	76.7
ShuffleNetV2 1x[47]	0.146	69.4
MobileNetV2[45]	0.3	72.0

ShuffleNetV2 的整体架构如图 3–5[47]所示。在综合分析了精度、参数量、计算量以及对移动端推理的友好程度后，本文选择了 ShuffleNetV2 1x 作为可选骨干网络之一，去掉最后一层卷积，并且抽取 8、16、32 倍下采样的特征图，也即 Stage2, Stage3, Stage4 的输出到 Lite-PAN 做多尺度的特征融合。

Layer	Output size	KSize	Stride	Repeat	Output channels			
					0.5×	1×	1.5×	2×
Image	224×224				3	3	3	3
Conv1	112×112	3×3	2	1	24	24	24	24
MaxPool	56×56	3×3	2					
Stage2	28×28		2	1	48	116	176	244
	28×28		1	3				
Stage3	14×14		2	1	96	232	352	488
	14×14		1	7				
Stage4	7×7		2	1	192	464	704	976
	7×7		1	3				
Conv5	7×7	1×1	1	1	1024	1024	1024	2048
GlobalPool	1×1	7×7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
# of Weights					1.4M	2.3M	3.5M	7.4M

图 3–5 ShuffleNetV2 的整体架构[47]

EfficientNet 作为目前性能最强的特征提取网络，采用 NAS 设计，也采用了 MobileNetV2 的线性瓶颈层作为自动设计的基本单元，其网络结构如表 3–4[51]所示。

表 3–4 EfficientNet 网络结构[51]

Stage	Operator	Resolution	Channels	Layers
1	Conv3×3	224×224	32	1
2	MBConv1, k3×3	112×112	16	1
3	MBConv6, k3×3	112×112	24	2
4	MBConv6, k5×5	56×56	40	2
5	MBConv6, k3×3	28×28	80	3
6	MBConv6, k5×5	28×28	112	3
7	MBConv6, k5×5	14×14	192	4
8	MBConv6, k3×3	7×7	320	1
9	Conv1×1&Pooling&FC	7×7	1280	1

为了在资源受限的边缘设备上运行，并解决硬件异构问题，对原本的 EfficientNet 进行了如下改进。去除在边缘设备上支持欠佳的 Squeeze-and-Excitation 结构；采用了量化技术，定点运算速度比浮点运算速度快，适用于算力有限的场景；用 ReLU6 替代所有 Swish 激活函数，从而显著提高训练后量化的质量。改进后的结构被称为 EfficientNet-Lite，由谷歌的研究人员在 2020 年 3 月完成。

EfficientNet-Lite 是目前性能最强的轻量级骨干网络，和前沿算法的性能对比如图 3–6 所示。在综合分析了精度、参数量、计算量以及对移动端推理的友好程度后，本文选择 EfficientNet-Lite1 作为高性能版本的骨干网络，去掉最后一层卷积，并且选取 Stage2, Stage4, Stage6 的输出特征图到 Lite-PAN 做多尺度的特征融合。

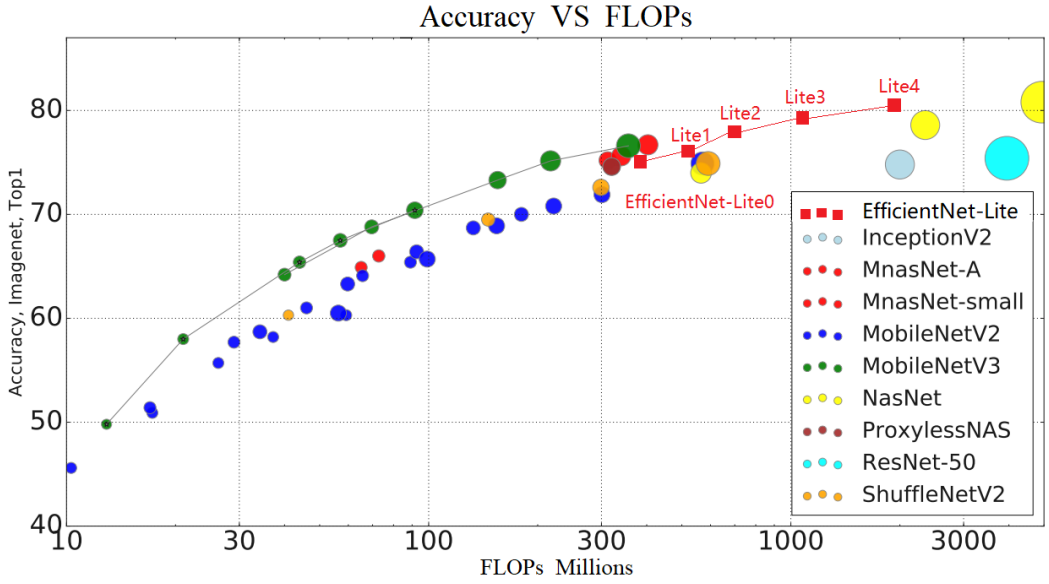


图 3–6 EfficientNet-Lite 网络性能对比（根据文献[50]绘制，增加了 EfficientNet-Lite）

本文采用骨干网络可插拔的设计，支持插拔不同的骨干网络。骨干网络采用在ImageNet 数据集上进行预训练的模型，直接加载 ImageNet 预训练权重，有利于模型快速收敛。

3.5 本章小结

本章首先对常见神经网络轻量化方法的优缺点进行了对比分析，并从理论和实际两个层面对神经网络轻量化的评价指标进行了研究；接着介绍了轻量级特征提取网络的结构特点和轻量化基础模块；最后对常见的轻量级特征提取网络的性能进行了对比分析，在综合分析了精度、参数量、计算量以及对移动端推理的友好程度后，在支持骨干网络可插拔的基础上选择了 ShuffleNetV2 与 EfficientNet-Lite 作为本文的骨干网络。

第 4 章 轻量化的多尺度特征融合网络研究

4.1 引言

多尺度特征融合一直是目标检测领域的一个研究重点。深度卷积神经网络的高层特征图，感受野比较大，包含较多的语义信息，有利于目标的分类；但是分辨率低，细节纹理信息表征能力弱，不利于目标的定位。低层特征图感受野小，对于几何细节表征能力强，特征图分辨率高，有利于目标的定位；但是语义信息欠缺，不利于目标的分类。对于小目标物体而言，由于自身分辨率较低，小目标物体的特征在深度神经网络向后传播的过程中会逐层削弱，然而若是只使用比较小的感受野去关注小目标本身的特征，将会丢失全局的语义信息。相反地，如果只使用比较大的感受野，将会忽略小目标物体的特征。因此，采用多尺度特征融合的思想，融合具有高分率的低层特征和高语义信息的高层特征，从而产生既有较高定位信息又有较高语义信息的特征，并在这些特征的基础上进行检测，可以获得较好的检测效果。但是，以前的融合网络结构复杂，参数量大不利于实际部署。本文提出了一个新的轻量化的多尺度特征融合的结构 Lite-PAN（Lite Path Aggregation Network），在保证融合效果的前提下，较之前的算法更加轻量化，计算量和参数量都有明显下降。

4.2 多尺度特征融合网络分析

特征金字塔网络[21]（Feature Pyramid Network，FPN）是第一个在目标检测中使用多尺度特征融合网络，之后出现了它的许多变体如 PANet[77]，NAS-FPN[78]，Bi-FPN[53]，Balanced FPN[23]，Recursive FPN[80]等。但是它们有着相同的基本组成，如不同尺度的骨干网络的特征图、横向连接，自顶向下连接，特征融合层等；也有着类似的融合策略，如对尺度较小的特征图先上采样再融合；以及都采用了多尺度预测的方法，就是在融合后的多个层上进行预测。本节就多尺度特征融合网络的结构和特征融合方法进行了研究，为设计轻量化的特征融合网络提供了指南。

4.2.1 网络结构分析

特征金字塔网络的基本结构如图 4–1 所示。图中 C2-C6 是来自骨干网络（特征提取网络）不同阶段输出的、不同尺寸的特征图。骨干网络在特征提取的过程中，每个下采样阶段特征图尺寸缩小一倍，在图中用“0.5×”表示。P2-P6 是特征融合后生成的特征图，下面以 P4 为例说明特征融合的过程。首先，来自骨干网络的 C4 经过一个 1×1 卷积进行通道数的缩减，这个过程也叫横向连接（Lateral Connection）；然

后将 P5 进行上采样；最后，将这两个特征图进行融合，经过一个 3×3 的卷积得到 P4。P2 和 P3 以同样的方式获得，P6 直接由 C5 下采样获得，P5 由 C5 经横向连接获得。

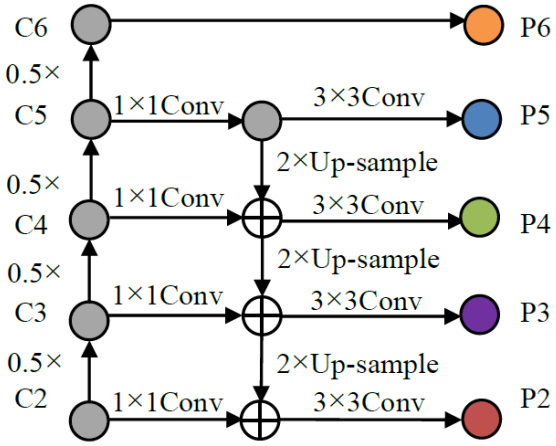


图 4–1 FPN 网络结构示意图（根据文献[21]绘制）

FPN 融合了多尺度的信息，提高了目标检测的检测效果，启发研究人员对卷积神经网络有了更深的认识。但是 FPN 还存在一些问题，比如高层级特征与低层级别特征之间路径过长，增加了访问准确位置信息的难度；特征融合能力不够等。为了解决这些问题，PANet[77]在 FPN 的基础上，创建了自底向上的路径增强（Path Aggregation），用来缩短信息路径，并且利用低层级的准确定位信息来增强特征金字塔，如图 4–2 (a) 所示。PANet 是第一个提出先自底向上，再自顶向下，双向融合的模型。在 PANet 的基础上继续改进，研究人员又提出了 NAS-FPN[78], Bi-FPN[53]等变种，他们的主要结构如图 4–2 所示。

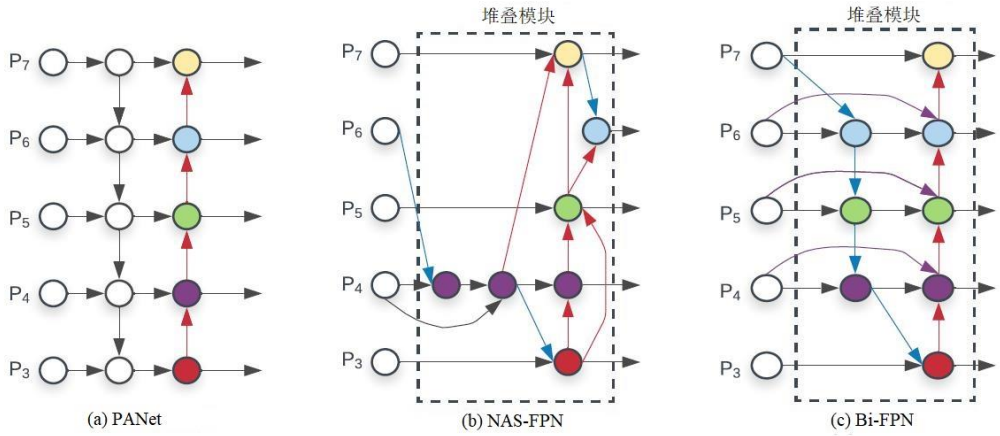


图 4–2 FPN 变种的结构分析[53]

NAS-FPN[78]使用神经网络架构搜索，搜索到不规则的网络拓扑，然后级联使用相同的“堆叠模块”。基于 AmoebaNet 的 NAS-FPN 检测器需要 167 M 的参数量和 3045 BFLOPs（比 RetinaNet 多 30 倍）的运算量。庞大的模型尺寸和昂贵的计算成本阻碍

了它们在许多现实世界应用程序中的部署，比如机器人和自动驾驶汽车，在这些应用程序中，模型尺寸和延迟受到了高度限制。并且该网络拓扑结构不规则，可解释性较差。

Bi-FPN[53]“堆叠模块”的设计原则是：“如果某一节点只有一条输入边，则移除该节点。”因为，如果一个节点只有一条未进行特征融合的输入边，那么它对旨在融合不同特征的特征融合网络的贡献就会更小。另外 Bi-FPN 中还引入了跨连接融合，并且级联了 3 个相同的“堆叠模块。Bi-FPN 以 EfficientDet 为骨干网络，模型相比 NAS-FPN 轻量了许多。但其中连接复杂，且采用了级联的结构，不符合本文的轻量化设计原则。

本文选择以结构规则，连接简单的 PANet 为基础，提出了 Lit-PAN 模型，比 Bi-FPN 更加轻量，对低端的嵌入式设备更加友好。4.3.1 节对 Lite-PAN 的结构进行了详细介绍。

4.2.2 特征融合方法分析

高层低分辨率特征图经过上采样与低层高分辨率特征图进行特征融合。其中，常用的上采样方法包括，双线性插值、最近邻插值、转置卷积等。特征融合方法主要有，通道拼接[81]（Concat），逐元素相乘[34]（Element-wise Multiply）以及逐元素相加[21]（Element-wise Sum）等，如图 4–3 所示。

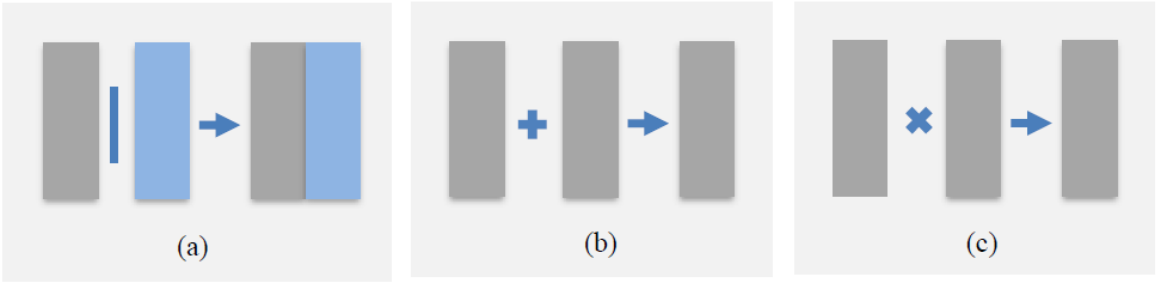


图 4–3 常用特征融合方法示意图， (a) 通道拼接 (b) 逐元素相加 (c) 逐元素相乘

通道拼接操作将低层特征图和高层特征图的通道合并，逐元素相乘操作将低层特征图和高层特征图矩阵的对应元素进行逐元素相乘，逐元素相加操作将低层特征图和高层特征图矩阵的对应元素进行逐元素相加。

按照深度学习常用框架中的表示方式，采用 4 维矩阵来表示数据，将图像特征表示为 (*B, C, W, H*)，其中 *B* 表示 Batch 数，即每次有 *B* 张图片用于训练，*C* 表示通道的数量，和图片的通道数定义一致，*W*，*H* 表示特征图的尺寸。

在进行特征融合的计算时要考虑上述 4 维矩阵中，每一个维度的匹配问题。其中，逐元素相加和逐元素相乘的方法，需要各个维度的大小都一致的。

比如，图 4–1 中的 *C4* 和 *P5* 融合，就要求满足公式（4–1）。

$$\begin{cases} B_{C4}=B_{P5} \\ C_{C4}=C_{P5} \\ W_{C4}=W_{P5} \\ H_{C4}=H_{P5} \end{cases} \tag{4-1}$$

各个维度匹配之后，就可采用逐元素相加或者逐元素相乘的方法进行特征融合。表示成公式（4–2）或公式（4–3），式中分别用 EltwSum 和 EltwMul 表示逐元素相加和逐元素相乘的融合过程。

$$P4 = \text{EltwSum}\ (C4,\ P5) \tag{4-2}$$

$$P4 = \text{EltwMul}\ (C4,\ P5) \tag{4-3}$$

逐元素相加和相乘的方法要求各个维度大小均一致，条件比较严格。如果通道维度 C 不一致，如公式（4–4）所示。仍以图 4–1 为例，这时可采用通道拼接的方法，把 $P5$ 拼接到 $C4$ 后面，或者 $C4$ 拼接到 $P5$ 后面，拼接后得到的 $P4$ 的通道维度为 $C_{C4}+C_{P5}$，最后得到 P4 的维度为 $(B_{P4},\ C_{C4}+C_{P5},\ W_{P4},\ H_{P4})$。

$$\begin{cases} B_{C4}=B_{P5} \\ C_{C4}\neq C_{P5} \\ W_{C4}=W_{P5} \\ H_{C4}=H_{P5} \end{cases} \tag{4-4}$$

4.3 轻量化改进的特征融合网络

4.3.1 Lite-PAN 的网络结构

通过对常见特征融合网络结构的对比分析，本文提出了轻量化的特征融合网络 Lite-PAN。其网络结构如图 4–4 所示。

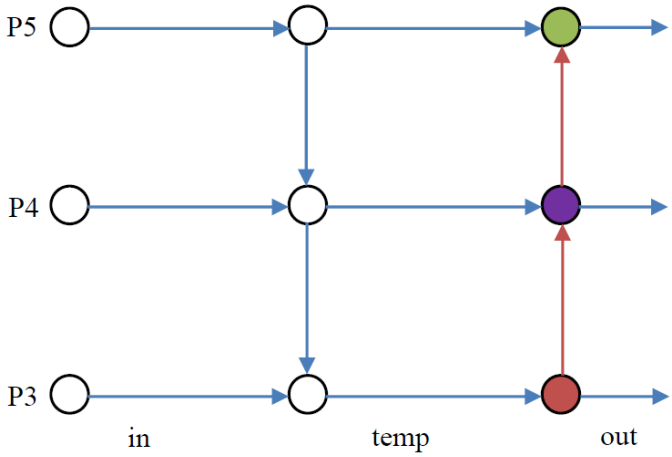


图 4–4 Lite-PAN 的网络结构

图中 P^{in} 表示输入到 Lite-PAN 的特征图的集合：

$$P^{in}=\left(P_3^{in}, P_4^{in}, P_5^{in}\right) \tag{4-5}$$

P_i^{in} 表示第 i 个特征图，它的尺寸是输入图像的 $1/2^i$。

Lite-PAN 自顶向下融合的过程采用传统的 FPN 结构，该部分的输出作为该网络的中间值（temp），表示为 P_i^{temp}。自顶向下部分的融合过程如公式（4–6）所示，其中 *Resize* 表示分辨率的匹配，既可以表示上采样也可以表示下采样。*Conv* 表示卷积处理的过程。

$$\begin{cases} P_5^{temp}=Conv\left(P_5^{in}\right) \\ P_4^{temp}=Conv\left(P_4^{in}+Resize\left(P_5^{temp}\right)\right) \\ P_3^{temp}=Conv\left(P_3^{in}+Resize\left(P_4^{temp}\right)\right) \end{cases} \tag{4-6}$$

Lite-PAN 自底向上融合的过程如公式（4–7）所示，P_i^{out} 表示 Lite-PAN 的输出。

$$\begin{cases} P_3^{out}=Conv\left(P_3^{temp}\right) \\ P_4^{out}=Conv\left(P_4^{temp}+Resize\left(P_3^{out}\right)\right) \\ P_5^{out}=Conv\left(P_5^{temp}+Resize\left(P_4^{out}\right)\right) \end{cases} \tag{4-7}$$

通过自底向上的网络通路利用底层精确的定位信息增强了网络的特征表达能力缩短了底层与最顶层特征之间的信息路径。与自顶向下的融合路径相呼应，达到了分类任务和定位任务之间的平衡。

4.3.2 Lite-PAN 的特征融合方法

DSSD[34]采用解卷积（Deconvolution）也就是转置卷积的方法实现上采样，但是解卷积的卷积核也是需要学习的，同样带有权重，采用解卷积实现上采样势必会引入额外的参数，不利于模型的轻量化。

本文以轻量化为设计原则，在自顶向下的融合过程中，上采样操作采用双线性插值实现，实现小尺度特征图与大尺度特征图的匹配。

原版的 PANet 和 YOLOv4 在多尺度特征中，采用步长为 2 的卷积进行下采样，实现大尺度特征图与小尺度特征图的匹配。而本文在自底向上的融合过程中，下采样操作同样由插值实现，进一步减少了参数量。整个 Lite-PAN 只保留了 1×1 的卷积用于特征维数的匹配。

在融合阶段本文选择逐元素相加（Elw-Sum）的方法，将多尺度特征图按位相加，减少了特征融合模块的计算量。本文提出的 Lite-PAN 的特征融合方法如图 4–5 与图 4–6 所示，其中参数设置以骨干网络为 ShuffleNetV2 时为例。

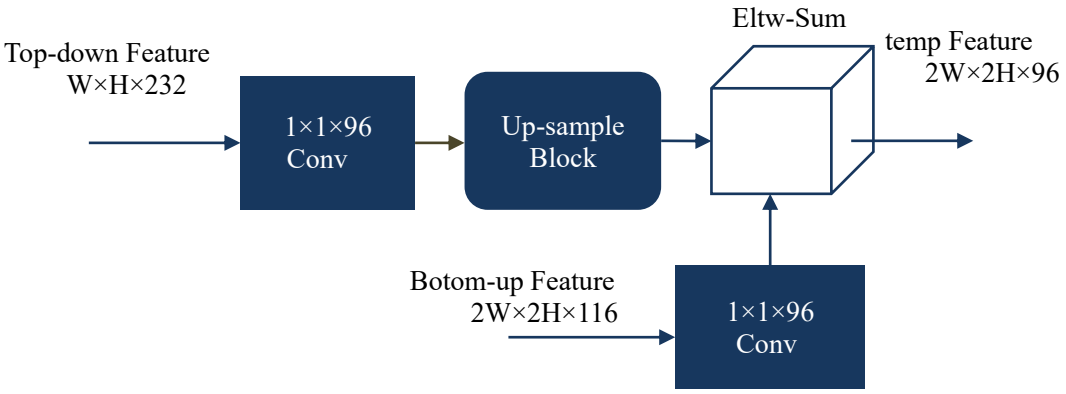


图 4–5 Lite-PAN 特征融合方法，自顶向下阶段，以 P_3^{in} 与 P_4^{temp} 融合得到 P_3^{temp} 为例

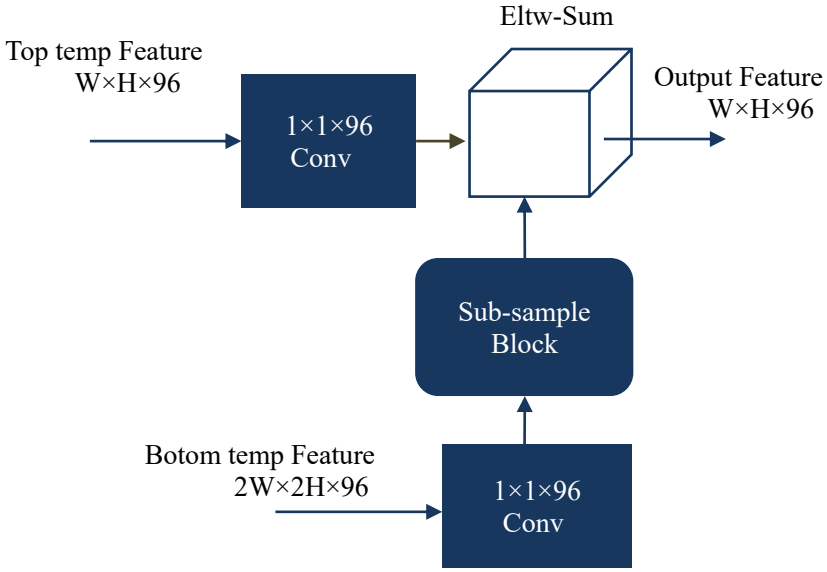


图 4–6 Lite-PAN 特征融合方法，自底向上阶段，以 P_3^{out} 与 P_4^{temp} 融合得到 P_4^{out} 为例

4.3.3 在本文轻量化框架中的应用

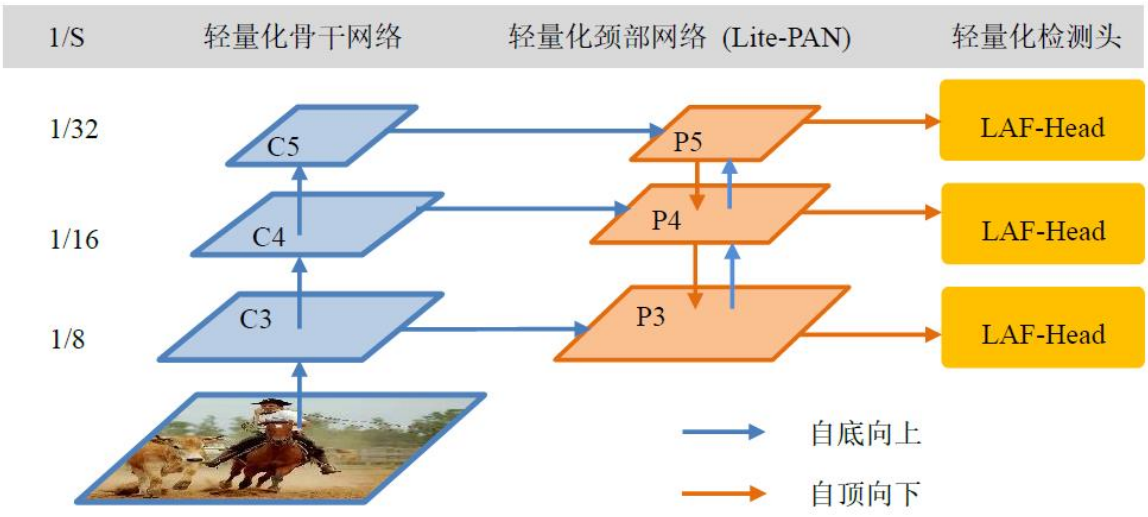


图 4–7 Lite-PAN 在本文轻量化框架中的应用

Lite-PAN 位于轻量化骨干网络与检测头之间，对骨干网络提供的多尺度特征图进行融合，输出不同分辨率的融合后的特征图。由检测头在不同尺度的特征图上进行分层级独立预测，如图 4–7 所示。其中轻量化检测头的设计详见本文第 5 章。

4.4 实验分析

4.4.1 实验环境

本文实验均在实验室深度学习服务器上完成，该服务器配置了 8 张 NVIDIA TITAN RTX 显卡，详细配置参数如表 4–1 所示。其中，CUDA 是 NVIDIA 推出的适用于 NVIDIA GPU 的并行计算语言。CuDNN 是 NVIDIA 开发的针对深度学习的计算加速库。

表 4–1 服务器配置

配置	版本
CUDA	11.0
cudatoolkit	10.1
CuDNN	8.0.3
系统	Ubuntu18.04
深度学习框架	PyTorch1.7.0

4.4.2 实验参数设置

本章实验的主要目的是验证特征融合网络 Lite-PAN 的性能，为了控制变量，方便对比，对于检测头的设计依然使用有锚框的预测结构。

本章采用三层级预测的结构，各层级锚框设置按照 SSD[3]中的方法，如公式（4–8）所示，随着特征图尺寸减小，锚框尺寸线性增加。

$$S_k=S_{min}+\frac{S_{max}-S_{min}}{m-1}(k-1),\ k\in[1,m] \tag{4-8}$$

其中，m 表示独立预测的特征图的个数，本文设置为 3；S_k 表示锚框大小相对于原图的比例；S_{max} 与 S_{min} 分别表示比例的最大值和最小值。分层预测时锚框的宽高比设置为（1，2，3，1/2，1/3）。

本章实验训练阶段在 4 张显卡上进行，每张显卡的 BatchSize 设置为 96，采用带动量的 SGD 优化器。训练集选择 COCO2014 训练集及部分验证集，测试集选择 COCO2014 验证集部分数据。采用水平翻转、随机缩放、以及色彩抖动的操作进行数据增强。

4.4.3 消融实验

为验证 Lite-PAN 结构的有效性，本文在 COCO 数据集上进行了消融实验。本文对比算法和本文算法均采用 ShuffleNetV2 1x 作为骨干网络，采用 416×416 的输入分辨率。然后，对比算法不加入 Lite-PAN 机制，本文的算法加入 Lite-PAN 机制。本文的算法既有自顶向下的融合路径，又有自底向上的融合路径，命名为 LMDet（Lightweight Multiscale Detector）。实验结果如表 4–2 所示。

表 4–2 有无 Lite-PAN 融合机制对算法性能的影响

算法名称	骨干网络	分辨率	融合机制	mAP
Light-Head R-CNN[87]	ShuffleNetV2 1x	416*416	无	22.5%
LMDet	ShuffleNetV2 1x	416*416	Lite-PAN	23.5%

在以 ShuffleNetV2 1x 为骨干网络的情况下引入 Lite-PAN 机制以后，算法的平均精度提升了 1%。Lite-PAN 机制的引入使得多尺度的特征充分融合，高层特征引导低层网络更好的关注细节信息，低层的网络指导高层的网络更好的学习语义信息，达到了分类任务和定位任务之间的平衡，有利于不同大小物体的检测。并且多层级预测的结构使得不同尺寸的目标在不同层级上预测，很好的解决了重叠遮挡造成的漏检和误检问题。可视化的检测效果如图 4–8 所示。

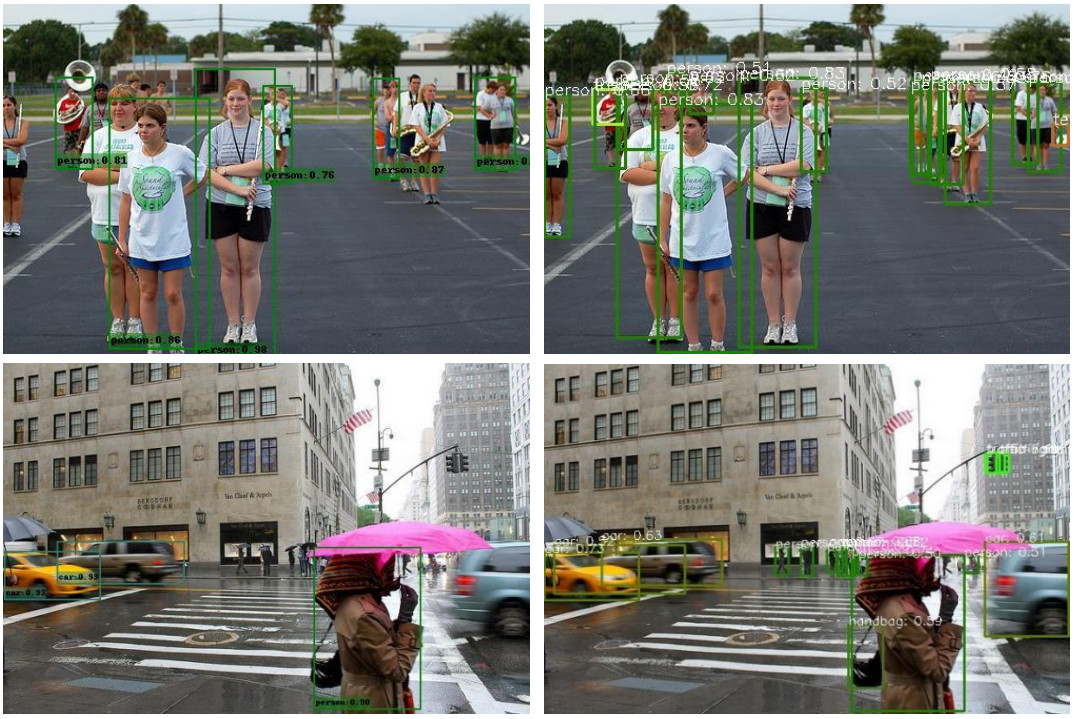


图 4–8 引入 Lite-PAN 机制前后算法检测效果对比图，左侧未加入 Lite-PAN 模块，右侧加入 Lite-PAN 模块

4.5 本章小结

本章对轻量化的多尺度特征融合网络进行了研究，提出了轻量化的特征融合策略，设计了 Lite-PAN 这一轻量化的颈部网络结构。最后对 Lite-PAN 的各项性能进行了实验分析。实验表明，加入该轻量化的网络结构对于不同尺度的目标具有很好的检测效果，性能超过了未采用该机制的网络模型，由于采用了分层预测的机制，很好的解决了严重遮挡、重叠的目标的检测问题。

第 5 章 无锚框的轻量化小目标预测方法研究

5.1 引言

目前所有的主流目标检测算法，如 R-CNN 系列, SSD 系列和 YOLO 系列都依赖于一组预先定义的锚框（Anchor），在锚框的基础上“微调”生成预测框，收敛更快，效果更好，锚框一直被认为是目标检测成功的关键。然而基于锚框的算法有五大缺点：

(1) 锚框的尺寸和长宽比是固定的，对于目标尺寸的变化没有较好的鲁棒性，难以应对尺寸变化较大的情况，尤其是小目标较多的情况。

(2) 预先定义的锚框，依赖于对数据集的统计，阻碍了检测器的泛化性能，遇到新的检测任务，需要对锚框的大小、数量和宽高比进行重新设计。

(3) 锚框的使用引入了许多超参数，当这种做法和多尺度预测结构相结合时，会变得更加的复杂，更重要的是检测结果对这些超参数敏感，比如在 RetinaNet 中改变这些超参数，会影响其在 COCO 上 4%的 AP 性能[27]。

(4) 为实现较高的召回率，需要大量的锚框来确保真实框被足够的覆盖。如 DSSD 需要超过 40K 个，FPN 需要超过 180K 个。锚框同样涉及复杂的计算，比如计算锚框和真实框的交并比。

(5) 实际中只有小部分锚框会和真实框相重叠，这造成了正负样本之间的巨大的不均衡。由于小目标物体尺寸的原因，分配到的正样本框数量更少，不利于小目标物体的训练。

因此，本文采用无锚框（Anchor-Free）的设计思想，提出了轻量化无锚框的检测头部网络 LAF-Head（Lite Anchor-Free Head）。

5.2 基于中心回归的轻量化无锚框检测头设计

5.2.1 方法背景

常见的无锚框预测方法都是基于关键点或中心的，如 CornerNet[27]利用一对原始边界框的角点，ExtremeNet[83]利用上下左右四个关键点，来预测边框位置。虽然类似于上述两种基于关键点的检测方法能有很好的鲁棒性，但是需要进行大量的后处理，包括对关键点之间的关联性分析，对于同属于一个物体的关键点进行划分等。后处理过程降低了算法的效率，迟滞了前向处理的过程。

后来 CenterNet[27]提出了直接检测物体中心点和尺寸的思想，每个物体仅有一个正样本点，不需要后处理，仅通过提取热图（Heatmap）上局部最大值的位置即可。

并且没有手动区分前景背景的阈值，通过高斯分布生成监督信息,越远离中心点监督信息越接近于 0。但这种做法监督信息较少，训练时间较长。

同样基于中心回归的思想，FCOS[30]首先将图像划分成网格，然后以落在真值框内的网格的中心为正样本，而不是使用和真值框超过 IoU 阈值的锚框作为正样本，如图 5–1 所示。上述无锚框算法的优缺点对比如表 5–1 所示。

表 5–1 具有代表性的无锚框检测算法总结

检测模型	检测机制	优点	缺点	使用范围
CornerNet[26]	一对角点	Corner Pooling，Embeddings，提高模型精度	角点分组难度和计算量较大，对边缘敏感，忽略内部信息	多目标检测
FCOS[29]	点+点到四边的距离	多尺度预测解决目标模糊性设计复杂度低	多尺度监督信息需增强，中心度可解释性需增强	可用于实例分割
CenterNet[28]	一个中心点+长宽	简单高效没有 NMS 后处理	只使用中心点回归可获得监督信息较少	可用于 2D, 3D 目标检测及人体姿态估计

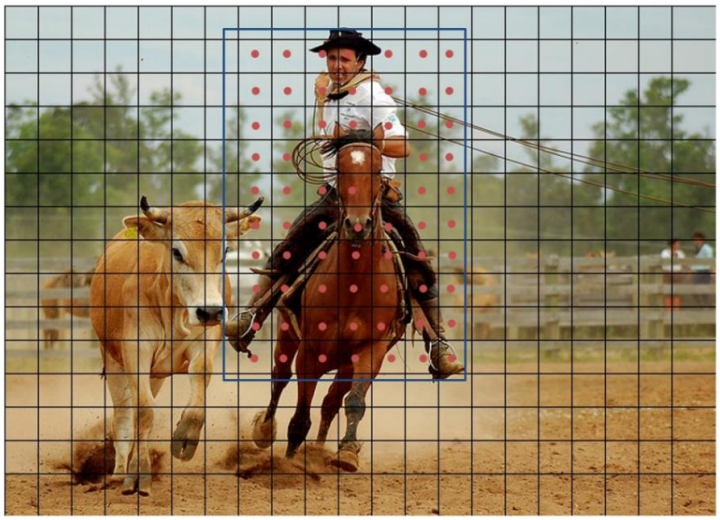


图 5–1 基于中心的回归方法—中心的选择，深蓝色边框为真实框，红色点为落在真值框中的网格的中心点，也即正样本点，网格为特征图映射回原图的形象表示

5.2.2 基于中心的回归方法

通过对比分析典型的无锚框检测算法的优缺点，本文选择了基于中心的回归方法，中心的选择如图 5–1 所示，将落入真实框中的网格中心点作为正样本点，直接对网格中心到真实框四边的距离进行回归。

由于预测是在特征图上进行的，而真实框的标注是对原图的标注，并且最终的预测的结果要映射回原图。所以需要沟通从特征图到原图的映射关系。

设 $F_i \in R^{H\times W\times C}$ 为骨干网络的第 i 层特征图，s 为从特征图映射回原图的步长。输入图像中的真值框表示为 $B_i=\left(x_0^{(i)}, y_0^{(i)}, x_1^{(i)}, y_1^{(i)}, c^{(i)}\right)\in R^4\times\{1,2,\ldots,C\}$。其中 $\left(x_0^{(i)}, y_0^{(i)}\right)$ 与 $\left(x_1^{(i)}, y_1^{(i)}\right)$ 分别表示边界框的左上角坐标与右下角坐标，$c^{(i)}$ 表示边界框中目标的类别。C 表示类别数，对于 COCO 数据集而言，C=80。对于特征图 F_i 中的每个坐标点 (x, y)，将其映射回原图表示为公式（5–1），如图 5–2 所示。

$$\left(\left\lfloor\frac{s}{2}\right\rfloor+xs, \left\lfloor\frac{s}{2}\right\rfloor+ys\right) \tag{5-1}$$

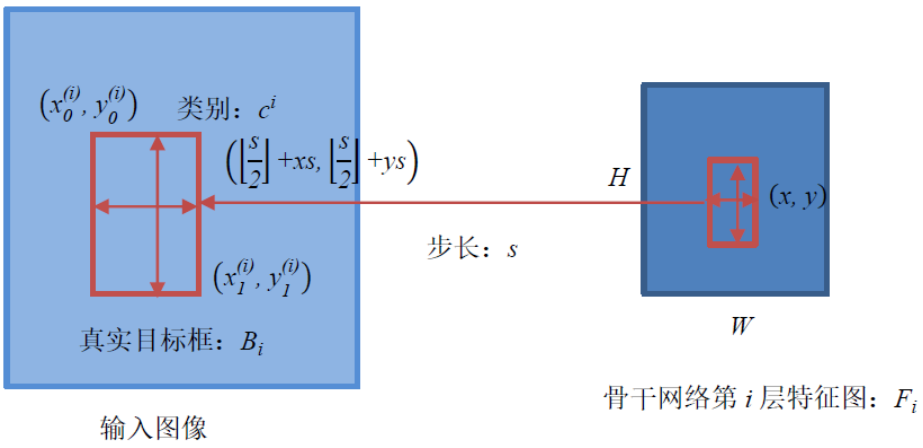


图 5–2 从特征图映射回原图

本文用一个四维实向量 $\boldsymbol{t}^*=(l^*, t^*, r^*, b^*)$ 作为位置的回归目标。其中，l^*, t^*, r^*, b^* 为从网格中心到真实框四边的距离，如图 5–3 所示。

图 5–3 基于中心的回归方法—回归目标

如果一个位置有多个边界框，也即网格中心像素点落在两个真实框的重叠区域，则认为它是一个模糊样本，这时选择面积最小的边界框作为回归目标。在 5.2.3 中节，将展示本文的轻量化多层预测结构。采用多层级预测的方式，在不同层级预测不同大小的目标，模糊样本的数量可以显著减少，因此它们几乎不影响检测性能。如果位置坐标 (x, y) 与一个真实框 B_i 相关联，该位置的训练回归目标可表示为公式（5–2）。

$$\begin{cases} l^*=x-x_0^{(i)} \\ t^*=y-y_0^{(i)} \\ r^*=x_1^{(i)}-x \\ b^*=y_1^{(i)}-y \end{cases} \tag{5-2}$$

基于锚框的算法只把 IoU 大于阈值的候选框作为正样本，本文可以利用更多的前景信息来训练网络，这也是无锚框的算法表现优于基于锚框的同类算法的原因之一。

对于分类部分，同样落入真值框内的网格的中心点将被视为为正样本，并且该位置的类别标签 c^* 为真实框的类别标签。否则，它是一个负样本并且 $c^*=0$（背景类）。与训练过程相对应，网络的最后一层输出一个 80 维（COCO 数据集）的类别标签向量 $\boldsymbol{c}$ 和一个 4 维的边框位置向量 $\boldsymbol{t}=(l, t, r, b)$。

在深度神经网络模型训练的过程中，通过计算损失函数的值并使其最小化来得到训练好的模型。目标检测任务中的损失函数通常包括两部分，分类损失和定位损失，本文的损失函数如公式（5–3）所示。

$$\begin{aligned} \mathrm{L}\left(\{\boldsymbol{c}_{x,y}\},\{\boldsymbol{t}_{x,y}\}\right)&=\frac{1}{N_{pos}}\sum_{\mathrm{x,y}} L_{cls}\left(\boldsymbol{c}_{x,y}, \boldsymbol{c}^*_{x,y}\right) \\ &+\frac{\lambda}{N_{pos}}\sum_{x,y} I_{\{c^*_{x,y}>0\}} L_{reg}\left(\boldsymbol{t}_{x,y}, \boldsymbol{t}^*_{x,y}\right) \end{aligned} \tag{5-3}$$

其中 $\boldsymbol{c}_{x,y}$ 与 $\boldsymbol{t}_{x,y}$ 表示网络的预测值，$\boldsymbol{c}^*_{x,y}$ 与 $\boldsymbol{t}^*_{x,y}$ 表示真实值。分类损失 L_{cls} 是广义 Focal Loss[85]，这点和原版的 FCOS 不同。回归损失 L_{reg} 采用 IoU Loss[65]。N_{pos} 表示正样本数，λ 为了平衡分类和回归损失，一般取值为 1。在特征图 F_i 上计算所有样本点损失的总和。$I_{\{c^*_{x,y}>0\}}$ 是指示函数，当 $c^*>0$ 时，该指示函数值为 1，其他情况该值为 0。

本文采用广义 Focal Loss 替换 Focal Loss 的主要目的是为了轻量化设计，详见 5.2.3 节。

5.2.3 轻量化的预测分支

近年来，单阶段检测算法的发展趋势是引入一个独立的预测分支来估计定位质量，预测的质量有助于分类，从而提高检测性能。比如 YOLOv3 中的 Objectness 分支[1]，FCOS 中的 Centerness 分支[30]等。也即将预测结构分为三个基本分支，分类、定位和预测框质量估计分支，如图 5–4 所示。

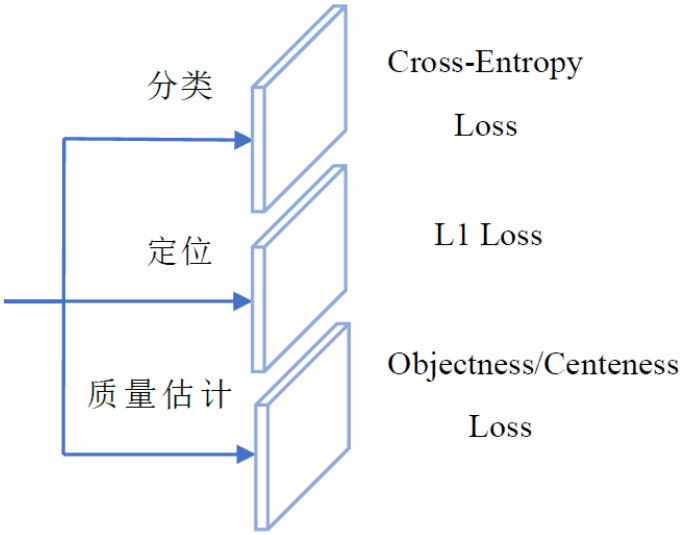


图 5–4 预测结构的三个基本分支

文献[85]指出了在训练和推理阶段的预测框质量估计和分类评分使用不一致的问题。也即，在目前的目标检测算法中，预测框质量估计和分类评分通常是独立训练的，但在推理过程中是综合使用的，比如将预测框质量得分与分类得分相乘。

并且在实际的训练中发现，预测框质量估计分支在轻量级的模型上很难收敛。并且有的预测框质量估计分支对于小目标并不友好，比如文献[30]中的中心度分支（Centerness），如图 5–5 所示。

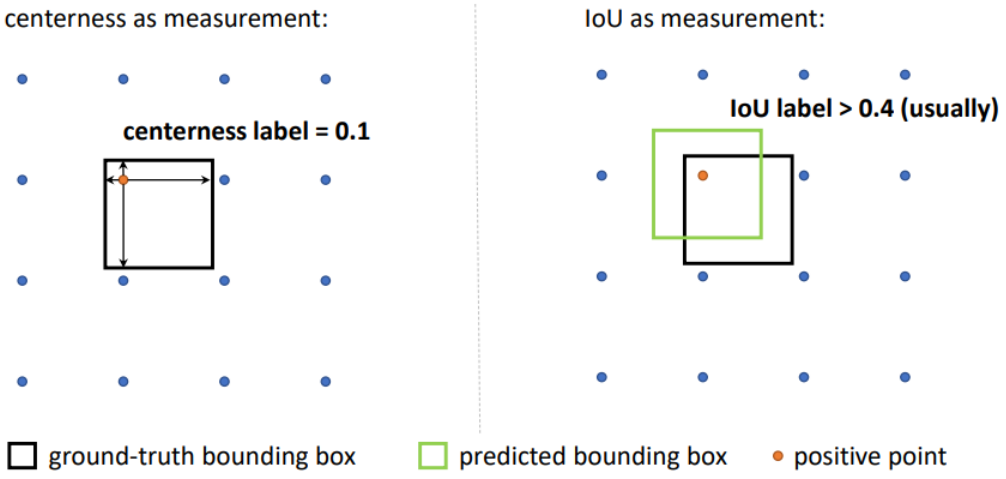


图 5–5 真值框与正样本点的可能情况[30]

图 5–5 中矩阵点表示 Stride=8 的特征层（P3 层），映射回原图划分的网格的中心点。比如有一个小目标落在了图 5–5 黑色框的位置，这时根据中心度的定义计算得到的中心度得分极度小，甚至接近于 0。

也即预测框质量估计得分很小，再后续与分类得分相乘，由于两者相乘的积很小，经过 NMS 过滤，这个检测框就会丢失，造成漏检情况的发生。

质量估计分支存在上述缺陷，并且该分支本身就占有大量卷积运算，增加了检测头的计算开销。因此如何将质量估计分支去掉，或者将其和其他两个分支之一合进行合并，成为了避免上述缺陷的关键解决方案。

本文采用了将质量估计分支与分类分支合并的思想，将预测框质量估计得分合并到分类预测向量中，形成预测框质量和分类的联合表示，并使用另外一个向量来表示边框的定位。

由于采用了分类-质量估计联合的表示，标签变成了 0~1 之间的连续值。这时我们既要保证 Focal Loss 此前的平衡正负、难易样本的特性，又需要让其支持连续数值的监督。

这时原来的 Focal Loss 便不再适用，文献[85]将 Focal Loss 从离散推广到了连续，得到了广义 Focal Loss。本文引入广义 Focal Loss[85]作为合并分支的损失函数。改进后的预测结构表示如图 5–6 所示。

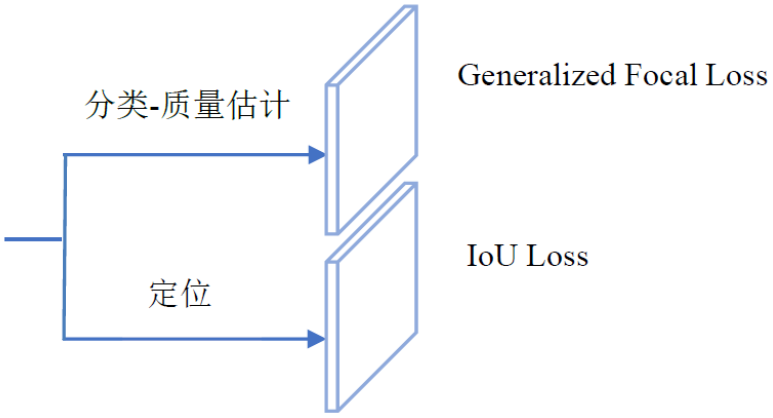


图 5–6 分类-质量估计分支合并的预测结构

分类分支和质量估计分支合并后的预测结构，不仅避免了质量估计分支的缺陷，还省去了质量估计分支的大量卷积，减少了预测部分的计算开销，从而使模型更加轻量化。

5.2.4 轻量化的多层级预测

本文在 Lite-PAN 输出的三个不同层级的特征图上进行独立预测，不同于基于锚框的算法将不同大小的锚框分配给不同的特征层次，本文直接限制了每个层次的锚框回归的范围。

更具体地说，本文首先计算所有特征层级上每个位置的回归目标 (l^*, t^*, r^*, b^*)。接下来，如果一个位置满足 $max(l^*, t^*, r^*, b^*)>m_i$ 或者 $max(l^*, t^*, r^*, b^*)<m_{i-1}$，它被设置为一个负样本，因此不再需要回归一个边界框。这里 m_i 是特征层次 i 需要回归的最大距离。在本文中，m_2、m_3、m_4、m_5 分别设为 0、64、128、256。由于不同大小的目标被分配到不同的特征级别，并且大多数重叠发生在大小相当不同的目标之间。如果一个位置，即使使用多级预测，仍然被分配给多个真实框，我们只需选择面积最小的真值框作为目标。

本文的检测头部网络是在 FCOS 检测头的基础上进行轻量化改进的，原版的 FCOS 检测头如图 5–7[30]所示。

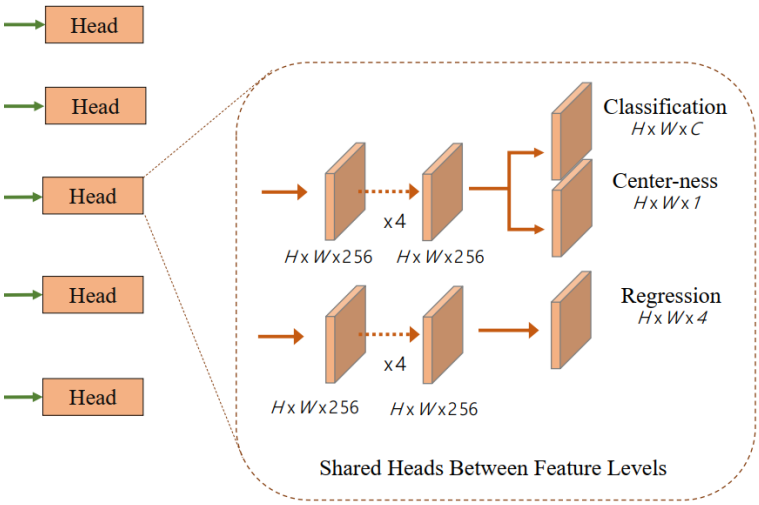


图 5–7 FCOS 中的检测头部网络[30]

在 5.2.3 节进行了分类和质量估计分支合并的基础上，基于轻量化的设计原则，本文又进行了以下三点改进。

第一，使用深度可分离卷积替换了普通卷积，并且将卷积级联的个数从 4 组减少为 2 组。并且将通道数从将 256 维压缩至 96 维，在以 EfficientNet 为骨干网络时压缩至 128 维。

第二，将组归一化替换为批归一化[86]。相对于批归一化（Batch Normalization，BN），组归一化（Group Normalization，GN）的使用有很多优点。但是 BN 操作可以和卷积融合，在推理阶段，BN 能够将其归一化的参数直接融合进卷积，从而省去 BN 的计算，而 GN 不可以。为了省去归一化操作的时间，本文选择使用 BN 替换 GN。

第三，在不同的层级的特征之间不共享权重。FCOS 系列使用了共享权重的检测头，对特征金字塔网络输出的多尺度特征图使用同一组卷积预测检测框，接着每一层使用一个可学习的缩放值作为系数，对预测出来的框进行缩放，从而实现检测框从特征图到原图的映射。

采用共享权重的方法可以降低网络的参数量，这对于有数百个通道的大模型非常有用。但是对于轻量化的模型，共享权重的检测头并没有很大意义。

首先，轻量化检测头本身卷积是轻量化的卷积，共享权重并不会带来很大的参数量约减；其次，共享权重会带来检测能力的下降，对于轻量化的网络更是如此，共享权重使得其检测能力进一步下降；最后，轻量化的模型目前主要面向 CPU 部署，共享权重并不会对推理过程产生加速。轻量化改进后的检测头命名为 LAF-Head（Lite Anchor-Free Head），如图 5–8 所示。

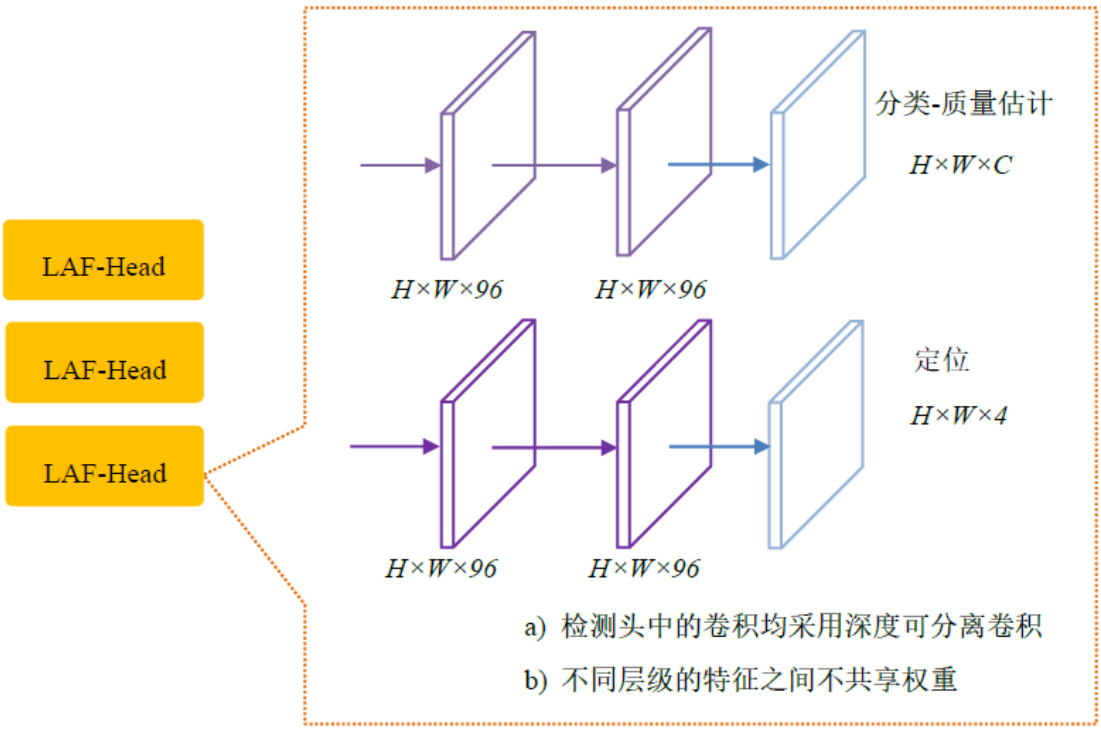


图 5–8 本文轻量化改进的检测头 LAF-Head

借助神经网络模型结构可视化工具，将检测头部网络结构可视化，如图 5–9 所示，图中 dw 表示深度卷积，pw 表示逐点卷积，为深度可分离卷积的两个部分，图中 96 代表卷积通道数为 96。

至此，本文提出的单阶段无锚框轻量化小目标检测算法 LMDet-S（Lightweight Multiscale Detector for Small Object）的三大模块（Backbone，Neck，Head）的设计全部完成。5.3 节将对本文提出的模块的有效性进行实验验证，并对算法整体性能进行实验分析。

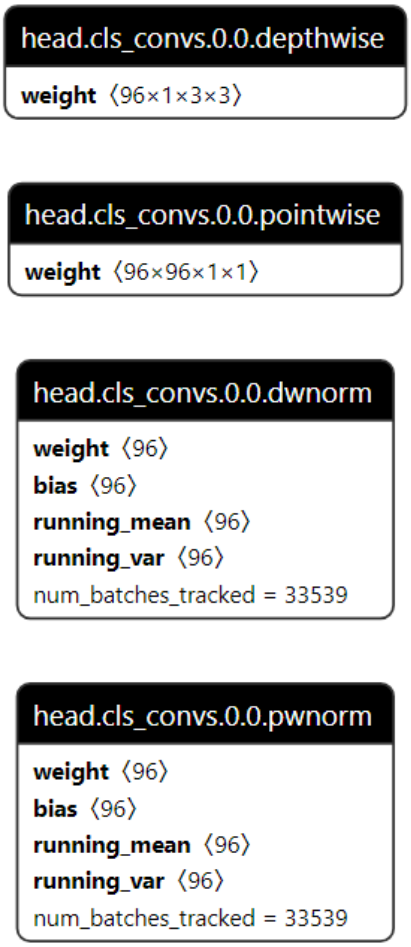


图 5–9 轻量化后的检测头部网络模型

5.3 总体实验

5.3.1 实验参数设置

在 4.4.1 节所述的实验环境下选用 4 张显卡，在 COCO 数据集上进行实验。

以 ShuffleNetV2 为骨干网络时，每张显卡的 BatchSize 设置为 96，采用 SGD 优化器，采用按步衰减的学习率调整策略，初始学习率设置为 0.14，动量和权重衰减分别设置为 0.9 和 0.0001，选用 LeakyReLU 激活函数。

以 EfficientNet-Lite 为骨干网络时，每张显卡的 BatchSize 设置为 64，也采用 SGD 优化器，采用按步衰减的学习率调整策略，初始学习率设置为 0.07，动量和权重衰减分别设置为 0.9 和 0.0001，选用 ReLU6 为激活函数。

上述两种情况均采用水平翻转、随机缩放、以及色彩抖动的操作进行数据增强。未使用可变形卷积、空间金字塔池化等其它优化方法。

5.3.2 目标检测性能实验

为了保证对比的公平性，本章所有基础设置与对比算法确保一致。该节实验主要探讨引入轻量化无锚框检测头 LAF-Head 对于目标检测性能，特别是小目标检测性能的影响。对比方法采用的是基于锚框的方法。没有特别说明的话，本节采用 EfficientNet-Lite 1x 作为骨干网络。表 5–2 统计了引入 LAF-Head 对目标检测精度的影响。

表 5–2 LAF-Head 对目标检测精度的影响-横向比较（单位：%）

模型	分辨率	mAP	AP_{50}	AP_S	AP_M	AP_L
YOLOv4-Tiny[54]	416*416	21.7	42.1	10.2	26.3	30.9
LMDet-S-416	416*416	30.3	47.1	12.2	32.2	43.1

与 YOLOv4-Tiny[54]的对比实验显示，本文由于采用了轻量化无锚框检测头，平均精度提高了 8.6%，小目标检测精度提高了 2%。

表 5–3 讨论了不同分辨率对目标检测精度的影响。从表中可以看出，随着输入图片分辨率的增加，检测精度也相应提高。原因在于随着输入图片分辨率的增加，输入网络的信息也在增多。

表 5–3 不同输入分辨率对检测精度的影响（单位：%）

模型	分辨率	mAP	AP_{50}	AP_S	AP_M	AP_L
LMDet-S-512	512*512	32.5	50.1	15.2	34.2	48.1
LMDet-S-416	416*416	30.3	47.1	12.2	32.2	43.1

该实验表明，在硬件条件允许的情况下，增加输入图像的分辨率，对于检测模型精度的提升也有很大的作用，可以得到更好的检测效果。

图 5–10 展示了在远景小目标的情形下，本文算法能够检测出更多的小目标。

图 5–10 远景小目标检测效果对比图，左侧为 YOLOv4-tiny，右侧为 LMDet-S

图 5–11 展示了在存在大量密集物体的情况下，本文算法有更好的检测效果。

图 5–11 密集小目标检测效果对比图，左侧为 YOLOv4-tiny，右侧为 LMDet-S

可以看出本章加入 LAF-Head 模块对于检测效果的提升，特别是对于小目标检测效果的提升有很大的促进作用。

LMDet-S 适用于检测各种各样的物体，包括拥挤的、被遮挡的、高度重叠的、非常大和非常小的物体。

如图 5–12 中的网球拍和手提包，由于和人体重合度较高，在对比算法中没有检测出来，本文算法由于采用了多层级基于中心的预测方法可以很好的应对拥挤、重叠的物体的检测。

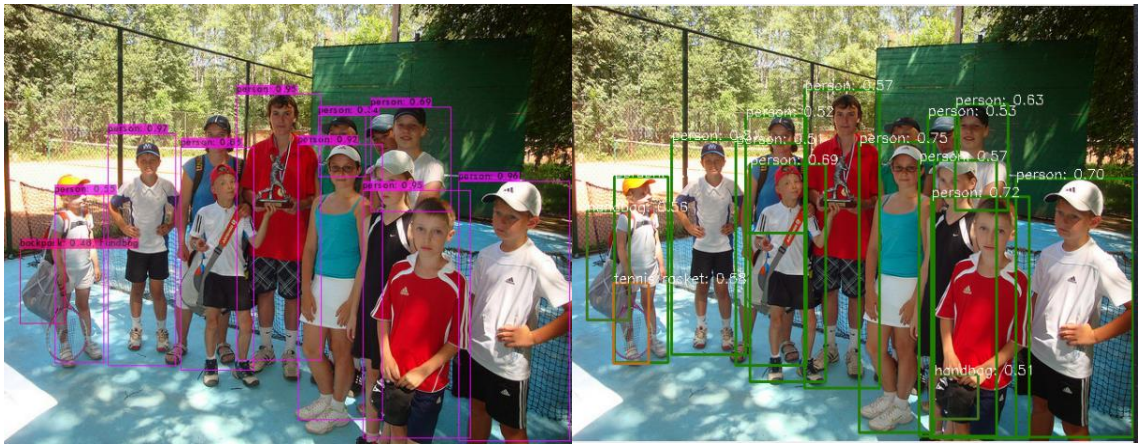


图 5–12 对于拥挤、重叠物体的检测效果对比图，左侧为 YOLOv4-Tiny，右侧为 LMDet-S

相似的情况如图 5–13 所示，在这些复杂场景中，物体边界模糊、界定不清晰（比如与人体框重叠度较高的棒球手套和棒球棒，以及被人体高度遮挡的背包等）。对于检测器来说这些目标的预测具有很强的不确定性。

图 5–13 对于重叠遮挡图像的检测效果对比图，左侧为 YOLOv4-Tiny，右侧为 LMDet-S

本文采用多层级预测的结构，不同层级预测不同尺度的目标。从本文的实验中可以看出，本文算法可以很好的处理重叠物体框产生的模糊性。另外本文采用了基于中心的回归方式，对于与大物体重叠度较高的小物体的检测比较友好。

5.3.3 算法轻量化实验

为验证本文轻量化方法的有效性，将本文算法与目前最新的轻量级目标检测算法 YOLOv4-Tiny[54]进行了轻量化性能的对比实验，实验结果如表 5–4 所示。

从表 5–4 可以看出，在采用相同输入分辨率的情况下，本文算法相比 YOLOv4-Tiny[54]于在平均精度提升 8.6%的同时，浮点运算量下降了 42%，参数量下降了 34%。将输入分辨率加大到 512*512，在浮点运算量相当的情况下，和对比算法相比，本文算法平均精度提升了 10.8%。

为了更直观的比较算法的性能，本文以 FLOPs 为横轴，以在 COCO 数据集上的平均精度为纵轴，绘制了散点图，如图 5–14 所示。

表 5–4 模型轻量化指标对比

模型	骨干网络	分辨率	mAP	FLOPs	参数量
YOLOv3-Tiny[2]	DarkNet53	416*416	17.6%	5.6B	8.86M
YOLOv4-Tiny[54]	Tiny-CSP-DarkNet53s	416*416	21.7%	6.96B	6.06M
LMDet-S-416	EfficientNet-Lite1	416*416	30.3%	4.06B	4.0M
LMDet-S-512	EfficientNet-Lite1	512*512	32.5%	7.1B	4.7M

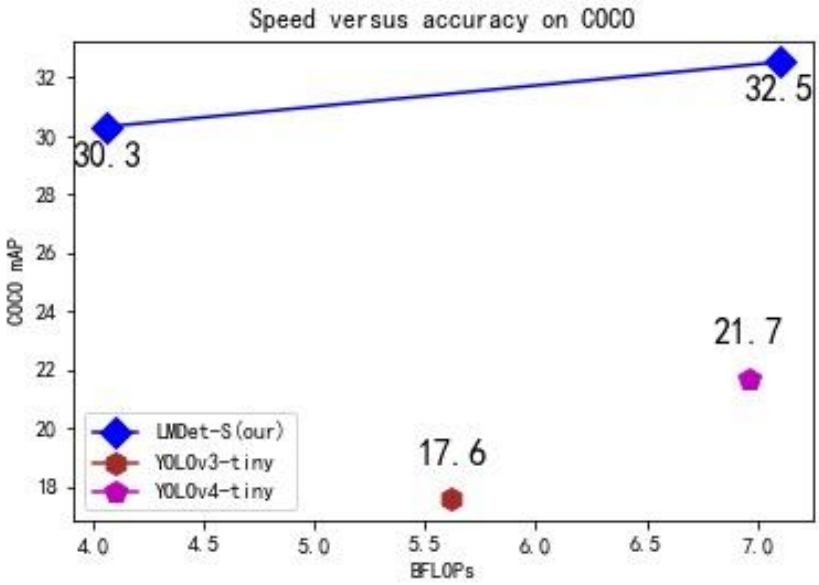


图 5–14 算法性能对比

5.4 综合实验

5.4.1 小目标数据集实验

本文在大型标准数据集上验证了算法的各项性能以后，又在专门的小目标数据集上进行了尝试。选择在 2020 年公布的，面向弱小人体目标检测的 TinyPerson 数据集上进行实验。

TinyPerson 中的大部分图像分辨率很大，如果直接将图像输入网络，一方面会导致 GPU 内存不足，另一方面采样的倍数过大，容易导致信息丢失。因此，本文在训练和测试过程中，先将原始图像切分成一些有重叠区域的子图像，然后再输入网络。

由于小目标数据集数据量较少，为了模型更好的收敛，本文采用了根据尺度匹配（Scale Match）策略[79]在 COCO 数据上得到的预训练权重。完整实验流程如图 5–15 所示。

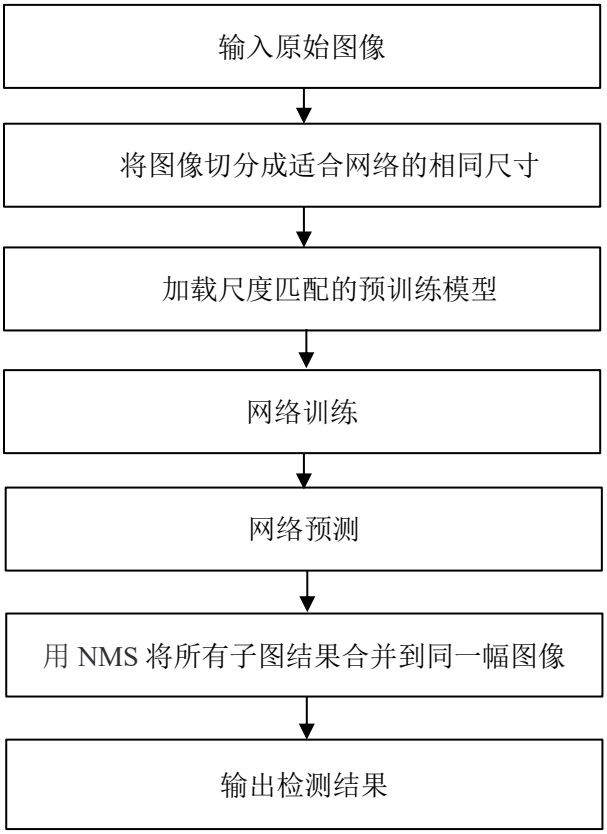


图 5–15 小目标数据集检测流程

在基于“切图策略”与“尺度匹配”的基础上，本文采用了 2 张显卡在 TinyPerson 数据集上进行训练，采用按步衰减的学习率调整策略，初始学习率设置为 0.0008，权重衰减设置为 0.0001，仅采用水平翻转的数据增强。根据文献[77]进行了其他参数的设置。

与本文第 4 章与第 5 章中将所有图像下采样调整到相同分辨率不同，本章由于使用了“切图策略”，在输入网络时无需缩放，只需使用对应分辨率的子图。检测效果如图 5–16 所示，检测精度如表 5–5 所示。

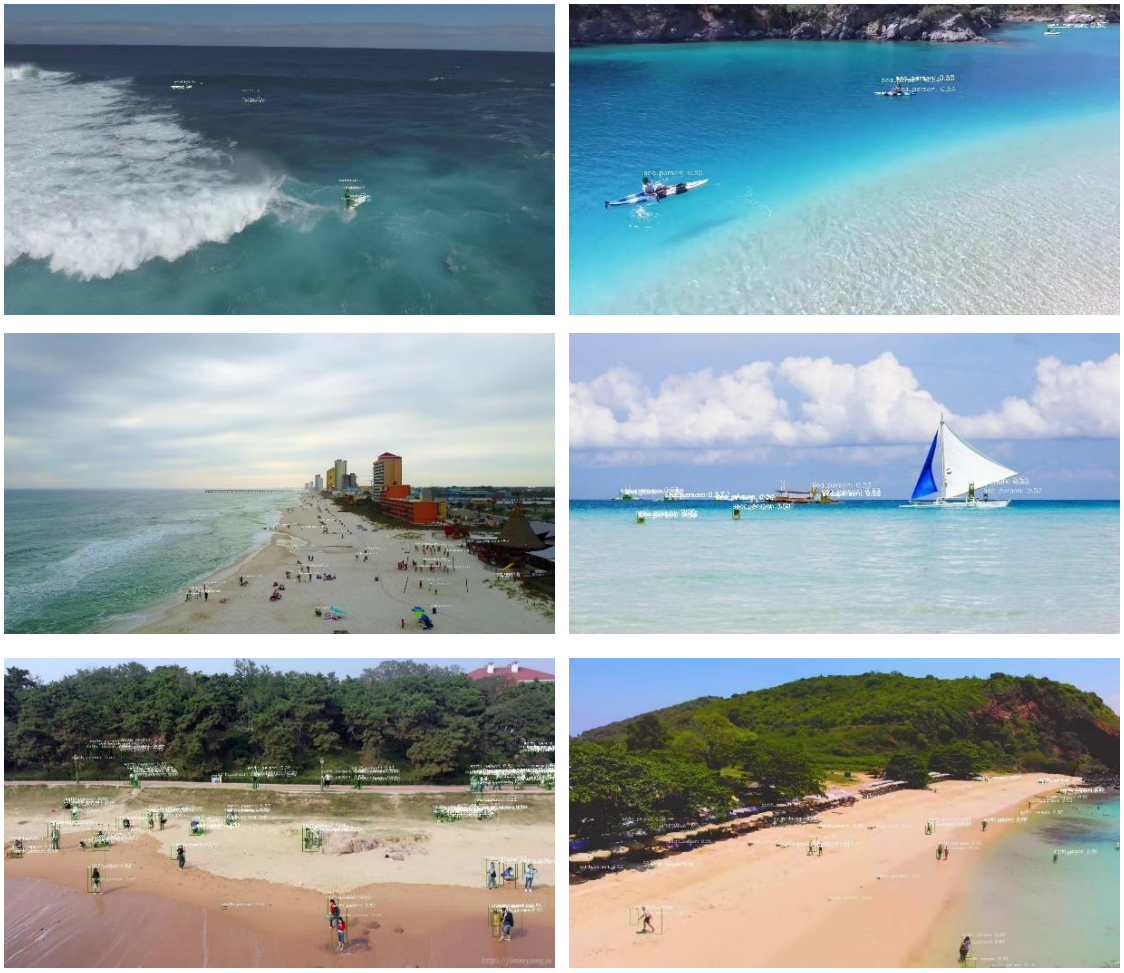


图 5–16 LMDet-S 在 TinyPerson 数据集上的检测效果

表 5–5 LMDet-S 在 TinyPerson 数据集上的检测精度（单位：%）

模型	骨干网络	分辨率	AP_{50}^{small}	AP_{50}^{tiny}
LMDet-S	EfficientNet-Lite1	512*512	30.75	13.7

作为一个新发布的数据集，目前，在 TinyPerson 上的实验，大部分都是基于多尺度训练、多模型集成的[88]，还没有公开的轻量化模型的实验，又因本文算法性能已经在前文实验中得到验证，本实验只是应用验证，所以此处只给出本文实验结果。

5.4.2 移动平台部署实验

为验证本文算法在移动边缘设备上落地部署的可行性，本文在智能手机这一常见的嵌入式设备上，进行了算法的移动平台部署实验。从模型训练到实机部署的实验流程如图 5–17 所示。

在实机部署时首先要将训练好的模型转换为 ONNX（Open Neural Network Exchange）格式，方便在不同深度学习框架之间迁移。

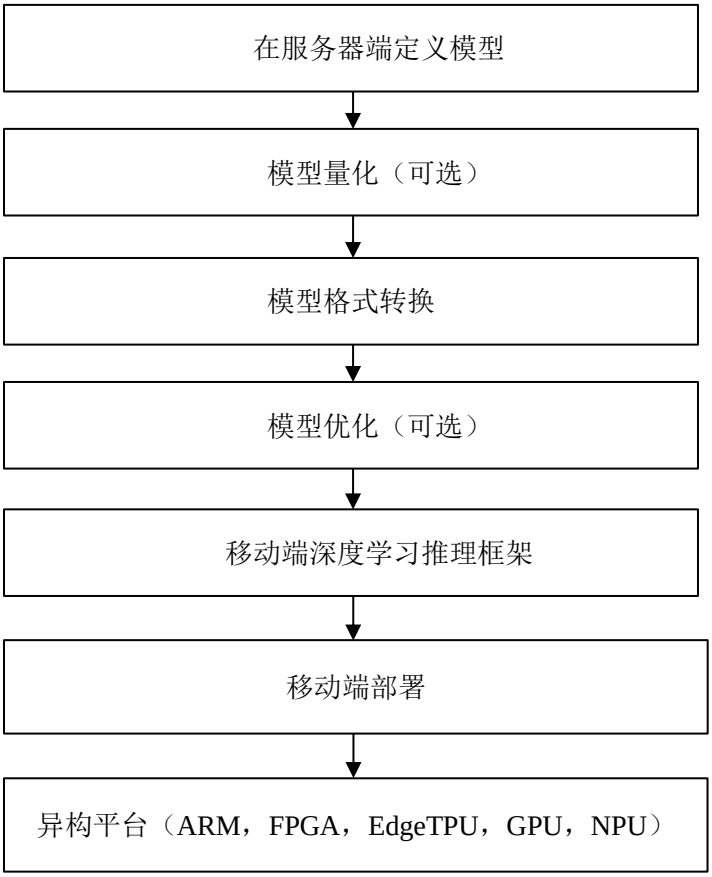


图 5–17 轻量化目标检测模型部署流程图

一般应用于移动平台的神经网络推理框架有，谷歌的 TensorFlow-Lite，腾讯的 NCNN，百度的 Paddle-Lite，以及目前处于测试阶段的 PyTorch-mobile 等。借助移动端神经网络推理框架可以方便的把在服务器训练好的模型部署到异构边缘设备上。其中，NCNN 是一个面向手机端部署和应用的，为手机端优化的高性能神经网络前向推理框架。

本实验选择 NCNN 作为前向推理框架，以 AndroidStudio 为开发环境，在型号为 vivo-iQOO-Z1x，处理器为骁龙 765G（2xA76+6xA55）的手机平台上进行了实验。实验场景如图 5–18 所示。

本文还与其他轻量级目标检测算法进行了部署的对比实验，如表 5–6 所示。以本文算法可达到 30FPS，达到了实时的标准。在相同的实机平台上，本文算法的检测速度大约是 YOLOv4-Tiny 的 4 倍，是 PP-YOLOv3 的 2.5 倍。

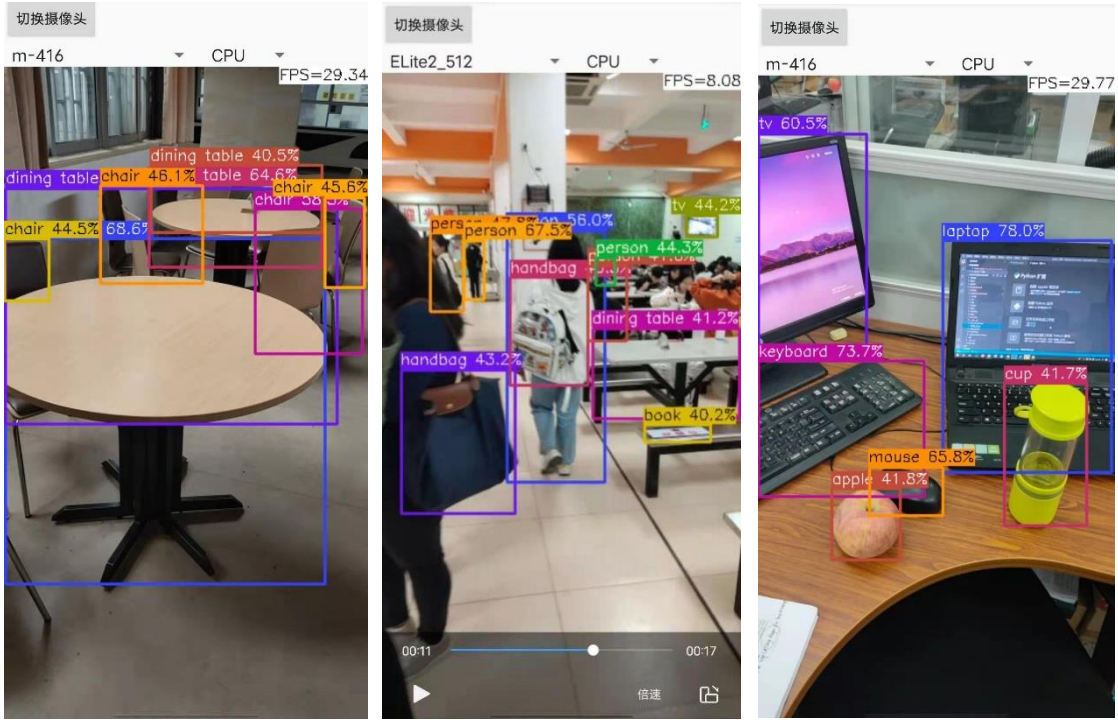


图 5–18 移动平台部署实验场景

表 5–6 实机测试算法性能比较表

模型	骨干网络	输入分辨率	FPS	模型尺寸
YOLOv4-Tiny[54]	Tiny-CSP-DarkNet53s	416*416	7.8	23.0MB
PP-YOLOv3[89]	MobilenetV3	320*320	12	11MB
LMDet-S-416	EfficienNet-Lite1	416*416	13	3.9MB
LMDet-S-416	ShuffleNetV2 1x	416*416	30	2.0MB

5.5 本章小结

本章首先对常见的无锚框检测算法进行了对比研究，然后在基于中心回归的基础上提出了轻量化无锚框的预测结构 LAF-Head，对预测分支和多层级预测结构进行了轻量化改进；其次对 LAF-Head 的性能进行了验证实验，又对算法进行了总体实验，并对实验结果进行了分析；接着在小目标数据集 TinyPerson 上进行了实验，最后进行了实机部署的实验。

第 6 章 总结与展望

6.1 全文工作总结

从最初的传统方法，到现在的基于深度学习的方法，目标检测近二十年的发展极大的促进了实际生产生活中大量应用场景的智能化升级。但是目前的目标检测算法还存在着一些问题，比如小目标检测能力不足，模型不够轻量，难以实际落地等。

在边缘人工智能火热发展的背景下，为促进目标检测算法在资源受限设备上的部署应用，并提高小目标的检测性能。本文设计了轻量化的目标检测算法 LMDet-S，并在实机平台上进行了部署。本文的主要工作可以总结为以下三点。

(1) 对常见的轻量级神经网络进行了研究，对其中的轻量化方法，以及轻量化基础模块的原理进行了分析，并从理论和实际两个层面对神经网络轻量化的评价指标进行了研究。

(2) 对轻量化的多尺度特征融合网络进行了研究，提出了轻量化的特征融合策略，设计了 Lite-PAN 这一轻量化的颈部网络结构。很好的解决了严重遮挡、重叠的目标的检测问题。

(3) 对常见的无锚框检测算法进行了对比研究，然后在基于中心回归的基础上提出了轻量化无锚框的预测结构 LAF-Head，对预测分支和多层级预测结构进行了轻量化改进。总体实验表明，与前沿算法 YOLOv4-Tiny 相比，本文算法浮点运算量下降了 42%，参数量减少了 34%，同时平均精度提高了 8.6%，小目标检测精度提高了 2%，并在智能终端上达到了 30FPS 的检测速度。

6.2 未来研究展望

本文提出的算法与目前的目标检测算法相比，虽然模型得到了大幅的轻量化，足以在资源受限的移动嵌入式设备上运行。但从轻量化模型的检测精度来看，总体上还处于发展阶段。本文的算法如果和重量级的模型相比，检测精度还有较大的进步空间。未来还有一些工作可以展开，下面对其进行展望。

(1) 采用神经网络架构搜索方法，添加实际硬件条件的约束，自动生成面向资源受限硬件平台的轻量化网络。

(2) 采用自动超参数搜索或超参数优化的方法，以训练中的平均精度为优化目标，进行超参数的自动设置与优化。

(3) 可以结合 Transformer 机制在目标检测中的应用，探索现有单阶段、两阶段、无锚框检测框架之外的新的检测机制，从底层机理上设计新的目标检测算法。

人工智能技术不断发展，赋能各行各业实现了“智能+”。未来，轻量化的小目标检测算法，可以应用在机器人上，让机器人更好的识别目标，从而更好的完成避障、抓取等任务；可以应用在无人机上，进行航拍遥感遥测；还可以应用在自动驾驶、安防监控、机器视觉等领域，促进生产生活的智能化升级。

参考文献

[1] Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger[C]. IEEE Conference on Computer Vision and Pattern Recognition. Honolulu. HI. USA. 2017. 6517-6525.

[2] Redmon J, Farhadi A. YOLOv3: An incremental improvement[J]. arXiv preprint, arXiv:1804.02767, 2018.

[3] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector[C]. European Conference on Computer Vision. Amsterdam. Netherlands. 2016. 21-37.

[4] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: Common Objects in Context[C]. European Conference on Computer Vision. Zurich. Switzerland. 2014. 740-755.

[5] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]. IEEE International Conference on Computer Vision. Venice. Italy. 2017. 2999-3007.

[6] He K, Gkioxari G, Dollar P, et al. Mask R-CNN[C]. IEEE International Conference on Computer Vision. Venice. Italy. 2017. 2961-2969.

[7] Uijlings J R R, Ven De Sande K E A, Gevers T, et al. Selective Search for Object Recognition[J]. International Journal of Computer Vision, 2013, 104(2): 154-171.

[8] Girshick R. Fast R-CNN[C]. IEEE International Conference on Computer Vision. Boston. MA. USA. 2015. 1440-1448.

[9] Barnich O, Van Droogenbroeck M. ViBe: A Universal Background Subtraction Algorithm for Video Sequences[J]. IEEE Transactions on Image Processing, 2011, 20(6): 1709-1724.

[10] Lowhur A, Chuah M C. Dense Optical Flow Based Emotion Recognition Classifier[C]. IEEE International Conference on Mobile Ad Hoc and Sensor Systems. Dallas. TX. USA. 2015. 573-578.

[11] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]. IEEE Conference on Computer Vision and Pattern Recognition. San Diego. CA. USA. 2005. 1. 886-893.

[12] Pavani S K, Delgado D, Frangi A F. Haar-like features with optimally weighted rectangles for rapid object detection[J]. Pattern Recognition, 2010, (43): 160-172.

[13] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features[C]. IEEE Conference on Computer Vision and Pattern Recognition. Kauai. HI. USA. 2001. 1. I-I.

[14] Neubeck A, Van Gool L. Efficient non-maximum suppression[C]. International Conference on Pattern Recognition. Hong Kong. China. 2006. 3. 850-855.

[15] Bodla N, Singh B, Chellappa R, et al. Soft-NMS--improving object detection with one line of code[C]. IEEE International Conference on Computer Vision. Venice. Italy. 2017. 5561-5569.

[16] Krizhevsky A, Sutskever I, Hinton G. ImageNet Classification with Deep Convolutional Neural Networks[C]. International Conference on Neural Information Processing Systems. Lake Tahoe. NV. USA. 2012. 1. 1097–1105.

[17] Jia D, Wei D, Socher R, et al. ImageNet: A large-scale hierarchical image database[C]. IEEE Conference on Computer Vision and Pattern Recognition. Miami. FL. USA. 2009. 248-255.

[18] Girshick R B, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]. IEEE Conference on Computer Vision and Pattern Recognition. Columbus. OH. USA. 2014. 580–587.

[19] He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for vi-sual recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014, 37 (9):1904-16.

[20] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39 (6):1137-1149.

[21] Lin T-Y, Dollar P, Girshick R, et al. Feature pyramid networks for object detection[C]. IEEE Conference on Computer Vision and Pattern Recognition. Honolulu. HI. USA. 2017. 936-944.

[22] Cai Z, Vasconcelos N. Cascade R-CNN: High Quality Object Detection and Instance Segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021, 43 (5): 1483-1498.

[23] Pang J, Chen K, Shi J, et al. Libra R-CNN: Towards Balanced Learning for Object Detection[C]. IEEE Conference on Computer Vision and Pattern Recognition. Long Beach. CA. USA. 2019. 821-830.

[24] Redmon J, Divvala S K, Girshick R B, et al. You only look once: Unified, real-time object detection[C]. IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas. NV. USA. 2016. 779-788.

[25] Bochkovskiy A, Wang C-Y, Liao H-Y M. YOLOv4: Optimal speed and accuracy of object detection[J]. arXiv preprint, arXiv:2004.10934, 2020.

[26] Jocher G. YOLOv5. [DB/OL]. https://github.com/ultralytics/yolov5, 2020.

[27] Law H, Deng J. CornerNet: Detecting objects as paired keypoints[C]. European Conference on Computer Vision. Munich. Germany. 2018. 642–656.

[28] Law H, Tang Y, Russakovsky O, et al. CornerNet-Lite: Efficient Keypoint-Based Object Detection[J]. arXiv preprint, arXiv:1904.08900, 2019.

[29] Duan K, Bai S, Xie L, et al. Centernet: Keypoint triplets for object detection[C]. IEEE International Conference on Computer Vision. Seoul. Korea. 2019. 6568-6577.

[30] Tian Z, Shen C, Chen H, et al. FCOS: Fully convolutional one-stage object detection[C]. IEEE International Conference on Computer Vision. Seoul. Korea. 2019. 9626-9635.

[31] Yang Z, Liu S, Hu H, et al. RepPoints: Point set representation for object detection[C]. IEEE International Conference on Computer Vision. Seoul. Korea. 2019. 9657-9666.

[32] Carion N, Massa F, Synnaeve G, et al. End-to-End Object Detection with Transformers[C]. European Conference on Computer Vision. Glasgow. UK. 2020. 213-229.

[33] Felzenszwalb P F, Girshick R B, Mcallester D, et al. Object Detection with Discriminatively Trained Part-Based Models[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2010, 32 (9):1627-1645.

[34] Fu C, Liu W, Ranga A, et al. DSSD: Deconvolutional single shot detector[J]. arXiv preprint, arXiv:1701.06659, 2017.

[35] Zhao Q, Sheng T, Wang Y, et al. M2Det: A single-shot object detector based on multi-level feature pyramid network[C]. The Thirty-Third AAAI Conference on Artificial Intelligence. Hawaii. USA. 2019. 3023-3033.

[36] Singh B, Davis L S. An Analysis of Scale Invariance in Object Detection-SNIP[C]. IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City. UT. USA. 2018. 3578-3587.

[37] Singh B, Najibi M, Davis L S. SNIPER: Efficient multi-scale training[C]. Conference on Neural Information Processing Systems. Montréal. Canada. 2018. 1-11.

[38] Shrivastava A, Gupta A, Girshick R. Training regionbased object detectors with online hard example mining[C]. IEEE conference on computer vision and pattern recognition. 2016. 761–769.

[39] Kisantal M, Wojna Z, Murawski J, et al. Augmentation for small object detection[J]. arXiv preprint, arXiv:1902.07296v1, 2019.

[40] Noh J, Bae W, Lee W, et al. Better to Follow, Follow to Be Better: Towards Precise Supervision of Feature Super-Resolution for Small Object Detection[C] IEEE International Conference on Computer Vision. Seoul. Korea. 2019. 9724-9733.

[41] Lim J S, Astrid M, Yoon H J, et al. Small Object Detection using Context and Attention[J]. arXiv preprint, arXiv:1912.06319, 2019.

[42] Han S, Mao H, Dally W J, et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding[J]. arXiv preprint, arXiv:1510.00149, 2019.

[43] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. Computer Science, 2015, 14 (7): 38-39.

[44] Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint, arXiv:1704.04861, 2017.

[45] Sandler M, Howard A G, Zhu M, et al. MobileNetV2: Inverted residuals and linear bottlenecks[C]. IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City. UT. USA. 2018. 4510-4520.

[46] Zhang X, Zhou X, Lin M, Sun J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices[C]. IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City. UT. USA. 2018. 6848-6856.

[47] Ma N, Zhang X, Zheng H T, et al. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design[C]. European Conference on Computer Vision. Munich. Germany. 2018. 1. 12-18.

[48] Tan M, Chen B, Pang R, et al. MnasNet: Platform-Aware Neural Architecture Search for Mobile[C]. IEEE Conference on Computer Vision and Pattern Recognition. Long Beach. CA. USA. 2019. 2815-2823.

[49] Wu B, Dai X, Zhang P, et al. FBNet: Hardware-Aware Efficient ConvNet Design via Differentiable Neural Architecture Search[C]. IEEE Conference on Computer Vision and Pattern Recognition. Seoul. Korea. 2019. 10734-10742.

[50] Howard A, Sandler M, Chen B, et al. Searching for MobileNetV3[C]. IEEE International Conference on Computer Vision. Seoul. Korea. 2019. 1314-1324.

[51] Tan M, Le Q V. Efficientnet: Rethinking model scaling for convolutional neural networks[J]. arXiv preprint, arXiv:1905.11946, 2019.

[52] Huang J, Rathod V, Sun C, et al. Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors[C]. IEEE Conference on Computer Vision and Pattern Recognition. Honolulu. HI. USA. 2017. 3296-3297.

[53] Tan M, Pang R, Le Q V. EfficientDet: Scalable and Efficient Object Detection[C]. IEEE Conference on Computer Vision and Pattern Recognition. Seattle. WA. USA. 2020. 10778-10787.

[54] Wang C-Y, Bochkovskiy A, Liao H-Y M. Scaled-YOLOv4: Scaling Cross Stage Partial Network[J]. arXiv preprint, arXiv:2011.08036v2, 2021.

[55] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. arXiv preprint, arXiv:1409.1556v4, 2015.

[56] Szegedy C, Liu W, Jia Y, et al. Going Deeper with Convolutions[J]. arXiv preprint, arXiv:1409.4842, 2014.

[57] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. The IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas. NV. USA. 2016. 770-778.

[58] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex[J]. Journal of Physiology, 1962, 160(1):106-154.

[59] Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position[J]. Biological Cybernetics, 1980, 36 (4): 193-202.

[60] Lecun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86 (11): 2278-2324.

[61] Xu J, Li Z, Du B, et al. Reluplex made more practical: Leaky ReLU[C]. IEEE Symposium on Computers and Communications. Rennes. France. 2020. 1-7.

[62] He K, Zhang X, Ren S, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification[C]. IEEE International Conference on Computer Vision. Santiago. Chile. 2015. 1026-1034.

[63] Yu F, Koltun V. Multi-Scale Context Aggregation by Dilated Convolutions[J]. arXiv preprint, arXiv:1511.07122, 2016.

[64] Dai J, Qi H, XIONG Y, et al. Deformable Convolutional Network[C]. IEEE International Conference on Computer Vision. Venice. Italy. 2017. 764-773.

[65] Yu J, Jiang Y, Wang Z, et al. Unitbox: An advanced object detection network[J]. arXiv preprint, arXiv:1608.01471, 2016.

[66] Polyak B T. Some methods of speeding up the convergence of iteration methods[J]. USSR Computational Mathematics and Mathematical Physics, 1964, 4 (5): 1-17.

[67] Sutskever I, Martens J, Dahl G, et al. On the importance of initialization and momentum in deep learning[C]. International Conference on Machine Learning. Atlanta. GA. USA. 2013. 1139-1147.

[68] Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization[J]. Journal of Machine Learning Research, 2011: 2121-2159.

[69] Zeiler M D. Adadelta:an adaptive learning rate method[J]. arXiv preprint, arXiv:1212.5701, 2012.

[70] Tieleman T, Hinton G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude[J]. COURSERA: Neural networks for machine learning, 2012, 4 (2): 26-31.

[71] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint, arXiv:1412.6980, 2014.

[72] Loshchilov I, Hutter F. Fixing weight decay regularization in adam[J]. arXiv preprint, arXiv:1711.05101, 2017.

[73] Xie S, Girshick R, Dollar P, et al. Aggregated residual transformations for deep neural networks[C]. IEEE Conference on Computer Vision and Pattern Recognition. Honolulu. HI. USA. 2017. 1492–1500.

[74] Huang G, Liu Z, Maaten L V D, et al. Densely connected convolutional networks[C]. IEEE Conference on Computer Vision and Pattern Recognition. Honolulu. HI. USA. 2017. 4700–4708.

[75] Li Z, Peng C, Yu G, et al. DetNet: Design backbone for object detection[C]. European Conference on Computer Vision. Munich. Germany. 2018. 334–350.

[76] Chen Y, Yang T G, Zhang X, et al. DetNAS: Backbone search for object detection[J]. In Advances in Neural Information Processing Systems, 2019: 6638–6648.

[77] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation. IEEE Conference on Computer Vision and Pattern Recognition[C]. Salt Lake City. UT. USA. 2018. 8759–8768.

[78] Ghiasi G, Lin T-Y, Le Q V. NAS-FPN: Learning scalable feature pyramid architecture for object detection[C]. The IEEE Conference on Computer Vision and Pattern Recognition. Long Beach. CA. USA. 2019. 7029-7038.

[79] Yu X, Gong Y, Jiang N, et al. Scale Match for Tiny Person Detection[C]. IEEE Winter Conference on Applications of Computer Vision. Snowmass Village. CO. USA. 2020. 1257-1265.

[80] Qiao S, Chen L-C, Yuille A. DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution[J]. arXiv preprint, arXiv:2006.02334, 2020.

[81] Shrivastava A, Sukthankar R, Malik J, et al. Beyond skip connections: Top-down modulation for object detection[J]. arXiv preprint, arXiv:1612.06851, 2016.

[82] Everingham M, Eslami S M A, Van Gool L, et al. The pascal visual object classes challenge: A retrospective[J]. International Journal of Computer Vision, 2015, 111 (1): 98-136.

[83] Zhou X, Zhuo J, Krahenbuhl P. Bottom-up object detection by grouping extreme and center points[C]. The IEEE Conference on Computer Vision and Pattern Recognition. Long Beach. CA. USA. 2019. 850-859.

[84] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <1MB model size[J]. arXiv preprint, arXiv:1602.07360v1, 2016.

[85] Li X, Wang W, Wu L, et al. Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection[J]. arXiv preprint, arXiv:2006.04388v1, 2020.

[86] Ioffe S, Szeged Y C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[C]. International Conference on Machine Learning. Lille. France. 2015. 448-456.

[87] Li Z, Peng C, YU G, et al. Light-head R-CNN: In defense of two-stage object detector[J]. arXiv preprint, arXiv:1711.07264, 2017.

[88] Yu X, Han Z, Gong Y. The 1st Tiny Object Detection Challenge: Methods and Results[J]. arXiv preprint, arXiv:2009.07506, 2020.

[89] Deng K. PP-YOLO[DB/OL]. https://github.com/PaddlePaddle/PaddleDetection/blob/release/2.0-rc/configs/ppyolo/README_cn.md, 2020.

[90] Wang A. EfficientNet-Lite[DB/OL]. https://github.com/tensorflow/tpu/tree/master/models/official/efficientnet/lite, 2020.

攻读学位期间主要研究成果

一、发表的学术论文

（一）准备投稿论文

[1] Z. Fan, X. Hu. LMDet-S: A Lightweight Multi-scale Detector for Small Object Detection. (与第 4 章和第 5 章相关)

致谢

转眼又到了金凤花盛开的季节，值此论文完成之际，谨向各位帮助过我的老师和同学们表示由衷的感谢。

首先，感谢我的导师范衡教授。范教授学术底蕴深厚，治学严谨认真，对待学生宽厚和蔼，指引我在人工智能与计算机视觉的研究中更加深入。在我的研究生学习期间，范教授不仅提供了优良的研究环境和实验设备，还传授给我许多有价值的研究方法和思路，开阔了我的视野，培养了我的思辨能力和表达能力，让我在为学和为人方面都受益终生，在此向范教授表示衷心的感谢。

感谢我的师兄冯靖安、李冲、莫嘉杰、伍宇明、朱贵杰等，你们的研究启迪了我的思维。感谢我的师弟马培利、韦家弘等，在实验遇到难题的时候，和你们的讨论总能让我有所收获，谢谢你们。

感谢我的家人，你们的默默支持与无私付出是我不断前进的动力。

最后，感谢电子信息工程系各位老师的关怀与教诲，谢谢各位老师。

作者：胡星晨

2021 年 3 月 20 日