分类号	TP181	密级	公开
UDC	004.8	学位论文编号	D-10617-308-(2022)-03062

重庆邮电大学硕士学位论文

中文题目	面向智能路侧边缘计算的车辆行人视觉目标检测算法研究
英文题目	**Research on Vehicle-pedestrian Visual Object Detection Algorithm for Intelligent Road Side Edge Computing**
学　　号	**S190301066**
姓　　名	胡浪
学位类别	工学硕士
学科专业	控制科学与工程
指导教师	蒋建春　教授
完成日期	**2022 年 5 月 30 日**

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含他人已经发表或撰写过的研究成果，也不包含为获得 重庆邮电大学 或其他单位的学位或证书而使用过的材料。与我一同工作的人员对本文研究做出的贡献均已在论文中作了明确的说明并致以谢意。

作者签名：胡浪　　日期：2022年6月9日

学位论文版权使用授权书

本人完全了解 重庆邮电大学 有权保留、使用学位论文纸质版和电子版的规定，即学校有权向国家有关部门或机构送交论文，允许论文被查阅和借阅等。本人授权 重庆邮电大学 可以公布本学位论文的全部或部分内容，可编入有关数据库或信息系统进行检索、分析或评价，可以采用影印、缩印、扫描或拷贝等复制手段保存、汇编本学位论文。

（注：保密的学位论文在解密后适用本授权书。）

作者签名：胡浪　　导师签名：

日期：2022年6月9日　　日期：2022年6月10日

摘要

视觉目标检测是车路协同系统中路侧感知的关键技术之一，也是学术界持续的研究热点。路侧感知对视觉目标检测算法的精度与实时性均有较高要求。但现有的轻量化目标检测算法精度相对较低。通用目标检测算法的计算量则相对较大，在路侧边缘计算设备上部署难以满足应用的实时性需求，有必要借助模型压缩技术降低目标检测算法所需的计算量。此外，路侧场景也具有较高的复杂度，需要对目标检测算法进行场景适配，以达到更好的检测效果。

论文根据路侧感知对视觉目标检测算法的性能需求，研究并设计面向路侧边缘计算的车辆行人视觉目标检测算法，主要进行的几点工作如下：

1. 论文分析了当前主流的视觉目标检测算法与模型压缩算法框架，并对 YOLOX、ResRep 通道剪枝和基于注意力机制的知识蒸馏算法进行了概括，还总结了路侧场景特点以及路侧感知对视觉目标检测算法的性能需求。

2. 针对路侧感知的检测精度要求，论文在 YOLOX 目标检测算法的基础上，引入 RepVGG 卷积块和 Coordinate 注意力机制对 YOLOX 网络结构进行改进，并用 Alpha-IoU 损失和 VariFocal 损失优化 YOLOX 的损失函数。通过实验证明，论文所提方法能有效提升 YOLOX 对车辆行人目标的检测精度。

3. 针对路侧感知的实时性要求，论文结合 ResRep 通道剪枝与基于注意力机制的知识蒸馏算法，设计了一种适用于目标检测的模型压缩方法。通过实验证明，论文所提方法在无明显精度下滑的前提下，显著降低了模型所需的计算量。

4. 利用校内搭建的智能路侧平台，论文远程采集并标注实际的路侧场景图像用于测试论文所得模型的精度，并借助模型部署工具和视频流推理框架，将模型部署在路侧边缘计算平台上进行实时性测试。通过实验证明，论文设计的模型相比原版 YOLOX 更适合路侧场景，并具备一定的实际应用价值。

关键词：路侧感知，边缘计算，目标检测，模型压缩

Abstract

Visual object detection is one of the key technologies of roadside perception in cooperative vehicle infrastructure systems, and it is also a continuous research hotspot in academia. Roadside perception has high requirements on the accuracy and real-time performance of visual object detection algorithms. However, the existing lightweight object detection algorithms have relatively low accuracy. Meanwhile, the computational complexity of general object detection algorithms is relatively large, and it is difficult to deploy on roadside edge computing devices to meet the real-time requirements of applications. It is necessary to use model compression technology to reduce the computational complexity of the object detection algorithms. In addition, the roadside scenes also have high complexity, and the object detection algorithms need to be adapted to the scenes for better detection performance.

According to the performance requirements of roadside perception for visual object detection algorithms, this thesis studies and designs a vehicle-pedestrian visual object detection algorithm for roadside edge computing. The main tasks are as follows:

1. This thesis analyzes the current mainstream framework of visual object detection algorithms and model compression algorithms, and summarizes algorithms including YOLOX, ResRep channel pruning and knowledge distillation based on attention mechanism. It also summarizes the characteristics of roadside scenes and the performance requirements of roadside perception for visual object detection algorithms.

2. According to the detection accuracy requirements of roadside perception, on the basis of YOLOX object detection algorithm, this thesis introduces the RepVGG convolution block and the Coordinate attention mechanism to improve the network structure of YOLOX, and uses the Alpha-IoU loss and VariFocal loss to optimize the loss function of YOLOX. Experiments show that the method proposed in this thesis can effectively improve the detection accuracy of YOLOX for vehicle-pedestrian objects.

3. Aiming at the real-time requirements of roadside perception, this thesis designs a model compression method suitable for object detection by combining the ResRep channel pruning and the knowledge distillation based on the attention mechanism. Experiments show that the method proposed in this thesis significantly reduces the amount of computation required by the model without a significant drop in accuracy.

4. Using the intelligent roadside platform built on campus, this thesis remotely collects and annotates the actual roadside scene images to test the accuracy of the model obtained by this thesis, and deploys the model on the roadside edge computing platform with the help of model deployment tools and video streaming inference framework to test the real-time performance of the model. Experiments show that the model designed in this thesis is more suitable for roadside scenes than the vanilla YOLOX, and has certain practical application value.

Keywords: Roadside perception, Edge computing, Object detection, Model compression

目录

注释表

C-ITS	Cooperative Intelligent Transportation Systems，合作式智能交通系统
CVIS	Cooperative Vehicle Infrastructure Systems，车路协同系统
DSRC	Dedicated Short Range Communications，专用短程通信
HOG	Histogram of Oriented Gradient，方向梯度直方图
SVM	Support Vector Machines，支持向量机
DPM	Defomable Parts Model，可变形部件模型
HNM	Hard Negative Mining，难例挖掘
CNN	Convolutional Neural Network，卷积神经网络
ROI	Region of Interest，感兴趣区域
RPN	Region Proposal Network，区域生成网络
NLP	Natural Language Processing，自然语言处理
NMS	Non-maximum Suppression，非极大值抑制
NAS	Neural Architecture Search，神经架构搜索
BN	Batch Normalization，批归一化
RSU	Road Side Unit，路侧单元
C-V2X	Cellular Vehicle to Everything，蜂窝车联网
OBU	On Board Unit，车载单元
CV	Computer Vision，计算机视觉
FPN	Feature Pyramid Networks，特征金字塔网络
IoU	Intersection over Union，交并比
ATSS	Adaptive Training Sample Selection，自适应训练样本选择
OTA	Optimal Transport Assignment，最优传输分配
ASR	Automatic Speech Recognition，自动语音识别
AI	Artificial Intelligence，人工智能
CPU	Central Processing Unit，中央处理单元
GPU	Graphics Processing Unit，图形处理单元

AP	Average Precision，平均查准率
mAP	Mean Average Precision，平均查准率均值
FLOPs	Floating Point Operations，浮点运算量
FPS	Frames Per Second，每秒处理帧数
SGD	Stochastic Gradient Descent，随机梯度下降
EMA	Exponential Moving Average，指数滑动平均
CUDA	Compute Unified Device Architecture，统一计算设备架构
IVA	Intelligent Video Analysis，智能视频分析

第 1 章 绪论

1.1 研究背景及意义

随着中国经济的持续发展，我国汽车工业逐步完善，私人汽车逐渐大众化，汽车保有量呈现逐年上升的趋势。由公安部交通管理局公布的数据可知，截止 2020 年底，全国机动车保有量已达 3.72 亿辆，其中汽车 2.81 亿辆[1]。机动车保有量的持续增长会导致交通拥堵。根据高德地图等联合发布的《2020 年度中国主要城市交通分析报告》[2]显示，全国有 40%以上的主要城市在上下班高峰期处于拥堵或缓行状态。此外，大量的机动车也导致交通事故的频发。由中国国家统计局发布的《中国统计年鉴—2020》[3]可知，我国每年大约发生 20 多万起交通事故，直接造成的经济损失约 15 亿元。为应对日益凸显的交通问题，合作式智能交通系统（Cooperative Intelligent Transportation Systems，C-ITS）[4]的概念被提出。作为 C-ITS 最新的发展方向，车路协同系统（Cooperative Vehicle Infrastructure Systems，CVIS）[5]采用先进的信息与通信技术，全方位实施车车、车路实时信息交互，以形成安全、高效和环保的道路交通环境。

CVIS 的关键功能之一是实现对道路状态的实时感知，而感知部分主要在路侧端进行。目前，车载感知在技术层面逐渐进入瓶颈期，短时间内无法得到突破性的进展，且主流的车载感知方案大多配备昂贵的激光雷达以及多个辅助传感器，导致整车成本大幅增加。此外，车载感知也存在明显的不足，不能对车辆的周围环境进行多层次、远距离感知[6]。借助车联网通信技术，CVIS 能有效弥补车载感知存在盲区的问题，实现对道路状态的协同一体化感知，使得车辆具备超视距感知的能力。由于路侧感知设施主要由市政部门建设，因此车路协同技术的商用落地还具备低造车成本的优势。在智能网联汽车渗透率不足的当下，CVIS 可弥补单车智能在成本、安全等方面的缺陷，使得自动驾驶由单车智能转变为多智能体协同合作，从而加速智能网联汽车的普及。

在车路协同系统中，路侧感知主要通过架设在道路上的高清摄像头等设备完成对道路车辆、行人等目标的识别与定位，因此视觉目标检测算法是路侧感知的关键技术之一。为提升数据处理效率，并提供更快的应用响应速度，路侧感知采用边缘

计算设备就近处理高清摄像头输出的实时视频流。但受限于边缘计算设备的性能，目标检测算法须满足实时性指标。此外，路侧场景具备较高的复杂度，需要目标检测算法具有较高的精度，以在不同路况下表现良好。目前，现有的目标检测算法难以同时满足实时性与精度要求，需结合高效的网络结构以及模型压缩算法，并借助模型部署工具和视频流推理框架以满足应用需求。综上所述，研究面向路侧感知的视觉目标检测算法具有重要意义和实用价值。

1.2 国内外研究现状

1.2.1 车路协同系统发展现状

车路协同系统 CVIS 的概念于 21 世纪初诞生，是车与路不断智能化、网联化和协同化的产物。作为智能交通系统 ITS 的重要组成部分，CVIS 强调聪明的车、智能的路理念，通过全时空动态交通信息采集与融合，并借助群体智能理论与技术，实现对道路交通的智能决策和协同控制，从而提升交通出行的安全和效率。CVIS 主要包括信息感知与交互系统、车路协同控制系统和车车协同控制系统，利用环境感知设备获取自身以及外部交通环境信息进行动态信息整合，能为驾驶员提供车路引导、危险预警等服务[7]。相较于传统的固定式交通信息采集器，CVIS 可获取更加丰富的对交通信号控制有用的信息，例如车辆速度、位置、密度和排队长度等。这些信息可用于实现更为准确的交通流调度。

CVIS 受到国外发达国家的高度重视，美国、欧洲和日本等国家均大力投入到车路协同相关的研究当中。美国在 2009 年由美国交通部组织开展了 IntelliDrive[8] 研究计划，尝试利用专用短程通信（Dedicated Short Range Communications，DSRC）技术建立车车、车路之间的无线通讯网络。欧洲 ITS 组织在 2003 年提出了 e-Safety 概念，旨在为道路交通提供全面的安全解决方案。此后，欧洲 ITS 组织又相继推出 Coopers、CVIS 和 SafeSpot 等车联网研究项目，这些项目构成了目前欧洲车路协同技术的主要内容[9]。日本在 2004 年实施了 SmartWay 研究计划，包括构建智能汽车系统、智能道路系统和车路协同系统。2013 年，日本提出自动驾驶汽车商用化时间表，制定了未来道路交通的总体发展战略。2017 年，日本开展大规模的自动驾驶汽车车路协同现场操作测试，并对全球实体开放[10]。

我国根据交通运输发展战略，始终不遗余力改善道路交通安全，提高交通出行效率。2011 年，科技部制定国家高技术研究发展计划，设立智能车路协同关键技术研究项目。2019 年，交通部发布《数字交通发展规划纲要》[11]，旨在推动自动驾驶和车路协同技术研发，开展专用测试场地建设。2020 年，发改委等联合发布《智能汽车创新发展战略》[12]，强调结合 5G 商用部署，推动 5G 与车联网协同建设，开展特定区域智能汽车测试运行及示范应用等。在国家政策的支持下，我国 CVIS 发展势头强劲。目前，我国已在多个省市或地区建立车路协同示范区，主要应用场景包括交叉路口、智慧公交、货运车队、工业园区、智慧停车和共享汽车等，应用道路类型可归纳为城市道路、乡村道路、高速公路和封闭园区道路四大类。根据 2020 年世界智能网联汽车大会发布的《智能网联汽车技术路线图 2.0》[13]可知，预计 2025 年，我国 CVIS 将在高速公路、城市道路节点和园区封闭道路成熟应用，并向市场推出具备协同决策功能的车辆。

1.2.2 边缘计算发展现状

随着数据量的急剧增加，以及数据处理要求的愈发多样化，依靠中心云计算的模式面临诸多挑战。边缘计算（Edge Computing）[14]的提出有效解决了传统云计算存在的云中心负载、传输带宽、数据延迟与优化和安全隐私等问题。作为近几年工业界和学术界的共同关注热点，边缘计算凭借其高效的计算任务响应等特质，在智能家居、视频监控、智能制造和智能交通等领域被大量应用[15]。相较于传统云计算框架，边缘计算的优势主要在于可分割的应用程序与服务、可分布的资源与数据、快速响应和数据本地计算等[16]。在万物互联时代，边缘计算并非用于替代云计算，而是作为云计算服务的有效补充，其降低了云计算中心的数据能耗和运算负载，确保了数据安全与应用服务低时延。

近年来，国内外众多机构开展了大量针对边缘计算的研究。2015 年，欧洲电信标准协会发布《边缘计算白皮书》[17]，阐述了边缘计算内容、应用方向和部署计划等，开启了边缘计算的标准化工作。2016 年，国际计算机学会 ACM 与电气和电子工程师协会 IEEE 联合举办主题为边缘计算的科研学术论坛，讨论了边缘计算的研究方向和应用价值[18]。2018 年，工业互联网产业联盟发布《工业物联网边缘计算介绍白皮书》[19]，阐述了工业物联网领域中边缘计算的应用价值与优势。2019 年，边

缘计算产业联盟和工业互联网产业联盟发布《边缘计算安全白皮书》[20]，针对边缘计算应用的安全性问题，分析了不同领域中边缘计算应用面临的挑战和需求。

我国边缘计算研究处于快速发展时期。2016 年，中国信通研究院等联合成立边缘计算产业联盟[21]，成员包括院校、工业和交通等多个领域，提出边缘计算是在靠近物或数据源头的网络边缘侧，融合网络、计算、存储和应用核心能力的开放平台，用以就近提供边缘智能服务。2018 年，边缘计算产业联盟与工业互联网产业联盟联合发布边缘计算参考框架 3.0 版本[22]，详细阐释了边缘计算概念、架构功能设计和商业实践方案等。随着国内大量学者对边缘计算问题的不断深入研究，我国边缘计算技术正在不断进步当中。

1.2.3 视觉目标检测研究现状

视觉目标检测算法根据技术方案的不同，可分为传统目标检测算法和基于深度学习的目标检测算法两大类。传统方法通常依赖手工设计的特征提取方法与机器学习方法相结合的方式进行物体检测，深度学习方法则主要以端到端的方式，使用神经网络完成对图像中潜在物体的定位与分类。

1. 传统目标检测算法

传统目标检测算法主要通过结合滑动窗口与手工特征的方式，提取潜在目标区域的判别性特征，并利用机器学习方法对提取的特征进行分类。2001 年，P. Viola 等引入积分图技术，利用滑动窗口的方式快速提取 Haar 特征，并使用 AdaBoost 算法对产生的大量 Haar 特征进行级联分类，首次实现了人脸实时检测[23]。2005 年，N. Dalal 等提出方向梯度直方图（Histogram of Oriented Gradient，HOG）特征，并使用支持向量机（Support Vector Machines，SVM）对提取的特征进行分类，在行人检测任务中取得了显著的效果[24]。HOG 特征具有较好的图像几何和光学不变性，但存在计算量较大的问题。2008 年，P. Felzenszwalb 等在 HOG 方法的基础上，提出了可变形部件模型（Defomable Parts Model，DPM），并获得 PASCAL VOC 07-09 届的目标检测挑战赛冠军[25]。DPM 引入了一些具有前瞻性的方法，包括混合模型、难例挖掘（Hard Negative Mining，HNM）和边框回归等，对后续的目标检测算法有较为深远的影响[26]。尽管传统方法取得了一定的成果，但其也存在手工特征表示能力差、泛化能力弱和冗余度高等不足。

2. 基于深度学习的目标检测算法

随着深度学习的兴起，目标检测逐渐过渡到以卷积神经网络（Convolutional Neural Network，CNN）为代表的端到端方法。相较于传统方法，深度学习方法使用神经网络从图像中学习具有高度泛化的特征，以显著提升目标检测算法在不同场景中的检测性能。根据网络结构的不同，基于深度学习的目标检测算法可分为两阶段（Two-stage）和一阶段（One-stage）两大类。此外，针对嵌入式设备的轻量化目标检测算法在近几年也受到学术界的关注。

(1) Two-stage 目标检测算法

Two-stage 算法首先从图像中提取感兴趣区域（Region of Interest，ROI），然后对从 ROI 提取的特征进行分类和回归。2015 年，S. Ren 等[27]提出 Faster-RCNN，采用区域生成网络（Region Proposal Network，RPN）提取 ROI，从而大幅降低 ROI 生成的计算量，能以接近实时的速度进行检测。Faster-RCNN 也是首个端到端的深度学习目标检测算法。2018 年，Z. Cai 等[28]提出 Cascade RCNN，引入一种阈值递增的级联检测头部网络处理提取的 ROI，有效提升了 Two-stage 算法的检测性能。2020 年，N. Carion 等[29]提出 DETR，引入在自然语言处理（Natural Language Processing，NLP）领域大获成功的 Transformer 结构到目标检测任务中，实现无需非极大值抑制（Non-maximum Suppression，NMS）操作的完全端到端 Two-stage 算法。2021 年，P. Sun 等[30]提出 Sparse R-CNN，在无需 Transformer 等复杂结构的情况下，同样实现了完全端到端的目标检测。

(2) One-stage 目标检测算法

One-stage 算法可看作是对 Two-stage 算法的简化，其不包括提取 ROI 的过程，直接对网络提取的特征进行分类和回归，因此检测效率更高。2016 年，J. Redmon 等[31]提出 YOLO，在保持一定精度的同时，具备远超 Two-stage 算法的检测速度。YOLO 一经推出，便引领了 One-stage 算法的研究热潮。此后，J. Redmon 等又相继提出 YOLOv2[32]、v3[33]版本，进一步提升了检测性能。同年，W. Liu 等[34]提出 SSD，引入基于锚框（Anchor-based）的多尺度特征检测头部网络，结合在线难例挖掘和数据增广等技巧，取得了超越初代 YOLO 的检测性能。2017 年，T. Y. Lin 等[35]提出 RetinaNet 以及 Focal Loss 损失函数，达到接近 Two-stage 算法的性能。RetinaNet 也是众多 Anchor-based 目标检测算法的模板。

在 2019-2021 年间，大量 One-stage 算法成果涌现。其中，M. Tan 等[36]提出 EfficientDet，利用模型扩展方法，同时缩放网络的输入分辨率、宽度和深度，得到可满足不同设备算力的系列检测网络。X. Du 等[37]提出 SpineNet，利用神经架构搜索（Neural Architecture Search，NAS）得到目标检测适用的主干网络。A. Bochkovskiy 等[38]在 YOLOv3 的基础上，引入 Mosaic 数据增广等成果，进一步提出高效的 YOLOv4。除上述 Anchor-based 方法之外，无锚框（Anchor-free）目标检测算法也逐渐成为学术界的研究热点，具有代表性的包括 Z. Tian 等[39]提出的 FCOS 和 X. Zhou 等[40]提出的 CenterNet。Anchor-free 方法的优势在于，无需复杂的锚框设计和边框编解码过程，在训练时也更加节省存储空间和算力。此外，完全端到端的 One-stage 算法也受到关注，其中以 J. Wang 等[41]提出的 DeFCN 和 P. Sun 等[42]提出的 OneNet 为代表。在视觉 Transformer 兴起后，Z. Liu 等[43]提出 Swin Transformer，引入卷积结构范式到 Transformer 框架，在目标检测任务中取得显著的效果。

(3) 轻量化目标检测算法

在轻量化目标检测方面，2018 年，M. Sandler 等[44]在 SSD 的基础上，提出轻量化检测算法 SSDLite，实现在嵌入式设备上实时运行，但检测精度相对较低。2019 年，Z. Qin 等[45]提出 ThunderNet，采用轻量化主干网络和高效特征融合网络，在移动终端设备上首次实现 Two-stage 算法的实时检测。同年，A. Wong 等[46]在 YOLO 基础上，结合手工与计算机协同设计的方式，提出针对嵌入式设备的轻量化检测网络 YOLO-nano。2021 年，Y. Cai 等[47]提出 YOLObile，借助结构化剪枝与设计的并行加速推理方法，在移动端设备上首次实现具有通用算法性能的实时目标检测，但在模型部署过程中需要定制推理框架。

在路侧感知系统中，由于受到成本、功耗和体积等因素的限制，需要采用嵌入式边缘计算设备进行感知计算。现阶段目标检测算法的不足在于，通用检测算法的计算量较大，部署在路侧边缘计算设备上难以满足实时性要求。轻量化检测算法的精度又相对较低，不足以实现对车辆行人目标的准确感知。此外，通用检测算法也需要针对路侧场景特点进行适配。论文针对上述问题，研究结合高效网络结构和模型压缩算法，设计一种结构紧凑的目标检测模型，借助模型部署工具和视频流推理框架，以满足路侧感知对视觉目标检测算法的性能需求。

1.2.4 模型压缩研究现状

深度学习模型通常会占据较多的存储空间，所需的计算量也相当巨大，导致在工业化部署时难以达到应用的实时性要求。模型压缩可用于减少现有模型的计算量和参数量，使得模型能顺利在存储与算力受限的设备上部署。常见的模型压缩方法包括结构化剪枝、知识蒸馏和低精度量化等。

1. 结构化剪枝（Structural Pruning）

结构化剪枝通过评估网络中通道或卷积核的重要性，从而移除非关键的整组参数，使得裁剪后的网络依然是密集连接的，无需额外的计算库或硬件就能加速网络的推理过程。2017 年，Y. He 等[48]提出一种两步通道剪枝方法，首先通过 LASSO 回归对已训练的模型进行通道选择，然后使用最小二乘重构对网络的精度进行逐层恢复。同年，Z. Liu 等[49]提出一种自适应通道剪枝方法，在训练中对网络的批归一化（Batch Normalization，BN）层施加 L1 范数惩罚，用以自适应度量通道的重要性。训练结束后再对网络进行参数裁剪和微调。2019 年，J. Chen 等[50]提出一种无需微调的通道剪枝方法，在训练中通过辅助网络和辅助损失，动态学习卷积层的通道掩码。2020 年，Y. Wang 等[51]提出一种先剪枝、后训练的方法，首先在随机参数初始化的大网络中进行通道裁剪，然后从零训练剪枝得到的小网络。2021 年，J. Liu 等[52]提出一种基于判别感知的通道剪枝方法，在训练中引入判别感知损失对网络的重要参数进行鉴别，训练结束后再逐段进行通道裁剪和微调。

2. 知识蒸馏（Knowledge Distillation）

知识蒸馏旨在让学生小网络模仿教师大网络的输出，以使参数量与计算量较少的小网络也能有较好的性能，从而达到模型压缩的目的。2015 年，G. Hinton 等[53]首次提出知识蒸馏的概念，使用教师网络的预测概率分布作为软标签，用以监督学生网络的训练。2016 年，N. Komodakis 等[54]提出在蒸馏过程中引入注意力机制，使学生网络模仿教师网络的注意力热图。2019 年，X. Dong 等[55]首先利用 NAS 在大网络中搜索得到高效的小网络，然后用知识蒸馏对小网络进行迁移训练。同年，L. Zhang 等[56]提出一种结合特征空间、通道和全局信息的知识蒸馏方法，用以解决目标检测蒸馏效率较低的问题。2021 年，J. Guo 等[57]提出一种简单有效的特征解耦知识蒸馏方法，广泛适用于 Two-stage 和 One-stage 检测网络。

3. 低精度量化（Low-precision Quantization）

低精度量化通常用低带宽、定点的整数表示网络参数和激活值，在量化过程中将连续的浮点数转换为离散值会损失部分网络精度，计算加速也通常需要特定的计算库或硬件支持。使用低位、定点数据的优势在于，内存访问的效率更高，且定点计算比浮点计算更为高效，非常适合存储带宽有限、算力较弱的设备。因此，低精度量化通常是模型部署中的必要步骤。2016 年，M. Rastegari 等[58]提出 XNOR-Net，将网络浮点数参数量化为二值数，在少量精度下滑的情况下，实现 32 倍的内存节省和 58 倍的卷积计算加速。2018 年，B. Jacob 等[59]提出一种训练中 Int8 量化方法，仅需在网络的前馈过程中对参数和激活值进行量化，训练结束后即可得到量化的模型。此外，NVIDIA 也在 2017 年公开了一套无需训练的 Int8 量化方法，量化时仅需少量数据对模型参数进行校准，在工业界中被广泛采用。

1.3 论文主要研究工作

论文针对车路协同系统的路侧感知部分，研究面向路侧感知的视觉目标检测算法。在 YOLOX 目标检测算法的基础上，设计了一种高效的路侧车辆行人检测模型，以及一种结合通道剪枝与知识蒸馏的模型压缩方法，并在实验室深度学习工作站和路侧边缘计算平台上对论文所提方法进行验证。论文的研究重点如下：

1. 针对路侧感知的检测精度要求，结合高效网络设计思想，设计高效的卷积结构改进 YOLOX 网络结构，并在训练中使用优化的分类与回归损失函数，以提升模型对小目标和密集目标的检测精度。

2. 针对路侧感知的实时性要求，结合通道剪枝和知识蒸馏算法，设计一种适合目标检测的模型压缩方法，在模型精度无明显下滑的前提下，显著降低模型所需的计算量，达到提升模型推理速度的目的。

3. 在校内建设的路侧平台上，采集并标注若干张图像作为测试集，用于测试所得模型在实际路侧场景中的检测精度，并结合模型部署工具和视频流推理框架，部署模型到路侧边缘计算设备，以测试模型的检测速度。

1.4 论文组织结构

论文的组织结构和章节主要内容如下：

第 1 章：绪论。分析了论文的研究背景和研究意义，并结合国内外研究现状对论文相关领域进行概括。

第 2 章：路侧视觉感知系统架构及相关技术。分析了路侧视觉感知系统的基本概念和相关技术，包括系统架构组成与关键问题分析、视觉目标检测算法、YOLOX 目标检测算法和模型压缩算法等。

第 3 章：路侧视觉目标检测模型的优化设计。针对路侧感知的检测精度要求，结合高效网络设计思想，在 YOLOX 目标检测算法的基础上进行改进，设计一种高效的目标检测算法，并在 BDD100K 公开数据集上验证算法的性能。

第 4 章：路侧视觉目标检测模型的压缩方法设计。针对路侧感知的实时性要求，结合通道剪枝与知识蒸馏算法，设计一种适合目标检测的模型压缩方法，并对模型部署工具、视频流推理框架以及模型部署的过程进行分析，最后使用 BDD100K 公开数据集对所提方法进行验证。

第 5 章：测试与验证。借助校内建设的路侧平台，部署设计的目标检测模型到路侧边缘计算设备，对模型的精度和实时性进行测试。

第 6 章：总结与展望。对全文工作进行总结，并展望未来的研究内容。

第 2 章 路侧感知系统架构及相关技术

路侧感知用于对车路协同系统中的交通对象进行实时状态感知，使用的传感器包括激光雷达、毫米波雷达和高清摄像头等。考虑到性能和成本等因素，采用毫米波雷达与高清摄像头融合的感知方案最为常见。因此，视觉目标检测是路侧感知系统中的重要组成部分。本章首先分析路侧感知系统的架构组成以及关键问题，然后对视觉目标检测算法进行概括，并对 YOLOX 目标检测算法进行分析，最后概括主要的模型压缩算法，包括结构化剪枝、知识蒸馏和低精度量化。

2.1 路侧感知系统架构组成与关键问题分析

2.1.1 系统架构组成

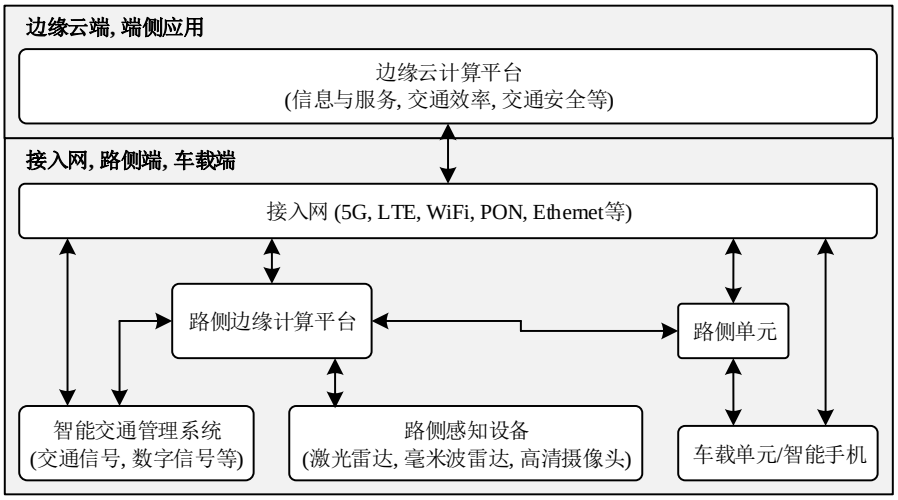


图 2.1 路侧感知系统架构

典型的路侧感知系统架构如图 2.1 所示。路侧边缘计算平台通过激光雷达、毫米波雷达和高清摄像头等感知设备，对道路交通状态进行实时感知，并将智能交通管理系统提供的交通信息和数字信号等一同发送到路侧单元（Road Side Unit，RSU）。RSU 则通过蜂窝车联网（Cellular Vehicle to Everything，C-V2X）通信将收到的信息广播给附近车载单元（On Board Unit，OBU）或智能手机，结合车载感知信息，从而实现车路一体化协同感知。协同感知获取的信息也将通过接入网上传到边缘云计

算平台，由区域交通调度管理系统进行处理，以实现信息与服务、交通安全和交通效率等车路协同应用。

2.1.2 关键问题分析

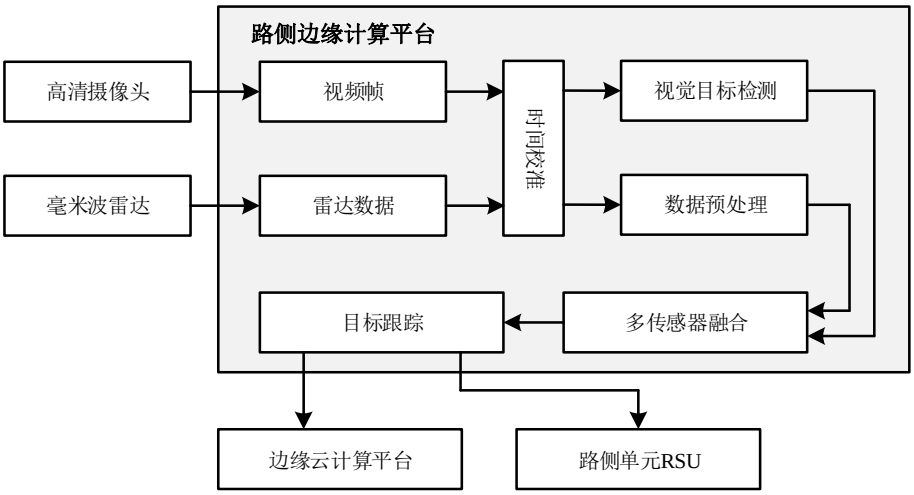


图 2.2 多传感器融合感知系统架构

在路侧感知系统中，以毫米波雷达和高清摄像头为感知设备的多传感器融合感知系统为例，典型的系统架构如图 2.2 所示。路侧边缘计算平台首先从高清摄像头和毫米波雷达分别获取实时视频帧和雷达输出的检测数据，为确保两部分数据的采集时间尽可能一致，需要执行一次时间校准操作。然后，将校准的视频帧输入视觉目标检测模型进行物体检测，同时对校准的雷达检测数据执行数据预处理操作，过滤掉无效数据和噪声。接着，将视觉检测结果与雷达检测数据进行多传感器融合计算，使视觉目标与雷达目标相互关联，得到融合后的目标信息，包括目标类别、位置、速度和方向等。最后，对融合感知的物体执行目标跟踪，并将稳定跟踪的目标信息发送到边缘云计算平台和路侧单元 RSU。

论文针对路侧感知系统中的视觉目标检测部分，研究的关键问题如下：

1. 路侧场景中的多尺度、多类别目标检测问题。路侧场景中背景较为复杂，车辆与行人目标的尺度多变，且密集目标、小目标较多，车辆也具有多种类别，对视觉目标检测算法的检测精度有较高要求。

2. 视觉目标检测算法的实时性与精度折衷问题。目前，现有的通用目标检测算法难以在路侧边缘计算设备上同时满足精度和实时性要求。需要设计高效的网络结构并结合模型压缩算法，在保持算法精度的同时，降低算法的计算量。

3. 边缘计算设备的多路高效检测问题。路侧边缘计算平台通常需同时处理多路摄像头的实时视频流数据，因此对目标检测模型的部署有较高要求，可借助模型部署工具和视频流推理框架，实现高效的多路视频流处理。

2.2 视觉目标检测算法概述

目标检测和图像分类、图像分割一起构成计算机视觉（Computer Vision，CV）领域的三大基本任务，并被广泛应用于视频监控、机器人导航、工业检测和航空航天等领域。目标检测的任务是从图像中定位感兴趣目标的位置，并对目标进行分类。目前，目标检测算法以深度神经网络方法为主，并可根据网络结构的不同分为 Two-stage 和 One-stage 两大类。尽管 One-stage 与 Two-stage 算法在网络结构方面有明显差异，但在算法框架上依然具有一致性。

2.2.1 视觉目标检测算法框架

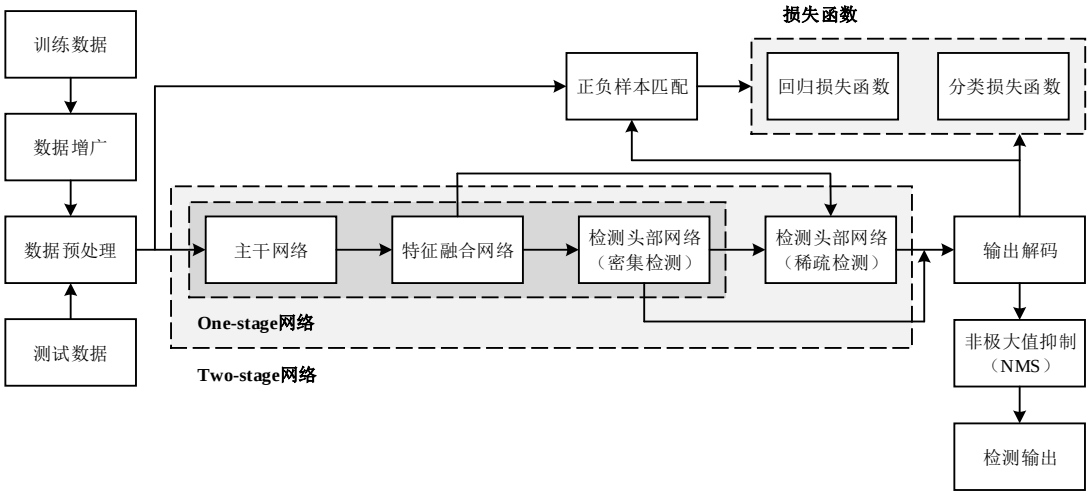


图 2.3 视觉目标检测算法框架

整合 One-stage 与 Two-stage 算法的目标检测框架如图 2.3 所示，其分为训练和测试两个部分。在训练部分，首先采集训练图像数据进行数据增广，目的是扩充训练数据集，然后对数据进行预处理，得到网络的输入。对于 One-stage 算法，网络包括主干网络、特征融合网络和密集检测头部网络，检测头部网络的输出经过解码后，结合输入数据进行正负样本匹配得到训练标签，最后分别计算网络输出的回归和分类损失。相比 One-stage 算法，Two-stage 算法包括额外的稀疏检测头部网络、

分类和回归损失函数，并将 One-stage 网络的密集检测头部视为区域生成网络 RPN。稀疏检测头部网络使用 RPN 网络的输出，从特征融合网络的输出提取潜在的前景目标特征进行进一步的分类和回归。在测试部分，首先对测试图像数据进行预处理，然后输入到 One-stage 或 Two-stage 网络得到预测输出，在对输出进行解码后，得到对潜在物体检测的边框、类别和置信度，最后执行一次非极大值抑制 NMS 操作去除冗余的检测边框，得到最终的检测结果。

2.2.2 网络结构组成

目标检测由于大多采用先预训练、后微调的训练方式，网络结构通常分为主干网络和检测头部网络两部分。在训练过程中，首先将分类网络在大规模分类数据集上进行预训练，然后将分类头部网络替换为检测头部网络，在检测数据集上进行微调。此后，为进一步提升目标检测算法的性能，特征融合网络被引入其中，并逐渐成为目标检测网络不可或缺的一部分。目前，主流的目标检测网络主要由主干网络、特征融合网络和检测头部网络组成。

1. 主干网络（Backbone Network）

主干网络用于提取图像的多尺度特征，决定了目标检测算法的整体复杂度，也是检测网络设计的关键部分。好的主干网络设计可在保持较低计算量的同时，具备较强的特征提取能力，从而使目标检测算法在实时性和精度方面达到较好的平衡。根据适用场景的不同，主干网络可分为两大类，一类是以 VGGNet[60]、ResNet[61]和 DenseNet[62]为代表的通用网络，此类网络通常具有参数量多、计算量大等特点，适用于具备较强算力的通用计算平台，例如工作站和云服务器等。另一类是以 SqueezeNet[63]、MobileNet[64]和 ShuffleNet[65]为代表的轻量化网络，此类网络一般参数量少且计算量小，适用于算力有限的嵌入式设备等。随着视觉 Transformer 的兴起，使用其作为目标检测主干网络的方法逐渐受到学术界的关注。其中，通用网络以 Swin Transformer[43]等作为代表，轻量化网络则有 MobileViT[66]等。视觉 Transformer 在检测数据集上微调后，能取得比卷积神经网络 CNN 更好的检测性能。但其通常需要千万级的图像数据集用于预训练。此外，视觉 Transformer 的计算量也较大，在达到相同精度的情况下，其效率通常比 CNN 低。

2. 特征融合网络（Neck Network）

特征融合网络用于对主干网络提取的多尺度特征进行融合，以提升模型对多尺度物体的检测性能。由于图像中的待检测物体通常具有不同大小的多种尺度，仅靠单一尺度的特征通常难以取得好的检测效果。因此，主流的目标检测算法会根据网络的感受野提取多个尺度的特征进行检测。然而，主干网络中的浅层网络感受野相对较小，提取的特征图分辨率较高，通常对语义信息的表示能力不足。深层网络则感受野较大，提取的特征图分辨率较低，具有丰富的语义信息，但会丢失较多空间细节信息。直接用多尺度特征进行检测的方式尽管能取得一定的效果，但对检测性能的提升有限。通过结合浅层特征的空间信息与深层特征的语义信息，特征融合网络能有效提升主干网络的表示能力。

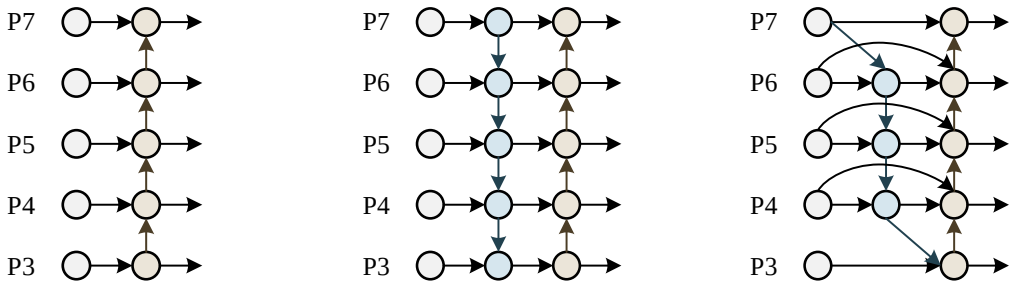


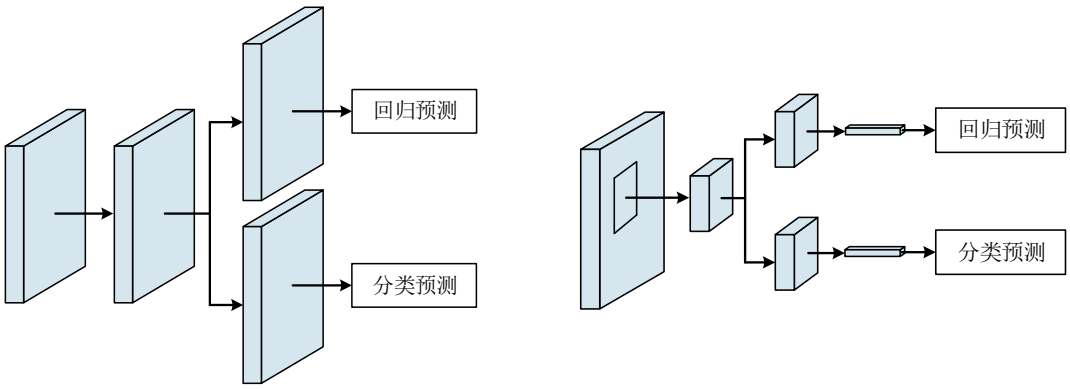
图 2.4 常见特征融合网络结构

常见的特征融合网络包括特征金字塔网络（Feature Pyramid Networks，FPN）[67]、PAFPN[68]和 BiFPN[36]等。典型的 FPN 结构如图 2.4 (a)所示，其输入为主干网络自底向上（Bottom-up）提取的多尺度特征 P3-P7，FPN 通过使用横向连接和上采样操作对多个尺度的特征进行自顶向下（Top-down）的融合，达到将深层特征的语义信息逐步向浅层特征传递的目的。FPN 的优点在于，相较于传统的图像金字塔方法，其通过重用特征的方式构建特征金字塔，在提升网络性能的同时显著减少计算量。典型的 PAFPN 结构如图 2.4 (b)所示，其在 FPN 的基础上，引入自底向上的特征融合通路，进一步实现特征空间信息与语义信息的相互结合。BiFPN 的网络结构则如图 2.4 (c)所示，其利用神经架构搜索 NAS，得到适用于视觉任务的高效特征融合网络结构，能有效提升网络的性能。

3. 检测头部网络（Head Network）

检测头部网络用于处理融合后的多尺度特征，对图像中的潜在目标进行定位和分类。相较于分类头部网络，检测头部网络能同时学习多种任务，是多任务学习

（Multi-task Learning）的典型实现方式。在目标检测任务中，检测头部网络除对目标进行边框回归和分类外，也可能对目标的置信度、质量或关键点等进行预测。典型的检测头部网络结构如图 2.5 所示，图 2.5 (a)是 One-stage 检测头部网络，其对单个尺度的特征进行密集预测，分别卷积得到空间分辨率相同的回归与分类预测分支。图 2.5 (b)是 Two-stage 检测头部网络，其使用 RPN 预测的 ROI，从单个尺度特征中提取对应的 ROI 特征，并通过卷积层和全连接层分别对 ROI 特征进行回归和分类预测。相较于 Two-stage 检测头部网络，One-stage 检测头部网络更为高效，其能一次完成对图像中目标的分类和回归预测，也因此更受工业界的青睐。



(a) One-stage（密集）检测头部网络结构 (b) Two-stage（稀疏）检测头部网络结构

图 2.5 One-stage 与 Two-stage 检测头部网络结构

2.2.3 数据增广

数据增广在视觉模型的训练中具有重要作用，能对现有数据集进行扩充，从而有效提升模型的泛化能力，减少模型因训练数据不足而出现的过拟合现象。通用的图像数据增广方法包括随机翻转、随机裁剪、随机旋转和颜色增强等。其中，随机翻转以一定的概率对图像进行随机左右或上下的翻转。随机裁剪使用随机生成的带有偏移量的图像边框截取原图像。随机旋转将图像顺时针或逆时针旋转一个随机的角度。颜色增强则对图像的亮度、色调和饱和度进行随机调整。

在通用数据增广方法之外，也有大量学者提出适合目标检测的数据增广方法，具有代表性的包括 Cutout[69]、Mixup[70]、Cutmix[71]和 Gridmask[72]等。其中，Cutout 将图像中的小块图像随机抹除。Mixup 按照一定的比例将两张图像进行混合。Cutmix 将一张图像中的小块图像随机裁剪，并粘贴到另一张图像中。Gridmask 则用随机生成的网格掩码，对图像中的部分区域进行抹除。

2.2.4 正负样本匹配

正负样本匹配用于提供网络在训练时的前景与背景标签，包括目标的类别和位置等，是决定网络能否正确执行目标检测任务的关键。常见的样本匹配方法包括最大交并比（Intersection over Union，IoU）分配[35]和自适应训练样本选择（Adaptive Training Sample Selection，ATSS）[73]等。

1. 最大交并比分配（Max IoU Assignment）

最大交并比分配是 Anchor-based 目标检测算法常用的样本匹配方法。在 Anchor-based 算法中，会根据网络输出的多尺度特征尺寸，在特征空间维度的每个点上预设若干个锚点（Anchor point），每个锚点包括一个特定长宽的锚框（Anchor box）。最大交并比分配正是针对 Anchor-based 算法的特点设计的，其样本匹配的流程如表 2.1 所示。

表 2.1 最大交并比分配算法

算法: 最大交并比分配

输入: 一张图像的N个物体标注边框: $B^{gt} = \{b_1^{gt}, \dots, b_N^{gt}\}$

网络预设的M个锚框: $M = \{b_1, \dots, b_M\}$

负样本 IoU 阈值: t_n, 默认值$t_n = 0.4$

正样本 IoU 阈值: t_p, 默认值$t_n = 0.5$

输出: M个锚框匹配的样本: $S = \{s_1, \dots, s_M\}$

1. 按照锚框所在特征的空间位置和步长, 将锚框逐个放置到图像对应位置;
2. 初始化$M \times N$的 IoU 矩阵: $K \leftarrow 0_{M \times N}$;
3. **for** each index $i \in [1, M]$ **do**
4. **for** each index $j \in [1, N]$ **do**
5. 计算B_i与B_j^{gt}的 IoU 值: $K_{ij} = IoU(B_i, B_j^{gt})$;
6. **end for**
7. **end for**
8. 计算K中每行最大值及其对应的行索引: $V^{max}, I^{max} = \text{Max}(K, dim = 1)$;
9. **for** each index $i \in [1, M]$ **do**
10. **if** $V_i^{max} < t_n$ **then**
11. 标记为负样本, 其值为 0: $S_i = 0$;
12. **else if** $V_i^{max} \geq t_n$ and $V_i^{max} \leq t_p$ **then**
13. 标记为忽略样本, 其值为-1, 不参与损失函数的计算: $S_i = -1$;

表 2.1 最大交并比分配算法（续表）

算法: 最大交并比分配
14. **else if** $V_i^{max} \geq t_p$ **then**
15. 取匹配的标注边框索引: $k = I_i^{max}$;
16. 标记为正样本, 其值为匹配的标注边框: $S_i = B_k^{gt}$;
17. **end if**
18. **end for**
19. **return** S;

尽管最大交并比分配简单实用，但其也存在一些明显的问题。例如部分小目标可能因没有合适的锚框与之匹配，导致在网络训练的过程中被忽略。因此，为提升检测性能，Anchor-based 算法通常会放置不同尺度和长宽比的密集锚框，以尽可能确保图像中的物体均能被匹配到。但也因此会增加样本匹配和边框编解码的计算量，并在训练时占用更多的存储空间。

2. 自适应训练样本选择 ATSS

ATSS 是 Anchor-free 目标检测算法常用的样本匹配方法。相较于 Anchor-based 算法，Anchor-free 算法可看作在特征空间维度的每个点上仅预设一个锚点，且没有对应的锚框。ATSS 的特点在于，利用 IoU 的均值和方差，可自适应确定样本分配的阈值，并能使 Anchor-free 算法能达到 Anchor-based 算法的检测性能。

表 2.2 自适应训练样本选择算法

算法: 自适应训练样本选择 (ATSS)
输入: 一张图像的物体标注边框集合: G
网络输出的多尺度特征层数: L
第i层多尺度特征的锚点集合: A_i
网络预设的全部锚点集合: A
ATSS 超参数: k, 默认值$k = 9$
输出: 正样本集合: P
负样本集合: N
1. 按照锚点所在特征的空间位置和步长, 将锚点逐个投影到图像对应位置;
2. **for** each ground-truth $g \in G$ **do**
3. 构建一个放置g的候选正样本的空集: $C_g \leftarrow \emptyset$;
4. **for** each level $i \in [1, L]$ **do**
5. $M_i \leftarrow$从A_i中选择k个中心离g中心 L2 距离最近的锚点;

表 2.2 自适应训练样本选择算法（续表）

算法: 自适应训练样本选择 (ATSS)
6. 将M_i加入到C_g中: $C_g = C_g \cup M_i$;
7. **end for**
8. 计算C_g与g的 IoU 值: $D_g = IoU(C_g, g)$;
9. 计算D_g的均值: $m_g = \text{Mean}(D_g)$;
10. 计算D_g的方差: $v_g = \text{Std}(D_g)$;
11. 计算g匹配的 IoU 阈值: $t_g = m_g + v_g$;
12. **for** each candidate $c \in C_g$ **do**
13. **if** $IoU(c, g) \geq t_g$ and center of c in g **then**
14. 将c加入到P中, 当c已被匹配时, 取 IoU 值大的一个作为匹配目标: $P = P \cup c$;
15. **end if**
16. **end for**
17. **end for**
18. 取A与P的差集: $N = A - P$;
19. **return** P, N;

ATSS 样本匹配的流程如表 2.2 所示。通过采用两步匹配的方式，其在第一步中利用先验知识大幅缩小搜索范围，并在第二步根据自适应阈值进一步筛选剩余的正样本。相较于最大交并比分配，ATSS 无需预设锚框，所需的锚点更少且计算量更小，在训练时也更加节省存储空间。由于每个物体标注边框的样本匹配 IoU 阈值各不相同，且分配到的正样本数量相近，使得 ATSS 样本匹配的灵活度更高，能加速网络收敛，并能帮助网络更好地学习物体特征。

2.2.5 损失函数

在监督学习（Supervised Learning）问题中，损失函数用于度量预测值与标签值之间的不一致程度。两者之间的误差越小说明模型的输出与真实数据分布越接近，模型的性能也越好。目标检测使用的损失函数包括分类和回归两大类，分别用于对待检测目标进行分类和边框回归。

1. 分类损失函数（Classification Loss Function）

目标检测常见的分类损失函数包括交叉熵损失（Cross Entropy Loss）、Focal Loss[35]和 Generalized Focal Loss[74]等。

(1) 交叉熵损失

交叉熵损失的表达式如下：

$$L = -(y\log(p) + (1-y)\log(1-p)) \tag{2.1}$$

式中，y——样本类别标签，取值范围为[0, 1]

p——预测值

当样本标签值为 0 或 1 时，交叉熵损失可简化为表达式如下：

$$L = -\log(p) \tag{2.2}$$

交叉熵损失的不足在于，当训练 One-stage 网络时，会存在正负样本比例失衡的情况。由于负样本数量远多于正样本，且多数负样本是容易分类的，可能使网络在优化过程中被大量的简单负样本所主导，从而降低网络性能。

(2) Focal Loss

在交叉熵损失的基础上，Focal Loss 采用样本加权的方式平衡正负样本在网络训练过程中的优化比重，使得网络能着重关注困难样本。

Focal Loss 的表达式如下：

$$L = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{2.3}$$

式中，α_t——正负样本加权系数，通常正样本取$\alpha_t = 0.25$，负样本取$\alpha_t = 0.75$

p_t——预测值，正样本取$p_t = p$，负样本取$p_t = 1-p$

γ ——难易样本加权系数，通常取$\gamma = 2.0$

通过在训练中降低简单样本的权重，同时平衡负样本的权重，Focal Loss 能有效缓解目标检测任务中正负样本比例失衡的问题，使得 One-stage 算法能具有媲美 Two-stage 算法的检测性能。

(3) Generalized Focal Loss

在 Focal Loss 的基础上，Generalized Focal Loss 针对 Anchor-free 算法中分类与回归分支存在预测不一致的问题，采用预测边框与标注边框的 IoU 值作为分类软标签，在无需质量预测分支的情况下，有效提升了网络的检测性能。

Generalized Focal Loss 的表达式如下：

$$L = -|y-\alpha|^{\beta}(y\log(\alpha) + (1-y)\log(1-\alpha)) \tag{2.4}$$

式中，y——IoU 类别软标签，取值范围为[0, 1]

α——预测值

β——样本加权系数，通常取$\beta = 2.0$

2. 回归损失函数（Regression Loss Function）

常见的回归损失包括 Smooth L1 损失[35]、IoU 损失、GIoU（Generalized IoU）损失[75]、DIoU(Distance IoU)损失[76]和 CIoU(Complete IoU)损失[76]等。

(1) Smooth L1 损失

Smooth L1 损失为 L1 与 L2 范数损失的结合体，其通过分段函数的形式，在误差值大于等于 1 时使用 L1 范数损失，当误差值小于 1 时则使用 L2 范数损失。如此能克服 L2 范数损失对离群点敏感的问题，也能缓解 L1 范数损失在误差小于 1 时导数不连续、优化效率低的不足。

Smooth L1 损失的表达式如下：

$$L = \begin{cases} 0.5(y-p)^2, & if|y-p| < 1.0 \\ |y-p| - 0.5, & otherwise \end{cases} \tag{2.5}$$

式中，y——样本回归标签，取值大于等于零

p——预测值

(2) IoU 损失

在目标检测中，度量边框回归精度的指标通常采用 IoU，其表示预测边框与标注边框之间的相交程度。因此，相比使用 Smooth L1 等损失进行间接边框回归的方式，直接使用 IoU 损失引导网络优化则更为直接有效。

IoU 损失的表达式如下：

$$L = 1.0 - \frac{B \cap B^{gt}}{B \cup B^{gt}} \tag{2.6}$$

式中，$B \cap B^{gt}$——预测边框与标注边框相交的面积

$B \cup B^{gt}$——预测边框与标注边框总面积减去相交的面积

可以看出，IoU 损失值会随 IoU 值的增大而减小，能直观地反应预测边框回归的状态，且 IoU 损失具有尺度不变性，其值不会随边框大小而改变。但 IoU 损失的问题在于，当预测边框与标注边框不相交时，IoU 损失值恒为 1，不能度量两个边框之间的距离，导致网络在此种情况下难以优化。此外，当预测边框与标注边框存在部分重叠时，IoU 损失也不能准确反映两个边框之间的重叠形态。

(3) GIoU 损失

针对 IoU 损失的问题，GIoU 引入最小包围边框的概念，使得在两个边框无相交时，能通过两者的最小包围边框面积来间接度量边框之间的距离。

GIoU 损失的表达式如下：

$$L = 1.0 - \frac{B \cap B^{gt}}{B \cup B^{gt}} + \frac{C - (B \cup B^{gt})}{C} \tag{2.7}$$

式中，C——包围预测边框与标注边框的最小边框面积

当两个边框相距非常远时，GIoU 损失值趋近于 2，并会随边框之间的距离拉近而减小。因此，通过引入最小包围边框，GIoU 损失能使预测边框不断向标注边框靠近。但在训练过程中，GIoU 损失倾向于先扩大预测边框，让其与标注边框相交，然后再逐渐缩小预测边框的优化方式，会导致训练需要更多的迭代次数。此外，当两个边框存在包含关系时，GIoU 损失也会退化为 IoU 损失。

(4) DIoU 损失

针对 GIoU 损失的不足，DIoU 损失利用最小包围边框的对角线距离和预测边框与标注边框之间的中心点距离，对 IoU 损失进行约束。

DIoU 损失的表达式如下：

$$L = 1.0 - \frac{B \cap B^{gt}}{B \cup B^{gt}} + \frac{\rho^2(B, B^{gt})}{c^2} \tag{2.8}$$

式中，$\rho^2(B, B^{gt})$——预测边框与标注边框的中心点欧式距离

c ——最小包围边框的对角线欧式距离

DIoU 损失的优点在于，直接对两个边框之间的距离进行度量，使得无需在训练中产生较大的最小包围边框，从而提升边框回归的收敛速度。但当两个边框的中心点重合时，DIoU 损失还是会退化为 IoU 损失。

(5) CIoU 损失

在 DIoU 损失的基础上，CIoU 损失进一步引入预测边框和标注边框的长宽比，以约束预测边框的形态。通过对两个边框的长宽比一致性进行度量，CIoU 损失使得预测边框在与标注边框中心点重合的情况下，依然能快速向标注边框靠近。

CIoU 损失的表达式如下：

$$L = 1.0 - \frac{B \cap B^{gt}}{B \cup B^{gt}} + \frac{\rho^2(B, B^{gt})}{c^2} + \alpha v \tag{2.9}$$

其中：

$$\alpha = \frac{v}{\left(1.0 - \frac{B \cap B^{gt}}{B \cup B^{gt}}\right) + v} \tag{2.10}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{B^{gt}}}{h^{B^{gt}}} - arctan\frac{w^{B}}{h^{B}}\right) \tag{2.11}$$

式中，α ——权重系数

v ——长宽比一致性质量评估参数

w^B, h^B ——预测边框的宽度和高度

$w^{B^{gt}}, h^{B^{gt}}$——标注边框的宽度和高度

2.2.6 非极大值抑制

除完全端到端目标检测算法外，其余目标检测算法在训练过程中，通常会在样本匹配时分配多个正样本用于检测单个目标。因此，在对图像进行物体检测时，会不可避免地产生大量冗余的检测边框。非极大值抑制 NMS 正是用于在冗余的检测边框中挑选出质量最好的一个，并去除其余与之高度重叠的边框。NMS 的示意图如图 2.6 所示，图 2.6 (a)是检测后未执行 NMS 的效果，可以看出存在有大量的冗余检测结果。图 2.6 (b)是执行 NMS 后的效果，此时每个目标仅对应一个检测边框。具体的 NMS 操作流程如表 2.3 所示。

(a) 执行 NMS 操作前 (b) 执行 NMS 操作后

图 2.6 非极大值抑制 NMS 示意图

NMS 的不足在于，需要手动设置一个 IoU 阈值超参数。但在不同的场景中，通常 IoU 阈值都是不同的，需要根据实际情况进行调整。此外，当两个目标较为接近时，NMS 会倾向于去除其中一个目标的全部检测结果，从而造成漏检。针对 NMS 的问题，Soft-NMS[77]对高度重叠的冗余边框进行置信度加权，重叠区域越大则置信度衰减越严重，当置信度衰减到低于预设的阈值时则被去除。尽管 Soft-NMS 引入了另一个置信度阈值超参数，且计算量也相比 NMS 更大，但在待检测目标较为密集的场景中，能取得更好的检测效果。

表 2.3 非极大值抑制算法

算法: 非极大值抑制 (NMS)

输入: 一张图像的N个检测边框: $B = \{b_1, \dots, b_N\}$

检测边框对应的置信度: $S = \{s_1, \dots, s_N\}$

NMS 阈值: t_{nms}

输出: 处理后的检测边框: D

检测边框对应的置信度: C

while $B \neq \emptyset$ **do**
  获取S中最大值的索引: $k = \text{ArgMax}(S)$;
  取出并去除B中第k项: $b = B_k$; $B = B - B_k$;
  将b加入到D中: $D = D \cup b$;
  取出并去除S中第k项: $s = S_k$; $S = S - S_k$;
  将s加入到C中: $C = C \cup s$;
  for each index $i \in [1, \text{length of } B]$ **do**
    if $IoU(b, B_i) \geq t_{nms}$ **then**
      将B_i从B中移除: $B = B - B_i$;
      将S_i从S中移除: $S = S - S_i$;
    end if
  end for
end while
return D,C;

2.3 YOLOX 目标检测算法分析

YOLO 系列目标检测算法一直是高效检测的代名词，在平衡检测速度与精度方面极为出色，并受到学术界和工业界的广泛关注。在 2020-2021 年间，YOLO 先后迎来重大更新。首先是 A. Bochkovskiy 等[38]提出 YOLOv4，在保证实时性的同时，大幅提升 YOLO 的检测精度。然后，G. Jocher 等提出 YOLOv5，其检测精度与 YOLOv4 接近，但具有更快的检测速度。在 YOLOv5 等的基础上，Z. Ge 等[78]提出 YOLOX，将 Anchor-free 范式引入 YOLO 算法，进一步提升了检测性能。YOLOX 也是目前最为高效的目标检测算法之一，适合应用于路侧感知系统。

2.3.1 CSPDarkNet 主干网络与 PAFPN 特征融合网络

YOLOX 的主干网络使用的是 CSPDarkNet[38]，特征融合网络使用的则是 PAFPN[68]，两者均来自 YOLOv5。其中，CSPDarkNet 起初由 A. Bochkovskiy 等提出，并作为 YOLOv4 的主干网络。然后，G. Jocher 等又将经过修改的 CSPDarkNet 作为 YOLOv5 的主干网络。PAFPN 特征融合网络最初被应用于图像分割任务，随后 Bochkovskiy 等将其引入到目标检测任务中。

1. CSPDarkNet 主干网络

CSPDarkNet 是结合跨阶段局部网络 CSPNet 和 YOLOv3 主干网络 DarkNet 的产物。其中，CSPNet 是由 C. Y. Wang 等[79]提出的一种高效主干网络，能在训练中引入更为丰富的梯度信息组合，从而减少网络所需的实际计算量。DarkNet 则是由 J. Redmon 等[33]在 ResNet 的基础上修改所得。相较于 ResNet，其具备更好的检测精度，同时推理的速度也更快。

表 2.4 CSPDarkNet 网络结构组成

输入尺寸	结构类型	Bottleneck 数量	残差连接	步长	输出
3×640×640	Focus	—	—	2	—
64×320×320	ConvBlock	—	—	2	—
128×160×160	CSPBottleneck	3	True	1	—
128×160×160	ConvBlock	—	—	2	—
256×80×80	CSPBottleneck	9	True	1	P3
256×80×80	ConvBlock	—	—	2	—
512×40×40	CSPBottleneck	9	True	1	P4
512×40×40	ConvBlock	—	—	2	—
1024×20×20	SPPBottleneck	—	—	1	—
1024×20×20	CSPBottleneck	3	False	1	P5

CSPDarkNet 的网络结构组成如表 2.4 所示。其中，表格第 3 列指 CSPBottleneck 结构中 Bottleneck 的数量。表格第 4 列指 Bottleneck 是否具有残差连接。表格第 5 列指当前结构中卷积层的步长。表格第 6 列指当前结构的输出特征是否标记为网络输出。从网络结构分析，CSPDarkNet 由四种卷积结构组成，其分别是 ConvBlock、Focus、CSPBottleneck 和 SPPBottleneck。假设网络的输入图像尺寸为 36×40×640，在经过表中各结构顺序处理后，将得到多尺度输出特征 P3、P4 和 P5，其对应的特

征尺寸分别为 256×80×80、512×40×40 和 1024×20×20。网络输出的多尺度特征将作为特征融合网络 PAFPN 的输入。

(1) ConvBlock 网络结构

ConvBlock 的网络结构如图 2.7 所示，其由卷积层 Conv2d、批归一化层 BatchNorm2d 和激活函数 SiLU 组成，是 CNN 中最为常见的一种卷积结构。ConvBlock 的参数主要包括输入通道数、输出通道数、卷积核大小、步长和填充大小等。其中，填充大小固定为卷积核大小整除 2，以使卷积所得特征的尺寸与卷积核大小无关。在 YOLOX 网络中，ConvBlock 也是 CSPDarkNet、PAFPN 和检测头部网络的基本组成部分。

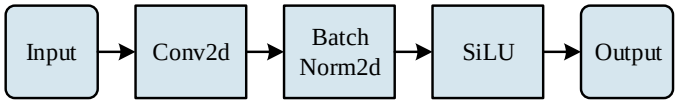


图 2.7 ConvBlock 网络结构

(2) Focus 结构

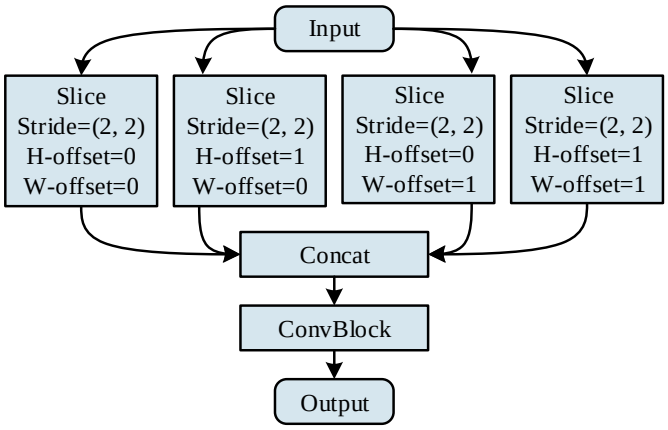
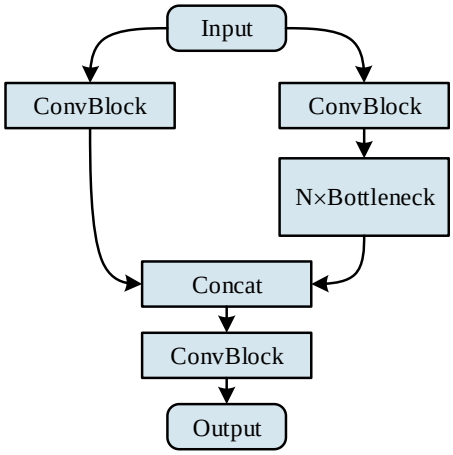


图 2.8 Focus 网络结构

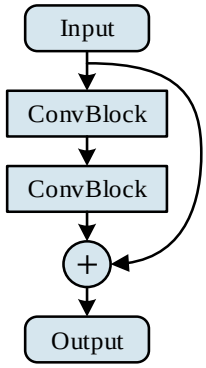
Focus 的网络结构如图 2.8 所示，其首先通过四个 Slice 切片层从不同的宽高偏移位置对输入图像的空间维度进行步长为 2 的切片，使得图像的空间分辨率减半，但通道数增加为原来的四倍。然后，用 Concat 层拼接切片的图像张量，并经过 ConvBlock 得到输出特征。在 CSPDarkNet 中，Focus 结构用于对输入图像执行下采样操作。相较于直接用 ConvBlock 进行下采样的方式，Focus 结构可保留更多的图像信息，在仅增加少量计算量的情况下，能有效提升网络的表示能力。

(3) CSPBottleneck 结构

CSPBottleneck 的网络结构如图 2.9 (a)所示，其包括左、右两个分支。左侧分支仅用 ConvBlock 对输入特征进行降维，使得输入特征的通道数减半。右侧分支则首先执行与左侧分支相同的降维操作，然后用若干个 Bottleneck 结构处理降维后的特征。Bottleneck 网络结构如图 2.9 (b)所示，其用两个 ConvBlock 对输入特征进行卷积得到输出特征。当具有残差连接时，输出特征会再与输入特征逐元素相加。在将左、右两个分支所得特征通过 Concat 层拼接后，最后经过 ConvBlock 得到输出特征。比起用若干个 Bottleneck 结构直接处理特征的方式，CSPBottleneck 所需的计算量更少，且能获得更好的检测性能。



(a) CSPBottleneck 网络结构



(b) Bottleneck 网络结构

图 2.9 CSPBottleneck 与 Bottleneck 网络结构

(4) SPPBottleneck 结构

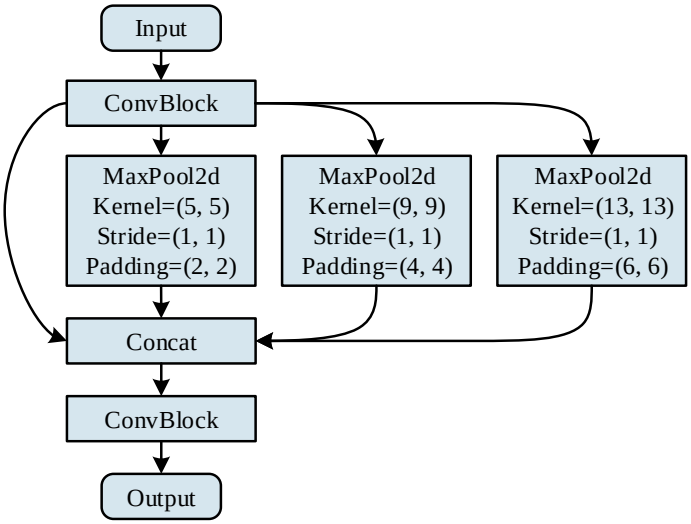


图 2.10 SPPBottleneck 网络结构

SPPBottleneck 的网络结构如图 2.10 所示，其在主干网络中以插件的形式存在，用于显著增加网络深层特征的感受野。SPPBottleneck 首先用 ConvBlock 对输入特征进行降维，然后用三个池化核大小分别为 5×5、9×9 和 13×13 的最大池化层 MaxPool2d 处理降维后的特征。在用 Concat 层拼接各分支特征后，最后经过 ConvBlock 得到输出特征。SPPBottleneck 在增加少部分计算量的情况下，能有效提升网络的检测性能，也因此在目标检测中被广泛应用。

2. PAFPN 特征融合网络

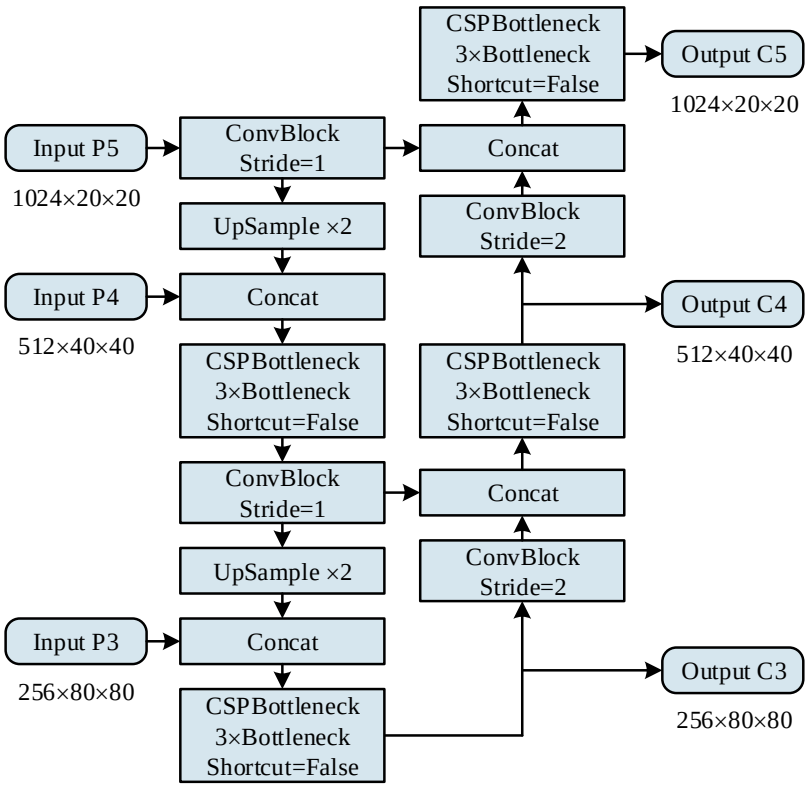


图 2.11 PAFPN 网络结构

PAFPN 的网络结构如图 2.11 所示，其由自顶向下和自底向上的两部分网络组成。在自顶向下的网络中，从输入特征 P5 开始，通过卷积、上采样和拼接操作，先后与输入特征 P4 和 P3 进行特征融合，并得到融合后的中间特征。在自底向上的网络中，从 P3 对应的中间特征开始，通过卷积、下采样和拼接操作，先后与 P4 和 P5 对应的中间特征进行特征融合，最终得到特征尺寸与输入特征相同的输出特征 C3、C4 和 C5。在 YOLOX 网络中，PAFPN 使用的每个 CSPBottleneck 结构均包括三个无残差连接的 Bottleneck。在 FPN 的基础上，PAFPN 通过引入额外的自底向上网络通路，能更好地提升网络的表示能力。

2.3.2 分类与回归解耦的检测头部网络

YOLOX 的检测头部网络结构如图 2.12 所示，其用三个参数不共享的网络，分别处理特征融合网络 PAFPN 输出的 C3、C4 和 C5 多尺度特征。假设网络预测 C 个类别，对于空间尺寸为 H×W 的单个尺度特征，首先用 ConvBlock 对输入特征进行降维，以减少后续网络的计算量。然后，分别用两组包括两个 ConvBlock 的网络卷积特征，得到解耦的分类与回归分支。对于分类分支，用 Conv2d 卷积得到分类预测。对于回归分支，则用两个 Conv2d 分别卷积得到置信度和边框回归预测。分类与回归解耦的检测头部网络优点在于，能明显提升模型的收敛速度，并能缓解分类和回归特征存在冲突的问题。

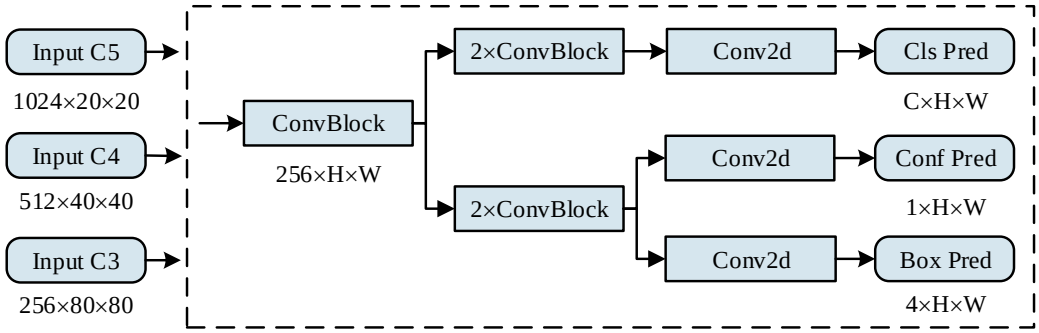


图 2.12 YOLOX 检测头部网络结构

2.3.3 模型扩展策略

表 2.5 YOLOX 模型扩展参数

模型名称	深度系数（depth）	宽度系数（width）
YOLOX-s	0.33	0.50
YOLOX-m	0.66	0.75
YOLOX-l	1.00	1.00
YOLOX-x	1.33	1.25

YOLOX 采用与 YOLOv5 相同的模型扩展策略，通过对网络的深度和宽度进行缩放，得到不同参数量的系列网络。在工业化部署时，可根据设备的算力和应用场景需求，选择合适大小的网络。YOLOX 模型扩展的参数如表 2.5 所示，其包括 YOLOX-s、YOLOX-m、YOLOX-l 和 YOLOX-x 四个模型，模型的网络深度与宽度系数依次递增，代表网络的通道数和层数逐个增加。以 YOLOX-s 为例，网络深度

指网络中 CSPBottleneck 中的 Bottleneck 数量为默认的 0.33 倍，网络宽度则指网络中每层的通道数为默认的 0.50 倍。

2.3.4 Mosaic 与 Mixup 数据增广

YOLOX 采用的数据增广方法包括 Mosaic[38]和 Mixup[70]等。Mosaic 最初在 YOLOv4 中被引入，并在随后作为 YOLOv5 和 YOLOX 的数据增广方法。Mixup 则是在图像分类中使用的一种简单且与数据无关的数据增广方法。通过结合 Mosaic 与 Mixup，能在训练时增加训练图像的背景复杂度，减少模型发生过拟合现象的可能性，并能有效提升模型的检测性能。

(a) Mosaic 示意图 (b) Mixup 示意图

图 2.13 Mosaic 与 Mixup 示意图

Mosaic 与 Mixup 的示意图如图 2.13 所示，图 2.13 (a)是 Mosaic 的示意图，其将四张图像经过随机裁剪后，拼成一张图像作为训练数据，使得图像的背景更为丰富，有助于减少 BN 层在统计特征均值和方差时对批大小（Batch Size）的依赖，使得模型能在单张显卡上完成训练。此外，Mosaic 对小目标检测也有一定的帮助。图 2.13 (b)是 Mixup 的示意图，其按照一定的比例混合两张图片，得到虚拟的训练数据，能在一定程度上提升模型的泛化能力。

2.3.5 SimOTA 正负样本匹配

SimOTA[78]样本匹配是最优传输分配（Optimal Transport Assignment，OTA）[80]算法的简化版本。OTA 将正负样本匹配视为最优传输问题进行求解，其可在全局视角下，结合网络损失值和中心先验（Center Prior）知识，自适应地确定每个标注框分配的正样本数量。相较于 OTA，SimOTA 通过引入 Dynamic Top-k 匹配策略，从而省去了 OTA 耗时的线性匹配过程，使得样本匹配更为高效。SimOTA 的样本匹配流程如表 2.6 所示。

表 2.6 简化的最优传输分配算法

算法: 简化的最优传输分配算法 (SimOTA)

输入: 一张图像的物体标注边框集合: G

网络输出的多尺度特征层数: L

第i层多尺度特征的步长: s_i

第i层多尺度特征的锚点集合: A_i

网络预设的全部锚点集合: A

SimOTA 超参数: k, 默认值$k = 10$

中心先验超参数: r, 默认值$r = 2.5$

输出: 正样本集合: P

负样本集合: N

1. 按照锚点所在特征的空间位置和步长, 将锚点逐个投影到图像对应位置;
2. **for** each ground-truth $g \in G$ **do**
3. 构建一个放置g的候选正样本的空集: $C_g \leftarrow \emptyset$;
4. **for** each level $i \in [1, L]$ **do**
5. $M_i \leftarrow$从A_i中选择落在以g中心向外扩展$r \times s_i$距离的矩形区域内的锚点;
6. 将M_i加入到C_g中: $C_g = C_g \cup M_i$;
7. **end for**
8. 计算C_g的分类损失: $L_{cls} = \text{FocalLoss}(C_g, g)$;
9. 计算C_g的边框回归损失: $L_{bbox}, V^{iou} = \text{IoULoss}(C_g, g)$;
10. 计算匹配代价: $L_{cost} = L_{cls} + 3 \times L_{bbox}$;
11. 取V^{iou}中最小的k个值: $V^{topk} = \text{Topk}(V^{iou}, \text{k} = k)$;
12. 将V^{topk}相加取整得到g应分配的正样本数量: $n = \text{Int}(\text{Sum}(V^{topk}))$;
13. 取L_{cost}中最小的n个值对应的索引: $I^{topk} = \text{ArgTopk}(L_{cost}, \text{k} = n)$;
14. **for** each index $i \in I^{topk}$ **do**

表 2.6 简化的最优传输分配算法（续表）

算法: 简化的最优传输分配算法 (SimOTA)
15. 取i对应的候选正样本: $c \leftarrow i$-th candidate of C_g;
16. 将c加入到P中, 当c已被匹配时, 取L_{cost}值小的一个作为匹配目标: $P = P \cup c$;
17. **end for**
18. **end for**
19. 取A与P的差集: $N = A - P$;
20. **return** P, N;

2.3.6 交叉熵分类与 IoU 回归损失函数

YOLOX 检测头部网络的输出包括边框回归、置信度和分类三个分支。其中，边框回归分支用于预测物体的边框，置信度分支用于区分图像中的物体与背景，分类分支则用于对物体进行进一步分类。

1. 边框回归损失函数

边框回归分支使用 IoU 的二次幂损失函数，在计算损失值时仅用到正样本。

具体的损失函数表达式如下：

$$L_{bbox} = \frac{1}{N^{obj}} \sum_{k=0}^{M} \sum_{i=0}^{H^k} \sum_{j=0}^{W^k} 1_{kij}^{obj} \times (1.0 - IoU^2(B_{kij}, B_{kij}^{gt})) \tag{2.12}$$

式中，N^{obj} ——单个批次训练数据中正样本的数量

M ——多尺度特征的数量

H, W ——单个尺度特征的高度和宽度

1^{obj} ——正样本掩码，正样本取 1，负样本则取 0

B, B^{gt}——预测边框与对应的标注边框

2. 置信度损失函数

置信度分支使用交叉熵损失，在计算损失值时用到全部正负样本。

具体的损失函数表达式如下：

$$L_{conf} = \frac{1}{N^{obj}} \sum_{k=0}^{M} \sum_{i=0}^{H^k} \sum_{j=0}^{W^k} \text{CrossEntropy}(p_{kij}, y_{kij}) \tag{2.13}$$

式中，p——预测值

y——样本类别标签

3. 分类损失函数

分类分支同样使用交叉熵损失，但在计算损失值时仅用到正样本。此外，分类分支的标签是正样本预测边框与对应标注边框的 IoU 值。

具体的损失函数表达式如下：

$$L_{cls} = \frac{1}{N^{obj}} \sum_{k=0}^{M} \sum_{i=0}^{H^k} \sum_{j=0}^{W^k} 1_{kij}^{obj} \times \text{CrossEntropy}(p_{kij}, q_{kij}) \tag{2.14}$$

式中，q——IoU 类别软标签

训练过程中 YOLOX 的总损失函数表达式如下：

$$L_{total} = \lambda_{bbox} L_{bbox} + \lambda_{conf} L_{conf} + \lambda_{cls} L_{cls} \tag{2.15}$$

式中，λ_{bbox}——边框回归损失平衡系数

λ_{conf}——置信度损失平衡系数

λ_{cls} ——分类损失平衡系数

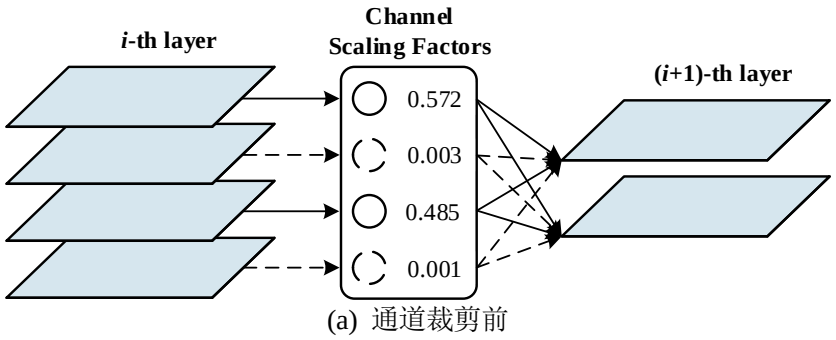
2.4 模型压缩算法概述

深度学习在自动语音识别（Automatic Speech Recognition，ASR）、CV 和 NLP 等领域均取得了显著的成果。现阶段的人工智能（Artificial Intelligence，AI）应用也主要依赖深度学习算法。然而，深度学习模型的参数量通常较多，计算量也较为巨大，需要借助模型压缩算法减少模型所需的计算量与存储空间，从而满足 AI 应用的实时性需求。目前，模型压缩几乎是模型部署的必要步骤，而主流的模型压缩算法包括结构化剪枝、知识蒸馏和低精度量化等。

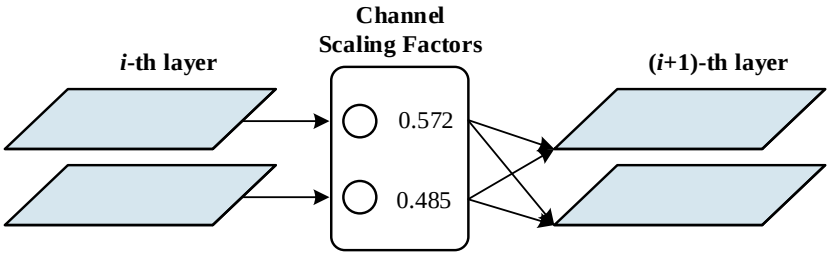
2.4.1 结构化剪枝

结构化剪枝通过移除网络中的部分卷积核或整组通道参数，以减少模型的参数和计算量。在结构化剪枝方法中，以通道为裁剪单位的剪枝方法最为常见，其通常包括通道筛选和精度恢复两个步骤。其中，通道筛选的目的是通过特定方法，例如 L1 范数惩罚和 LASSO 回归[50]等，鉴别在当前任务中网络参数的重要性。通常认为低重要性的通道参数不会对网络的性能产生太大影响，可以被安全去除。精度恢复则指在完成通道筛选及裁剪低重要性的部分通道参数后，微调剩余的网络参数，从

而恢复网络因参数裁剪而丢失的精度。此外，也有一些无需微调的通道剪枝方法，其在训练中逐步筛选通道，并同时恢复网络的精度。



(a) 通道裁剪前



(b) 通道裁剪后

图 2.14 通道剪枝示意图

典型的通道剪枝方法如图 2.14 所示，其在待裁剪的 CNN 中插入通道缩放因子（Channel Scaling Factors），并在训练中对其施加稀疏惩罚，用于自适应评估网络通道的重要性。图 2.14 (a)为训练完成，但未进行通道裁剪的网络示意图。其中，通道缩放因子的部分元素趋近于零，表明对应通道的重要性相对较低。图 2.14 (b)为裁剪后的网络示意图，其通过裁剪网络中相对低重要性的部分通道参数，以得到紧凑的网络。但此时网络的精度会相应下滑，还需对其进行微调。

2.4.2 知识蒸馏

知识蒸馏（Knowledge Distillation）引入教师网络和学生网络的概念，其将教师网络的输出作为软标签，用于监督学生网络的学习，达到将教师网络的知识传递给学生网络的目的。教师网络的性能通常优于学生网络，通过模仿教师网络的输出，能使学生网络获得比自身更好的性能。利用此特性，可将大网络作为教师网络，并将裁剪参数得到的小网络作为学生网络，从而实现模型压缩的效果。

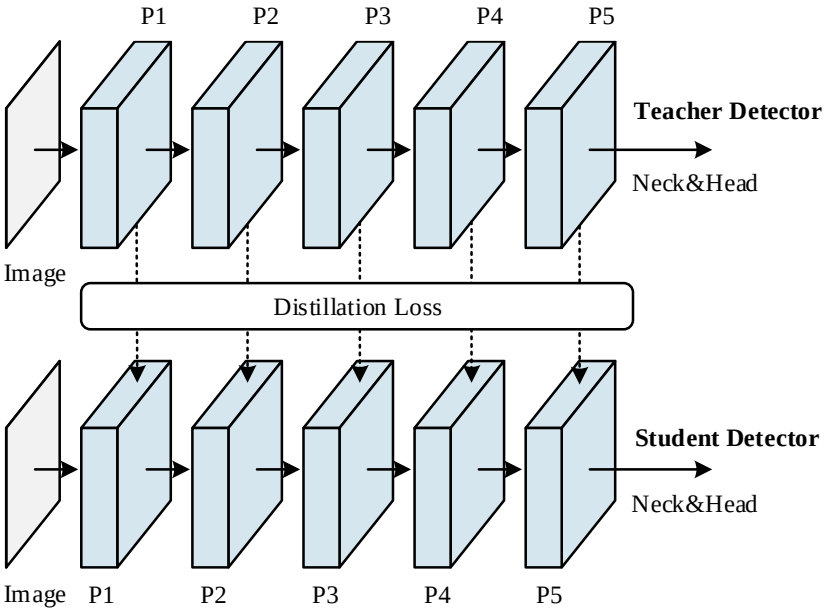


图 2.15 知识蒸馏示意图

在目标检测任务中应用知识蒸馏时，较为通用的做法是通过如图 2.15 所示的方式，使用 L2 范数[56]或 KL 散度[57]等蒸馏损失（Distillation Loss），让学生网络模仿教师网络的多尺度特征特性。通常情况下，性能越好的教师网络蒸馏得到的学生网络性能也越好。此外，教师网络可以不是单独存在的，能通过与学生网络共享部分或全部网络结构的方式进行高效蒸馏。

2.4.3 低精度量化

目前主流的深度学习框架如 PyTorch 和 TensorFlow 等，使用的默认数据类型均为 32 位浮点数。通常网络在单次推理过程中要进行上亿次浮点数计算，对设备的算力和存储带宽都有较高要求。对于算力与存储资源受限的嵌入式设备而言，使用 32 位浮点数计算的方式非常低效。通过低精度量化可将网络的参数和激活值替换为低带宽、定点的整数，从而加速网络计算，并降低所需的存储带宽，达到显著提升模型推理效率的目的。尽管低精度量化会不可避免地降低模型精度，但结合专用的计算库或硬件加速器，可成倍地加速模型推理。目前，常见的低精度量化类型包括 Int8、Int4 和二值。其中，工业界主要以 Int8 量化为主，代表包括谷歌的 TensorFlow Lite 量化方案[59]和 NVIDIA 的 TensorRT 量化方案等。

2.5 本章小结

本章对路侧感知系统的架构及其相关技术进行阐述，首先分析了路侧感知系统架构组成，以及论文研究的关键问题。然后，概括了视觉目标检测算法体系，包括目标检测的框架、网络结构组成、数据增广、样本匹配、损失函数和非极大值抑制。接着，分析了当前主流的目标检测算法 YOLOX，包括网络结构、模型扩展策略、数据增广、样本匹配和分类与回归损失函数。最后，概括了主流的模型压缩算法，包括结构化剪枝、知识蒸馏和低精度量化。本章对论文涉及到的背景知识进行较为全面的表述，为后续章节奠定基础。

第 3 章 路侧视觉目标检测模型的优化设计

路侧场景具有较高的复杂度，使用轻量化目标检测模型难以对车路协同系统中的车辆行人目标进行准确感知。尽管通用目标检测模型具备较好的检测精度，但仍需结合路侧场景特点进行算法适配，以提升模型在路侧场景中的检测性能。因此，进行针对路侧场景的目标检测模型优化设计的研究是具有实用价值的。本章首先分析路侧场景特点，然后概括适用于边缘计算设备的高效网络设计方法，并提出论文在 YOLOX 目标检测算法的基础上设计的模型优化方法。最后，在 BDD100K 数据集上开展具体的实验，以验证所提方法的有效性。

3.1 路侧场景特点分析

在路侧场景中，实时交通状态通过架设在道路上的毫米波雷达和高清摄像头等感知设备获取。相较于车载视角，路侧视角能观察到的范围更广，且道路背景也会随着道路类型、当地气候和光照条件等的不同而产生巨大变化。因此，在路侧场景中实现准确感知具有较高的难度。

3.1.1 车辆行人目标特点

(a)

(b)

(c) (d)

图 3.1 路侧场景中的车辆行人目标截图

典型的路侧场景如图 3.1 所示，图 3.1 (a)-(d)是同一路段不同路口或方向的交通实况截图。从图中可以看出，周围的树木等遮挡物较多，道路背景较为复杂，且车辆具有多个类别，其车身尺度也会在靠近或远离摄像头的过程中发生较大变化。此外，图像中的车辆小目标较多，有时车辆之间距离较近。当处于上下班高峰时，车辆行人的分布将更为密集。综上所述，路侧场景中车辆行人目标的特点包括：背景复杂、车辆类别较多、物体尺度多变、密集分布和小目标众多等。

3.1.2 路侧场景对算法的精度需求

在以毫米波雷达和高清摄像头作为感知设备的路侧融合感知系统中，视觉目标检测算法用于分析高清摄像头采集的实时视频流数据，对视频帧中的潜在物体进行定位和分类。针对路侧场景特点，目标检测算法需要具备如下特性：

1. 边框定位的准确度高。在路侧感知系统中，边框定位的准确度直接决定后续的物体测距精度。因此，需要目标检测算法具备好的边框回归性能，以应对路侧场景中物体尺度多变、密集分布的情况。

2. 分类能力强。路侧场景的背景复杂，且待检测物体的类别较多，需要目标检测算法具备好的分类性能，以确保车路协同应用的可靠性。

3. 具备较好的小目标与密集目标检测能力。在路侧场景中，小目标和密集目标较多，需要目标检测算法具备对其较好的检测性能，以扩大路侧感知的范围。

3.2 面向边缘计算设备的高效网络设计方法

边缘计算设备使用的计算单元通常包括中央处理单元（Central Processing Unit，CPU）和图形处理单元（Graphics Processing Unit，GPU）等。在设计网络时，应根据计算设备的特点对网络结构进行调整，以尽可能提升网络推理效率。

3.2.1 面向 CPU 的网络设计方法

边缘计算设备通常不单独采用 CPU 作为计算单元，但当应用场景较为简单时，可使用 CPU 承担全部计算任务以节省成本。CPU 的特点在于，单核心性能较强，适用于需要复杂逻辑控制的场景，但并行计算的能力较弱。

在设计面向 CPU 的高效目标检测网络时，主干网络应采用轻量化网络，特征融合网络可省略或使用经过轻量化设计的网络。此外，在网络结构的设计中，应使用深度可分离卷积（Depth-wise Separable Convolution）替代常规大核卷积，并适当减少注意力模块（Attention module）的使用，以及减少使用 Slice 切片、逐元素相加（Element-wise Add）、Split 分裂和 Concat 拼接等操作[81]。

3.2.2 面向 GPU 的网络设计方法

相较于 CPU，GPU 的单核心性能较弱，且难以实现复杂的逻辑控制。但 GPU 具备很强的并行计算能力，可同时调用成百上千的核心执行同一项简单的计算任务，此特性非常适合对矩阵乘法进行计算加速，而矩阵乘法正是深度学习算法的基础。因此，GPU 是最为常见的深度学习算法加速器。

在设计面向 GPU 的高效目标检测网络时，可参考的设计方法包括如下：

1. 逐点卷积（Pointwise Convolution）的输入与输出特征通道数相等时，内存访问的代价最小。在保持 1×1 卷积的理论计算量大致相同的情况下，输入与输出通道数相等时计算效率最高。

2. 组卷积的分组数不宜过多。组卷积能通过分组降低卷积的理论计算量，但过多的分组数会增加内存访问的代价，使得卷积计算效率降低。

3. 降低网络碎片化程度。网络的多分支结构有利于提升网络性能，但会增加网络的碎片化程度，导致网络计算的并行度降低。

4. 减少逐元素操作的使用。逐元素操作例如 Add 和 ReLU 等，尽管其理论计算量较小，但会明显增加内存访问的代价，从而降低网络的推理效率[82]。

3.3 基于 YOLOX 的目标检测模型优化设计

路侧感知要求视觉目标检测模型在具备较高实时性的同时，能准确对交通对象进行检测识别。目前，现有的轻量化目标检测算法在精度方面难以达到要求，通用目标检测算法则大多能满足精度要求。在通用目标检测算法中，主要分为 One-stage 和 Two-stage 两大类。通常 One-stage 算法的检测速度优于 Two-stage 算法。因此，论文考虑采用通用 One-stage 算法用于路侧感知。在现有的通用 One-stage 算法中，YOLOX 是平衡检测速度与精度最为出色的算法之一，且其有 Small、Middle、Large 和 Extreme 四个不同大小模型的版本，可根据边缘计算设备的算力，选择合适的模型用于路侧部署。但原版 YOLOX 对小目标和密集目标的检测性能相对较差，需要对其进行针对路侧场景特点的算法优化设计，从而在路侧场景中获得更好的检测效果。论文根据分析的路侧场景特点，在 YOLOX 的基础上，引入高效的 RepVGG[83]卷积结构和轻量化的 Coordinate[84]注意力机制，对 YOLOX 网络结构进行优化，并结合 VariFocal[85]分类损失函数和 Alpha-IoU[86]边框回归损失函数优化 YOLOX 的损失函数，以在提高模型推理效率的同时，提升模型的分类与边框定位能力，以及对小目标和密集目标的检测性能。

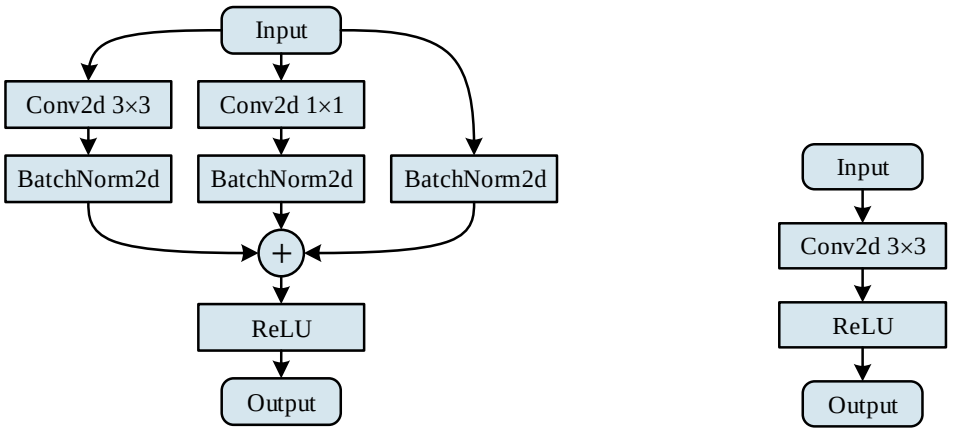
3.3.1 RepVGG 重参数化网络

RepVGG 是在 VGGNet 的基础上，结合 ResNet 残差结构思想得到的一种高效分类网络，其在训练阶段中引入残差分支，以提升网络的表示能力。在测试阶段中，其通过重参数化技术（Re-parameterization Technique），将具有残差分支的卷积块等价转换为单个 3×3 卷积层，从而加速网络的推理过程。从网络实现方式分析，RepVGG 的核心在于 RepVGG 卷积块和重参数化操作两个部分。

1. RepVGG 卷积块

RepVGG 网络主要由若干 RepVGG 卷积块堆叠而成，其卷积块网络结构如图 3.2 所示。图 3.2 (a)是训练阶段的 RepVGG 卷积块网络结构，其包括三个分支。第一个分支由 3×3 卷积层和 BN 层组成，第二个分支由 1×1 卷积层和 BN 层构成，第

三个分支则为由 BN 层组成的跳跃分支。输入特征经过三个分支并行处理后，将得到的各分支特征进行逐元素相加，再执行 ReLU 激活得到输出特征。图 3.2 (b)是 RepVGG 卷积块经过重参数化操作得到的等效网络结构，其仅由 3×3 卷积层和 ReLU 激活函数构成。相较于训练阶段，卷积块在测试阶段的结构更为高效，且 3×3 卷积计算通常受优化程度高，因此在模型部署时能进一步提升推理效率。



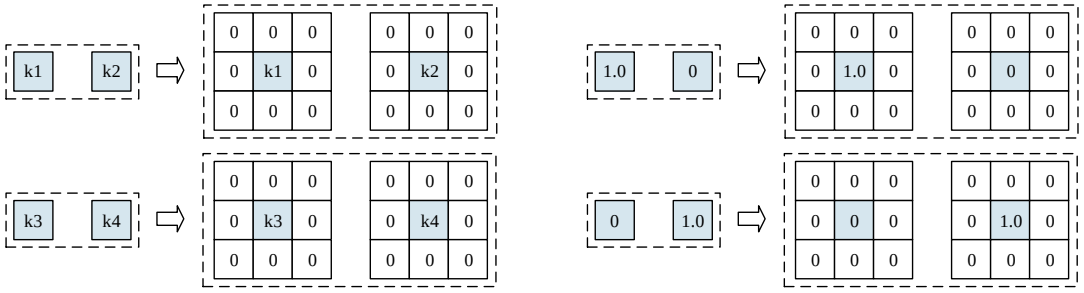
(a) 训练阶段　　(b) 测试阶段

图 3.2 RepVGG 卷积块网络结构

2. 重参数化操作

重参数化操作用于融合 RepVGG 卷积块的残差分支，得到单个 3×3 卷积层。在融合过程中，首先将卷积块中第二个分支的 1×1 卷积层通过零填充的方式扩展为 3×3 卷积层，并在第三个分支的 BN 层前面，构造出等价的 3×3 卷积层，使得三个分支有相同的网络结构，其中每个分支均由 3×3 卷积层和 BN 层构成。然后，融合每个分支中的 3×3 卷积层和 BN 层，得到单个 3×3 卷积层。最后，将三个分支的 3×3 卷积层进行合并，得到最终的单个 3×3 卷积层。

(1) 构造等价 3×3 卷积层



(a) 1×1 卷积分支构造过程　　(b) 跳跃分支构造过程

图 3.3 等价 3×3 卷积层构造过程

构造等价 3×3 卷积层的过程如图 3.3 所示，假设卷积层的输入和输出特征通道数均为 2。图 3.3 (a)为 1×1 卷积分支的构造过程，其直接在 1×1 卷积核的外围填充 0 得到等效的 3×3 卷积核。图 3.3 (b)是跳跃分支的构造过程，其先构造出等效的 1×1 卷积核，再进行零填充得到 3×3 卷积核。使用构造的等效 3×3 卷积层处理输入特征，将得到与原网络相同的输出特征。

(2) 融合卷积层与 BN 层

卷积层和 BN 层可通过算子融合的方式得到单个卷积层。假设输入特征为X，无偏置项的卷积层参数为W，BN 层参数为γ和β，输出特征为Y，则输入特征经过卷积和批归一化计算的表达式如下：

$$Y_i = \gamma_i \frac{\bar{X}_i - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} + \beta_i \tag{3.1}$$

其中：

$$\bar{X} = X * W_i \tag{3.2}$$

式中，i ——通道索引

μ, σ^2——逐通道统计的输入特征均值和方差

ϵ ——防止除零的小数

$*$ ——卷积操作符

对上式进行简单拆分组合后，可得其等价表达式如下：

$$Y_i = X * \frac{\gamma_i W_i}{\sqrt{\sigma_i^2 + \epsilon}} - \frac{\gamma_i \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} + \beta_i \tag{3.3}$$

将上式视为融合后的卷积表达式，则得到融合的卷积层参数W^f和偏置项b^f如下：

$$W_i^f = \frac{\gamma_i W_i}{\sqrt{\sigma_i^2 + \epsilon}} \tag{3.4}$$

$$b_i^f = -\frac{\gamma_i \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} + \beta_i \tag{3.5}$$

(3) 合并卷积分支

在融合 RepVGG 卷积块中三个分支的卷积层与 BN 层后，接着可对各分支的卷积层进行合并。假设三个分支卷积层的参数分别为W_1^f、W_2^f和W_3^f，对应偏置项分别为b_1^f、b_2^f和b_3^f，则合并后得到的卷积层参数$\overline{W}$和偏置项$\bar{b}$如下：

$$\overline{W} = W_1^f + W_2^f + W_3^f \tag{3.6}$$

$$\bar{b} = b_1^f + b_2^f + b_3^f \tag{3.7}$$

3.3.2 Coordinate 注意力机制

在 MobileNet[64]等工作中，通过实验证明了 SE 注意力机制能有效提升模型的性能。受到 SE 注意力机制的启发，Coordinate 注意力机制引入特征的空间位置信息，进一步增强了网络的表示能力。在 SE 特征通道加权的基础上，Coordinate 使用高效的空间位置信息嵌入和注意力热图生成方法，捕捉特征沿不同空间方向的长期依赖关系和精确的位置信息，有助于网络更准确地定位感兴趣目标。在仅增加少量计算量的情况下，Coordinate 能取得显著优于 SE 的性能。

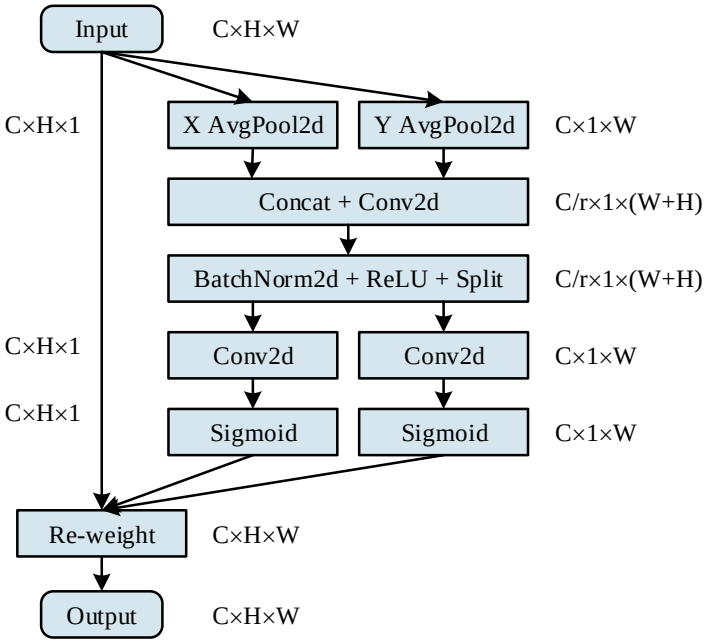


图 3.4 Coordinate 注意力机制网络结构

Coordinate 注意力机制的网络结构如图 3.4 所示，其计算过程可分为四步。假设输入特征的尺寸为 C×H×W，第一步对输入特征分别进行沿 X 轴和 Y 轴方向的均值池化，得到尺寸为 C×H×1 和 C×1×W 的特征，并将 X 轴方向所得特征的尺寸重排列为 C×1×H。第二步首先拼接 X 轴与 Y 轴方向所得特征，然后用卷积层、BN 层和 ReLU 激活函数对其进行处理，得到尺寸为 C/r×1×(W+H)的特征，其中 r 为通道缩减系数，通常取 8。第三步先执行 Split 分裂操作，得到尺寸分别为 C×1×H 和 C×1×W 的特征，并将前者的特征尺寸重排列为 C×H×1，再分别用两组卷积层和 Sigmoid 激

活函数处理特征，得到用于通道和空间位置加权的权重。第四步用所得权重对输入特征进行加权，最终得到维度为 C×H×W 的输出特征。

3.3.3 VariFocal 分类损失函数

Focal Loss 通过对训练样本进行加权的方式，有效缓解了训练 One-stage 网络时出现的正负样本比例失衡以及大量简单样本主导训练的问题，提升了 One-stage 网络的检测性能。Generalized Focal Loss 则使用预测边框与标注边框的 IoU 值作为分类软标签，在省去质量预测分支的同时，增强了 Anchor-free 算法中分类与回归分支预测的一致性。在上述方法的基础上，VariFocal 损失进一步对正负样本的加权方法进行改进，从而提升密集预测网络的性能。

VariFocal 损失的表达式如下：

$$L=\begin{cases}-q(q\log(p)+(1-q)\log(1-p)), & q>0\\ -\alpha p^{\gamma}\log(1-p), & q=0\end{cases} \tag{3.8}$$

式中，q ——IoU 类别软件标签，正样本大于 0，负样本等于 0

p ——预测值

α,γ——Focal Loss 的超参数

VariFocal 损失的特点在于，Focal Loss 中的加权项仅对负样本有效，以使样本数量相对较少的正样本能保留训练信息。此外，VariFocal 损失使用预测边框与标注边框的 IoU 值作为分类软标签，并用 IoU 软标签对正样本进行加权，使得 IoU 预测值大的正样本损失值更大，从而让网络在训练过程中能更加关注高质量的预测。通过更合理的样本加权方式，VariFocal 损失能增强网络的分类性能。

3.3.4 Alpha-IoU 回归损失函数

在目标检测任务中，IoU 是度量边框回归精度的重要指标，且其具备对边框的尺度不变性，因此适合作为边框回归的损失函数。然而，当预测边框与标注边框不存在相交时，其 IoU 值恒为零，会出现梯度消失的问题。针对此问题，GIoU、DIoU 和 CIoU 等 IoU 损失的改进方法通过引入 IoU 惩罚项，有效提升了网络训练的稳定性和收敛速度。Alpha-IoU 损失则是对 IoU 损失的进一步推广，其引入了 IoU 的次幂项，以及 IoU 次幂项附加的惩罚项，能概括目前现有的 IoU 损失。

Alpha-IoU 损失的表达式如下：

$$L = 1.0 - IoU^{\alpha_1} + p^{\alpha_2}(B, B^{gt}) \tag{3.9}$$

式中，α_1 ——IoU 的次幂超参数，取值大于 0

$p(B, B^{gt})$——附加惩罚项

α_2 ——附加惩罚项的次幂超参数，取值大于 0

当$\alpha_1 = 1$且无附加惩罚项时，Alpha-IoU 损失等价于 IoU 损失。通过附加不同的惩罚项，Alpha-IoU 损失可对现有 IoU 损失进行推广。

例如，对 GIoU 损失推广的 Alpha-GIoU 表达式如下：

$$L = 1.0 - IoU^{\alpha_1} + \left(\frac{C - (B \cup B^{gt})}{C}\right)^{\alpha_2} \tag{3.10}$$

式中，$(C - (B \cup B^{gt}))/C$——GIoU 损失的惩罚项

对 DIoU 损失推广的 Alpha-DIoU 表达式如下：

$$L = 1.0 - IoU^{\alpha_1} + \frac{\rho^{2\alpha_2}(B, B^{gt})}{c^{2\alpha_2}} \tag{3.11}$$

式中，$\rho^2(B, B^{gt})/c^2$——DIoU 损失的惩罚项

对 CIoU 损失推广的 Alpha-CIoU 表达式则如下：

$$L = 1.0 - IoU^{\alpha_1} + \frac{\rho^{2\alpha_2}(B, B^{gt})}{c^{2\alpha_2}} + (\alpha v)^{\alpha_2} \tag{3.12}$$

式中，$\rho^2(B, B^{gt})/c^2 + \alpha v$——CIoU 损失的惩罚项

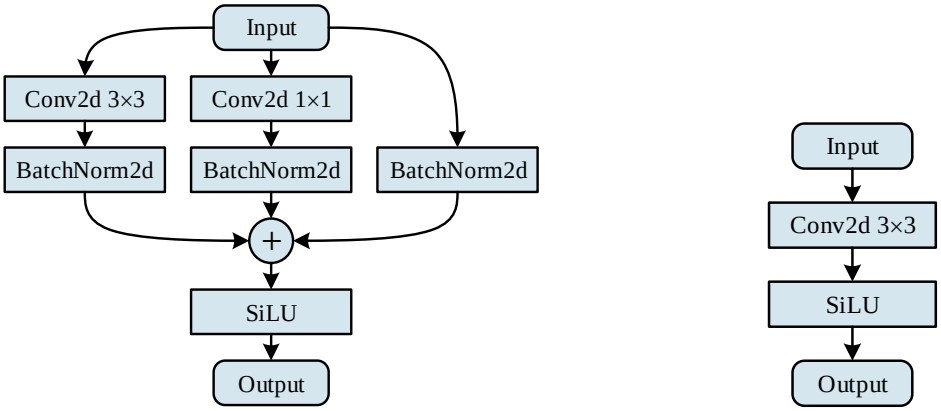
利用对 IoU 的幂变换，Alpha-IoU 损失可实现对网络训练损失的自适应加权，从而加速网络对高 IoU 目标的学习，并为优化多层次的目标提供更多空间。当设定α_1与α_2大于 1 时，在训练过程中网络会优先学习简单样本，且其学习速度会不断提升，直至 IoU 值等于 1。困难样本则在随后被逐渐学习，其学习速度也随着 IoU 值的提升而不断加快。Alpha-IoU 损失使得模型在实现不同水平的边框回归准确度方面有更高的灵活性，能取得比现有 IoU 损失更好的性能。此外，Alpha-IoU 损失对小数据集的噪声有更好的鲁棒性，且对小网络更为适用。

3.3.5 网络结构优化设计

在 YOLOX 目标检测算法的基础上，论文引入高效的 RepVGG 卷积块和 Coordinate 注意力机制，对 YOLOX 的网络结构进行优化。其中，RepVGG 卷积块用于提升模型的推理效率。Coordinate 注意力机制则用于增强网络的表示能力，使

网络能更好地处理多尺度目标，从而提升模型在路侧场景中的检测性能。具体的网络结构设计包括以下两个部分。

1. 替换 CSPDarkNet 和 PAFPN 中的 3×3 卷积块



(a) 训练阶段 (b) 测试阶段

图 3.5 改进的 RepVGG 卷积块网络结构

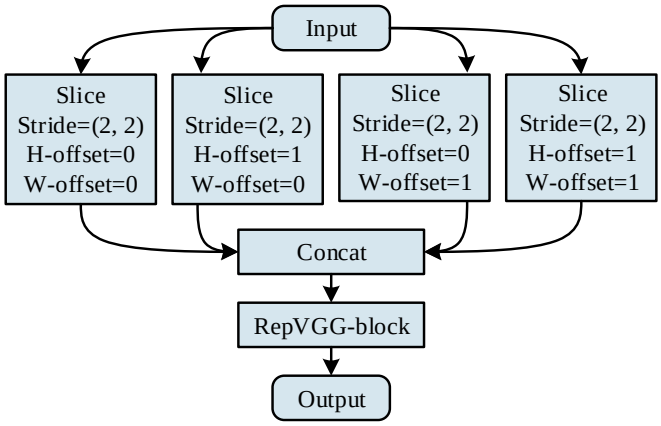
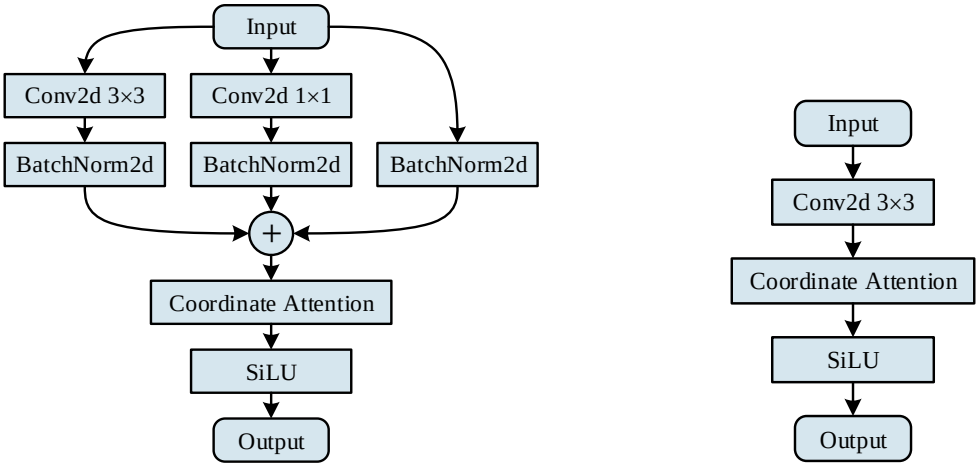


图 3.6 改进的 Focus 网络结构

YOLOX 网络中包含若干 3×3 卷积块，通过使用 RepVGG 卷积块对其进行替换，可在不增加推理计算量的同时，提升网络的表示能力。用于替换的 RepVGG 卷积块网络结构如图 3.5 所示。为保持网络激活函数的一致性，用 SiLU 激活函数替换原卷积块中的 ReLU 激活函数。对于 CSPDarkNet 和 PAFPN，用 RepVGG 卷积块替换 Focus 结构中的 ConvBlock，以及网络中用于下采样的 ConvBlock。当 ConvBlock 的步长为 2 或输入与输出通道数不相等时，替换的 RepVGG 卷积块中不包含跳跃分支。改进的 Focus 网络结构如图 3.6 所示。

2. 改进 CSPDarkNet 和 PAFPN 中的 Bottleneck 结构



(a) 训练阶段 (b) 测试阶段

图 3.7 论文设计的 RepVGG-CA 卷积块网络结构

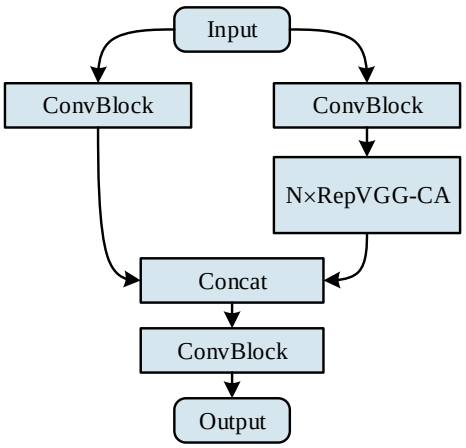


图 3.8 改进的 CSPBottleneck 网络结构

相较于 Bottleneck 结构，RepVGG 卷积块的推理效率更高。根据论文分析的高效网络设计方法，使用 RepVGG 卷积块替换 Bottleneck 结构，能降低网络的碎片化程度，并同时减少逐元素操作的使用。但直接用 RepVGG 卷积块替换网络中的 Bottleneck 结构，会导致模型性能略微降低。因此，论文使用结合 RepVGG 卷积块和 Coordinate 注意力机制得到的 RepVGG-CA 卷积块替换 Bottleneck，可在提升推理效率的同时，进一步增强模型的检测性能。RepVGG-CA 卷积块的网络结构如图 3.7 所示，替换 Bottleneck 结构得到的 CSPBottleneck 网络结构则如图 3.8 所示。论文使用 RepVGG-CA 卷积块替换 CSPDarkNet 中的 Bottleneck 结构，并用 RepVGG 卷积块替换 PAFPN 中 Bottleneck 结构。此外，根据研究表明，目标检测网络的特征融合

网络部分相比主干网络甚至更为重要[87]，因此将 PAFPN 中 RepVGG 卷积块的数量增加一倍，以使模型具备更好的多尺度目标处理能力。

在经过上述方法对 YOLOX 网络进行优化后，得到的 CSPDarkNet 网络结构如表 3.1 所示，PAFPN 网络结构则如图 3.9 所示。

表 3.1 改进的 CSPDarkNet 网络结构组成

输入尺寸	结构类型	RepVGG-CA 数量	残差连接	步长	输出
3×640×640	Focus	—	—	2	—
64×320×320	RepVGG-block	—	—	2	—
128×160×160	CSPBottleneck	3	True	1	—
128×160×160	RepVGG-block	—	—	2	—
256×80×80	CSPBottleneck	9	True	1	P3
256×80×80	RepVGG-block	—	—	2	—
512×40×40	CSPBottleneck	9	True	1	P4
512×40×40	RepVGG-block	—	—	2	—
1024×20×20	SPPBottleneck	—	—	1	—
1024×20×20	CSPBottleneck	3	False	1	P5

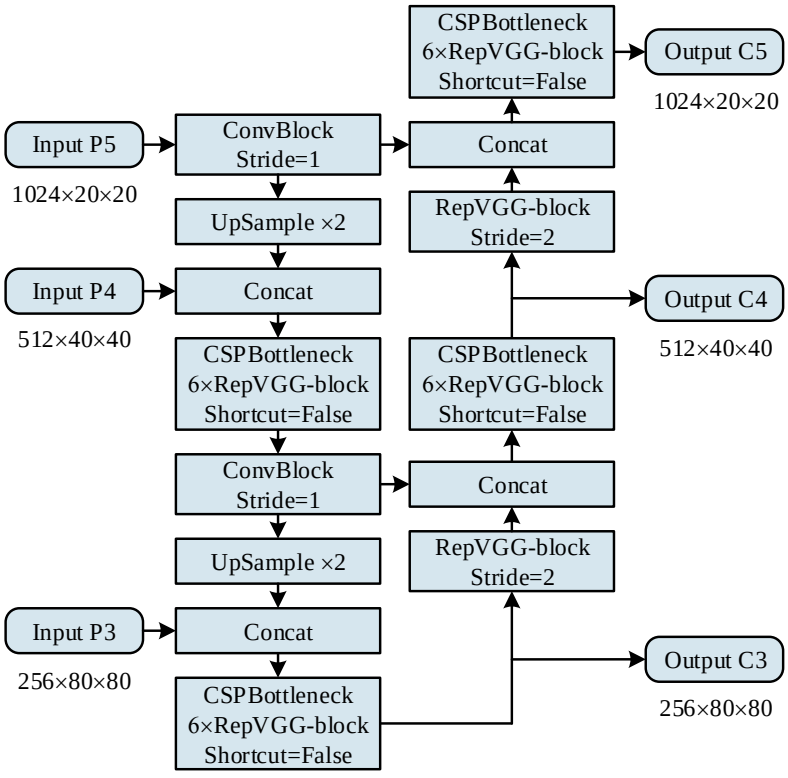


图 3.9 改进的 PAFPN 网络结构

3.3.6 损失函数优化设计

YOLOX 的检测头部网络输出包括边框回归、置信度和分类三个分支，分别用于对物体的边框位置、置信度和类别进行预测。其中，边框回归分支使用 IoU 的二次幂损失函数，置信度与分类分支则用交叉熵损失函数。本章通过引入 Alpha-IoU 损失和 VariFocal 损失对 YOLOX 使用的损失函数进行改进，目的在于提升模型在复杂场景下的目标分类和边框定位精度。具体的损失函数设计包括以下两个部分。

1. 改进边框回归分支的 IoU 损失

在路侧场景中，为确保车路协同应用的可靠性，实现对车辆行人目标的准确定位极为重要。因此，论文采用经 Alpha-IoU 损失推广得到的 Alpha-CIoU 损失替换原有的 IoU 二次幂损失，以提升模型的边框定位准确度。

优化的边框回归损失表达式如下：

$$\bar{L}_{bbox} = \frac{1}{N^{obj}} \sum_{k=0}^{M} \sum_{i=0}^{H^k} \sum_{j=0}^{W^k} 1_{kij}^{obj} \times \text{AlphaCIoU}(B_{kij}, B_{kij}^{gt}, \alpha_1, \alpha_2) \quad (3.13)$$

式中，N^{obj} ——单个批次训练数据中正样本的数量

M ——多尺度特征的数量

H, W ——单个尺度特征的高度和宽度

1^{obj} ——正样本掩码，正样本取 1，负样本则取 0

B, B^{gt}——预测边框和对应的标注边框

α_1, α_2 ——Alpha-IoU 损失的参数，默认值取$\alpha_1 = \alpha_2 = 3$

2. 改进置信度分支的交叉熵损失

相较于分类分支，置信度分支在计算损失时，会用到全部的正负样本。因此，在训练过程中使用交叉熵损失将会出现正负样本比例失衡，以及大量简单样本主导训练的问题。论文通过使用 VariFocal 损失替换交叉熵损失，可有效缓解训练 One-stage 网络的正负样本不平衡问题，以进一步增强模型的分类能力。

优化的置信度损失表达式如下：

$$\bar{L}_{conf} = \frac{1}{N^{obj}} \sum_{k=0}^{M} \sum_{i=0}^{H^k} \sum_{j=0}^{W^k} \text{VariFocal}(p_{kij}, y_{kij}, \alpha, \gamma) \quad (3.14)$$

式中，p ——预测值

y ——样本类别标签

α, γ——VariFocal 损失超参数，默认值取$\alpha = 0.75$，$\gamma = 2.0$

经过上述两部分方法优化后，最终得到的 YOLOX 损失函数如下所示：

$$L_{total} = \lambda_{bbox}\bar{L}_{bbox} + \lambda_{conf}\bar{L}_{conf} + \lambda_{cls}L_{cls} \tag{3.15}$$

式中，λ_{bbox}——边框回归损失平衡系数

λ_{conf}——置信度损失平衡系数

L_{cls} ——YOLOX 分类损失函数

λ_{cls} ——分类损失平衡系数

3.4 实验与分析

利用实验室现有的深度学习工作站环境，本节对本章设计的模型开展对比实验，以验证所提方法的有效性。实验选取的训练数据集为与路侧场景较为接近的 BDD100K[88]开放数据集，评价指标包括精度与实时性指标，对比的模型为原版 YOLOX，实验数据均来自相同的测试环境。

3.4.1 实验数据集

图 3.10 BDD100K 视频关键帧图像截图

实验使用 BDD100K 数据集训练模型。BDD100K 是由伯克利大学 AI 实验室于 2018 年公开的大规模、具有多样化场景的开放数据集，其包括不同城市、天气、时间段和交通场景的 10 万段高清视频，以及由手机记录的车辆行驶位置和姿态信息。为方便 CV 研究者开展工作，BDD100K 对每段视频的关键帧进行提取和标注，得到

10 万张图像以及包括图像标记、道路对象边框、可行驶区域、车道线标记和全帧实例分割五个类别的图像标注。截取的部分视频关键帧图像如图 3.10 所示。

表 3.2 BDD100K 图像标注的不同类别对象数目统计

类别名称	标注数量	类别名称	标注数量
Bus	16505	Truck	42963
Traffic Light	265906	Motorcycle	4296
Traffic Sign	343777	Car	1021857
Pedestrian	129262	Train	179
Bicycle	10229	Rider	6461

在实验过程中，使用的图像标注为 BDD100K 的道路对象边框部分，其将全部图像划分为训练集、验证集和测试集三部分。其中，训练集与测试集分别包含提供标注的 7 万和 1 万张图像，测试集则包含未提供标注的 2 万张图像。图像标注包括车辆、行人等 10 个类别，平均每张图像包含 9.7 个车辆标注和 1.2 个行人标注，共计约 184 万个对象标注边框。BDD100K 的不同类别对象数目统计如表 3.2 所示，其包括大量行人与多个类别车辆的对象，且对象的尺度变化较大，分布较为密集，并存在较多的物体遮挡、截断和小目标。此外，BDD100K 的背景也相当复杂，包含丰富的交通场景，与路侧场景特点相近。因此，BDD100K 适合用于测试路侧视觉目标检测模型的综合性能。

3.4.2 实验环境与评价指标

在开展实验前，需要预先搭建好实验环境，并选取用于评价模型性能的指标。实验环境的搭建包括硬件和软件两个部分，用于测试模型性能的评价指标则包括精度与实时性两个方面。

1. 实验环境搭建

表 3.3 深度学习工作站主要硬件配置

硬件类别	配置型号
CPU	Intel Core i9-10920X 12 核
GPU	2×NVIDIA RTX Titan 24GB
硬盘	1TB 固态 + 4TB 机械
内存	双通道 DDR4 64GB

实验中使用的深度学习工作站主要硬件配置如表 3.3 所示。工作站搭载英特尔酷睿 10 代 CPU 和 2 张 NVIDIA RTX Titan 24GB 显卡，还配置有 64GB 的双通道 DDR4 内存，以及 1TB 固态和 4TB 机械硬盘。结合 NVLink 桥接器，能满足一般深度学习模型的训练条件。

表 3.4 深度学习工作站主要软件配置

软件名称	版本	软件名称	版本
Ubuntu	18.04	PyTorch	1.7.1
CUDA	10.2	OpenCV	4.5.1
cuDNN	8.1.0	TensorRT	7.1.3
Python	3.8.5	DeepStream	5.1

深度学习工作站主要的软件环境配置如表 3.4 所示。工作站安装的操作系统为 Ubuntu 18.04，模型训练采用 PyTorch 深度学习框架，其底层使用 NVIDIA 的并行计算加速库和深度神经网络计算库。在训练过程中，还会用到 Python 环境下的 OpenCV 等视觉软件库处理训练图像。

2. 评价指标

评价指标用于客观反映模型在具体方面的性能。论文选取的评价指标主要关注模型的精度和实时性两个方面。在目标检测任务中，评价模型精度的指标主要包括平均查准率（Average Precision，AP）和平均查准率均值（Mean Average Precision，mAP）等。其中，AP 的计算方法主要包括 PASCAL VOC 和 COCO[89]两种。论文采用 COCO 方法，其先分别计算单个类别在不同 IoU 阈值下所得精准率-召回率（Precision-recall）曲线的 AP 值，再取各 AP 值的平均值。mAP 则是所有类别 AP 值的平均值。AP 或 mAP 值越高，表明模型的检测性能越好。

评价模型实时性的指标主要包括参数量 Params、理论计算量（Floating Point Operations，FLOPs）和每秒处理帧数（Frames Per Second，FPS）。其中，Params 指网络中参数的数量，FLOPs 用于度量模型在一次推理过程中需要执行的浮点数操作数量。通常模型的 Params 或 FLOPs 越大则实时性越低，但采用高效网络结构的模型能在 Params 和 FLOPs 较大的情况下，取得较高的推理效率。FPS 则统计模型每秒处理的图像数量，是度量模型实时性的硬性指标。

3.4.3 实验过程

实验使用 BDD100K 数据集的训练集，分别从零训练原版 YOLOX 和本章设计的模型。为确保实验的公平，训练模型时均采用默认配置。在训练本章设计的模型时，批大小（Batch Size）为 64，使用权重衰减（Weight Decay）系数为 0.0005、动量项系数为 0.9、初始学习率为 0.01 和包括 5 个 Epoch 预热（Warm-up）阶段的余弦衰减（Cosine Decay）学习率的随机梯度下降（Stochastic Gradient Descent，SGD）优化器。结合指数滑动平均（Exponential Moving Average，EMA）、Mosaic 和 Mixup 数据增广，以及多尺度训练等方法，训练模型 300 个 Epoch，并在最后 15 个 Epoch 中，关闭 Mosaic 和 Mixup 数据增广。训练结束后，使用 BDD100K 的验证集评估模型性能，模型测试的输入分辨率为 640×640。

3.4.4 结果分析

表 3.5 模型的精度评估结果

测试模型	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOX-s	28.1	48.6	27.5	9.5	34.8	54.5
本章模型	29.6	52.7	27.5	11.9	34.4	53.8

本章设计的模型与原版 YOLOX 在 BDD100K 验证集上得到的精度评估结果如表 3.5 所示。实验中选取的模型为网络深度系数为 0.33 和宽度系数为 0.50 的 Small 版本。表中AP_{50}和AP_{75}指 IoU 阈值分别为 0.50 和 0.75 时所得 AP 值，AP_S、AP_M和AP_L则指分别对小目标、中等大小目标和大目标进行评估得到的 AP 值。从表中可以看出，相较于原版 YOLOX，本章设计的模型在mAP、AP_{50}和AP_S评估指标上明显提升，表明本章设计的模型在分类与定位能力方面均有增强，同时对小目标的检测性能更好。此外，由于 BDD100K 数据集具有目标密集分布的特性，因此本章设计的模型在一定程度上也具有对密集目标更好的检测性能。

表 3.6 模型的各类别精度评估结果

测试模型	Bus	Traffic Light	Traffic Sign	Pedestrian	Bicycle	Truck	Motor-cycle	Car	Train	Rider
YOLOX-s	45.7	16.5	30.0	28.5	22.0	45.7	22.0	48.7	0	22.0
本章模型	44.8	24.4	35.8	30.0	22.2	44.1	22.8	50.4	0	21.1

本章设计的模型与原版 YOLOX 在 BDD100K 验证集上得到的各类别精度评估结果如表 3.6 所示。从表中可以看出，本章设计的模型对主要的 Pedestrian 和 Car 类别的检测精度要明显优于原版 YOLOX。但对 Bus 和 Truck 类别的检测效果相比原版 YOLOX 略差。因此可得出的结论是，本章设计的模型更适合检测小尺度的目标，此特点也符合路侧感知对模型的精度要求。

(a) YOLOX-s　　(b) 本章模型

图 3.11 模型的测试结果对比

本章设计的模型与原版 YOLOX 的测试结果对比如图 3.11 所示。从图中可以看出，相较于原版 YOLOX，本章设计的模型整体检测效果更好，且对小目标的漏检率更低，能为路侧感知提供更远距离的状态感知。

表 3.7 模型的实时性评估结果

测试模型	Params（Million）	FLOPs（Billion）	FPS（Average）
YOLOX-s	8.94	26.65	284
本章模型	9.62	27.85	304

本章设计的模型与原版 YOLOX 进行推理实时性评估的结果如表 3.7 所示。测试模型的推理批大小（Batch Size）为 64，测试的输入分辨率为 640×640，FPS 指模型平均每秒处理的图像数量。从表中可以看出，尽管本章设计的模型相比原版 YOLOX 具有更多的参数和更大的理论计算量，但得益于高效的 RepVGG 卷积块，模型的实际推理速度反而更快。

综上所述，本章设计的模型具有比原版 YOLOX 更好的分类与定位性能，且在一定程度上增强了对小目标和密集目标的检测能力。此外，本章设计的模型还具有比原版 YOLOX 更快的推理速度。通过实验证明，本章针对路侧场景提出的模型优化设计方法是有效的。

3.5 本章小结

本章对路侧场景下的视觉目标检测模型优化设计方法进行阐述，首先分析了路侧场景特点，包括车辆行人目标的特点，以及路侧场景对算法的精度需求。然后，概括了适用于边缘计算设备的高效网络设计方法。接着，提出了基于 YOLOX 的模型优化设计方法，包括对网络结构和损失函数的优化。最后，在 BDD100K 数据集上对所提方法开展实验，验证了方法的有效性。通过实验证明，本章所提方法能在提升模型推理效率的同时，使模型具有更好的分类与定位性能，且在一定程度上增强了模型对小目标和密集目标的检测能力。

第 4 章 路侧视觉目标检测模型的压缩方法设计

路侧感知对视觉目标检测算法的精度与实时性均有较高要求。在第 3 章中，论文针对路侧场景特点，完成了对 YOLOX 目标检测算法的场景适配。在保持推理效率的同时，有效提升了算法的检测精度。然而，直接部署优化的 YOLOX 模型到路侧边缘计算平台难以满足路侧感知的实时性要求，需要结合模型压缩技术进一步降低模型的计算量。本章首先分析在路侧场景中模型压缩的必要性，然后提出论文设计的结合通道剪枝与知识蒸馏算法的模型压缩方法，并概述在路侧边缘计算平台上部署模型时用到的工具，以及模型部署的过程。最后，在 BDD100K 数据集上开展具体的实验，以验证所提方法的有效性。

4.1 模型压缩的必要性

在车路协同系统中，为提升数据处理效率，并提供更快的应用响应速度，路侧感知采用边缘计算设备就近处理视频流等数据。考虑到设备成本、功耗和体积等因素，路侧边缘计算平台通常采用嵌入式设备执行计算任务，其导致直接在平台上部署通用目标检测模型的方式难以获得较好的实时性，有必要借助模型压缩方法降低模型所需的计算量，以在节省设备成本的同时，满足应用的实时性要求。因此，模型压缩在路侧视觉感知中具有重要作用。

4.1.1 边缘计算设备的成本与算力分析

在路侧感知的工业化部署中，需要用尽可能低的设备成本满足应用的算力需求。以论文使用的 NVIDIA Jetson 系列嵌入式开发套件为例，其部分型号的设备成本与算力统计数据如表 4.1 所示。从表中可以看出，算力越高的设备对应的成本也越高，且设备的性价比也随着算力的提升而逐步降低。当部署未经压缩的模型时，通常需要更高算力的设备才能满足应用的实时性要求，从而导致部署的整体成本大幅上升。借助模型压缩算法，可在模型精度无明显下滑的前提下，显著降低模型所需的计算量，从而可部署在算力适中但具有高性价比的设备上进行推理，整个应用部署的成本也因此相应降低。

表 4.1 NVIDIA Jetson 系列设备的成本与算力统计数据

设备型号	参考成本（RMB）	功耗（瓦）	峰值算力（TOPS）
Nano	900	5-10	0.47 (FP16)
TX2	2200	7.5-15	1.33 (FP16)
Xavier NX	3400	10-20	21 (Int8)
AGX Xavier	5500	10-30	32 (Int8)
Orin NX	9000	10-25	100 (Int8)
AGX Orin	17000	15-50	275 (Int8)

4.1.2 路侧场景对算法的实时性需求

在以高清摄像头和毫米波雷达为感知设备的路侧融合感知系统中，考虑到摄像头与毫米波雷达之间数据同步的问题，应使视觉模型处理视频帧的频率与毫米波雷达检测的频率相近，而后者的检测频率通常在 20 赫兹左右。此外，路侧边缘计算平台也需要能同时处理多路实时视频流。例如，典型的十字路口会架设 4-8 路的感知设备对各个方向的车辆行人进行实时检测。因此，针对路侧场景特点，视觉目标检测算法需要具备如下特性：

1. 在保持检测精度的前提下，单路摄像头的视频帧从采集到模型处理完毕所消耗的时间应在 50ms 以内，保持检测频率在 20 赫兹以上，且最好能达到甚至超过 30 赫兹，以确保交通状态感知的时效性。

2. 模型应具备从单路实时检测扩展到多路实时检测的能力。例如能在满足检测频率的同时，处理 4 路或 8 路的实时视频流。

4.2 结合通道剪枝与知识蒸馏的模型压缩方法设计

通用目标检测模型的计算量较大，将其直接部署在路侧边缘计算设备上难以满足路侧感知的实时性要求。因此，需要利用模型压缩方法降低模型所需的计算量，以提升模型的推理速度。在常见的模型压缩方法中，通道剪枝能裁剪网络中部分相对不重要的通道参数，知识蒸馏则能利用教师大网络得到性能出色的学生小网络。论文在第 3 章所得模型的基础上，设计了一种结合 ResRep[90]通道剪枝与基于注意力机制知识蒸馏[56]的目标检测模型压缩方法，用于在模型精度无明显下滑的前提下，显著提升模型在路侧感知中的推理速度。其中，ResRep 通道剪枝用

于去除网络中一定比例的通道参数，使得网络结构更为紧凑。知识蒸馏算法则用于进一步恢复网络因通道剪枝而损失的部分精度。

4.2.1 ResRep 通道剪枝

ResRep 是一种基于卷积重参数化（Re-parameterization）和梯度重置（Gradient Resetting）的通道剪枝算法，其动机在于将 CNN 类比为具有记忆和遗忘两个部分的动物大脑，其中网络的训练类比于记忆，网络的剪枝则类比于遗忘。在现有的大多数剪枝方法中，记忆和遗忘两个部分是存在耦合的。例如对卷积层的参数施加 L1 范数惩罚或 LASSO 回归等，会使网络在学习记忆与遗忘的过程中陷入两难的境地。RepRep 通过重参数化操作，将 CNN 中的单个卷积层等价拆分为两个部分，其中一部分用于记忆，另一部分则用于遗忘，从而实现对记忆和遗忘过程的解耦，并在图像分类任务中取得了显著的通道剪枝效果。

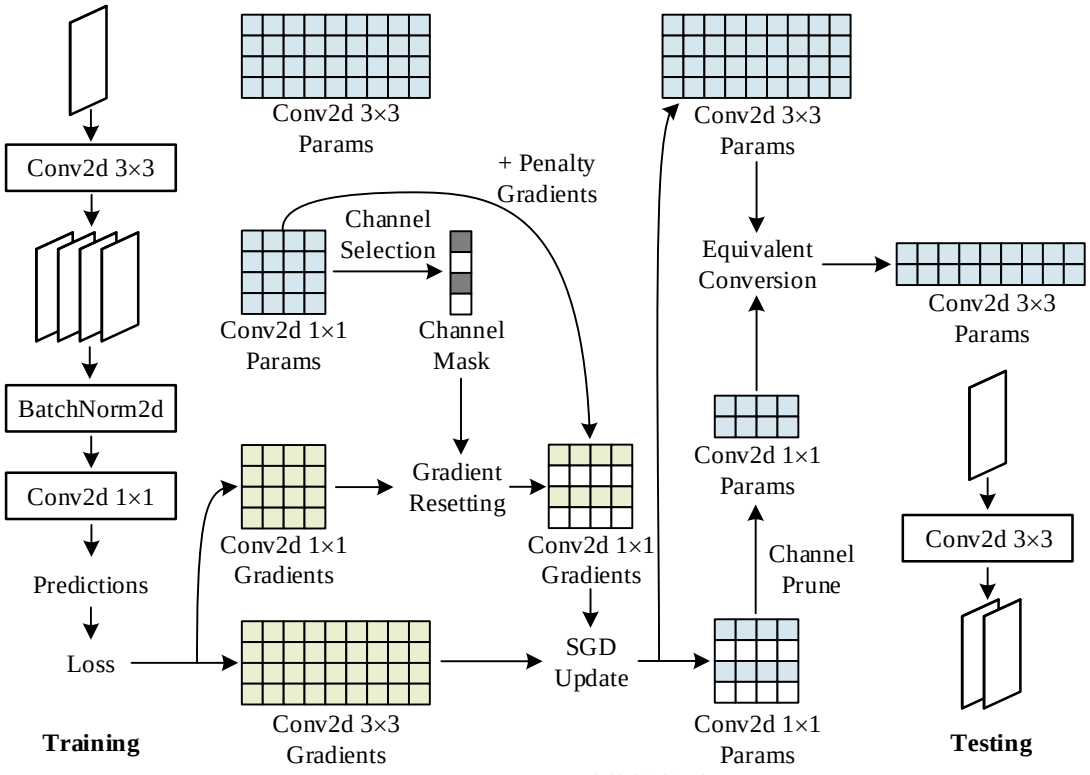


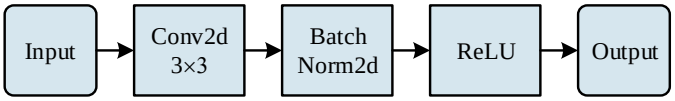
图 4.1 ResRep 通道剪枝算法

使用 ResRep 对典型的 3×3 卷积块进行通道剪枝的流程如图 4.1 所示，其包括训练和测试两个阶段。假设卷积块的输入通道数为 1，输出通道数为 4。在训练阶段中，首先用已训练的参数初始化待剪枝的卷积块，然后在卷积块中 BN 层的后面构造等价 1×1 卷积层，并用较小的学习率微调网络。在每次迭代更新的过程中，

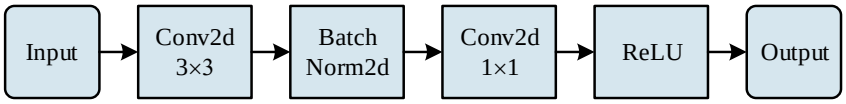
首先根据网络预测计算损失值，并通过反向传播得到卷积块中各参数对应的梯度。然后，对卷积块中 1×1 卷积层的参数逐步进行通道筛选（Channel Selection）操作，获得其对应的通道掩码（Channel Mask）用于梯度重置（Gradient Resetting），并在重置后的参数梯度中添加稀疏惩罚梯度。最后，通过随机梯度下降 SGD 优化器更新卷积块中的各参数。经过若干次迭代后，卷积块中 1×1 卷积层的部分通道参数将无限趋近于零，可在微调结束后被安全去除。在测试阶段中，首先融合卷积块中的 3×3 卷积层和 BN 层，然后对卷积块中的 1×1 卷积层参数进行通道裁剪（Channel Prune），并与融合后的3×3卷积层进行等价转换（Equivalent Conversion），得到剪枝的 3×3 卷积层用于测试。

1. 构造等价 1×1 卷积层

ResRep 构造等价 1×1 卷积层的方式与 RepVGG 相同，均使用单位矩阵作为卷积层的参数。插入等价 1×1 卷积层的 3×3 卷积块结构如图 4.2 所示，其输出与原卷积块的输出相同。



(a) 插入等价 1×1 卷积层前



(b) 插入等价 1×1 卷积层后

图 4.2 插入等价 1×1 卷积层的 3×3 卷积块结构

2. Group LASSO 稀疏惩罚

通过对网络中构造的 1×1 卷积层参数施加 Group LASSO 稀疏惩罚，ResRep 使其参数具有通道层面的稀疏性，从而自适应鉴别出网络中的重要通道，并为通道筛选过程提供依据。假设原网络的损失函数为L_{total}，网络中有n个构造的 1×1 卷积层，则引入 Group LASSO 惩罚项的损失函数如下：

$$\bar{L}_{total} = L_{total} + \lambda \sum_{i=0}^{n} P_{GroupLasso}(K_i) \tag{4.1}$$

式中，λ ——惩罚项系数

$P_{GroupLasso}$——Group LASSO 惩罚项

K ——网络中构造的 1×1 卷积层参数

3. 通道筛选过程

在 ResRep 通道剪枝中，通道筛选的目的是得到用于梯度重置的通道掩码。在剪枝训练开始时，网络中所有构造的 1×1 卷积层的通道均标记为有效，其通道对应的初始掩码全为 1。随着训练的进行，ResRep 以固定的操作频率，例如每 200 次迭代，逐步更新通道掩码，直至被标记为无效的通道数达到预设值。每次更新掩码时，ResRep 首先度量网络中所有构造的 1×1 卷积层的通道重要性，然后将其中最不重要的几个有效通道对应的掩码设为 0，使得该部分通道被标记为无效。ResRep 度量通道重要性的表达式如下：

$$t_j^{(i)} = ||K_j^{(i)}||_2, \forall 1 \le i \le n, \forall 1 \le j \le D^{(i)} \tag{4.2}$$

式中，t　——通道重要性评估系数，值越大表明对应通道越重要

$||\cdot||_2$——L2 范数操作符

D　——构造的 1×1 卷积层的通道数

4. 梯度重置过程

借助 Group LASSO 稀疏惩罚，ResRep 可自适应鉴别网络通道的重要性，从而筛选出部分待裁剪的通道。但 Group LASSO 惩罚的力度有限，难以使被标记为无效的通道参数不影响网络的预测。因此，需要通过梯度重置进一步实现对通道参数的裁剪。梯度重置的思想非常简单，其通过修改参数更新的方式达到目的。假设经过重置后的参数梯度为$G(K)$，则梯度重置的表达式如下：

$$G(K) = \frac{\partial L_{total}}{\partial K} m + \lambda G_{GroupLasso}(K) \tag{4.3}$$

式中，$\partial L_{total}/\partial K$　——反向传播所得参数梯度

m　——参数的通道掩码

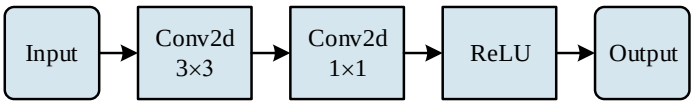
$G_{GroupLasso}(K)$——参数K的 Group LASSO 惩罚项梯度

从式中可以看出，梯度重置首先将反向传播得到的参数梯度中对应通道掩码为 0 的梯度部分置零，然后向经过置零的梯度添加 Group LASSO 惩罚梯度。使用梯度$G(K)$更新参数，会导致参数中被标记为无效的通道在迭代更新中不断趋近于零，直至整个通道的参数完全不影响卷积的输出。

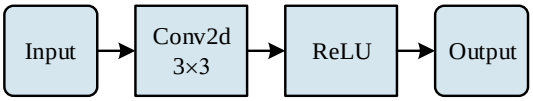
5. 通道裁剪与等价转换过程

剪枝训练结束后，网络中构造的 1×1 卷积层的部分通道参数将无限趋近于零。此时，可对网络参数进行通道裁剪和等价转换，得到裁剪后的紧凑网络用于推理。

以典型的 3×3 卷积块为例，首先将卷积块中的 3×3 卷积层与 BN 层进行融合，得到单个 3×3 卷积层，融合后的卷积块结构如图 4.3 (a)所示。然后，对构造的 1×1 卷积块进行通道裁剪，并与融合后的 3×3 卷积层进行等价转换，得到剪枝的 3×3 卷积层，其卷积块结构如图 4.3 (b)所示。



(a) 融合卷积与 BN 层的卷积块结构



(b) 等价转换的卷积块结构

图 4.3 算子融合与等价转换的 3×3 卷积块结构

假设融合后的 3×3 卷积层参数为W^f，偏置项为b^f。对于通道裁剪过程，首先根据预设的通道重要性阈值t_c，例如取$t_c = 0.00001$，得到用于参数裁剪的通道掩码。当t中元素大于等于t_c时，对应通道的掩码为 1，反之则为 0。然后，用所得通道掩码对参数K进行通道裁剪，得到裁剪后的参数$\overline{K}$。$\overline{K}$由K中对应通道掩码为 1 的参数部分组成。对于等价转换过程，假设转换后得到的 3×3 卷积层参数为$\overline{W}$，偏置项为$\bar{b}$，则等价转换的表达式如下：

$$\overline{W} = T(T(W^f) * \overline{K}) \tag{4.4}$$

$$\bar{b} = b^f * \overline{K} \tag{4.5}$$

式中，T——转置操作符，用于转置卷积层参数的输入与输出维度

$*$——卷积操作符

4.2.2 基于注意力机制的知识蒸馏

知识蒸馏（Knowledge Distillation）在图像分类任务中有较多显著的成果，但在目标检测任务中相关的研究工作还较少。由于目标检测任务本身的难度相比图像分类更高，且其网络结构也与图像分类有部分差异，导致现有的大多针对图像分类的知识蒸馏算法难以直接在目标检测中表现良好。目前，针对目标检测的知识蒸馏通常采用让学生网络模仿教师网络的多尺度特征特性的方法，其主要存在两个问题，一是目标检测中前景与背景特征存在不平衡的情况，学生网络难以在蒸馏的过程中捕捉到有用的前景信息。二是缺乏对特征元素之间关联信息的提取。

通过在知识蒸馏中引入注意力机制，可有效缓解上述问题。一方面，注意力机制能在大量背景特征中提取出关键的前景特征信息。另一方面，注意力机制也能对特征元素之间的关联信息进行建模，从而提升知识蒸馏的效果。

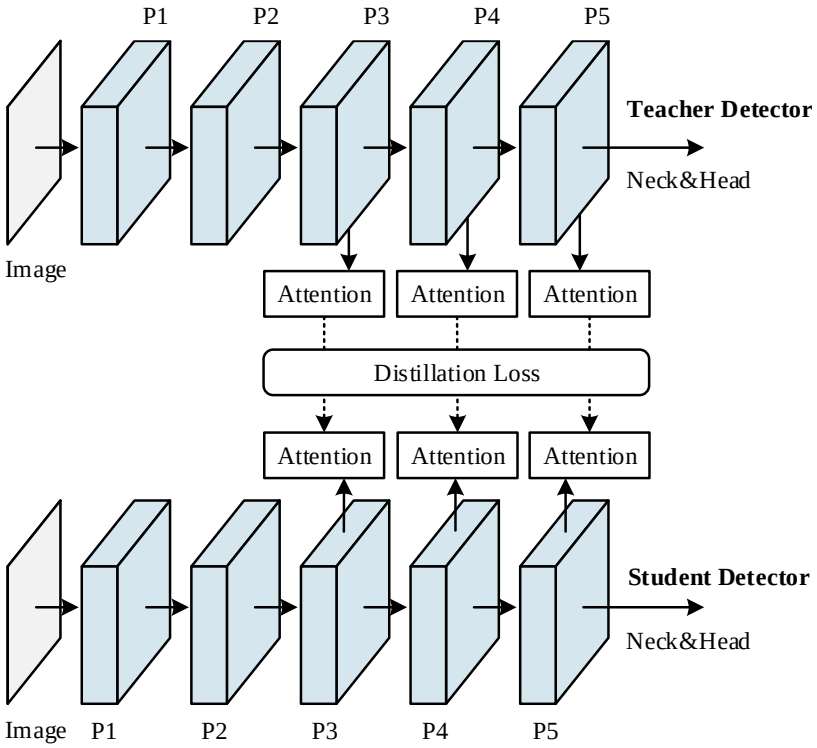


图 4.4 基于注意力机制的知识蒸馏算法框架

基于注意力机制的知识蒸馏算法框架如图 4.4 所示。相较于让学生网络直接模仿教师网络的多尺度特征特性的方法，基于注意力机制的知识蒸馏方法首先用注意力（Attention）模块分别学习教师网络与学生网络的注意力热图，然后让学生网络通过蒸馏损失（Distillation Loss）模仿教师网络的注意力热图，从而更好地捕捉有用的前景特征信息，并提取特征元素之间的关联信息。在基于注意力机制的知识蒸馏框架中，常用的注意力模块包括 Non-local[91]和 RCAA[92]等，蒸馏损失则包括 L2 范数损失[56]和 KL 散度损失[57]等。此外，为获得更好的知识蒸馏效果，可让学生网络同时模仿教师网络的多尺度特征特性和注意力热图，使得学生网络在蒸馏的过程中能进一步捕捉额外的特征细节信息。

4.2.3 算法整体框架

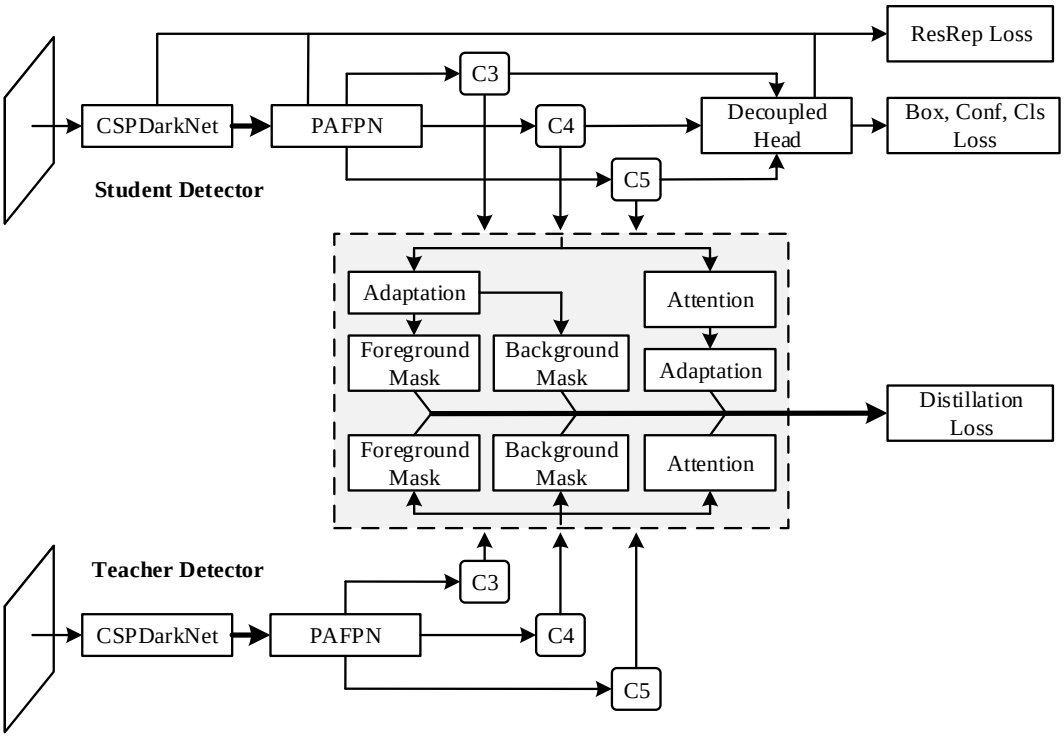


图 4.5 论文设计的模型压缩算法整体框架

在第 3 章优化的 YOLOX 模型基础上，论文结合 ResRep 通道剪枝与基于注意力机制的知识蒸馏算法，设计了一种适用于目标检测的模型压缩方法，其整体框架如图 4.5 所示。图中学生检测器（Student Detector）为待剪枝的小网络，教师检测器（Teacher detector）则为预先训练的大网络。

在训练过程中，知识蒸馏与 ResRep 通道剪枝同步进行，且相互之间几乎没有影响。对于 ResRep 通道剪枝部分，其首先用已训练的模型参数初始化学生检测器，并在网络所有卷积块中插入构造的等价 1×1 卷积层，然后对网络中构造的 1×1 卷积层施加 Group LASSO 稀疏惩罚，同时借助通道筛选和梯度重置操作，逐步裁剪网络中的通道参数，直至通道裁剪比例达到预设值。对于知识蒸馏部分，其通过设计的蒸馏辅助网络和蒸馏损失，使学生网络同时模仿教师网络的多尺度特征特性和注意力热图。蒸馏使用的多尺度特征为 PAFPN 输出的 C3、C4 和 C5 特征，每个尺度的特征均对应一组不共享参数的蒸馏辅助网络。训练结束后，去除蒸馏辅助网络，并对学生检测器网络中全部卷积块进行算子融合、通道裁剪和等价转换操作，得到剪枝的紧凑网络用于路侧端部署。

4.2.4 辅助网络设计

在论文设计的模型压缩算法中，为实现对第 3 章优化的模型进行 ResRep 通道剪枝和基于注意力机制的知识蒸馏，需引入相应的辅助网络。具体的辅助网络设计包括以下两个部分。

1. 修改 YOLOX 网络中的卷积块结构

通过对 YOLOX 网络中的卷积块结构进行修改，ResRep 在合适的位置插入构造的等价 1×1 卷积层，使得训练结束后能进行通道裁剪和等价转换操作。对于第 3 章优化的 YOLOX 网络，其卷积块包括 ConvBlock、RepVGG-block 和 RepVGG-CA。经修改的卷积块结构分别如图 4.6 和 4.7 所示，其分别在 ConvBlock 的 BN 层，以及 RepVGG-block 与 RepVGG-CA 的残差连接后面插入构造的等价 1×1 卷积层。在 ResRep 的测试阶段，构造的 1×1 卷积层将与融合后的卷积层进行合并。

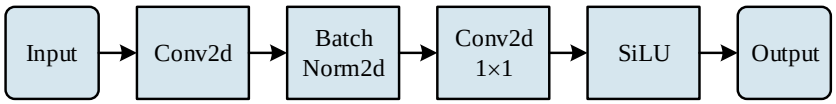
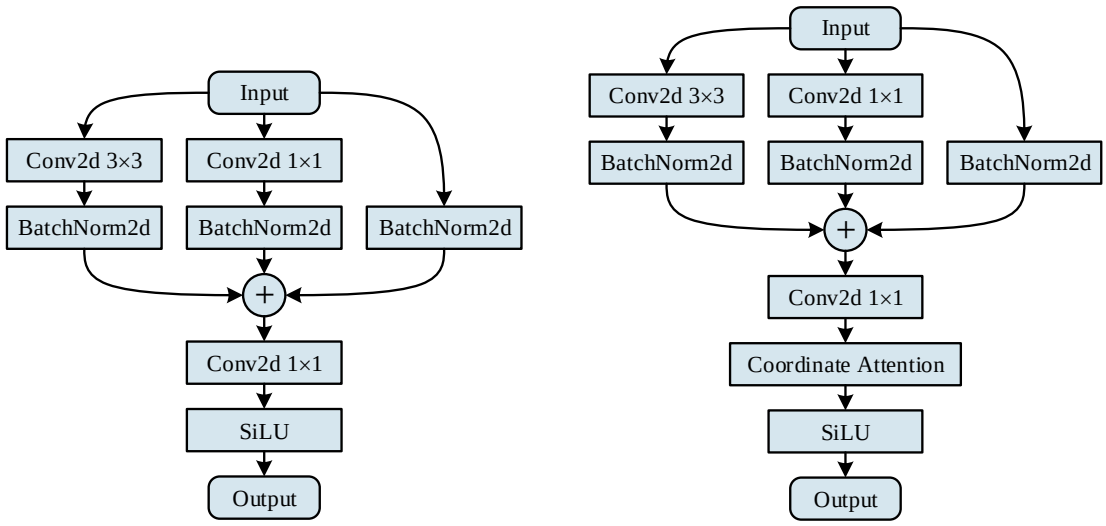


图 4.6 插入等价 1×1 卷积层的 ConvBlock



(a) RepVGG-block　　(b) RepVGG-CA

图 4.7 插入等价 1×1 卷积层的 RepVGG-block 和 RepVGG-CA

2. 引入解耦特征与注意力机制的蒸馏辅助网络设计

作为教师网络向学生网络传递知识的桥梁，蒸馏辅助网络在基于注意力机制的知识蒸馏中具有重要作用，其主要包括 Adaption 和 Attention 两个部分。其中，Adaption 用于对学生网络提取的多尺度特征进行通道适配，使其与教师网络提取

的特征通道数相同。Attention 用于提取学生网络与教师网络的注意力热图，通常使用视觉自注意力（Self-attention）模块实现功能。在论文设计的辅助网络中，Adaption 采用 3×3 卷积层，Attention 则采用 Non-local 模块。此外，为更好地捕捉多尺度特征中前景和背景特征的细节信息，论文引入解耦特征（Decoupled Feature）[57]，其用掩码分别获取前景与背景特征进行学习，能有效缓解前景与背景特征不平衡的问题，从而进一步提升知识蒸馏的效果。

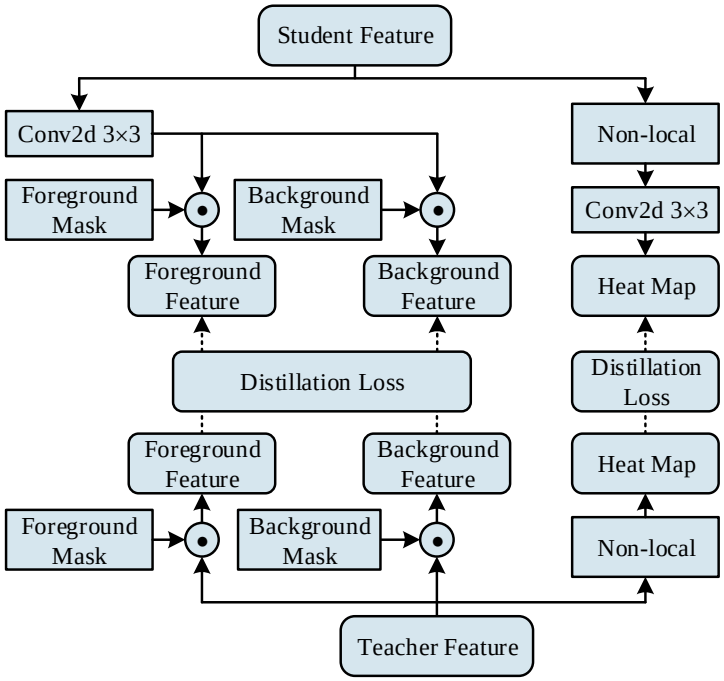


图 4.8 论文设计的蒸馏辅助网络结构

论文设计的蒸馏辅助网络结构如图 4.8 所示，其包括解耦特征和注意力热图两个蒸馏分支。在解耦特征蒸馏分支中，学生网络提取的特征经过 3×3 卷积层进行通道适配后，用正负样本匹配中所得前景与背景掩码分别对特征进行前景特征和背景特征的提取，然后与教师网络提取的前景和背景解耦特征一起计算蒸馏损失。在注意力热图分支中，学生网络提取的特征先经过 Non-local 模块得到注意力热图，然后用 3×3 卷积层对注意力热图进行通道适配，最后与教师网络提取的注意力热图一起计算蒸馏损失。

蒸馏辅助网络中 Non-local 模块的网络结构如图 4.9 所示。假设输入特征的尺寸为 C×W×H，其首先使用三个 1×1 卷积层分别得到左、中、右三个分支的特征，并对其分别进行特征尺寸的重排列。然后，将中、右两个分支的特征进行矩阵乘积，并将得到的特征经过 Softmax 激活函数后，再与左分支的特征进行矩阵乘积。

最后，在重排列所得特征的尺寸后，使用 1×1 卷积层对其进行特征映射，并将映射后的特征与输入特征进行逐元素相加，得到经过加权的输出特征。Non-local 模块通过大的网络感受野，在广阔视野中能捕捉更为全面的注意力热图，从而使网络更关注对蒸馏有益的特征信息。

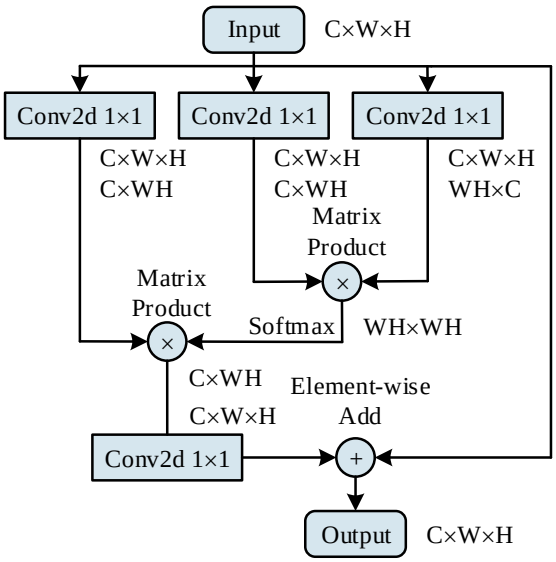


图 4.9 Non-local 模块网络结构

4.2.5 辅助损失函数设计

除第 3 章优化的模型损失函数外，论文设计的模型压缩方法在训练中还需借助辅助性质的 ResRep 损失（ResRep Loss）和蒸馏损失（Distillation Loss）对网络参数进行优化。具体的辅助损失函数设计包括以下两个部分。

1. 直接作用于参数梯度的 Group LASSO 损失

ResRep 通过对网络中构造的 1×1 卷积层参数施加 Group LASSO 损失，以实现对通道重要性的自适应鉴别。在 ResRep 梯度重置的过程中，由于其采用先对参数梯度进行置零，后添加 Group LASSO 惩罚梯度的方式，因此需要直接计算 Group LASSO 损失作用于参数的梯度。假设网络中构造的 1×1 卷积层参数为K，则计算 Group LASSO 损失梯度的表达式如下：

$$G_{GroupLasso}(K)=\frac{K}{||K||_E} \tag{4.6}$$

式中，$||K||_E$——参数K的欧几里得范数

2. 整合解耦特征与注意力热图的蒸馏损失设计

论文设计的蒸馏损失由解耦特征和注意力热图两个部分组成，其中解耦特征又包括前景特征和背景特征两个部分。假设在正负样本匹配中得到的前景掩码为m^{fg}，背景掩码为m^{bg}，学生网络提取的前景特征为F^{sfg}，背景特征为F^{sbg}，注意力热图为H^s，教师网络提取的对应特征分别为F^{tfg}、F^{sbg}和H^t。

则前景特征部分的蒸馏损失表达式如下：

$$L_{fg} = \frac{1}{2N^{obj}} \sum_{k=0}^{M} \sum_i (F_{ki}^{sfg} - F_{ki}^{tfg})^2 m_{ki}^{fg} \tag{4.7}$$

式中，N^{obj}——单个批次训练数据中正样本的数量

M　——多尺度特征的数量

背景特征部分的蒸馏损失表达式如下：

$$L_{bg} = \frac{1}{2N^{obj}} \sum_{k=0}^{M} \sum_i (F_{ki}^{sbg} - F_{ki}^{tbg})^2 m_{ki}^{bg} \tag{4.8}$$

注意力热图部分的蒸馏损失表达式如下：

$$L_{map} = \sum_{k=0}^{M} \sum_i (H_{ki}^{s} - H_{ki}^{t})^2 \tag{4.9}$$

在训练过程中，假设原网络的损失函数为L_{total}，则加入结合解耦特征与注意力热图蒸馏损失的总损失表达式如下：

$$\bar{L}_{total} = L_{total} + \frac{1}{M}(\lambda_{fg} L_{fg} + \lambda_{bg} L_{bg} + \lambda_{map} L_{map}) \tag{4.10}$$

式中，λ_{fg}　——前景特征损失平衡系数

λ_{bg}　——背景特征损失平衡系数

λ_{map}——注意力热图损失平衡系数

在每次迭代更新的过程中，还需单独计算网络中构造的 1×1 卷积层参数对应的 Group LASSO 惩罚梯度。

4.2.6 模型压缩训练过程

论文设计的模型压缩方法在训练过程中包括三个阶段。其中，一阶段包括训练开始时用于预热（Warm-up）的若干个 Epoch，其用第 3 章优化的模型参数初始化网络，并在网络所有卷积块中插入构造的等价 1×1 层，且对网络中构造的 1×1 卷积层参数施加 Group LASSO 惩罚，但不进行 ResRep 通道筛选与梯度重置操作，也不对网络进行知识蒸馏。二阶段包括预热后使用 Mosaic 和 Mixup 数据增强的若

干个 Epoch，其开始逐步对网络中构造的 1×1 卷积层进行通道筛选与梯度重置操作，但不包括知识蒸馏的部分。三阶段包括停止 Mosaic 和 Mixup 数据增强后的若干个 Epoch。在此阶段中，ResRep 的通道筛选操作已完成，开始引入教师网络、蒸馏辅助网络和蒸馏损失恢复网络的部分精度。相较于从训练开始进行知识蒸馏的方式，论文设计的模型压缩训练过程在保证性能的前提下，可有效对 ResRep 通道剪枝与知识蒸馏进行训练过程的解耦，并能节省大量的训练时间。

4.3 目标检测模型的移植与量化

在对第 3 章优化的模型进行压缩后，可得到用于路侧边缘计算平台部署的紧凑模型。模型部署通常需要借助软件工具对模型进行移植和量化等，根据边缘计算平台的不同，其所用到的软件工具也各不相同。论文使用 NVIDIA Jetson 系列嵌入式开发套件作为路侧边缘计算设备，部署模型时使用的软件工具包括 TensorRT 和 DeepStream。其中，TensorRT 用于对模型进行移植和量化，DeepStream 则用于实现对多路视频流的高效推理。

4.3.1 TensorRT 模型部署工具

TensorRT 是 NVIDIA 在统一计算设备架构（Compute Unified Device Architecture，CUDA）基础上构建的神经网络推理优化库，其可用于人工智能、自主机器和高性能计算等场景，并支持从目前主流的深度学习框架中导入训练的模型进行优化，以及部署到数据中心或嵌入式设备。为满足实时服务和嵌入式应用的需求，TensorRT 内置有用于对模型进行 Int8 或 Float16 量化的工具包，通过低精度量化后的模型可显著减少应用延迟。

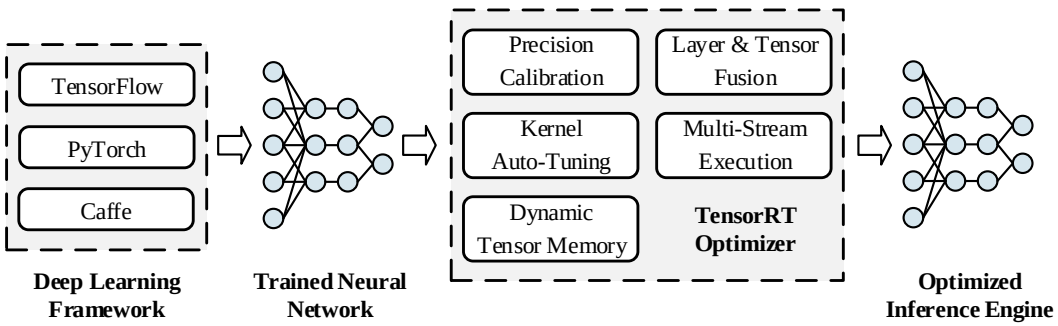


图 4.10 TensorRT 模型部署整体框架

使用 TensorRT 部署模型的整体框架如图 4.10 所示，其首先从 TensorFlow、PyTorch 或 Caffe 等深度学习框架中导入训练好的模型。然后，使用 TensorRT 优化器对模型进行优化。最后，根据优化的模型生成适配目标设备的推理引擎。在模型优化的过程中，主要包括精度校准、层与张量融合、内核自动调整、多流执行和动态张量显存五个部分。其中，精度校准对模型进行 Int8 或 Float16 量化，可在保持模型精度的同时，极大限度地提高吞吐量。层与张量融合是对模型计算图中的部分节点进行合并，从而优化对 GPU 显存和带宽的使用。内核自动调整是根据目标 GPU 设备自动选择最佳的数据层和算法。多流执行使得 TensorRT 具备并行处理多个输入流的能力。动态张量显存则能减少显存占用，并在推理过程中实现高效重复利用内存。

4.3.2 DeepStream 视频流推理框架

DeepStream 是由 NVIDIA 在 GStreamer 媒体框架基础上开发的智能视频分析（Intelligent Video Analysis，IVA）库，其具有多平台、可扩展和安全加密等特点，可部署在本地、边缘侧和云端。DeepStream 完全兼容 TensorRT，支持多种用于图像分类、目标检测和图像分割的模型，并提供有包括 SSD、Mask-RCNN、YOLOv3 和 RetinaNet 等应用例程。为方便开发人员快速投入应用，DeepStream 还提供有 Graph Composer 低代码编程工具。借助 DeepStream 库，用户可将 AI 应用于流视频，并可同时优化视频编解码、图像缩放和转换，以及边缘侧到云端的连接，从而实现完整的端到端性能优化。

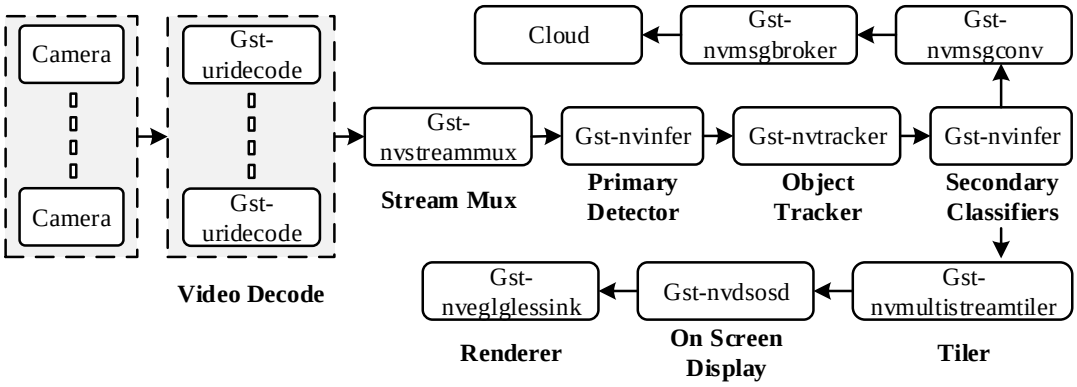


图 4.11 DeepStream 视频流分析整体框架

DeepStream 视频流分析的整体框架如图 4.11 所示，其首先从接入的若干路摄像头采集实时视频流，并对每路视频流使用 Gst-uridecode 组件进行视频解码。然后，用 Gst-nvstreammux 组件将解码得到的多路视频帧组成一个批次，并作为视觉模型的输入。最后，用 Gst-infer 组件调用目标检测模型处理多路视频帧，并用 Gst-nvtracker 组件对得到的检测结果执行目标跟踪，得到每个目标独有的编号。当配置有分类模型时，Gst-nvinfer 组件还会从输入视频帧中裁剪出每个目标对应的图像块，并调用分类模型进行分类。在视频流分析的过程中，本地或边缘侧可用 Gst-nvmultistreamtiler 和 Gst-nvdsosd 组件对多路视频流与分析结果进行可视化，并可用 Gst-nveglglessink 组件保存或转换可视化的视频。此外，DeepStream 也支持用 Gst-nvmsgconv 和 Gst-nvmsgbroker 组件将分析数据上传至云端。

4.3.3 模型移植与量化过程

部署经过优化与压缩的模型到 NVIDIA Jetson 嵌入式开发套件的主要工作包括对模型的移植和量化。在模型移植过程中，论文首先使用 TensorRT 库函数对优化的 YOLOX 网络进行重构，包括编写用于输出解码的插件。然后，将训练好的模型文件进行格式转换，用于对构建的 TensorRT 网络进行赋值。在模型量化过程中，首先随机从 BDD100K 的验证集中抽取 1000 张图像，作为 Int8 量化的精度校准数据集，然后调用 TensorRT 优化工具对网络的参数和激活值进行校准，并最终得到用于目标设备推理的引擎文件。

4.4 实验与分析

在实验室现有的深度学习工作站上，本节对本章设计的模型压缩方法开展对比实验，以验证所设计方法的有效性。实验使用的数据集为 BDD100K，评价指标包括模型压缩比率、精度和实时性指标，对比的基线（Baseline）方法为 Z. Liu 等[49]提出的通道剪枝算法。实验数据均来自相同的测试环境。

4.4.1 实验环境与评价指标

实验中使用的深度学习工作站软、硬件环境与第 3 章实验一致，评价指标除第 3 章实验用到的 AP、mAP、Params、FLOPs 和 FPS 外，还包括 FLOPs 裁剪比率和 Params 裁剪比率。其中，FLOPs 裁剪比率为剪枝减少的 FLOPs 与原网络 FLOPs 的比值，Params 裁剪比率则为剪枝减少的参数量与原网络参数量的比值。

4.4.2 实验过程

在 BDD100K 训练集上，分别用基线方法和本章设计的模型压缩方法对第 3 章优化的模型进行训练。训练基线方法时，使用与第 3 章实验相同的训练配置，并对网络中 BN 层的γ参数施加系数为 0.0001 的 L1 范数惩罚。训练结束后，对网络进行通道裁剪，并对剩余的网络参数用相同的训练配置微调 100 个 Epoch。训练本章设计的模型压缩方法时，初始学习率为 0.001，共训练 400 个 Epoch，其中包括预热阶段的 5 个 Epoch，以及关闭 Mosaic 和 Mixup 数据增广后的 25 个 Epoch。对于 ResRep 部分，Group LASSO 惩罚项系数λ取 0.0001，每 200 次迭代进行一次掩码更新，每次更新时将 4 个有效通道标记为无效。对于知识蒸馏部分，λ_{fg}系数取 3.0，λ_{bg}系数取 12.0，λ_{map}系数取 0.002。训练完成后，再利用 TensorRT 和 DeepStream 将压缩后的模型部署在深度学习工作站上进行实时性测试。

4.4.3 结果分析

表 4.2 模型压缩方法的精度评估结果

测试模型	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOX-s	28.1	48.6	27.5	9.5	34.8	54.5
第 3 章模型	29.6	52.7	27.5	11.9	34.4	53.8
基线模型 (50%)	28.2	51.1	26.4	11.6	33.2	50.2
基线模型 (60%)	26.1	47.5	24.1	10.0	30.8	47.8
本章模型 (50%)	28.4	51.2	26.5	11.5	33.3	51.1
本章模型 (60%)	27.8	50.7	25.6	11.4	33.0	49.9

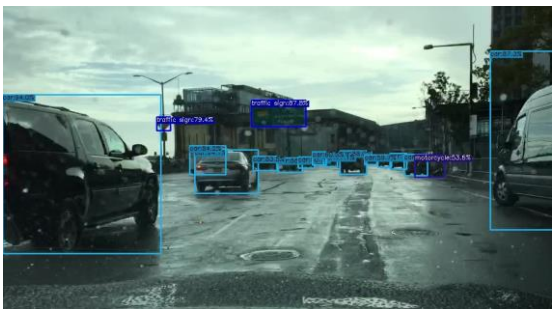
论文设计的方法与基线方法在 BDD100K 验证集上得到的精度评估结果如表 4.2 所示。实验中选取的模型为网络深度系数为 0.33 和宽度系数为 0.50 的 Small

版本。表中的 50%和 60%表示对第 3 章所得模型进行通道参数裁剪的比例。从表中可以看出，在裁剪 50%通道参数的情况下，本章设计的方法精度相比原版 YOLOX 更高，且比基线方法也略高。在裁剪 60%通道参数的情况下，本章设计的方法精度比原版 YOLOX 稍差，但相比基线方法有明显的优势。其原因可能在于，基线方法在训练结束后，一次性完成对通道参数的裁剪。当裁剪比例较大时，网络的部分重要通道参数将被直接去除，从而导致精度出现较大幅度的下滑。本章设计的方法则采用在训练中逐渐裁剪通道的方式，网络会在裁剪的过程中逐步进行微调，因此能更好地保留网络性能。

表 4.3 模型的各类别精度评估结果

测试模型	Bus	Traffic Light	Traffic Sign	Pedestrian	Bicycle	Truck	Motor-cycle	Car	Train	Rider
YOLOX-s	45.7	16.5	30.0	28.5	22.0	45.7	22.0	48.7	0	22.0
第 3 章模型	44.8	24.4	35.8	30.0	22.2	44.1	22.8	50.4	0	21.1
基线模型(50%)	42.3	24.0	35.4	29.3	19.9	42.3	18.7	49.8	0	20.1
基线模型(60%)	38.9	22.3	32.8	26.6	17.6	39.7	17.0	48.3	0	18.1
本章模型(50%)	43.5	24.0	35.4	29.3	19.8	43.0	18.8	49.8	0	20.4
本章模型(60%)	41.1	23.6	35.0	28.9	19.9	41.2	18.6	49.4	0	20.0

本章设计的方法与基线方法在 BDD100K 验证集上得到的各类别精度评估结果如表 4.3 所示。在裁剪 50%通道参数的情况下，本章设计的方法对主要的 Pedestrian 和 Car 类别的检测精度与基线方法相同。但在裁剪 60%通道参数的情况下，本章设计的方法检测精度明显优于基线方法。

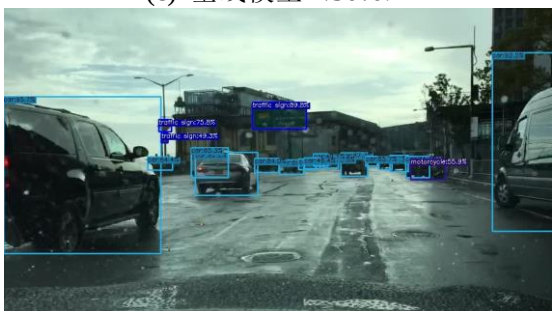


(a) YOLOX-s

(b) 第 3 章模型


(c) 基线模型（50%）


(d) 基线模型（60%）


(e) 本章模型（50%）


(f) 本章模型（60%）

图 4.12 模型的测试结果对比

本章设计的方法与基线方法的测试结果对比如图 4.12 所示。从图中可以看出，在对第 3 章模型进行通道裁剪后，所得模型的检测效果与未裁剪模型相近。相较于基线方法，本章设计的方法对小目标的检测效果更好。

表 4.4 模型的实时性评估结果

测试模型	Params（Million）	FLOPs（Billion）	Params 裁剪比率	FLOPs 裁剪比率	FPS（Average）
YOLOX-s	8.94	26.65	—	—	284
第 3 章模型	9.62	27.85	—	—	304
基线模型 (50%)	3.04	19.75	68.32%	29.06%	362
基线模型 (60%)	2.06	16.76	78.57%	39.82%	409
本章模型 (50%)	2.92	19.21	69.58%	31.01%	373
本章模型 (60%)	2.07	16.71	78.50%	40.00%	411

论文设计的方法与基线方法进行推理实时性评估的结果如表 4.4 所示。测试模型的推理批大小（Batch Size）为 64，测试的输入分辨率为 640×640。在裁剪 50% 通道参数的情况下，本章设计的方法相比基线方法有更高的参数与 FLOPs 裁剪比率，同时 FPS 相比原版 YOLOX 和第 3 章模型分别提升 31%和 22%。在裁剪 60% 通道参数的情况下，本章设计的方法与基线算法的参数与 FLOPs 裁剪比例相近，FPS 则相比原版 YOLOX 和第 3 章模型则分别提升 44%和 35%。

表 4.5 模型部署的实时性测试结果（Float16 量化）

测试模型（FP16）	FPS（1 路）（Average）	FPS（4 路）（Average）	FPS（8 路）（Average）
YOLOX-s	559	174	84
第 3 章模型	608	175	84
基线模型 (50%)	709	177	86
基线模型 (60%)	742	178	86
本章模型 (50%)	711	177	86
本章模型 (60%)	745	178	86

论文设计的方法与基线方法部署到深度学习工作站，并进行 Float16 量化的推理实时性测试结果如表 4.5 所示，测试的输入分辨率为 640×640。在处理 1 路视频流时，本章设计的方法在裁剪 50%通道参数的情况下，FPS 相比原版 YOLOX 和第 3 章模型分别提升 27%和 17%。当裁剪 60%通道参数时，FPS 则分别提升 33%和 22%。但在同时处理 4 路或 8 路视频流时，本章设计的方法 FPS 与其他对比方法相近。其原因可能在于，当以高 FPS 同时处理 4 路或 8 路视频流时，深度学习工作站出现了硬件性能瓶颈，其推理速度不再由模型的计算量主导，转而由内存带宽、总线带宽和视频流编解码性能等因素所决定。

表 4.6 模型部署的实时性测试结果（Int8 量化）

测试模型（Int8）	FPS（1 路）（Average）	FPS（4 路）（Average）	FPS（8 路）（Average）
YOLOX-s	618	180	88
第 3 章模型	675	181	88
基线模型 (50%)	750	181	89
基线模型 (60%)	751	182	89
本章模型 (50%)	750	181	89
本章模型 (60%)	751	182	89

论文设计的方法与基线方法部署到深度学习工作站，并进行 Int8 量化的推理实时性测试结果如表 4.6 所示，测试的输入分辨率为 640×640。由于深度学习工作站出现了性能瓶颈，本章设计的方法仅在处理 1 路视频流时，在裁剪 50%或 60%通道参数的情况下，FPS 相比原版 YOLOX 和第 3 章模型分别有 21%和 11%的提升。通常情况下，使用 Int8 量化能带来更大的推理速度提升，但其也会导致模型

的精度出现明显下滑，并需要将网络的激活函数替换为 ReLU6 进行重新训练。通常在 Float16 量化不满足实时性要求的情况下，才会采用 Int8 量化方法。

综上所述，本章设计的方法具有比基线方法更好的模型压缩效果，能在模型精度无明显下滑的前提下，显著提升模型的检测速度。

4.5 本章小结

本章对路侧场景下的视觉目标检测模型压缩方法进行阐述，首先分析了在路侧场景中模型压缩的必要性，包括对边缘计算设备的成本与算力分析，以及路侧场景对算法的实时性需求。然后，提出了论文设计的结合通道剪枝与知识蒸馏算法的模型压缩方法。接着，分析了在路侧边缘计算平台上部署模型所用到的工具以及模型部署的过程。最后，开展具体的实验验证了论文所提方法的有效性。通过实验证明，本章设计的模型压缩方法能在模型无明显精度下滑的前提下，显著降低模型的计算量，从而大幅提升模型的推理速度。

第 5 章 测试与验证

在对 YOLOX 目标检测模型进行优化和压缩，以及在深度学习工作站上进行部署测试后，验证了论文所提方法的有效性。本章利用学校现有的智能路侧平台对论文所得模型进行精度和实时性测试。其中，精度测试部分使用路侧平台远程采集并标注真实的路侧场景图像作为测试集，用于评估模型的检测精度。实时性测试部分则部署模型到路侧边缘计算平台，以测试模型的推理速度。

5.1 测试环境搭建

5.1.1 测试平台搭建

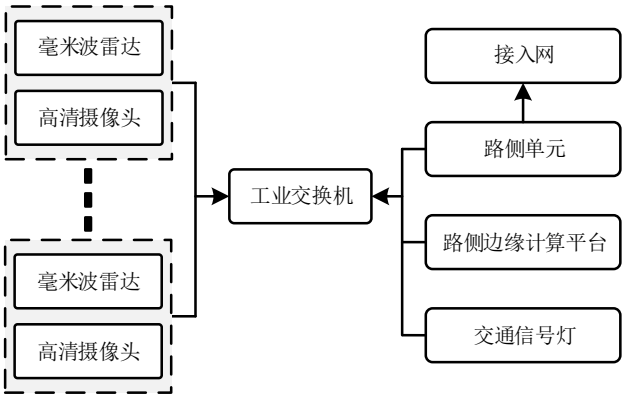


图 5.1 校内路侧平台系统框架

校内搭建的路侧平台系统框架如图 5.1 所示，其通过工业交换机组成一个局域网络，并连接有路侧单元、路侧边缘计算平台、交通信号灯和若干路毫米波雷达与高清摄像头。其中，高清摄像头使用的型号为华为 M6741-E-Z37，其支持白天与夜晚不同时段的高清视频流采集,且输出视频流的时延较低,适合作为路侧感知设备。华为 M6741-E-Z37 的主要参数如表 5.1 所示，外形则如图 5.2 所示。

表 5.1 华为 M6741-E-Z37 摄像头主要参数

摄像头属性	技术参数
图像传感器	1/1.8 英寸 CMOS
有效像素	400 万
视频编码格式	H.265/H.264/M-JPEG

表 5.1 华为 M6741-E-Z37 摄像头主要参数（续表）

摄像头属性	技术参数
最大分辨率	2560×1440
镜头焦距	5.6-208mm
光学变焦倍率	37 倍
传输方式	RJ45 以太网口
电源电压	AC 24V
工作温度	-40℃-+60℃

图 5.2 华为 M6741-E-Z37 摄像头

路侧边缘计算平台采用 NVIDIA Jetson AGX Xavier 开发套件，其搭载 ARMv8 八核 CPU 和具有 512 CUDA 核心的 Volta GPU，并配置有 32GB 的 LPDDR4x 高速内存，在 Int8 数据格式的计算峰值性能可达 32 TOPS，能满足路侧边缘计算平台的算力要求。NVIDIA Jetson AGX Xavier 的主要参数如表 5.2 所示，外形如图 5.3 所示，配置的主要软件环境则如表 5.3 所示。

表 5.2 NVIDIA Jetson AGX Xavier 开发套件主要参数

开发套件属性	技术参数
峰值算力	32 TOPS (Int8)
CPU	8 核 Armv8.2 64 位、8MB L2 + 4MB L3
GPU	NVIDIA Volta 架构、512 CUDA 核心
内存	32GB 256 位 LPDDR4x
存储	64GB eMMC 5.1
视频编解码	H.265/H.264
网络	10/100/1000M 以太网
功耗	20W/40W

图 5.3 NVIDIA Jetson AGX Xavier 开发套件

表 5.3 路侧边缘计算平台主要软件环境

软件名称	版本	软件名称	版本
Ubuntu	18.04	OpenCV	4.1.1
CUDA	10.2	TensorRT	7.1.3
cuDNN	8.0	DeepStream	5.1

在测试过程中，路侧边缘计算单元通过组建的局域网，采集高清摄像头的实时视频流进行视觉目标检测，并将检测结果与毫米波雷达发送的检测数据进行多传感器融合计算，得到的融合感知信息将发送至路侧单元进行消息广播。

5.1.2 测试数据集准备

利用搭建的测试平台，论文采用远程采集并标注的实际路侧场景图像作为测试数据集，用于测试第 4 章所得模型的性能，截取的部分路侧场景图像如图 5.4 所示。测试数据集包括不同时段、天气的 411 张图像，共标注有 6 个类别，每个类别对象的数目统计如表 5.4 所示。

表 5.4 测试数据集标注的不同类别对象数目统计

类别名称	标注数量	类别名称	标注数量
Pedestrian	4137	Car	1107
Bicycle	39	Truck	40
Motorcycle	158	Bus	0

图 5.4 测试数据集图像截图

5.2 测试过程与结果分析

5.2.1 测试过程

本章对论文所得模型进行精度与实时性的对比测试。在精度测试过程中，分别用原版 YOLOX、第 3 章和第 4 章所得模型在测试数据集上进行精度评估，评估方法采用 COCO。在实时性测试过程中，首先借助 TensorRT 和 DeepStream 工具，将原版 YOLOX、第 3 章和第 4 章所得模型部署到路侧边缘计算单元。然后，对其分别进行包括 1 路、4 路和 8 路视频流推理的实时性测试。

5.2.2 测试结果分析

表 5.5 模型在测试数据集上的精度评估结果

测试模型	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOX-s	33.4	54.0	34.8	3.1	30.8	49.0
第 3 章模型	36.4	58.2	38.2	5.2	32.3	58.0
第 4 章模型 (50%)	35.0	55.8	36.7	4.6	30.7	53.8
第 4 章模型 (60%)	32.6	51.9	34.1	3.9	30.8	51.2

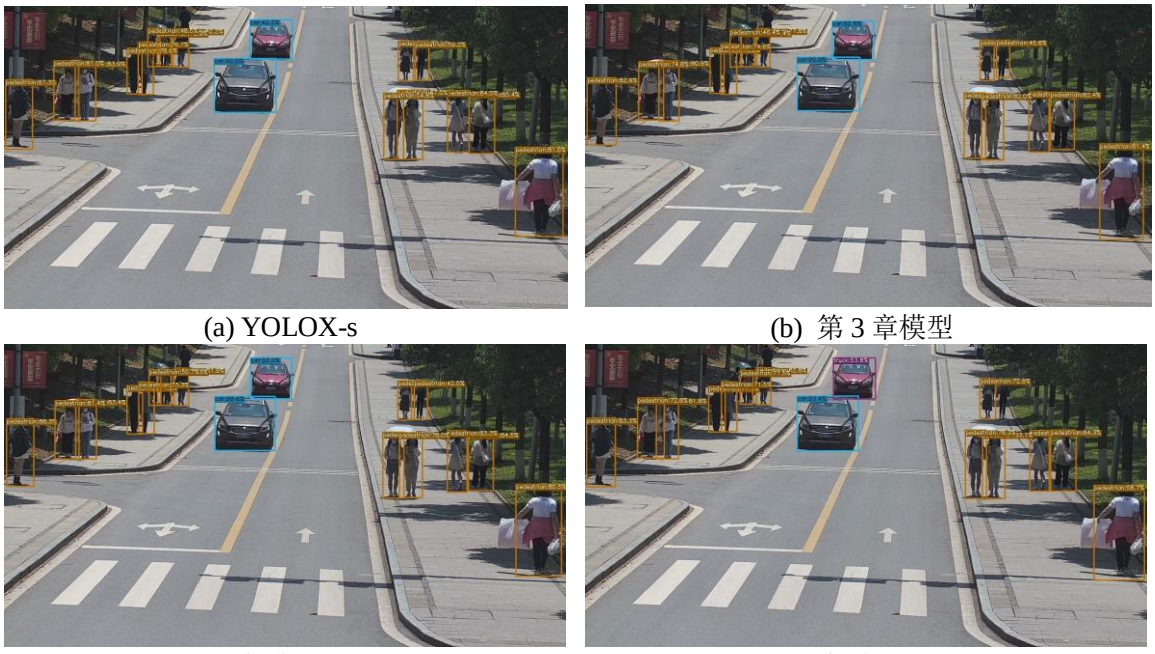
论文所得模型与原版 YOLOX 在测试数据集上得到的评估结果如表 5.5 所示。实验中选取的模型为网络深度系数为 0.33 和宽度系数为 0.50 的 Small 版本。从表中可以看出，在校内实际路侧场景中，将论文第 3 章模型裁剪 50%通道参数时，所得

模型相比原版 YOLOX 有明显的检测精度优势。在裁剪 60%通道参数时，所得模型的检测精度略低于原版 YOLOX，但AP_S和AP_L指标依然占优。表明论文所提方法在路侧场景中具有更好的泛化性能。

表 5.6 模型在测试数据集上的各类别精度评估结果

测试模型	Bus	Pedestrian	Bicycle	Truck	Motorcycle	Car
YOLOX-s	0	44.9	13.6	29.9	22.3	56.1
第 3 章模型	0	48.1	22.0	25.4	28.1	58.4
第 4 章模型 (50%)	0	45.6	13.7	32.9	24.7	57.9
第 4 章模型 (60%)	0	44.1	12.9	26.8	23.2	56.0

论文所得模型与原版 YOLOX 在测试集上得到的各类别精度评估结果如表 5.6 所示。在对主要的 Pedestrian 和 Car 类别的评估中，相较于原版 YOLOX，论文第 3 章模型在裁剪 50%通道参数的情况下，其检测精度更高。在裁剪 60%通道参数的情况下，检测精度略低于原版 YOLOX。



(a) YOLOX-s (b) 第 3 章模型

(c) 第 4 章模型（50%） (d) 第 4 章模型（60%）

图 5.5 模型的测试结果对比

论文所得模型与原版 YOLOX 的测试结果对比如图 5.5 所示。从图中可以看出，论文第 3 章模型在裁剪 50%通道参数的情况下，其检测效果相比原版 YOLOX 更好。在裁剪 60%通道参数的情况下，检测效果相比原版 YOLOX 略差，但依然能对图中的小目标和密集目标进行较为准确的检测。

表 5.7 模型部署的实时性测试结果（Float16 量化）

测试模型（FP16）	FPS（1 路）（Average）	FPS（4 路）（Average）	FPS（8 路）（Average）
YOLOX-s	101	31	16
第 3 章模型	108	33	17
第 4 章模型 (50%)	135	41	21
第 4 章模型 (60%)	146	45	23

论文所得模型与原版 YOLOX 部署到路侧边缘计算平台，并采用 Float16 量化的推理实时性测试结果如表 5.7 所示，测试的输入分辨率为 640×640。从表中可以看出，在处理 1 路与 4 路视频流时，待测模型的 FPS 均超过 30。但在同时处理 8 路视频流时，原版 YOLOX 与第 3 章模型的 FPS 低于 20。相对比地，将第 3 章模型裁剪 50%或 60%通道参数时，所得模型的 FPS 均超出 20。表明论文所提方法能满足路侧感知对视觉目标检测算法的实时性要求。

表 5.8 模型部署的实时性测试结果（Int8 量化）

测试模型（Int8）	FPS（1 路）（Average）	FPS（4 路）（Average）	FPS（8 路）（Average）
YOLOX-s	140	46	24
第 3 章模型	152	48	26
第 4 章模型 (50%)	185	59	32
第 4 章模型 (60%)	187	63	35

论文所得模型与原版 YOLOX 部署到路侧边缘计算平台，并采用 Int8 量化的推理实时性测试结果如表 5.8 所示，测试的输入分辨率为 640×640。从表中可以看出，使用 Int8 量化能进一步提升模型的推理速度。在同时处理 8 路视频流时，待测模型的 FPS 均超过 20。但经论文所提方法压缩后的模型 FPS 能超过 30，可为具有高实时性要求的路侧感知场景提供低时延的状态感知。

综上所述，论文所提方法在实际路侧场景中表现良好。相较于原版 YOLOX，论文设计的模型在具备更好的检测精度的同时，还具有更快的检测速度，因此更适合用于路侧感知，并具备一定的实用价值。

5.3 本章小结

本章阐述在实际路侧场景中测试模型性能的内容，首先对学校现有的智能路侧平台进行概括，包括测试平台搭建和测试数据集准备。然后，对论文所得模型进行性能测试，包括用采集并标注的路侧场景图像作为测试集测试模型的精度，以及部署模型到路侧边缘计算平台进行实时性测试两个部分。测试结果表明，论文设计的面向智能路侧的视觉目标检测模型在实际路侧场景中能取得较好的检测精度，并能满足路侧感知的实时性要求，具备一定的实际应用价值。

第 6 章 总结与展望

6.1 全文总结

车路协同是智能交通发展的大趋势，路侧感知则是车路协同系统的重要组成部分。考虑到性能与成本等因素，以毫米波雷达和高清摄像头为感知设备的路侧融合感知方案最为常见。因此，视觉目标检测算法在路侧感知中具有重要作用。目前，视觉目标检测以深度学习方法为主，并在过去一段时间取得了显著的突破。作为视觉任务的基本算法之一，目标检测也一直是学术界的研究热点。在路侧感知中，对视觉目标检测算法的精度与实时性均有较高要求。然而，现有轻量化目标检测模型的精度相对较低，难以准确感知车辆和行人对象。通用目标检测模型的计算量又相对较大，部署到路侧边缘计算设备难以满足实时性需求，有必要结合模型压缩技术降低模型所需的计算量。此外，路侧场景也具有较高的复杂度，需要对通用目标检测模型进行场景适配，以达到更好的检测效果。论文在 YOLOX 目标检测算法的基础上，针对路侧场景特点以及路侧感知对算法的性能需求，设计了一种高效的车辆行人目标检测模型，全文工作的总结如下：

1. 论文对当前主流的视觉目标检测算法和模型压缩算法框架进行了概括，并分析了 YOLOX、ResRep 通道剪枝和基于注意力机制的知识蒸馏算法。此外，论文还总结了路侧场景特点以及路侧感知对视觉目标检测算法的性能需求。

2. 针对路侧感知的检测精度要求，论文在 YOLOX 目标检测算法的基础上，引入 RepVGG 卷积块和 Coordinate 注意力机制对 YOLOX 的网络结构进行优化，并用 Alpha-IoU 损失和 VariFocal 损失改进 YOLOX 的损失函数。在 BDD100K 数据集上开展的实验证明，论文所提方法能有效提升对车辆行人目标的检测精度。

3. 针对路侧感知的实时性要求，在第 3 章所得模型的基础上，论文结合 ResRep 通道剪枝与基于注意力机制的知识蒸馏算法，设计了一种适用于目标检测的模型压缩方法，以满足路侧感知对算法的实时性需求。通过在 BDD100K 数据集上开展的实验证明，论文所提方法相比基线方法有更好的模型压缩效果，能在模型无明显精度下滑的前提下，大幅降低模型所需的计算量。

4. 利用校内搭建的智能路侧平台，论文远程采集并标注实际路侧场景图像用于对论文所得模型进行精度测试，并借助 TensorRT 和 DeepStream 工具将模型部署在路侧边缘计算平台上进行实时性测试。通过实验证明，论文所得模型相比原版 YOLOX 更适合路侧场景，并具备一定的实际应用价值。

6.2 展望

论文针对车路协同系统的路侧视觉感知部分，设计了一种高效的车辆行人目标检测模型，并通过实验证明了论文所提方法的有效性。然而，论文也存在一定的不足，还需要进一步研究和改进：

1. 论文研究的检测对象为车辆和行人，未包括交通信号与标识、动物和障碍物等对象，在检测类别上存在一定的局限性。此外，论文使用的 BDD100K 数据集在视角上也与路侧场景存在一定的差异，会影响模型在路侧场景中的检测精度。

2. 论文使用 CNN 构建目标检测网络，但视觉 Transformer 有望取代 CNN 成为目标检测的主流架构。在未来的工作中，可尝试将 CNN 与视觉 Transformer 进行结合，从而在边缘计算设备上取得更好的检测效果。

3. 论文研究的 2D 目标检测算法能满足基本的车路协同应用需求，但随着车路协同系统的不断发展，相信 3D 视觉目标检测将逐渐成为研究重点。

参考文献

[1] 中华人民共和国公安部. 中华人民共和国公安部数据[EB/OL]. URL: [2021-01-07]. https://www.mps.gov.cn/n2254098/n4904352/c7647179/content.html.

[2] 高德地图. 2020 年度中国主要城市交通分析报告[EB/OL]. URL: [2021-01-18]. https://report.amap.com/download.do.

[3] 中国国家统计局. 中国统计年鉴—2020[EB/OL]. URL: [2020-09-20]. http://www.stats.gov.cn/tjsj/ndsj/2020/indexch.htm.

[4] Sjoberg K., Andres P., Buburuzan T., et al. Cooperative intelligent transport systems in Europe: current deployment status and outlook[J]. IEEE Vehicular Technology Magazine, 2017, 12(2): 89-97.

[5] 郭戈, 许阳光, 徐涛等. 网联共享车路协同智能交通系统综述[J]. 控制与决策, 2019, 34(11): 2375-2389.

[6] 张长隆, 鲍海兴, 杜仙童等. 路侧感知在智能网联汽车中的应用与未来[J]. 人工智能, 2019(01):58-66.

[7] 王翀. 车路协同环境下的高速公路匝道区域控制关键技术研究[D]. 东南大学, 2019.

[8] Hasan M. Z., Fink R., Vargas M., et al. Development and evaluation of traffic signal algorithms to support future Measure of Effectiveness (MOE)[C]//2016 IEEE International Conference on Electro Information Technology (EIT). IEEE, 2016: 0407-0412.

[9] 柴琳果. 智能车路协同交叉口间隙耦合运行控制方法[D]. 北京交通大学, 2018.

[10] Sugimoto Y., Kuzumaki S. Sip-adus: an update on Japanese initiatives for automated driving[M]//Road Vehicle Automation 5. Springer, Cham, 2019: 17-26.

[11] 中华人民共和国交通运输部. 数字交通发展规划纲要[EB/OL]. URL: [2019-07-25]. https://xxgk.mot.gov.cn/jigou/zhghs/201907/t20190725_3230528.html.

[12] 中华人民共和国国家发展和改革委员会. 智能汽车创新发展战略[EB/OL]. URL: [2020-02-10]. https://www.ndrc.gov.cn/xxgk/zcfb/tz/202002/t20200224_1221077.html.

[13] 高风.《智能网联汽车技术路线图 2.0》解读[J]. 物联网技术, 2020, 10(11):3-4.

[14] 赵梓铭, 刘芳, 蔡志平等. 边缘计算:平台、应用与挑战[J]. 计算机研究与发展, 2018, 55(02):327-337.

[15] 孙晓芳. 基于边缘计算的森林火灾视频识别技术的研究[D]. 东北林业大学, 2020.

[16] 葛悦涛, 尹晓桐. 边缘计算的发展趋势综述[J]. 无人系统技术, 2019, 2(02):60-64.

[17] Satyanarayanan M. The emergence of edge computing[J]. Computer, 2017, 50(1): 30-39.

[18] IEEE, ACM. The first IEEE/ACM symposium on edge computing[EB/OL]. URL: [2016-10-27]. http://acm-ieee-sec.org/2016/index.html.

[19] Shi W., Cao J., Zhang Q., et al. Edge computing: vision and challenges[J]. IEEE internet of things journal, 2016, 3(5): 637-646.

[20] Mao Y., Zhang J., Letaief K. B. Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems[C]//2017 IEEE wireless communications and networking conference (WCNC). IEEE, 2017: 1-6.

[21] 边缘计算产业联盟. 边缘计算产业联盟白皮书[EB/OL]. URL: [2016-11-30]. http://www.ecconsortium.org/Lists/show/id/32.html.

[22] Li M., Yu F. R., Si P., et al. Green machine-to-machine communications with mobile edge computing and wireless network virtualization[J]. IEEE Communications Magazine, 2018, 56(5): 148-154.

[23] 刘禹欣, 朱勇, 孙结冰等. Haar-like 特征双阈值 Adaboost 人脸检测[J]. 中国图象图形学报, 2020, 25(08):1618-1626.

[24] 龚露鸣, 徐美华, 刘冬军等. 基于混合高斯和 HOG+SVM 的行人检测模型[J]. 上海大学学报(自然科学版), 2018, 24(03):341-351.

[25] 曾接贤,程潇. 结合单双行人 DPM 模型的交通场景行人检测[J]. 电子学报, 2016, 44(11):2668-2675.

[26] Zou Z., Shi Z., Guo Y., et al. Object detection in 20 years: A survey[J]. arXiv preprint arXiv:1905.05055, 2019.

[27] Ren S., He K., Girshick R., et al. Faster r-cnn: towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28: 91-99.

[28] Cai Z., Vasconcelos N. Cascade r-cnn: delving into high quality object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6154-6162.

[29] Carion N., Massa F., Synnaeve G., et al. End-to-end object detection with transformers[C]//European Conference on Computer Vision. Springer, Cham, 2020: 213-229.

[30] Sun P., Zhang R., Jiang Y., et al. Sparse r-cnn: end-to-end object detection with learnable proposals[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 14454-14463.

[31] Redmon J., Divvala S., Girshick R., et al. You only look once: unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.

[32] Redmon J., Farhadi A. Yolo9000: better, faster, stronger[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271.

[33] Redmon J., Farhadi A. Yolov3: an incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[34] Liu W., Anguelov D., Erhan D., et al. Ssd: single shot multibox detector[C]//European conference on computer vision. Springer, Cham, 2016: 21-37.

[35] Lin T. Y., Goyal P., Girshick R., et al. Focal loss for dense object detection[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.

[36] Tan M., Pang R., Le Q. V. Efficientdet: scalable and efficient object detection[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 10781-10790.

[37] Du X., Lin T. Y., Jin P., et al. Spinenet: learning scale-permuted backbone for recognition and localization[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 11592-11601.

[38] Bochkovskiy A., Wang C. Y., Liao H. Y. M. Yolov4: optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[39] Tian Z., Shen C., Chen H., et al. Fcos: a simple and strong anchor-free object detector[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[40] Zhou X., Wang D., Krähenbühl P. Objects as points[J]. arXiv preprint arXiv:1904.07850, 2019.

[41] Wang J., Song L., Li Z., et al. End-to-end object detection with fully convolutional network[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 15849-15858.

[42] Sun P., Jiang Y., Xie E., et al. Onenet: towards end-to-end one-stage object detection[J]. arXiv preprint arXiv:2012.05780, 2020.

[43] Liu Z., Lin Y., Cao Y., et al. Swin transformer: hierarchical vision transformer using shifted windows[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 10012-10022.

[44] Sandler M., Howard A., Zhu M., et al. Mobilenetv2: inverted residuals and linear bottlenecks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 4510-4520.

[45] Qin Z., Li Z., Zhang Z., et al. Thundernet: towards real-time generic object detection on mobile devices[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 6718-6727.

[46] Wong A., Famuori M., Shafiee M. J., et al. Yolo nano: a highly compact you only look once convolutional neural network for object detection[J]. arXiv preprint arXiv:1910.01271, 2019.

[47] Cai Y., Li H., Yuan G., et al. Yolobile: real-time object detection on mobile devices via compression-compilation co-design[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(2): 955-963.

[48] He Y., Zhang X., Sun J. Channel pruning for accelerating very deep neural networks[C]//Proceedings of the IEEE international conference on computer vision. 2017: 1389-1397.

[49] Liu Z., Li J., Shen Z., et al. Learning efficient convolutional networks through network slimming[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2736-2744.

[50] Chen J., Zhu Z., Li C., et al. Self-adaptive network pruning[C]//International Conference on Neural Information Processing. Springer, Cham, 2019: 175-186.

[51] Wang Y., Zhang X., Xie L., et al. Pruning from scratch[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(07): 12273-12280.

[52] Liu J., Zhuang B., Zhuang Z., et al. Discrimination-aware network pruning for deep model compression[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.

[53] Hinton G., Vinyals O., Dean J. Distilling the knowledge in a neural network[J]. arXiv preprint arXiv:1503.02531, 2015.

[54] Komodakis N., Zagoruyko S. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer[C]//ICLR. 2017.

[55] Dong X., Yang Y. Network pruning via transformable architecture search[J]. Advances in Neural Information Processing Systems, 2019, 32.

[56] Zhang L., Ma K. Improve object detection with feature-based knowledge distillation: towards accurate and efficient detectors[C]//International Conference on Learning Representations. 2020.

[57] Guo J., Han K., Wang Y., et al. Distilling object detectors via decoupled features[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 2154-2164.

[58] Rastegari M., Ordonez V., Redmon J., et al. Xnor-net: imagenet classification using binary convolutional neural networks[C]//European conference on computer vision. Springer, Cham, 2016: 525-542.

[59] Jacob B., Kligys S., Chen B., et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 2704-2713.

[60] Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[61] He K., Zhang X., Ren S., et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[62] Huang G., Liu Z., Van Der Maaten L., et al. Densely connected convolutional networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 4700-4708.

[63] Iandola F. N., Han S., Moskewicz M. W., et al. Squeezenet: alexnet-level accuracy with 50x fewer parameters and< 0.5 MB model size[J]. arXiv preprint arXiv:1602.07360, 2016.

[64] Howard A., Sandler M., Chu G., et al. Searching for mobilenetv3[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 1314-1324.

[65] Zhang X., Zhou X., Lin M., et al. Shufflenet: an extremely efficient convolutional neural network for mobile devices[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6848-6856.

[66] Mehta S., Rastegari M. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer[J]. arXiv preprint arXiv:2110.02178, 2021.

[67] Lin T. Y., Dollár P., Girshick R., et al. Feature pyramid networks for object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2117-2125.

[68] Liu S., Qi L., Qin H., et al. Path aggregation network for instance segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 8759-8768.

[69] DeVries T., Taylor G. W. Improved regularization of convolutional neural networks with cutout[J]. arXiv preprint arXiv:1708.04552, 2017.

[70] Zhang H., Cisse M., Dauphin Y. N., et al. Mixup: beyond empirical risk minimization[J]. arXiv preprint arXiv:1710.09412, 2017.

[71] Yun S., Han D., Oh S. J., et al. Cutmix: regularization strategy to train strong classifiers with localizable features[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 6023-6032.

[72] Chen P., Liu S., Zhao H., et al. Gridmask data augmentation[J]. arXiv preprint arXiv:2001.04086, 2020.

[73] Zhang S., Chi C., Yao Y., et al. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 9759-9768.

[74] Li X., Wang W., Wu L., et al. Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection[J]. Advances in Neural Information Processing Systems, 2020, 33: 21002-21012.

[75] Rezatofighi H., Tsoi N., Gwak J. Y., et al. Generalized intersection over union: a metric and a loss for bounding box regression[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 658-666.

[76] Zheng Z., Wang P., Liu W., et al. Distance-iou loss: faster and better learning for bounding box regression[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(07): 12993-13000.

[77] Bodla N., Singh B., Chellappa R., et al. Soft-nms--improving object detection with one line of code[C]//Proceedings of the IEEE international conference on computer vision. 2017: 5561-5569.

[78] Ge Z., Liu S., Wang F., et al. Yolox: exceeding yolo series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.

[79] Wang C. Y., Liao H. Y. M., Wu Y. H., et al. Cspnet: a new backbone that can enhance learning capability of CNN[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020: 390-391.

[80] Ge Z., Liu S., Li Z., et al. Ota: optimal transport assignment for object detection[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 303-312.

[81] Cui C., Gao T., Wei S., et al. Pp-lcnet: a lightweight CPU convolutional neural network[J]. arXiv preprint arXiv:2109.15099, 2021.

[82] Ma N., Zhang X., Zheng H. T., et al. Shufflenet v2: practical guidelines for efficient cnn architecture design[C]//Proceedings of the European conference on computer vision (ECCV). 2018: 116-131.

[83] Ding X., Zhang X., Ma N., et al. Repvgg: making vgg-style convnets great again[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 13733-13742.

[84] Hou Q., Zhou D., Feng J. Coordinate attention for efficient mobile network design[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 13713-13722.

[85] Zhang H., Wang Y., Dayoub F., et al. Varifocalnet: an iou-aware dense object detector[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 8514-8523.

[86] He J., Erfani S., Ma X., et al. Alpha-iou: a family of power intersection over union losses for bounding box regression[J]. Advances in Neural Information Processing Systems, 2021, 34.

[87] Jiang Y., Tan Z., Wang J., et al. Giraffedet: a heavy-neck paradigm for object detection[J]. arXiv preprint arXiv:2202.04256, 2022.

[88] Yu F., Chen H., Wang X., et al. Bdd100k: a diverse driving dataset for heterogeneous multitask learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 2636-2645.

[89] Lin T. Y., Maire M., Belongie S., et al. Microsoft coco: common objects in context[C]//European conference on computer vision. Springer, Cham, 2014: 740-755.

[90] Ding X., Hao T., Tan J., et al. Resrep: lossless cnn pruning via decoupling remembering and forgetting[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 4510-4520.

[91] Wang X., Girshick R., Gupta A., et al. Non-local neural networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7794-7803.

[92] Huang Z., Wang X., Huang L., et al. Ccnet: criss-cross attention for semantic segmentation[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 603-612.

致谢

时间飞速流逝，转眼就即将毕业。在读研的三年期间，我完成了专业课程，通过了英语六级，获得了学校奖学金，参与了竞赛并获奖，自学了深度学习与计算机视觉，承担了实验室科研项目，撰写了论文与专利，以及顺利找到了工作等等。这些经历不仅使自己的眼界更为开阔，也让自己学到了很多有益终生的知识。但最令我难忘的，还是自己在实验室的那无数个白天与夜晚，以及自己对科学与技术的那一份热爱。在学校、导师、同学、父母和亲友的帮助下，我得以不断成长，借此机会，我想向你们表达我最真诚的谢意。

首先，我想感谢我的导师蒋建春教授。论文的顺利完成离不开您的悉心指导，您学识渊博，对教育、科研始终保持严谨而热诚的态度，并在研究生期间不断督促着我们进步，同时也在学习和生活当中给予了我们很多宝贵的意见。

然后，我想感谢实验室各位老师、辅导员老师和同学们。在日常生活中，我或多或少受到你们的诸多关照，非常感激能在这个阶段认识你们。

接着，我想感谢各位老师、专家和学者在百忙之中评阅我的论文，帮助我更好地完成论文。

最后，我想感谢我的父母和亲友，感谢你们在我读研期间给予我的莫大支持与关怀。在今后的学习、生活和工作当中，我将不忘初心，心怀感恩，不断提升自我，并争取能为社会做出一点微薄贡献，让自己的生命更具意义。

攻读硕士学位期间从事的科研工作及取得的成果

参与科研项目：

[1] 5G 产品规模研发试验（2018ZX03001023-006），国家科技重大专项，项目起止: 2018.6-2020.6.

[2] 基于 5G 低时延高可靠性特性的车联网络研发及应用（cstc2019jscx-zdztzxX0021），省市重大专项，项目起止: 2019.09-2022.09.

[3] 5G 车载智能终端研发及应用（cstc2019jscx-fxydX0052），省市重点项目，项目起止: 2019.09-2021.09.

发表专利：

[1] 蒋建春，**胡浪**等．一种多功能无障碍车远程监控与辅助驾驶系统．中国，202011452916.2[P]. 2020.

竞赛获奖：

[1] **胡浪**. 第十一届“蓝桥杯” C/C++程序设计竞赛（研究生组）. 省部级. 一等奖. 2020.