学校代码：10284
分 类 号：TP302.7
密　　级：公开
U D C：004.93
学　　号：MG20370042

南京大学

硕 士 学 位 论 文

论 文 题 目　基于结构重参化与信息聚合的视频目标检测研究

作 者 姓 名　肖伟康

专 业 名 称　计算机科学与技术

研 究 方 向　目标检测

导 师 姓 名　申富饶

2023 年 5 月 24 日

答辩委员会主席 戴 新 宇

评　　阅　　人 戴 新 宇

徐 明 华

论文答辩日期　　2023 年　5 月 22 日

研究生签名：

导 师 签 名：

Research on Video Object Detection Based on Structural Re-parameterization and Information Aggregation

by

Weikang Xiao

Supervised by

Professor Shen Fu-Rao

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of
MASTER
in
Computer Science and Technology

School of Artificial Intelligence
Nanjing University

May 24, 2023

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于结构重参化与信息聚合的视频目标检测研究

计算机科学与技术 专业 2020 级硕士生姓名：肖伟康

指导教师（姓名、职称）：申富饶教授

摘　要

目标检测尤其是视频目标检测任务在我们的生活中随处可见，从交通车辆监控到野生动物检测，视频目标检测可以帮助提高人们的生活质量和推动社会快速发展。传统通过人工观看视频，从中找出感兴趣目标的方法效率太低，耗时耗力。近年来以深度学习为主的目标检测方法在检测精度和推理速度上取得了惊人的成就，开始大规模应用于实际任务。实际监控视频数据会出现目标运动模糊、失焦、遮挡和奇异姿态等现象，静态目标检测算法在低质量视频帧上表现得不尽如人意。视频目标检测方法能够利用目标的时序信息，达到更高的检测精度，然而视频数据标注成本过高，并且现有的方法推理速度较慢，在实际任务中往往不是首选之策。因此本文为了解决这些问题，设计了一个新的静态目标检测模型，并在此基础上新增了一个信息聚合模块，将其扩展为视频目标检测模型，主要研究如下：

1. 本文以现有的静态目标检测模型为基础，设计了推理速度更快的模型YoloX-Lite。我们提出了一个新的基础结构 Rep-Bottleneck，整个结构只有最后一个算子是非线性的激活函数。随后将 YoloX-Lite 在训练和推理阶段使用到的结构解耦，训练阶段使用 Rep-Bottleneck 多分支结构，推理阶段使用结构重参化技术将多个线性运算算子合并为一个，提高了模型整体的检测速度，与此同时不影响模型的检测精度。在颈部网络中，YoloX-Lite 重新设计了数据流向，并制定了算子使用的规则。为了能更好的聚合不同层次的特征，YoloX-Lite 在颈部网络不同层信息融合前，使用通道注意力机制让更加重要的信息在不同层之间传递。我们在标准数据集上进行实验，结果表明 YoloX-Lite 相较于 YoloX 实现了更快的推理速度和更高的检测精度。

2. 本文在静态目标检测模型 YoloX-Lite 的基础上，提出了一个可以聚合全局和局部信息的视频目标检测模型，简称为 Yolo-GLA。该模型设计了一

个候选区选择模块，可以从 YoloX-Lite 的结果中，挑选出最有可能存在目标的一组候选框，排除了大量的低质量结果对后续信息聚合的干扰，同时降低了计算量。Yolo-GLA 设计了新型的信息聚合方式，同时聚合了全局目标的外观信息和局部目标的精确位置信息，扩大了信息聚合的范围，使得当前帧的特征表示更加丰富。我们在公开的视频目标检测数据集上实验发现，相较于 YoloX-Lite，同时结合全局和局部信息，能明显提高模型的检测精度。

3. 在视频目标检测模型 Yolo-GLA 的基础上，本文设计了一个完整的视频目标检测系统。该系统包含两个模块，分别部署在中心侧和边缘侧。中心侧模块包含数据标注，数据分析和一键训练等功能。边缘侧模块集成了 Yolo-GLA 算法，可以部署在野外环境中的边缘设备，实现全天候目标检测。整个系统实现了从数据标注到模型训练，再到模型部署的完整流程，帮助视频目标检测任务顺利落地，节省成本，提高人们的工作效率。

关键词： 静态目标检测；结构重参化；视频目标检测；全局-局部信息聚合

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Research on Video Object Detection Based on Structural Re-parameterization and Information Aggregation

SPECIALIZATION: Computer Science and Technology

POSTGRADUATE: Weikang Xiao

MENTOR: Professor Shen Fu-Rao

Abstract

Object detection, especially video object detection, is ubiquitous in our daily lives. Video object detection can help improve people's quality of life and promote rapid social development such as traffic vehicle monitoring and wild animal detection. The traditional method of manually watching the video to find the object of interest is inefficient, time-consuming and labor-intensive. In recent years, object detection methods based on deep learning have achieved amazing results in detection accuracy and inference speed. Object detection has begun to be applied to practical tasks on a large scale. In the actual surveillance video data, there are some lower-quality video frames such as target motion blur, out of focus, occlusion, and singular posture. The performance of static object detection algorithms on low-quality frames is unsatisfactory. Video object detection methods can use the context information of the object to achieve higher detection accuracy. However, the cost of video data labeling is too expensive, and the inference speed of existing methods is slow, which often make video object detection not the first choice in practical tasks. Therefore, this paper designs a new static target detection model, and adds an information aggregation module on this basis, and expands it into a video target detection model. The main research is as follows:

1. This paper proposes a new model called YoloX-Lite based on the existing static object detection model with faster inference speed. We design a new basic structure called Rep-Bottleneck, in which only the last operator of the entire structure is a nonlinear activation function. We then decouple the network used in the training and inference stages of YoloX-Lite. The multi-branch structure of Rep-Bottleneck is used in the training stage, and the structure re-parameterization technology is used in the inference

stage to combine multiple linear operators into one, which improves the overall detection speed of the model without affecting its detection accuracy. In the neck network, we redesign the data flow direction and formulate the paradigm used by operators. In order to better aggregate the features of different levels, We use a channel attention module to transfer more important information between different layers before the information fusion of different layers of the neck network. We conduct experiments on standard dataset. The results show that YoloX-Lite achieves faster inference speed and higher detection accuracy than YoloX.

2. Based on the static object detection model YoloX-Lite, this paper proposes a video object detection model called Yolo-GLA that can aggregate global and local information. We design a proposal region selection module, which can select a group of proposal regions that are most likely to have an object from the results of YoloX-Lite. This module eliminates the interference of a large number of low-quality results on subsequent information aggregation, while reducing the computational load. We design a new information aggregation method, which aggregates the appearance information of the global object and the precise position information of the local object, which not only expands the scope of information aggregation, but also makes the feature representation of the current frame richer. We experiment on the public ImageNet VID dataset and found that combining global and local information at the same time can significantly improve the detection accuracy of the entire model.

3. Based on the video target detection model Yolo-GLA, we design a complete video target detection system. The system consists of two modules, which are deployed on the center side and the edge side respectively. The center-side module includes functions such as data labeling, data analysis, and one-click training. The edge side module integrates the Yolo-GLA algorithm, which can be deployed on edge devices in the wild environment to achieve all-weather target detection. The whole system realizes the complete process from data labeling to model training, and then to model deployment, which helps video object detection tasks to be implemented smoothly, saves costs, and improves people's work efficiency.

keywords: Static Object Detection, Structural Re-parameterization, Video Object Detection, Global-Local Information Aggregation

目　　录

插图清单

附表清单

第一章　绪论

1.1　研究背景及意义

近些年来，由于各类计算设备不断改进，算力得到明显提高，第二次人工智能（Artificial Intelligence，AI）浪潮中的大量算法得以实现，促使机器学习(Machine Learning) 尤其是深度学习（Deep Learning）领域内的各个研究方向蓬勃发展。为了解决过去神经网络（Neural Network）训练过程中梯度消失导致难以训练的问题，2006 年 Hinton 等人[1] 采用贪婪逐层预训练的方式训练了一个深度信念网络（Deep Belief Network），使得深度学习重新崛起，成为了第三次人工智能浪潮的开端。在这股深度学习的研究热潮中，Krizhevsky 等人[2] 在 2012 年设计了一个多层卷积神经网络 AlexNet，在图像识别竞赛 ILSVRC[3] 上大幅领先传统方法，展现了深度学习相较于传统方法的卓越性能，也进一步推动研究人员挖掘深度学习在计算机视觉（Computer Vision，CV）[4–6]、自然语言处理（Neural Language Process，NLP）[7, 8]、数据挖掘（Data Mining，DM）[9, 10] 等领域的可能性。深度学习不仅在学术界欣欣向荣，也逐渐走进我们的生活中。Google 旗下的 DeepMind 部门利用强化学习（Reinforcement Learning）技术开发了一款围棋机器人 AlphaGo[11]，在 2016 年胜过了当时的围棋大师李世石，让整个社会都深刻认识到了人工智能在实际应用中的巨大潜力，各个领域都在积极探索如何利用人工智能技术扩大产品影响力，推动了人工智能方法的落地。

计算机视觉技术是生活中应用十分广泛的人工智能技术之一。从门禁中的人脸识别[12]，到车道线检测，再到漫画脸生成[13]，计算机视觉给我们的生活带来了显而易见的便利，越来越多的科研人员也投入到了视觉任务的研究，尤其是目标检测（Object Detection）。目标检测是计算机视觉领域中的基础任务之一，其目的是识别出一张图像中感兴趣物体的类别，并且标出物体在图像中的具体位置。而视频目标检测（Video Object Detection）则是需要识别出一段视频中的每一帧图像中的目标类别，并且定位目标所在的位置。在我国，越来越多的场景需要通过监控视频发现并定位特定目标，从而进一步分析目标的行为和状态，达到解决社会问题或者辅助科学研究的目的。然而当前实际情况是摄像

头拍摄到实际画面，经过视频编解码传递到边缘设备或者远端服务中心，再由专业人员人工观看并查找特定的目标。由于监控视频范围广、内容多等因素，人工查找耗时耗力，效率有限。利用视频目标检测技术代替人工查找，全天候不间断搜寻视频中的目标，极大地降低了监测成本，同时也扩大了监控范围。

视频目标检测具有广泛的实际应用价值，可以应用在自动驾驶（Autonomous Driving）行业，例如架设在车辆上的摄像设备实时拍摄周围的环境，目标检测算法利用拍摄到的视频快速准确识别出车身周围的场景以及当前路况中存在的障碍物、红绿灯等目标，同时确定目标的位置，帮助驾驶员及时修改行车路线；视频目标检测还具有辅助自然科学调研的价值，环境保护人员为了了解环境保护区内野生动物的现状，常常需要深入保护区拍摄野生动物，随后人工分类拍摄资料，效率低下，通过在环境保护区中安装摄像头，并装载视频目标检测算法，全天候自动分类和定位视频中的野生动物，帮助环境保护人员实时掌握动物状态；视频目标检测还是其他视觉任务的基础，例如目标追踪（Target Tracking）需要先定位到视频中每个目标的位置，然后计算出每个目标在视频中的行动轨迹，对视频目标检测的研究能够推动目标追踪任务发展。考虑到视频目标检测的实际意义和科研意义，对视频目标检测任务的研究是十分有必要的。

1.2 研究现状及挑战

1.2.1 研究现状

按照视频目标检测的发展历史，主流的研究方法可以分为以下三类：传统方法，早期的目标检测通过人工设计复杂的特征表示以及各种加速技巧有效利用计算资源；基于深度学习的静态目标检测方法，这类方法利用了神经网络自动提取特征的优势，通过大规模数据训练出了远超传统检测器的效果；基于深度学习的视频目标检测方法，这类研究主要目的在于通过深度学习有效利用视频中相邻帧天然存在的时间和空间相关性信息，聚合相邻帧的特征以提高检测精度或者扩散关键帧（Key Frame）的特征以提高检测速度。下面我们将从这三类方法出发，介绍目标检测的研究现状。

传统方法需要人工设计图像的特征表示，例如哈尔（Haar）特征[14] 和方向梯度直方图（Histogram of Oriented Gradients，HOG）特征[15]，同时使用不同大小的滑动窗口在图像上扫描以提取不同尺度的特征。传统方法分为基于统计和

基于知识两种。基于统计的方法[15–17]将目标整体看作一个像素矩阵，通过相似度来判断目标是否存在。基于知识的方法则是借鉴了“分而治之”的思想，例如 DMP（Deformable Part-based Model）[18]，利用先验知识将目标看作是不同部位的组合，使用多个辅助分类器判断目标不同部位是否存在和一个根分类器判断目标整体是否存在。通过多级的分类器，DMP 成为了传统方法的代表之作，连续三届获得 PASCAL VOC[19] 目标检测大赛的冠军。人工设计的特征比较适合于某个特定邻域，其表达能力也依赖于设计者对该领域的熟悉程度，并不能在所有数据上都有良好的表现。滑动窗口的设计增加了特征生成的时间，也一并生成了大量冗余的低质量特征表示，阻碍了检测模型的落地。

随着神经网络（Neural Network）开始在图像识别（Image Classification）任务中大放光彩，基于神经网络的目标检测模型也随之出现。这类以神经网络为基础的方法可以针对目标特点自动提取语义丰富的特征表示，相较于人工设计特征的方法泛化能力和表达能力都有明显提升。

早期静态目标检测根据方法的特点可以更细致地分为两阶段方法（Two-Stage）和单阶段（One-Stage）方法。两阶段目标检测代表性的方法包括 R-CNN[20]、Fast R-CNN[21]、SPPNet[22]、Faster R-CNN[23] 等。这些方法将检测过程划分为两个阶段，第一个阶段使用单独的网络生成候选框（Region Proposal）集合，用于表示目标可能的类别和出现的区域。第二个阶段使用另外一个网络在候选框集合中继续微调，得到更加精确的结果。两阶段的目标检测类似于 DMP，通过分级检测提高了检测效果，相对而言检测时间也有所增加。尽管 Faster R-CNN 使用候选区生成网络（Region Proposal Network，RPN）代替 select search[24] 算法来加快获取候选区集合，仍然不能实现实时处理视频数据。不仅如此，两阶段方法受限于候选框和池化层的特点，对于小目标检测并不友好。

单阶段目标检测方法以基于 SSD（Single Shot MultiBox Detector）[25–27] 和基于 YOLO（You Only Look Once）[28–33] 为主，省略了候选区生成这一步骤，直接端到端（End-to-End）地预测目标的类别和位置信息。很明显，单阶段目标检测方法相较于两阶段方法速度有了大幅提升，更加适合于实际任务。为了能提高对不同尺寸目标的检测效果，单阶段方法大多利用了特征金字塔（Feature Pyramid Network，FPN）[34] 及其变体[35, 36] 融合深层的语义信息和浅层的精确位置信息。

无论是单阶段目标检测，还是两阶段目标检测，都需要依赖于预先设计

好的锚框（Anchor），锚框的准确性也严重影响了模型的准确性。为了能摆脱锚框的限制，近年来逐渐出现了一些不基于锚框（Anchor-free）的目标检测方法[37, 38]，通过计算一个概率热力图来预测可能的关键点位置，然后预测关键点覆盖的区间内是否存在目标，目标的类型以及具体位置。当前不基于锚框的方法已经展现出了引人遐想的潜力，能够结合两阶段方法的检测精度和单阶段方法的推理速度，成为了主流的研究方向。

传统方法和静态目标检测方法均将视频数据视为无上下文关系的图像数据的集合，忽视了视频数据中存在的时序信息。视频目标检测着眼于更好的利用视频中存在的时序和空间信息，向着消除冗余特征以提高检测速度和聚合相关特征以提高检测精度两个方向发展。视频目标检测主要分为基于光流场（Optical Flow Field）、基于 LSTM（Long Short-Term Mermory）、基于追踪和基于注意力机制（Attention）的方法。

基于光流场的方法[39–43] 对于视频中的关键帧和非关键帧有着不同的处理。关键帧会通过特征提取网络生成特征图，同时利用光流场聚合其他关键帧的信息，生成新的特征编码，非关键帧的特征表示则是利用光流场由关键帧传播得到。考虑到非关键帧在视频数据中占据更大的比例，非关键帧的处理方式可以节省提取冗余特征的时间。这类方法的性能取决于光流场的效果，如何生成更加准确的光流场以及缩短光流场生成的时间是这类方法的重点研究方向。

为了构建端到端的视频目标检测网络，基于 LSTM 的方法[44–46] 将时序信息直接编码为隐变量，输入图像一方面会从隐变量中获取前文信息，另一方面也会编码生成包含当前帧信息的、新的隐变量传递下去。同一时期，基于追踪的方法[47–49] 在静态目标检测器的基础上新增了一个追踪器，用于生成下一帧候选框可能出现的位置，辅助检测器工作，形成了一个多任务网络。基于追踪的方法同时学习视频目标检测和目标追踪，利用目标追踪的特点来反哺目标检测的性能。基于 LSTM 的方法缺点在于隐变量无法充分表征视频的时序信息，而基于追踪的方法则是受限于信息聚合的范围，只能聚合短时范围内的目标信息。

前述的几类视频目标检测大多是在整个特征图上聚合视频的时序和空间信息，基于注意力机制的方法将信息聚合范围缩小到局部候选框附近，只增强目标自身的信息。基于注意力机制的方法[50–54] 使用余弦相似度或者自注意力机制（Self-Attention）计算来自不同帧的候选框之间的相似度分数，然后加权求和得到新的特征表示。这种在候选框层面上聚合时序信息的方法，可以不用顾虑目标形变过大造成特征难以对齐的问题，扩大了信息聚合的范围。使用候选框

聚合信息的另一种潜在优势是不用考虑背景信息变化的影响，排除了大量的冗余信息，同时节省了注意力机制的计算成本。基于注意力机制的视频目标检测已经成为了视频目标检测的主流研究方向，具有广阔的研究空间。

1.2.2 研究挑战

视频目标检测任务是由静态目标检测任务发展而来，相较于静态目标检测，视频目标检测的输入数据从静态图像转换成了视频数据，需要找出每一帧图像中的感兴趣目标。由于实际监控视频的特点，视频目标检测任务存在以下几个技术难点：

1. 视频标注数据难以获得。视频标注数据要求标注出视频中每一帧图像上所有目标在图像中的位置以及目标所属的类别。监控视频每秒至少包含 25 帧图像，一段 10 秒的视频数据相较于单张静态图像数据需要 250 倍的标注成本。因此在实际应用场景中，往往能采集到的是大量的图像标注数据以及少量的视频标注数据，这也就导致了静态目标检测算法比视频目标检测算法更多地应用于监控视频目标检测任务中。

2. 视频中存在低质量图像帧。当前监控视频主要以视频流的形式在网络中传输，由于视频流在传输和编解码的过程中存在损耗，会出现图像失真的现象。同时监控视频中的目标处于不断运动的状态，容易出现运动模糊、视频散焦、目标遮挡和姿态奇异等情况。这些原因导致解析得到的图像质量很低，即使人工也很难通过单张图片分辨出目标，给目标检测带来很大的压力。

3. 推理速度和检测精度难以平衡。现有的目标检测方法通过增加模型的深度和组件的数量，在很多数据集上都表现的十分优秀。过高的参数量导致模型推理速度十分缓慢，在高算力的设备上仍然不能实时处理视频数据，而降低模型参数量又会使得检测效果不满足要求。受限于实际的生产环境，检测模型需要尽量提高推理速度的同时维持理想的检测精度。

当前监控视频目标检测主要采用静态目标检测算法和视频目标检测算法，其中静态目标检测算法将视频中的每一帧视为没有上下文信息的图像，在监控视频中容易受到图像质量、运动模糊和目标遮挡等的影响，导致检测效果并不理想。视频目标检测算法有效利用了前后帧之间的相关性信息，在检测精度上有了明显提升。但是由于视频标注数据过于高昂的成本，以及现有高精度的视频目标检测模型参数量过大，很难应用于实际目标检测任务中。为了解决上述问题，本文设计了一个高效快速的静态目标检测模型，并在此的基础上聚合全

局和局部时序信息，提出了一个新颖的视频目标检测模型，利用少量的视频标注数据，就能够更好地平衡推理速度和检测精度。

1.3 研究内容与贡献

本文基于目标检测问题展开研究，在分析了现有的静态目标检测网络缺点之后，提出了一个快速的目标检测模型，该模型通过结构重参化等技术实现了在不影响模型精度的前提下，加快模型的推理速度。本文在前述静态目标检测模型的基础上，设计了一个能够更高效聚合时序信息的视频目标检测模型，使用部分视频标注数据即可提高对低质量图像的检测效果。最后本文还设计了一款视频目标检测软件，支持数据标注、模型训练和边缘部署等功能，使得本文提出的算法可以直接应用于社会生产中。本文的主要研究如下：

1. 本文以静态目标检测模型 YoloX 为基础，设计了一个轻量的模型 YoloX-Lite。该模型修改了 Bottleneck 结构的算子顺序，使得整个结构只有最后一个算子是非线性的激活函数，命名为 Rep-Bottleneck。YoloX-Lite 将训练和推理解耦，训练阶段使用原始 Rep-Bottlneck 多分支结构，推理阶段利用结构重参化技术将多个线性算子合并为一个卷积操作，提高了模型整体的检测速度，并且不会影响模型检测精度。在颈部网络中，YoloX-Lite 重新设计了数据流向，并制定了算子使用的规则。为了能更好的聚合不同层次的特征，YoloX-Lite 使用通道注意力机制让更加重要的信息在不同层之间传递。本文在标准数据集上进行实验，结果表明，YoloX-Lite 相较于 YoloX 达到了实现了更快的推理速度和更高的检测准确率。在和主流的目标检测模型进行对比后，YoloX-Lite 也具有速度优势。最后本文针对模型中的改进之处进行了消融实验，验证了其有效性。

2. 本文在静态目标检测模型 YoloX-Lite 的基础上，聚合全局和局部信息，设计了一个视频目标检测模型（Enhanced Yolo with Global and Local Information Aggregation，Yolo-GLA）。该模型通过候选区选择模块，从 YoloX-Lite 的结果中，挑选出最有可能存在目标的一组结果，排除大量的低质量结果对后续信息聚合的干扰，同时降低了计算量。根据结果中的位置信息，Yolo-GLA 结合 YoloX-Lite 颈部网络生成的特征生成对应的候选区特征，保存了目标丰富的特征信息。Yolo-GLA 将特征聚合分为多个阶段，第一个阶段聚合全局候选框和局部候选框之间的外观特征，帮助目标可以被划分到正确的类别；第二个阶段聚合局部特征和当前帧特征的外观和位置信息，保证了目标可以被精确定位。

本文在公开的 ImageNet VID[3] 数据集上实验发现，和其他主流方法相比，同时结合全局和局部信息，能提高整个模型的检测精度，同时维持了可靠的推理速度。本文最后对 Yolo-GLA 中部分超参数展开敏感性实验，验证了各自的有效性。

3. 在视频目标检测模型 Yolo-GLA 的基础上，我们设计了一个完整的视频目标检测软件。该软件分为中心侧和边缘侧，中心侧包含数据标注，数据分析和一键训练等功能。边缘侧使用 libtorch 等框架，集成了 Yolo-GLA 算法，可以部署在野外环境中的边缘设备，实现全天候目标检测。整个软件实现了从数据标注到模型训练，再到模型部署的完整流程，帮助视频目标检测任务顺利落地，节省成本，提高人们的工作效率。

1.4 本文组织结构

本文分为六个章节，整体组织结构如图 1-1，各章节主要内容如下：

第一章为绪论，简要介绍视频目标检测任务的背景和实际意义，随后陈述了视频目标检测主流的几类方法以及分析了各自的优缺点，引出了视频目标检测任务现存的研究难点，最后简述为了解决这些问题我们所作的工作。

第二章为相关工作部分，本文介绍了视频目标检测任务的特点，并对其进行形式化定义。随后，本文分别从静态目标检测和视频目标检测两个方向出发，依次介绍各自领域中主流研究方法的思想，并分析了这些方法的优缺点。

第三章在 YoloX 的基础上，提出了一个基于结构重参化的静态目标检测方法 YoloX-Lite，详细描述了 YoloX-Lite 各模块的细节。通过对比实验表明 YoloX-Lite 相较于其他方法达到精度与速度之间更合适的平衡，同时还进行消融实验验证了各个模块的作用。

第四章以第三章提出的 YoloX-Lite 模型为基础，聚合全局-局部特征，搭建了一个视频目标检测模型 Yolo-GLA。在该章节中，本文介绍了 Yolo-GLA 的设计思想和实现细节，最后通过实验验证在标准数据集上 Yolo-GLA 模型的实际效果。

第五章利用本文提出的视频目标检测算法 Yolo-GLA，搭建了一个鸟类监控视频目标检测系统。该系统支持数据处理，迭代训练视频目标检测模型和实时检测视频流数据等功能。该系统证明了本文提出的算法的有效性和实用性。

第六章为全文的总结和展望，总结了本文的内容和做出的贡献，并分析了

本文提出的方法的不足之处，提出了下一步的改进思路以及展望了该领域未来的发展方向。

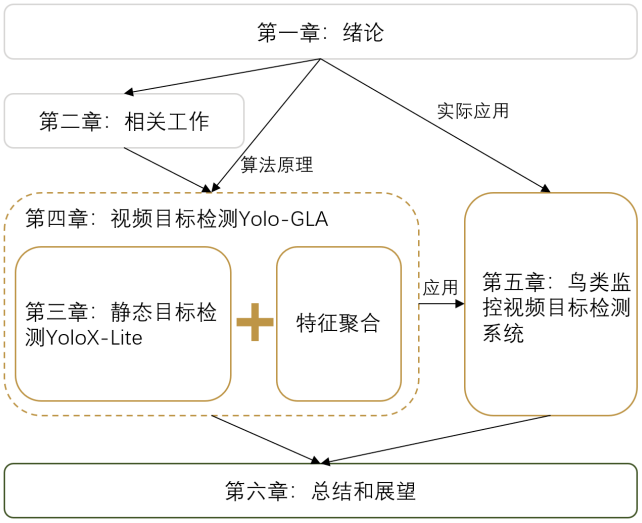

图 1-1: 本文整体的组织结构

第二章　相关工作

本章首先介绍了视频目标检测任务，包括监控视频组成原理和目标检测任务的定义。随后从静态目标检测和视频目标检测两类方法出发，介绍了一些重要的目标检测方法。

2.1　视频目标检测任务

2.1.1　视频组成原理

部署在实际环境中的摄像头采集得到模拟视频信号，然后将其编码和压缩成数字视频数据，生成监控视频，传输到网络中，通常帧率为 25fps（frames per second）。

考虑到视频信号中存在大量重复的信息，人们在发送端对数字视频数据进行编码，大大压缩了视频数据的体积。编码器会将视频划分成多张相邻帧组成的一组图像（Group of Pictures, GOP），解码器则会在接收阶段将 GOP 解码并渲染播放。每个 GOP 由一张 I 帧和多张 P 帧组成，其中 I 帧包含一张图像的所有信息，可以独立被解码，也称为关键帧。P 帧需要依赖前一帧的信息才能完成编码，表示了当前帧与前面的 I 帧或者 P 帧之间的差异。因此相比于完整的图像，P 帧占用的存储空间更少，十分依赖前一帧和最近的关键帧的准确性，对错误传输十分敏感。

监控视频使用到的传输协议通常为用户数据报协议（User Datagram Protocol, UDP），UDP 协议并不保证数据的准确到达，在复杂的实际网络中，监控视频传输时常会出现丢包的情况，这也导致了接收段解码得到的图像会出现像素缺失。

此外，摄像头下目标的运动、环境中的天气等因素会造成目标在视频中部分帧会呈现出运动模糊（Motion Blur）、遮挡（Occlusion）、奇异姿态（Pose Changes）和视频失焦（Video Defocus）四类现象，给后续处理带来困难，如图 2-1所示：

1. 运动模糊。摄像头在拍摄画面时，需要维持一段时间的曝光，传感器将一段时间光信号转换为电信号，最终形成完整的画面。如果在曝光时间内目标处于高速运动的状态，就会导致目标处于多个位置的光信号进入摄像头，形成了多个状态的叠加，也就是运动模糊。

2. 遮挡。目标和摄像头之间存在障碍物，使得在摄像头中只呈现出目标部分外观信息。

3. 姿态奇异。目标在运动过程中呈现出来的和正常目标差异较大的姿态，例如行人蹲下看报纸。

4. 视频失焦。暴露在环境中的摄像头会因为大风造成相机摆动，或者会受到阴天或者大雾等能见度低的情况，导致相机自动对焦困难，目标在视频中呈现模糊的状态。

运动模糊 遮挡 姿态奇异 视频失焦

图 2-1: 四类低质量视频帧

2.1.2 目标检测任务

目标检测任务是计算机视觉领域中的基本任务之一，其目的简而言之是解决两个问题，图像内的物体是什么，以及物体在哪儿。目标检测根据待检测目标的类型可以分为小目标检测、遮挡目标检测等，根据输入的数据类型分为静态目标检测和视频目标检测。目标检测任务的结构如图2-2所示，$(x_{i-t}\dots x_{i+t})$ 表示输入一组在 $(i-t,i+t)$ 时间范围内的图像，F_M 表示目标检测算法，$cls_{i,j}$ 表示模型推理得到第 i 帧图像中的第 j 个目标的类别，$box_{i,j}$ 表示模型推理得到第 i 帧的第 j 个目标的位置，可以形式化定义为:

$$F_M(x_{i-t}\dots x_i\dots x_{i+t})\rightarrow\{(cls_{i,1},box_{i,1})\dots(cls_{i,j},box_{i,j})\}. \tag{2-1}$$

对于静态目标检测，$t=0$，对于视频目标检测 t 通常大于 0。输入一张图像后，模型会推理得到一组分类和定位结果，表示图中所有目标的类别和位置信息。

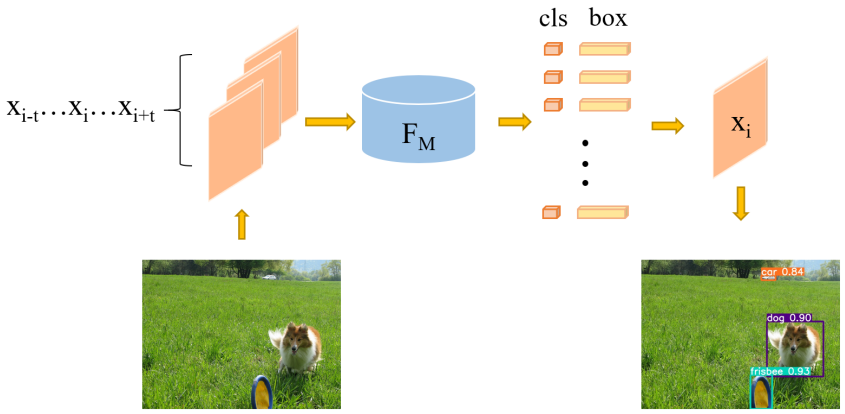

图 2-2: 目标检测任务结构

2.2 静态目标检测方法

静态目标检测算法将输入图像视为无上下文关系的单张图片，仅仅将可利用的信息局限于单张图片，以此来检测出感兴趣的目标和目标所在的位置。考虑到实际场景下数据采集和部署的难易程度，静态目标检测反而是更常用于解决现实目标检测问题。本节将介绍两阶段、单阶段和无锚框目标检测中具有代表性的算法。

2.2.1 Faster R-CNN

两阶段目标检测算法起源于 R-CNN[20]，为了提高检测精度，这类算法将检测过程分为生成候选框和在候选框基础上检测两个步骤，诞生了一系列相关算法[21–23, 34, 55, 56]，Faster R-CNN[23] 就是其中集大成者。如图 2-3所示，Faster R-CNN 分为两个子网络：候选区生成网络和检测网络（Detector），两个子网络共用同一个骨干网络（Backbone）。

骨干网络通常由图像分类网络除去最后的全连接层构成，在 ImageNet 数据集[3] 上预训练过后，可以很好地提取输入图像的特征表示。Faster R-CNN 预先设计好了一组特定比例的锚框，这些锚框分布于骨干网络提取到的特征图上每一个点。随后候选区生成网络将特征图像输入到两个分支中，第一个分支用于调整预设锚框的位置，第二个分支用于预测锚框对应位置是前景还是背景，两个分支的预测结果会筛选出更可能是前景的锚框，输出到检测网络。

检测网络采用 ROI Pooling 技术，将任意大小的特征图划分成 $n \times n$ 大小的网格，每个网格内的特征进行最大池化计算，这样对于任意大小的输入特

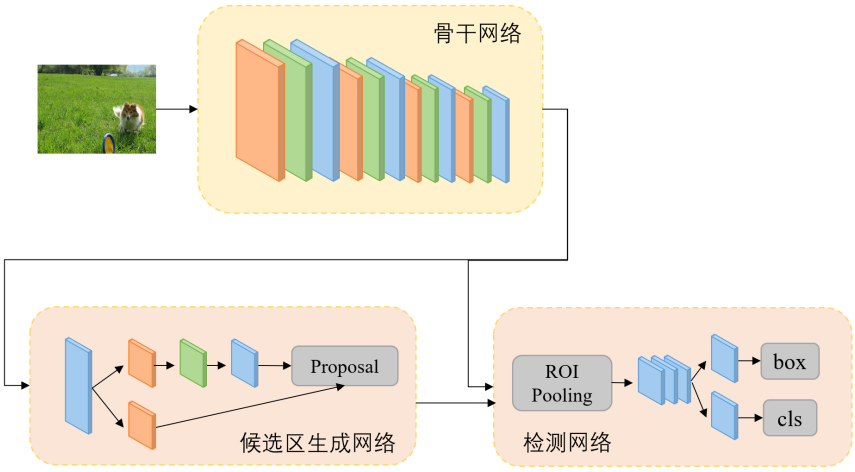

图 2-3: Faster R-CNN 网络结构

征，都能得到同样尺寸的输出特征，最后通过分类和回归分支预测目标类别和坐标。Faster R-CNN 的训练过程类似于一种迭代训练，交替训练候选区生成网络和检测网络，直到网络检测效果不再提升。Faster R-CNN 在 PASCAL VOC、Micrsoft COCO 等目标检测数据集中都达到了当时最好的准确度，直到今天仍然影响着很多研究者。

这种两阶段目标检测方法将检测过程分阶段完成，逐步细化的思想，能收获更高的检测精度，同时也严重影响了模型的推理速度。不仅如此，这类方法对不同分辨率大小的目标一视同仁，放在同一个特征图上进行检测。然而特征图下采样率过大会导致小目标在特征图上失去了语义信息，特征图下采样率太小又会导致其表达能力不够丰富，影响检测精度，两阶段目标检测方法还不能有效处理小目标的检测。

2.2.2 YoloX

两阶段目标检测固然精度很高，但是交替训练的不便之处以及检测速度过慢，影响了这类方法在实际场景中的应用。相较于两阶段目标检测，工业场景下明显更偏爱以 Yolo[28] 为代表的单阶段目标检测算法。Yolo 省略了候选框生成这一步骤，相反，将图片划分成 $n \times n$ 大小的网格，如图 2-4[28]，只有物体中心所在的网格负责预测目标的类别和位置。同时，yolo 预先为每个网格设置了一组特定大小的锚框，模型预测目标位置时，只需要预测网格基础上的调整值即可，这样就可以将变化剧烈的锚框坐标转化为小范围回归问题。

2020 年提出的 YoloV4[31] 将 Yolo 系列目标检测算法整理成特有的框架，由

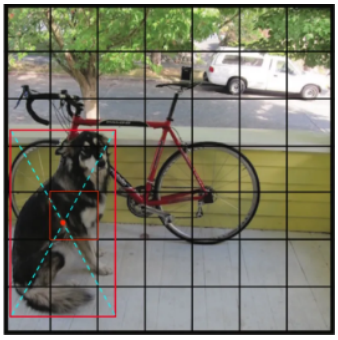

图 2-4: Yolo 输入图像网格划分

骨干网络、颈部网络（Neck）和头部网络（Head）组成，分别用于特征提取、特征融合以及最后的预测。统一的框架让研究人员能快速高效利用上领域内最新的模型结构和训练策略，YoloX[32] 就是在此基础上对头部网络和训练方式做出修改。如图 2-5，YoloX 将传统的头部网络解耦，划分为分类分支和回归分支，每条分支都包含 2 个 3×3 的卷积用于解耦，分类分支用于预测目标所属类别（cls），回归分支用于预测目标属于前景还是背景（obj），以及目标的位置信息（box）。解耦头的设计是考虑到了分类任务和回归任务存在空间不匹配

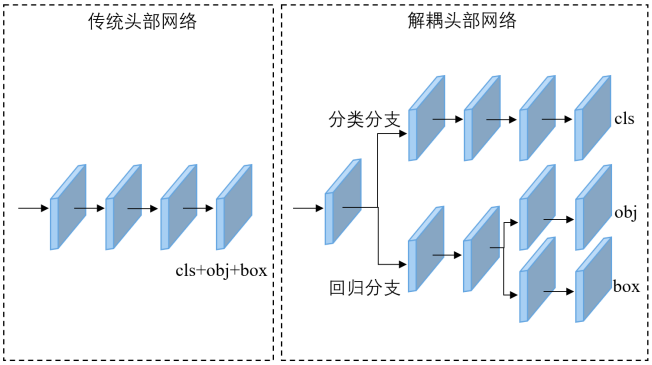

图 2-5: 传统头部网络和解耦头部网络

的问题，即两个任务聚焦目标的区域不同，分类任务更加关注提取的特征和已有的哪一类最为接近，而定位更加关注预测值和真实位置之间坐标的差异。解耦头的出现，使得 YoloX 可以直接预测目标的坐标位置，不用在预设的锚框上进行调整，整个网络也不再依赖于预先设置锚框的准确性，成为了不基于锚框的方法。

Yolo 框架强大的兼容性和实用性吸引了很多人参与其中，只需要针对性修改框架中的组件，就可以改进整个网络，极大地推动了 YOLO 系列算法的发展，成为当前工业界最受欢迎的目标检测方法。

2.2.3 CenterNet

前面两个小节中提到的两阶段目标检测和单阶段目标检测算法中大多方法需要通过聚类分析预先设计好一组锚框，然后网络在锚框的基础上对其进行修正，这也使得锚框的准确性严重影响了网络的推理效果。有研究人员开始探索不基于锚框的目标检测方法，直接预测目标所在的位置。在 2019 年提出的不基于锚框的目标检测方法 CenterNet[38] 达到了单阶段目标检测方法的检测精度和速度，吸引了很多人的注意力。CenterNet 的网络结构相较于 Faster R-CNN 和 YoloV4 更加简单，仅需要将骨干网络中提取到的特征进行一系列反卷积即可输入到预测网络中，并没有复杂的设计。值得注意的是 CenterNet 的边界框损失函数 $Loss$ 和其他方法有明显区别，具体为:

$$Loss = L_k + \lambda_{size}L_{size} + \lambda_{off}L_{off}, \tag{2-2}$$

L_k 为目标中心点损失，用于惩罚当前位置是否是目标中心点。L_{size} 为目标大小损失，用于惩罚预测值和目标大小的差距，采用了 L1 损失。L_{off} 为目标中心点偏置损失，考虑到当前特征层的分辨率和输入图像不一致，会导致预测中心点可能存在偏移的情况，同样采用 L1 损失函数。λ_{size} 和 λ_{off} 均为超参数。CenterNet 会将训练图片的目标中心位置采用高斯核，转换成一个概率分布:

$$Y_{xyc} = \exp(-\frac{(x-\widetilde{p}_x)^2 + (y-\widetilde{p}_y)^2}{2 * \sigma_p^2}), \tag{2-3}$$

Y_{xyc} 表示位置 (x, y, c) 为中心点的概率，这样的处理使得离中心点越近的点具有更高的价值。L_k 借鉴了 Focal Loss[57] 的思想，将中心点的预测视为一个类别不平衡问题:

$$L_k = -\frac{1}{N}\sum_{xyc}\begin{cases}(1-\hat{Y}_{xyc})^{\alpha}\log(\hat{Y}_{xyc}), if Y_{xyc}=1\\(1-Y_{xyc})^{\beta}(\hat{Y}_{xyc})^{\alpha}\log(\hat{Y}_{xyc}), otherwise,\end{cases} \tag{2-4}$$

N 为图中关键点的数量，α 和 β 是超参数，$\hat{Y}_{xyc}$ 是 CenterNet 预测当前位置为中心点的概率，L_k 缓解正负样本不平衡的同时，使得模型更加关注困难样本的训练。

不基于锚框的方法不会受到预先设置锚框的准确性干扰，模型结构也更为简单，具有很大的研究价值，但是在部署简便性上还远不如单阶段目标检测，仍存在进步空间。

2.3 视频目标检测方法

为了能更好地利用视频中存在的时序和空间信息，辅助检测低质量图像帧中的目标，视频目标检测从基于光流场、基于长短记忆网络、基于追踪和基于注意力机制等角度切入，本节会选择其中有代表性的方法介绍。

2.3.1 FGFA

由于目标运动的原因，同一实例对象的特征通常在多帧之间空间上不是对齐的，简单叠加聚合特征甚至可能降低性能。视频目标检测方法从检测框层级和完整特征图层级聚合视频的时序信息和空间信息，检测框层级的聚合方法不能实现端到端的训练，同时对空间信息利用也有限。FGFA[40] 利用了特征层级的时间和空间信息相关性，通过对运动路径上的邻近特征进行聚合，从而提高视频目标检测的精度。具体表现为，FGFA 在学习过程中对运动进行建模，计算附近帧和参考帧之间的运动：

$$M_{i\mapsto j} = F(x_i, x_j), \tag{2-5}$$

其中 x_i 和 x_j 分别是当前帧和参考帧，$F(\cdot,\cdot)$ 用于计算两帧之间的光流差异，得到当前帧到参考帧的光流变化 $M_{i\mapsto j}$。第二步根据光流信息将参考帧提取到的特征 f_j 映射到当前帧，获取映射变换后的特征 $f_{j\mapsto i}$:

$$f_{j\mapsto i} = W(f_i, M_{i\mapsto j}), \tag{2-6}$$

在特征聚合阶段，FGFA 利用余弦相似度计算映射后的特征和参考帧自身提取到的特征的关联程度，记为权重值 $w_{j\mapsto i}$，用于加权求和多个特征图，这样就得到理想的融合了视频时序和空间信息的特征图 $\bar{f}_i$：

$$\begin{aligned} w_{j\mapsto i} &= exp(\frac{f_{j\mapsto i}\cdot f_i}{|f_{j\mapsto i}||f_i|}), \\ \bar{f}_i &= \sum_{j=i-K}^{i+K} w_{j\mapsto i} f_{j\mapsto i}. \end{aligned} \tag{2-7}$$

将聚合好的特征反馈给检测网络 N_{det}，生成最后的预测结果 y_i：

$$y_i = N_{det}(\bar{f}_i). \tag{2-8}$$

FEGA 在当时取得了最高的检测精度，一举赢下了 ImageNet VID 2017 视频目标检测比赛，启发了后续大量的方法，直到现在也在被很多研究人员重点关注。

2.3.2 LSTM-SSD

前述基于光流场的方法严重依赖于光流场网络的估计效果，计算光流场也是相当耗时的一步，不适合部署在实际应用中。Liu 等人[45] 在静态目标检测方法的基础上引入了长期短期记忆网络[58]，直接将时间上下文并入到特征级别，形成了端到端的网络 LSTM-SSD。长短记忆网络 LSTM 已经成功应用于涉及很多顺序数据的任务中，通过多个门状态，允许 LSTM 对空间和时间信息进行编码，具体定义为：

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\ C_t &= f_t * C_{t-1} + i_t * \tanh(W_C \cdot [h_{t-1}, x_t] + b_C), \\ h_t &= o_t * \tanh(C_t), \end{aligned} \tag{2-9}$$

其中 f_t、i_t 和 o_t 分别为遗忘门，输入门和输出门，C_t 为隐变量，h_t 为输出值。LSTM-SSD 将目标检测 SSD 网络[25] 提取得到的特征视为 LSTM 的输入，相邻帧之间的时序信息通过隐变量 C_t 完成传递，视频目标检测任务可以被形式化为:

$$F(X_t, C_{t-1}) = (D_t, C_t), \tag{2-10}$$

其中 X_t 为 SSD 提取到的特征图，D_t 为聚合时序信息和空间信息的特征图，隐变量 C_t 携带了视频中的时序信息和空间信息，在整个序列中不断传递和更新。LSTM-SSD 为了能细化聚合的特征，堆叠了多个 LSTM 层，但是 LSTM 的计算成本过高以至于整个网络的推理时间难以接受。因此 Liu 等人使用深度可分离卷积[59]（Depthwise and Pointwise Convolution）代替了 SSD 和 LSTM 中所有的普通卷积操作，使得整个网络的推理时间减少了 8 到 9 倍。不仅如此，Liu 等人还观察到 LSTM-SSD 中 LSTM 层的输出通道数远少于输入的通道数，设计了

一个降通道维度的映射 Bottleneck-LSTM：

$$b_t = \sigma(W_b \cdot [h_{t-1}, x_t]). \tag{2-11}$$

用来代替遗忘门、输入门和输出门，大幅降低 LSTM 层的计算量，表达效果也不弱于传统的 LSTM。LSTM-SSD 结合了快速的静态图像目标检测方法和长期短期记忆网络，细化和传播了帧之间的特征映射，从而实现了时态感知，能够在低功耗移动设备和嵌入式设备上实时运行。

2.3.3 TCEA

基于光流的方法和利用长期短期记忆网络的方法，局限于利用局部帧来聚合时序信息。然而相邻几帧之间的信息十分冗余，对提升检测器效果有限，增加聚合帧的范围又会带来计算负担。He 等人[60] 受到空洞卷积[61]（Dilated Convolution）的启发提出 TCEA 网络，增加附近帧的采样间隔，可以有效增加时间感受野，帮助融合更多的有用信息，同时没有增加计算量。TCEA 根据目标的运动速度提出了一个自适应的步长预测网络，该网络以相邻两帧图像作为输入，计算出一个变化分数，分数值越高，表明物体运动越慢。随后利用一个预先设计好的标准计算采样间隔，速度越快，采样间隔越小；速度越慢，采样间隔越大，保证了采样得到的当前帧和参考帧之间存在丰富的时序信息。

前文提到了 TCEA 的输入帧采样间隔可能会很大，同一个物体在几张输入图像中的位置会存在明显差异。直接使用提取到的特征图聚合时序信息，容易生成大量错误分类或者定位不准确的结果。使用光流场和长期短期记忆网络已经无法满足对齐特征的空间信息，TCEA 借鉴了 STSN[62] 中的特征对齐的方法，提出了 DeformAlign 网络，结构如图 2-6[60] 所示。时间 t 和时间 i 提取到的

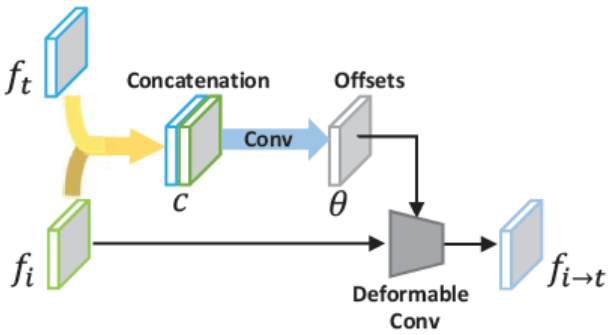

图 2-6: DeformAligned 结构

特征图会采用两个 3×3 的卷积和一个 1×1 的卷积生成一个偏移参数 Θ:

$$\begin{aligned}\Theta &= f_{\Theta}(f_i, f_t) \\ &= \{\delta p_n | n = 1, \ldots, |R|\},\end{aligned} \tag{2-12}$$

其中 p 为特征图上的像素点，R 为一个 3×3 的矩阵，记录了像素点的运动。参考帧特征 f_i 和偏移参数 Θ 输入到可变形卷积（Deformable Convolution[63]）中就能学习到一个特征对齐后的特征 $f_{i\rightarrow t}$，用于下一步特征聚合。

TCEA 在特征聚合阶段参考了 CBAM（Convolutional Block Attention Module）[64] 的思想，将特征聚合过程分为聚合时序信息和聚合空间位置信息两个阶段。第一个阶段的时序注意力模块用于计算编码空间内多帧之间的相似度，使得模型知道哪一个时刻的特征应该投入更多的注意力。第二个阶段则是将时序注意力模块计算好的特征融合在一起，得到一个聚合了时序信息的特征图，用于下一步计算空间注意力机制。第三个阶段主要用于关注特征图中哪些区域是更重要的，在计算空间注意力权重的时候，采用平均池化和最大池化来增大感受野，加权求和得到最后的特征图充分融合了时序信息和空间信息。

2.3.4 D and T

以往目标检测总是被认为服务于目标追踪领域的研究，Feichtenhofer 等人[65] 反其道而行之，提出训练一个多任务网络 D and T，同时训练目标追踪和目标检测任务，让追踪器来辅助提高检测器的效果。D and T 网络从全卷积神经网络 R-FCN[66] 扩展而来，结构如图 2-7[65]。每帧图像会先输入到 R-FCN 中的特征提取网络中获得对应的特征图，R-FCN 在最后一层额外增加了一个卷积层，用于区分位置敏感特征和位置不敏感特征，随后把所有提取到的特征图输入到候选区生成网络生成一组候选框，表示目标可能出现的区域，最后通过 RoI Pooling 层统一候选框大小，预测目标所在的位置和类别信息。

D and T 在 R-FCN 的基础上新增了一个追踪器 RoI Tracking，用于追踪上一帧的目标在下一帧可能出现的位置。和计算两帧之间光流场的方法相似，D and T 引入目标追踪中常用的相关性滤波，在像素点层级计算局部区域内特征图 x^t 和 $x^{t+\tau}$ 之间的相关性表示 $x_{corr}^{t,t+\tau}$：

$$x_{corr}^{t,t+\tau}(i, j, p, q) =< x^t(i, j), x^{t+\tau}(i + p.j + q) >, \tag{2-13}$$

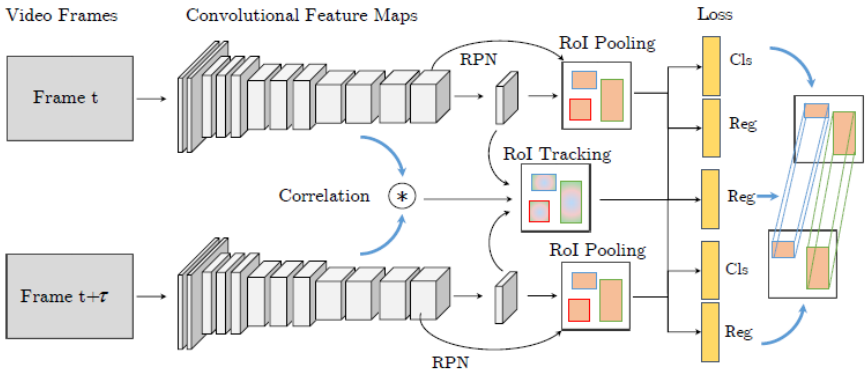

图 2-7: D and T 网络结构

其中 p 和 q 记录局部区域内的偏移值，i 和 j 为特征图上像素点的横纵坐标，相关性表示和候选框同时输入到 RoI Tracking 中就能得到目标的运动轨迹。D and T 最重要的贡献是设计了一个多任务损失函数，联合训练目标检测和目标追踪：

$$Loss = L_{cls} + L_{reg} + L_{tra}, \tag{2-14}$$

L_{cls}、L_{reg} 和 L_{tra} 分别表示分类损失、回归损失和追踪损失，分类损失和回归损失与 R-FCN 网络一致，追踪损失采用 L1-Smooth 损失函数，标签的计算过程类似于边界框损失，用于表示两帧之间位置的差异。时刻 t 和 $t+\tau$ 之间的标签可以表示为:

$$\begin{aligned} \Delta_x &= \frac{B_x^{t+\tau} - B_x^t}{B_w^t}, \ \Delta_y = \frac{B_y^{t+\tau} - B_y^t}{B_h^t}, \\ \Delta_w &= \log(\frac{B_w^{t+\tau}}{B_w^t}), \ \Delta_h = \log(\frac{B_h^{t+\tau}}{B_h^t}), \end{aligned} \tag{2-15}$$

(B_x,B_y,B_w,B_h) 表示目标所在检测框的中心坐标和长宽值。考虑到目标在多帧之间可能并不是一一对应的，因此 D and T 计算跟踪损失时只会考虑多帧中同时存在的目标。实验表明，在 D and T 中添加跟踪损失可以使静态图像学习进行更好的特征学习，明显提高了检测器的检测效果。

2.3.5 SELSA

本节前述的几种方法主要是在整个特征层上聚合时序信息，D and T 虽然是在候选框层面上聚合时序信息，然而也是先在整个特征层上完成了特征对齐。

这些方法都将输入限制在了邻近帧内，短时数据提取到的特征信息仍然存在大量冗余，不利于进一步提升视频目标检测效果。Wu 等人[50] 关注于扩大聚合图像帧的范围，直接将范围扩大到整个视频，提出 SELSA 网络。长时范围内物体的位置会存在相当大的差异，采用光流场或者可变形卷积等方法很难对齐特征，因此 SELSA 在候选框层级聚合语义信息，这样就不会受到位置信息的干扰，当然，也就没有利用上这部分的性质。

SELSA 的原理并不复杂，网络前半部分以 Faster R-CNN 为基础，随机选择视频中的三帧图像输入到 Faster R-CNN 中。每帧图像由特征提取网络提取到的特征会通过候选框生成网络生成一系列候选框，一般选择为 300 个候选框，随后输入到 ROI Pooling 层统一候选框对应特征的尺寸。三帧图像共计 900 个候选框会计算彼此之间的余弦距离作为相关性度量：

$$c_{ij}^{kl} = \phi(x_i^k)^T \varphi(x_j^l), \tag{2-16}$$

其中 x_i^k 表示第 i 帧图像中的第 k 个候选框，x_j^l 表示第 j 帧图像中的第 l 个候选框。ϕ 和 φ 为转换函数，用于计算余弦距离，c 表示相关性度量。为了将相关性度量和注意力机制中的权重参数 w 联系起来，SELSA 采用了 Sigmoid 函数将相关性度量限制在了 $(0, 1)$ 之间，参考帧的 300 个候选框就能利用权重参数聚合所有候选框的信息：

$$\bar{x}_i^k = \sum_{l \in \omega} \sum_{j=1}^{N} w_{ij}^{kl} x_j^l. \tag{2-17}$$

训练阶段计算损失函数的时候，也只会考虑参考帧提取到的 300 个候选框。SELSA 完全弃用了时序信息，只利用了时间维度上全局的语义信息，对于快速运动的物体带来了更多的检测性能提升。

2.4 本章小结

本章主要介绍了监控视频原理和视频目标检测任务的定义，随后依次介绍了静态目标检测和视频目标检测中具有代表性的方法，并且分析了不同方法的优缺点。

第三章　基于结构重参化的静态目标检测模型

本章主要关注静态目标检测问题。我们分析了静态目标检测任务的意义，论述了现有方法可以改进的地方。随后以静态目标检测器 YoloX 为基础，结合结构重参化等技术，提出了 YoloX-Lite 网络。实验表明，YoloX-Lite 相较于 YoloX 实现了更快的推理速度和检测精度。

3.1　研究动机

静态目标检测已经是计算机视觉领域中应用范围最广，研究人员最多的任务之一。提高静态目标检测模型的效果不仅可以推动更多相关项目的落地，同时也帮助视频目标检测领域进一步发展。在2.2节中，我们详细论述了现有的几类静态目标检测算法的特点，其中两阶段方法推理速度太慢，不基于锚框的方法刚刚起步，只有 Yolo 这类单阶段目标检测的方法最受工业界的喜爱。基于 Yolo 的方法从实际应用角度出发，集成了很多卓越的思想，在检测精度和推理速度方面都相当优秀。

当前基于 Yolo 的方法仍然存在不足之处，只有应用参数量相当大的模型才能达到理想的检测精度，然而这类模型往往要求过于昂贵的计算设备，不适合实际应用。因此，我们需要改进基于 Yolo 的方法，在低参数量的基础上谋求更高的精度和更快的速度。以 YoloX 为例，我们分析后发现其所用的模型分支较多、基础模块参数量较大，影响了推理速度。同时 YoloX 在颈部网络特征融合时对所有特征一视同仁，不符合注意力机制的思想，很明显不同的特征对于其他层的影响是有区别的。这些问题都说明 YoloX 还有很多可以改进的空间。

3.2 YoloX-Lite

针对 YoloX 在低参数量的基础上速度和精度存在的巨大提升空间，我们提出了一个能更好权衡推理速度和检测精度的静态目标检测网络 YoloX-Lite。在 YoloX-Lite 中，我们设计了 Rep-Bottleneck 代替 YoloX 中原始的 Bottleneck 模块，并使用结构重参化技术解耦训练和推理模型，提升了模型的推理速度。为了能更好融合不同层次的特征信息，我们提出了 Slim-Fusion Neck 网络，该结构重新设计了信息的流动路径，结合了注意力机制提高了信息融合的效率。经过实验证明，YoloX-Lite 相较于经典的目标检测框架仍然有着不错的表现。

本节我们会详细介绍 YoloX-Lite 的设计原理，首先会介绍整个模型框架，随后依次介绍每个模块的具体结构。

3.2.1 模型结构

YoloX-Lite 模型结构如图 3-1所示：

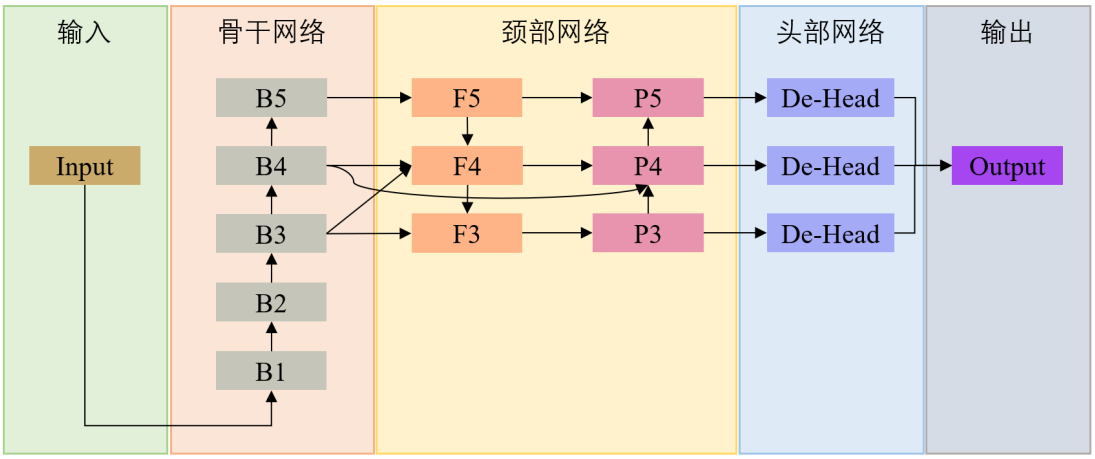

图 3-1: YoloX-Lite 模型结构

1. 输入。YoloX-Lite 的输入是一张图像，图像的大小会缩放到 32 的整数倍，以符合模型下采样的要求。在训练阶段，输入图像会使用翻转、色彩变换和 Mosaic[31] 等数据增强方法，以扩大训练数据集。

2. 骨干网络 (Backbone)。骨干网络主要用于提取输入图像的特征信息，分为 $B1$ 到 $B5$ 这 5 个层次，提取到的特征图分辨率逐层缩小，同时包含的语义信息也逐渐丰富。当前主流的特征提取网络以多分支网络为主，相较于单路网络而言，提取到的特征表达能力更强，但是不利于实际部署。我们参考结构重参

化技术，提出了 Rep-Bottleneck 结构，将训练阶段和推理阶段用到的特征提取网络解耦，训练阶段使用多支路网络以达到更好的精度；推理阶段通过结构重参化技术，合并分支，提高推理速度。实验表明，结构重参化技术可以在保持原有模型相同检测精度的基础上，提高模型的推理速度。

3. 颈部网络 (Neck)。颈部网络主要用于融合高层的语义信息和浅层的精确位置信息，分为自顶向下（图 3-1中 $F5$ 到 $F3$）和自底向上（图 3-1中 $P3$ 到 $P5$）两个阶段。我们在 PANet[67] 网络结构的基础上，引入 BiFusion[33] 网络的思想，增加了两条支路，制定了新的算子使用规则，用于更好地融合不同层次的特征。为了轻量化整个颈部网络，我们使用 Rep-Bottleneck 结构代替 Bottleneck 完成特征融合。颈部网络通常直接融合来自不同层次的特征，但是不同层次的特征表示的语义信息存在差别，在每次融合前使用注意力机制区分重要信息传递下去，可以让特征融合更有效率。我们将整个颈部网络定义为 Slim-Fusion Neck，实验表明，Slim-Fusion Neck 相较于 PANet 能达到更高的检测精度。

4. 头部网络 (Head)。头部网络主要用于从处理好的特征中，预测出目标的类别信息和位置信息。我们维持了 YoloX 中的解耦头结构，将分类分支和回归分支区分开来，直接预测目标的实际坐标值，整个网络也就变成了不基于锚框的检测方法。相较于传统的基于锚框的 Yolo 方法，不用计算先验的锚框信息，整个网络的效果不再受制于先验锚框的准确性。

5. 输出。YoloX-Lite 三层的预测结果会通过映射到原图上去，随后合并所有的预测值，并使用非极大抑制（No Maximum Suppression, NMS）等算法去除冗余的预测框，得到最后的输出结果。

3.2.2 结构重参化

骨干网络通常是从图像分类网络变换而来，分为单路特征提取网络[2, 68] 和多路特征提取网络[59, 69–71]。这类以神经网络为基础的特征提取器相较于传统人工设计特征的策略性能更加优秀，可以自动提取表达能力更强的特征。单路特征提取网络简单直观，数据流向单一。但是随着网络深度增加，这类模型在训练的过程中容易出现梯度消失的现象。梯度消失指的是模型在反向传播计算梯度时，会通过链式法则逐层计算可学习参数的偏导数，当偏导数较小并且网络层数逐渐加深，就会使得偏导数接近于 0，参数停止更新。多路特征提取网络也存在独有的问题，那就是过多的分支导致计算效率和显存利用率低下。这类网络在计算过程中，每条支路都需要单独开辟一块显存用于存储中间生成的特

征，也无法利用上硬件高效的并行计算库，要求所有支路计算完成后才能进入到下一阶段。

本文参考结构重参化思想，提出 Rep-Bottleneck 结构，替代 YoloX 中的 Bottleneck 结构，解耦训练和推理用到的模型。我们使用多路特征提取网络训练，以达到更好特征提取效果，随后使用结构重参化技术，合并分支，在推理阶段依靠硬件自带的加速库实现更快的推理速度。

YoloX 使用的骨干网络 CSPDarknet53 如图 3-2，整个网络只包含标准卷积（CBS）、C3 和最后的池化层 SPPF[33]。标准卷积由卷积层、批归一化（Batch Normalization，BN[70]）层和激活层（SiLU[72]）组成，而 C3 则是由三个标准卷积和一系列 Bottleneck 堆叠而成。每个 Bottleneck 包含了两个卷积核大小分别为 1×1、3×3 的标准卷积和一个自加操作，Bottleneck 数量由骨干网络的大小决定，当骨干网络要求更好的特征提取效果时，Bottleneck 的数量和通道数都会增加。

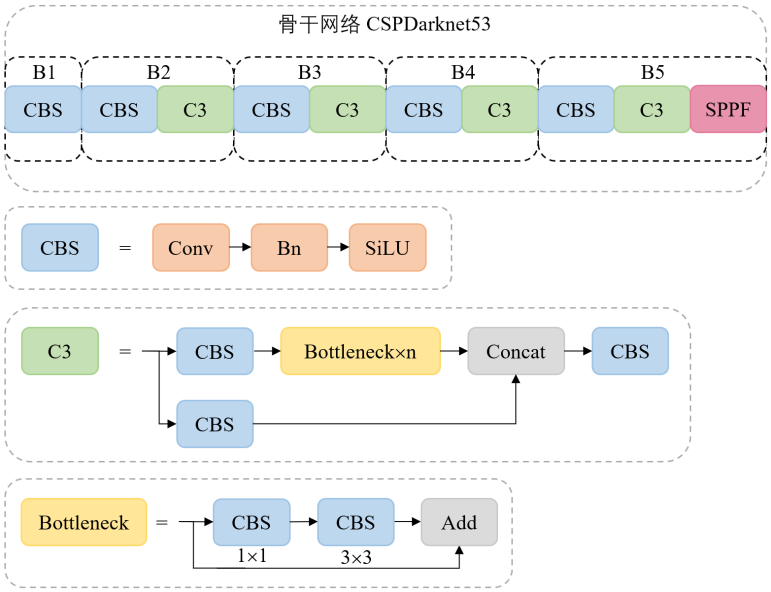

图 3-2: CSPDarknet53 网络结构

我们发现由于 Bottleneck 的自加操作，整个骨干网络存在了过多的分支，严重影响了模型的推理速度。为了缩减分支数量，我们提出了 Rep-Bottleneck 结构，如图 3-3所示。我们移除了 Bottleneck 中的第一个激活层，并将第二个激活层迁移到自加操作之后，使得整个结构只有最后一层为非线性算子，便于后续的结构重参化。

结构重参化技术指的是模型在训练过程中，仍然使用多分支的网络结构，

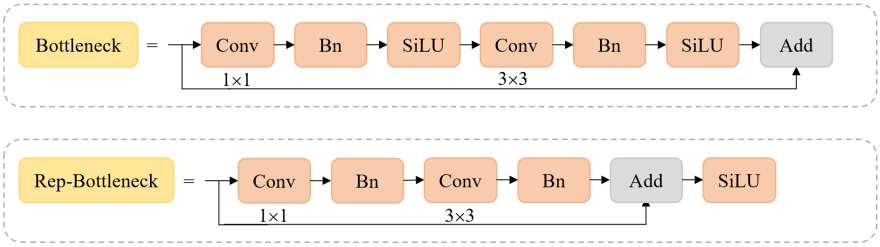

图 3-3: Bottleneck 和 Rep-Bottleneck 结构

以此达到最好的检测效果。而在推理之前，合并分支，最大可能利用上硬件的并行计算能力来缩短推理时间，同时不影响模型的效果。我们设计结构重参化的标准是尽量将其他操作转换为卷积核大小为 3×3 的卷积操作，这是因为现有的 Nvidia 的 cudnn 加速库对 3×3 的卷积核计算更加友好。Ding[71] 通过实验验证相同条件下不同卷积核大小的计算密度，结果如表 3-1所示，说明了 3×3 卷积操作的计算密度远远了高出其他大小的卷积。

表 3-1: 不同大小卷积核理论计算量、计算耗时和计算密度。所有数据在批量大小为 32，通道数为 2048，分辨率为 56×56，步长为 1，显卡为 GeForce 1080ti 环境下测试

卷积核大小	理论计算量 (B)	计算耗时 (ms)	计算密度
1*1	420.9	84.5	9.96
3*3	3788.1	198.8	**38.10**
5*5	10522.6	2092.5	10.57
7*7	20624.4	4394.3	9.38

接下来我们会具体讲解结构重参化的过程。第一步是融合卷积层和批归一化层，我们知道卷积操作可以形式化定义为：

$$Y = W_{conv}X + Bias_{conv}, \tag{3-1}$$

其中 W_{conv} 是卷积核参数，$Bias_{conv}$ 是偏置，X 和 Y 分别是输入和输出。而批归一化操作可以形式化定义为：

$$\begin{aligned} \hat{X} &= \frac{X - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}, \\ Y &= \gamma\hat{X} + \beta, \end{aligned} \tag{3-2}$$

其中 μ_β 和 σ_β^2 为输入数据的均值和标准差，γ 和 β 是可学习参数，这四个参数

在训练结束后就已经固定好了，X 为批归一化层的输入，Y 是批归一化层的输出。合并卷积层和批归一化层，可以得到：

$$y = \frac{\gamma}{\sqrt{\sigma_\beta^2 + \epsilon}} W_{conv} x + (\beta + \frac{Bias_{conv} - \gamma\mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}), \tag{3-3}$$

可以视 $\frac{\gamma}{\sqrt{\sigma_\beta^2+\epsilon}} W_{conv}$ 为合并后卷积层的权重，$(\beta + \frac{Bias_{conv}-\gamma\mu_\beta}{\sqrt{\sigma_\beta^2+\epsilon}})$ 为合并后卷积层的的偏置，这样原始的卷积层和批归一化层就合并为了一个卷积核大小为 3×3 的卷积层，合并后的结构如图 3-4所示。

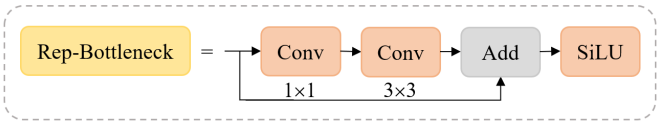

图 3-4: 合并卷积层和批归一化层后的 Rep-Bottleneck 结构

结构重参化的第二步是将 1×1 和 3×3 的卷积合并成一个卷积层。在 Rep-Bottleneck 中 1×1 的卷积层主要是为了降维，我们假设输入特征图为 X，通道数为 C，其中通道为 $i \in C$ 的特征图为 X_i。假设 1×1 卷积的输入通道为 C_{in}，输出通道为 C_m，卷积核值为 $W_1^{C_m \times C_{in} \times 1 \times 1}$，偏置值为 $Bias_1^{C_m}$，3×3 卷积的输出通道为 C_{out}, 值为 $W_2^{C_{out} \times C_m \times 3 \times 3}$，偏执为 $Bias_2^{C_{out}}$。Rep-Bottleneck 中输入通道和输出通道相等，即 $C_{in} = C_{out} = C$。串行的两个卷积计算可以形式化为：

$$\begin{aligned} Y_1^j &= \sum_{i \in C} w_1^{j,i} X^i + Bias_1^j, j \in C_m, \\ Y_2^k &= \sum_{j \in C_m} W_2^{k,j} Y_1^j + Bias_2^k, k \in C, \end{aligned} \tag{3-4}$$

其中 Y_1^j 表示第一个卷积层的通道为 j 的输出结果，Y_2^k 表示第二个卷积层的通道为 k 的输出结果，这里我们使用“WX”的形式来简化表示卷积运算，实际上并不能简单视为两个矩阵直接相乘。合并两个卷积计算，得到:

$$Y_2^k = \sum_{j \in C_m} W_2^{k,j} (\sum_{i \in C} w_1^{j,i} X^i + Bias_1^j) + Bias_2^k, k \in C. \tag{3-5}$$

由于上式中的 $w_1^{j,i}$ 和 $Bias_1^j$ 是标量，因此我们可以把两个权重合并为一个，得到：

$$Y_2^k = \sum_{i \in C} (\sum_{j \in C_m} w_1^{j,i} W_2^{k,j}) X^i + (\sum_{j \in C_m} Bias_1^j W_2^{k,j} + Bias_2^k), k \in C. \tag{3-6}$$

合并后第 k 个卷积核的权重为 $\sum_{j\in C_m} w_1^{j,i} W_2^{k,j}$，偏置为 $\sum_{j\in C_m} Bias_1^j W_2^{k,j} + Bias_2^k$。

结构重参化的第三步是将自加操作转换为卷积操作。假设对于自加操作转换后的卷积，其输入通道为 C_{in}、输出通道为 C_{out}、卷积核大小为 3×3，卷积核的维度为 $C_{out} \times C_{in} \times 3 \times 3$，很明显 $C_{in} = C_{out}$。我们参考了深度可分离卷积(Depthwise Convolution)[59] 的实现，对于卷积核中第 $i \in C_{out}$ 个大小为 $C_{in} \times 3 \times 3$ 的张量，我们使该张量第 i 个通道的 3×3 矩阵中心位置的参数值为 1，其余所有位置的值为 0。该卷积操作计算得到的输出等于输入，自加支路也就转换为了 3×3 的卷积层，转换后的结构如图 3-5。

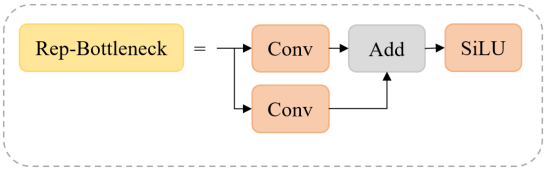

图 3-5: 自加支路转换为卷积操作

结构重参化的最后一步就是利用卷积操作的可叠加操作，将自加操作转换而来的卷积和另外一条支路的卷积操作叠加起来，这样整个 Rep-Bottleneck 就转换为了一个卷积层和一个激活层，如图 3-6所示。

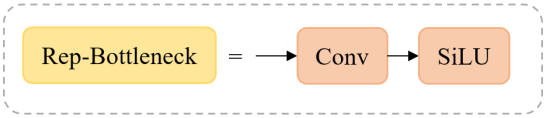

图 3-6: Rep-Bottleneck 结构重参化后的结构

如上述所说，结构重参化的整个过程并没有影响到模型原本的表达效果，只是将多个线性计算合并成了一个卷积操作，极大提高了计算效率。我们将使用 Rep-Bottleneck 代替 C3 和 CSPDarknet53 中的 Bottleneck，并将其命名为 RepC3 和 Rep-CSPDarknet53，如图 3-7。

3.2.3 Slim-Fusion

众所周知，特征提取网络越深，提取到的特征感受野也就越大，语义信息比较丰富，但是分辨率低，目标位置信息比较粗略。常见的特征提取网络最后一层下采样率为 $\frac{1}{32}$，该层中的每个像素点的感受野对应原始输入图像中的 32×32 的区域，也就是说原图中像素小于 32×32 的目标对应的特征在最后一

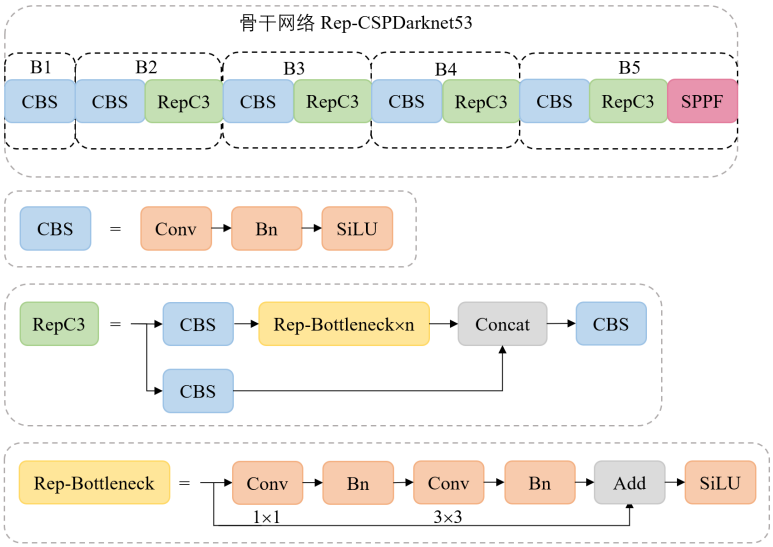

图 3-7: Rep-CSPDarknet53 结构

层大概率已经丢失，深层的特征图并不适合检测小目标物体。与此同时，特征提取网络提取得到的浅层特征语义信息比较少，但是目标位置准确，感受野较小，能保留小目标的信息。为了让浅层特征图也能拥有丰富的语义信息，FPN[34] 将高层语义信息自上而下传递给浅层特征；为了让高层特征图保留更多的精确位置信息，PANet 在 FPN 的基础上新增了自底向上的传递路径。如何高效融合不同尺度的特征，取其精华，弃其糟粕，已经是提高目标检测效果的重中之重。

缩短信息流动路径和扩充分支来增加路径数量的思想会产生比较强的性能，总结起来就是“分割-转换-合并”和“特征重用”，近年来很多神经网络的改进都是这种思路。我们参考这种思想提出了 Slim-Fusion Neck，整体结构如图 3-8所示。

我们从骨干网络中提取到三个层次特征图，分别为 $B3$、$B4$ 和 $B5$，表示由浅到深的三层输入表示。在自顶而下的特征融合过程中，我们使用一个标准卷积对 $B5$ 降维，将其直接作为特征图 $F5$ 输出给其他层。随后拼接特征图 $B3$、$B4$ 和 $F5$，并依次使用标准卷积降维，使用 RepC3 结构进一步提取特征，以及通道注意力（Channel Attention Module，CAM）层生成特征图 $F4$，CAM 的具体结构会在后文详细讲解。$F4$ 经过上采样后统一到 $B3$ 的分辨率，拼接两个特征图，再使用一个标准卷积降维，得到特征图 $F3$。

自底而上和自顶向下的过程相反，浅层的特征图会使用相同的 RepC3 结构增强融合后特征图的表达能力，然后使用卷积核为 3，步长为 2 的标准卷积操

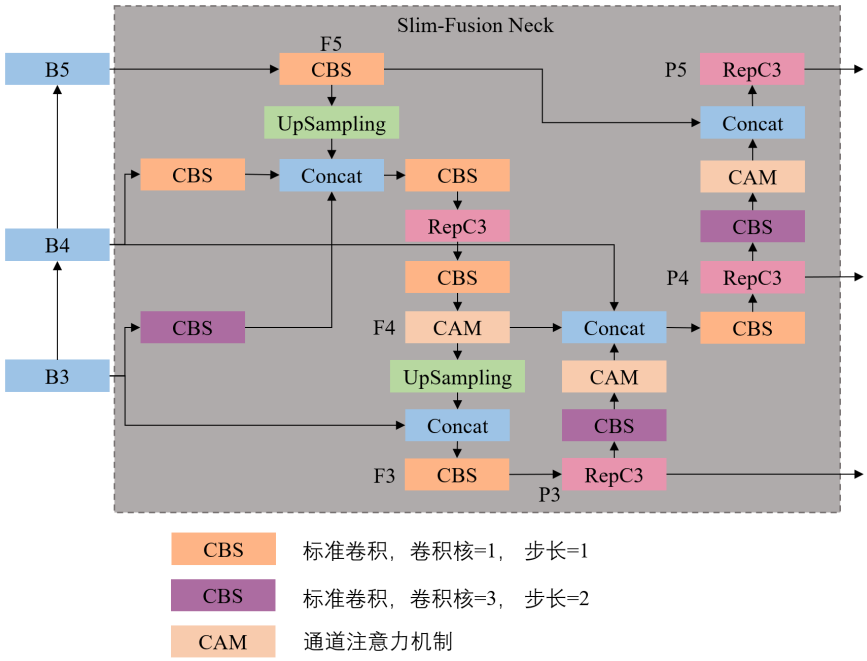

图 3-8: Slim-Fusion Neck 网络结构

作降低特征图的分辨率。每次卷积操作都不可避免丢失一些信息，我们注意到中间层的输出距离 $p4$ 过远，忽略了 $B4$ 层原始的信息，因此我们新增了一条支路，在生成 $P4$ 的过程中直接叠加 $B4$ 层的原始输入，以便提取到更加合适的信息。自底向上的过程中我们使用分而治之的思想，每层的输出更加关注预测特定大小的目标。

视觉注意力可以帮助人类自动忽略次要信息和噪声，更加关注于目标的主体。研究人员模拟注意力机制的作用，让模型重点关注特定部分的特征，在视觉、自然语言处理和语音等领域都取得了瞩目的效果。我们发现在不同层次信息融合时，PANet 视各层信息重要性相同。然而在特征提取的过程中，不同层次的特征图语义信息存在差别，包含的噪声信息也不同，每次融合应该仅仅传递重要的特征，而不是将背景信息和噪声信息也一并融合。基于上述的分析，我们在每次合并多层特征前在主干中使用通道注意力模块，将更多的注意力聚焦于关键特征，使得融合后的特征能包含更多有效信息。

在 Slim-Fusion Neck 中我们选择使用 CBAM[64] 模块前半部分的通道注意力机制，整体结构如图 3-9。对于输入的特征图，CAM 会在每个通道上做平均池化和最大池化，然后经过共享的全连接层生成代表通道重要性的向量表示，合并两个向量，同时使用 sigmoid 函数归一化到 $[0, 1]$ 之间，即得到最后的注意力权重值，使用加权求和计算出最后分配了注意力的特征。

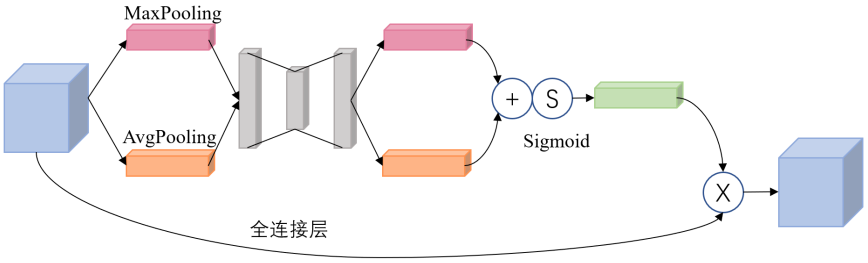

图 3-9: CAM 网络结构

为了提高颈部网络的推理速度，我们利用了前一节提出的 Rep-Bottleneck 融合特征。这里我们不重复介绍 Rep-Bottleneck 的实现细节。

我们将颈部网络的设计思想总结为以下几条规则：

- 同尺度的特征图使用卷积核大小为 1，步长为 1 的卷积降低通道数。
- 大尺度的特征图使用卷积核大小为 3，步长为 2 的卷积降低分辨率。
- 小尺度的特征图使用双线性插值扩大分辨率。
- 不同层次的特征融合前使用通道注意力机制降低噪声的干扰。
- 使用 Rep-Bottleneck 结构加快推理速度。

3.2.4 训练策略

在本小节中我们会详细介绍损失函数的选择和数据增强的使用细节，其余的实验设置会在 3.3一节中详述。

损失函数是用于度量模型预测结果和真实标签的偏差，通过惩罚机制更新模型的参数，使得模型倾向于真实标签值。我们在 2.2.2一节描述过解耦头是由两条互不干扰的分类分支和回归分支组成，分类分支用于预测目标的类别，回归分支用于预测目标属于前景还是背景，以及目标所处位置。整个网络的损失函数 $Loss$ 由分类损失 $Loss_{cls}$、前后景损失 $Loss_{obj}$ 和位置回归损失 $Loss_{box}$ 组成：

$$Loss = Loss_{cls} + Loss_{obj} + \lambda_{box} Loss_{box}. \tag{3-7}$$

我们使用交叉熵（Cross Entropy Loss）损失作为分类损失和前后景损失，见 3-8和 3-9，其中 N_b 表示训练过程中每个迭代批次的标签数量，N_t 表示类别数量，N_o 值为 2，表示预测结果是目标还是背景，$I(\cdot)$ 表示指示函数，输入的真值会转换为 1，输入的假值会转换为 0，$p_{i,c}$ 表示第 i 个结果属于类别 c 的

概率，

$$Loss_{cls} = -\frac{1}{N_b}\sum_{i=1}^{N_b}\sum_{c=1}^{N_t} I(y_i = c)\log(p_{i,c}), \tag{3-8}$$

$$Loss_{obj} = -\frac{1}{N_b}\sum_{i=1}^{N_b}\sum_{c=1}^{N_o} I(y_i = c)\log(p_{i,c}). \tag{3-9}$$

我们使用交并比损失（Intersection over Union Loss，IoU Loss）作为边框回归损失，公式见3-10，Box_{gt} 为真实标签的边框位置，Box_{pred} 为预测边框位置。$Loss_{box}$ 要求预测框和真实框之间的交并比尽可能小。

$$\begin{aligned} IoU &= \frac{Box_{gt} \cap Box_{pred}}{Box_{gt} \cup Box_{pred}}, \\ Loss_{box} &= -\ln IoU. \end{aligned} \tag{3-10}$$

数据增强本质是从无到有生成一批原本不存在的数据，扩充我们有限的数据。神经网络的训练过程中，梯度会走最陡的路径，导致模型选择最容易解决问题的特征，也就是我们常说的过拟合。通过数据增强方法，让训练中的模型可以见到更多的困难样本，避免模型走最简单的路，而是去走最正确的路。目标检测中的 MixUp[73] 和 Mosaic[31] 数据增强通过“裁剪 + 缩放 + 拼接”的方式，生成大量新数据，在各类目标检测网络中都表现出优秀的性能。然而这种“裁剪 + 缩放 + 拼接”的方式使得数据过分偏离于实际图像分布，限制了模型的表达效果。因此，我们使用 MixUp 和 Mosaic 数据增强方法正常训练 300 轮，随后新增 30 轮训练，关闭数据增强，只训练原始图像数据，使最后的结果更加贴近于实际场景。

3.3 实验与分析

3.3.1 实验数据分析

我们选择使用目标检测领域常用的 Micrsoft COCO 2017 数据集进行训练和测试。

COCO 数据集是一个可用于图像检测、语义分割和图像标题生成的大规模数据集，背景包含沙滩、街道、农场等生活中常见的场景。COCO 数据集一共包含 33 万张图像，150 万个目标，平均每张图片包含 4.5 个目标，具体训练集和验证集划分见表3-2。COCO 数据集中所有目标被划分为 80 个类别，如

图3-10所示，大部分类别的标注框都超过 1000 个。COCO 将目标按照标注框大小划分为大目标（分辨率大于 96×96），中目标（分辨率小于 96×96，并且大于 32×32）和小目标（分辨率小于 32×32），在数据集中分别占比 24%，32%和 41%，可想而知 COCO 数据集作为目标检测的标准数据集还是有比较高的检测难度。

表 3-2: COCO 训练集和测试集划分

	图片数量	标注数量
训练集	118287	844701
测试集	5000	36683

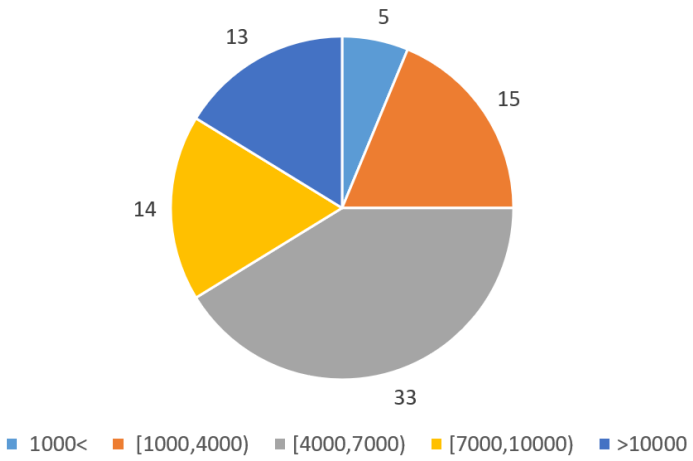

图 3-10: COCO 数据集中不同标注框数量对应的类别数量

3.3.2 评价指标

本实验使用目标检测通用的评价指标 mAP（mean Average Precision）评估模型分类和边框匹配的精确度。每个类别的目标都会计算出 AP（Average Precision）分数，所有类别 AP 分数的平均值即 mAP。在引出 AP 分数的计算细节前，我们先介绍几个前置数据的概念。首先是 IoU 的概念，IoU 用于描述两个目标重叠的区域面积占两个目标合并总面积的百分比，见公式3-10。其次 TP、FP、FN 和 TN 的定义见表3-3，用于表示预测值和真值之间的关系，对应到目标检测中 TP 即预测类别和真实类别一致，并且预测坐标和真实坐标的 IoU 大于阈值 t。 最后介绍精确率（Precision）和召回率（Recall）的概念，精确率

表 3-3: TP、FP、FN 和 TN 的定义

	标签为真	标签为假
预测为真	True Positive(TP)	False Positive(FP)
预测为假	False Negative(FN)	True Negative(TN)

用于描述预测值为真的目标中正确的百分比，召回率用于描述标签为真的目标中被预测正确的百分比。我们用 TP 等指标来描述就是:

$$\begin{aligned} Precision &= \frac{TP}{TP+FP}, \\ Recall &= \frac{TP}{TP+FN}. \end{aligned} \tag{3-11}$$

接下来我们会引出 AP 的计算过程，我们将预测结果根据置信度由大到小排序，设置好 IoU 阈值为 t。从置信度最大的预测结果开始，我们依次计算到当前结果时的精确率和召回率，这样我们就得到了一组转换后的结果，以召回率作为横坐标，精确率作为纵坐标绘制折线图，折线图和横坐标所围成的面积即当前类别的 AP 分数。考虑到 IoU 阈值对 mAP 计算有较大影响，静态目标检测通常根据 IoU 阈值将 mAP 划分为 mAP50 和 mAP50:95，分别表示 IoU 阈值为 0.5 时的 mAP 结果，以及 IoU 阈值从 0.5 到 0.95，步长为 0.05 的所有 mAP 的平均值，一般直接简称 mAP50:95 为 mAP。从目标分辨率大小来划分，又可以将 mAP 划分为 mAP_s，mAP_m 和 mAP_l，分别用来表示小目标、中目标和大目标的 mAP 结果。

3.3.3 实验设置

实验训练和测试使用的硬件平台配置如表3-4所示，使用到的推理软件 Pytorch 版本为 0.9，编程语言 Python 版本为 3.9，显卡驱动版本为 455.32，并行计算库 CUDA 版本为 11.1。

训练阶段，输入图像分辨率设置为 (640, 640)，以便和其他目标检测模型对比。每次迭代的批量大小设置为 64，模型的总训练轮次为 330，前 5 轮用于预热，最后 30 轮关闭 MixUp 和 Mosaic 等数据增强方法。训练阶段用到的优化器为随机梯度下降 (Stochastic Gradient Descent, SGD)，动量为 0.9，初始学习率设置为 0.04，学习率衰减方法选择余弦退火方案，退化权重设置为 0.0005，使用指数加权平均增强模型的鲁棒性。非极大值抑制阈值设置为 0.65，用于过滤结

表 3-4: 实验硬件平台配置

配置	参数
操作系统	Linux
CPU	Intel(R) Xeon(R) E5-2678 v3
GPU	Nvidia RTX 2080ti * 4
内存	252G

果中的重复值。损失函数中位置回归损失权重 λ_{box} 设置为 5。

根据 Bottleneck 的数量和特征的通道数，YoloX 被划分为 YoloX-n、YoloX-t、YoloX-s、YoloX-l 和 YoloX-x 几个类型。其中 YoloX-n 使用空洞卷积代替标准卷积，不能转换为 Rep-Bottleneck，而 YoloX-l 和 YoloX-x 模型参数过大，不符合低参数量的要求，本文最后选择在 YoloX-t 和 YoloX-s 上进行改进。考虑到 YoloX-t 的通道数过低，Slim-Fusion Neck 结构并没有明显提升检测精度，反而带来了时延开销，因此我们将 YoloX-t 结合 Rep-Bottleneck 的结构称为 YoloX-Lite-t，将同时结合 Rep-Bottleneck 和 Slim-Fuion Neck 的 YoloX-s 称为 YoloX-Lite-s。

3.3.4 对比实验

（1）与基线方法对比

本文在 COCO 数据集上验证了 YoloX-Lite 的效果，具体结果如表3-5所示。可以发现，YoloX-Lite-t 相较于 YoloX-t，其 mAP 下降 0.4，单张图片的推理时延下降了 1.21ms，体现出每个 Rep-Bottleneck 结构减少一次非线性映射带来的精度下降，以及结构重参化技术合并线性算子带来的巨大的速度提升。而从 YoloX-Lite-s 的表现来看，在降低了 0.26ms 推理时延的前提下，mAP 指标增加了 0.7，反映出 YoloX-Lite-s 能够实现更高的检测精度和更快的推理速度。不仅如此，YoloX-Lite-s 在小型目标和中型目标上表现更加优秀，适合小目标检测。还可以发现，不同模型的参数量和推理速度并没有直接联系，符合前文介绍的推理速度受到模型参数量，以及模型结构复杂度等多个方面的影响，不仅仅受限于模型参数量。

图3-11展示了 YoloX-Lite-s 和 YoloX-s 在 COCO 数据集中的代表性结果。其中第一行使用我们的方法检测，用红色框表示检测结果，红色箭头用于表示

表 3-5: YoloX 和 YoloX-Lite 在 COCO 数据集上的检测效果

模型	mAP	mAP50	mAP_s	mAP_m	mAP_l	Params(M)	Times(ms)
YoloX-t	**32.7**	**50.0**	**13.4**	**35.7**	**48.8**	5.06	9.35
YoloX-Lite-t	32.3	49.5	13.6	35.0	48.4	**4.92**	**8.14**
YoloX-s	40.5	59.3	23.3	44.8	**54.1**	**8.97**	10.05
YoloX-Lite-s	**41.2**	**60.1**	**24.4**	**45.4**	53.3	9.48	**9.79**

YoloX-s 未能检测到的结果，第二行使用 YoloX-s 检测，用黄色框表示检测结果。我们可以从中观察到，对于大型目标而言，我们的方法和 YoloX-s 表现基本一致。当涉及到中小型目标时，例如第一列中的牛、第二列的椅子和第三列画中的人，我们的方法能够完整检测到目标，效果明显更好。

图 3-11: YoloX-Lite-s 和 YoloX-s 在 COCO 数据集中的代表性结果

（2）与当前主流方法对比

表 3-6给出了不同目标检测模型在 COCO 数据集上的检测效果，每列数据效果最好的两个数值以红色和蓝色分别标出。除了本文设计的 YoloX-Lite-s 之外，我们引入了一些主流的目标检测方法作为对比，其中 MaskRCNN 为两阶段方法，YoloX-s、YoloV5-s、YoloV6-s 和 EfficientD1 为单阶段方法，CornerNet 和 Centernet(DLA) 为不基于锚框的方法。为了保证公平，所有模型输入的图像分辨率均为 (640,640)，推理速度均在 GeForce GTX 2080ti 显卡上测试得到。可以发现，我们的方法表现出了最快的推理速度，同时 mAP 也仅仅落后于

表 3-6: 主流模型在 COCO 数据集上的检测效果

模型	mAP	mAP50	Params(M)	Times(ms)
YoloX-Lite-s	41.2	60.1	9.48	9.79
MaskRCNN	39.8	62.3	-	90.91
YoloX-s	40.5	59.3	8.97	10.05
YoloV5-s	37.4	57.1	7.20	10.20
YoloV6-s	43.1	61.8	18.54	12.11
EfficientD1[36]	39.3	59.2	11.59	33.68
CornerNet	40.5	56.5	122.00	243.90
Centernet	39.2	57.1	77.20	35.71

YoloV6-s。考虑到 YoloV6-s 参数量接近我们方法的一倍，速度也有比较明显的差距，我们的方法仍然具有优势。相较于其他方法，我们的方法在精度和速度方面，都有着明显地提升。

3.3.5 消融实验

（1）Rep-Bottleneck 数量的影响

表 3-7: Rep-Bottleneck 数量对 YoloX-Lite-t 的影响

结构	mAP	mAP50	Param	GFlops	Time
Backbone×0, Neck×0	32.7	50.0	5.06	6.49	9.35
Backbone×1, Neck×0	32.3	49.7	4.98	6.38	8.46
Backbone×2, Neck×0	33.3	50.6	5.78	7.50	8.92
Backbone×1, Neck×1	32.3	49.5	**4.92**	**6.32**	**8.14**
Backbone×1, Neck×2	32.8	49.6	5.44	6.77	8.50
Backbone×1, Neck×4	**34.1**	**50.4**	6.48	7.67	9.31

为了探索 Rep-Bottleneck 数量对 YoloX-Lite 检测速度和推理精度带来的影响，本文增加了 YoloX-Lite-t 中骨干网络和颈部网络中 Rep-Bottleneck 数量，实验结果见表3-7。表中第一列和第二列分别表示骨干网络和颈部网络中 Rep-Bottleneck 结构的数量，其中“×0”表示不使用 Rep-Bottleneck，即原生 YoloX-t 网络。后续几行中的“Backbone× n, Neck× m”表示骨干网络和颈部网络中的 Bottleneck 替换为 Rep-Bottleneck，并且分别重复 n 次和 m 次。观察表

中数据可以发现，使用 Rep-Bottleneck 代替 YoloX 骨干网络中的 Bottleneck 结构在损失 0.4mAP 的基础上，参数量降低了 0.08M，推理时间降低了 0.89ms。当骨干网络中的 Rep-Bottleneck 数量翻倍，mAP 增加了 0.6，推理时间降低了 0.43ms。如果同时替换骨干网络和颈部网络中的 Bottleneck 结构，mAP 仍然为 32.3，mAP50 下降到 49.5，但是实现了最快的推理速度 8.14ms。当颈部网络中的 Rep-Bottleneck 数量翻倍时，mAP 和原生的 YoloX 十分接近，但是推理速度却降低到 8.50ms。后续我们继续增加颈部网络中的 Rep-Bottleneck 的数量，实现了 mAP 增长到 34.1。这组实验充分说明了 Rep-Bottleneck 结构结合结构重参化技术，可以在损失较小的精度下，大幅提高模型推理速度。也可以通过叠加 Rep-Bottleneck 的数量实现保证理想推理速度的前提下，提高模型的检测精度。在本文中我们选择仅仅替换原生 YoloX 网络中的 Bottleneck 结构，并没有过多重复 Rep-Bottleneck。

（2）Slim-Fusion neck

为了验证 Slim-Fusion Neck 网络的作用，我们设计了三个实验，第一个实验将原生的 YoloX-s 网络仅替换骨干网络中的 Bottleneck 为 Rep-Bottleneck，颈部网络仍然是 PANet，命名为 P-Neck；第二个实验在 P-Neck 的基础上将颈部网络中的 Bottleneck 也替换为 Rep-Bottleneck 结构，命名为 SF-Neck⁻；第三个实验在实验二的基础上使用了完整的 Slim-Fusion Neck 网络，命名为 SF-Neck，实验结果见表 3-8。

表 3-8: Slim-Fusion Neck 对 YoloX-Lite 的影响

结构	mAP	mAP50	mAP_s	mAP_m	mAP_l
P-Neck	40.0	58.7	23.2	43.9	**53.7**
SF-Neck⁻	40.1	58.9	23.3	44.3	53.4
SF-Neck	**41.2**	**60.1**	**24.4**	**45.4**	53.3

观察表 3-8，我们可以发现相较于 P-Neck，SF-Neck⁻ 的 mAP 提升了 0.1，对小型目标和中型目标检测更加精准，对大型目标则 mAP 降低了 0.3。当 IoU 阈值变为 0.5 时，SF-Neck⁻ 的 mAP50 提升了 0.2，说明边框预测结果精度上升。对于第三行增加通道注意力机制的 SF-Neck 结构，mAP 有比较明显的提升，分别提升了 1.2 和 1.1，mAP50 也是最高的。SF-Neck 结构在大型目标层面比另外两个结构表现稍差，小目标和中等目标检测效果表现更好。这符合使用

注意力机制的目的，在低参数量的网络中，提取到的特征本来表达能力就不够充分，在颈部网络融合不同层特征时，注意力机制更加关注重要信息，对于小目标和中型目标而言减少了背景信息的干扰，提高了检测效果。对于中等目标来说，其在不同层特征中已经提取到了丰富的信息，需要通过特征融合更加精确定位位置，这时注意力机制就起到了作用，将更加重要的信息传递给了深层特征用于精确定位目标，体现在 mAP50 和 mAP_m 指标有明显提升。

我们在 YoloX-Lite 中集成 Grad-CAM[74] 算法，用于展示不同模型训练出来的参数在测试时更加关注图像中的哪些区域。我们使用 P-Neck 和 SF-Neck 检测鸟类图像，如图 3-12所示。图中央为一只鸟，P-Neck 除了检测到鸟以外，还误检测到了水中的渔网。而 SF-Neck 很明显在整幅图像中，只注意到了目标，没有受到背景或者噪声信息的干扰，表现出了更加优秀的检测性能。

P-Neck 的类激活热力图　　SF-Neck 的类激活热力图

图 3-12: YoloX-Lite 类激活图

（3）迭代次数以及关闭数据增强

为了验证数据增强对 YoloX-Lite 的影响，我们通过恢复训练模式，训练 330 轮 YoloX-Lite-t，分别验证了全程不关闭数据增强（图 3-13灰色曲线），最后 45 轮迭代关闭数据增强（图 3-13橘色曲线），以及最后 30 轮迭代关闭数据增强（图 3-13蓝色曲线）。可以发现在第一种训练方案下，模型在 300 轮迭代后已经收敛，mAP50:95 最高达到 29.59。而第二种方案下，模型在 285 轮处关闭数据增强后，模型精度有比较明显的提升，最终达到了 32.07。观察灰色曲线可以发现在 285 轮迭代时模型还没有完全还没有收敛，我们将关闭数据增强操作延迟至 300 轮时，直到模型收敛，再关闭数据增强继续训练，可以看到模型的精度有了进一步的提升，最高达到了 32.27，相较于前一种关闭数据增强时机 mAP50:95 提高了 0.21。通过上述三种方案的对比，可以证明在目标检测任务中，训练结束阶段应该关闭数据增强使得训练数据更加接近真实数据分布，同

时 YoloX-Lite 应该在模型收敛后再关闭数据增强，理想的迭代轮数应该是 300 轮正常训练，再加上 30 轮关闭数据增强训练。

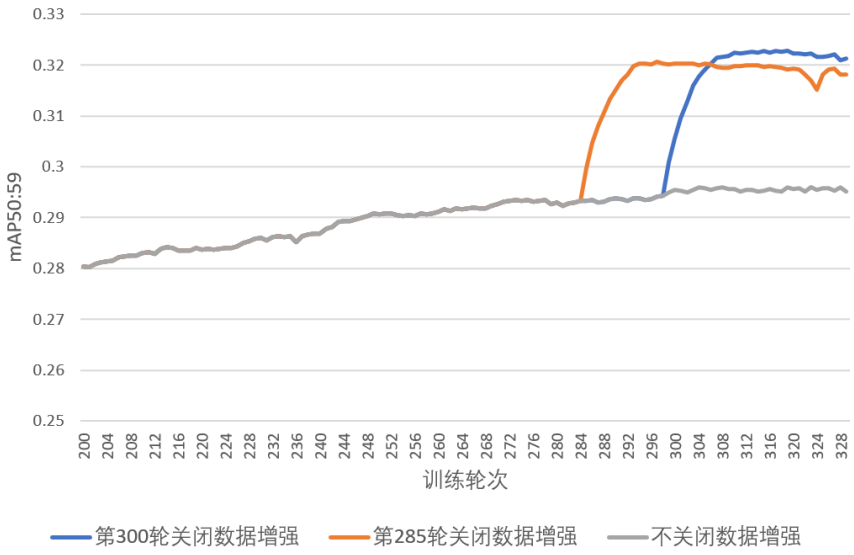

图 3-13: YoloX-Lite 最后 130 轮训练对应的验证集 mAP

（4）在其他方法上的效果

为了验证 Rep-Bottleneck 和 Slim-Fusion Neck 在其他模型上能起到同样的作用，我们将其移植到 YoloV5-n 中，结果见表 3-9。其中 YoloV5-Rep-n 表示仅仅使用 Rep-Bottleneck 代替 YoloV5-n 中的 Bottleneck 结构，YoloV5-Lite-n 表示同时使用 Rep-Bottleneck 和 Slim-Fusion Neck。可以从表中看出，当仅仅使用 Rep-Bottleneck 时，YoloV5-Rep-n 同样表现出推理速度大幅加快，而 mAP 仅仅下降了 0.4，这与 YoloX-n 表现一致。至于 YoloV5-Lite-n，尽管使用了 Slim-Fusion Neck 结构，相较于 YoloV5-n 速度和精度均没有明显的优势，符合前文提到的当通道数过少时，Slim-Fusion Neck 并不会带来明显的效果。

表 3-9: YoloV5 移植 Rep-Bottleneck 和 Slim-Fusion Neck 的效果

模型	mAP	mAP50	mAP_s	mAP_m	mAP_l	Times(ms)
YoloV5-n	27.4	45.6	13.6	31.1	**35.7**	9.20
YoloV5-Rep-n	27.0	44.9	13.1	31.3	34.8	**8.35**
YoloV5-Lite-n	**27.5**	**45.8**	**14.2**	**31.4**	35.4	9.02

3.4 本章小结

本章我们从静态目标检测模型 YoloX 展开研究，分析了 YoloX 可改进之处，提出了平衡检测精度和速度更好的网络 YoloX-Lite。我们修改了 Bottleneck 结构中算子的计算顺序，使得整个结构只有最后一个算子为非线性计算。我们使用结构重参化解耦训练和测试，将训练阶段的多分支网络合并为单路结构，加快了推理速度并不会损失精度。不仅如此，我们还修改了原始的颈部网络，制定了新的信息流动路径和算子使用规则，引入注意力机制控制不同层融合的信息，提高了模型的检测精度。我们在标注数据集和实际数据集上进行测试，发现 YoloX-Lite 在更好地平衡了精度和速度之间的关系。最后我们针对不同的模块进行了消融实验，充分证明每个模块的有效性。

第四章　基于全局-局部信息聚合的视频目标检测算法

在前一章节中我们提出了静态目标检测模型 YoloX-Lite。在本章中，我们将该 YoloX-Lite 扩展为视频目标检测模型。我们分析了现有方法的不足之处，从实际应用角度出发，在静态目标检测模型的基础上，提出了一个聚合全局和局部时序信息的视频目标检测算法，使用易于获得的大量图片标注数据和少量视频标注数据进行训练，大幅提高了模型的检测精度。

4.1　研究动机

随着各领域智能化的推动，我们生活中很多场景都利用上了目标检测技术。但是在实际视频目标检测任务中，数据质量给检测器带来了很大的困难。监控视频数据由于编解码和丢包问题，很容易会出现相邻两帧图像肉眼观察完全一致，底层像素值实际上发生了变化；加之实际检测环境是十分复杂的，目标的运动以及背景的变化也会造成部分视频帧中目标模糊、姿态奇异等现象，极大地影响了检测器的工作。不仅数据质量带来了问题，数据采集也是一个亟需解决的问题。实际任务中，图片数据相对于视频数据标注成本小，也更容易采集，因此我们往往能收集到大量的图片标注数据和少量的视频标注数据。

现有的视频目标检测模型大多需要利用大量的视频标注数据进行训练，模型复杂，检测速度也不理想。静态目标检测模型现在参与研究者众多，发展成熟，每天都有科研工作者在改进，检测精度和速度都有保障。考虑到前述实际监控视频目标检测任务中的问题，工业领域更倾向于使用静态目标检测模型代替视频目标检测模型。但是静态目标检测模型将视频帧视为单张图像，图像之间毫无关联，忽视了视频帧之间的时间和空间相关性，相较于视频目标检测方法稳定性更差，检测精度也存在提高的空间。因此，本文从静态目标检测模型出发，设计了一个特征聚合模块，可以聚合长时范围内目标的外观信息和短时范围内目标的位置信息，明显提高了静态检测器的检测精度。

4.2 基于全局-局部信息聚合的视频目标检测算法

针对4.1描述的视频目标检测中存在的问题以及现有方法的不足之处，我们基于 Yolo 框架设计了一个能够聚合全局和局部时序信息的视频目标检测模型（Enhanced Yolo with Global and Local Information Aggregation, Yolo-GLA）。我们考虑到静态目标检测模型在实际应用中的优势，以发展迅速的 Yolo 框架为模型基础搭建 Yolo-GLA。同时为了解决静态检测器无法利用视频时空信息的问题，我们设计了一个即插即用的特征聚合模块，该模块可以聚合全局和局部的上下文信息，用于增强当前帧的特征。实验表明 Yolo-GLA 相较于当前静态目标检测算法精度更高，相较于视频目标检测模型速度更快，足以在实际任务中应用。

本节会详细介绍 Yolo-GLA 的整体设计，随后按照数据的流动方向依次介绍不同模块的功能，最后介绍模型的训练策略。

4.2.1 模型结构

模型的整体结构如图4-1所示。

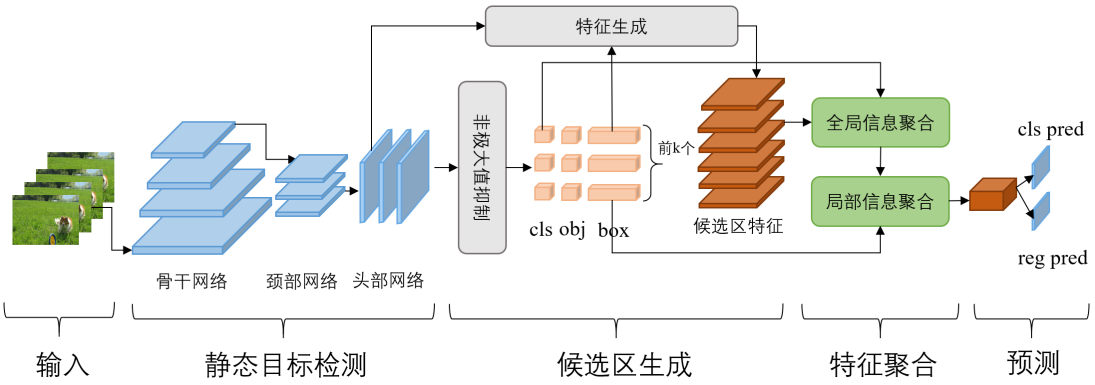

图 4-1: Yolo-GLA 网络结构

Yolo-GLA 包含以下几个模块：

1. 输入。Yolo-GLA 的输入是从一段视频数据中随机采样得到的图像帧，假设当前帧的时间为 t，局部帧从 $(t-n, t+n)$ 范围内随机取出，全局帧从局部帧范围之外，并且在整个视频之内随机取出。

2. 静态目标检测模块。静态目标检测模块主要用于获取所有可能存在目标的候选框，可以是基于 Yolo 框架的任意模型，有较高的自由度，这里我们

选择使用在第3章中介绍的 YoloX-Lite。YoloX-Lite 使用同一分布下的大量图片数据进行训练，从而获得初步的检测结果。在训练视频数据时，我们固定住 YoloX-Lite 的参数，仅训练后续的特征聚合模块，降低对视频标注数据的需求。在推理阶段，YoloX-Lite 使用结构重参化技术，合并多分支网络为单路网络，将多个算子操作合并为一个卷积层，使之更适合硬件推理加速。

3. 候选区生成模块。候选区生成模块（Proposal Generate Model，PGM）的主要作用是从静态目标检测模型的输出中，挑选出更可能存在目标的候选区。静态目标检测模块会输出特征图上每个位置可能存在目标的分数（obj）、目标对应的位置（box）以及目标所属类别的置信分数（cls），为了排除大量背景结果的干扰，同时降低整体的计算量，PGM 会从输出中挑选存在目标概率最大的 k 个结果。随后 PGM 会根据结果中对应的位置信息，从 YoloX-Lite 中提取出对应的特征图，生成候选框对应的特征。考虑到候选框大小不一致，会统一特征图的尺寸。

4. 全局-局部信息聚合模块。全局-局部聚合模块 (Global-Local information Aggregation，GLA) 的主要作用是聚合全局外观信息和局部位置信息，增强特征的表达能力。我们发现短时范围内图像帧差异太小，局部时序信息存在大量冗余，对模型效果提升有限。因此需要扩大聚合信息的范围，Yolo-GLA 直接使用整个视频中的任意图像作为全局信息聚合到参考帧中。我们将时序信息聚合分为两个阶段，第一个阶段利用全局时序信息增强局部时序信息，第二个阶段利用局部时序信息增强当前帧信息。两个阶段的聚合过程都采用自注意力机制，不同的是第一个阶段会使用候选框的类别置信度分数[75]，第二个阶段会额外引入候选框的位置信息[76]。实验表明仅聚合全局信息比仅聚合局部信息精度更高，分阶段聚合全局信息和局部信息精度更进一步，推理速度则会更慢。

5. 预测模块。预测模块的输入为经过特征聚合的特征向量，参考 YoloX 的解耦头设计，我们使用两个全连接层分别预测分类和边框位置信息。

4.2.2 框架设计思想

我们将视频目标检测的方法总结为两个核心问题：聚合哪些信息和以什么样的方式聚合。

对于聚合信息的范围。如图4-2所示，我们根据信息聚合的范围将常见的视频目标检测算法分为聚合局部信息、聚合全局信息和同时聚合局部-全局信息。局部信息可以理解为图像中的目标难以定位到候选框位置或者定位到的候

选框位置误差较大，需要根据局部几帧图像之间的变化来准确定位目标。 而

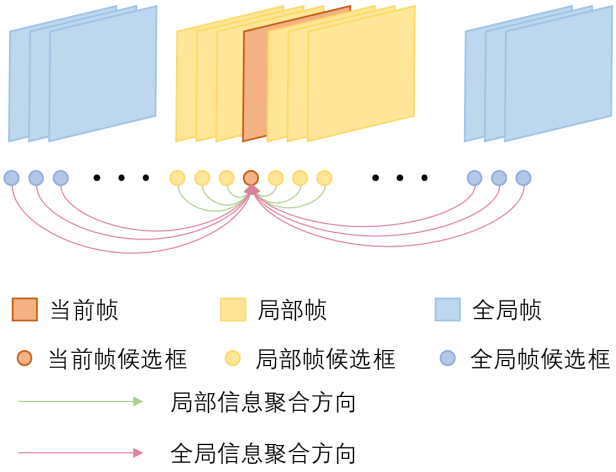

图 4-2: 其他方法信息聚合方式

全局信息可以理解为对于当前帧中低质量的目标，在长时范围内有个清晰的姿态和其在外观上十分相似，可以通过长时范围的清晰图像来增强当前帧的类别信息。然而局部信息通常包含的信息十分冗余，模糊的目标可能在短时间内一直无法被准确分类，仅仅聚合局部信息不能充分利用视频的信息。全局信息可以利用整个视频的信息来辅助增强当前帧的特征，但是当前帧的提取到的目标候选区误差过大，会导致聚合后的特征也难以预测到准确的位置信息。因此 Yolo-GLA 选择使用全局图像来增强局部图像，然后再使用局部图像增强当前帧信息，过程如图 4-3所示。

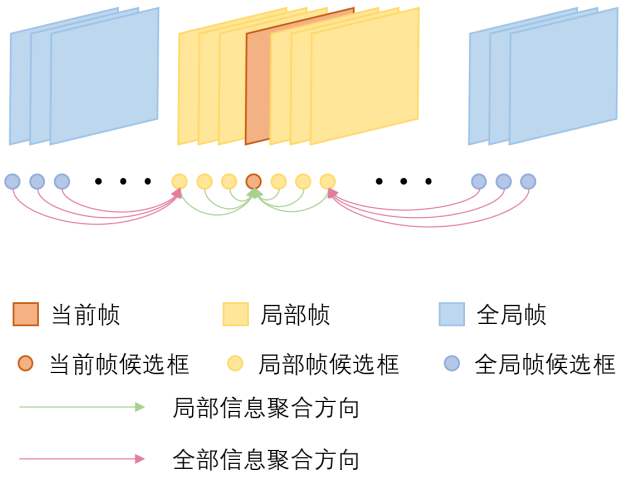

图 4-3: Yolo-GLA 信息聚合方式

对于以什么样的方式聚合。在相关工作一节我们介绍了很多聚合完整特征图的视频目标检测方法，由于目标的运动，不同帧中目标出现的位置有所差异，在信息聚合前需要对齐特征，通常采用光流场、可变形卷积等方法。这类方法限制参考帧的范围在 1 到 2 秒之内，否则位置差异过大，特征很难对齐。相反，基于候选框的聚合操作天然省略了特征对齐的步骤，只需要提取到准确的候选框即可，大幅扩大了信息聚合的范围，因此 Yolo-GLA 选择聚合候选框来提高模型性能。

从人的直观感觉来讲，使用长时间隔的图像来辅助分析当前帧图像时，我们往往是通过两张图像中目标外观的相似度来判断，而不是使用目标的位置信息。而使用短时间隔的图像时，我们不仅会从目标外观相似度来分析，而且会通过目标的之间的位置关系来判断当前帧中目标应该出现在哪个位置。基于这种思路，我们将聚合过程分为两个阶段，第一个阶段聚合全局图像的外观信息，第二个阶段聚合局部图像的外观和位置信息。

4.2.3 静态目标检测模块

通常候选区生成的过程总是出现在两阶段目标检测算法中，在 Yolo-GLA 中我们选择将整个单阶段目标检测模型作为候选框生成的前置网络，主要出于以下几个目的：

1.Yolo-GLA 已经将训练过程分为两个阶段：第一个阶段使用具有相同数据分布的图像数据集训练静态目标检测模型，第二个阶段固定静态目标检测模型的参数，仅仅训练后续的时序信息聚合模块。而两阶段目标检测算法也需要分阶段训练，使得整个 Yolo-GLA 训练过程变得更为复杂，因此选择使用单阶段目标检测来简化训练过程。

2. 我们最初的目标是在现有优秀的静态目标检测器的基础上，以最小的代价聚合时序信息，提高模型的检测精度。因此，我们希望在 PGM 选择部分候选框时，静态目标检测器已经具有初步的检测能力，提取到所有可能存在目标的候选框，并且置信度比较高。然而两阶段目标检测模型使用 RPN 网络提取到的候选框，置信度和位置信息都过于粗略，直接使用会遗漏真正的目标。我们最后选择使用成熟的 Yolo 系列的方法作为前置的目标检测器。

3.Yolo 作为工业界最受欢迎的方法，有大量的研究人员在着手改进，每个月都会迭代新的版本。使用 Yolo 作为前置的静态目标检测方法，让 Yolo-GLA 随时都能利用上最新的技术。同时基于 Yolo 框架的方法已经能在当前主流的推

理框架下部署，帮助 Yolo-GLA 能支持更多的推理框架。

基于上述原因，我们选择使用前述介绍的 YoloX-Lite 作为 Yolo-GLA 的前置网络。

4.2.4 候选区生成模块

图像数据在经过静态目标检测器后，会得到一组结果，包括目标是否为前景、位置信息以及目标所属类别置信度分数。未经过处理的检测结果中存在大量的背景框和重叠框，以图 4-4为例，我们得到了三个检测结果，分别记为 a、b 和 c，都是指向同一个目标。很明显，这三个检测结果应该只需要输出存在目标概率最大的检测框，也符合我们的直观感受。 我们选择使用非极大抑制算法

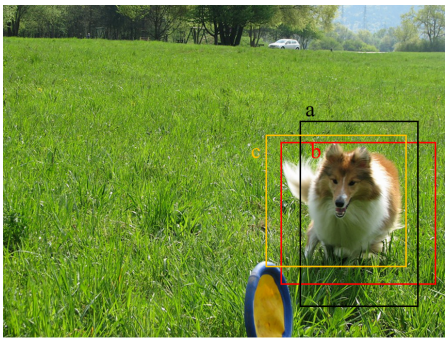

图 4-4: 检测结果重复

过滤掉大量冗余的结果，得到 n 个预测值，随后按照目标为前景的概率由大到小选择其中前 k 个预测结果，进一步排除低质量结果的干扰和降低后续过程的计算复杂度。这 k 个结果和 YoloX-Lite 头部网络生成的特征图一起送到特征生成模块（Feature Generate Module，FGM），特征生成模块结果如图 4-5。

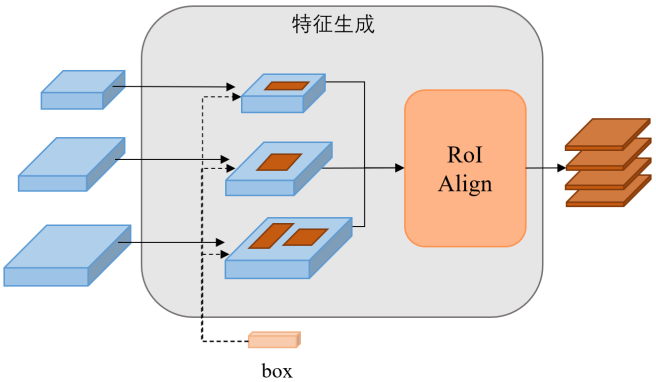

图 4-5: 特征生成模块

在 YoloX-Lite 头部网络中，颈部网络输出的三个尺度特征图已经被统一到相同的通道数，利用预测结果中的位置信息，可以裁剪出候选框对应的特征图。我们在静态目标检测算法中介绍过，头部网络采用分而治之的思想，不同大小的目标由不同层次的特征图进行预测，深层特征图不会负责预测小目标，因此提取到的特征通常也不是仅仅一个像素点。裁剪得到的特征图一般情况下分辨率都是不一致的，我们使用 RoI Align[55] 算法代替 RoI Pooling[23] 将所有的候选框统一到到相同的分辨率。RoI Pooling 将特征图划分为 $n \times n$ 个区域，每个区域内使用平均池化，RoI Pooling 在划分区域的过程中，会丢失边缘的部分信息，影响整体分类和定位效果。而 RoI Align 在划分区域时，并不关心是否区域是否落在了特征图中的像素点上，对于每个区域内的值，RoI Align 会使用双线性插值重新生成，然后再使用最大池化。整个候选框生成过程类似于两阶段目标检测算法 Faster R-CNN 中 RPN 网络。

4.2.5 全局-局部信息聚合模块

Yolo-GLA 参考人脑处理视觉信息的方式，设计了全局-局部信息聚合模块。对于当前帧的模糊目标，人往往会参考其他图像帧中的目标来辅助识别。一方面人类会从间隔较长的图像中搜索是否存在外观相似的目标来判断当前帧中目标的类别，另一方面人类会从相近的几帧中找到相对位置发生变化的目标来判断当前帧中目标的大致范围。而 Yolo-GLA 也是将信息聚合分为两个阶段。第一个阶段中，Yolo-GLA 只聚合全局外观信息，将全局帧外观信息聚合到局部帧图像中。第二个阶段 Yolo-GLA 聚合局部位置信息，将局部帧信息聚合到当前帧中，这样就完成了整个时序信息聚合过程。

信息聚合的方式通常是使用注意力机制计算多个特征之间的相似度，根据相似性度量加权求和多个特征。视频目标检测中常用欧几里得距离相似度、余弦相似度和键值对注意力机制来作为相似性度量的计算准则。欧几里得距离相似度只关注两个特征之间数值的绝对差异，余弦相似度相反更加关注于两个特征的方向，对特征具体数值的绝对值大小并不敏感。近年来，键值对注意力机制由于更加优秀的性能，受到研究者们的欢迎，在自然语言处理和计算机视觉任务上大放异彩。Yolo-GLA 在键值对向量的基础上引入外观信息和位置信息，分别用于聚合全局特征和局部特征。

首先介绍全局外观信息聚合。如图4-6，定义 f 帧图像经过候选区生成模块提取到的特征为 $P = \{P_1, P_2, \ldots, P_f\}$，其中 $P_i \in \mathbb{R}^{d \times k}$，$d$ 为特征的维度，k 为

每张图像提取的候选框数量，P 表示目标自身的颜色、大小、形状等特征。$C = \{C_1, C_2, \ldots, C_f\}$，其中 $C_i \in \mathbb{R}^{d \times k}$，$C$ 表示静态目标检测器预测目标所属类别的置信度，我们将其复制扩充至和候选区 P 相同的维度。 为了让模型关注多个

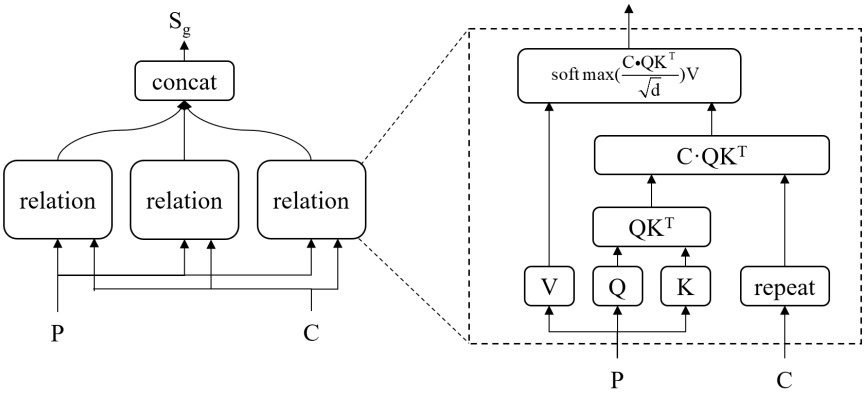

图 4-6: 全局信息聚合

方面的信息，我们将输入向量按照通道划分成多个子空间，每个子空间内计算候选区特征之间的相关性（relation），最后将不同空间的结果拼接（concat）起来得到最后的输出向量 S_g。在计算候选区特征两两之间的相似度分数时，全局信息聚合会使用映射函数将原始候选区特征映射为任务向量 Q、键向量 K 和值向量 V，根据公式4-1计算任务向量和键向量之间的相似性，

$$R1 = QK^T. \tag{4-1}$$

在静态目标检测器中，我们已经预测过一个目标所属类别概率，这个数据可以作为参考信息辅助视频目标检测，我们通过点积形式将类别置信度融入到相似度分数中，

$$R2 = C \cdot (R1). \tag{4-2}$$

当维度过高时，最终的相关性分数会比较大，softmax 计算会趋向梯度平缓区域，使得网络收敛困难。因此需要在 softmax 前除以 $\sqrt{d}$ 进行降维，降低维度对相似度分数的影响。最后根据公式4-3将计算得到的相似性度量和值向量 V 加权求和，即得到了聚合全局外观信息的特征向量 S_g，

$$S_g = softmax(\frac{R2}{\sqrt{d}})V. \tag{4-3}$$

局部位置信息聚合和全局信息聚合计算过程比较相似，流程如图4-7所

示。不同之处在于局部位置信息聚合除了候选框特征之外，还需要引入静态目标检测预测的目标位置坐标信息计算位置权重 ω_G。局部位置信息聚合模块的

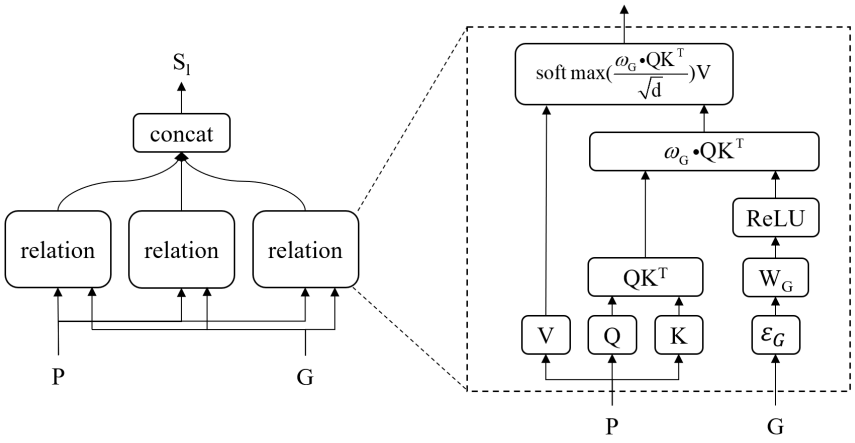

图 4-7: 局部信息聚合

输入是静态目标检测预测的位置坐标 (x, y, w, h)，(x, y) 表示目标的中心点坐标，w 和 h 表示目标的长和宽。由于目标的坐标值变化范围较大，同时多帧之间目标的位置差异也十分明显，直接使用原始坐标会导致训练不稳定，因此将坐标变换为：

$$\begin{aligned} \delta x &= \log(\frac{|x_i - x_j|}{w_i}), \\ \delta y &= \log(\frac{|y_i - y_j|}{h_i}), \\ \delta w &= \log(\frac{w_j}{w_i}), \\ \delta h &= \log(\frac{h_j}{h_i})), \end{aligned} \tag{4-4}$$

i 和 j 表示第 i 个候选框特征和第 j 个候选框特征，（δx，δy，δw，δh）分别表示转换后的目标中心点坐标和对应的检测框的长宽值。到目前为止，位置信息的维度为 $\mathbb{R}^{k\times 4}$，我们借鉴 RDN[76] 一文中的位置嵌入方法，将之前得到的位置信息映射到高维空间中，如公式4-5所示：

$$\delta = [\sin(\frac{\delta \times 100}{1000^{\frac{i}{8}}}), \cos(\frac{\delta \times 100}{1000^{\frac{i}{8}}})], i \in [0, 7], \tag{4-5}$$

其中 $\delta \in [\delta x, \delta y, \delta w, \delta h]$。接着使用一个全连接层和非线性激活函数，得到权重参数 ω_G。将 ω_G 作为位置信息融合到注意力机制中，结合值向量 V 加权求和，

得到最后同时聚合全局信息和局部信息的特征向量 S_l，

$$S_l = softmax(\frac{\omega_G \cdot (QK^T)}{\sqrt{d}})V. \tag{4-6}$$

4.3 实验与分析

4.3.1 实验数据分析

我们选择使用 ImageNet Det 作为基础的静态目标检测的训练集，使用 ImageNet VID 作为视频目标检测的训练集。

（1）ImageNet Det

ImageNet Det 是 ImageNet 大规模视觉目标检测比赛提供的数据集，包含 200 类目标。ImageNet DET 中目标出现的背景包含家庭，街道，城市，农场等常见场景，目标也是从实际场景中拍摄得到。ImageNet DET 包含 ImageNET VID 中出现的所有类别目标，我们从 DET 数据中提取和 VID 类别相同的数据，并划分为训练集和验证集，数据量见表 4-1。

表 4-1: ImageNet DET 数据集组成

	训练集	验证集
图片数量	118287	5000
标注数量	844701	36683

（2）ImageNet VID

ImageNet VID 作为视频目标检测使用最广泛的数据集，包含 30 个类别目标。其中包括视频数据和预先将视频切分成图像帧数据。通常一段视频中仅包含一类目标，并且存在失焦、遮挡和姿态奇异等现象，体现了视频目标检测的难点。VID 数据集将整个数据集划分为训练集和验证集，数据量见表 4-2，对应到每个类别目标的数量分布见图 4-8。VID 数据集同时按照目标的运动速度，将其划分为运动快速（fast）的目标、运动速度中度（medium）的目标和运动缓慢（slow）的目标，分别占比 18.2%，34.7% 和 37.1%。

表 4-2: ImageNet VID 数据集组成

	训练集	验证集
视频数量	3862	555
图片数量	1122397	176126

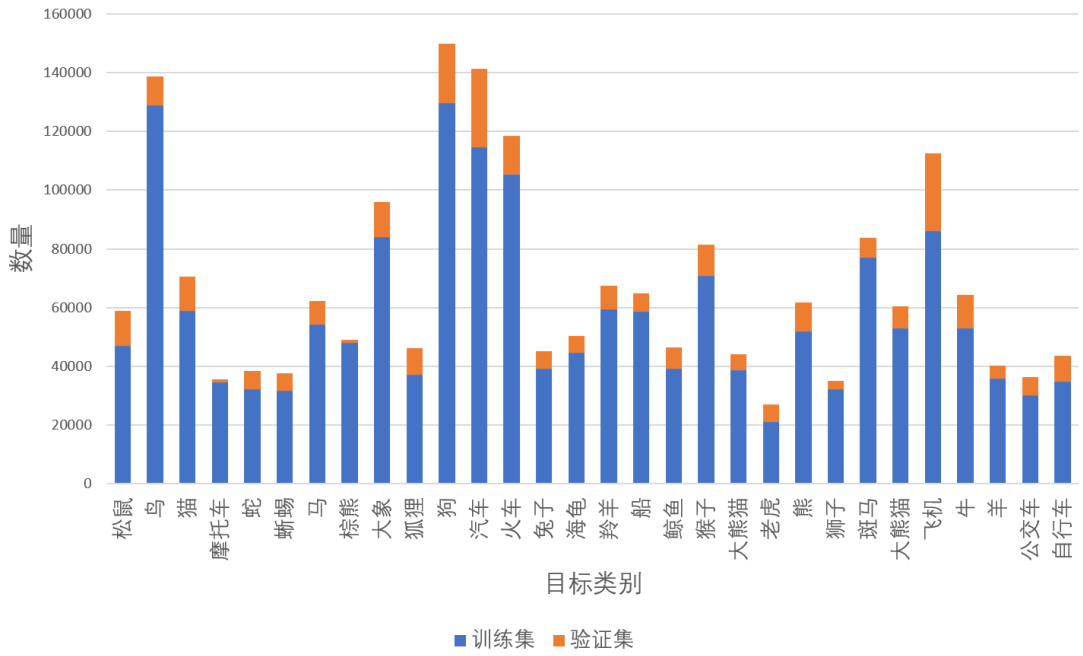

图 4-8: VID 数据集中不同类别目标对应的数量分布

4.3.2 实验评价指标

视频目标检测使用的评价指标为 mAP50 等指标，我们在 3.3.2中已经介绍过 mAP50 具体的计算过程。不仅如此，在视频目标检测中，mAP 会根据目标移动速度分为 mAP_{fast}、mAP_{medium} 和 mAP_{slow}，用于评估不同移动速度下目标的检测精度。不同移动的速度使用当前帧和前 10 帧之间的 IoU 差异分类，如果 IoU 值大于 0.9，则定义当前目标的移动速度为 slow；如果 IoU 在 [0.7,0.9] 区间内，则定义当前目标的移动速度为 medium；如果 IoU 小于 0.7，则定义当前目标的移动速度为 fast。

4.3.3 实验环境和训练策略设置

实验训练和测试硬件平台如表4–3所示，使用到的推理软件 Pytorch 版本为 0.9，编程语言 Python 版本为 3.9，显卡驱动版本为 455.32，并行计算库 cuda 版本为 11.1。

表 4–3: 实验硬件平台配置

配置	参数
操作系统	Linux
CPU	Intel(R) Xeon(R) E5-2678 v3
GPU	RTX 2080ti * 4
内存	252G

我们使用提取后的 ImageNet Det 数据集训练 YoloX-Lite 网络，整个训练过程中参数的设置与上一章中介绍的训练 YoloX-Lite 一致。在训练 Yolo-GLA 时固定 YoloX-Lite 网络参数，只训练后续的特征聚合模块。对于 ImageNet VID 数据，我们从中每段视频中随机采样 10% 的数据作为训练集，避免训练大量重复数据造成训练不稳定。我们设置候选框选择的数量 $k = 30$。在全局帧和局部帧选取数量上，设置 $n = 32$，我们随机采集 3 帧距离参考帧 32 步以内的图像，随机采集 12 帧距离参考帧大于 32 步的图像。我们设置训练 Yolo-GLA 的时间为 10 轮，前 8 轮使用数据增强方法，最后 2 轮训练关闭数据增强方法。和 YoloX-Lite 一致，我们使用 SGD 作为优化器，使用余弦学习率调整方案调整每轮训练的学习率参数。候选区选择模块中我们设置 NMS 过滤阈值为 0.75，在最后的后处理中设置 NMS 的过滤阈值为 0.5。对于输入图像的分辨率，我们随机将输入图像缩放至 (352×352) 到 (672×672)，以增强模型的泛化能力。在测试时，图片会统一缩放到 (576×576) 并且使用全部图像进行测试。

4.3.4 对比实验

在本节中，我们对比了 Yolo-GLA 和基础的 YoloX-Lite 在 ImageNet VID 数据集上的效果，训练过程中使用到的数据和采样方式一致，结果见表4–4。可以从中发现，Yolo-GLA 相较于静态检测模型 YoloX-Lite 的检测效果有着比较明显的提高，随着目标运动速度越快，提升越明显。结果符合我们对静态目标检测模型的预期，在出现运动模糊的情况时，视频目标检测模型能更好地利用

上其他帧的时序信息辅助当前帧的检测。同时 Yolo-GLA 由于多了聚合信息的步骤，耗时相较于 YoloX-Lite 增加了 3.41ms。

表 4-4: Yolo-GLA 和 YoloX-Lite 在 ImageNet VID 上的检测效果

模型	mAP50	mAP_{slow}	mAP_{medium}	mAP_{fast}	Times(ms)
Yolo-GLA	**78.7**	**86.2**	**80.1**	**65.3**	13.2
YoloX-Lite	70.8	81.5	72.6	56.1	**9.79**

图4-9展示了我们的方法和 YoloX-Lite 在 COCO 数据集中的代表性结果，图中左上角数字表示图像位于视频中的第几帧。其中第一行使用我们的方法检测，用红色框表示检测结果，第二行使用 YoloX-Lite 检测，用黄色框表示检测结果。我们可以从中观察到，对于第 54 和第 65 帧，目标呈现清晰的姿态时，我们的方法和 YoloX-Lite 均能检测到目标。而对于第 80 帧，目标进入阴影中，姿态已经模糊时，我们的方法仍然能够检测到目标，而 YoloX-Lite 此时已经检测失败。

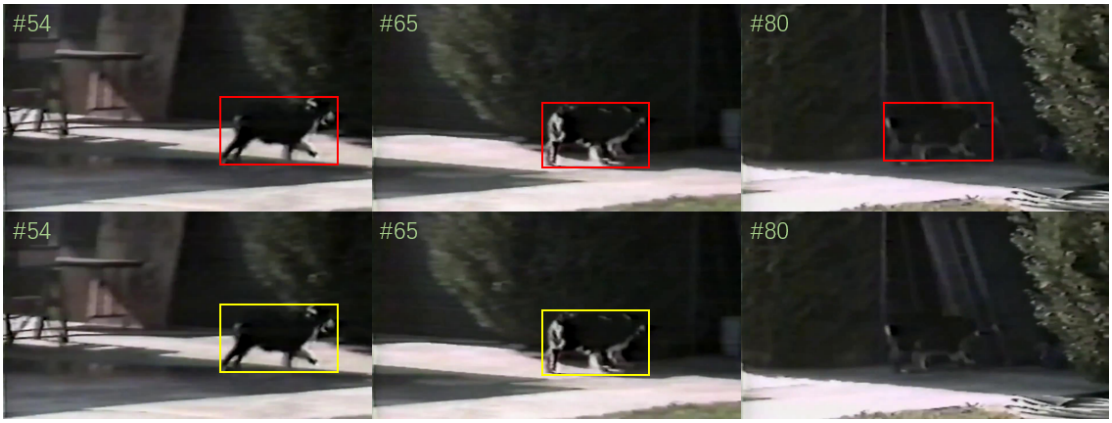

图 4-9: YoloX-GLA 和 YoloX-Lite 在 ImageNet VID 数据集中的代表性结果

我们同时验证了当前主流方法在 ImageNet VID 数据集上的效果，实验结果如表4-5所示，表中所有模型均没有使用后处理方法，并且骨干网络为 R101 表示使用 ResNet101 作为骨干网络，X101 表示使用 ResNeXt101，RepCSP 表示使用 Rep-CSPDarknet，CSP 表示使用 CSPDarknet。在计算网络推理速度时，未做说明的网络均使用 2080ti 计算推理速度，T 表示使用 Tian V 显卡计算推理速度，V 表示使用 Tian V 显卡，K 表示使用 K40 显卡，H 表示使用华为 Mate8 手机。

表 4-5: 主流模型在 ImageNet VID 上的检测效果

模型	骨干网络	mAP50	Times(ms)
Yolo-GLA	**RepCSP**	**78.7**	**13.2**
THPM[41]	Mobilenet[77]	60.2	25.6(H)
DFF[39]	R101	73.1	20.3(K)
R-RCN[66]	R101	73.9	4.1(K)
FGFA	R101	76.3	104.2
YoloV-S	CSP	77.3	11.3
SELSA	R101	81.8	94.3(V)
MEGA	R101	82.9	230.4
TROIA[78]	X101	84.3	285.7
MAMBA[79]	R101	84.6	110.3(T)

从表格中我们可以发现，主流的大部分视频目标检测算法的推理速度十分缓慢，即使使用昂贵的 2080ti 或者 TITAN V 显卡，SELSA 和 MEGA 等方法仍然不能实现实时目标检测，而 Yolo-GLA 依靠快速的前置网络实现了在 2080ti 上单张图片的推理速度为 13.2ms，足以应用在实际任务中。相较于检测速度较快的 THPM、DFF 等模型，Yolo-GLA 在检测精度上也有着明显的优势。

4.3.5 敏感性分析

（1）候选区数量

我们在本节中探索候选区生成模块中输出候选区数量 k 对 Yolo-GLA 的影响，受到显卡存储空间的限制，我们将 k 的数值从 10 调整到 75，实验结果如表 4-6所示。

表 4-6: 候选区数量 k 对 Yolo-GLA 的影响

k	10	20	**30**	50	75
mAP50	77.4	78.3	**78.7**	78.9	79.0
Time	11.6	12.2	**13.2**	14.8	20.5

从上表中的实验结果可以看出，随着 k 值增大，Yolo-GLA 的检测精度也在逐渐提高，检测速度在逐渐变慢。这符合我们的普遍认知，随着候选区生成

模块输出的候选区越多，包含真实目标的概率也就越大，因此 mAP50 也会提高，同时由于信息聚合模块需要处理更多候选区，检测时间也会相应增加。当候选区增加到一定值时，绝大部分优质的候选框已经被提取到了，继续增大 k 值对结果影响并不明显，相反还会由于过多的候选框造成检测时间大幅增加。我们最后选择 $k = 30$ 实现推理速度和检测精度的平衡。

（2）全局帧和局部帧数量

我们在本节中探索信息聚合模块中全局帧和局部帧的占比对模型的影响。受到显存的限制，预先设置全局帧加上局部帧的总数为 16，实验结果如表 4-7所示。

表 4-7: 全局帧和局部帧数量对 Yolo-GLA 的影响

全局帧数量：局部帧数量	16:0	14:2	12:4	8:8	0:16
mAP50	78.0	78.2	**78.7**	77.4	75.6

在上表的实验中，我们逐渐减小全局帧的数量，增加局部帧的数量，当局部帧数量为 0 时，即只聚合全局信息，当全局帧数量为 0 时，即只聚合局部信息。我们发现随着全局帧数量逐步减少，mAP50 指标先增加后减少，直到只聚合局部信息。全局信息越多，模型能聚合更加广泛的信息，有助于增强模型的泛化能力，但是全局信息无法提供更加精准的定位信息，这时，适当增加局部帧，可以帮助模型获取更加精准的目标位置特征。同时，还可以发现，仅仅聚合全局信息明显比只聚合局部信息 mAP 值更高，因为当目标出现遮挡或者运动模糊时，很可能局部帧中的目标大多是遮挡或者运动模糊的，并不能给目标提供有益的帮助。

4.4 本章小结

本章我们从视频目标检测模型开展研究，分析了现有方法的可改进之处，提出了聚合全局和局部信息的视频目标检测模型 Yolo-GLA。Yolo-GLA 在 YoloX-Lite 的基础上新增了候选框生成模块，用于从 YoloX-Lite 的检测结果中生成高质量的可能包含目标的特征图，同时也降低后续的计算量。随后设计了一个聚合全局外观信息和局部位置信息的模块，能够更加高效聚合视频时序信息，提高检测模型检测精度。我们在标准数据集上进行测试，发现仅聚合全局

信息的情况下，Yolo-GLA 能够达到更快的速度和可靠的检测精度，同时聚合全局信息和局部信息则能达到最好的检测精度。最后我们针对部分超参数进行了敏感性分析实验，充分证明每个模块的有效性。

第五章　鸟类监控视频目标检测系统

我们在前述几章中介绍了一个可以应用于实际场景的视频目标检测方法：Yolo-GLA。本章会介绍一个实际的鸟类监控视频目标检测的案例，我们在这个案例中以 Yolo-GLA 为基础设计了一套完整的鸟类监控视频目标检测系统。

5.1　系统研发背景

鸟类的发现和统计一直是环境保护的一个重要内容。通过记录鸟群种类和数量、鸟类出现时间以及停留时间，如图 5-1所示，可以有效反映鸟群的当前健康状态，进一步还能分析出保护区中的生态环境现状，方便专业人员开展环境保护工作。 传统情况下，总是需要环境保护人员携带录像设备身临保护区，

赤麻鸭　　黑水鸡

图 5-1: 两种环境保护鸟类

人工拍摄目标后，再由专家对视频中的鸟类分类和计数。这样的方法需要花费大量时间拍摄和处理视频数据，耗时耗力，工作量很大。因此我们可以利用目标检测算法，直接部署在边缘硬件设备中，自动检测保护区中出现的鸟类，并将结果反馈给服务中心，进行下一步的数据统计和分析，降低环保人员的工作量，节约人力成本。

目前市面上的监控视频监测系统基本都是依赖 Yolo 系列的静态目标监测方法，并且部署一次就再也不会更新。然而鸟类监控视频由于现实环境的影响，

容易出现失焦、运动模糊等现象，在这些情况下静态目标检测器的检测效果并不稳定。同时对于珍稀鸟类的采集是一个长期工作，难以在短时间内采集到大量的数据，要求非人工智能领域的用户可以不断采集数据，重新训练模型并且更新参数。考虑到现有系统的一系列问题，我们设计了一个从数据标注到自动化训练和测试，再到边缘部署的完整鸟类监控视频目标检测系统。

5.2 系统设计

在本节中，我们首先分析系统需求，总结每个需求的难点以及需要达到的效果，随后针对需求设计一个完整的系统架构。

5.2.1 系统需求设计

明确系统的功能需求是开发软件系统的第一步，准确的需求可以帮助开发人员免于在错误的方向上投入大量的时间。下面我们梳理了鸟类监控视频目标检测系统的关键需求：

1. 数据标注。用户不论是从互联网上下载，还是从环境保护区中拍摄得到的鸟类图像和视频，都是不包含标注信息的原生数据。而模型在训练过程中需要图像和视频数据准确标注出所有出现过的鸟类种类，以及其出现的位置坐标。现存的标注工具需要额外配置运行环境，对于非专业人员来说使用起来相当麻烦。因此需要在系统中集成数据标注功能，简化标注流程。

2. 数据分析。无论是用户采集到的训练数据，还是本系统自动检测到的数据，都需要一套自动化的数据分析工具，统计每种鸟类的数量、分布区域和出现时间等等重要信息，辅助专家进行环境保护。

3. 模型训练和测试。实际鸟类数据采集相当困难，尤其是珍惜鸟类很难拍到，导致最初用于训练的数据集规模很小，并且分布不平衡。因此要求系统可以不断接收新采集到的训练数据，用于迭代训练目标检测模型，增强模型的检测能力。

4. 模型管理。在迭代训练目标检测模型的过程中，我们积累了大量的训练结果，适用于不同情况。考虑到存储空间的限制，需要筛选出合适的模型参数，存储在数据库中。因此需要一个模型管理功能，人为选择模型参数是应用、保留或者删除。

5. 模型部署。保护区内环境恶劣，基础设施有限，适合部署低算力的边缘

计算设备。要求我们将实验环境下训练出来的模型转换为可以在边缘设备上的部署的格式，同时建立边缘设备和中心服务器之间的连接，可以完成模型参数和检测数据的交互。

我们根据实际情况提取出了以上 5 点重要需求，符合用户的使用习惯和主观感受。

5.2.2 系统架构设计

在明确了本系统需要完成的目标后，我们统计了备选的技术方案，选型合适的硬件设备和软件框架，设计出了一个符合的系统架构。整体架构如图 5-2所示，我们将整个系统分为边缘侧和中心服务侧，边缘侧用于部署目标检测算法，并完成实际检测，而服务侧完成和用户交互的各种操作。接下来会依次介绍不同模块的作用：

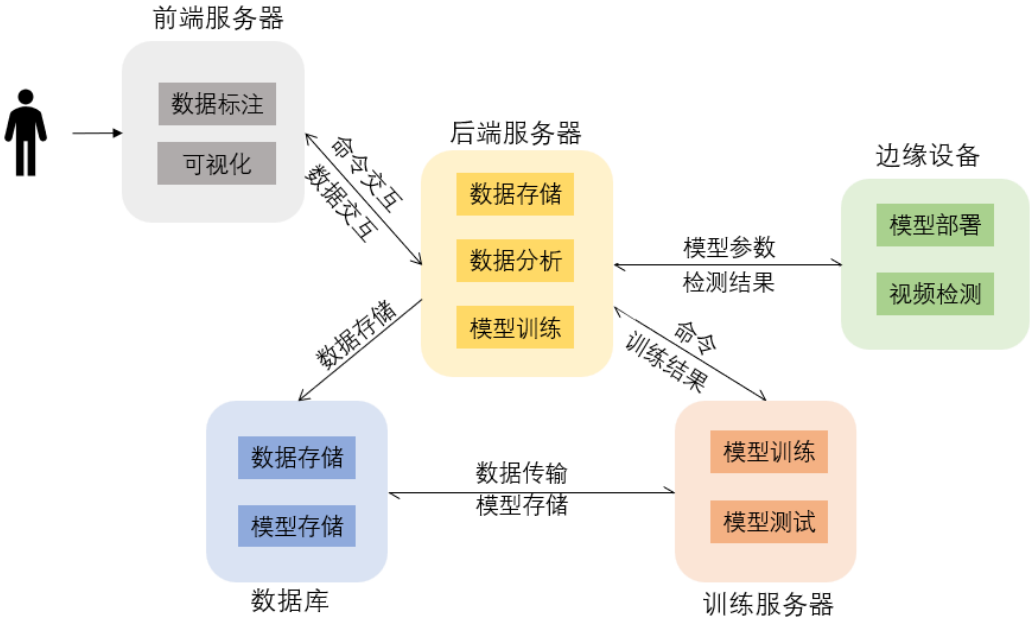

图 5-2: 鸟类监控视频目标检测系统架构

1. 前端服务器。前端服务器主要部署前端服务框架，提供用户可视化操作界面。前端服务器只和后端服务器之间存在交互，包括命令交互和数据交互。除了数据标注功能在前端服务器中实现以外，其他功能在前端服务器中仅仅起到命令传递和可视化的作用，内部的实现逻辑由后端服务器负责。

2. 后端服务器。后端服务器是整个架构的调度中心和功能实现中心。所有的数据都是由前端服务器传递给后端服务器，随后转发给数据库服务器、训练

服务器、边缘设备等等。绝大部分的功能都是由后端服务器完成调度，甚至模型训练也是由后端服务器完成前期准备工作。考虑到后端服务器会承担相当大的通信压力和计算压力，因此后端服务器需要用到高性能的 CPU 处理器。

3. 数据库。数据库服务器主要用于管理数据、训练好的模型参数以及一些统计信息。考虑到未来数据采集的数量会达到百万级，甚至是千万级，我们选择处理速度更快的非关系型数据库作为数据管理框架。同时，为了应对单机设备故障事件，我们采用主从备份技术，选择一个主机，两个备份机，所有的数据库服务器都是大存储容量的设备。

4. 训练服务器。训练服务器主要用于提供模型训练和测试需要用到的硬件环境。在本系统中，我们选择大显存的 GPU 设备，提高训练速度。训练服务器的训练由后端服务器启动，同时训练的所有中间结果都会及时回调给后端服务器，训练中用到的数据则会请求数据库得到。

5. 边缘设备。边缘设备主要部署在野外环境下，并且可以提供检测算法需要的算力支持。视频目标检测算法 Yolo-GLA 会提前部署在边缘设备中，需要用到的模型参数由后端服务器发送而来。边缘设备会利用内置的算法解析摄像设备传输的视频流信息，随后将检测到的鸟类图像连同检测结果及时反馈给后端服务器。注意，除了边缘设备，其他服务器均部署在非环境保护区内。

5.3 系统实现

5.3.1 系统开发环境

鸟类目标检测系统是一个集合可视化操作界面、数据管理、神经网络模型训练和边缘模型部署服务于一体的大规模系统。本节会依次介绍不同服务器的开发环境和技术选型。

从设备所处位置来看，我们将其划分为边缘侧和中心侧。边缘侧包括架设在保护区中的摄像头和部署了视频目标检测算法的低算力设备，这里选择使用 Nvidia 公司出品的 Jetson TX2，内置系统为 Ubuntu 18.04。为了保证模型部署阶段的精度和训练得到的效果一致，我们选择使用 LibTorch 推理框架部署视频目标检测模型。选择 RTSP 协议、OpenCV 和 FFmpeg 工具拉取来自摄像头的视频流数据，并通过消息队列 ActiveMQ 将检测好的结果传递给中心侧。整个边缘侧都是采用 C++ 语言开发完成，最大化模型的检测速度。

从用户交互角度来看，我们将中心侧服务器划分为前端和后端。前端以

React 框架和 Ant-Design 组件库搭建，使用 JavaScript 语言开发。而后端则是以 Python 语言为主，具体的，采用了 Flask 框架，以及 Celery 等工具。前端和后端都是计算密集型服务，我们选择使用 Intel i7-12700 作为处理器，并配置 256G 固态硬盘提供中间缓存。

从数据存储角度来看，我们将设备分为数据存储型设备和数据计算型设备。数据存储型设备指的就是数据库服务器，我们选择使用非关系型数据库 Redis。这是考虑到未来数据容量上升到百万级甚至是千万级时，关系型数据库和非惯性型数据库之间数据处理速度之间的巨大差异。同时，为了避免单机数据库服务器损害导致灾难性的后果，我们增加了主从备份的功能。所有数据库服务器都使用大容量的机械硬盘而不是固态硬盘。

由于目标检测模型训练时要求占用服务器中绝大多数计算资源，将该功能继承到后端服务器中会导致整个系统卡死，无法处理其他任务。因此我们将模型训练过程单独放置在一个训练服务器中，提供模型训练需要的环境。在本系统中，训练服务器由 2 张型号为 Nvidia RTX 2080ti 的显卡组成，模型训练使用到的框架为 Pytorch。

5.3.2 系统功能模块

5.2.1一节介绍了本系统需要完成的主要几点需求，我们在本节中会具体介绍其中的模型训练和测试、模型管理以及边缘部署三个较复杂功能的实现原理，并通过流程图展示其中的逻辑。

1. 模型训练和测试，整个模型训练和测试流程如时序图5-3所示。用户发起训练命令后，后端服务器开始准备训练要求的数据编号，随后将编号和训练启动命令发送给训练服务器。训练服务器根据接收到的数据编号从数据库中请求到具体的图片和对应的标签，并进行数据预处理。完成训练准备工作后，训练服务器通知后端服务器训练已经开始，可以断开连接了。在迭代训练过程中，我们设置了一个定时器，定期向后端发送训练的中间结果，并反馈给用户观察。当整个训练结束后，会进入测试阶段，验证训练结果，最后将训练的中间结果信息和训练得到的参数信息存储到数据库中存储起来。

2. 模型管理，模型管理流程如时序图5-4所示。用户在进入模型管理页面后，前端系统会自动发送查看现存模型的请求，后端会将该请求转发给数据库，拿到模型相关的所有数据信息显示给用户观看。用户如果选择删除模型，则会由后端向数据库发送请求，删除对应的模型数据。如果选择应用模型，后

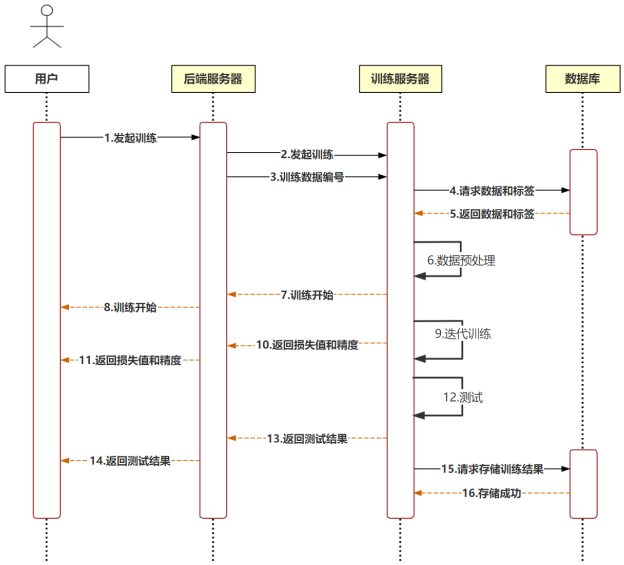

图 5-3: 模型训练和测试时序图

端服务器则会直接将模型发送给所有边缘设备，由边缘设备自行完成模型参数升级。边缘设备成功升级后，会反馈给后端模型更新成功的信号。

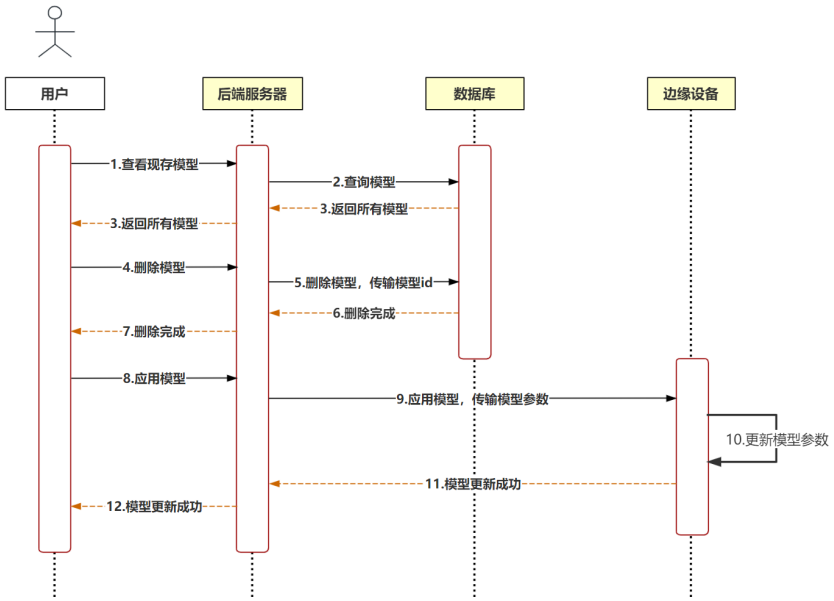

图 5-4: 模型管理时序图

3. 边缘部署。在边缘设备部署之前，我们已经在其中预置了目标检测模型和配套的辅助工具，模型需要用到的参数则是由后端服务器发送过来。边缘设备和后端服务器之间的数据交互全部以消息队列的方式传递，并且设置好

消息重传、消息缓存等配置，不会出现信息丢失的问题。位于保护区内的摄像头将拍摄到的视频信息按照 RTSP 协议的格式传递给边缘设备，边缘设备利用 OpenCV 和 FFmpeg 工具从 RTSP 视频流中解析出每一帧图像，经过图像预处理后传递给视频目标检测模型进行检测。如果当前帧不存在目标，则检测下一帧，反之，将该图像和检测结果通过消息队列反馈给后端服务器进行下一步数据分析，标注上目标所处的位置以及出现等信息。

5.4 系统结果展示

在本节中我们会通过可视化界面演示鸟类检测系统中部分的核心功能，所有界面都通过浏览器访问和登录。

图 5-5展示了数据上传时的界面。用户可以点击左上角的区域，选择文件或者文件夹路径，将其上传到服务器中，也可以将文件直接拖拽到对应区域内完成上传，支持常见格式的图片和视频数据。选择好对应的文件后，需要填写拍摄时间、拍摄地点等信息，方便后续数据管理。最后点击开始提交按钮即可完成上传，上传完成的数据会在下方显示对应的上传信息。注意如果上传数据量过大，需要等待一段时间才能在下方显示。

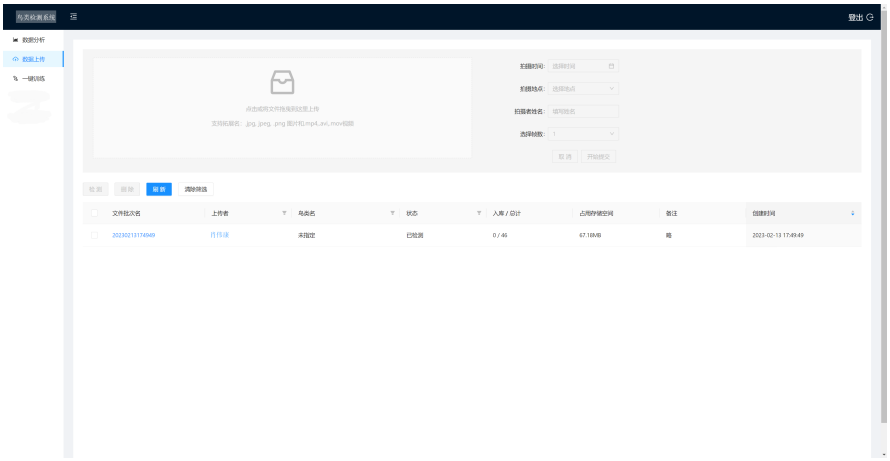

图 5-5: 数据上传界面

图 5-6展示了数据标注时的界面。用户在数据上传完成后，点击对应下方对应批次的编号，即可进入数据标注页面。页面中右边栏包括该批次的所有图片数据，选择任意一张图像即可以大图形式显示在左侧。右侧栏上方提供了图像显示规格和当前批次的数据信息。选择好图片后，使用鼠标在图像中框住目标，并选择对应的目标类别，选择确认修改后就完成标注。

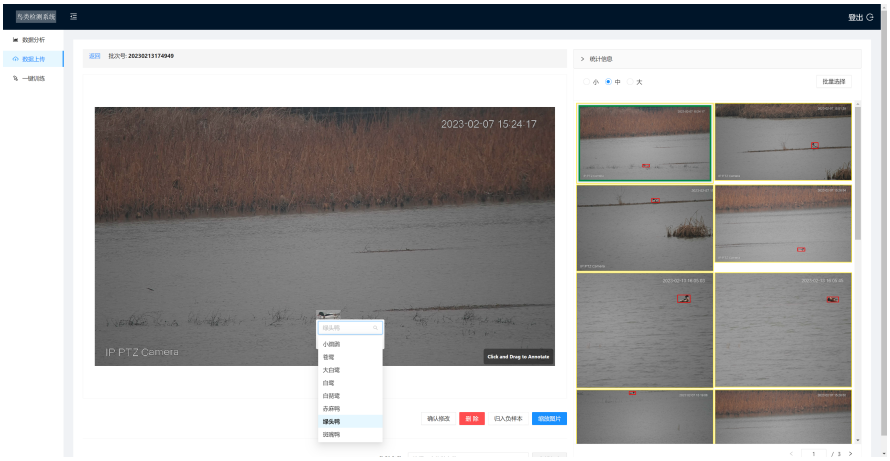

图 5-6: 数据标注界面

图5-7展示了模型训练时的界面。用户在点击右上角一键训练按钮后，服务器开始准备视频目标检测模型训练，并将训练过程中的各种指标数据映射到界面中心位置。用户可以根据损失函数的变化曲线，判断训练状态。上边栏反映了当前训练服务器的状态信息，包括显存占用量、显卡温度等等，辅助用户检测硬件设备和安排训练时间。模型训练完成后会在下边栏显示，包括该模型的训练和测试时间、状态、测试指标、以及是否部署在边缘侧。我们提供了几个按钮，用于管理模型，包括测试、保存、替换以及删除，测试指的是使用当前的测试集测试模型的准确率，保存指的是存储到数据库中，替换指的是更新边缘检测模型、删除指的是从数据库中删除选中的模型。

图 5-7: 模型训练界面

图5-8展示了边缘设备的检测结果，该功能位于另外一个页面中。图中左边栏用于选择具体的设备，右边栏显示当前摄像头的相关信息，包括摄像头所

处位置、使用到的焦距和光圈数值，用户可以通过按钮更改焦距和光圈，以及移动摄像头的视角。实时检测结果会显示在页面中心。

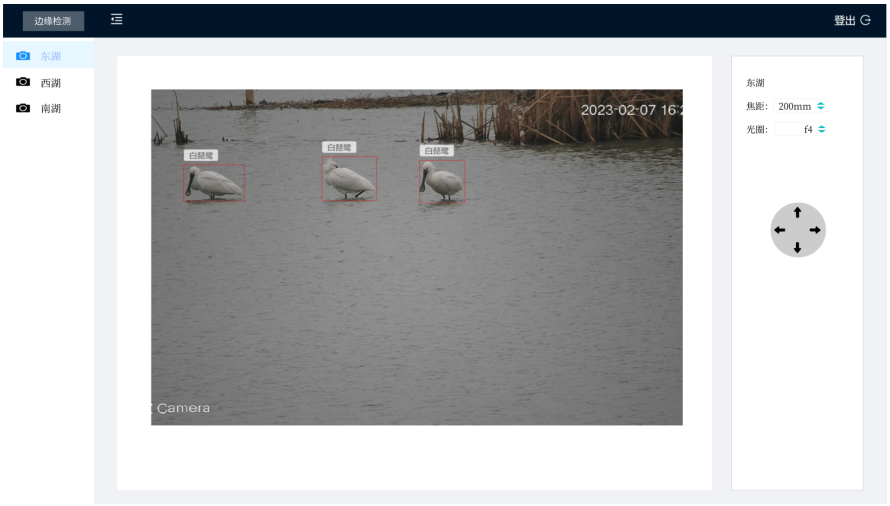

图 5-8: 边缘检测界面

5.5 本章小结

本章我们详细介绍了鸟类监控视频目标检测系统的设计方案和实现细节。我们首先介绍了系统的研发背景，了解到当前鸟类发现和统计方法的痛点，以及引入自动化检测的必要性。随后具体分析了整个系统的重要需求，明确了开发工作的目标，并以此为基础设计了鸟类监控视频检测的整体架构。我们将整个架构按照功能划分为 5 个部分，详细说明了每个模块的具体作用、开发环境和需要用到的开发工具。为了清晰表述重要模块的流程，我们以几个时序图为依据展开介绍。最后通过可视化的图像界面演示了系统的工作流程和操作细节。

第六章　总结与展望

目标检测尤其是视频目标检测任务在我们的生活中随处可见，从交通道路监控到野生动物检测，视频目标检测可以帮助提高人们的生活质量和推动社会快速发展。传统通过人工观看视频，从中找出目标的方法效率太低，耗时耗力。近年来以深度学习为主的目标检测方法已经开始大规模应用于实际任务。然而由于视频数据中目标运动会出现模糊、失焦、遮挡和奇异姿态等现象，以及视频数据采集困难，导致现有的视频目标检测方法效果仍有较大的提升空间。本文从目标检测任务出发，在分析了现有的静态目标检测网络缺点之后，提出了一个轻量的目标检测模型，该模型通过结构重参化等技术实现了在不明显影响模型精度的前提下，明显加快模型的推理速度。本文在静态目标检测模型的基础上，聚合了全局外观信息和局部位置信息，设计了一个能够更有效聚合时序信息的视频目标检测模型，能够更精确检测视频中的目标。最后本文还设计了一款视频目标检测系统，支持数据标注、模型训练和边缘部署等功能，使得本文提出的算法可以直接应用于社会生产中。

在静态目标检测任务中，本文以静态目标检测模型 YoloX 为基础，设计了一个轻量的模型 YoloX-Lite。该模型修改了 Bottleneck 结构的算子顺序，使得整个结构只有最后一个算子是非线性的激活函数，命名为 Rep-Bottleneck。YoloX-Lite 将训练和推理解耦，训练阶段使用原始 Rep-Bottlneck 多分支结构，推理阶段使用结构重参化技术将多个线性运算算子合并为一个卷积层，极大提高了模型整体的检测速度，并且对模型检测精度的影响微乎其微。在颈部网络中，YoloX-Lite 重新设计了数据流向，并制定了算子使用的规则。为了能更好的聚合不同层次的特征，YoloX-Lite 使用通道注意力机制让更加重要的信息在不同层之间传递。本文在标准数据集上进行实验，结果表明，YoloX-Lite 相较于 YoloX，检测精度和推理速度达到了更好的平衡。在和主流的目标检测模型进行对比后，YoloX-Lite 也有明显优势。最后本文针对模型中不同改进之处的作用采取了消融实验，验证了其有效性。

在视频目标检测任务中，本文在静态目标检测模型 YoloX-Lite 的基础上，我们新增了一个聚合全局和局部信息的模块，组成了视频目标检测模型 Yolo-

GLA。该模型通过候选区选择模块，从 YoloX-Lite 的结果中，挑选出最有可能存在目标的一组结果，排除大量的低质量结果对后续信息聚合的干扰，同时降低了计算量。根据结果中的位置信息，Yolo-GLA 结合 YoloX-Lite 中提取的特征图，生成了具有丰富信息的候选区特征。Yolo-GLA 将特征聚合分为多个阶段，第一个阶段聚合全局候选框和局部候选框之间的外观特征，保证了目标可以聚合到充分的信息；第二个阶段聚合局部特征和当前帧特征的外观和位置信息，保证了目标可以被精确定位。本文在公开的 ImageNet VID 数据集上实验发现，和其他主流方法相比，同时结合全局和局部信息，能明显提高整个模型的检测精度，并且维持了理想的推理速度。本文最后对 Yolo-GLA 中部分超参数展开了敏感性分析实验，验证了各自的有效性。

在视频目标检测模型 Yolo-GLA 的基础上，我们设计了一个完整的视频目标检测软件。该软件分为中心侧和边缘侧，中心侧包含数据标注，数据分析和一键训练等功能。边缘侧使用 libtorch 等框架，集成了 Yolo-GLA 算法，可以部署在野外环境中的边缘设备，实现全天候目标检测。整个软件实现了从数据标注到模型训练，再到模型部署的完整流程，帮助视频目标检测任务顺利落地，节省成本，提高人们的工作效率。

本文从实际任务场景出发，针对视频目标检测任务展开了研究。在本文的研究基础上，我们认为有如下几个研究方向可以继续深入：

1. 静态目标检测模型在低参数量情况下提升检测精度。当前静态目标检测在高算力场景下的检测精度已经十分优秀了，然而由于实际应用中的成本限制，往往是采用低算力的边缘设备同时处理多台摄像机传输的视频。可是当前静态目标检测算法在低参数量时的表现退化十分明显，还存在较大的进步空间。在低参数量的情况下增大目标检测模型的精度，有助于降低目标检测技术落地的成本。

2. 针对小目标的视频目标检测。当前主流视频目标检测主要针对中型或者大型的目标检测，小目标由于自身大小限制，不能使用较深的网络提取特征，提取到的特征包含的信息也不够丰富。在多帧信息聚合时，很难通过特征之间的相似度来增强当前帧中小目标信息。设计一个适用于小目标信息聚合的方式有助于扩大目标检测场景，推动视频目标检测的发展。

参考文献

[1] HINTON G E, SALAKHUTDINOV R R. Reducing the Dimensionality of Data with Neural Networks[J]. Science, 2006, 313(5786): 504–507.

[2] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet Classification with Deep Convolutional Neural Networks[J]. Communications of the ACM, 2017, 60(6): 84–90.

[3] RUSSAKOVSKY O, DENG J, SU H, et al. ImageNet Large Scale Visual Recognition Challenge[J]. International Journal of Computer Vision (IJCV), 2015, 115(3): 211–252.

[4] HUANG H, LIN L, TONG R, et al. UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation[C] // ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.]: IEEE, 2020: 1055–1059.

[5] ROH S-D, CHUNG K-S. DAFA: Diversity-Aware Feature Aggregation for Attention-Based Video Object Detection[J]. IEEE Access, 2022, 10: 93453–93463.

[6] HARALICK R M, SHAPIRO L G. Image segmentation techniques[J]. Computer Vision, Graphics, and Image Processing, 1985, 29(1): 100–132.

[7] DONG Q, WANG M, ZHOU H, et al. Consecutive Decoding for Speech-to-text Translation[C] // Proceedings of the AAAI Conference on Artificial Intelligence. 2021: 12738–12748.

[8] HE H, WANG Q, YU Z, et al. Synchronous Interactive Decoding for Multilingual Neural Machine Translation[C] // Proceedings of the AAAI Conference on Artificial Intelligence. 2021: 12981–12988.

[9] KUMAR S, ZHANG X, LESKOVEC J. Predicting Dynamic Embedding Trajectory in Temporal Interaction Networks[C] // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. [S.l.] : Association for Computing Machinery, 2019 : 1269 – 1278.

[10] CHIANG W-L, LIU X, SI S, et al. Cluster-GCN: An Efficient Algorithm for Training Deep and Large Graph Convolutional Networks[C] // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. [S.l.] : Association for Computing Machinery, 2019 : 257 – 266.

[11] SILVER D, HUANG A, MADDISON C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature, 2016, 529(7587) : 484 – 489.

[12] MARRIOTT R T, ROMDHANI S, CHEN L. A 3D GAN for Improved Large-Pose Facial Recognition[C] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.] : IEEE Computer Society, 2021 : 13445 – 13455.

[13] WU R, GU X, TAO X, et al. Landmark Assisted CycleGAN for Cartoon Face Generation[J], 2019.

[14] PAPAGEORGIOU C P, OREN M, POGGIO T. A General Framework for Object Detection[C] // Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271). [S.l.] : IEEE Computer Society, 1998 : 555 – 562.

[15] DALAL N, TRIGGS B. Histograms of Oriented Gradients for Human Detection[C] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.] : IEEE Computer Society, 2005 : 886 – 893.

[16] VIOLA P, JONES M. Rapid object detection using a boosted cascade of simple features[C] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.] : IEEE Computer Society, 2001 : 511 – 511.

[17] VIOLA P, JONES M J. Robust Real-Time Face Detection[J]. International Journal of Computer Vision, 2004, 57(2) : 137 – 154.

[18] FELZENSZWALB P, MCALLESTER D, RAMANAN D. A Discriminatively Trained, Multiscale, Deformable Part Model[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.] : IEEE Computer Society, 2008 : 1 – 8.

[19] EVERINGHAM M, VAN GOOL L, WILLIAMS C K I, et al. The Pascal Visual Object Classes (VOC) Challenge[J]. International Journal of Computer Vision, 2010, 88(2) : 303 – 338.

[20] GIRSHICK R, DONAHUE J, DARRELL T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C] // 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [S.l.] : IEEE Computer Society, 2014 : 580 – 587.

[21] GIRSHICK R. Fast R-CNN[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2015 : 1440 – 1448.

[22] HE K, ZHANG X, REN S, et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(9) : 1904 – 1916.

[23] REN S, HE K, GIRSHICK R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J], 2015, 28.

[24] UIJLINGS J R, VAN DE SANDE K E, GEVERS T, et al. Selective Search for Object Recognition[J]. International Journal of Computer Vision, 2013, 104(2) : 154 – 171.

[25] LIU W, ANGUELOV D, ERHAN D, et al. Ssd: Single shot multibox detector[C] // Proceedings of the European Conference on Computer Vision (ECCV). [S.l.] : Springer International Publishing, 2016 : 21 – 37.

[26] AGRAWAL N, PRABHAKARAN V, WOBBER T, et al. Design Tradeoffs for SSD Performance[C] // USENIX 2008 Annual Technical Conference. [S.l.] : USENIX Association, 2008 : 57 – 70.

[27] ZHAI S, SHANG D, WANG S, et al. DF-SSD: An Improved SSD Object Detection Algorithm Based on DenseNet and Feature Fusion[J]. IEEE Access, 2020, 8 : 24344 – 24357.

[28] 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 (CVPR). 2016 : 779 – 788.

[29] REDMON J, FARHADI A. YOLO9000: Better, Faster, Stronger[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017 : 7263 – 7271.

[30] REDMON J, FARHADI A. YOLOv3: An Incremental Improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[31] BOCHKOVSKIY A, WANG C-Y, LIAO H-Y M. YOLOv4: Optimal Speed and Accuracy of Object Detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[32] GE Z, LIU S, WANG F, et al. YOLOX: Exceeding YOLO Series in 2021[J]. arXiv preprint arXiv:2107.08430, 2021.

[33] LI C, LI L, JIANG H, et al. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications[J]. arXiv preprint arXiv:2209.02976, 2022.

[34] LIN T-Y, DOLLAR P, GIRSHICK R, et al. Feature Pyramid Networks for Object Detection[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017 : 2117 – 2125.

[35] 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 (CVPR). 2018 : 8759 – 8768.

[36] TAN M, PANG R, LE Q V. EfficientDet: Scalable and Efficient Object Detection[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2020 : 10781 – 10790.

[37] LAW H, DENG J. CornerNet: Detecting Objects as Paired Keypoints[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2018 : 734 – 750.

[38] ZHOU X, WANG D, KRÄHENBÜHL P. Objects as Points[J]. arXiv preprint arXiv:1904.07850, 2019.

[39] ZHU X, XIONG Y, DAI J, et al. Deep Feature Flow for Video Recognition[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017 : 2349 – 2358.

[40] ZHU X, WANG Y, DAI J, et al. Flow-Guided Feature Aggregation for Video Object Detection[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017 : 408 – 417.

[41] ZHU X, DAI J, ZHU X, et al. Towards High Performance Video Object Detection for Mobiles[J]. arXiv preprint arXiv:1804.05830, 2018.

[42] HETANG C, QIN H, LIU S, et al. Impression Network for Video Object Detection[J]. arXiv preprint arXiv:1712.05896, 2017.

[43] ZHU X, DAI J, YUAN L, et al. Towards High Performance Video Object Detection[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018 : 7210 – 7218.

[44] LIU M, ZHU M, WHITE M, et al. Looking Fast and Slow: Memory-Guided Mobile Video Object Detection[J]. arXiv preprint arXiv:1903.10172, 2019.

[45] LIU M, ZHU M. Mobile Video Object Detection With Temporally-Aware Feature Maps[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018 : 5686 – 5695.

[46] ZHANG C, KIM J. Modeling Long-and Short-Term Temporal Context for Video Object Detection[C] // IEEE International Conference on Image Processing (ICIP). 2019 : 71 – 75.

[47] MAO H, KONG T, OTHERS. CaTDet: Cascaded Tracked Detector for Efficient Object Detection from Video[J], 2019, 1 : 201 – 211.

[48] KIM H-U, KIM C-S. CDT: Cooperative Detection and Tracking for Tracing Multiple Objects in Video Sequences[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2016 : 851 – 867.

[49] LUO H, XIE W, WANG X, et al. Detect or Track: Towards Cost-Effective Video Object Detection/Tracking[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 33. 2019 : 8803 – 8810.

[50] WU H, CHEN Y, WANG N, et al. Sequence Level Semantics Aggregation for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019 : 9217 – 9225.

[51] DENG J, PAN Y, YAO T, et al. Relation Distillation Networks for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019 : 7023 – 7032.

[52] DENG H, HUA Y, SONG T, et al. Object guided external memory network for video object detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019 : 6678 – 6687.

[53] GUO C, FAN B, GU J, et al. Progressive Sparse Local Attention for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019 : 3909 – 3918.

[54] DENG H, HUA Y, SONG T, et al. Object Guided External Memory Network for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2020 : 10337 – 10346.

[55] HE K, GKIOXARI G, DOLLAR P, et al. Mask R-CNN[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017 : 2961 – 2969.

[56] QIAO S, CHEN L-C, YUILLE A. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution[C] // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021 : 10213 – 10224.

[57] 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 (ICCV). 2017 : 2980 – 2988.

[58] GRAVES A. Long Short-Term Memory[J], 2012(8) : 37 – 45.

[59] SZEGEDY C, LIU W, JIA Y, et al. Going Deeper With Convolutions[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015 : 1 – 9.

[60] HE F, GAO N, LI Q, et al. Temporal Context Enhanced Feature Aggregation for Video Object Detection[C] // Proceedings of the AAAI Conference on Artificial Intelligence. 2020 : 10941 – 10948.

[61] YU F, KOLTUN V. Multi-Scale Context Aggregation by Dilated Convolutions[J], 2016.

[62] BERTASIUS G, TORRESANI L, SHI J. Object Detection in Video with Spatiotemporal Sampling Networks[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2018 : 331 – 346.

[63] DAI J, QI H, XIONG Y, et al. Deformable Convolutional Networks[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017 : 764 – 773.

[64] WOO S, PARK J, LEE J-Y, et al. CBAM: Convolutional Block Attention Module[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2018 : 3 – 19.

[65] FEICHTENHOFER C, PINZ A, ZISSERMAN A. Detect to Track and Track to Detect[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017 : 3038 – 3046.

[66] DAI J, LI Y, HE K, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks[J], 2016 : 379 – 387.

[67] LIU S, QI L, QIN H, et al. Path Aggregation Network for Instance Segmentation[J], 2018 : 8759 – 8768.

[68] SIMONYAN K, ZISSERMAN A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J], 2015.

[69] 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 (CVPR). 2016 : 770 – 778.

[70] IOFFE S, SZEGEDY C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[C] // Proceedings of the 32nd International Conference on Machine Learning. [S.l.] : pmlr, 2015 : 448 – 456.

[71] 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 (CVPR). 2021 : 13733 – 13742.

[72] ELFWING S, UCHIBE E, DOYA K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning[J]. Neural Networks, 2018, 107 : 3 – 11.

[73] ZHANG H, CISSE M, DAUPHIN Y N, et al. mixup: Beyond Empirical Risk Minimization[J], 2018.

[74] SELVARAJU R R, COGSWELL M, DAS A, et al. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017 : 618 – 626.

[75] SHI Y, WANG N, GUO X. YOLOV: Making Still Image Object Detectors Great at Video Object Detection[J]. arXiv preprint arXiv:2208.09686, 2022.

[76] HU H, GU J, ZHANG Z, et al. Relation Networks for Object Detection[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018 : 3588 – 3597.

[77] HOWARD A, SANDLER M, CHU G, et al. Searching for MobileNetV3[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019 : 1314 – 1324.

[78] GONG T, CHEN K, WANG X, et al. Temporal ROI Align for Video Object Recognition[C] // Proceedings of the AAAI Conference on Artificial Intelligence. 2021 : 1442 – 1450.

[79] SUN G, HUA Y, HU G, et al. MAMBA: Multi-level Aggregation via Memory Bank for Video Object Detection[C] // Proceedings of the AAAI Conference on Artificial Intelligence. 2021 : 2620 – 2627.

致　　谢

转眼间，我的研究生生活就要告一段落。在宝贵的三年时间里，我在南京大学结交到了许多朋友，收获了快乐，也学到了知识和为人处事的道理。

我很感谢申老师在这三年里对我的淳淳教导。犹记得最开始我上手一个任务时，因为完全没做过这方面的任务，对其十分抵触，担心自己做不快，做不好。是申老师耐心指导我，要去尝试新的事物，遇到了困难和挑战也不用害怕，尽管放手去做，碰到问题请教老师同学就好了。最后我也克服困难，顺利完成了任务。每周申老师都会抽出时间来和我们一对一谈话，从指导科研内容到关心大家身心健康，还配备了运动器材提供给大家，鼓励我们多出门运动，保证好身体才能做更好的科研。

同时也很感谢赵健老师，每周的组会，赵老师都会提出建设性意见，给大家的科研解答疑问或者提供新的思路。赵老师还经常分享一些写论文的技巧，写论文会用到的工具，让大家写论文期间少走了很多弯路。

世间最美好的东西，莫过于有几个头脑和心地都很正直的朋友。很幸运，我在 RINC 结交了很多好朋友。做学术并不是时时刻刻都是有趣的，相反枯燥的时间更多。正是有了这些好朋友，每天一同去食堂吃饭，定期去打打球，拍拍照，聊聊心事，让平淡的科研生活增添了许多色彩。我们马上就要各奔东西了，希望我们能在更高处相见。

最后我想感谢我的父母，多年苦读，不事生产，是父母给予了我物质和情感上的支撑。无论遇到什么样的麻烦，父母总是我坚强的后盾。现在我也要迈入社会了，可以给家里承担一些责任了，轮到我来报答养育之恩了。

简历与科研成果

基本信息

肖伟康，男，汉族，1998 年 5 月出生，湖北省天门人。

教育背景

2020 年 9 月 — 2023 年 6 月 南京大学人工智能学院 硕士

2016 年 9 月 — 2020 年 6 月 吉林大学软件工程学院 本科

攻读硕士学位期间完成的学术成果

1. Weikang Xiao, Fengshan Liu, Jian Zhao, Furao Shen. Faster-SOINN: A faster unsupervised incremental clustering method”, under-review.
2. 申富饶，赵健，肖伟康. 《一种基于改进特征融合网络的鸟类视频目标检测方法》(202310011137.6)

攻读硕士学位期间参与的科研课题

1. 国家自然科学基金面上项目“基于深度感知增量式联想记忆神经网络的信息融合系统研究”（项目编号 61876076，课题年限 2019 年 1 月—2022 年 12 月），负责目标检测相关问题的研究。

版权及论文原创性说明

任何收存和保管本论文的单位和个人，未经作者本人授权，不得将本论文转借他人并复印、抄录、拍照或以任何方式传播，否则，引起有碍作者著作权益的问题，将可能承担法律责任。

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写的作品成果。本文所引用的重要文献，均已在文中以明确方式标明。本声明的法律结果由本人承担。

作者签名：_____

_____年_____月_____日