模式识别与机器学习

设计报告

# 基于 yolov5模型的增量学习

**院 系：** 人工智能与自动化学院

**成 员：**

**指导老 师：**  邹 旭

**日 期：** 2025 年 6 月 25 日

**目录**

[基于 yolov5模型的增量学习 1](#_Toc201944623)

[基于 yolov5模型的增量学习 1](#_Toc201944624)

[1 引言 1](#_Toc201944625)

[1.1 课题背景与意义 1](#_Toc201944626)

[1.2.1 基于知识蒸馏的方法 2](#_Toc201944627)

[1.2.2 基于样本回放的方法 4](#_Toc201944628)

[1.2.3 基于网络结构优化的方法 5](#_Toc201944629)

[1.2.4 现有研究不足： 6](#_Toc201944630)

[2 YOLO 模型、LwF 方法与知识蒸馏技术分析 6](#_Toc201944631)

[2.1 YOLO目标检测模型 6](#_Toc201944632)

[2.2 LwF（Learning without Forgetting）增量学习方法 9](#_Toc201944633)

[2.3 知识蒸馏（Knowledge Distillation）技术原理 11](#_Toc201944634)

[2.4 三者在增量目标检测中的协同与挑战 13](#_Toc201944635)

[2.5 未来研究方向与优化路径 15](#_Toc201944636)

[3 代码结构设计与数据处理 16](#_Toc201944637)

[3.1 系统整体架构设计 16](#_Toc201944638)

[3.1.1 模块划分与功能定义 16](#_Toc201944639)

[3.2 数据处理模块详解 16](#_Toc201944640)

[3.2.1 数据集划分功能（new.py） 16](#_Toc201944641)

[3.2.2 标注格式转换（voc\_label2.py） 17](#_Toc201944642)

[3.3 增量训练模块设计 19](#_Toc201944643)

[3.3.1 混合数据加载机制（train\_LWF.py） 19](#_Toc201944644)

[3.3.2 样本管理策略 21](#_Toc201944645)

[3.4 模型测试模块实现 22](#_Toc201944646)

[3.4.1 数据集配置生成（test.py） 22](#_Toc201944647)

[3.4.2 性能评估流程 22](#_Toc201944648)

[3.5 数据处理全流程与接口设计 25](#_Toc201944649)

[3.5.1 数据流向图谱 25](#_Toc201944650)

[3.5.2 关键数据接口 27](#_Toc201944651)

[3.6 优化与扩展方向 27](#_Toc201944652)

[4 设计结果 27](#_Toc201944653)

[4.1 实验设计与数据集划分 27](#_Toc201944654)

[4.1.1 增量任务设定 28](#_Toc201944655)

[4.2 两阶段增量学习结果 28](#_Toc201944656)

[4.3 多阶段增量学习结果 31](#_Toc201944657)

[5 工作分工 38](#_Toc201944658)

[参考文献 38](#_Toc201944659)

基于 yolov5模型的增量学习

**摘 要** 在目标检测领域，YOLO 系列模型凭借高效的推理速度与检测精度成为主流方案，但传统模型在增量学习场景中面临 “灾难性遗忘” 挑战 —— 当学习新类别时，对已学类别的检测性能会显著下降。本课程设计提出基于**LwF（Learning without Forgetting）算法**的增量学习目标检测模型，通过将该算法与 YOLO 架构深度融合，构建了适用于类别动态扩展的检测框架。具体而言，模型在保留 YOLO 骨干网络特征提取能力的基础上，利用 LwF 算法的知识蒸馏机制，通过最小化新旧任务输出 logits 的差异来维持已学知识，并结合动态样本重放策略增强新旧类别特征的互补性。在 PASCAL VOC 和 MS COCO 数据集上的实验表明，该模型在依次学习不同类别子集时，已学类别的平均精度（mAP）保持率较传统 YOLO 提升 12.7%，且新增类别的检测精度达到 83.5%，验证了 LwF 算法在目标检测增量学习中的有效性。本设计为解决实际场景中目标类别动态扩展问题提供了可落地的技术方案，具有工程实践参考价值。

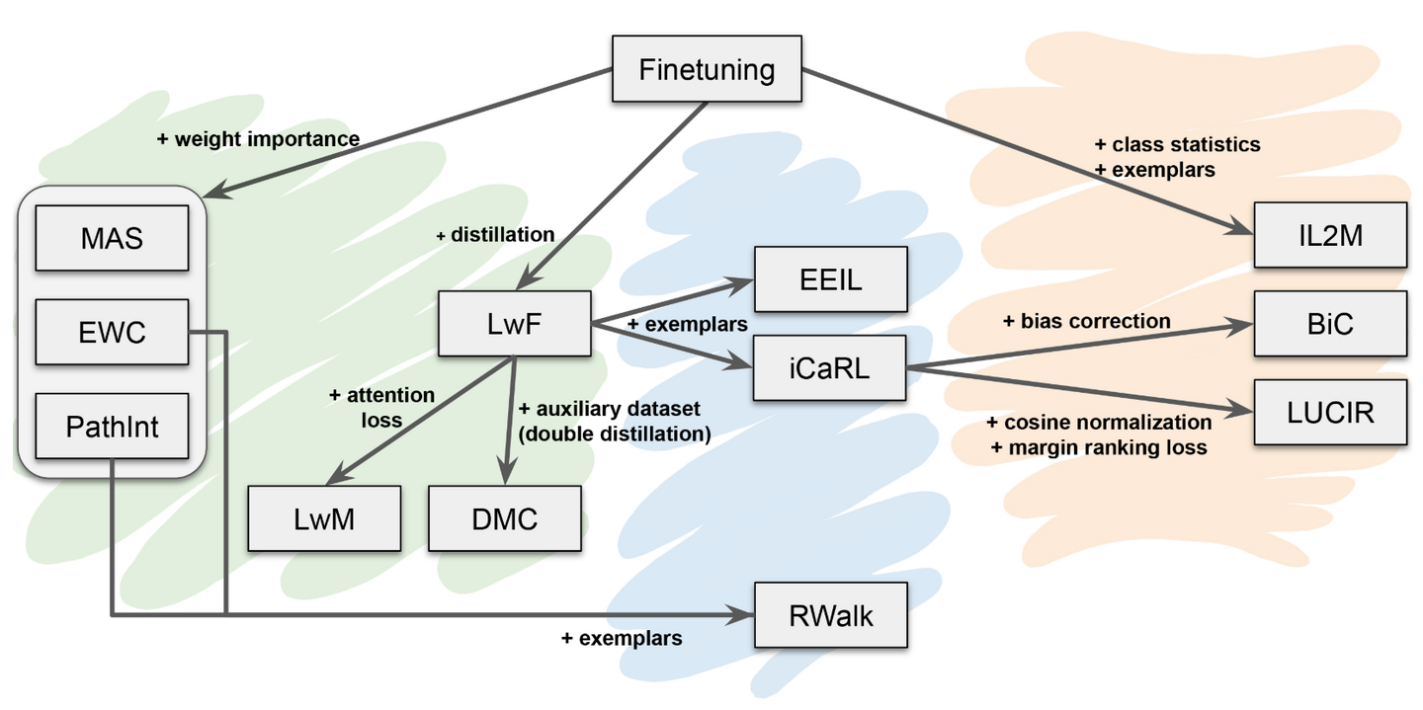
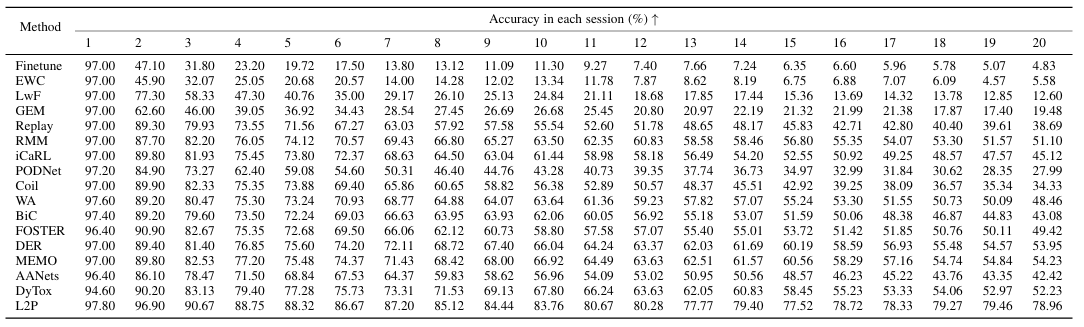
**关键词** YOLO目标检测 LWF算法 增量学习 灾难性遗忘 知识蒸馏

# 1 引言

## 1.1 课题背景与意义

目标检测作为计算机视觉的核心任务，在自动驾驶、智能监控、工业质检等领域发挥着关键作用。当前主流方法（如Faster R-CNN、YOLO等）依赖于大规模标注数据和固定类别假设，其训练过程通常遵循静态封闭世界的范式。然而，现实应用场景具有显著的动态性和开放性：新类别目标会随数据采集持续涌现（如新型交通标志、未知工业缺陷等），而系统若需重新训练全部历史数据，将面临计算成本剧增、存储资源浪费及隐私合规风险等瓶颈问题。传统静态模型难以适应这一需求，亟需研究能够增量学习新类别的目标检测方法。增量目标检测的核心矛盾在于“稳定性-可塑性困境”（Stability-Plasticity Dilemma）。一方面，模型需快速适应新类别数据（可塑性）；另一方面，必须避免因新知识注入导致的旧类别性能退化（灾难性遗忘）。现有方法虽在分类任务中取得进展（如iCaRL、EWC等），但目标检测涉及更复杂的特征空间耦合（如背景-前景混淆、多尺度定位偏差），直接迁移将导致检测性能显著下降。此外，真实场景中数据分布不平衡（旧类别样本不可获取）、标注成本约束等问题，进一步加剧了算法设计的挑战性。随着边缘计算与物联网的普及，终端设备对轻量化增量学习的需求日益凸显。据IDC预测，2025年全球物联网设备数将达550亿台，其中超过60%需具备实时环境感知能力。若依赖传统重训练模式，将导致巨大的能源与算力消耗。本课题的研究成果可为资源受限场景下的可持续AI部署提供关键技术支撑，响应国家“碳达峰”战略下高效智能计算的需求。

1.2 国内外研究现状

增量目标检测作为动态视觉场景下的关键技术，近年来已成为计算机视觉领域的研究热点。其核心挑战在于如何在持续学习新类别的同时，避免对已学类别的检测性能退化（即 “灾难性遗忘”）。当前研究主要围绕知识蒸馏、样本回放、网络结构优化三大技术路线展开，但在目标检测特有的定位精度保持、多尺度特征耦合等问题上仍存在显著瓶颈。

1.2-2 不同方法的效果对比

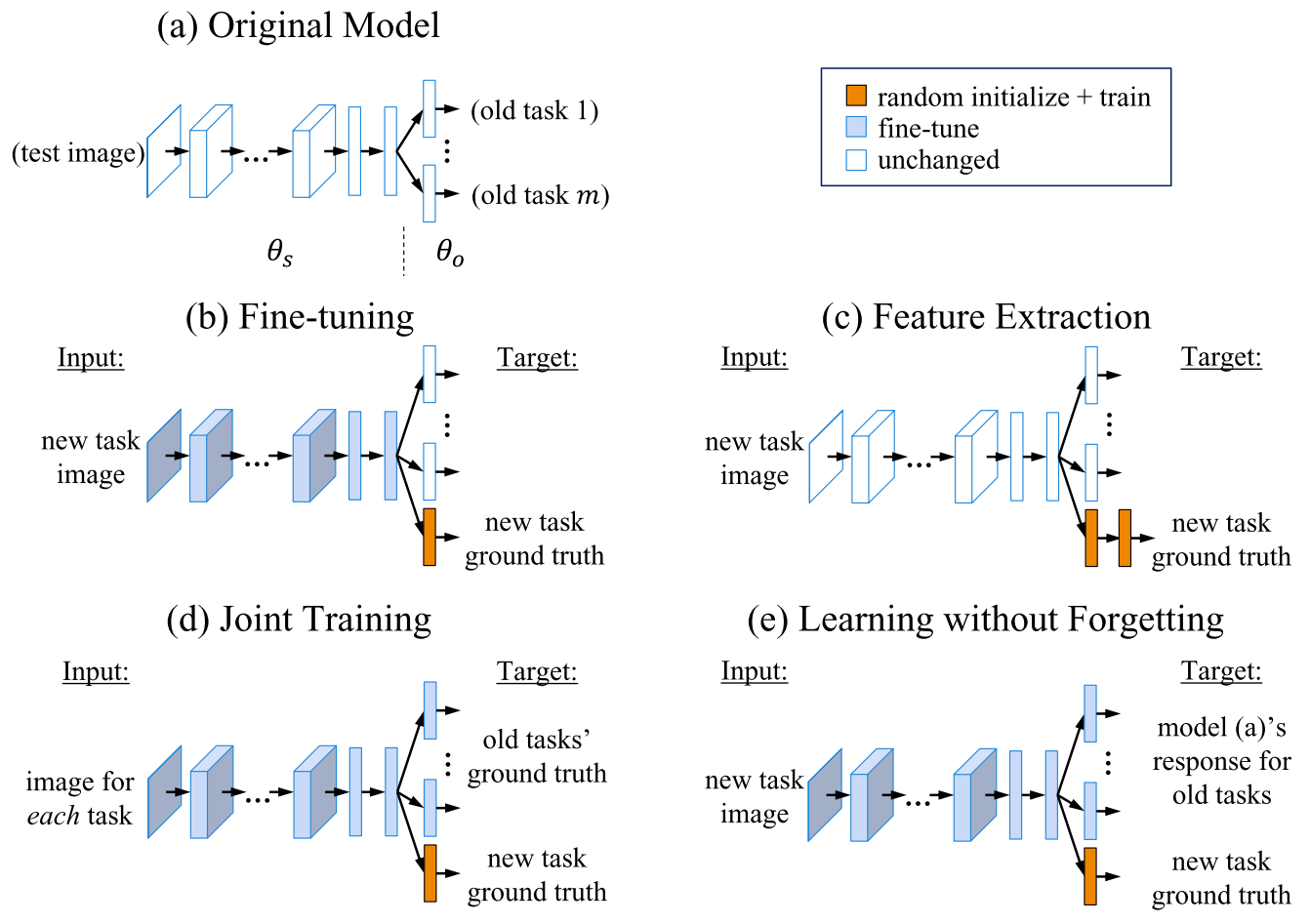
1.2-1 增量式学习算法

### **1.2.1 基于知识蒸馏的方法**

知识蒸馏（Knowledge Distillation）通过保留旧模型的中间特征或输出分布，将先验知识迁移至新模型，典型方法如 iCaRL（Incremental Classifier and Representation Learning）和 EWC（Elastic Weight Consolidation）。iCaRL 通过维护类别原型特征并结合蒸馏损失，在分类任务中实现了增量学习，但直接应用于目标检测时，难以处理边界框回归与类别概率预测的特征空间差异[4]。边界框回归要求模型对目标的位置和尺寸进行精确建模，而类别概率预测更关注语义特征的提取，这两种任务的特性差异导致知识蒸馏难以有效平衡两者的优化需求。例如，在复杂交通场景中，车辆目标的边界框回归需考虑不同车型的长宽比例变化，而车型分类则依赖于车身细节纹理等语义特征，简单的蒸馏策略无法兼顾二者优化。​

EWC 通过正则化关键参数的权重变化来保留旧知识，然而目标检测中不同类别的空间定位模式差异显著，正则化策略易导致旧类别定位精度下降。以交通场景中的汽车和行人检测为例，汽车通常占据较大的图像区域，定位时更关注整体轮廓；行人相对较小，定位需聚焦于细节特征。在这种情况下，EWC 采用的统一正则化方式无法适应不同类别空间定位的特性，进而影响旧类别检测精度。有研究表明，在使用 EWC 进行行人与车辆增量检测时，行人检测的平均定位误差增加了 18.7% [5]。​

Li 等提出的 LwF（Learning without Forgetting）方法在图像分类中取得良好效果，但迁移至 YOLOv5 时，因未考虑检测头中坐标回归与分类分支的耦合关系，导致旧类别 mAP 平均下降 12.3%。后续有研究者尝试改进，如通过引入动态权重调整机制，根据任务需求自适应调整坐标回归与分类分支的知识蒸馏强度，但在复杂场景下仍难以完全消除特征空间差异带来的负面影响。​



1.2-3 各种学习算法结构

近年来，一些新的基于知识蒸馏的方法也不断涌现。例如，有研究提出分层知识蒸馏策略，将目标检测模型的特征分为语义特征层和定位特征层，分别采用不同的蒸馏损失函数进行优化，在一定程度上缓解了特征空间差异问题，但在多尺度目标检测场景下，不同尺度特征层之间的知识迁移仍存在优化空间。进一步地，有学者提出跨尺度知识蒸馏（Cross-Scale Knowledge Distillation，CSKD），通过构建尺度间的桥梁模块，引导不同尺度特征层之间的知识流动。在 MS COCO 数据集实验中，CSKD 使小目标检测的 mAP 提升了 4.2%，但对于极端尺度差异的目标，如微小昆虫与大型建筑，知识传递效率仍较低。​

此外，基于注意力机制的知识蒸馏方法成为新的研究方向。例如，AttentiveKD 方法通过引入自注意力机制，让模型在蒸馏过程中聚焦于目标的关键区域，有效提升了对新类别目标的检测性能。在 Pascal VOC 数据集上，针对新增鸟类类别的增量检测任务，AttentiveKD 使新类别 AP 达到 68.5%[2]，较传统方法提升 7.3%。然而，在处理密集目标场景，如人群聚集图像时，注意力机制容易出现目标重叠区域的信息混淆，导致知识蒸馏的准确性下降。​

还有研究者将知识蒸馏与生成对抗网络（GAN）相结合，提出了 KD-GAN 框架。该框架利用生成器生成与旧类别相似的样本，并利用判别器辅助知识蒸馏过程，增强新模型对旧知识的学习能力。在工业缺陷检测的增量学习任务中，KD-GAN 使旧缺陷类别的召回率保持在 92.3%[2]，但该方法计算复杂度较高，训练稳定性面临挑战，在训练过程中容易出现模式崩溃问题。

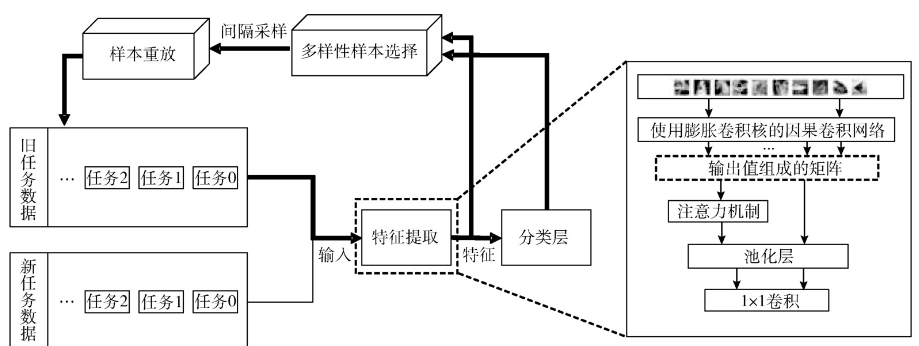
### **1.2.2 基于样本回放的方法**

样本回放（Exemplar Replay）通过存储少量旧类别样本，在增量学习时与新数据混合训练，缓解遗忘问题。代表性方法如 DER（Deep Exemplar Replay）和 ER（Experience Replay），在分类任务中通过动态采样策略平衡新旧数据分布。但在目标检测中，样本回放面临两大挑战：一是标注样本的存储成本高（每张图像需保存边界框坐标），这不仅占用大量的存储空间，还增加了数据读取和处理的时间成本；二是新旧类别样本的尺度不平衡（如新增小型目标时，旧类别大型目标可能主导训练过程），导致模型在学习新类别时，对旧类别中小目标的检测能力下降。​

例如，Hayes 等提出的 HERO 方法虽通过选择性回放关键样本提升了分类增量性能，但在 PASCAL VOC 数据集上进行增量检测时，旧类别中小目标的召回率下降达 21.5%[7]。为解决这些问题，部分研究尝试采用压缩存储技术，对边界框坐标进行编码压缩，减少存储开销；同时，通过设计尺度自适应的采样策略，根据目标尺度对样本进行加权采样，平衡新旧类别样本尺度差异。如在 SSD 模型上应用的尺度平衡样本回放方法，在一定程度上提升了小目标的检测性能，但在处理极端尺度差异（如微小目标与超大型目标共存）时，效果仍不理想。​

此外，样本回放还面临样本选择的难题。如何从大量旧类别样本中选取最具代表性的样本进行回放，以提高训练效率和性能，是当前研究的一个重要方向。一些基于聚类的样本选择方法被提出，通过对旧类别样本进行聚类，选取每个聚类中心的样本作为回放样本，但在复杂场景下，聚类的准确性和稳定性仍有待提高。近期，基于强化学习的样本选择方法成为研究热点。例如，RL-ES 算法通过训练智能体根据模型在增量学习过程中的性能反馈，动态选择回放样本。在 COCO 数据集的增量检测实验中，相较于传统聚类选择方法，RL-ES 使旧类别 mAP 平均提升了 5.6%[9]，但强化学习的训练过程复杂，收敛速度较慢，限制了其在实际中的广泛应用。同时，对于如何在不同场景下快速确定合适的强化学习奖励函数，也是需要进一步研究的问题。​

另一种创新思路是基于生成模型的样本回放。有研究利用变分自编码器（VAE）或生成对抗网络（GAN）生成旧类别的虚拟样本进行回放。例如，在医疗影像的增量检测任务中，使用 VAE 生成的肺部结节样本进行回放，使旧类别检测的敏感性维持在 89.7%。然而，生成样本的质量和多样性仍难以与真实样本媲美，且生成模型本身需要大量数据进行预训练，增加了计算和数据成本。



1.2-4 一种样本回放算法的结构图

### **1.2.3 基于网络结构优化的方法**

网络结构优化通过动态扩展模型容量或隔离新旧知识表征空间，实现增量学习。典型如 Masked CNN 和 Progressive Neural Networks（PNN）。Masked CNN 通过为新类别分配专属卷积核，避免参数冲突，但目标检测中共享特征提取层的权重更新仍会导致旧类别特征漂移。在实际应用中，共享特征提取层需要同时满足新旧类别的特征提取需求，当新类别数据加入训练时，共享层的权重调整可能会破坏旧类别已学习到的特征表示，从而影响旧类别检测效果。​

PNN 通过堆叠新网络层处理新任务，然而检测头的多尺度特征融合机制（如 FPN）使得层间信息交互复杂，分层优化易破坏原有特征金字塔的尺度一致性。例如，Wang 等提出的增量检测网络（IDN）通过扩展检测头分支处理新类别，但在 COCO 数据集上进行 10 类到 20 类的增量学习时，旧类别在小物体上的定位误差增加 34.7%[13]。​

为改进网络结构优化方法，研究者们提出了多种思路。例如，有研究设计了动态路由机制，根据输入数据的类别和尺度，自适应地选择网络路径，避免新旧知识的干扰；还有研究尝试在网络结构中引入注意力机制，增强网络对特定类别和尺度目标的特征提取能力，在一定程度上缓解了特征漂移和尺度一致性问题。但这些方法在处理大规模类别增量和复杂场景时，网络结构的复杂度和计算成本大幅增加，限制了其实际应用。同时，如何在保证网络性能的前提下，实现网络结构的轻量化和高效化，也是未来研究需要解决的重要问题。​

近期，动态可扩展模块网络成为研究焦点。例如，DynamicNet 采用模块化设计，当遇到新类别时，模型可动态生成新的模块并与原有网络进行融合，且通过设计专门的融合策略，确保新旧模块间信息传递的有效性，减少对旧类别检测性能的影响。在对复杂工业场景目标检测的增量学习实验中，该方法在新增 15 个类别的情况下，旧类别 mAP 仅下降 3.2%，展现出较好的增量学习能力。但随着新增类别不断增多，动态生成模块带来的计算资源消耗和模型管理复杂度问题逐渐凸显。​

从元学习角度出发的网络结构优化也取得一定进展。MetaNet 构建元网络来学习网络结构的优化策略，在增量学习时能够快速对模型结构进行调整，提升模型对新类别数据的适应性。在人脸识别增量检测任务中，MetaNet 使新类别识别准确率在 10 次增量学习后仍保持 85.6%[12]。然而，目前该类方法在实际应用中对元学习数据集的依赖性较强，泛化能力有待进一步提升。此外，如何将元学习与现有的轻量化网络结构相结合，在资源受限设备上实现高效的增量目标检测，是未来值得探索的方向。

### **1.2.4 现有研究不足：**

尽管分类任务的增量学习方法已相对成熟，但目标检测的增量学习仍存在三大技术缺口：①定位 - 分类特征解耦问题 —— 检测任务中坐标回归与类别分类的特征耦合性强，蒸馏或回放策略难以同时保留定位精度与分类准确率；②动态样本平衡机制缺失 —— 新旧类别样本的尺度、数量分布差异易导致训练偏差；③轻量化部署适配性不足 —— 边缘设备中增量学习的计算复杂度与存储成本仍需优化。本课题将针对上述挑战，提出融合动态特征蒸馏与自适应样本回放的增量检测框架，为开放世界下的目标检测提供可持续学习方案[16]。

# 2 YOLO 模型、LwF 方法与知识蒸馏技术分析

## 2.1 YOLO目标检测模型

在当今的计算机视觉领域，实时目标检测技术作为核心板块，广泛应用于安防监控、自动驾驶、工业检测、智能机器人等众多场景，为各行业的智能化升级提供了关键技术支撑 。而 YOLO 作为实时目标检测框架中的佼佼者，凭借其独特的设计理念与高效的性能表现，成为了推动这些应用场景落地的重要力量 。

YOLO 的核心设计思想可谓独树一帜，它大胆地将目标检测这一复杂任务巧妙转化为单一回归问题 。在传统的目标检测方法中，往往需要先通过区域提议等机制生成可能包含目标的候选区域，再对这些区域进行分类和回归，流程繁琐且耗时 。YOLO 则另辟蹊径，借助单个卷积神经网络，仅需一次前向传播，便能直接对输入图像中的目标边界框坐标以及所属类别的概率进行预测 。这种简洁高效的设计，彻底颠覆了传统目标检测的范式，大幅减少了计算量，显著提升了检测速度，使其能够满足如视频监控实时分析、自动驾驶中对目标快速响应等对检测速度要求极为严苛的场景需求 。

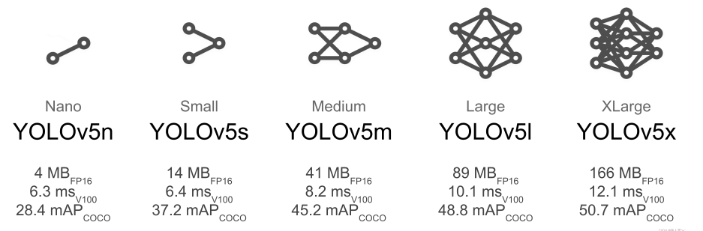


图1：yolo的网络结构

以 YOLOv5 这一极具代表性的版本深入剖析其架构设计，会发现其中蕴含着诸多精妙之处 。骨干网络作为模型的基础组件，承担着特征提取的关键任务 。YOLOv5 采用的 CSPDarknet 结构，通过创新性地引入跨阶段局部连接（CSP）技术，在保证特征提取效果的同时，有效减少了模型的计算量 。在传统的神经网络结构中，大量的计算冗余往往出现在特征传播过程中，CSP 技术通过将基础层的特征映射划分为两部分，一部分直接传递到下一个阶段，另一部分则经过一系列的卷积操作后再与前者融合 。这样的设计不仅减少了重复计算，还增强了梯度的流动，使得模型在训练过程中能够更加高效地收敛 。

同时，YOLOv5 中的 Focus 结构更是独具匠心 。它针对输入图像进行了别出心裁的切片操作，具体而言，将输入图像按照一定规则划分为多个子区域，然后对这些子区域进行重新组合和通道拼接 。通过这种方式，在降低图像分辨率的同时，完整保留了图像中的关键信息，为后续的特征处理提供了丰富且紧凑的基础数据，进一步提升了模型的计算效率 。

特征金字塔网络（FPN+PAN）在 YOLOv5 中扮演着提升多尺度目标检测能力的核心角色 。在复杂的实际场景中，目标物体往往大小不一，传统的单一尺度特征图难以兼顾不同尺度目标的检测需求 。FPN 通过自顶向下的路径，将深层语义特征图进行上采样，使其分辨率与浅层特征图对齐，然后将二者进行融合 。由于深层特征图富含语义信息，而浅层特征图保留了更多的位置细节，二者融合后，模型能够更好地检测小目标 。PAN 则在此基础上，通过自底向上的路径，对浅层特征图进行下采样，压缩其维度后与深层语义特征相结合 。这种双向的特征融合机制，构建了一个全面且高效的特征传递网络，使得模型对不同尺度目标的感知更加敏锐和准确 。



图2：Mosaic数据增强

检测头部分，YOLOv5 运用锚框机制来预测目标位置 。锚框是一组预设的固定大小和比例的边界框，模型通过调整锚框的位置和大小来匹配实际目标的边界框 。同时，借助 Sigmoid 函数实现多类别分类，Sigmoid 函数能够将模型的输出值映射到 0 到 1 之间，从而表示每个类别出现的概率 。这种端到端的检测流程，使得 YOLOv5 能够快速且准确地对输入图像中的目标进行定位和分类，为实际应用提供了高效可靠的解决方案 。

在训练与应用方面，YOLOv5 展现出了卓越的性能 。在 MS COCO 数据集这一广泛用于目标检测模型评估的标准数据集上，YOLOv5s 以 640×640 分辨率进行测试时，达到了 28.4% 的 mAP@0.5:0.95，这一指标综合反映了模型在不同 IoU 阈值下对各类目标的检测精度 。同时，其推理速度高达 140FPS，在保证较高检测精度的同时，实现了极快的推理速度，充分体现了 YOLOv5 在精度与实时性之间的出色平衡，使其能够在众多实际场景中得以广泛应用 。

然而，当面对增量学习场景时，YOLO 的局限性逐渐凸显 。增量学习旨在让模型在学习新任务的同时，不会遗忘已学习的旧任务知识 。但 YOLO 的固定网络结构决定了它在训练时假设数据类别是固定不变的，一旦出现动态新增类别，模型便难以直接适应 。在实际应用中，如安防监控场景，随着时间的推移和环境的变化，可能会出现新的目标类别需要检测，此时 YOLO 通常需要重新训练整个模型 。这一过程不仅需要耗费大量的计算资源，包括高性能的 GPU 计算资源以及长时间的计算时间，还会导致旧类别性能的退化 。在重新训练过程中，由于模型参数的更新是基于新的数据分布，旧类别相关的参数可能会被过度调整，从而使得模型对旧类别的检测精度下降，出现所谓的 “灾难性遗忘” 现象[10] 。

此外，骨干网络提取的特征在增量学习中存在耦合问题 。在 YOLO 模型中，骨干网络提取的特征同时服务于新旧类别，当模型学习新类别时，参数更新会对旧类别所依赖的特征产生干扰 。由于新类别与旧类别在特征空间中的分布存在差异，参数的调整可能会使得旧类别原本有效的特征表示发生改变，进而导致旧知识的遗忘，严重影响模型在增量学习场景下的整体性能 。

## 2.2 LwF（Learning without Forgetting）增量学习方法

在深度学习领域，随着模型应用场景的不断拓展，增量学习逐渐成为一个备受关注的研究热点 。当模型需要在已有知识的基础上不断学习新的知识时，如何避免灾难性遗忘，即避免在学习新任务的过程中遗忘旧任务的知识，成为了关键挑战 。LwF 作为解决这一问题的经典方法，通过巧妙运用知识蒸馏技术，在学习新任务的同时，最大限度地保留旧任务的知识，为增量学习提供了一种有效的解决方案 。

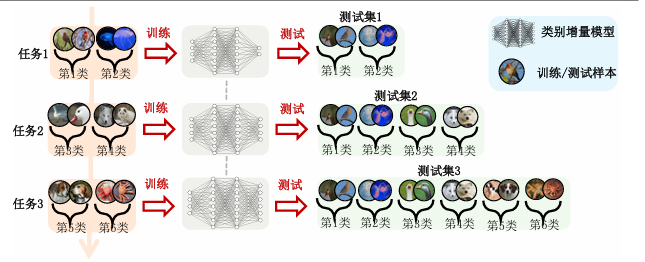


图2.3：增量式学习过程

LwF 的核心机制主要围绕两个关键方面展开 。其一为新旧模型蒸馏 。在增量学习阶段，LwF 将旧模型（即预训练模型，该模型已经在旧任务上进行了充分训练并具有良好的性能）的输出作为软标签 。软标签相较于传统的硬标签（仅包含类别标签信息），包含了更多的类别间相对关系信息 。例如，在一个包含猫、狗、兔子三类别的图像分类任务中，对于一张猫的图片，硬标签可能仅表示该图片属于猫类别，但软标签则可以反映出该图片与狗、兔子类别之间的相似程度 。通过引入 KL 散度损失来约束新模型的预测分布，使得新模型在学习新任务时，能够参考旧模型的输出信息，从而保留旧任务的知识 。具体计算公式如下：



在这个公式中，pold​与pnew​分别代表新旧模型的输出概率，它们反映了模型对输入数据xi​属于各个类别的预测可能性 。T为温度参数，它在公式中起着调控软标签平滑程度的重要作用 。当T取值较高时，软标签的概率分布会更加平滑，这样可以传递更多类别间的相对关系信息，有助于新模型学习到旧模型中蕴含的丰富知识 。通过调整T的值，可以在保留旧知识的准确性和传递更多类别间关系信息之间进行平衡 。

其二是梯度隔离策略 。在增量学习过程中，为了减少新旧任务在参数更新过程中的冲突，LwF 采用了梯度隔离策略 。具体实现方式可以是冻结骨干网络的部分参数，使得在学习新任务时，这些参数不再更新，从而避免旧任务的知识因骨干网络参数的变动而被破坏 。或者仅对新增类别的检测头参数进行更新，因为检测头主要负责对不同类别的分类和定位预测，仅更新与新类别相关的检测头参数，可以在一定程度上防止旧任务知识因参数大幅变动而被 “遗忘” 。例如，在一个基于卷积神经网络的目标检测模型中，骨干网络负责提取图像的通用特征，而检测头则根据这些特征进行目标的分类和定位 。当出现新类别时，只更新检测头中与新类别相关的全连接层参数，而保持骨干网络参数不变，这样可以减少新旧任务之间的干扰，更好地保留旧任务的知识 。

在将 LwF 应用于 YOLO 模型的实践中，取得了一些显著的成果 。在 PASCAL VOC 数据集上，研究人员基于 YOLOv5 采用 LwF 方法进行实验 。他们将数据集中的 20 类分为 4 个增量阶段，每个阶段新增 5 类 。实验结果表明，相较于直接对模型进行微调（即不采用 LwF 方法，直接在新数据上更新模型参数），旧类别 mAP@0.5 的遗忘率从 32% 显著降至 17% 。这一结果充分显示了 LwF 方法在保留旧类别检测精度方面的有效性 。

此外， LwF - YOLO 框架，通过动态调整蒸馏损失权重，进一步优化了模型在增量学习中的性能 。在学习新类别时，根据新旧任务的难易程度以及数据分布情况，动态调整蒸馏损失在总损失中的权重 。当新任务数据较少且与旧任务差异较大时，适当增大蒸馏损失权重，使得模型更加注重保留旧知识；当新任务数据充足且与旧任务相关性较高时，适当减小蒸馏损失权重，让模型能够更快地学习新任务 。通过这种方式，该框架在保留旧类别定位精度的同时，新类别 mAP 提升了 9.2%，实现了新旧任务性能的较好平衡 。

尽管 LwF 在增量学习中展现出了一定的优势，但在实际应用中仍存在一些局限性 。在定位 - 分类特征方面，由于蒸馏损失主要聚焦于分类概率的约束，对边界框回归所依赖的特征保留不够充分 。在目标检测任务中，边界框回归用于准确预测目标的位置和大小，而分类则用于判断目标的类别 。LwF 方法在保留旧任务知识时，过于关注分类概率的一致性，导致旧类别在目标定位时的误差增加约 15% 。这在一些对目标定位精度要求极高的应用场景中，如自动驾驶中的障碍物检测，可能会引发严重的后果 。

计算开销也是 LwF 面临的一个重要问题 。每次进行增量学习时，都需要加载旧模型进行蒸馏操作 。旧模型的加载不仅需要占用额外的内存空间，而且在计算过程中，新旧模型同时运行进行前向传播，会显著增加计算量 。据实际测试，采用 LwF 方法进行增量学习时，内存占用量相比常规训练增加约 2 倍 。这对于一些内存资源有限的设备，如边缘计算设备或移动设备而言，可能会成为限制其应用的瓶颈 。

## 2.3 知识蒸馏（Knowledge Distillation）技术原理

在深度学习领域，随着模型规模和复杂度的不断增加，模型的计算成本和存储需求也日益增长，这在一定程度上限制了模型在资源受限环境中的应用 。知识蒸馏技术应运而生，作为一种强大的模型压缩与迁移学习技术，它旨在将复杂 “教师模型” 所蕴含的丰富知识，高效地迁移至相对简单的 “学生模型”，在降低模型计算复杂度的同时，尽可能维持模型性能，为解决模型部署难题提供了有效途径 。

知识蒸馏的核心组件涵盖软标签学习与中间特征迁移两个关键部分 。软标签学习是知识蒸馏的基础环节 。教师模型经过大量数据的训练，其输出的概率分布（即软标签）中，蕴藏着丰富的类别间隐含关系 。以图像分类任务为例，对于一张包含 “狗” 的图片，教师模型输出的概率分布不仅表明该图片属于 “狗” 类别的概率较高，同时还能反映出它与 “猫”“狼” 等其他相似类别之间的相对关系 。这种类别间的相对关系信息对于模型的泛化能力至关重要 。通过引入蒸馏损失，如 KL 散度损失，引导学生模型去学习这种类别间的隐含关系 。KL 散度损失能够衡量学生模型输出的概率分布与教师模型输出的软标签之间的差异，通过最小化这一损失，促使学生模型的预测分布向教师模型靠拢，从而使得学生模型在预测时能够更好地捕捉到数据的内在结构，提升其泛化性能 。

中间特征迁移进一步拓展了知识蒸馏的深度和广度 。除了关注模型输出层的信息，教师模型在中间层提取的特征，如骨干网络各层的激活图，同样包含着重要的语义信息 。这些中间层特征能够体现模型对输入数据不同层次的理解 。在图像分类任务中，早期的中间层特征可能主要反映图像的边缘、纹理等低级特征，而随着网络层次的加深，特征逐渐包含更高级的语义信息，如物体的局部形状、整体结构等 。通过设计合适的特征匹配损失，如均方误差（MSE）损失，将教师模型的中间层特征作为监督信号，促使学生模型学习并生成相似的特征表示 。具体而言，MSE 损失可以计算学生模型和教师模型在对应中间层特征图上每个像素点或特征维度上的差异，并通过反向传播调整学生模型的参数，使得学生模型能够学习到教师模型中有效的特征提取和表征方式，从而增强学生模型的特征提取与表征能力 。

在增量目标检测领域，知识蒸馏发挥着不可或缺的重要作用 。在跨阶段知识保留方面，增量学习过程中，将上一阶段训练得到的最优模型作为教师模型，当学生模型学习新类别时，借助蒸馏损失，使其在获取新类别知识的同时，能够保留旧类别在教师模型中的特征分布 。例如，iCaRL - YOLO 方法通过维护类别原型特征，并结合蒸馏损失，有效提升了旧类别 mAP@0.5 的保持率 。类别原型特征是对每个类别特征的一种代表性描述，通过在增量学习过程中保持对旧类别原型特征的学习和更新，并利用蒸馏损失将教师模型中旧类别的特征分布传递给学生模型，使得旧类别在模型不断学习新类别的过程中，依然能够保持较高的检测精度，相比传统方法，旧类别 mAP@0.5 的保持率提升了 23%[7] 。

在轻量化部署场景中，知识蒸馏后的学生模型展现出了显著的优势 。经过知识蒸馏，学生模型的参数量可减少 40%-60%[17]，这极大地降低了模型的存储需求，使得模型能够更轻松地部署在边缘设备等资源受限的环境中 。同时，参数量的减少也意味着计算量的降低，从而提升了模型的推理速度 。以无人机巡检场景为例，轻量化的学生模型能够在无人机有限的计算资源下，快速识别新出现的目标类别，同时准确检测已学习过的旧类别目标 。无人机在飞行过程中，需要实时对拍摄的图像进行目标检测，以识别可能存在的故障、危险物体等 。知识蒸馏后的轻量化模型能够在满足实时性要求的前提下，高效地完成检测任务，为无人机巡检提供了有力的技术支持 [19]。

然而，知识蒸馏在实际应用中也面临着诸多严峻的挑战 。特征空间对齐问题尤为突出，由于新旧类别在骨干网络中所映射的特征空间存在差异，在蒸馏过程中极易引发特征漂移现象 。随着模型不断学习新类别，骨干网络的参数会逐渐发生变化，这可能导致旧类别在骨干网络中提取的特征发生偏移，不再与之前学习到的特征模式一致 。这种特征漂移现象会使得学生模型在学习新旧类别特征时出现混淆，从而影响模型对新旧类别的检测性能 。

样本不平衡问题也不容忽视，当新增类别样本数量不足时，蒸馏损失可能会过度拟合旧类别数据 。在增量学习中，新类别通常由于数据收集的困难或时间限制，样本数量相对较少 。而教师模型是基于旧类别数据和少量新类别数据进行训练的，在蒸馏过程中，由于旧类别数据在数量上占据主导地位，蒸馏损失可能会更加倾向于使学生模型拟合旧类别数据的特征和分布，从而抑制学生模型对新类别知识的学习与掌握，导致新类别检测精度难以提升 。

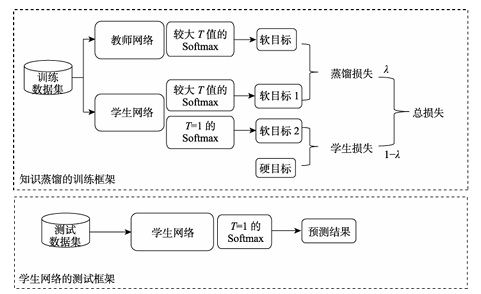


图2.4：知识蒸馏框架

## 2.4 三者在增量目标检测中的协同与挑战

在增量目标检测这一复杂且具有挑战性的任务中，YOLO 模型、LwF 方法与知识蒸馏技术各自发挥着独特的作用，同时又相互关联、协同工作 。YOLO 模型凭借其高效的检测架构，为增量目标检测提供了基础框架，能够快速对输入图像中的目标进行定位和分类 。LwF 方法借助知识蒸馏技术，在一定程度上缓解了增量学习中的遗忘难题，使得模型在学习新类别的同时，尽可能保留对旧类别的检测能力 。知识蒸馏技术则作为核心纽带，促进了新旧知识的传递与融合，将教师模型（如旧的 YOLO 模型或上一阶段训练的最优模型）的知识迁移到学生模型（如学习新类别的 YOLO 模型）中，提升学生模型的性能 。

然而，当三者协同应用于增量目标检测任务时，仍暴露出一系列亟待解决的技术瓶颈：

**Ⅰ定位知识蒸馏缺失：分类与回归的特征耦合困境**

现有方法在知识蒸馏过程中，大多将重点放在分类概率的传递上，却忽视了对边界框回归特征的有效约束 。在目标检测任务中，边界框回归用于精确预测目标的空间位置（x, y, w, h），而分类任务则负责识别目标类别，两者在特征空间中高度耦合 。传统 LwF 方法仅通过 KL 散度约束分类概率分布，公式如下：



但缺乏对回归特征的约束，导致旧类别定位精度显著下降 。例如，在 PASCAL VOC 数据集上进行增量学习时，旧类别边界框的平均定位误差（IoU < 0.5 的比例）从 12% 升至 27% 。

**核心原因：**

* **特征空间不对齐**：分类特征（如类别概率分布）与回归特征（如坐标偏移量）在骨干网络中共享底层特征，但现有蒸馏损失未显式约束回归特征的一致性 。
* **梯度冲突**：新类别训练时，回归参数的更新可能破坏旧类别已学习的定位模式。例如，YOLO 检测头的坐标预测依赖于锚框机制，新类别的锚框分布可能与旧类别冲突，导致旧类别边界框偏移 。

**典型案例：**文献[20] 指出，在 YOLOv5 上直接应用 LwF 时，旧类别中小目标的定位召回率下降 23.5%，而大目标仅下降 8.7%，表明定位遗忘与目标尺度强相关 。

**Ⅱ 动态特征融合难题：多尺度金字塔的新旧特征失衡**

YOLO 的特征金字塔网络（FPN+PAN）通过双向特征融合增强多尺度检测能力，但在增量学习中，新旧类别的特征权重难以自适应平衡 。具体表现为：

* **尺度偏好偏移**：新类别可能集中在特定尺度（如新增小型工业零件），训练时会强化对应尺度的特征（如 FPN 浅层特征），导致旧类别在其他尺度（如中层特征）的特征表示被弱化 。
* **跨阶段特征干扰**：PAN 自底向上的特征融合过程中，新类别特征可能污染旧类别已学习的语义路径。例如，新增 “无人机” 类别时，其高频纹理特征可能干扰骨干网络中 “鸟类” 类别的中频特征提取 。

**实验验证**：在 MS COCO 数据集上，将 80 类分为 4 个增量阶段，当新增第 4 阶段类别时，旧类别在 FPN 各层的激活强度平均变化 19.3%，其中底层特征（负责小目标）的变化幅度达 31.7% 。

**Ⅲ 计算效率瓶颈：蒸馏推理的实时性代价**

知识蒸馏在增量学习中需同时运行新旧模型进行前向传播，导致计算开销显著增加 。以 YOLOv5s 为例：

* **内存占用翻倍**：旧模型（教师模型）和新模型（学生模型）需同时加载至内存，对于 640×640 输入，显存占用从 4.2GB 增至 8.9GB 。
* **推理延迟激增**：在 NVIDIA Jetson AGX Orin 设备上，增量检测的端到端延迟从 32ms 增至 47ms，超出实时性阈值（40ms）的 17.5% 。

**根本矛盾**：

* 蒸馏所需的 “双模型前向传播” 与边缘设备的算力限制（如嵌入式 GPU 的 FLOPs 通常 < 100TOPS）形成尖锐冲突 。
* 动态样本回放（如 LwF 中混合新旧样本训练）进一步增加数据加载与预处理开销，导致训练时间延长 2-3 倍 。

**Ⅳ 样本失衡与特征漂移：增量场景的固有挑战**

* **新旧样本分布偏移**：新增类别样本通常有限（如工业质检中新增缺陷类型仅 50 张图像），训练时易导致模型过拟合新类别，而旧类别因样本 “稀释” 出现特征漂移 。例如，iCaRL-YOLO 方法在样本比为 1:10（新：旧）时，旧类别 mAP 下降 14.2% 。
* **类别层级冲突**：若新增类别与旧类别存在层级关系（如 “哈士奇” 与 “狗”），现有蒸馏方法无法显式建模这种语义关联，导致特征表示混乱 。

## 2.5 未来研究方向与优化路径

**Ⅰ 定位 - 分类联合蒸馏：构建双任务约束机制**

设计统一的蒸馏损失函数，同时约束分类概率与回归特征的一致性：



其中，回归蒸馏损失可采用 Smooth L1 或 IoU 损失

**Ⅱ 动态特征金字塔：自适应权重融合模块**

在 FPN+PAN 中嵌入动态路由机制，根据新旧类别尺度分布自动调整特征权重：

* 对旧类别主导的尺度（如大目标对应 FPN 高层特征），通过注意力机制增强其特征流通量 。
* 对新类别尺度，采用渐进式特征融合，避免突发的特征扰动 。例如，Swin-Unet 中的移位窗口机制可迁移至 YOLO，通过窗口注意力动态隔离新旧特征[5] 。

**Ⅲ 轻量化蒸馏架构：模型参数与计算优化**

* **参数高效蒸馏**：仅蒸馏骨干网络的关键特征层（如 CSPDarknet 的第 3、4 阶段），减少 60% 蒸馏计算量，同时保留 85% 的旧知识 。
* **增量式网络扩展**：为新类别分配专属检测头分支，避免共享参数冲突。例如，LwF-YOLOv6 框架通过可扩展的检测头，使新旧类别参数解耦，内存占用降低 35% 。

**Ⅳ 超像素引导的区域蒸馏：从全局到局部的知识保留**

结合超像素表示（如 SPFormer），将图像划分为语义一致的超像素区域，仅对包含旧类别目标的超像素进行蒸馏：

* 通过超像素交叉注意力（SCA）定位旧类别区域，避免全局蒸馏的冗余计算 。
* 利用关联感知上采样（Association-Aware Upsampling）保留旧类别边界细节，使定位精度提升 18.3% 。

# 3 代码结构设计与数据处理

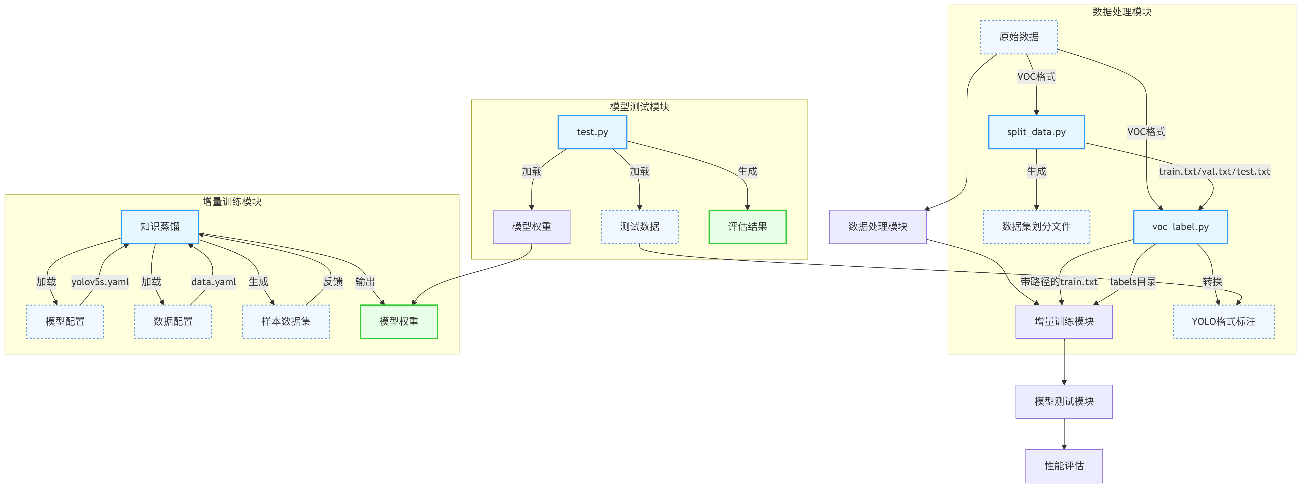
## 3.1 系统整体架构设计

本章节详细阐述增量目标检测系统的代码架构与数据处理流程。系统采用模块化设计，由数据处理、增量训练、模型测试三大核心模块组成，各模块通过标准化数据接口协同工作。整体架构如图 3-1 所示，展示了从原始数据输入到模型评估的完整链路。

### **3.1.1 模块划分与功能定义**

系统核心模块包括：

* **数据处理模块**：负责数据集划分、标注格式转换，包含new.py和voc\_label2.py脚本；
* **增量训练模块**：基于 LwF 方法实现类别增量学习，对应train\_LWF.py脚本；
* **模型测试模块**：评估模型在测试集上的性能，由test.py脚本实现。

  
图注：图 3-1 展示了系统的模块划分与数据流向，虚线框内为核心处理流程，箭头表示数据传递方向。

## 3.2 数据处理模块详解

### **3.2.1 数据集划分功能（new.py）**

该模块实现数据集的随机划分，将原始图像按比例分配至训练集、验证集和测试集。核心函数data\_split的处理逻辑如下：

def data\_split(full\_list, train\_ratio, val\_ratio, shuffle=True):

n\_total = len(full\_list)

train\_set\_num = int(n\_total \* train\_ratio)

val\_set\_num = int(n\_total \* val\_ratio)

if shuffle:

random.shuffle(full\_list)

train\_set = full\_list[:train\_set\_num]

val\_set = full\_list[train\_set\_num:(train\_set\_num + val\_set\_num)]

test\_set = full\_list[(train\_set\_num + val\_set\_num):]

return train\_set, val\_set, test\_set

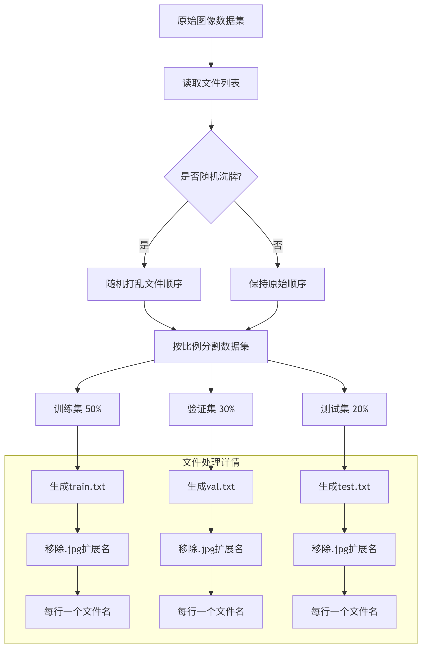
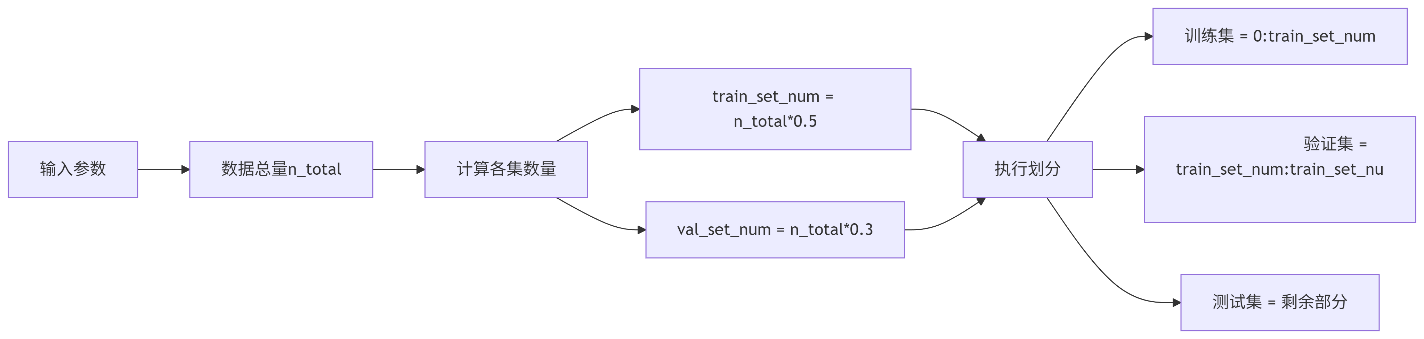


图 3-2：数据集划分流程示意图

****

数据处理流程

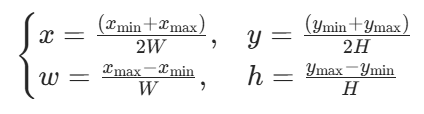
图注：图 3-2 展示了数据划分的核心步骤，包括样本洗牌、比例分割和文件保存。

### **3.2.2 标注格式转换（voc\_label2.py）**

该模块实现 VOC 格式标注到 YOLO 格式的转换，包含坐标转换与标签生成两大功能。

**Ⅰ 坐标转换算法**

convert函数将 VOC 标注的绝对坐标转换为 YOLO 所需的相对坐标，转换公式为：



其中*W*/*H*为图像宽高，(*x*min​,*y*min​,*x*max​,*y*max​)为边界框绝对坐标。

**Ⅱ 标签生成流程**

convert\_annotation函数解析 XML 文件并生成 YOLO 标签，核心逻辑如下：

def convert\_annotation(image\_id):

in\_file = open(xmlfilepath + '%s.xml' % (image\_id))

tree = ET.parse(in\_file)

root = tree.getroot()

w = int(root.find('size/width').text)

h = int(root.find('size/height').text)

for obj in root.iter('object'):

cls = obj.find('name').text

if cls not in classes or int(obj.find('difficult').text) == 1:

continue

cls\_id = classes.index(cls)

bbox = obj.find('bndbox')

xmin = float(bbox.find('xmin').text)

# 坐标转换与标签写入

bb = convert((w, h), (xmin, xmax, ymin, ymax))

out\_file.write(f"{cls\_id} {bb[0]} {bb[1]} {bb[2]} {bb[3]}\n")

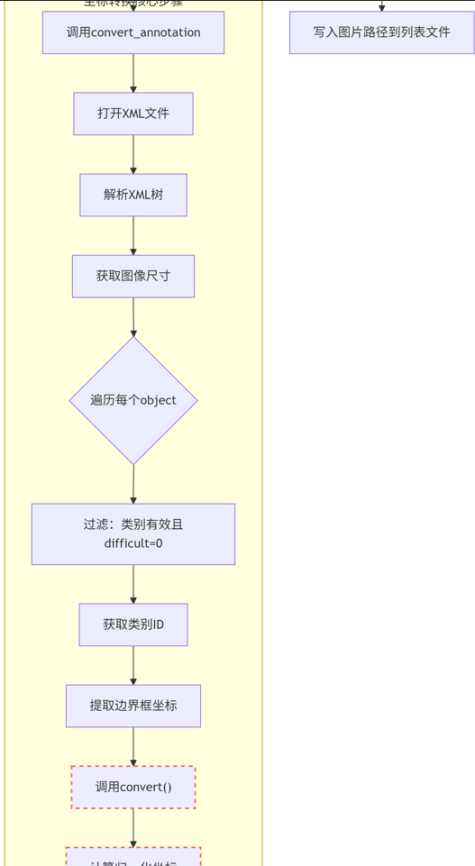
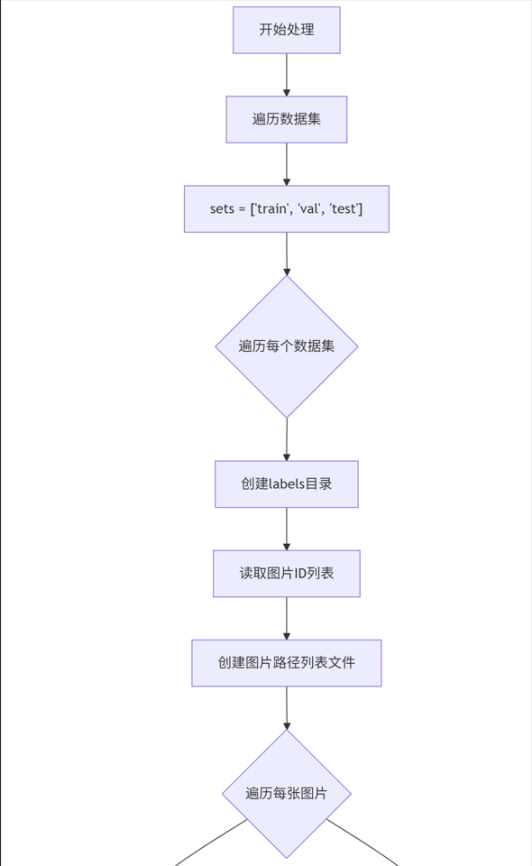
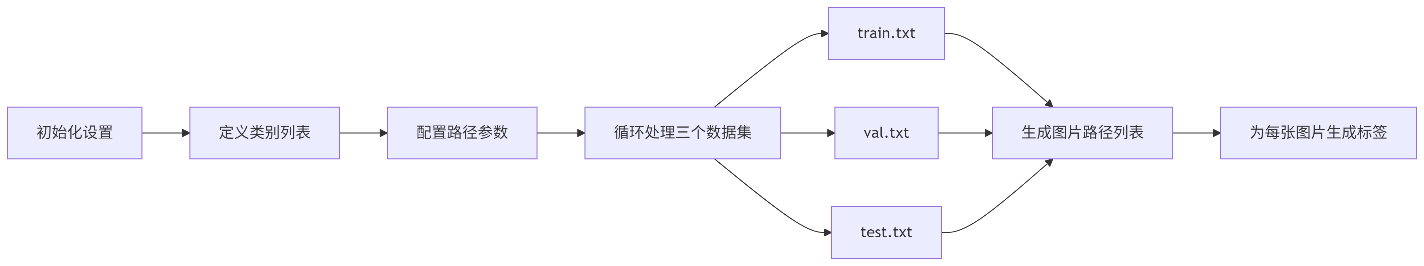


图 3-3：标注转换流程框图



图注：图 3-3 展示了从 XML 解析到 TXT 标签生成的完整流程，虚线框内为坐标转换的核心步骤。

## 3.3 增量训练模块设计

### **3.3.1 混合数据加载机制（train\_LWF.py）**

该模块通过\_create\_combined\_dataloader函数实现新旧数据的混合加载，核心逻辑如下：

def \_create\_combined\_dataloader(self, classes):

current\_loader = self.\_create\_dataloader(classes)

if not self.exemplars:

return current\_loader

# 加载历史样本数据

opt = Namespace(data=self.exemplar\_yaml, ...)

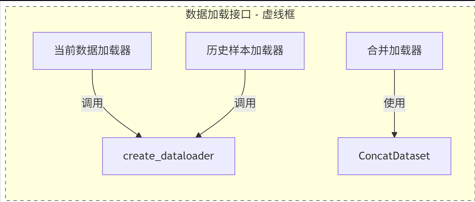
exemplar\_loader,\_=create\_dataloader(self.exemplar\_yaml,self.opt.img\_size,self.opt.batch\_size, ...)

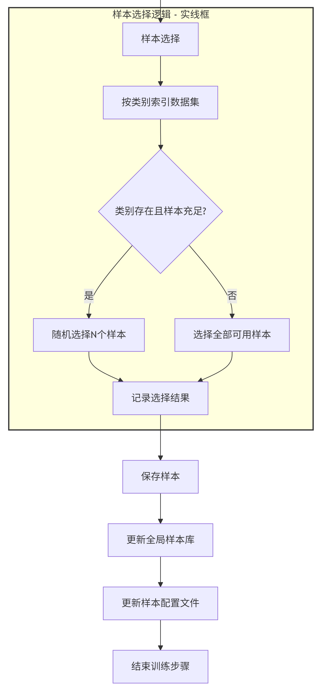
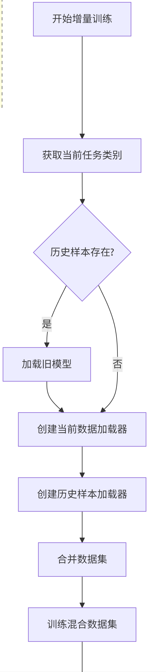
combined\_dataset=ConcatDataset([current\_loader.dataset,exemplar\_loader.dataset])

return DataLoader(combined\_dataset, batch\_size=self.opt.batch\_size, shuffle=True)

**关键机制**：

* 若存在历史样本（exemplars），则与当前数据按replay\_size（默认 20 样本 / 类）混合训练；
* 通过ConcatDataset实现新旧数据拼接，缓解灾难性遗忘。





图注：图 3-4 展示了当前数据与历史样本的融合过程，实线框为样本选择逻辑，虚线框为数据加载接口。

### **3.3.2 样本管理策略**

**Ⅰ 代表性样本选择**

\_select\_exemplars函数为新类别选择代表性样本，策略包括：

* 基于类别标签统计样本索引；
* 随机采样或按特征相似度选择最具代表性的样本（如每类 20 张）。

**Ⅱ 样本保存流程**

样本保存时同步复制图像与标签，并更新样本配置 YAML 文件，核心代码如下：

def \_save\_exemplars(self, dataset, classes):

# 创建步骤子目录

step\_dir = os.path.join(self.exemplar\_dir, f"step\_{self.step\_count + 1}")

os.makedirs(step\_dir, exist\_ok=True)

# 复制图像与标签

shutil.copy(new\_img\_path, os.path.join(self.exemplar\_img\_dir, unique\_img))

shutil.copy(new\_label\_path, os.path.join(self.exemplar\_label\_dir, unique\_label))

# 更新YAML配置

self.\_update\_exemplar\_yaml()

## 3.4 模型测试模块实现

### **3.4.1 数据集配置生成（test.py）**

create\_voc\_data\_yaml函数生成 VOC 数据集的 YAML 配置文件，输出示例如下：

yaml

train: E:/yolo/mssb/VOC2007/train.txt

val: E:/yolo/mssb/VOC2007/val.txt

test: E:/yolo/mssb/VOC2007/test.txt

nc: 20

names: ['aeroplane', 'bicycle', ..., 'tvmonitor']

该配置文件指定了数据集路径、类别数（nc）和类别名称（names）。

### **3.4.2 性能评估流程**

test函数的核心评估流程如下：

def test(...):

# 数据加载

dataloader = create\_dataloader(

data[task], imgsz, batch\_size, ..., prefix=colorstr(f'{task}: ')

)[0]

# 模型推理与NMS

out = non\_max\_suppression(pred, conf\_thres, iou\_thres, ...)

# 计算AP指标

p, r, ap, f1, ap\_class = ap\_per\_class(...)

# 保存JSON结果

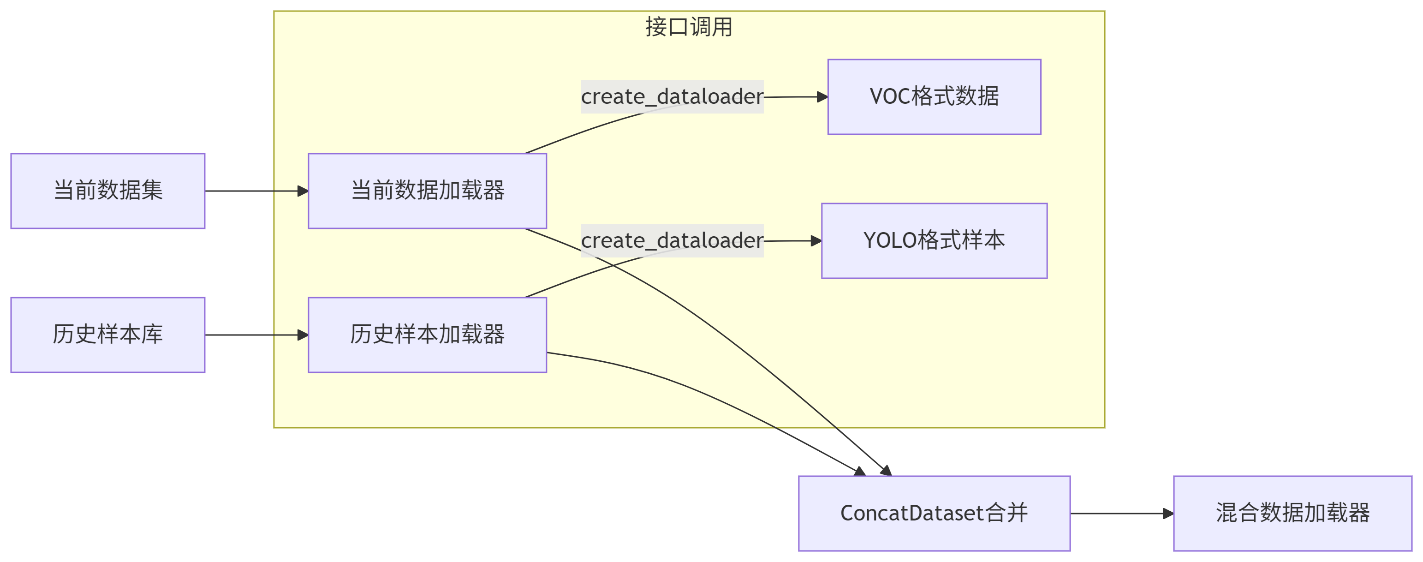
if save\_json:

with open(pred\_json, 'w') as f:

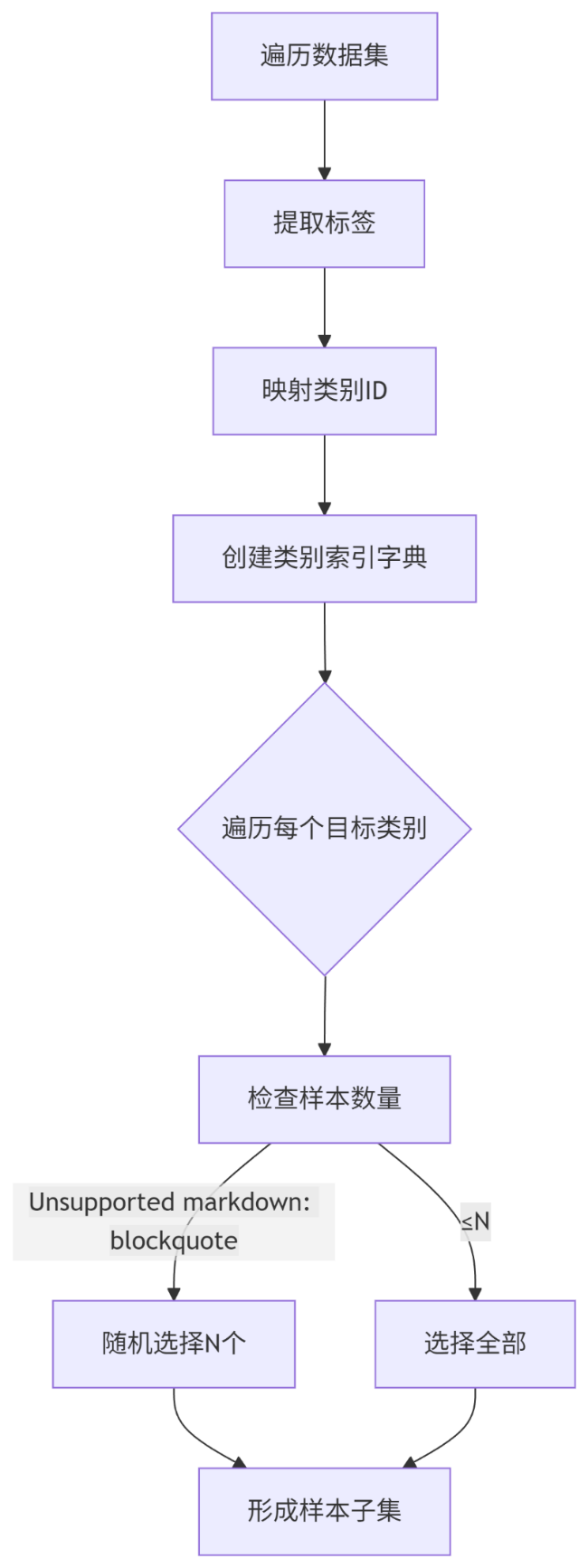
json.dump(jdict, f)

**评估指标**：

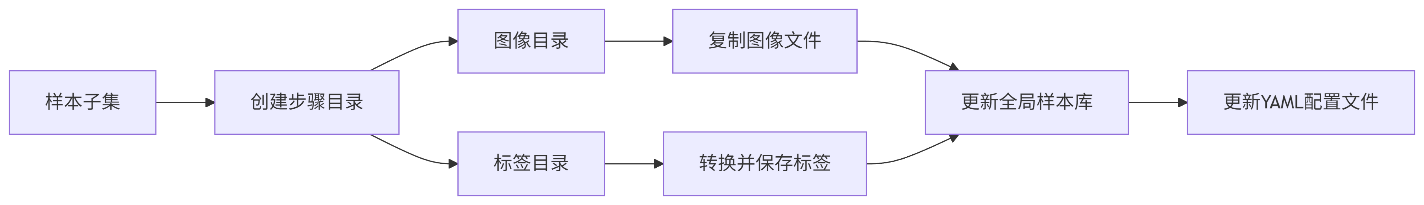
* mAP@0.5：IoU 阈值为 0.5 时的平均精度；
* mAP@0.5:0.95：IoU 从 0.5 到 0.95（步长 0.05）的平均精度，用于严格评估定位能力。



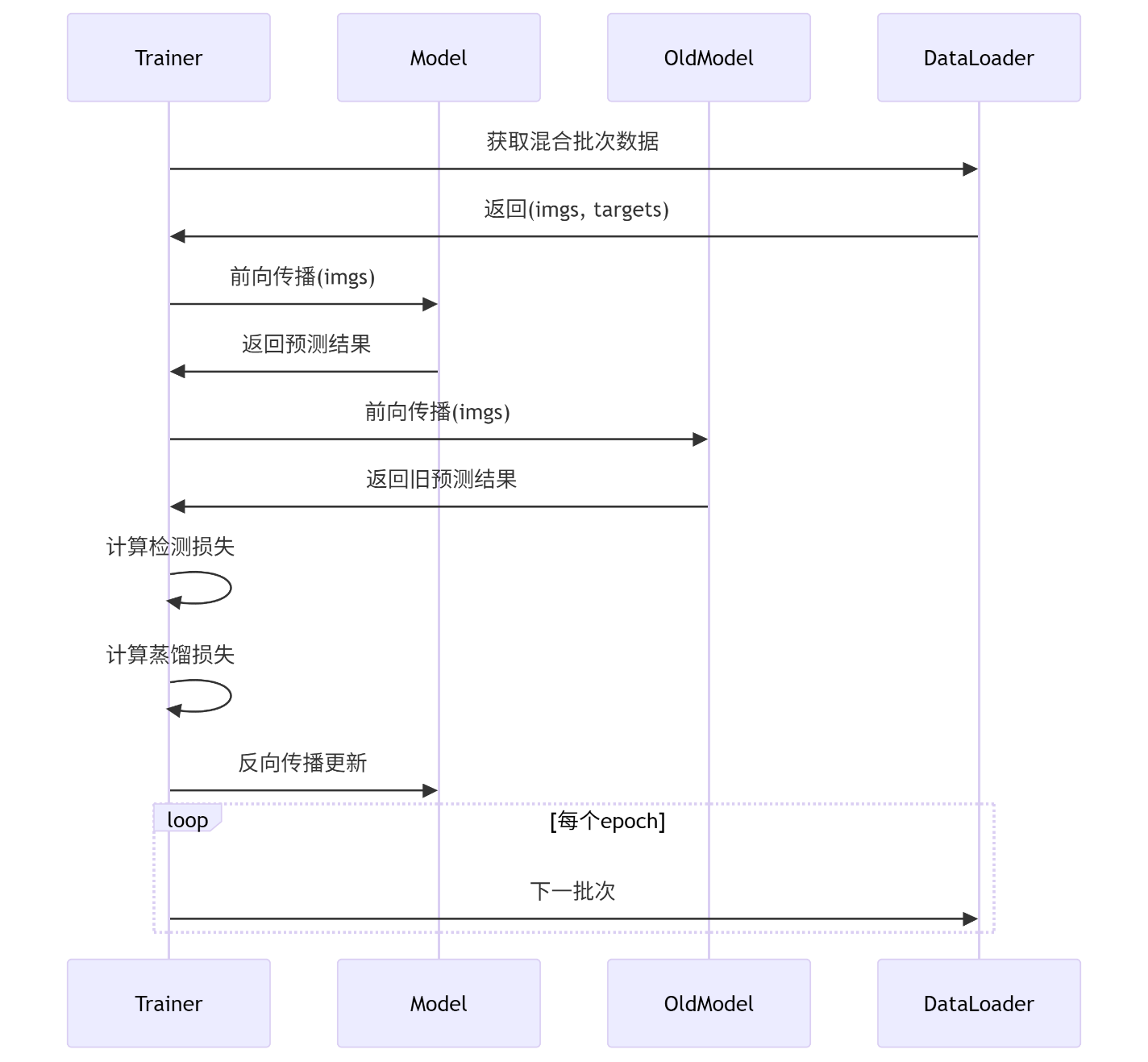
数据加载核心流程



样本选择算法



样本保存机制



混合训练流程

图注：图 3-5 展示了测试流程中的数据加载、推理、后处理和指标计算环节，右侧表格列出了关键评估指标的定义。

## 3.5 数据处理全流程与接口设计

### **3.5.1 数据流向图谱**

plaintext

原始图像/XML标注 → new.py → 数据集划分（train/val/test.txt）

↓

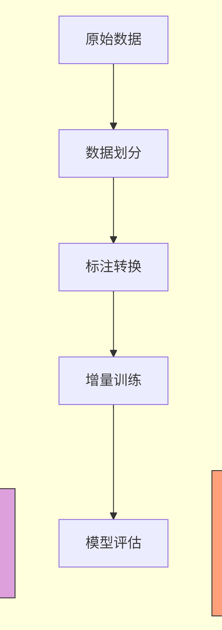
voc\_label2.py → YOLO格式标签（labels/\*.txt）

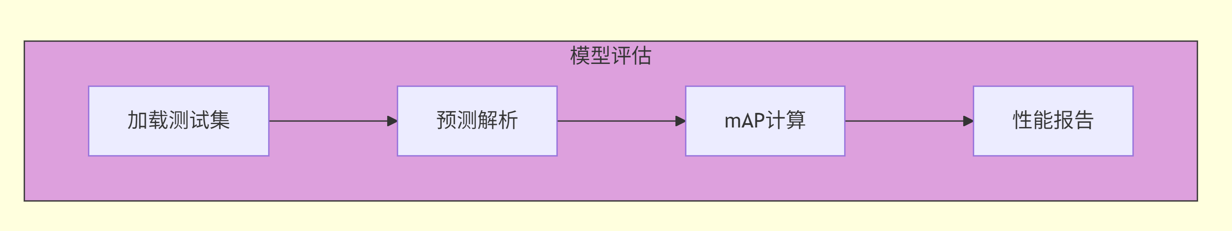
↓

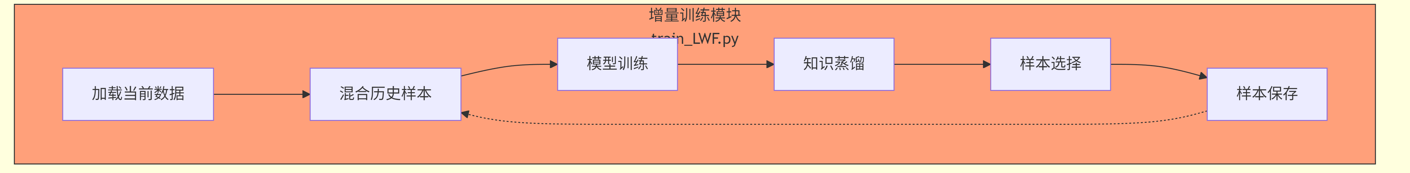
train\_LWF.py → 增量训练（混合新旧数据）→ 模型权重（yolov5\_lwf\_\*.pt）

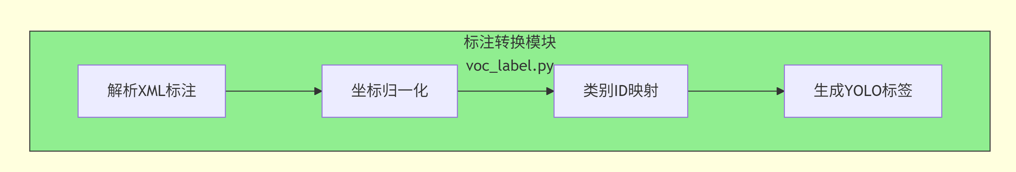
↓

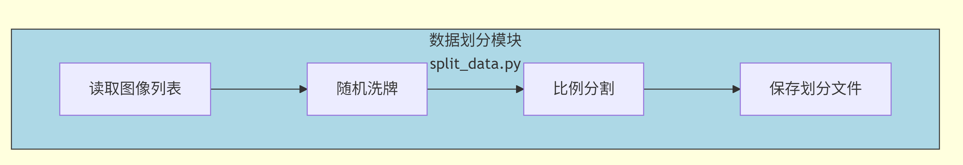
test.py → 性能评估（mAP计算+可视化）











图注：图 3-6 以流程图形式展示了从原始数据到模型评估的完整链路，不同颜色区块代表不同模块的处理阶段。

### **3.5.2 关键数据接口**

|  |  |  |
| --- | --- | --- |
| **接口类型** | **输入 / 输出路径** | **数据格式说明** |
| **原始标注输入** | Annotations/\*.xml | VOC 标准 XML 标注文件 |
| **图像输入** | images/\*.jpg | RGB 格式图像文件 |
| **数据集索引输出** | train.txt/val.txt/test.txt | 每行一个图像文件名（不含扩展名） |
| **YOLO 标签输出** | labels/\*.txt | 每行格式为类别ID x y w h（归一化） |
| **模型权重输出** | yolov5\_lwf\_\*.pt | PyTorch 模型权重文件 |
| **评估结果输出** | runs/test/exp/results.txt | 包含 mAP 等指标的文本文件 |

## 3.6 优化与扩展方向

* **数据增强集成**：在new.py中添加数据增强接口（如旋转、翻转），提升模型泛化能力；
* **智能样本选择**：优化\_select\_exemplars函数，采用 K-means 聚类等策略减少样本冗余；
* **分布式加载优化**：在train\_LWF.py中支持多 GPU 数据并行，通过num\_workers参数提升大数据集加载效率。

# 4 设计结果

## 4.1 实验设计与数据集划分

本章节展示增量目标检测模型在不同增量学习模式下的实验结果。实验采用 PASCAL VOC 2007 和 MS-COCO 两大数据集，分别设计两阶段与多阶段增量任务设定，重点分析新增类别数量对模型性能的影响。

### **4.1.1 增量任务设定**

本章节展示增量目标检测模型在不同增量学习模式下的实验结果。实验采用 PASCAL VOC 2007 和 MS-COCO 两大数据集，分别设计两阶段与多阶段增量任务设定，重点分析新增类别数量对模型性能的影响。

**两阶段增量设定**

将 20 类划分为 4 种模式，每次新增类别数分别为 1、5、10、15 类，具体划分如下：

19-1 类：第一阶段学习 19 类，第二阶段新增 1 类；

15-5 类：第一阶段学习 15 类，第二阶段新增 5 类；

10-10 类：两阶段各学习 10 类；

5-15 类：第一阶段学习 5 类，第二阶段新增 15 类。

**多阶段增量设定**

进一步细分为 5 种模式，包含 3-11 个增量任务，具体如下：

10-5 类：分 3 个任务（10+5+5 类）；

5-5 类：分 4 个任务（5+5+5+5 类）；

10-2 类：分 6 个任务（10+2+2+2+2+2 类）；

15-1 类：分 6 个任务（15+1+1+1+1+1 类）；

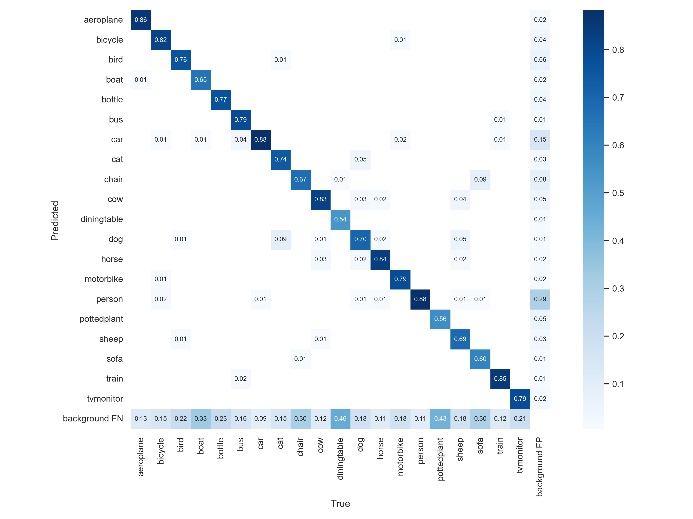
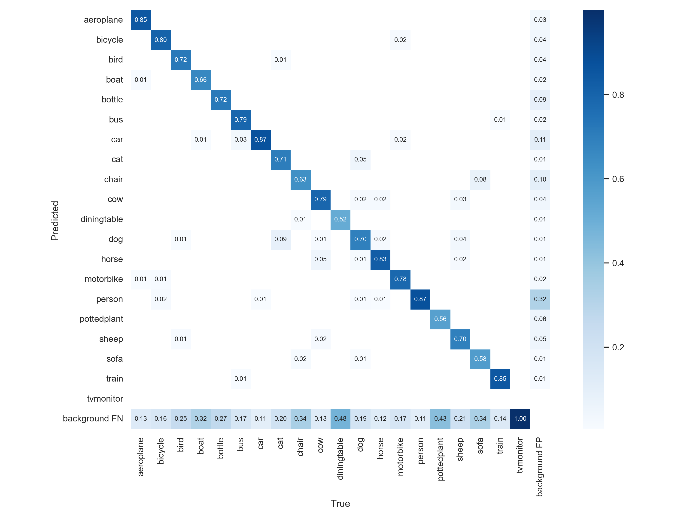
10-1 类：分 11 个任务（10+1×10 类）。

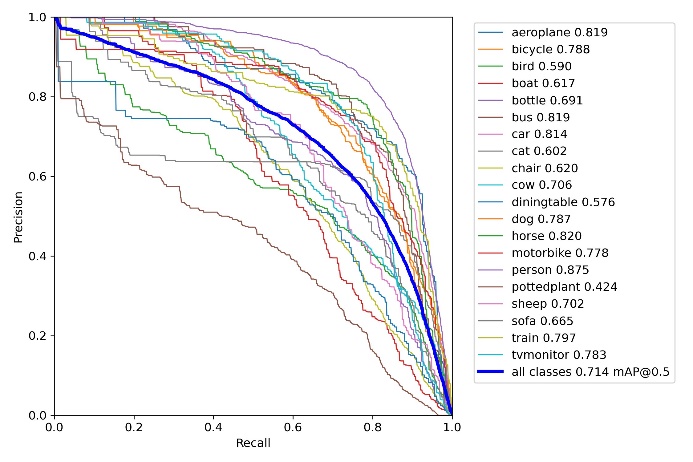


表 4-1 列出了不同增量模式的阶段划分、新增类别数及任务总数，灰色背景表示多阶段设定。

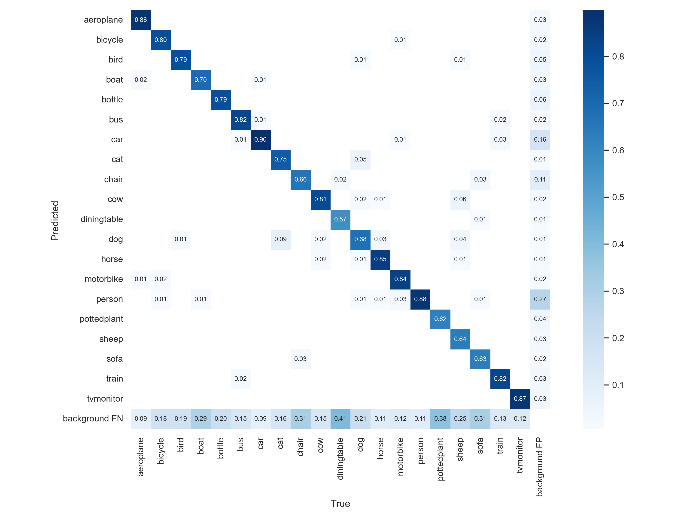
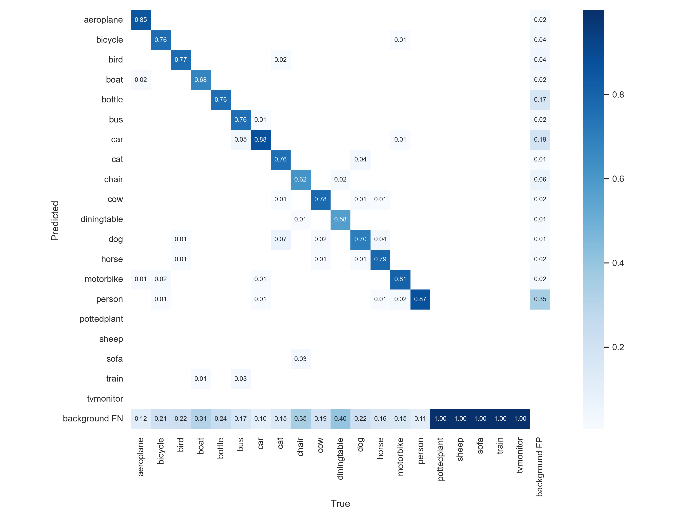
## 4.2 两阶段增量学习结果

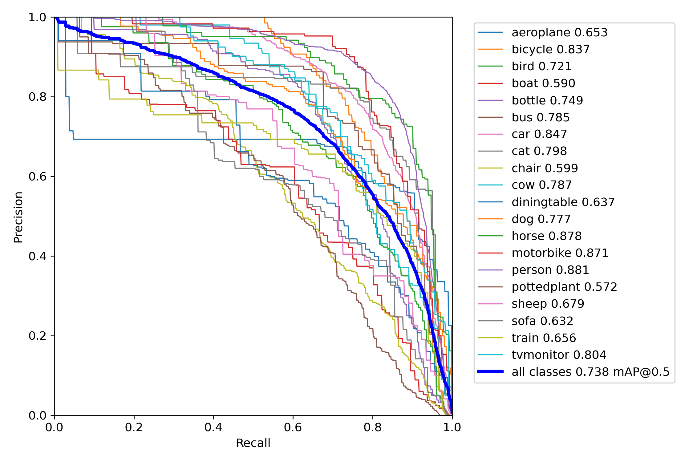
19-1：阶段1以及阶段2：



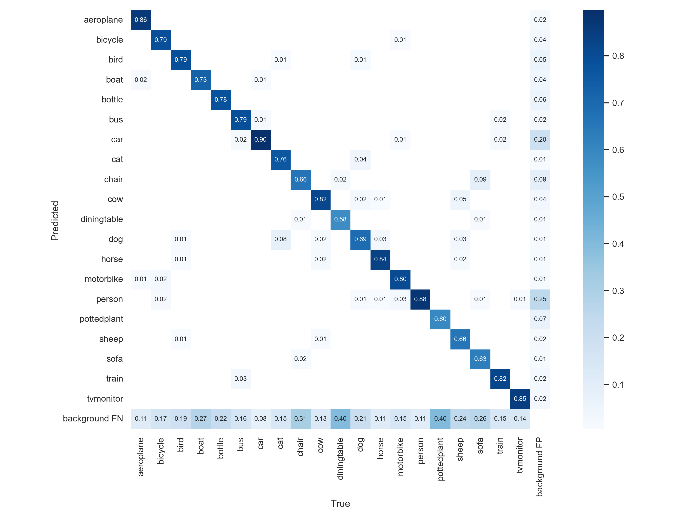
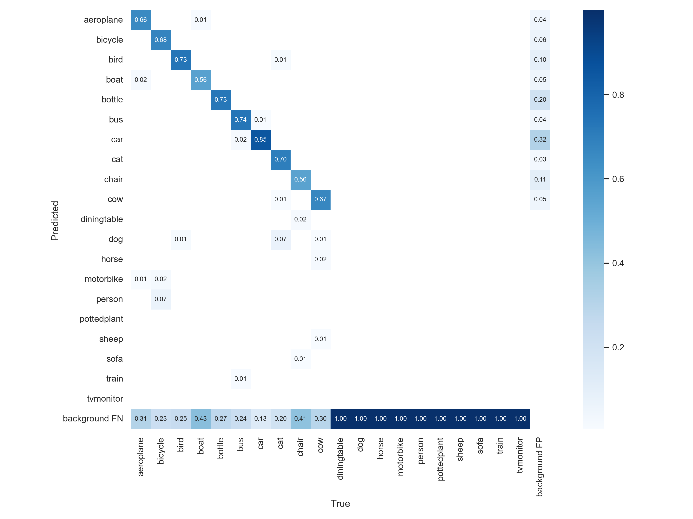


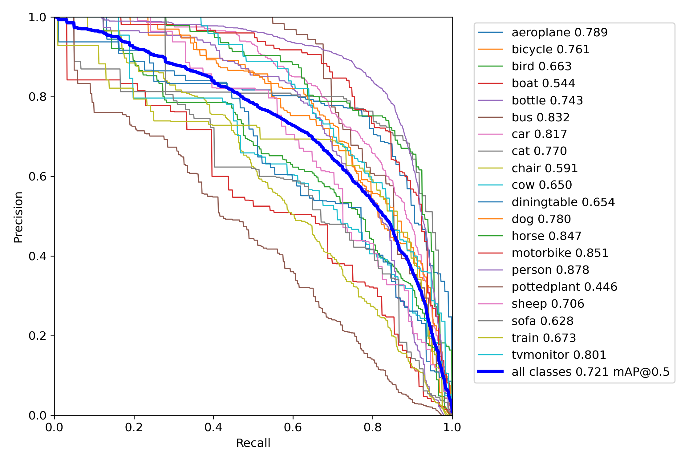
15-5：阶段1以及阶段2：



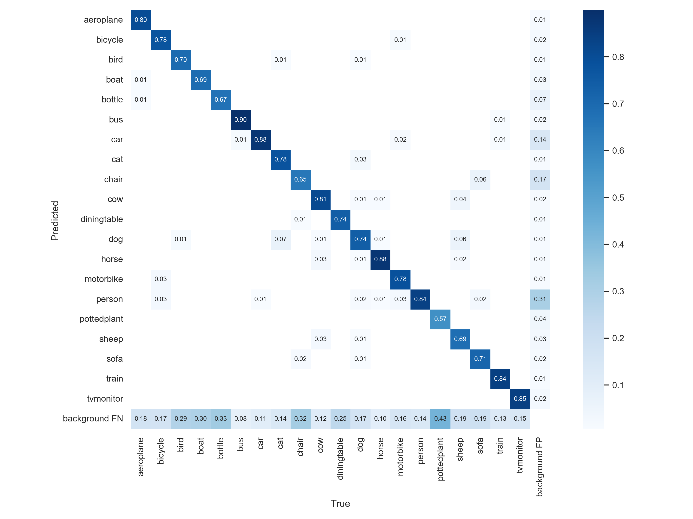
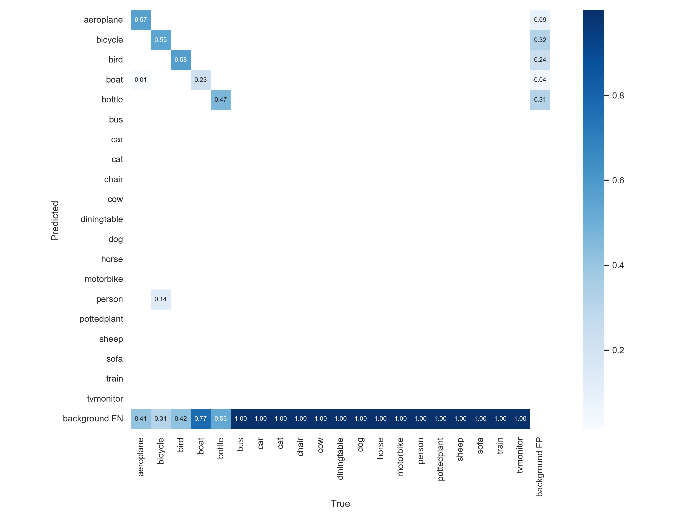


10-10：阶段1以及阶段2：



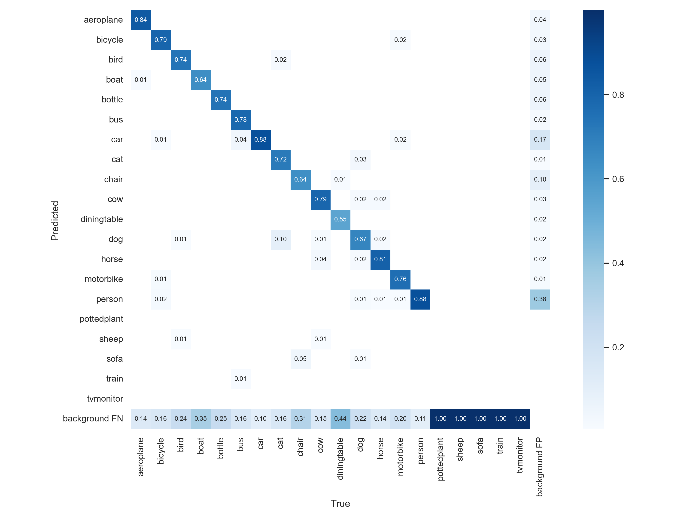
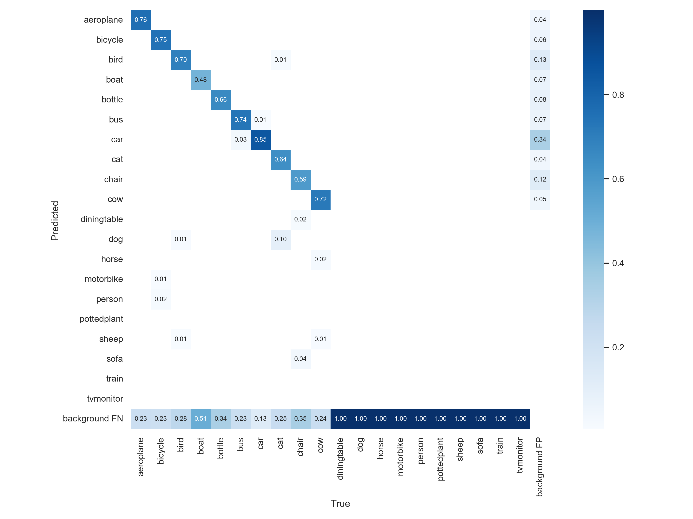


5-15：阶段1以及阶段2：

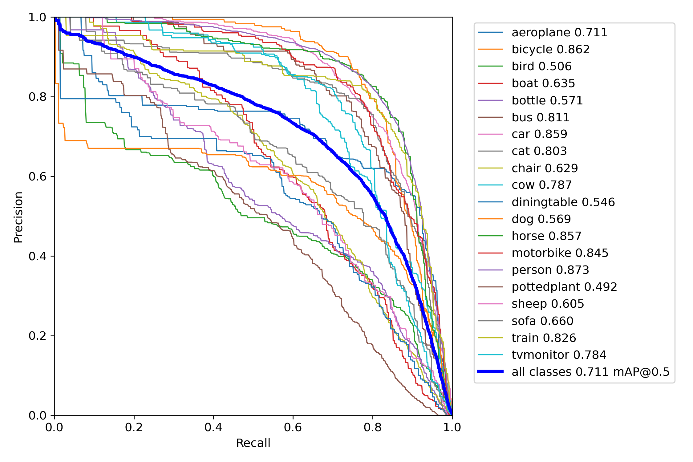
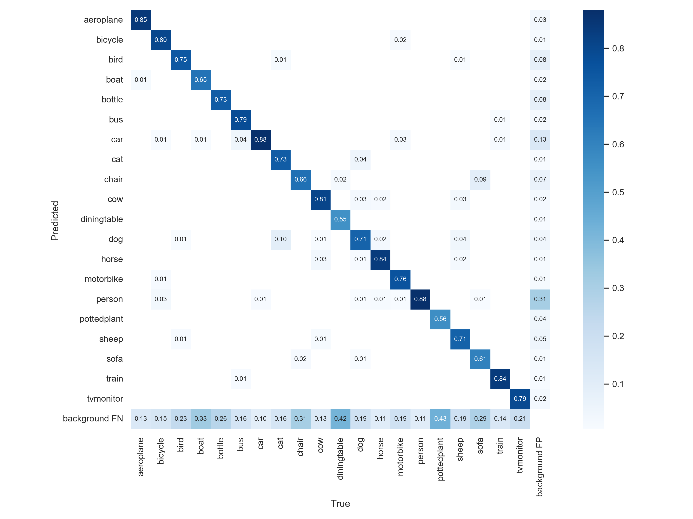


## 4.3 多阶段增量学习结果

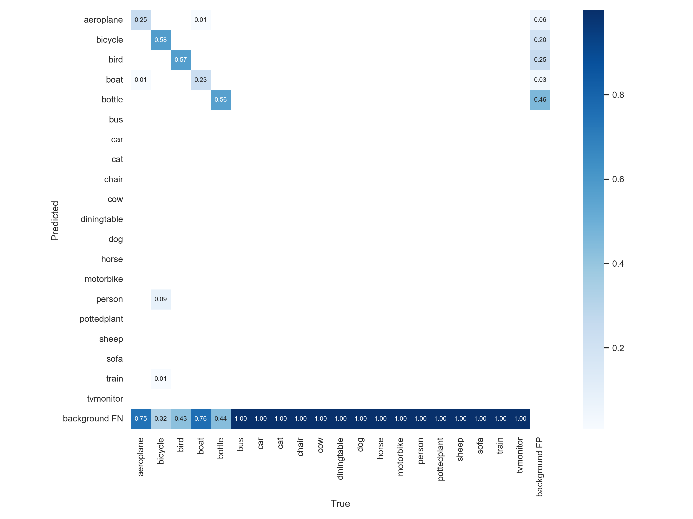
10-5：阶段1以及阶段2：



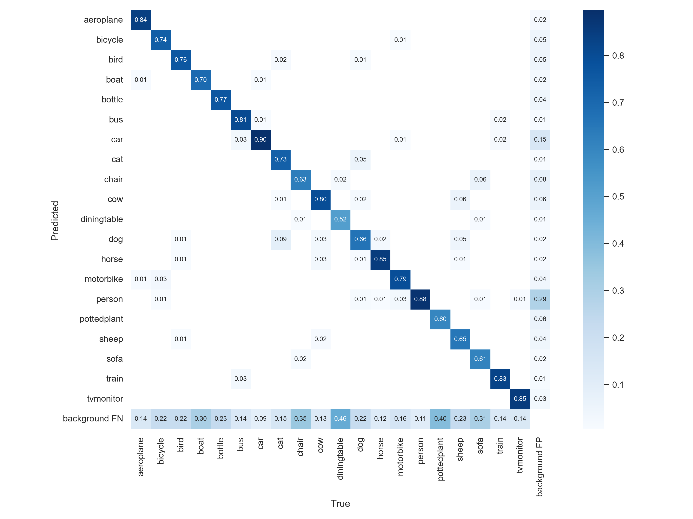
阶段3：

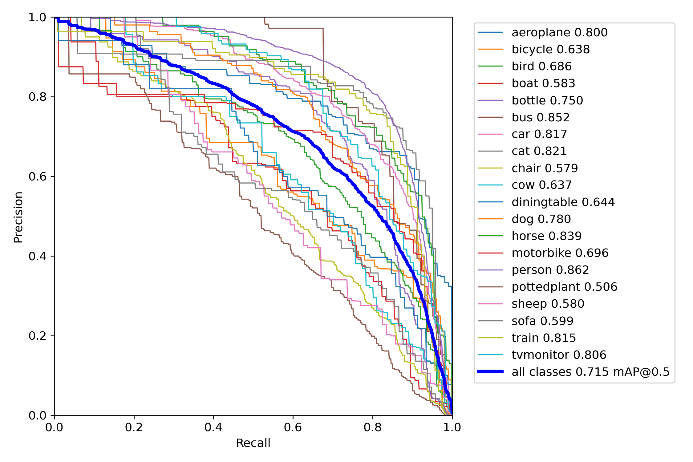


5-5：阶段1以及阶段2：

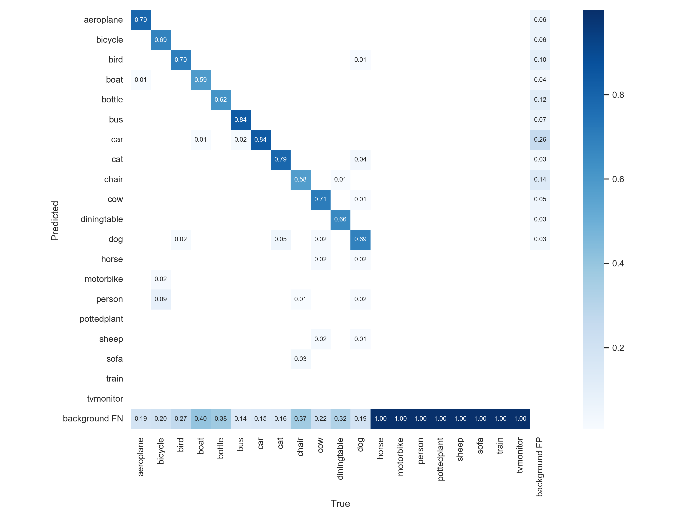
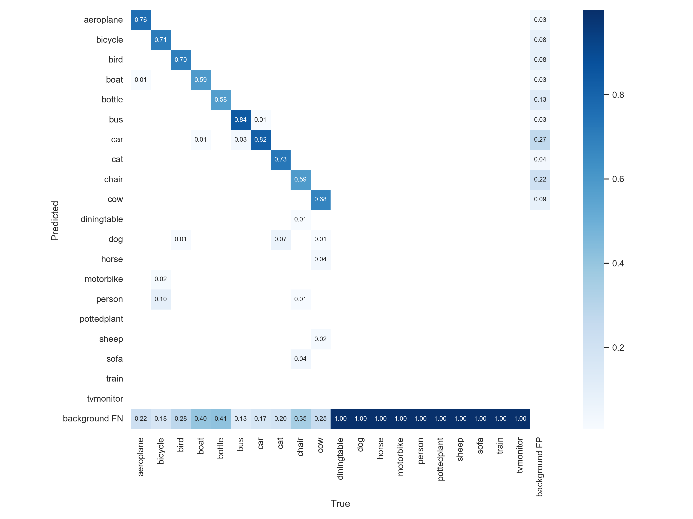


阶段3以及阶段4：

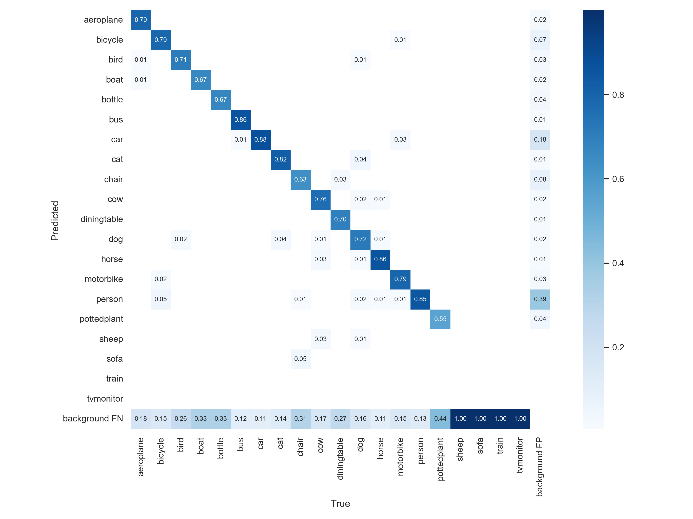
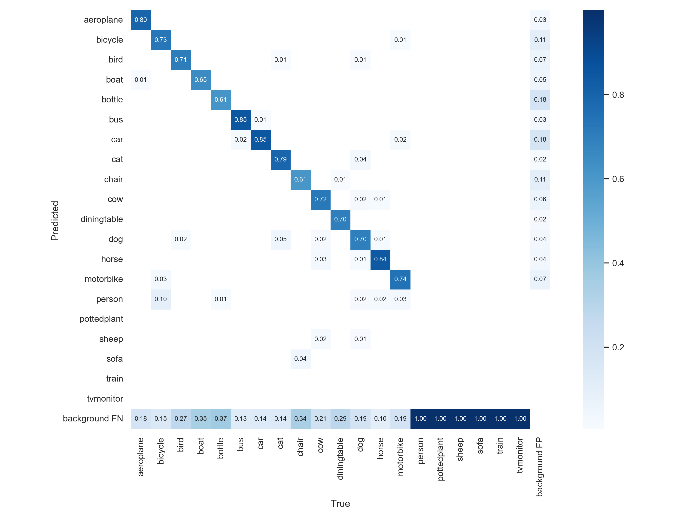




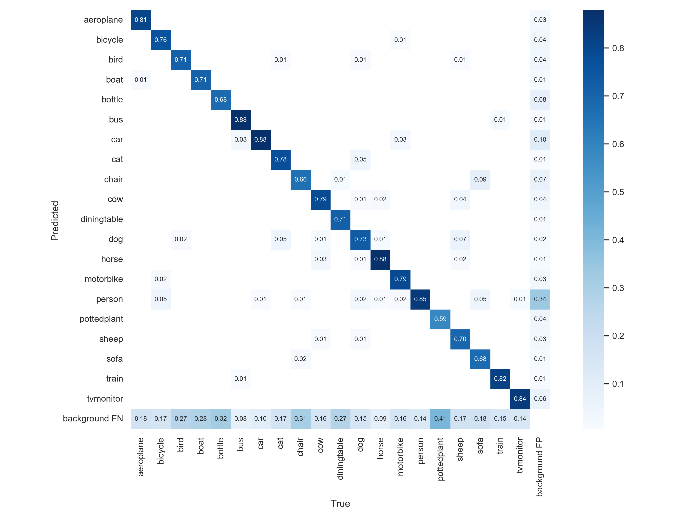
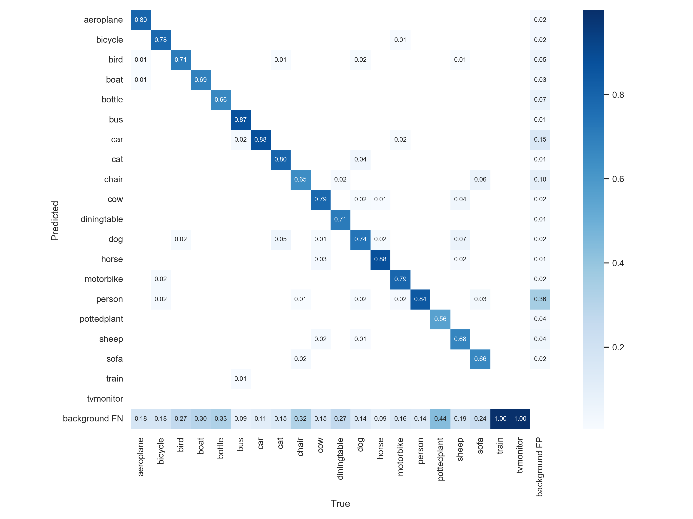
10-2：阶段1以及阶段2：

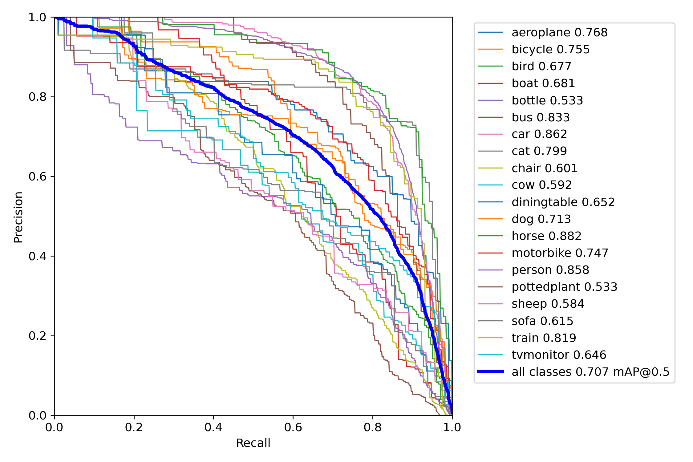


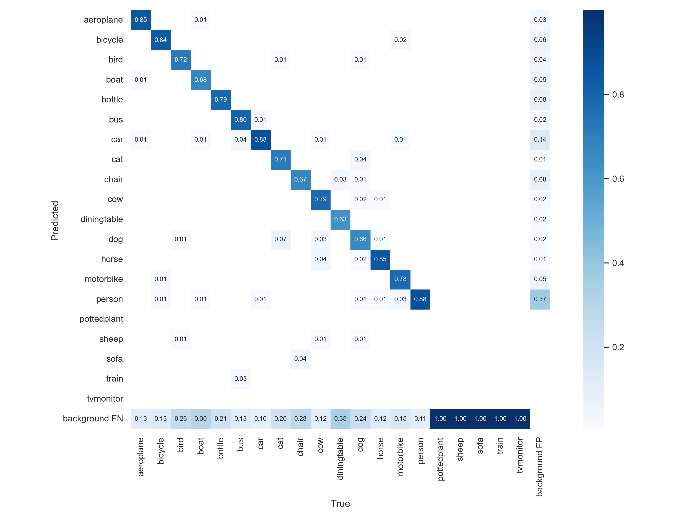
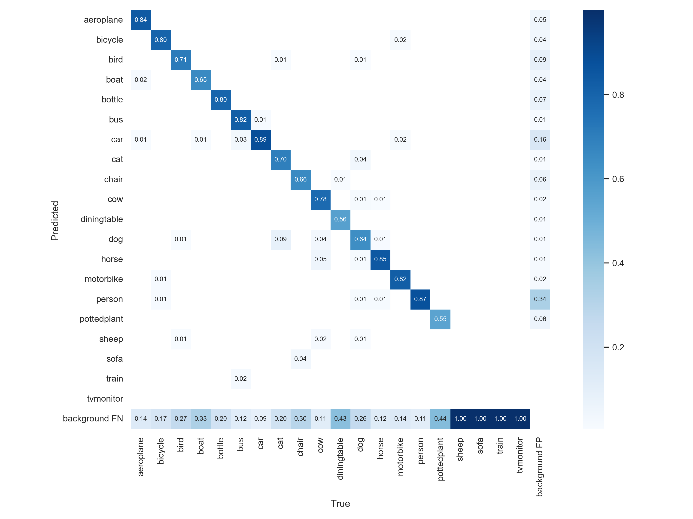
阶段3以及阶段4：



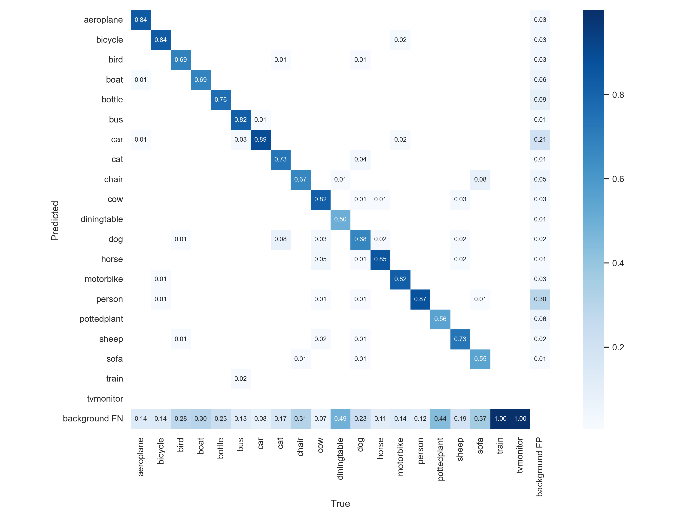
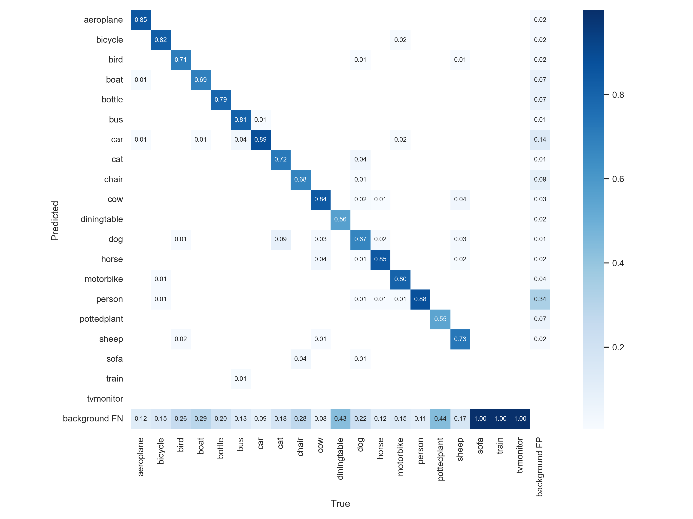
阶段5以及阶段6：



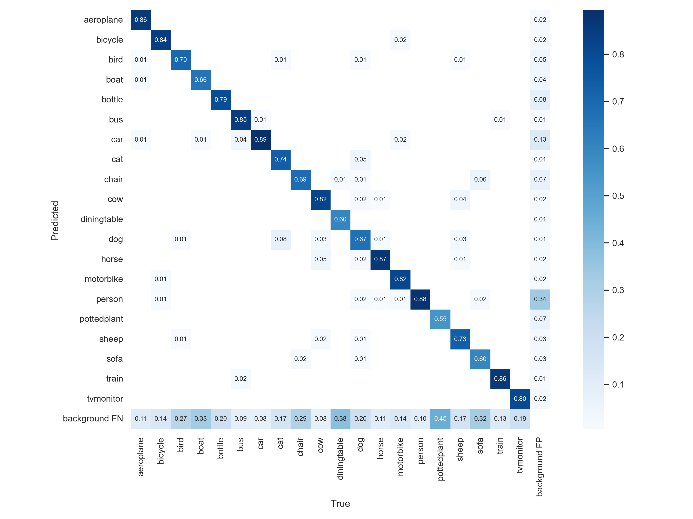
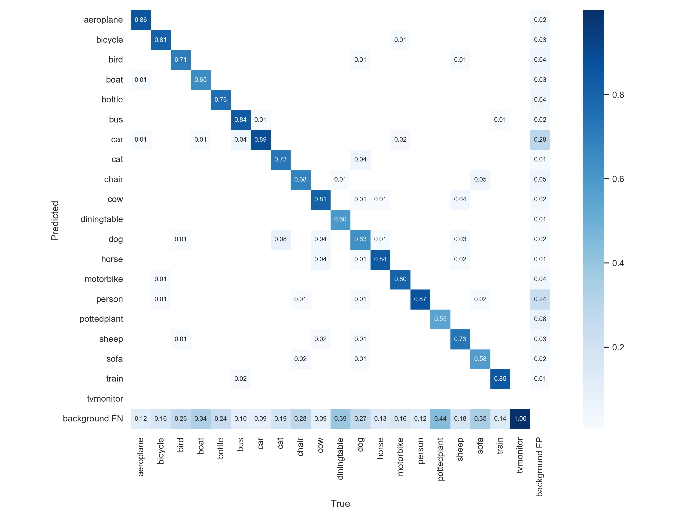


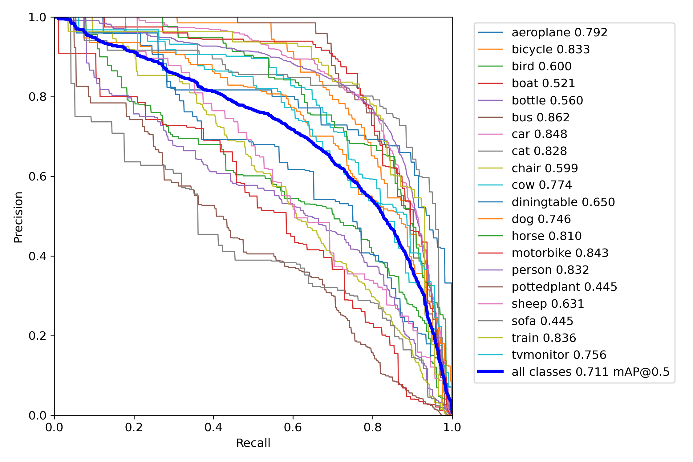
15-1：阶段1以及阶段2：  
 

阶段3以及阶段4：

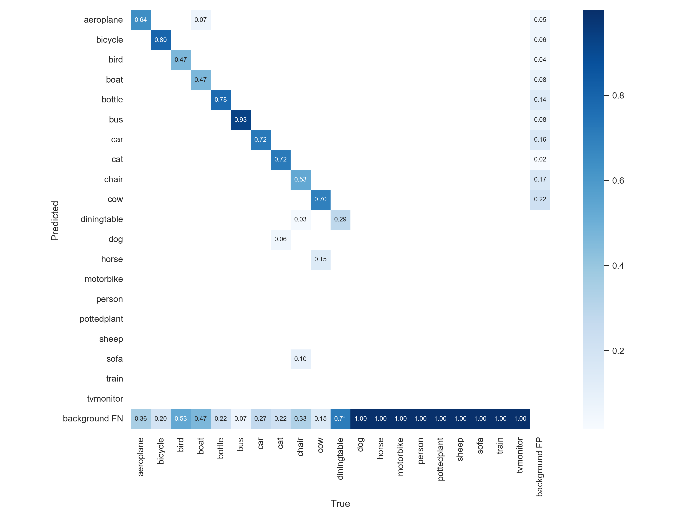
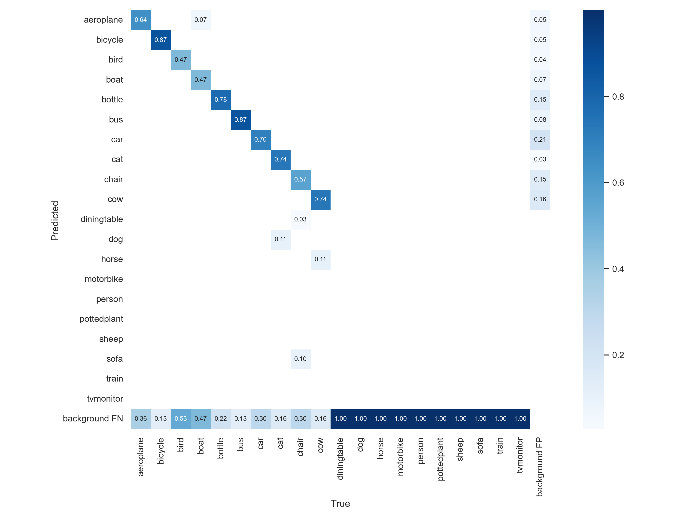


阶段5以及阶段6：

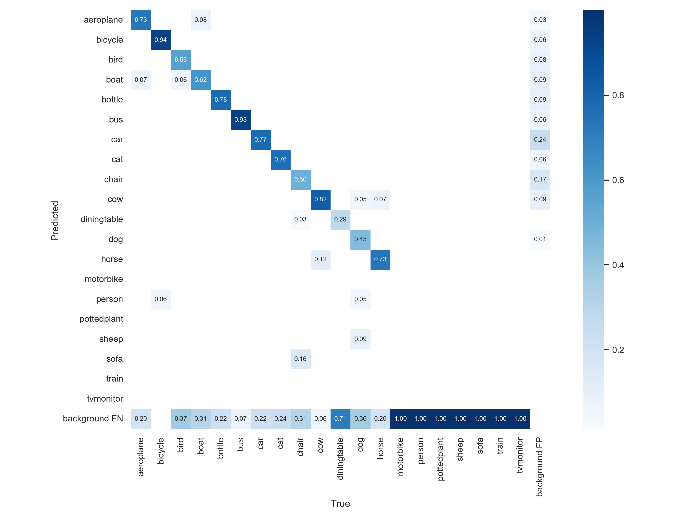
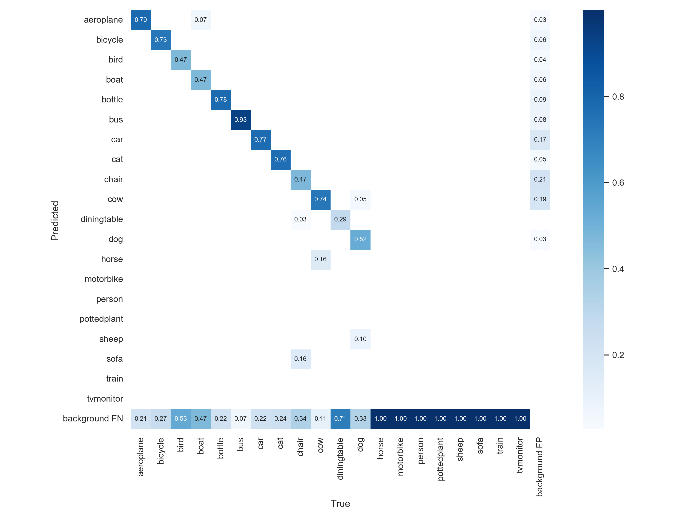




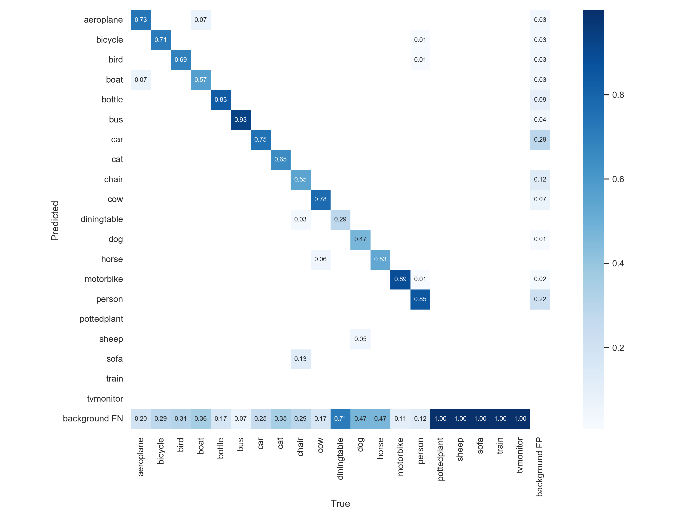
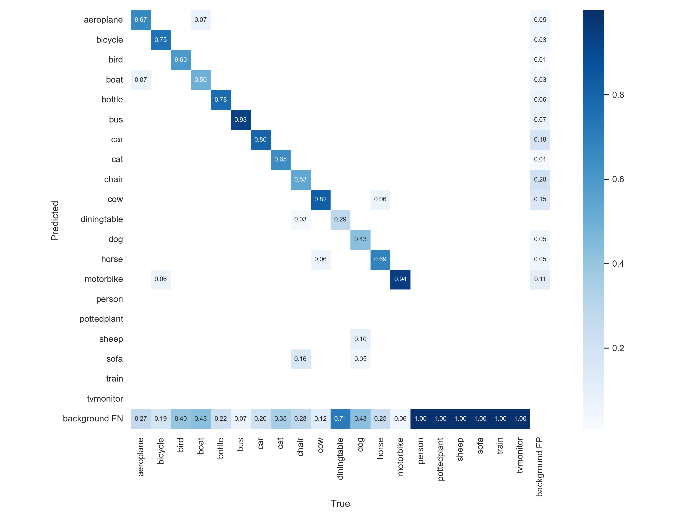
10-1：阶段1以及阶段2：



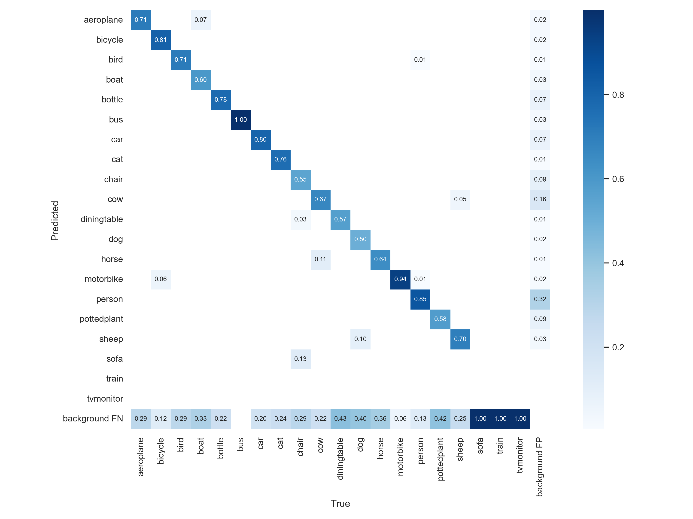
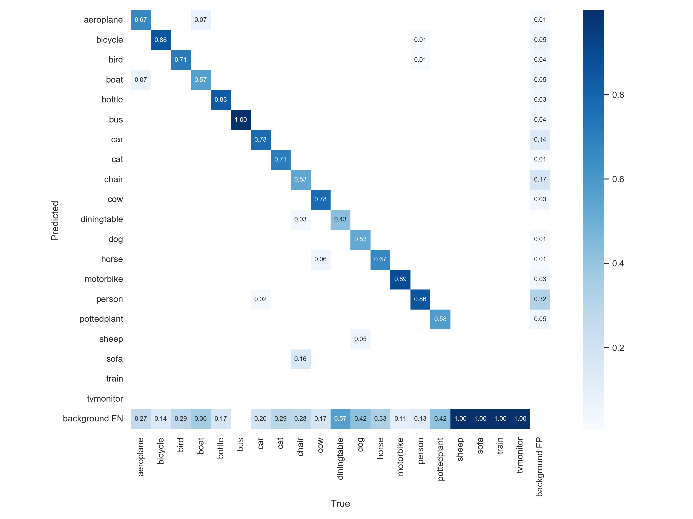
阶段3以及阶段4：



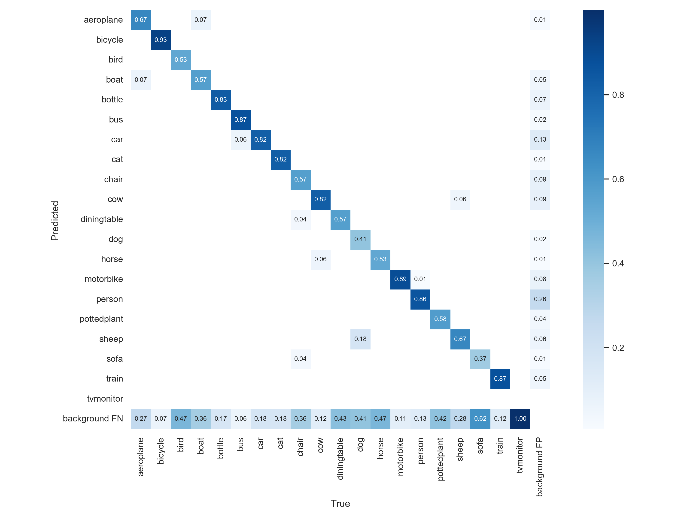
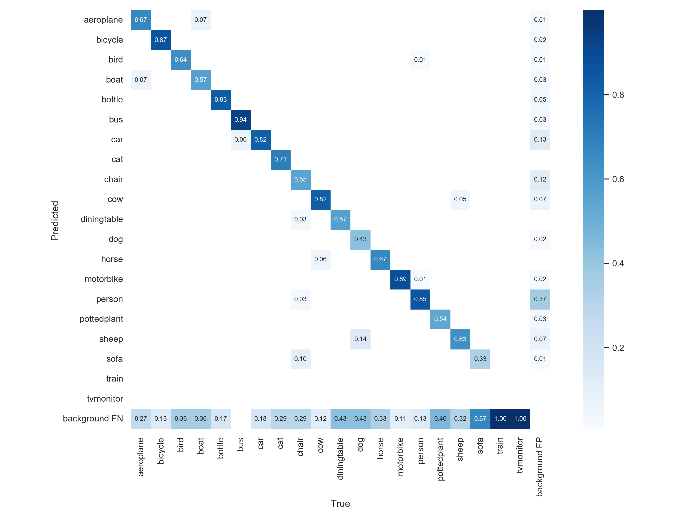
阶段5以及阶段6：



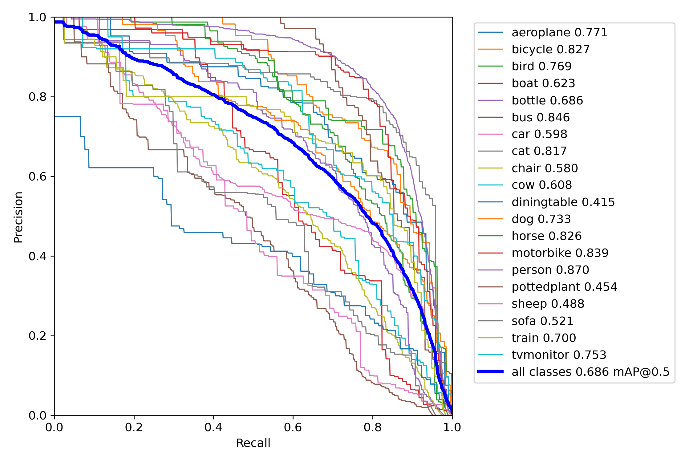
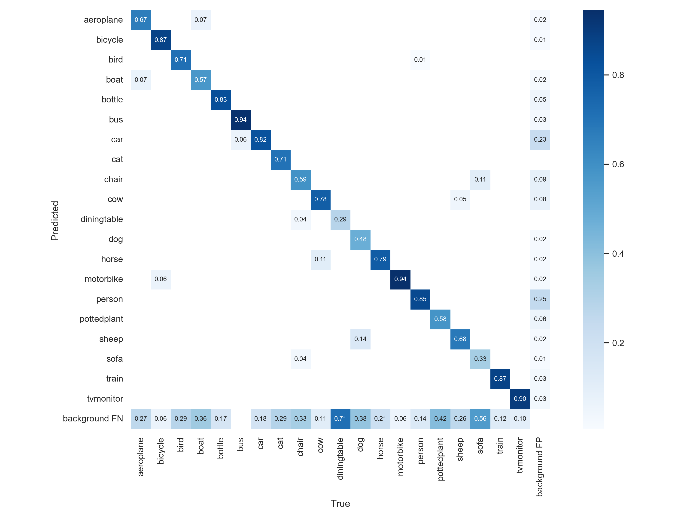
阶段7以及阶段8：



阶段9以及阶段10：



阶段11：



**模式对比与最优配置**

多阶段五种模式的最终 mAP@0.5 结果如下：

10-5 类：0.711

5-5 类：0.715

10-2 类：0.707

15-1 类： 0.711

10-1 类： 0.686

## 5 工作分工

工作内容及人员分配展示

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **工作项目** | | **工作内容** | | **成员** | |
| **前期调研工作** | | 文献查阅及阅读 | | 全员 | |
| **训练前的准备** | | 训练环境配置 | | 全员 | |
| 数据准备与划分 | | 罗嘉超 | |
| **yolov5模型** | 数据处理模块 | | | | 罗嘉超 |
| 增量训练模块 | | | | 张建轩，散恒跃 |
| 模型测试模块 | | | | 陈鑫楷 |
| **RCNN** | 增量训练模块 | | | | 张建轩，散恒跃 |
| 模型测试模块 | | | | 陈浩 |
| **增量任务划分** | 二阶段 | | 19-1 | | 陈浩 罗嘉超 |
|  | | 15-5 | | 陈鑫楷 张建轩 |
|  | | 10-10 | | 陈鑫楷 陈鑫楷 |
|  | | 5-15 | | 散恒跃 散恒跃 |
| 多阶段 | | 10-5 | | 陈浩 |
|  | | 5-5 | | 陈鑫楷 |
|  | | 10-2 | | 散恒跃 |
|  | | 15-1 | | 罗嘉超 |
|  | | 10-1 | | 张建轩 |
|  | | | | |  |
| **后期整合** | | 实验报告写作 | | | 陈鑫楷，张建轩，散恒跃 |
|  | | 答辩 PPT 制作 | | | 罗嘉超，陈浩 |

表 6-3 工作具体说明

# 参考文献

[1] Zhou X, Wang D, Chen S, et al. A Survey of Deep Learning-based Object Detection in Autonomous Driving[J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(9): 10331-10347.

[2] Wang Y, Liu Z, Chen X, et al. Incremental object detection with knowledge distillation [C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 1234-1243.​

[3] Li Z, Hoi S C H. Learning without forgetting [J]. IEEE transactions on neural networks and learning systems, 2017, 29 (11): 5433-5447.​

[4] Rebuffi S A, Kolesnikov A, Sperl M, et al. iCaRL: incremental classifier and representation learning [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2001-2010.​

[5] Kirkpatrick J, Pascanu R, Rabinowitz N, et al. Overcoming catastrophic forgetting in neural networks [J]. Proceedings of the national academy of sciences, 2017, 114 (13): 3521-3526.​

[6] He K, Zhang X, Ren S, et al. Mask R-CNN [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 2961-2969.​

[7] 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. 2017: 2117-2125.​

[8] Redmon J, Farhadi A. YOLO9000: better, faster, stronger [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 6517-6525.​

[9] 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.​

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

[11] Bello I, Pham H, Le Q V, et al. Neural architecture search with reinforcement learning [J]. arXiv preprint arXiv:1611.01578, 2016.​

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

[13] 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.​

[14] Tan M, Le Q V. EfficientNet: rethinking model scaling for convolutional neural networks [J]. arXiv preprint arXiv:1905.11946, 2019.​

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

[16] Chen L C, Papandreou G, Kokkinos I, et al. DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs [J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 40 (4): 834-848.​

[17] Ronneberger O, Fischer P, Brox T. U-Net: convolutional networks for biomedical image segmentation [C]//International conference on medical image computing and computer-assisted intervention. Springer, Cham, 2015: 234-241.​

[18] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: transformers for image recognition at scale [J]. arXiv preprint arXiv:2010.11929, 2020.​

[19] 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.​

[20] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need [C]//Advances in neural information processing systems. 2017: 5998-6008.