

合肥工业大学

《机器视觉》课程实验

实验题目 课程实验四：校园共享单车检测

学生姓名 高志鸿

学 号 2023217584

专业班级 智能科 23-3 班

一、实验目的

本实验旨在完成“共享单车目标检测”的端到端实现与验证，能够在自然场景图片中自动定位并统计自行车（bicycle）目标。通过本实验掌握基于深度学习的一阶段目标检测流程：模型加载、阈值控制、类别筛选、检测框可视化以及批量推理输出。最终输出包括：检测结果叠加图（带框与置信度标签）与每张图的自行车数量统计，便于后续做共享单车投放监测、占道分析等应用扩展。

二、实验环境

- 操作系统：Windows
- 开发语言：Python 3
- 主要依赖：OpenCV (cv2)、ultralytics (YOLOv8)、NumPy
- 工程结构：`exp4/`
 - 推理脚本：`exp4/main.py`
 - 模型加载：`exp4/model.py`
 - 输入目录：`exp4/input/`（本实验共 3 张：`1.jpg`、`2.jpg`、`3.jpg`）
 - 输出目录：`exp4/output/`（输出：`result_1.jpg`、`result_2.jpg`、`result_3.jpg`）

三、实验原理与方法

1. 目标检测问题定义

目标检测需要同时输出“类别（是什么）”与“位置（在哪里）”。本实验的检测对象为共享单车，具体落到实现层面是从模型输出的多个候选框中筛选类别为 bicycle 的检测框，并在原图上绘制矩形框与置信度标签，最终达到“定位 + 计数”的目标。

2. YOLOv8 检测框架

本实验使用 ultralytics 提供的 YOLOv8 预训练模型 `yolov8n.pt`。YOLO 系列属于一阶段检测器，特点是将候选框生成与分类回归整合到一次前向推理中，推理速度快、部署方便，适合实时或批量图片处理场景。模型输出包含若干预测框 (bbox)、每个框的类别 ID (cls) 与置信度 (conf)，后处理会对重叠框进行抑制（常见为 NMS 思想），保留更可信的检测结果。

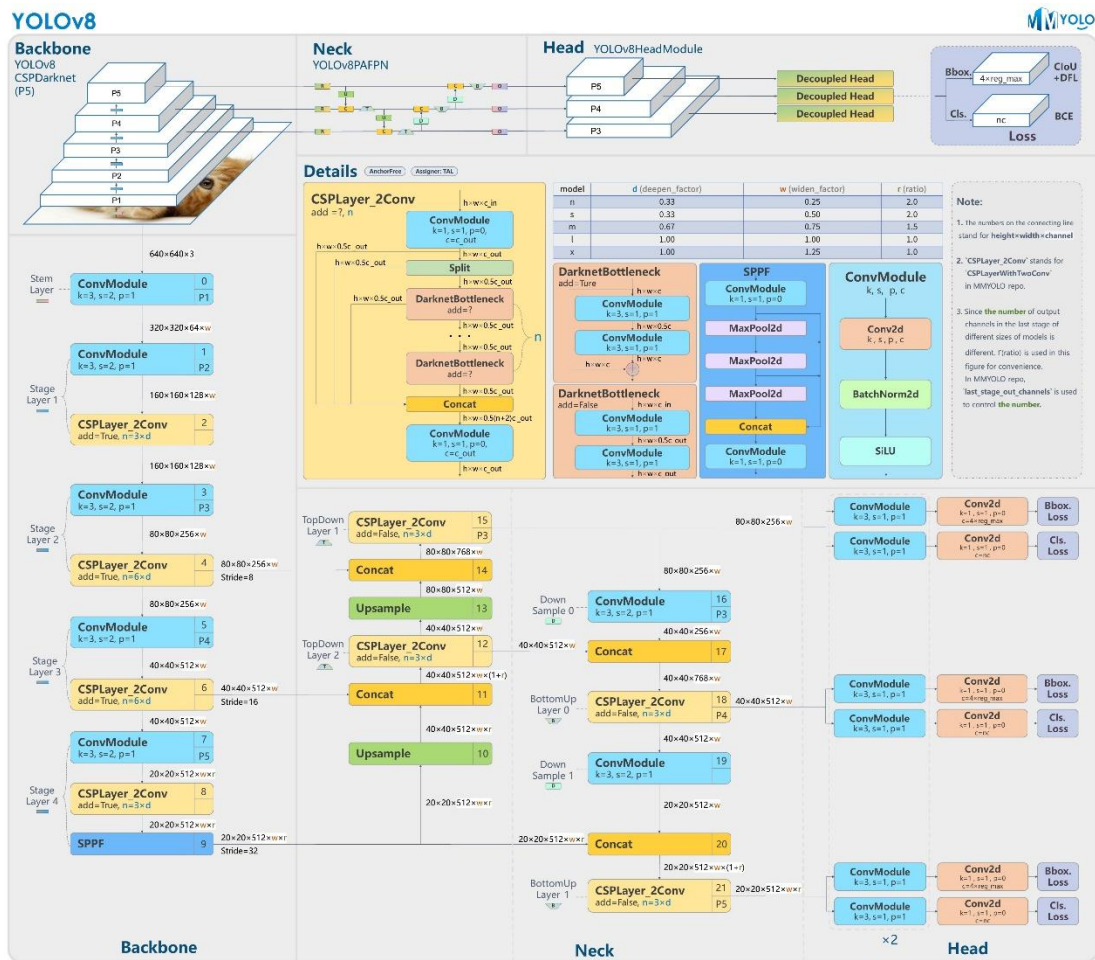


Figure 1 YOLOv8 网络框架图

3. 预训练数据与类别体系（COCO）

`yolov8n.pt` 默认在 COCO 数据集上训练，包含 80 类常见目标。实验中根据 COCO 的类别索引，bicycle 通常对应 `class_id = 1`。因此脚本在推理后对所有检测框进行类别筛选，仅保留 `cls_id == 1` 的框作为共享单车检测结果；其它类别（如 person，甚至是外形接近的摩托车、电动车）即使被模型检出，也不会计入单车数量。

4. 阈值策略与输出可解释性

脚本设置 `conf=0.25` 作为置信度阈值：只保留置信度不低于阈值的预测框，以在“漏检”和“误检”之间取得平衡。阈值偏低会让更多候选框通过，召回率可能更高但误检也更多；阈值偏高会使输出更干净，但可能漏掉远处或遮挡较多的单车。为了增强可解释性，脚本将每个单车框以绿色矩形绘制，并在框上方标注 `Bicycle {conf:.2f}`，便于直观看到模型对每个检测的信心程度。

四、实验内容与步骤

1. 模型加载与初始化

- 在 `exp4/main.py` 中调用 `load_model('yolov8n.pt')` 加载 YOLOv8 模型（`exp4/model.py` 内部使用 `YOLO(model_name)`）。当本地不存在时，ultralytics 会自动处理下载与缓存。
- 设置目标类别：`target_class_id = 1`（bicycle）。

2. 批量读取输入图片

- 脚本从 `exp4/input/` 遍历读取所有图片文件（支持 jpg/jpeg/png/bmp），并逐张推理。

3. 推理与检测框解析

- 执行 `results = model(img_path, conf=0.25)` 得到推理结果列表。
- 对每张图片的 `result.bboxes` 遍历检测框，取出：
 - `cls_id = int(box.cls[0])`
 - `conf = float(box.conf[0])`
 - `x1, y1, x2, y2 = map(int, box.xyxy[0])`
- 仅当 `cls_id == 1` 时判定为自行车检测，计数 +1，并绘制框与文字标签。

4. 结果保存

- 每张图输出保存为：`exp4/output/result_{filename}`
 - 示例：`result_1.jpg`、`result_2.jpg`、`result_3.jpg`
- 终端会打印每张图片检测到的自行车数量，形成可追溯的运行记录。

五、实验数据记录

- 输入数据：`exp4/input/1.jpg`、`2.jpg`、`3.jpg`（共 3 张）
- 运行参数：
 - 模型：`yolov8n.pt`（YOLOv8n）
 - 置信度阈值：`conf=0.25`
 - 目标类别：`bicycle (class_id=1)`
- 实际推理输出（控制台记录）：
 - `1.jpg`：检测到 5 辆自行车；推理耗时约 44.1ms（含预处理/后处理统计输出）
 - `2.jpg`：检测到 1 辆自行车；推理耗时约 50.3ms

- '3.jpg': 检测到 1 辆自行车；推理耗时约 39.7ms
- 输出文件（已生成）：
 - 'exp4/output/result_1.jpg'
 - 'exp4/output/result_2.jpg'
 - 'exp4/output/result_3.jpg'

六、实验结果与分析

1.实验结果

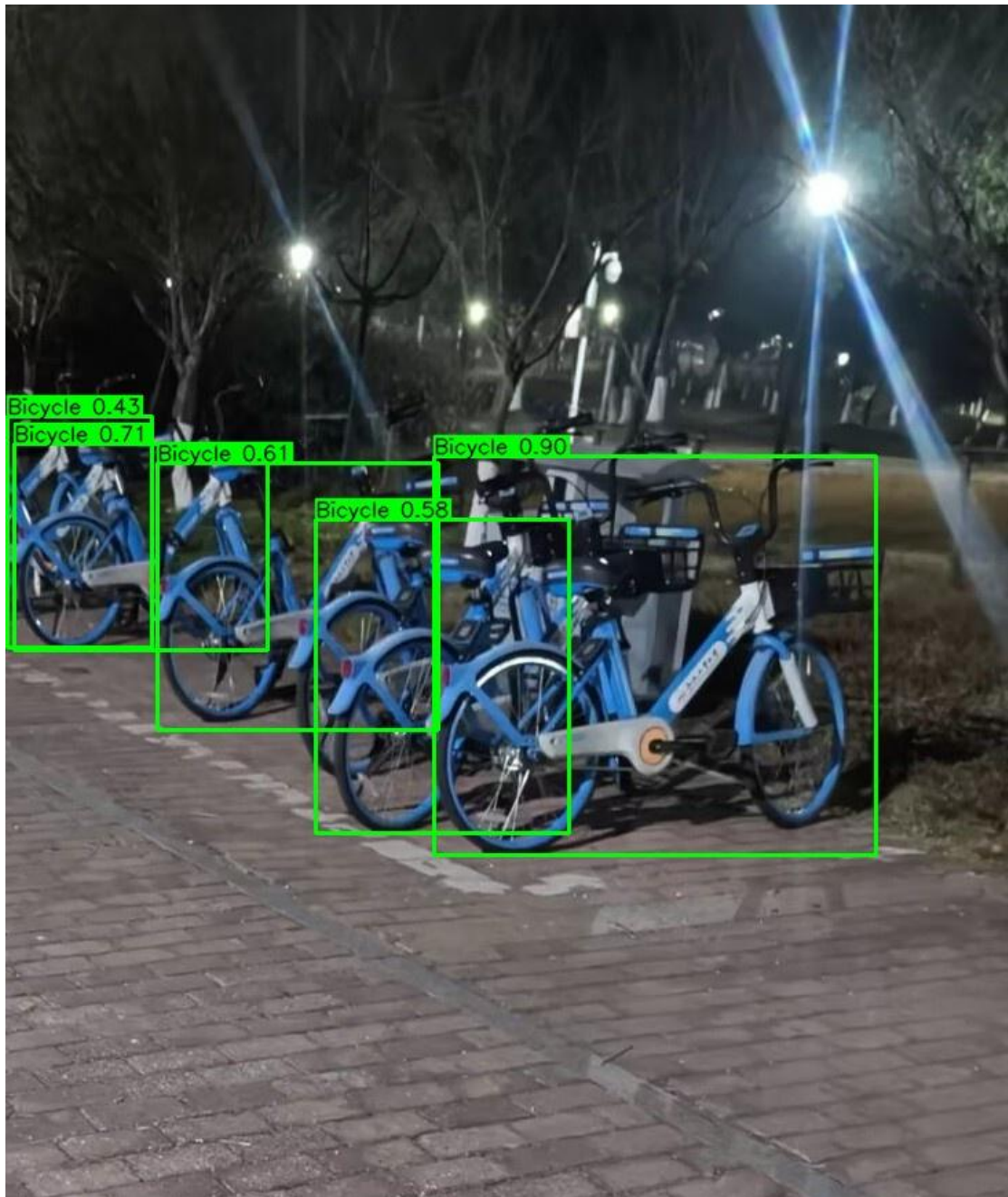


Figure 2 output_1

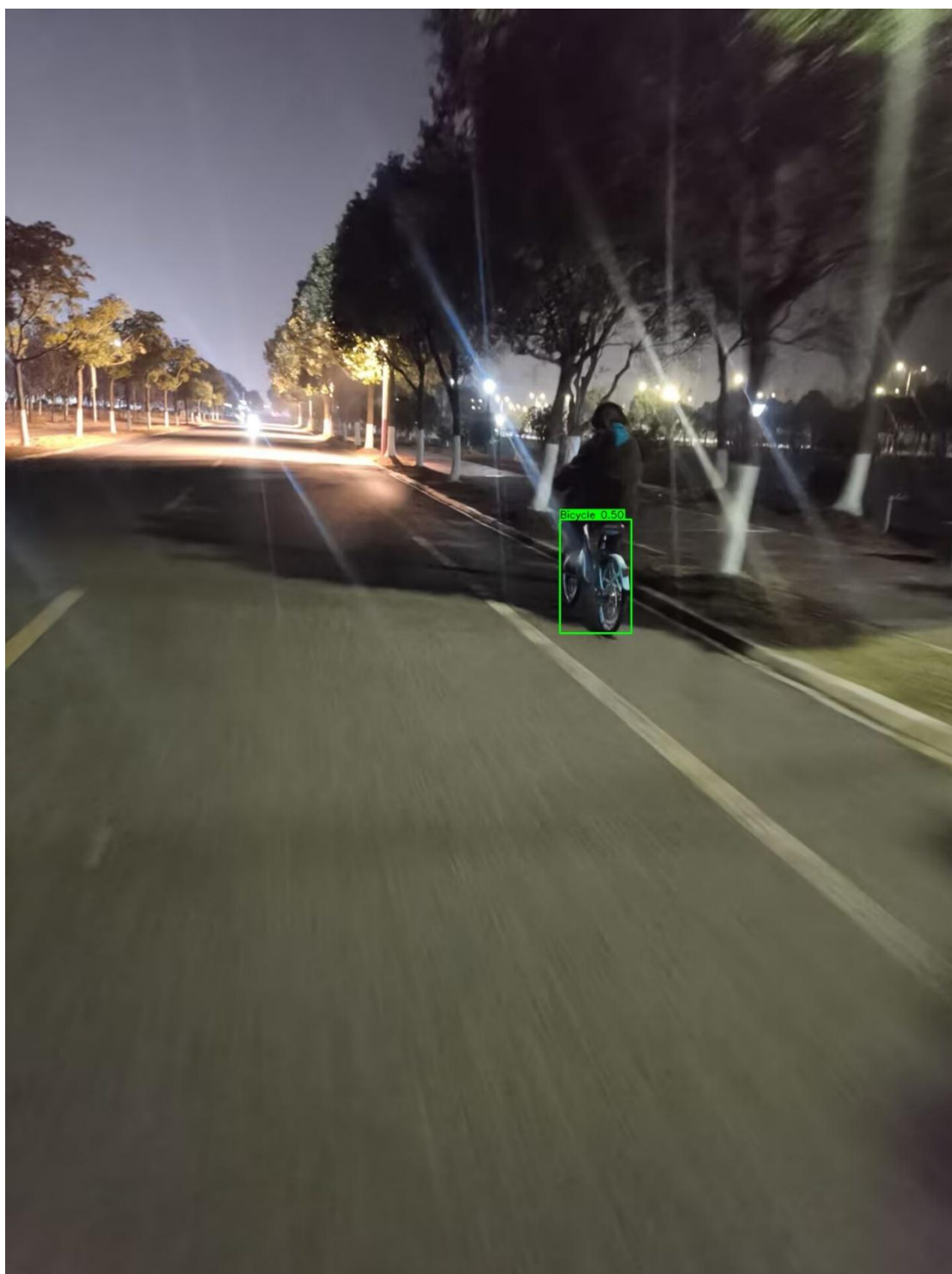


Figure 3 output_2

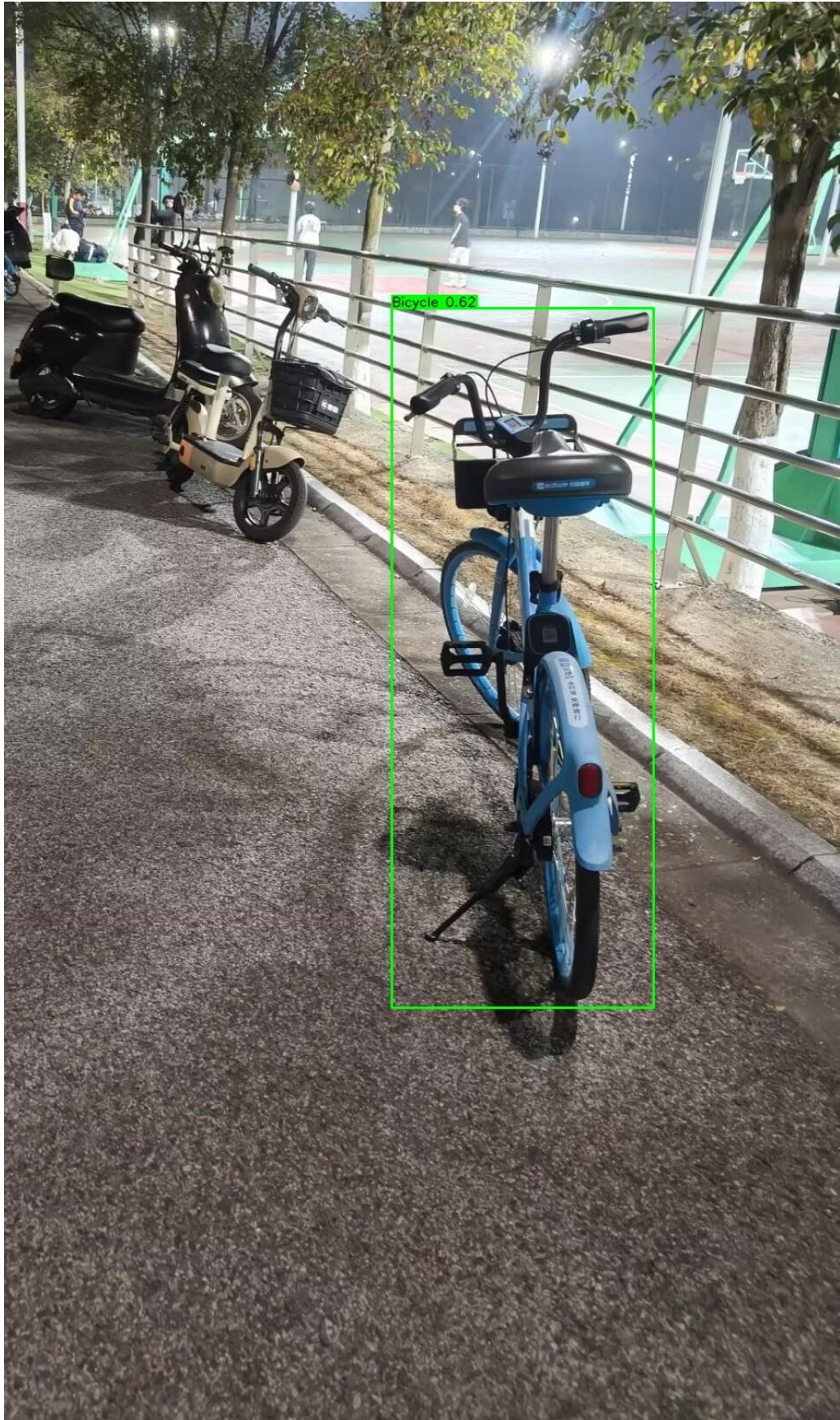


Figure 4 output_3

2.结果分析

从结果来看，模型能够在自然场景中稳定检出自行车目标，并完成定位与计数。对于单车数量较多的场景（如 `1.jpg`），输出能够同时给出多个目标框，说明模型具备多目标检测能力；对于包含人或其它交通工具的场景（如 `2.jpg`、`3.jpg`），虽然模型也检测到了 `person` 等类别，但脚本通过类别筛选只统计 `bicycle`，使结果更聚焦于“共享单车检测”这一任务目标。

阈值 `conf=0.25` 在本次样例上能兼顾一定召回与输出干净度：远处或被遮挡的单车若置信度偏低可能被过滤，但输出框的可信度整体更高，适合做展示与统计。与此同时，基于 COCO 预训练模型的一个现实点是：共享单车属于 `bicycle` 的子集，且共享单车外观、颜色、停放形态可能与 COCO 中的 `bicycle` 分布存在差异，因此在极端视角、强遮挡、夜间或密集堆叠情况下仍可能出现漏检或误检；这属于“数据分布差异”带来的典型现象。

七、结论

本实验基于 YOLOv8n 预训练模型实现了共享单车的自动检测流程，能够对输入目录中的图片进行批量推理，输出带检测框与置信度标签的可视化结果图，并统计每张图自行车数量。实验验证了 YOLOv8 在目标检测任务上的易用性与较好的工程落地能力：模型加载方便、推理速度快、输出结果可解释性强，能够满足共享单车场景中“快速定位与计数”的基本需求。

八、实验体会与个人理解

这次实验让我对“深度学习目标检测的工程闭环”有了更清晰的认识。相比传统方法依赖手工特征与规则阈值，YOLOv8 直接从数据中学习多尺度特征表达，能够在复杂背景下同时完成定位与分类。尤其是在共享单车这种真实场景任务中，光照变化、背景干扰、遮挡与视角变化都很常见，纯规则方法往往需要大量手工调参才能勉强适配，而预训练检测器可以在较少改动的情况下就给出可用结果，这种“即插即用”的能力非常适合快速原型与工程验证。

我也体会到阈值与类别筛选在落地应用中的关键作用。目标检测模型输出的是“多类别、多候选框”的结果，如果不做筛选与阈值控制，最终呈现会非常杂乱，统计也会失真。将任务目标明确为 `bicycle` 后，通过 `cls_id==1` 的过滤，系统的输出立刻变得可解释且一致；而 `conf` 阈值则体现了工程上对误检/漏检的取舍：做监管统计时可能更关注少误检，需要更高阈值；做巡检预警时可能更关注少漏检，需要适当降低阈值并在后处理阶段再做规则校验。这种“业务目标决定参数策略”的思维，是我在实验中收获非常大的一点。

此外，这个实验也让我更直观地理解了“数据分布”对模型效果的影响。

YOLOv8n 在 COCO 上训练，能识别 `bicycle`，但共享单车的堆叠停放、遮挡形态、车筐/广告牌等细节可能与 COCO 的样本不同，因此在某些场景下会出

现把单车识别成 **motorbike**（或反过来）的情况。要进一步提升准确率，最有效的方向并不是盲目换更大的模型，而是引入更贴近共享单车场景的数据进行微调，或在后处理阶段加入更细粒度的类别合并策略（例如同时考虑 **bicycle** 与 **motorbike**，再结合外观/尺寸约束做二次筛选）。这让我意识到：模型能力、数据质量与后处理策略三者共同决定了最终系统的可靠性。

最后，从可展示性角度看，本实验输出的 ``result_*.jpg`` 叠加了检测框与置信度标签，使结果非常适合用于答辩演示：不仅能说明“检测到了什么”，还能说明“模型对这个检测有多确定”。这种结果可视化与日志记录的习惯，也让我在做计算机视觉工程时更容易形成“可解释、可复现、可迭代”的工作方式。