YOLO代码复现框架

|  |  |
| --- | --- |
| 项 | 说明 |
| 名称 | YOLO代码复现框架 |
| 行业 | 人工智能 |
| 技术 | 1. 深度学习； 2. 图像处理； 3. PyTorch； 4. CoCo数据集与VOL数据集 |
| 其他 | 程序设计 |

# 一． 项目摘要

YOLO，全称“You Only Look Once”，是一种先进且流行的实时目标检测系统。它的核心思想与名称一致：只需“看”一次图像，就能预测出图像中所有目标的位置和类别。这与传统的目标检测方法（如基于区域提议的R-CNN系列）形成鲜明对比，后者需要先在图像中生成大量可能包含目标的区域（Region Proposals），然后再对这些区域进行分类和微调，因此速度较慢。

YOLO因其速度和精度的良好平衡，被广泛应用于：

1. 自动驾驶：车辆、行人、交通标志的实时检测。
2. 视频监控：入侵检测、人流统计、异常行为识别。
3. 机器人技术：机器人视觉导航和物体抓取。
4. 医疗影像：细胞检测、病灶区域定位。
5. 工业检测：产品缺陷检测、零件计数。
6. 遥感图像分析：建筑物、船舶、飞机等的检测。

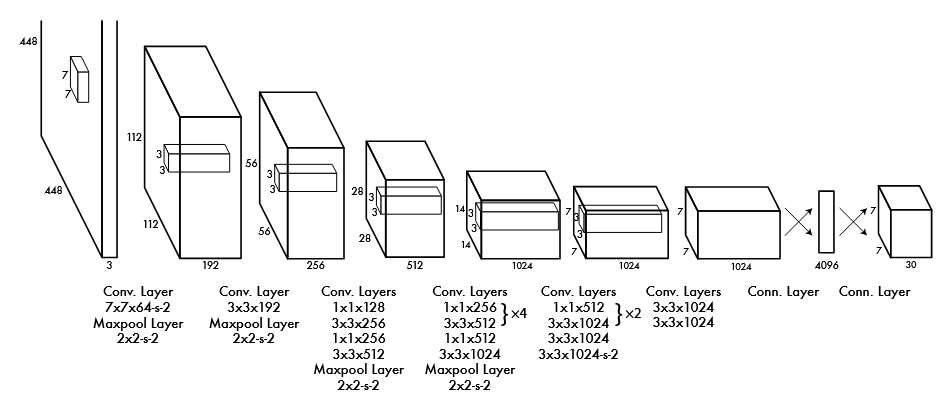
YOLO框架通过其革命性的“单次检测”思想，彻底改变了目标检测领域，使其从复杂的多步骤流程变成了一个高效、统一的端到端系统。尽管早期版本有不足之处，但通过持续的迭代和创新（如锚框、多尺度预测、无锚点设计等），YOLO系列始终在速度与精度之间保持着最佳平衡，成为实时目标检测领域事实上的标准和首选框架。

# 二． 项目需求

根据论文作者的原始《You Only Look Once Unified, Real-Time Object Detection》复现YOLO代码，并按照商业开源代码发布YOLO框架。作者论文地址：<https://arxiv.org/abs/1506.02640> 。

YOLO网络已经从v1发展到v13。在开发中，从v1复现代码，并根据v1到v13以及实际的应用场景，制定YOLO版本改进计划。

下面是YOLOv1的网络结构：



下面是YOLO的版本介绍：

|  |  |  |
| --- | --- | --- |
| YOLOv1（2015年） | arXiv论文 | You Only Look Once Unified, Real-Time Object Detection |
| https://arxiv.org/abs/1506.02640 |
| 代码 | https://pjreddie.com/darknet/yolov1/ |
| https://github.com/pjreddie/darknet |
| https://github.com/AlexeyAB/darknet |
| https://github.com/ProgrammerZhujinming/YOLO |
| YOLOv2（2016） | arXiv论文 | YOLO9000 Better, Faster, Stronger |
| https://arxiv.org/abs/1612.08242 |
| 代码 | https://pjreddie.com/darknet/yolov2/ |
| https://github.com/longcw/yolo2-pytorch |
| https://github.com/AlexeyAB/darknet |
| YOLOv3（2018） | arXiv论文 | YOLOv3 An Incremental Improvement |
| https://arxiv.org/abs/1804.02767 |
| 代码 | https://github.com/ultralytics/yolov3 |
| YOLOv4（2020） | arXiv论文 | YOLOv4 Optimal Speed and Accuracy of Object Detection |
| https://arxiv.org/abs/2004.10934 |
| 代码 | https://github.com/AlexeyAB/darknet |
| https://github.com/pjreddie/darknet |
| YOLOv5（2020） | arXiv论文 | 没有正式发表的论文（主要在于图像增强） |
| 代码 | https://github.com/ultralytics/yolov5 |
| YOLOv6（2022） | arXiv论文 | YOLOv6 A Single-Stage Object Detection Framework for Industrial Applications |
| https://arxiv.org/abs/2209.02976 |
| 代码 | https://github.com/meituan/yolov6 |
| YOLOv7（2022） | arXiv论文 | YOLOv7 Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors |
| https://arxiv.org/abs/2207.02696 |
| 代码 | https://github.com/wongkinyiu/yolov7 |
| https://github.com/pjreddie/darknet |
| https://github.com/AlexeyAB/darknet |
| YOLOv8（2024） | arXiv论文 | What is YOLOv8 An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector |
| https://arxiv.org/abs/2408.15857 |
| 代码 | https://github.com/DataXujing/YOLOv8 |
| https://github.com/ultralytics/ultralytics |
| YOLOv9（2024） | arXiv论文 | YOLOv9 Learning What You Want to Learn Using Programmable Gradient Information |
| https://arxiv.org/abs/2402.13616 |
| 代码 | https://github.com/WongKinYiu/YOLO |
| https://github.com/wongkinyiu/yolov9 |
| https://github.com/ultralytics/ultralytics |
| YOLOv10（2024） | arXiv论文 | YOLOv10 Real-Time End-to-End Object Detection |
| https://arxiv.org/abs/2405.14458 |
| 代码 | https://github.com/THU-MIG/yolov10 |
| YOLOv11（2024） | arXiv论文 | YOLOv11: An Overview of the Key Architectural Enhancements |
| https://arxiv.org/abs/2410.17725 |
| 代码 | https://github.com/ultralytics/ultralytics |
| YOLOv12（2025） | arXiv论文 | YOLOv12 Attention-Centric Real-Time Object Detectors |
| https://arxiv.org/abs/2502.12524 |
| 代码 | https://github.com/sunsmarterjie/yolov12 |
| YOLOv13（2025） | arXiv论文 | YOLOv13: Real-Time Object Detection with Hypergraph-Enhanced Adaptive Visual Perception |
| https://arxiv.org/abs/2506.17733 |
| 代码 | https://github.com/iMoonLab/yolov13 |

开发者在是实现YOLO框架的同时，并实现一个应用场景DEMO，比如交通目标检测、农业作物目标检测等。并提供同时期相关算法的模型分析。

# 项目功能

YOLO将整个目标检测任务重新定义为一个单一的回归问题，直接从图像像素映射到边界框坐标和类别概率。

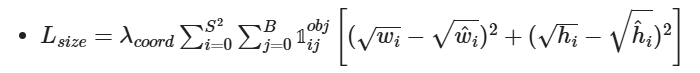
1. 划分网格：将输入图像划分为一个 S x S 的网格（例如，19x19）。
2. 负责机制：每个网格单元负责预测那些中心点落在该单元内的目标
3. 预测内容：每个网格单元会预测 B 个边界框（Bounding Boxes） 以及这些框的置信度（Confidence Score）。同时，它还会预测 C 个类别概率。
4. 边界框：包含4个值（中心点x, y，宽度w，高度h），都是相对于网格单元的比例值。
5. 置信度：表示该框包含一个目标的置信程度以及预测框的准确性（IoU）。计算公式为：Pr(Object) \* IoU(truth, pred)。如果网格中没有目标，置信度应为零。
6. 类别概率：Pr(Class\_i | Object)，表示当网格中存在目标时，该目标属于每个类别的条件概率。
7. 最终输出：将网格预测、边界框预测和类别预测组合在一起，形成一个巨大的张量（Tensor）。例如，在PASCAL VOC数据集上（20个类别），使用S=7, B=2，最终输出为 7 x 7 x (5\*2 + 20) = 7 x 7 x 30 的张量。

其中损失函数计算如下：

1. 正样本中心点损失



1. 正样本高宽度损失



1. 正样本置信度损失



1. 负样本置信度损失



1. 正样本分类类别损失



# 四．项目使用的技术与学习建议大纲

|  |  |
| --- | --- |
| 技术名称 | 推荐学习资料 |
| YOLO论文 | 《You Only Look Once Unified, Real-Time Object Detection》 |
| YOLO | 《YOLO目标检测》 |
| 机器视觉 | 《计算机视觉十讲》 |
| OpenCV | 《OpenCV计算机视觉编程攻略》 |
| GPU | 《GPU编程实战（基于Python和CUDA）》 |
| 深度学习 | 《深度学习（PyTorch版）》 |

# 五．附录

