

人工智能程序设计

python



```
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
    turtle.circle(40, 80/2)
    turtle.fd(40)
    turtle.circle(16, 180)
    turtle.fd(40 * 2/3)
```



人工智能程序设计

13.3 目标检测：YOLO

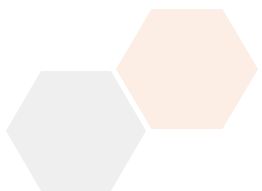
北京石油化工学院 人工智能研究院

刘 强

章节导入

目标检测是计算机视觉中最重要的任务之一

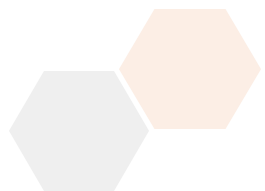
- 不仅要识别图像中的物体类别
- 还要精确定位物体的位置
- YOLO以其速度快、精度高的特点在工业界广泛应用



13.3.1 目标检测基础原理

学习内容:

- 目标检测任务定义
- 检测结果的表示方式
- 评价指标体系



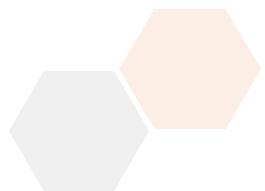
目标检测的核心任务

目标检测的核心任务是在给定图像中找到所有感兴趣的物体

- 为每个物体确定其类别和边界框位置
- 同时解决"是什么"和"在哪里"两个问题

与图像分类的区别:

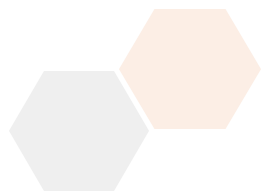
- 图像分类只需要给出整张图像的类别
- 目标检测需要定位每个物体的具体位置



检测结果的表示

检测结果通常包含：

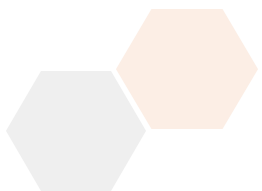
内容	说明
边界框	四个坐标值 (x, y, width, height)
类别标签	物体所属的类别
置信度分数	检测的可信程度



评价指标体系

主要评价指标：

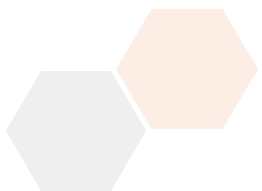
- **精确率 (Precision)**：正确检测的比例
- **召回率 (Recall)**：找到的目标比例
- **平均精度 (AP)**：单类别的综合指标
- **mAP**：所有类别AP的平均值，最常用的综合评价指标



13.3.2 YOLO算法详解

学习内容:

- YOLO的核心思想
- 网格划分策略
- 网络架构设计

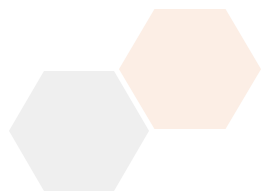


YOLO的核心思想

统一检测框架是YOLO的最大创新

- 将目标检测问题转化为单一的回归问题
- 传统两阶段方法需要先生成候选区域再分类
- YOLO直接从整张图像预测边界框和类别概率

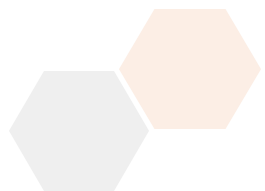
YOLO = You Only Look Once



网格划分策略

网格划分策略将输入图像划分为 $S \times S$ 的网格

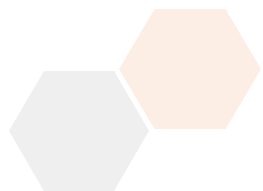
- 每个网格单元负责检测中心点落在该单元内的物体
- 简化了检测流程
- 使得算法能够进行端到端的训练



边界框预测机制

每个网格单元预测B个边界框，每个边界框包含5个值：

参数	说明
x, y	中心坐标
w, h	宽度和高度
confidence	置信度（包含物体的概率 × 边界框准确性）



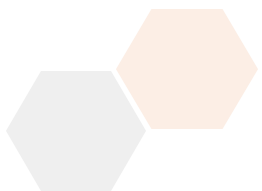
YOLO网络架构

骨干网络结构:

- 采用类似GoogLeNet的卷积神经网络
- 包含24个卷积层和2个全连接层
- 使用 1×1 卷积减少参数量

输出层设计:

- 输出 $7 \times 7 \times 30$ 的特征图
- 包含整张图像的语义信息和空间信息

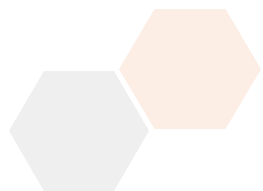


YOLO核心预测结构

YOLO最终预测层的设计:

```
## YOLO核心预测结构
## 最终预测层: 输入特征 -> 网格预测
self.predictor = nn.Linear(4096, 7 * 7 * (5 * 2 + 20)) # 7x7网格, 2个框, 20类

## 预测结果重塑: [batch, features] -> [batch, 7, 7, 30]
predictions = predictions.view(batch_size, 7, 7, -1)
```

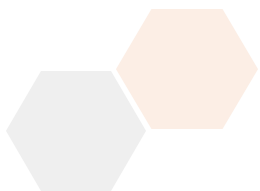


预训练模型选择

YOLO系列版本提供了不同规模的预训练模型：

COCO数据集：预训练模型能检测80个常见类别

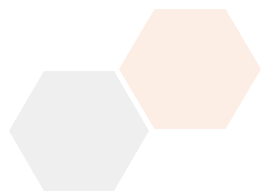
版本	特点
YOLOv5n	轻量级，速度最快
YOLOv5s	平衡版本
YOLOv5m	中等规模
YOLOv5l/x	高精度，计算量大



13.3.3 YOLO目标检测项目实施

学习内容:

- 智能监控系统
- 工业质量检测
- 实时人员和车辆检测

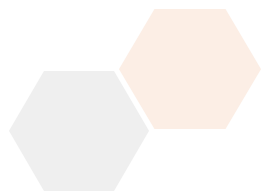


应用场景概述

智能监控系统：实时人员和车辆检测，识别异常行为

自动驾驶辅助：检测道路上的车辆、行人、交通标志

工业质量检测：检测产品缺陷、零件缺失等质量问题



工业质量检测实现

生产线质量控制通过YOLO实现对产品的自动化质量检测：

```
## 工业质量检测核心流程
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')
cap = cv2.VideoCapture(camera_id)

while True:
    ret, frame = cap.read()
    if not ret: break

    # YOLO检测
    results = model(frame)
    detections = results.pandas().xyxy[0]

    # 检测产品和缺陷
    products = detections[detections['name'] == 'product']
    defects = detections[detections['name'].isin(['scratch', 'dent', 'missing_part'])]

    if len(defects) > 0:
        print(f"质量警报：检测到 {len(defects)} 个缺陷")
```

实时人员和车辆检测

流量统计应用：针对人员和车辆进行实时检测和计数

```
## 实时人员和车辆检测核心流程
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')
target_classes = ['person', 'car', 'truck', 'bus', 'motorcycle', 'bicycle']

while True:
    ret, frame = cap.read()
    if not ret: break

    # YOLO检测
    results = model(frame)
    detections = results.pandas().xyxy[0]

    # 筛选和统计目标
    targets = detections[detections['name'].isin(target_classes)]
    person_count = len(targets[targets['name'] == 'person'])
    vehicle_count = len(targets[targets['name'] != 'person'])

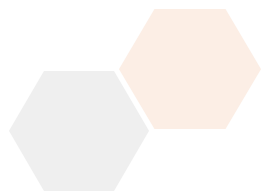
    print(f"People: {person_count}, Vehicles: {vehicle_count}")
```

实践练习

练习 13.3.1: Ask AI搭建YOLO应用环境

1. 使用Ask AI帮助你搭建YOLO目标检测应用环境
2. 安装ultralytics和相关依赖库
3. 下载并测试预训练模型
4. 验证GPU加速功能

提示：向AI描述："请帮我搭建YOLO目标检测的应用环境，包括ultralytics库的安装、预训练模型的下载，以及基本的检测功能测试。"

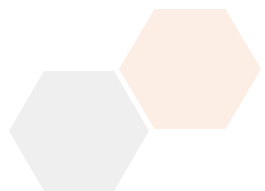


实践练习

练习 13.3.2: Ask AI实现智能监控系统

1. 使用Ask AI帮助你实现基于YOLO的智能监控系统
2. 实现实时视频流检测
3. 添加特定目标的报警功能
4. 支持检测结果的记录和统计

提示：向AI请求："基于YOLO预训练模型，请帮我实现一个智能监控系统，能够检测指定类别的目标并触发报警。"



实践练习

练习 13.3.3: Ask AI开发行业应用

1. 选择一个感兴趣的应用场景（如交通、零售、工业等）
2. 设计针对性的检测方案
3. 实现原型系统并分析效果

提示：询问AI："如何将YOLO应用到[具体行业]中？请提供完整的解决方案。"

