利用YOLO进行目标检测

利用已经训练好的权重参数

第一步 环境配置

创建一个新的虚拟环境，安装pytorch>=1.8;python>=3.8,预装ultralytics;

第二步 参数下载 / 数据集下载 / 标注数据集

前往yolov8官网下载已经训练好的权重，本次实验，分别基于预训练权重、yolov8s、yolov8x6三种权重进行和展开；

数据集下载：roboflow上有许多开源的数据集

标注数据集：自己利用labelimg或者roboflow进行数据集的标注

模型使用

调用yolo模型并加载参数

```
import cv2
from ultralytics import YOLO
model=YOLO("yolov8s.pt") #这里就是下载官网训练的参数，将自己下载好的参数保存至同样的目录下，并且更改名称即可
```

注意，参数需要保存至根目录下才可以直接这么调用

视频读取及输出路径的确定

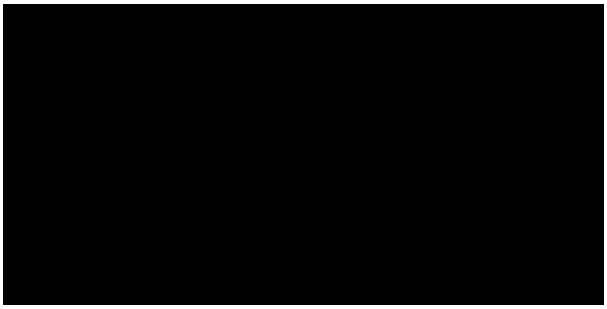
```
video_path="D:\\大三上学期\\计算机视觉\\第九周\\第八次实验\\football.mp4"
cap=cv2.VideoCapture(video_path)

### 输出

width=int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))#宽
height=int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))#高
fps=cap.get(cv2.CAP_PROP_FPS)#视频帧率
fourcc=cv2.VideoWriter_fourcc(*'mp4v')#视频格式
out=cv2.VideoWriter("C:\\Users\\86157\\Desktop\\output\\yolov8n.avi",fourcc,fps,(width,height))
```

对视频逐帧进行目标检测并且输出保存

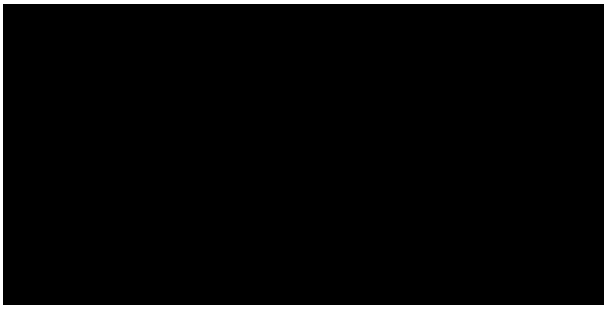
```
while cap.isOpened():
success,frame=cap.read()
if success:
    results=model(frame)
    annotated_frame=results[0].plot()
    cv2.imshow("YOLOv8 inference",annotated_frame)
    out.write(annotated_frame)

    if cv2.waitKey(1) & 0xFF==ord("q"):
        break
else:
    break
cap.release()
out.release()
cv2.destroyAllWindows()
```

结果展示及分析

YOLOv8x6:



YOLOv8n:



YOLOv8s:

性能评估:

在实验过程中，采取不同的训练权重时，可以观察到，如果想要目标检测的效果好，就必须采取更为庞大且细致的参数，并且在实际的对视频进行处理的过程中：处理效果好的视频帧率往往越高，处理速度越慢，处理精度越低的模型帧率往往越低，处理速度越快。

本次实验中：

速度：YOLOv8n>YOLOv8s>YOLOv8x6

精度：YOLOv8n<YOLOv8s<YOLOv8x6

平均帧率：

YOLOv8n: 62.5

YOLOv8s: 40

YOLOv8x6: 2.5

可以看出，为了提高精度，模型必须降低处理每一帧的速度，从而得到更为精确的结果

模型性能优化

在YOLOv8x6的训练过程中，对足球的检测有时候会检测成排球； 原因是因为视频质量的问题，以及可能检测物品的多样性； 如果我们采用自己训练的数据集并且仅仅采用两类，就可以分出足够好的结果，保证不进行误判

下面对自定义数据集训练YOLOv8进行尝试

利用自定义数据集进行训练

数据集下载

> 本次训练的数据集在roboflow上下载开源数据集，大小为1个G;

利用roboflow标注数据集或者下载数据的好处在于： 可以直接输出为yolov8需要的数据集（txt）格式，而不需要自己进行转化

训练

环境配置

同上，数据集较大，在云服务器上进行训练操作；

训练过程

```
1.yaml文件修改：

train: /hy-tmp/ultralytics-main/datasets/football/train/images
val: /hy-tmp/ultralytics-main/datasets/football/test/images
test: /hy-tmp/ultralytics-main/datasets/football/valid/images

nc: 2
names: ['football', 'player']
```

为防止报错，这里的路径应设置为绝对路径

```
2.训练命令

 yolo task=detect mode=train model=yolov8n.pt data=/hy-tmp/ultralytics-
main/datasets/football/data.yaml epochs=20
```

这里的数据集路径也设置为绝对路径

```
3.导出参数
yolo task=detect mode=export model=runs/detect/train3/weights/best.pt format=onnx


4.验证和测试
只需要更改模式，参数来源为上述默认路径
#验证
yolo task=detect mode=val model=runs/detect/train3/weights/best.pt
data=data/fall.yaml device=0
#测试
yolo task=detect mode=predict model=runs/detect/train3/weights/best.pt
source=data/images device=0
```

将保存的参数用于预测

具体过程同上述下载参数

结果分析

1.训练轮数为5轮时，预测效果较差，仅能检测到人，无法检测到足球。

2.训练轮数为20轮时，预测效果还是较差，考虑是初始权重问题，更改初始权重进行训练

3.yolov8s(训练轮数：20轮):