一、配置项目和依赖（Yolov5）

环境配置：

- Python: 3.8.2
- PyTorch: 1.13.1+cpu
- Ultralytics: 8.0.203
- OpenCV: 4.8.1.78
- YOLOv5版本: v7.0 (commit 915bbf2)

1.下载python3.8

```
C:\Users\zheng>python --version
Python 3.8.2

C:\Users\zheng>where python
C:\Users\zheng\AppData\Local\Programs\Python\Python38\python.exe
```

2.创建纯净虚拟环境

```
python -m venv yolov5_finall
call yolov5_final\Scripts\activate

Linux
source yolov5_finallll/bin/activate
source yolov5/bin/activate
```

```
C:\Users\zheng>python -m venv yolov5_final

C:\Users\zheng>call yolov5_final\Scripts\activate
```

```
zxh@zxh-VMware-Virtual-Platform:~/桌面$ python -m venv yolov5_final
zxh@zxh-VMware-Virtual-Platform:~/桌面$ source yolov5_final/bin/activate
(yolov5_final) zxh@zxh-VMware-Virtual-Platform:~/桌面$
```

3.升级pip并配置国内镜像

pip版本过低时，下载依赖会报错，所以可以先升级pip

```
python -m pip install --upgrade pip
pip config set global.trusted-host pypi.tuna.tsinghua.edu.cn
pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple

linux
python -m pip install --upgrade pip
```

```
(yolov5_final) C:\Users\zheng>python -m pip install --upgrade pip
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting pip
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/c9/bc/b7db44f5f39f9d0494071bddae6880eb645970
366d0a200022a1a93d57f5/pip-25.0.1-py3-none-any.whl (1.8MB)
     |                                        | 1.8MB 1.3MB/s
Installing collected packages: pip
  Found existing installation: pip 19.2.3
    Uninstalling pip-19.2.3:
      Successfully uninstalled pip-19.2.3
Successfully installed pip-25.0.1

(yolov5_final) C:\Users\zheng>pip config set global.trusted-host pypi.tuna.tsinghua.edu.cn
Writing to C:\Users\zheng\AppData\Roaming\pip\pip.ini

(yolov5_final) C:\Users\zheng>pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simpl
e
Writing to C:\Users\zheng\AppData\Roaming\pip\pip.ini
```

```
(yolov5_final) zxh@zxh-VMware-Virtual-Platform:~/桌面$ python -m pip install --u
pgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/c9/bc/b7db44f5f39f9d049407
1bddae6880eb645970366d0a200022a1a93d57f5/pip-25.0.1-py3-none-any.whl (1.8MB)
     |                                        | 1.8MB 4.6MB/s
```

4.安装PyTorch

```
pip install torch==1.13.1+cpu torchvision==0.14.1+cpu -f
https://download.pytorch.org/whl/torch_stable.html

linux
python3 -m pip install torch==1.13.1+cpu torchvision==0.14.1+cpu -f
https://download.pytorch.org/whl/torch_stable.html

python3 -m pip install torch==1.13.1+cpu torchvision==0.14.1+cpu -f
https://download.pytorch.org/whl/torch_stable.html
```

```
(yolov5_final) C:\Users\zheng>pip install torch==1.13.1+cpu torchvision==0.14.1+cpu -f https://downlo
ad.pytorch.org/whl/torch_stable.html
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Looking in links: https://download.pytorch.org/whl/torch_stable.html
Collecting torch==1.13.1+cpu
  Using cached https://download.pytorch.org/whl/cpu/torch-1.13.1%2Bcpu-cp38-cp38-win_amd64.whl (164.2
 MB)
Collecting torchvision==0.14.1+cpu
  Using cached https://download.pytorch.org/whl/cpu/torchvision-0.14.1%2Bcpu-cp38-cp38-win_amd64.whl
(1.1 MB)
Collecting typing-extensions (from torch==1.13.1+cpu)
```

```
(yolov5_final) zxh@zxh-VMware-Virtual-Platform:~/桌面$ pip install torch==1.13.1
+cpu torchvision==0.14.1+cpu -f https://download.pytorch.org/whl/torch_stable.ht
ml
Looking in links: https://download.pytorch.org/whl/torch_stable.html
Collecting torch==1.13.1+cpu
  Downloading https://download.pytorch.org/whl/cpu/torch-1.13.1%2Bcpu-cp38-cp38-
linux_x86_64.whl (199.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 199.1/199.1 MB 31.6 MB/s eta 0:00:00
```

5.安装核心依赖（锁定版本）

```
pip install numpy==1.23.5 opencv-python==4.8.1.78 tqdm==4.66.1

linux同
```

```
(yolov5_final) C:\Users\zheng>pip install numpy==1.23.5 opencv-python==4.8.1.78 tqdm==4.66.1
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting numpy==1.23.5
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/4c/42/6274f92514fbefcb1caa66d56d82ac7ac89f7
652c0cef1e159a4b79e09f1/numpy-1.23.5-cp38-cp38-win_amd64.whl (14.7 MB)
Collecting opencv-python==4.8.1.78
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/38/d2/3e8c13ffc37ca5ebc6f382b242b44acb43eb4
89042e1728407ac3904e72f/opencv_python-4.8.1.78-cp37-abi3-win_amd64.whl (38.1 MB)
Collecting tqdm==4.66.1
```

6.克隆代码（使用GitHub官方源）

```
git clone https://github.com/ultralytics/yolov5 -b v7.0 --depth 1
cd yolov5
git checkout v7.0 -f

Linux
git clone https://github.com/ultralytics/yolov5 -b v7.0 --depth 1
cd yolov5
git checkout v7.0 -f
```

```
(yolov5_final) C:\Users\zheng>git clone https://github.com/ultralytics/yolov5 -b v7.0 --depth 1
Cloning into 'yolov5'...
remote: Enumerating objects: 164, done.
remote: Counting objects: 100% (164/164), done.
remote: Compressing objects: 100% (140/140), done.
remote: Total 164 (delta 27), reused 143 (delta 23), pack-reused 0 (from 0)
Receiving objects: 100% (164/164), 968.75 KiB | 2.63 MiB/s, done.
Resolving deltas: 100% (27/27), done.
Note: switching to '915bbf294bb74c859f0b41f1c23bc395014ea679'.
```

7.安装项目依赖

```
pip install -r requirements.txt
pip install ultralytics==8.0.203  # 关键兼容版本

pip install -r requirements.txt
pip install ultralytics==8.0.203
```

```
(yolov5_final) C:\Users\zheng\yolov5>pip install -r requirements.txt
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting gitpython (from -r requirements.txt (line 5))
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/1d/9a/4114a9057db2f1462d5c8f8390ab7383925fe
1ac012eaa42402ad65c2963/GitPython-3.1.44-py3-none-any.whl (207 kB)
Collecting ipython (from -r requirements.txt (line 6))
```

```
(yolov5_final) C:\Users\zheng\yolov5>pip install ultralytics==8.0.203
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting ultralytics==8.0.203
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/da/96/70962de8b746747bea2a64988851622900c89
c8955eaf06ce546e7211942/ultralytics-8.0.203-py3-none-any.whl (644 kB)
Requirement already satisfied: matplotlib>=3.3.0 in c:\users\zheng\yolov5_final\lib\site-packages (fr
om ultralytics==8.0.203) (3.7.5)
Requirement already satisfied: numpy>=1.22.2 in c:\users\zheng\yolov5_final\lib\site-packages (from u
ltralytics==8.0.203) (1.23.5)
```

```
(yolov5_final) zxh@zxh-VMware-Virtual-Platform:~/桌面/yolov5$ pip install -r req
uirements.txt
Collecting gitpython (from -r requirements.txt (line 5))
  Downloading GitPython-3.1.44-py3-none-any.whl.metadata (13 kB)
Collecting ipython (from -r requirements.txt (line 6))
  Downloading ipython-8.12.3-py3-none-any.whl.metadata (5.7 kB)
Collecting matplotlib>=3.2.2 (from -r requirements.txt (line 7))
  Downloading matplotlib-3.7.5-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86
_64.whl.metadata (5.7 kB)
```

```
(yolov5_final) zxh@zxh-VMware-Virtual-Platform:~/桌面/yolov5$ pip install ultral
ytics==8.0.203
Collecting ultralytics==8.0.203
  Downloading ultralytics-8.0.203-py3-none-any.whl.metadata (31 kB)
Requirement already satisfied: matplotlib>=3.3.0 in /home/zxh/桌面/yolov5_final/
lib/python3.8/site-packages (from ultralytics==8.0.203) (3.7.5)
Requirement already satisfied: numpy>=1.22.2 in /home/zxh/桌面/yolov5_final/lib/
python3.8/site-packages (from ultralytics==8.0.203) (1.23.5)
Requirement already satisfied: opencv-python>=4.6.0 in /home/zxh/桌面/yolov5_fin
al/lib/python3.8/site-packages (from ultralytics==8.0.203) (4.8.1.78)
```

8.运行测试

（这时直接运行 **YOLOv5 会自动下载** `yolov5s.pt` **文件**）

```
python detect.py --source data/images/bus.jpg --weights yolov5s.pt --device
cpu

python detect.py --source data/images/bus.jpg --weights yolov5s.pt --device
cpu
```

```
(yolov5_final) C:\Users\zheng\yolov5>python detect.py --source data/images/bus.jpg --weights yolov5s.
pt --device cpu
detect: weights=['yolov5s.pt'], source=data/images/bus.jpg, data=data\coco128.yaml, imgsz=[640, 640],
 conf_thres=0.25, iou_thres=0.45, max_det=1000, device=cpu, view_img=False, save_txt=False, save_conf
=False, save_crop=False, nosave=False, classes=None, agnostic_nms=False, augment=False, visualize=Fal
se, update=False, project=runs\detect, name=exp, exist_ok=False, line_thickness=3, hide_labels=False,
 hide_conf=False, half=False, dnn=False, vid_stride=1
YOLOv5  v7.0-0-g915bbf2 Python-3.8.2 torch-1.13.1+cpu CPU

Downloading https://github.com/ultralytics/yolov5/releases/download/v7.0/yolov5s.pt to yolov5s.pt...
100%|                                                                                | 14.1M/14.1M [00:00<00:00, 36.5MB/s]

Fusing layers...
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients
image 1/1 C:\Users\zheng\yolov5\data\images\bus.jpg: 640x480 4 persons, 1 bus, 174.3ms
Speed: 1.0ms pre-process, 174.3ms inference, 2.5ms NMS per image at shape (1, 3, 640, 640)
Results saved to runs\detect\exp
```

其他

```
尝试其他预训练模型：
python detect.py --weights yolov5m.pt
使用自己的图片：
python detect.py --source path/to/your/image.jpg
训练自定义模型：
python train.py --data custom.yaml --weights yolov5s.pt
退出虚拟环境：
deactivate
进入：
call C:\Users\zheng\yolov5_final\Scripts\activate
```

.pt转换成onnx

```
(yolov5_final) C:\Users\zheng\yolov5>python export.py --weights
runs/train/exp/weights/best.pt --include onnx --opset 12 --simplify
```

二、准备数据集和训练

1.安装使用labelimg

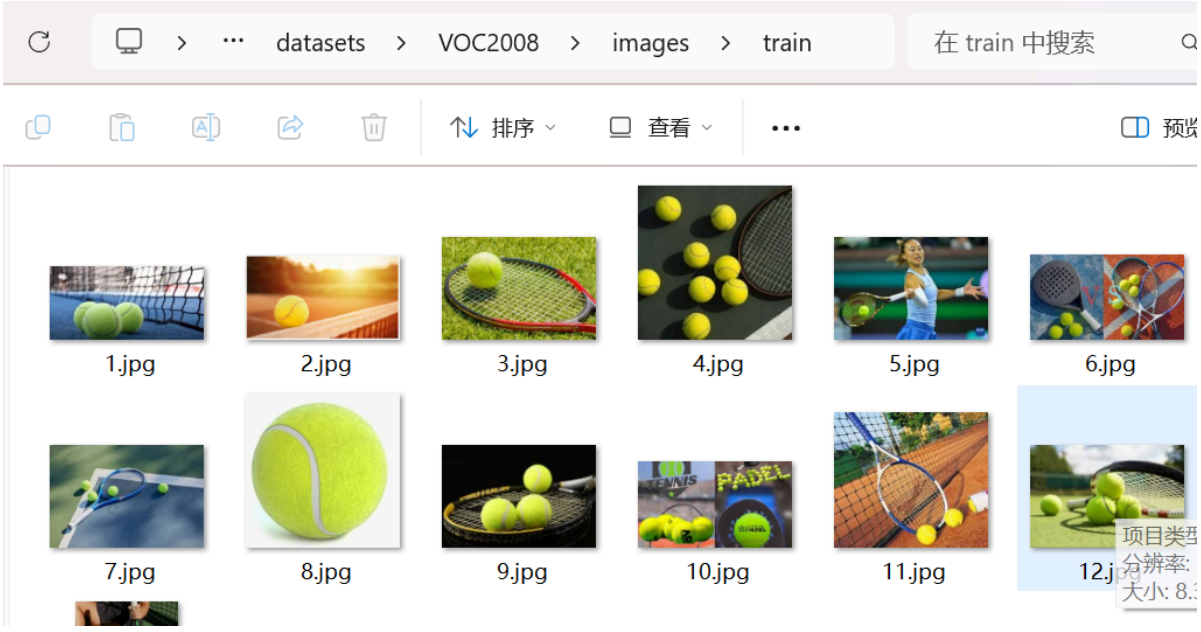
```
pip install labelimg -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
C:\Users\zheng\yolov5>pip install labelimg -i https://pypi.tuna.tsinghua.edu.cn/simple
WARNING: pip is being invoked by an old script wrapper. This will fail in a future version of pip.
Please see https://github.com/pypa/pip/issues/5599 for advice on fixing the underlying issue.
To avoid this problem you can invoke Python with '-m pip' instead of running pip directly.
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting labelimg
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/c5/fb/9947097363fbbfde3921f7cf7ce9800c89f909d26a506145aec37c75cda7/labelImg-1.8.6.tar.gz (247 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Collecting pyqt5 (from labelimg)
  Using cached https://pypi.tuna.tsinghua.edu.cn/packages/56/d5/68eb9f3d19ce65df01b6c7b7a577ad3bbc9ab3a5dd3491a4756e71838ec9/PyQt5-5.15.11-cp38-abi3-win_amd64.whl (6.9 MB)
Collecting lxml (from labelimg)
```

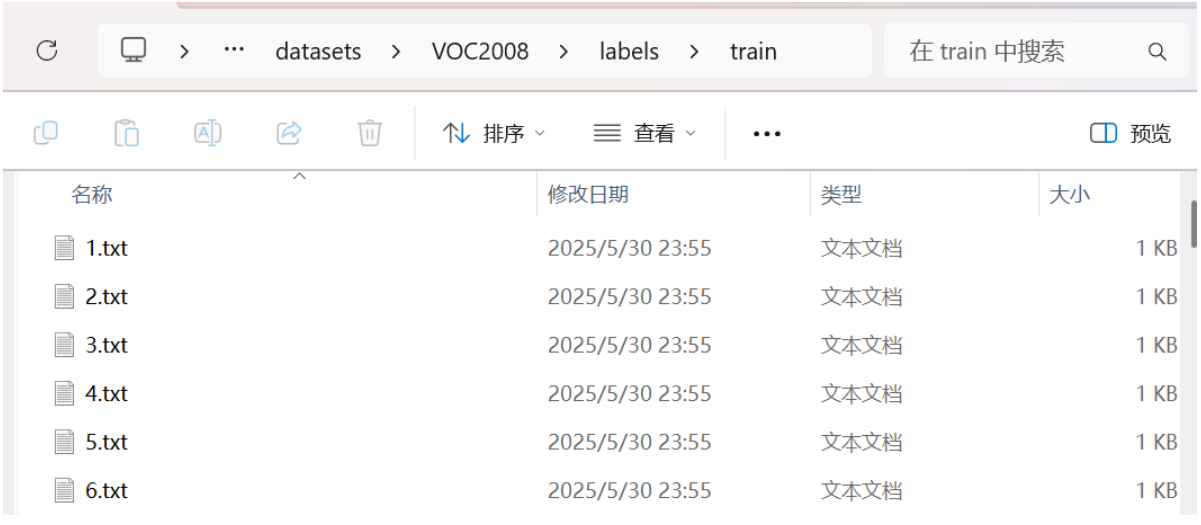
安装好后，输入labelimg，就打开了labelimg

对图片进行标注，使用yolo格式，为了能够自动对上，目录结构应该这样：

```
yolov5_project/
├── datasets/
|   ├── images/
|   |   ├── train/      # 训练集图片
|   |   |   ├── image1.jpg
|   |   |   └── ...
|   |   └── val/        # 验证集图片
|   |       ├── image101.jpg
|   |       └── ...
|   └── labels/
|       ├── train/      # 训练集标签
|       |   ├── image1.txt
|       |   └── ...
|       └── val/        # 验证集标签
|           ├── image101.txt
|           └── ...
└── data.yaml           # 数据集配置文件
```



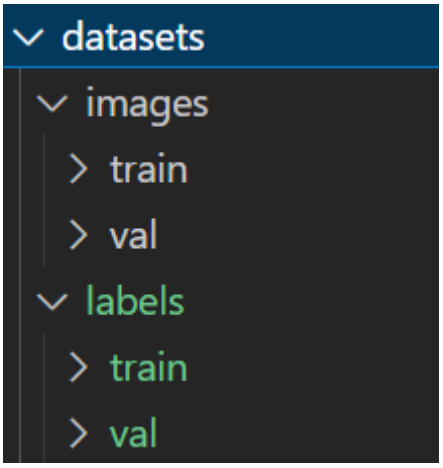
标注后得到对应的txt文件



每个图片对应一个 `.txt` 文件，格式为：

> <x_center> <y_center>
>
> 1. 对象的类别索引（整数），从0开始
> 2. <x_center>: 边界框中心点的x坐标，归一化到0~1（即相对于图像宽度的比例）
> 3. <y_center>: 边界框中心点的y坐标，归一化到0~1（即相对于图像高度的比例）
> 4. : 边界框的宽度，归一化到0~1（即相对于图像宽度的比例）
> 5. : 边界框的高度，归一化到0~1（即相对于图像高度的比例）
>
> 例如：0 0.452 0.312 0.125 0.178
>
> 1 0.781 0.534 0.088 0.102

把这个数据集datasets文件夹放到项目中



2.配置文件

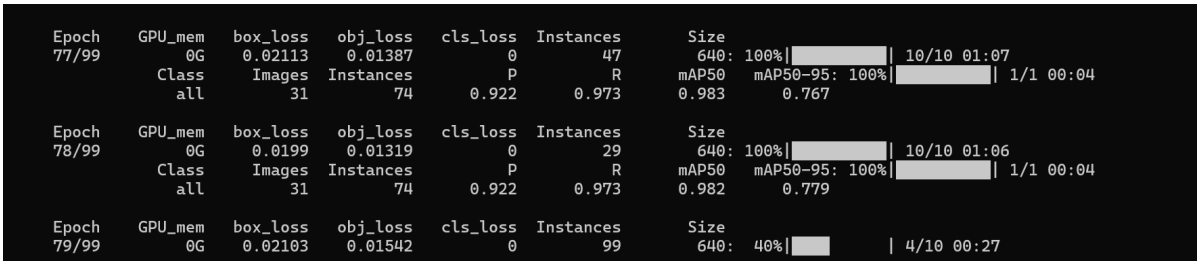
在 `yolov5` 目录下创建 `data/custom.yaml`（文件名可自定），内容如下：

```yaml
# 数据集路径（注意使用正斜杠）
path: C:/Users/zheng/yolov5/datasets
train: images/train  # 训练集相对路径
val: images/val      # 验证集相对路径

# 类别数（根据实际修改）
nc: 1

# 类别名称（示例，按实际修改）
names:
  0: wangqui
```

3.训练

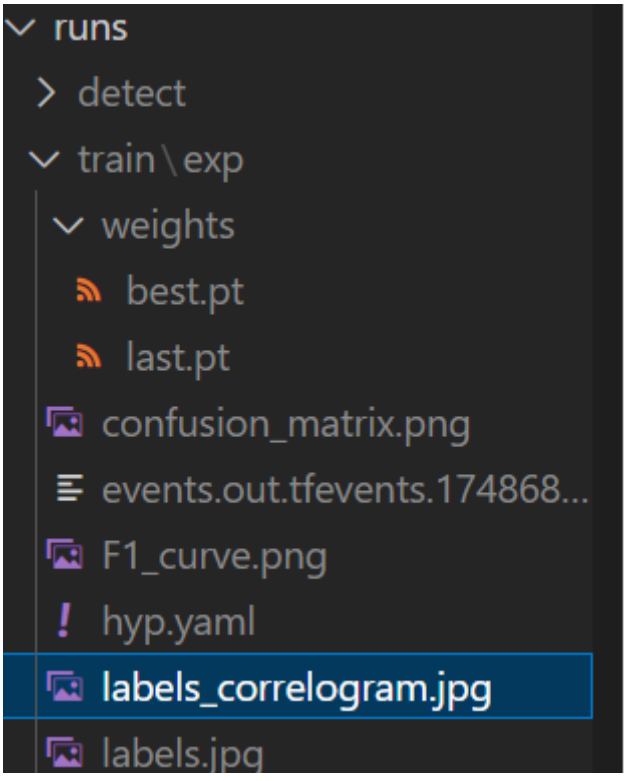
```
Windows 版命令

python train.py --data data/custom.yaml --weights yolov5s.pt --img 640 --batch 16 --epochs 100 --device cpu

Linux命令

python train.py \
```

```
    --data data/custom.yaml \
    --weights yolov5s.pt \
    --img 640 \
    --batch 16 \
    --epochs 100 \
    --device cpu

 说明
 # 训练命令（Linux/Mac）
python train.py \
    --data data/custom.yaml \     # 数据配置文件路径
    --weights yolov5s.pt \       # 预训练权重
    --img 640 \                  # 输入图像尺寸
    --batch 16 \                 # 批次大小（根据GPU显存调整）
    --epochs 100 \               # 训练轮次
    --device cpu \               # 使用CPU训练（GPU用户改为 --device 0）
    --name my_training \         # 实验名称（可选）
    --cache ram                  # 使用内存缓存加速（可选）
```



训练结束，会得到一些数据和在**run/train/exp/weights目录**下会产生两个权重文件：

best.pt和last.pt

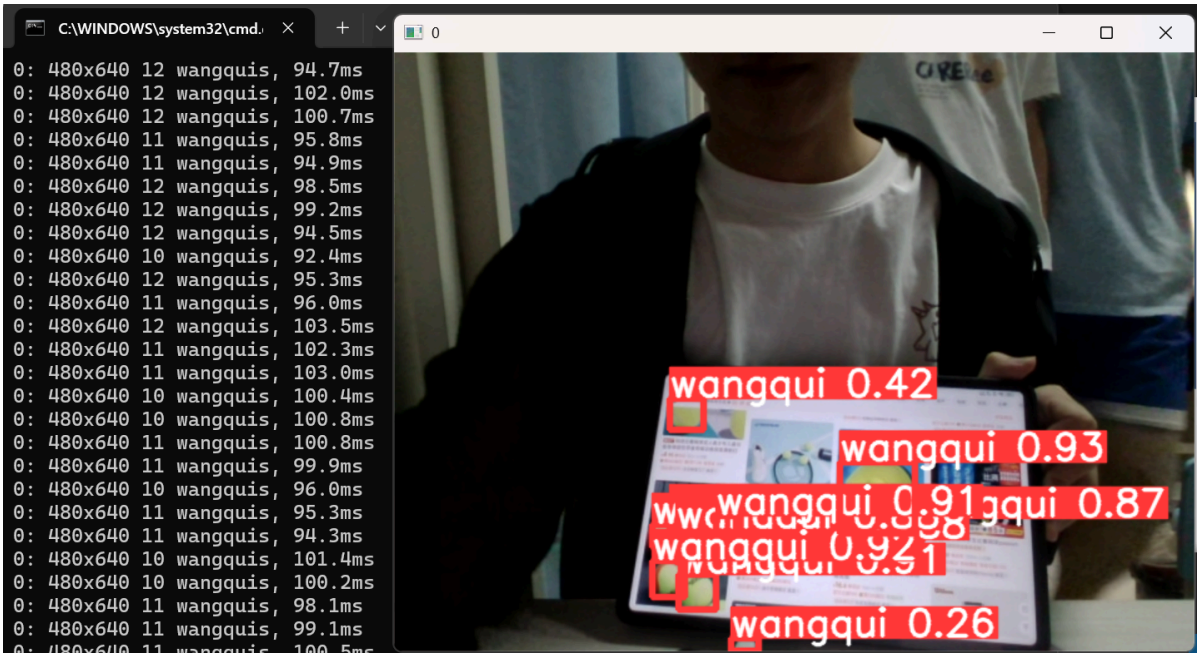


4.推理整个文件夹

```
python detect.py --weights best.onnx --source "/home/zxh/桌面/yolov5/VOC2008/VOC2008/images/train" --device cpu

python detect.py --weights best.onnx --source "/home/zxh/桌面/yolov5/VOC2008/VOC2008/images/train" --device cpu

onnx

python detect.py \
  --weights best.onnx \           # 指定 ONNX 模型路径
  --source "/home/zxh/桌面/yolov5/VOC2008/VOC2008/images/train" \  # 输入图像目录
  --device cpu                    # 使用 CPU 推理

  python detect.py \
  --weights best.onnx \
  --source "/home/zxh/桌面/yolov5/VOC2008/VOC2008/images/train" \
  --device cpu
```

5.摄像头实时推理命令

```
python detect.py --weights runs/train/exp/weights/best.pt --source 0 --device cpu
```



可以调整检测灵敏度

```
python detect.py --weights runs/train/exp/weights/best.pt --source "test.jpg" --conf-thres 0.5 --iou-thres 0.4 --device cpu
```

6.转化成onnx模型

为了方便在不同环境推理，需要弄成onnx文件

在 YOLOv5 中，将训练好的 `.pt` 模型转换为 `ONNX` 格式的命令如下：

```
python export.py --weights yolov5s.pt --include onnx --opset 12
```

```
(yolov5_final) C:\Users\zheng\yolov5>python export.py --weights C:\Users\zheng\yolov5\runs\train
\exp\weights\best.pt --include onnx --opset 12
export: data=C:\Users\zheng\yolov5\data\coco128.yaml, weights=['C:\\Users\\zheng\\yolov5\\runs\\
train\\exp\\weights\\best.pt'], imgsz=[640, 640], batch_size=1, device=cpu, half=False, inplace=
False, keras=False, optimize=False, int8=False, dynamic=False, simplify=False, opset=12, verbose
=False, workspace=4, nms=False, agnostic_nms=False, topk_per_class=100, topk_all=100, iou_thres=
0.45, conf_thres=0.25, include=['onnx']
YOLOv5  v7.0-0-g915bbf2 Python-3.8.2 torch-1.13.1+cpu CPU

Fusing layers...
Model summary: 157 layers, 7012822 parameters, 0 gradients, 15.8 GFLOPs

PyTorch: starting from C:\Users\zheng\yolov5\runs\train\exp\weights\best.pt with output shape (1
, 25200, 6) (13.8 MB)

ONNX: starting export with onnx 1.12.0...
ONNX: export success  2.6s, saved as C:\Users\zheng\yolov5\runs\train\exp\weights\best.onnx (27.
2 MB)

Export complete (4.5s)
Results saved to C:\Users\zheng\yolov5\runs\train\exp\weights
Detect:          python detect.py --weights C:\Users\zheng\yolov5\runs\train\exp\weights\best.on
nx
Validate:        python val.py --weights C:\Users\zheng\yolov5\runs\train\exp\weights\best.onnx
PyTorch Hub:     model = torch.hub.load('ultralytics/yolov5', 'custom', 'C:\Users\zheng\yolov5\r
uns\train\exp\weights\best.onnx')
Visualize:       https://netron.app
```

参数说明

参数	作用
`--weights yolov5s.pt`	指定要转换的 PyTorch 模型文件（如 `yolov5s.pt`）。
`--include onnx`	指定输出格式为 ONNX。
`--opset 12`	指定 ONNX 算子集版本（推荐 12 或更高）。
`--dynamic`（可选）	导出动态维度（适用于可变输入尺寸）。
`--simplify`（可选）	使用 `onnx-simplifier` 优化模型结构。

作用

1. **跨平台部署**

   ONNX 是通用模型格式，可在多种框架（如 TensorRT、OpenVINO、ONNX Runtime）中运行，便于移植到不同硬件（CPU/GPU/移动端）。

2. **性能优化**

   ONNX 模型可通过工具（如 TensorRT）进一步优化，提升推理速度，

   并且脱离 PyTorch 环境，简化部署流程。