# 说明文档

目录

[说明文档 1](#_Toc200497221)

[训练部分 2](#_Toc200497222)

[PyTorch 2](#_Toc200497223)

[YOLO 3](#_Toc200497224)

[SSD和RetinaNet 5](#_Toc200497225)

[ROS部分 8](#_Toc200497226)

[ROS2 8](#_Toc200497227)

[ROS及Linux常用命令 9](#_Toc200497228)

[程序使用说明 10](#_Toc200497229)

[其他工具 13](#_Toc200497230)

[补充 14](#_Toc200497231)

## 训练部分

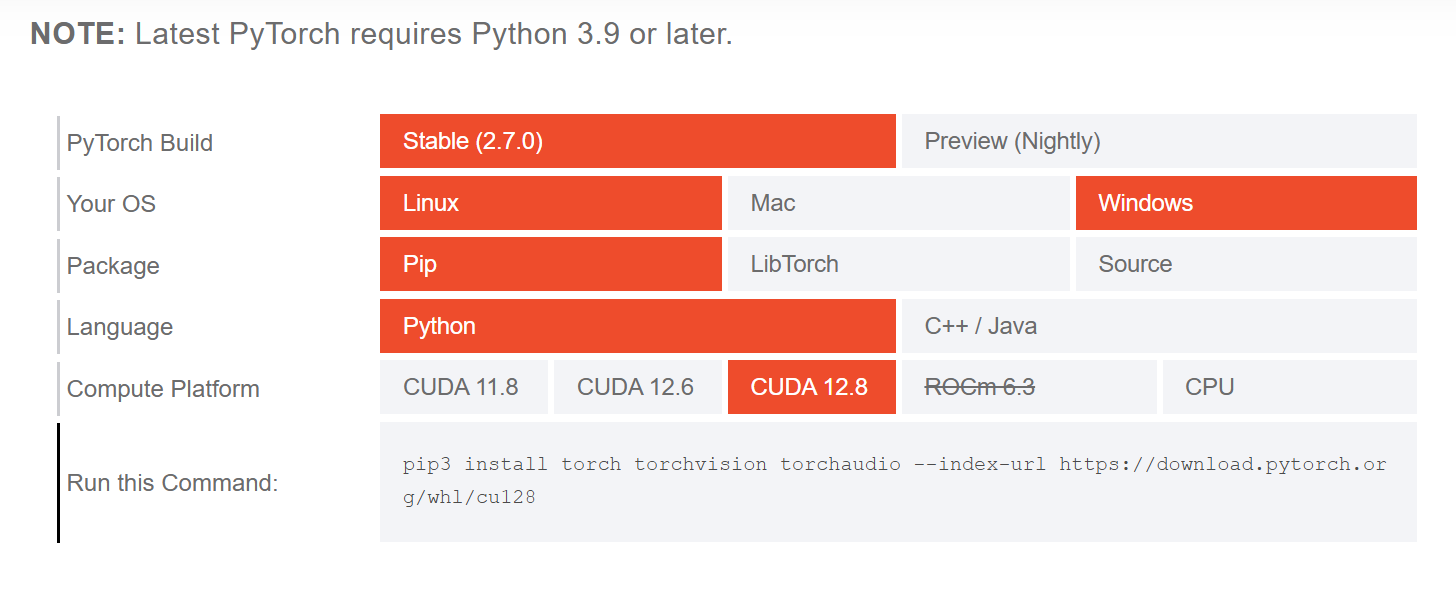
珍惜时间，珍爱生命，建议使用算力云平台进行训练。

以下是windows下的训练方式，大同小异

### PyTorch

官网：<https://pytorch.org/>

安装pytorch：<https://pytorch.org/get-started/locally/>



Nvdia选cuda，没有显卡选cpu。

实在不会的话，安装教程可参考<https://www.bilibili.com/video/BV1Vx4y1i7ds/>

~~Conda和vscode太基础了默认会使用。~~

### YOLO

**You Only Look Once**

使用Ultralytics框架。

官网：<https://docs.ultralytics.com/zh/>

入门教程参考：<https://www.bilibili.com/video/BV1A56bYtEFR/>，其中的装饰器，可以参考：<https://www.bilibili.com/video/BV1Gu411Q7JV/>

Ultralytics仓库：https://github.com/ultralytics/ultralytics

YOLO发展：<https://www.bilibili.com/video/BV1oN4113717/>

YOLO理论：<https://www.bilibili.com/video/BV1yi4y1g7ro/>

文件部分：

yolo\_amp\_ultralytics

|--ultralytics-8.3.110 ultralytics仓库

|--datasets 训练数据存放的地方

|--program 使用的程序

|-- models 下载的模型

|-- runs 训练/测试/验证/预测…结果

|-- windows\_v1.8.1 手动标注工具

|--其他不重要

yolo\_amp\_ultralytics是主文件夹，ultralytics-8.3.110是Ultralytics仓库。

终端cd切换到ultralytics-8.3.110目录，然后是用pip install -e . 进行本地安装。

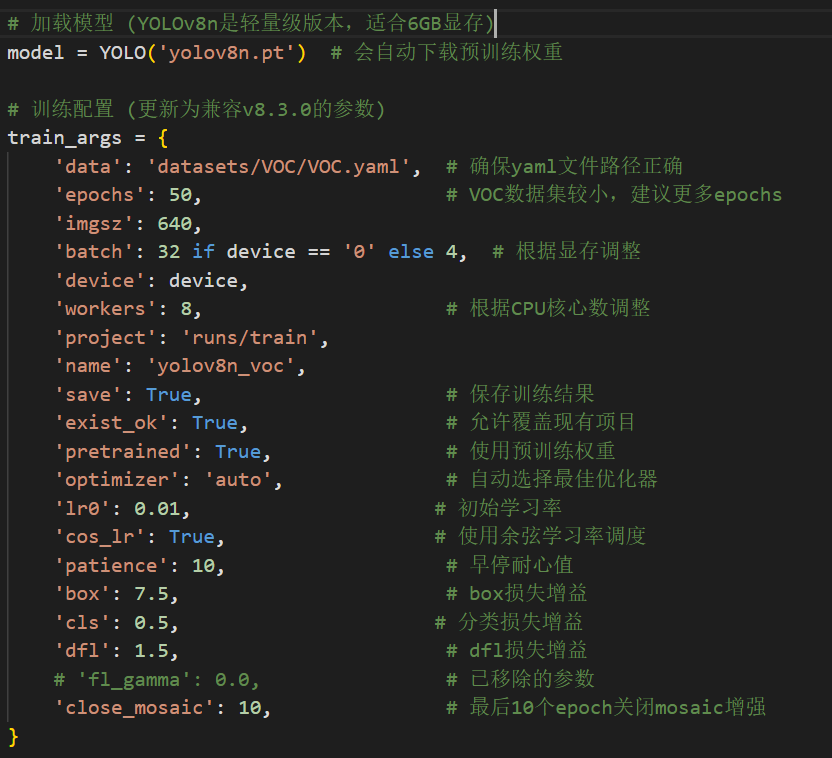
然后进入program，运行detectv11.py，自带一个示例程序，可以检查安装有没有问题。

普通训练可以使用train.py，觉得慢可以使用train\_turbo.py

predict.py是一个预测程序。

Validate.py是一个验证集程序，benchmark.py是性能评估程序。如果遇到报错，解决办法见<https://zhuanlan.zhihu.com/p/1892947083977270162>

~~其他的程序是一些玩具。~~

~~~~超参数说明

因为训练和验证使用的超参数有重合部分，这里用训练的超参数举例。

model = YOLO('yolov8n.pt') 改成想要训练的模型

'data': 'datasets/VOC/VOC.yaml', 训练集模板路径

'epochs': 50, 训练轮数

'batch': 32 if device == '0' else 4, batch，3060 6G可以到32,4090可以到96或128（128无明显训练速度提示）

'workers': 8, 填cpu核心数

'project': 'runs/train', 训练模型保存到的文件夹

'name': 'yolov8n\_voc', 训练模型保存到的文件夹名

### SSD和RetinaNet

**SSD**

**Single Shot MultiBox Detector**

理论：<https://www.bilibili.com/video/BV1fT4y1L7Gi/>

源码解析：<https://www.bilibili.com/video/BV1vK411H771/>

源码：<https://github.com/WZMIAOMIAO/deep-learning-for-image-processing/tree/master/pytorch_object_detection/ssd>

**RetinaNet**

**Focal Loss for Dense Object Detection**

理论：RetinaNet：<https://www.bilibili.com/video/BV1vK411H771/>

源码：<https://github.com/WZMIAOMIAO/deep-learning-for-image-processing/tree/master/pytorch_object_detection/retinaNet>

文件说明：

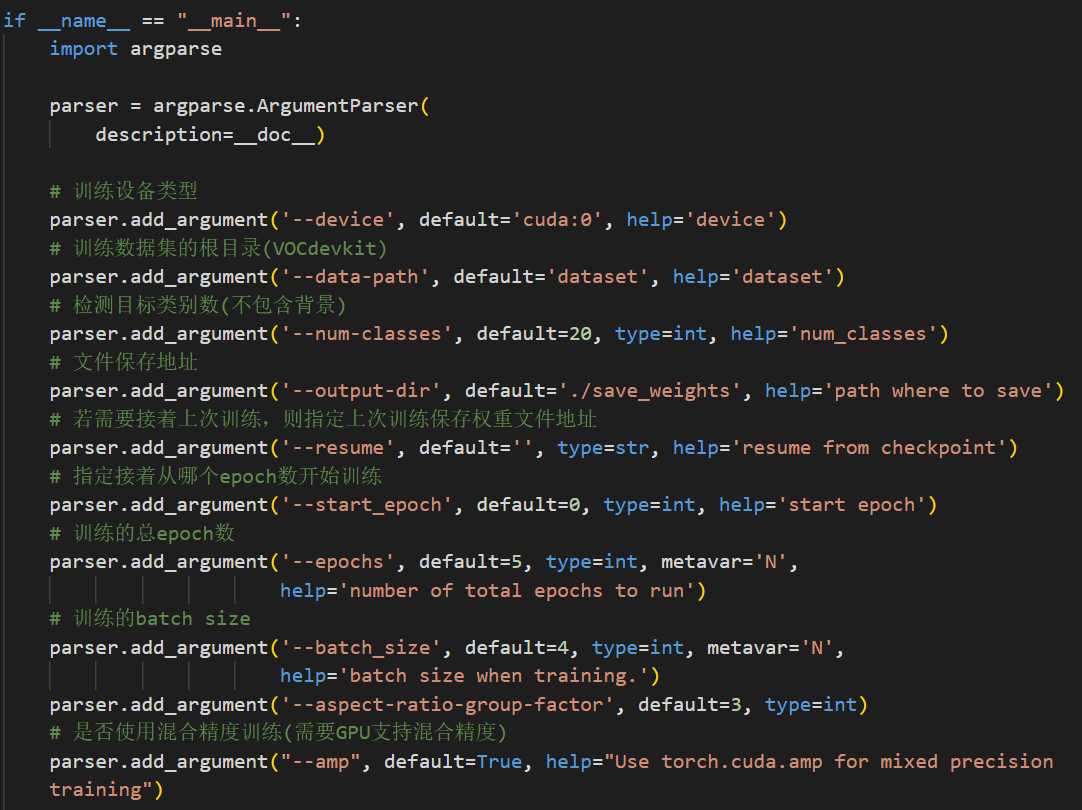
**ssd**：train\_ssd300.py用于训练模型。

**retinanet**：train.py用于训练模型。

其余部分两个模型一样

validation.py用于验证，validation\_tensorboard.py加入了tensorboard~~，并没有什么用~~。

predict.py用于预测。

训练/验证超参数：

因为训练和验证使用的超参数类似，这里用训练的超参数举例。

parser.add\_argument('--data-path', default='dataset', help='dataset') dataset改成训练/测试集路径。

parser.add\_argument('--num-classes', default=20, type=int, help='num\_classes') 根据训练/测试集种类进行修改

parser.add\_argument('--epochs', default=5, type=int, metavar='N',help='number of total epochs to run') 修改epoch数（训练轮数），如果只是看一下程序能不能用，建议改成1，训练模型建议30。

parser.add\_argument('--batch\_size', default=16, type=int, metavar='N', help='batch size when training.') 修改batch数，cpu的不知道

ssd：3060 6G可以用16或32，4090可以到64，内存16G的建议4-8容易爆内存，训练速度较为正常。

retinanet：3060 6G用4,4090用32，改大了会爆显存，内存16G的不一定能训练，训练速度非常慢。

## ROS部分

### ROS2

ROS2教程： 古月·ROS2入门21讲<https://www.bilibili.com/video/BV16B4y1Q7jQ/>

图文教程<https://book.guyuehome.com/>

教程源码： Gitee：<https://gitee.com/guyuehome/ros2_21_tutorials>

Github：<https://github.com/guyuehome/ros2_21_tutorials>

### ROS及Linux常用命令

见detector\_ws/笔记.md

### 程序使用说明

算力资源有限，所以实验思路是将Gazebo仿真和模型检测分开。

文件结构：

ros

|--detector\_ws 工作区

|--src 代码包

|--benchmark 地图 ~~没错，这里是地图，名字是瞎起的~~

|--retina\_detector retina

|--ssd\_detector ssd

|--yolo\_detector yolo

|--rosbag2\_2025\_04\_26-23\_56\_14 录制的gazebo仿真

#### world地图

cd detector\_ws

colcon build #编译

source install/local\_setup.bash #环境变量

ros2 launch benchmark city.launch.py #运行

#### 导入机器人模型、遥控和bag录制

export TURTLEBOT3\_MODEL=waffle\_pi #导入waffle\_pi

ros2 run teleop\_twist\_keyboard teleop\_twist\_keyboard #使用键盘遥控

ros2 bag record /camera/image\_raw #使用bag录制/camera/image\_raw

新开一个终端，

ros2 bag play /home/ros/rosbag2\_2025\_04\_26-23\_56\_14 #使用录制的bag

#### rosbag topic转mp4

/其他/rosbag2video.py

先运行脚本再运行rosbag。基于ffmpeg。

示例：python3 rosbag2video.py --topic /camera/image\_raw --output output.mp4 --fps 30 --crf 21

#### YOLO SSD和RetinaNet

cd detector\_ws

colcon build #编译

source path/install/local\_setup.bash #环境变量

**摄像头话题**：新开一个终端，输入：rviz2，打开rviz，话题是/camera/image\_raw的Image

（需搭配摄像头话题使用）

ros2 launch yolo\_detector yolo\_detector.launch.py #yolo运行

（会保存为yolo\_raw，且会覆盖，**记得保存**）

ros2 launch ssd\_detector.launch.py #SSD运行

（会保存为detector\_ws/ssd\_results）

ros2 launch retina\_detector.launch.py #RetinaNet运行

（会保存为detector\_ws/retina\_results）

**修改yolo模型**

打开detector\_ws\src\yolo\_detector\config\params.yaml，修改model\_path: "yolo11n.pt"

**修改ssd和retinanet模型**

打开detector\_ws\src\retina\_detector\src，替换resNetFpn-model-29.pth

ssd的类似

## 其他工具

/实验结果/video.py。将文件夹中图片转为mp4

/实验结果/sample\_images.py。文件夹中图片随机抽取其中10%

使用ps“联系表”批量拼图<https://www.bilibili.com/video/BV1VZ4y1r7hy/>

## 补充

值得一看的内容：

卷积神经网络基础<https://www.bilibili.com/video/BV1b7411T7DA/>

基础看完可以看LeNet、AlexNet、ResNet（这三个是CNN）、 VGG。

论文精读：跟李沐学AI <https://space.bilibili.com/1567748478>

Python基础：<https://space.bilibili.com/3546660493855256/lists/4227087>

手搓一个简单的CNN：<https://space.bilibili.com/3546660493855256/lists/3283204>

以上内容可以节约2个月左右的找教程时间

其他：

晚上12点后使用deepseek，不会出现 服务器繁忙，请稍后再试

文献管理工具推荐：Zotero