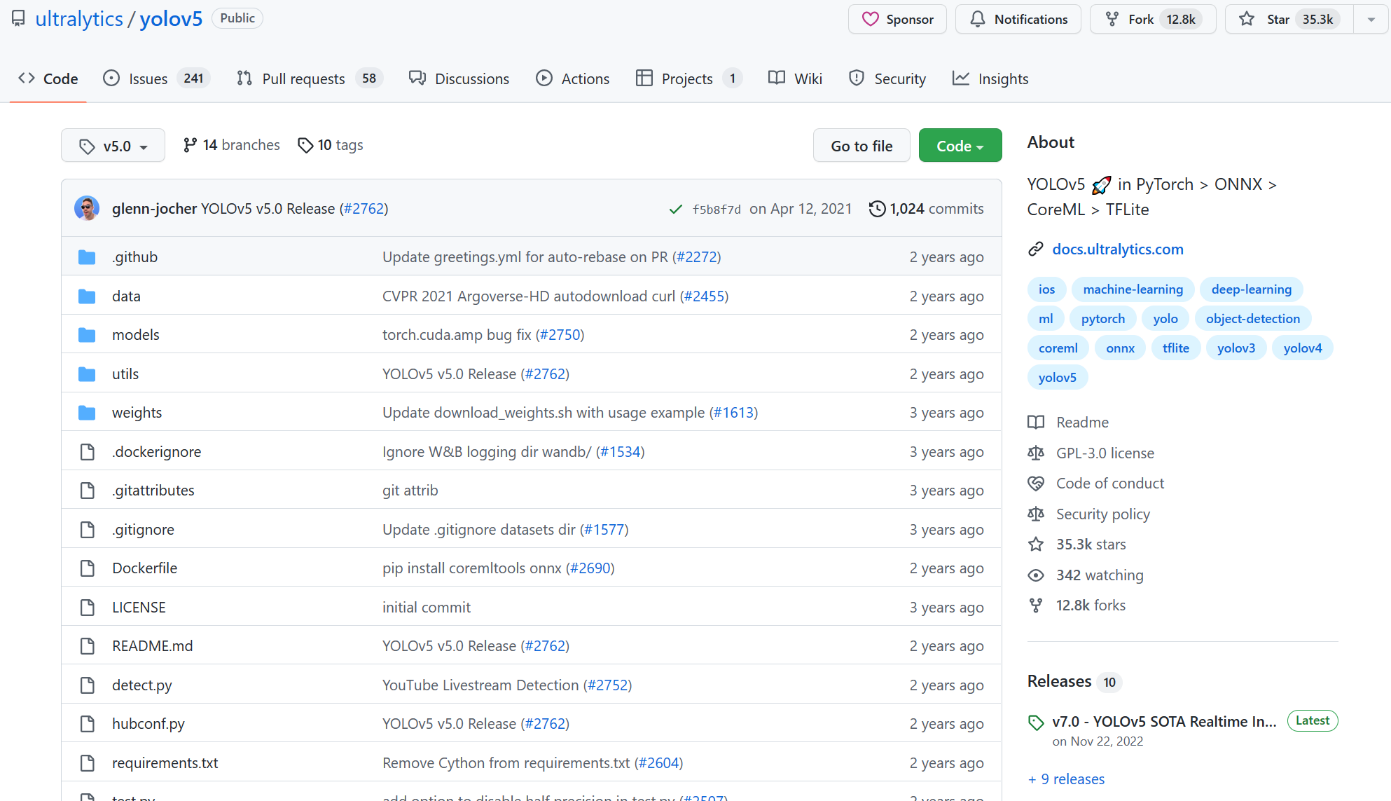
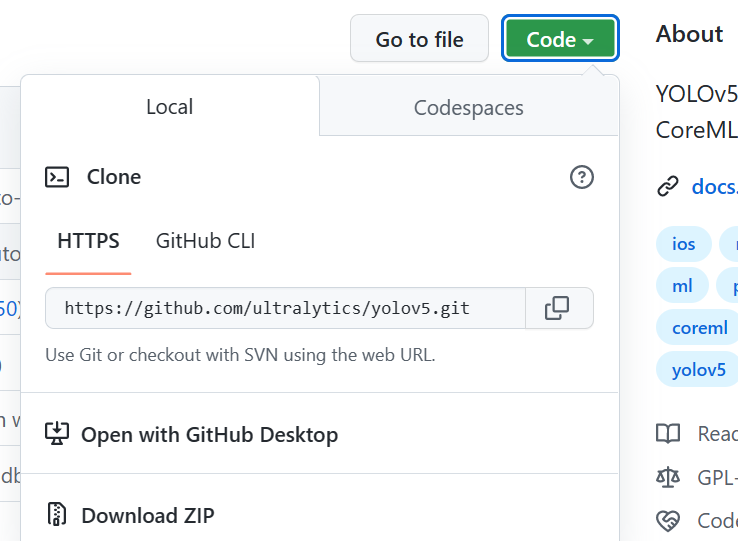
1. **如何从github上下载yolov5模型**

首先去GitHub的yolov5官网，界面如下

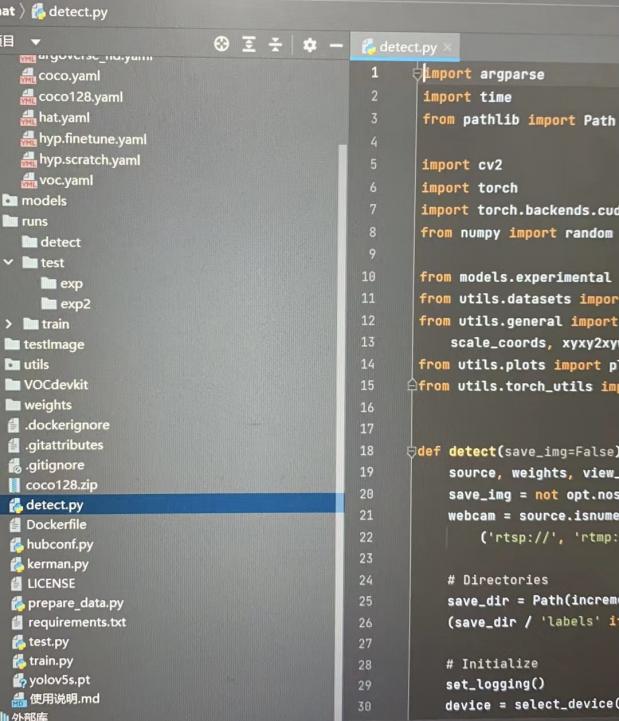
点击code，Download ZIP将整个代码下载下来

然后解压缩用PyCharm打开

除此之外还要配置环境，安装Anaconda和Pytorch，如果用CPU训练，直接安装CPU版本的Pytorch，如果要用GPU，则需要进行NVIDIA驱动安装与更新，和Cuda的安装。

下载后，进入pytorch环境进入yolov5文件夹。

yolov5的项目结构：



（1）data文件夹里放着yaml的参数配置文件；

（2）models文件夹放着训练模型；

（3）weights文件夹里放着训练好的权重参数；

（4）train.py文件是进行训练自己的数据集的函数；

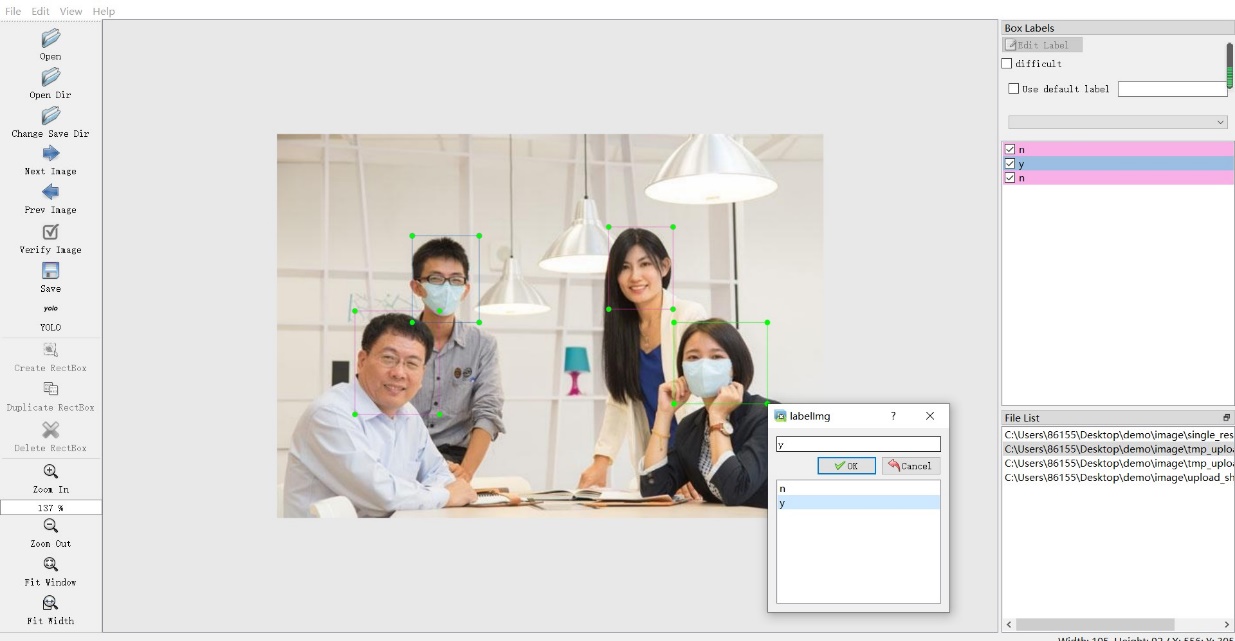
（5）detect.py文件是利用训练好的权重参数进行目标检测；

（6）test.py文件是用来测试训练的结果的函数；

（7）requirement.txt文件

1. 如何建立数据集

1.准备数据集

此yolov5s工程用于口罩检测，准备5000~8000数据集。数据集应统一后缀格式，统一选择 .jpg格式。

2.标注数据集

使用lambelimg进行数据打标签的内容，将标签分为两类：戴口罩（mask）、未戴口罩（face），我们框选要进行识别的部分后将其分为标签中的一种，最后将数据格式转换为yolo格式。

3.将数据集分类

将数据集按比例（0.7：0.2：0.1）分为训练集、测试集、验证集。

1. 如何进行训练

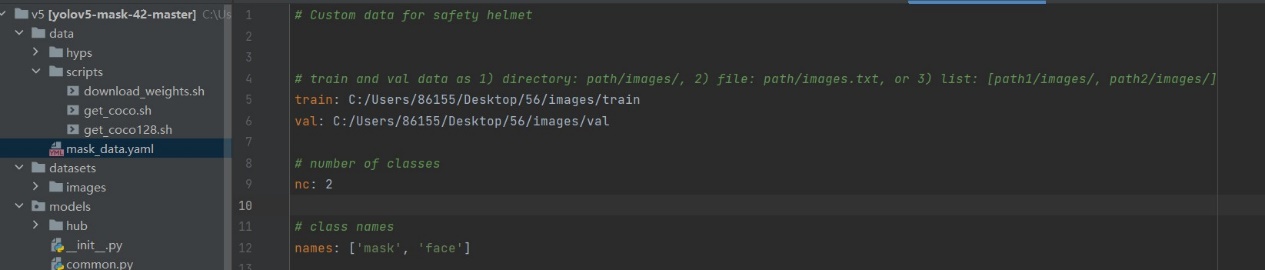
1.找代码

可以在github网站（或者其他开源网站）上下载yolov5的开源源代码。

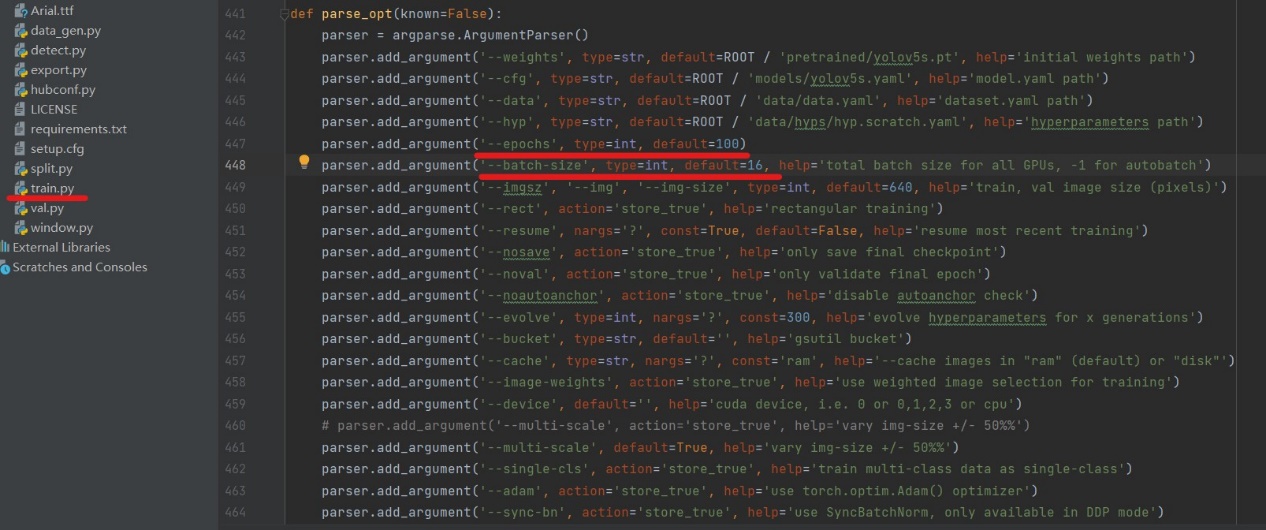
2.选择预训练权重

Yolov5中有四种选择yolov5l、yolov5m、yolov5s、yolov5x。其中yolov5s网络最小，速度最快，AP精度最低，比较适合检测大目标且追求速度，也可以应用于嵌入式设备。我们最终选择应用yolov5s网络。

3.修改数据配置文件参数

找到data目录下的yaml文件，在train,val后填入训练集和测试集的绝对路径；在#number of classes nc:后面填入检测的类别数。（此工程为检测口罩和人脸，所以类别数填入2）；在name后面填入需要识别的类别英文名称。

4.修改模型参数

找到train.py文件，将模型主要参数调整为自己所需参数值。例如将weight权重路径填入，我们选择的是yolov5s.pt;将自己修改好的模型yolov5s.yaml文件路径填入到cfg’中；数据.yaml文件路径填入到data’中；epochs训练轮数我们调整为100轮，batch-size我们调整为16。全部调整完就可以运行train.py进行模型训练了。

1. 如何评估训练结果

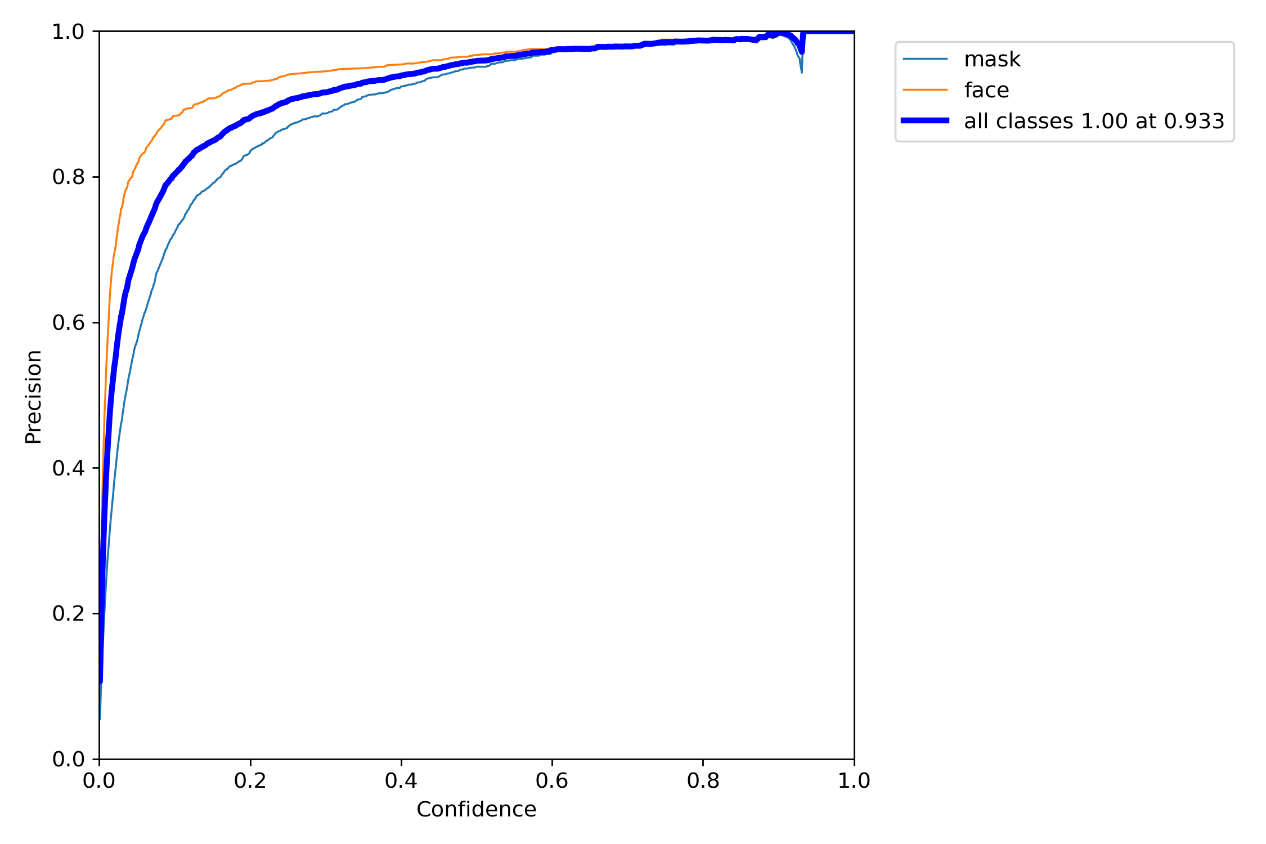
模型评估指标分别为检测精度和检测速度。检测精度可以利用precision, recall , mAP, confusion\_matrix, P-R曲线进行分析。检测速度可以利用fps每秒帧数，前传耗时进行分析。

精确率是指分类正确的正样本个数占分类器判定为正样本的样本个数的比例，即Precision=TP/(TP+FP)

召回率是指分类正确的正样本个数占真正的正样本个数的比例，即Recall=TP/(TP+FN)

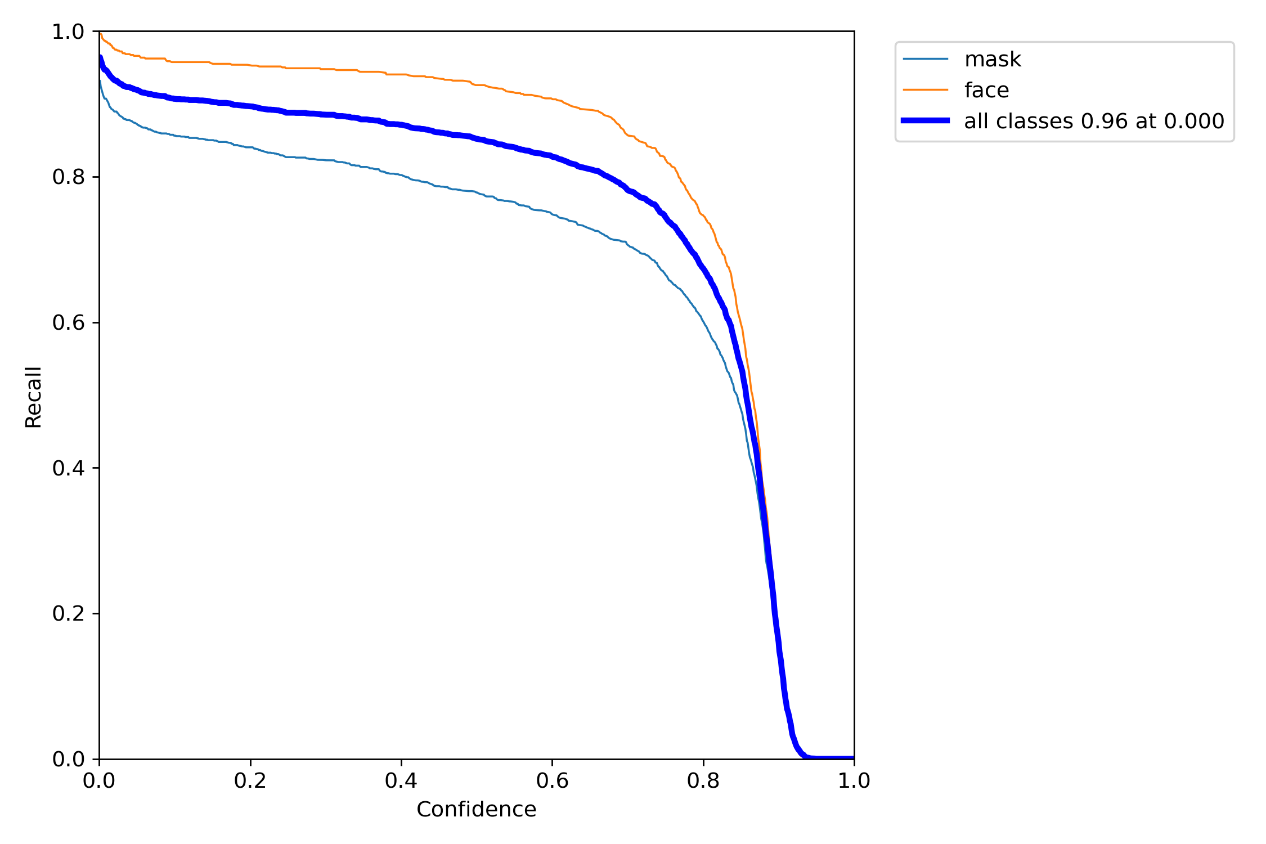
精度率和召回率是互相矛盾的，往往模型具有较高的精确率的时候召回率会很低，反之召回率越高则精确率越低。

P\_curve:精确率与置信度关系图



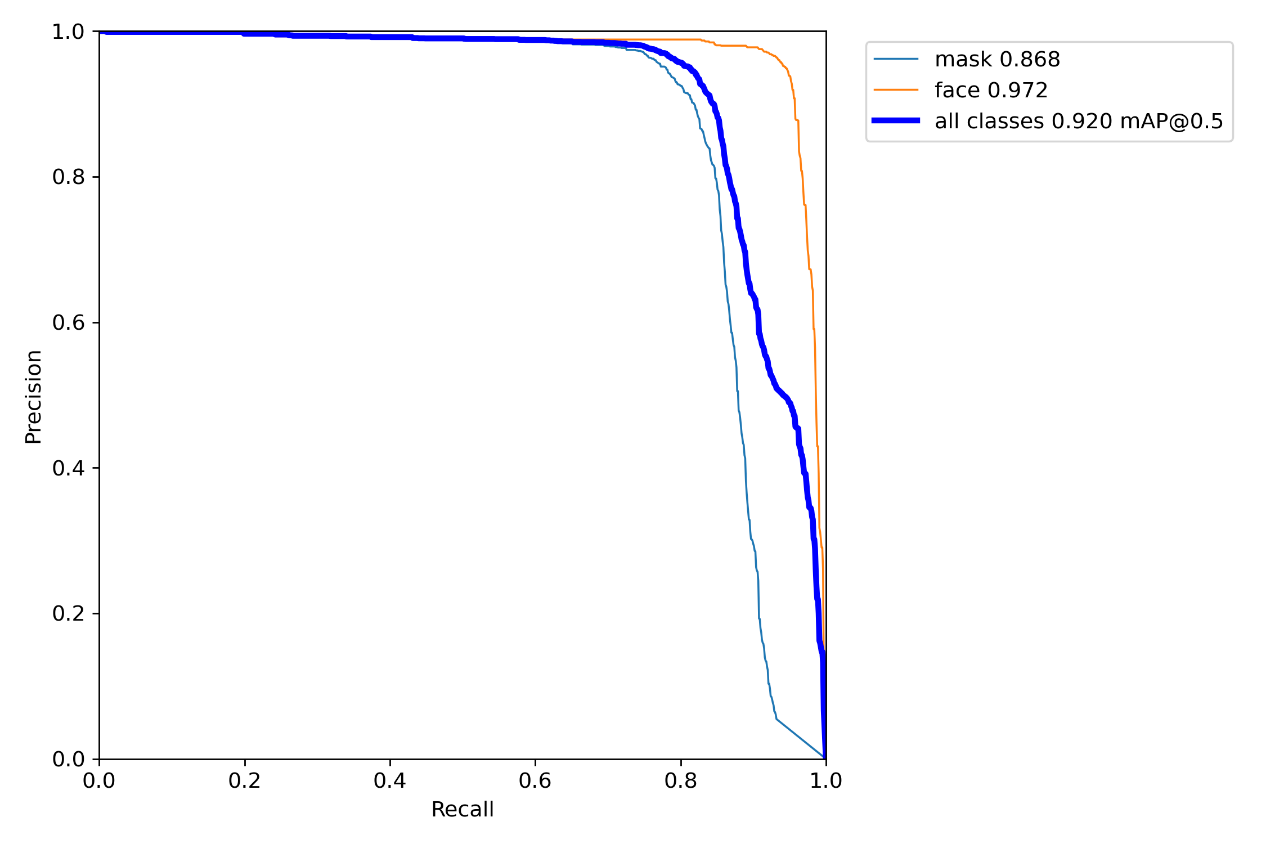
当置信度为某一特定值时，各类别的精确率。

R\_curve:召回率与置信度关系图



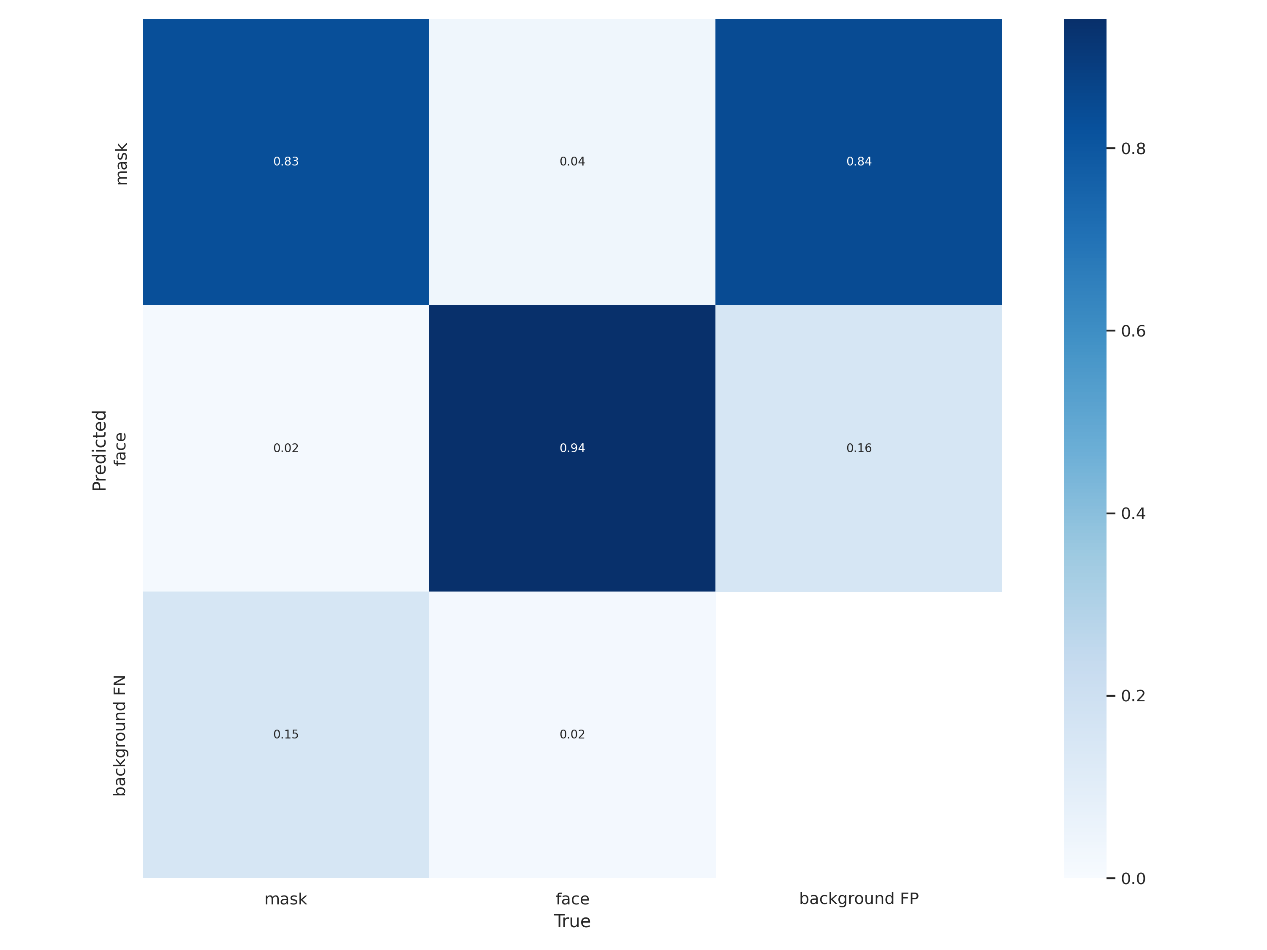
当置信度为某一特定值时，各类别的召回率。

PR\_curve:精确率与召回率关系图



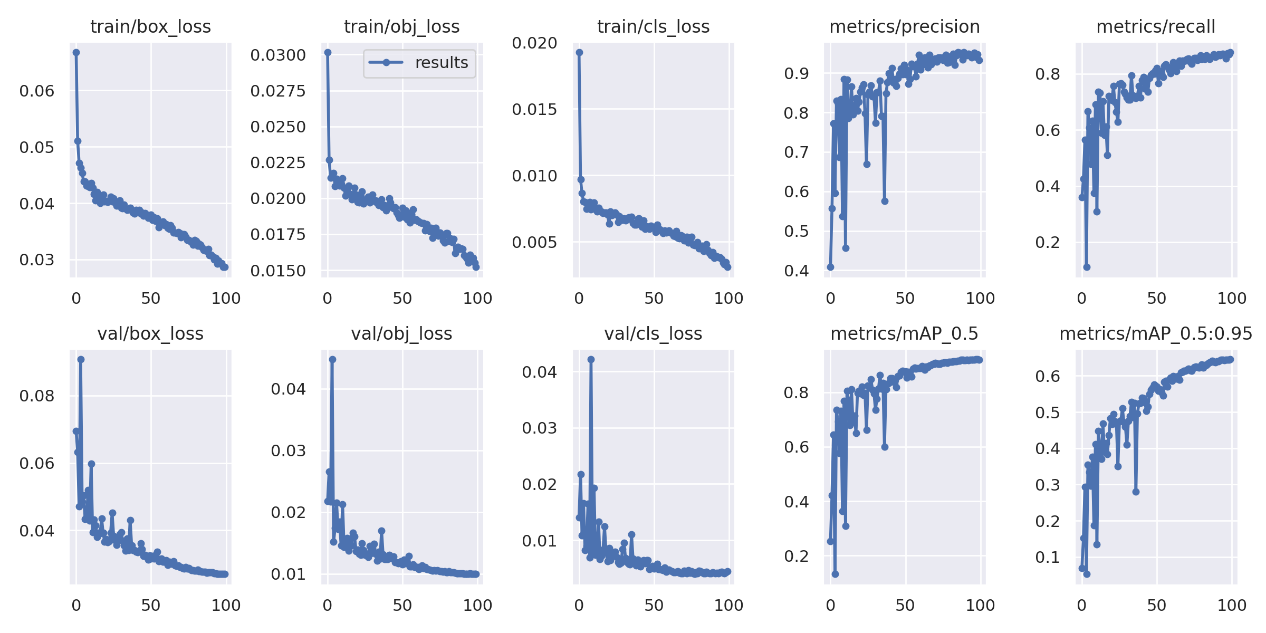
因为精度率和召回率是互相矛盾的，往往模型具有较高的精确率的时候召回率会很低，反之召回率越高则精确率越低。最理想情况为（1，1）点，mAP值越接近于1效果越好，两者围成的面积就是mAP值。

confusion\_matrix：混淆矩阵



混淆矩阵是用于评价分类模型效果的NxN矩阵，其中N是目标类别的数目。矩阵将实际类别和模型预测类别进行比较，评价模型的预测效果。比如样本的实际类别是狗，若模型预测类别也是狗，则说明对于该样本模型预测对了。若模型预测类别为猫，则说明对于该样本模型预测错了。对全部样本数据进行统计，可以判断模型预测对了的样本数量和预测错了的样本数量，从而可以衡量模型的预测效果。分类最理想情况为：除了background其余形成对角矩阵。

result.png：可视化训练结果解析

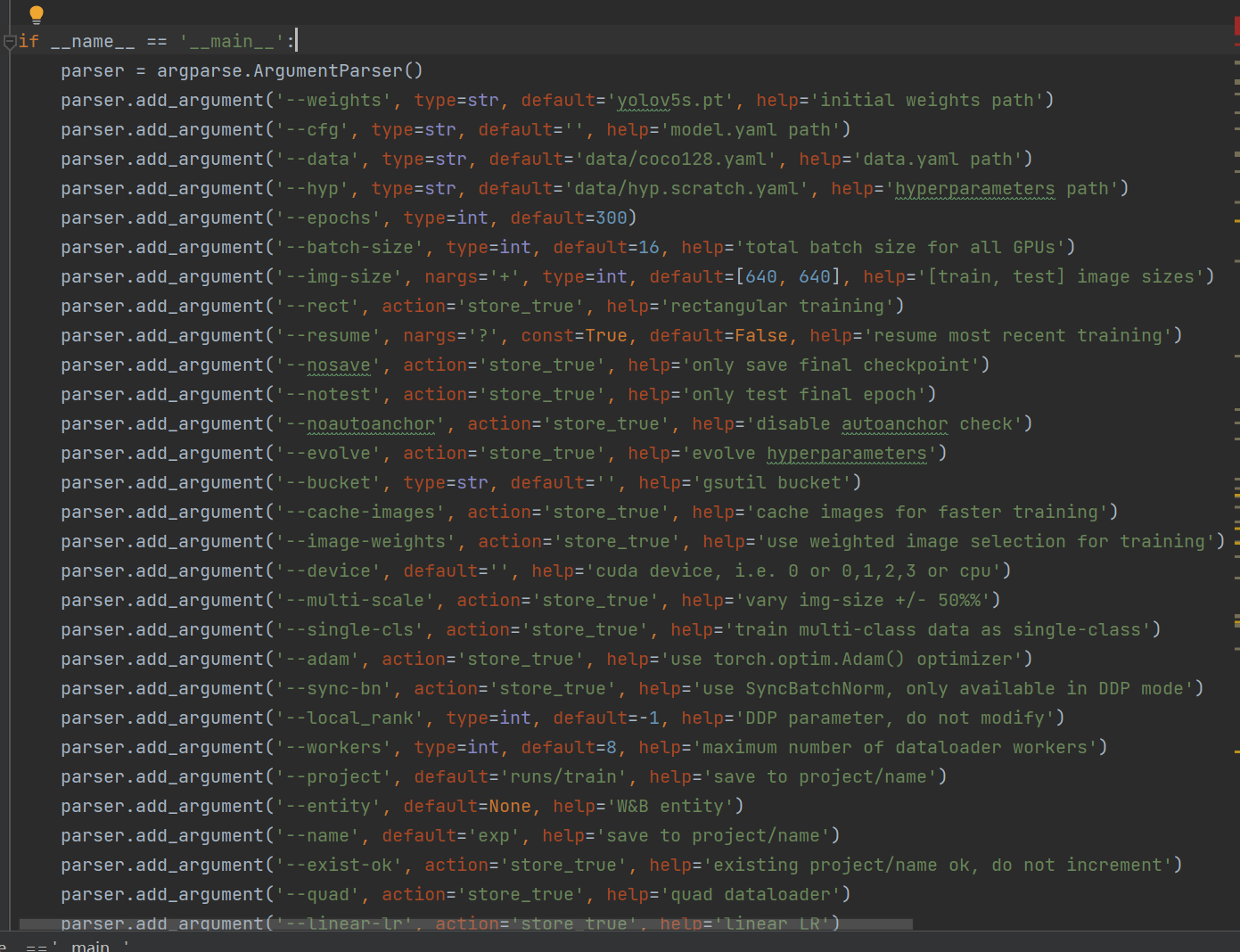


Box:推测为损失函数均值，越小方块越准。

cls(classification): 推测为分类loss均值，越小分类越精准。

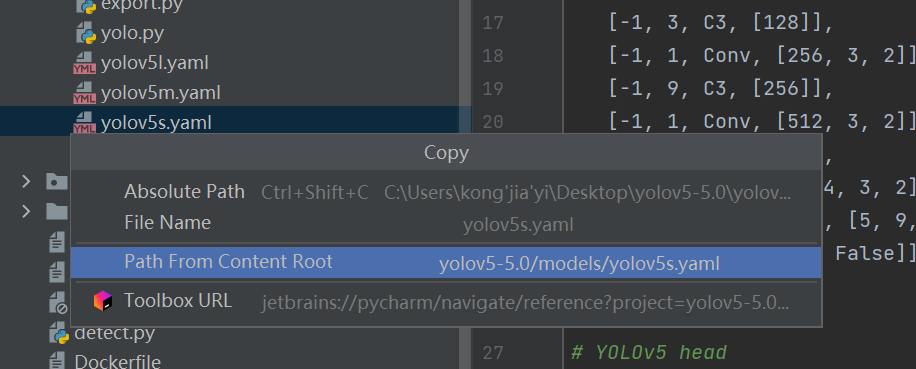
obj(objectness): 推测为目标检测loss均值，越小目标检测越精准。

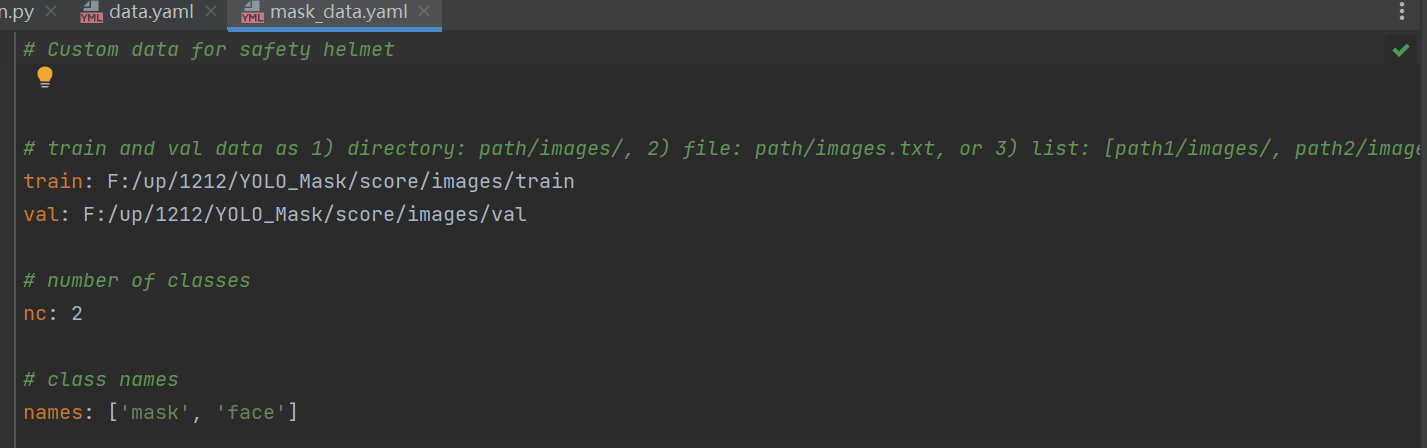
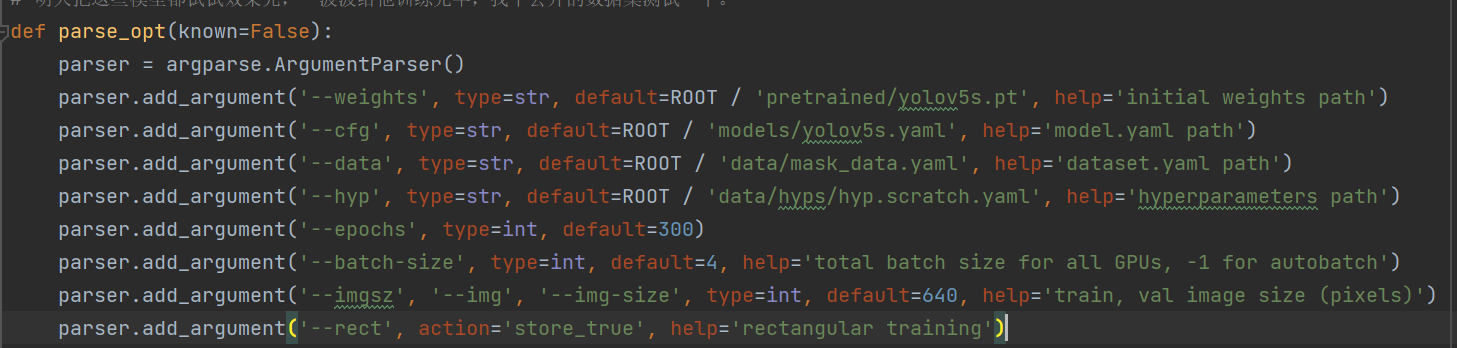
精度和召回率波动越小训练效果越好。

5.如何调整训练参数

Yolov5原始代码有很多参数，在我们训练中，需要调整的不多

**Weights：**模型的初始权重，我们采用默认值Yolov5s.pt；

**cfg：**模型结构，在口罩识别的过程中选择的是Yolov5s，在models中选择Yolov5s的路径，Yolov5s是比较轻量而且稳定的结构，params只有7.3M，训练速度较快；

**data：**数据集所在目录，修改data目录下的cc128.yaml的文件为mask\_data.yaml，train：+训练集的绝对路径，val：+验证集的绝对路径，number of classes标签的类别为2，标签为mask or face，然后把mask\_data.yaml的路径复制过来，如下

**hyp：**超参数，一般采用模型原有的默认参数

**epochs**：迭代次数，默认值是300，我们选择的是100，因为数据集较大，而且迭代次数过多可能会出现过拟合的情况

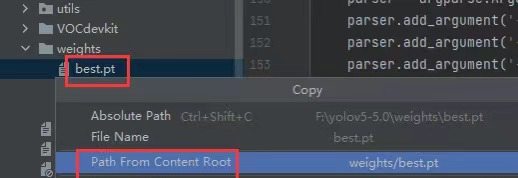
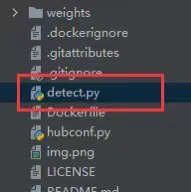
**batch-size：**一次对多少数据进行训练，取决于显存的大小，但是batch-size对训练结果的影响很小，几乎可以忽略

**imgs：**图片的缩放尺寸，Yolov5模型会处理训练集里的图片，把图片缩放到设定的尺寸，默认的是640，我们在口罩训练过程中没有做更改，如果内存太小可以选择改成320，但是不推荐。

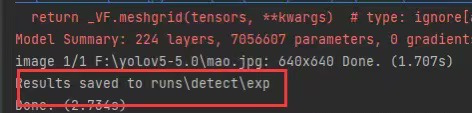
6.如何利用训练完成的模型

训练好后会出现一个runs文件夹，里面有权重的文件夹，best是最好的权重，last是最后一次训练的权重。

（1）复制训练好的best权重；

（2）粘贴到weights文件夹下；

（3）打开detect.py做推理的这个文件

1. 将best权重的路径复制粘贴

（5）运行结果保存