

# 文档：使用 YOLOv8 训练 RoboMaster Buff 检测与预测模型

---

版本：V1.0

编写日期：2025-09-17

编写人：张俊杰

适用对象：26 赛季视觉组梯队

## 目录

1. 目标.....	3
2. 环境准备 .....	3
3. 模型训练.....	19
4. 模型导出：pt $\rightarrow$ onnx（这一步大家网上查） .....	22
5. 模型转换：onnx $\rightarrow$ OpenVINO IR（这一步大家网上查） .....	22
6. 总结.....	22

## 1. 目标

本文档旨在指导各位在 Ubuntu22.04 环境下（在学习这个文档前，要求大家在自己的主系统或者虚拟机安装 anaconda 环境，熟悉 conda 命令行），使用 YOLOv8 完成 RoboMaster Buff 检测与预测模型的训练、导出和推理准备流程。通过本文档，各位将掌握数据集格式转化、模型训练、格式转换（pt → onnx → OpenVINO）等关键步骤。

## 2. 环境准备

本文档将建立在 AutoDL 平台上新建 Ubuntu22.04 容器中进行，确保具备 GPU 加速环境。（AutoDL 使用，第一种方式接着在上个文档配置的 Ubuntu22.04 的 vscode + ssh 远程连接容器、第二种方式就是使用 AutoDL 容器自带的终端（效果差不多，这种更考验命令的熟练度）链接：[AutoDL](#)）

必要环境包括：

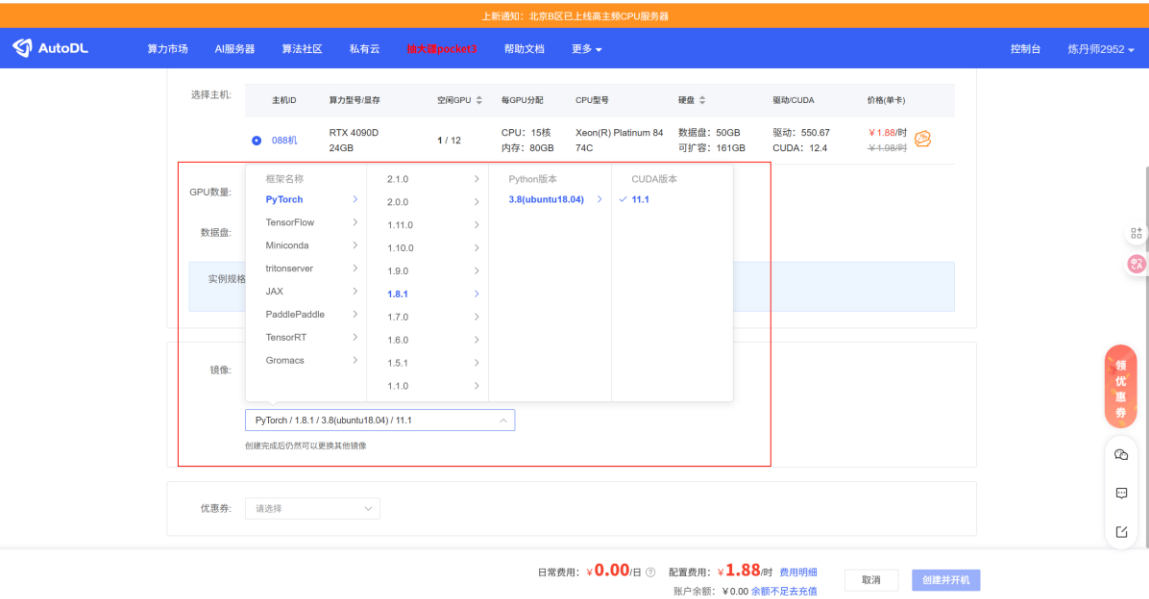
- Python >= 3.8（我们创建容器的时候选择的镜像就是 3.8 版本了）
- PyTorch（pip install torch==1.8.1+cu111 torchvision==0.9.1+cu111 torchaudio==0.8.1 -f [https://download.pytorch.org/whl/torch\\_stable.html](https://download.pytorch.org/whl/torch_stable.html)）
- Ultralytics YOLOv8（pip install ultralytics）
- OpenVINO 工具包（pip install openvino-dev）

环境搭建命令：

```
``bash
先开启 AutoDL 学术资源加速：source /etc/network_turbo
pip install openvino-dev
``
```



镜像：选择图示的镜像即可，版本不需要太高，1.8.1 比较稳定（无法创建需要先充钱保证于余额有钱的）



上传数据集：创建后会跳转到如下页面，点击文件存储，初始化对应的区域

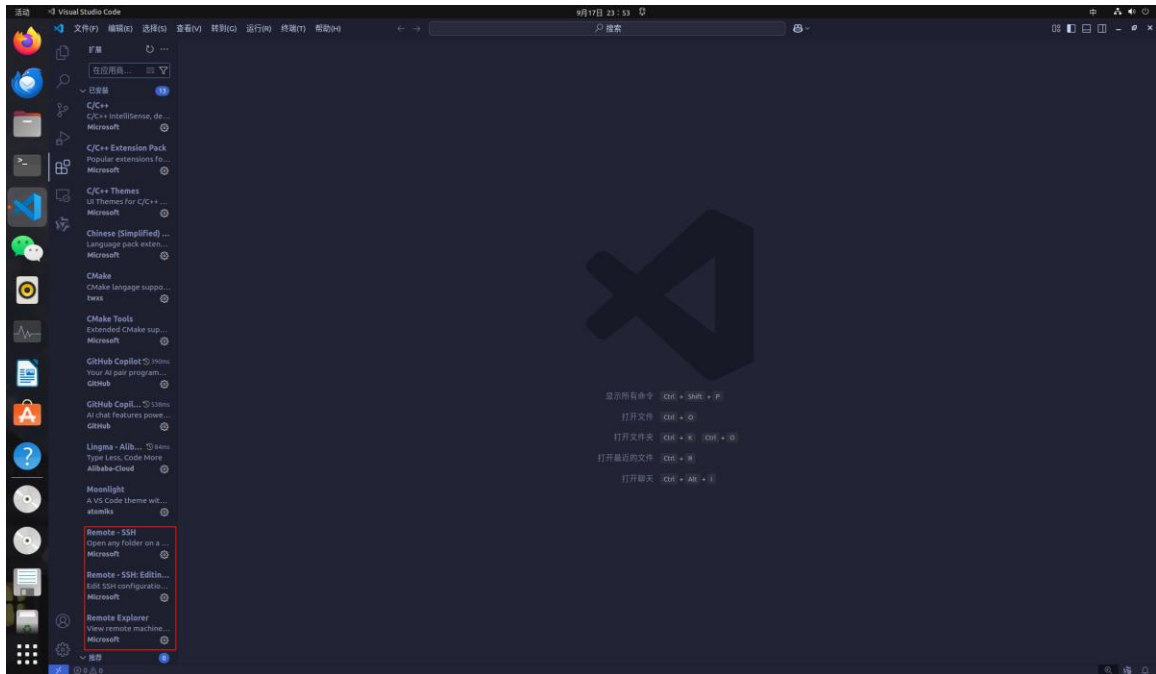


即可上传 ZIP 压缩文件了

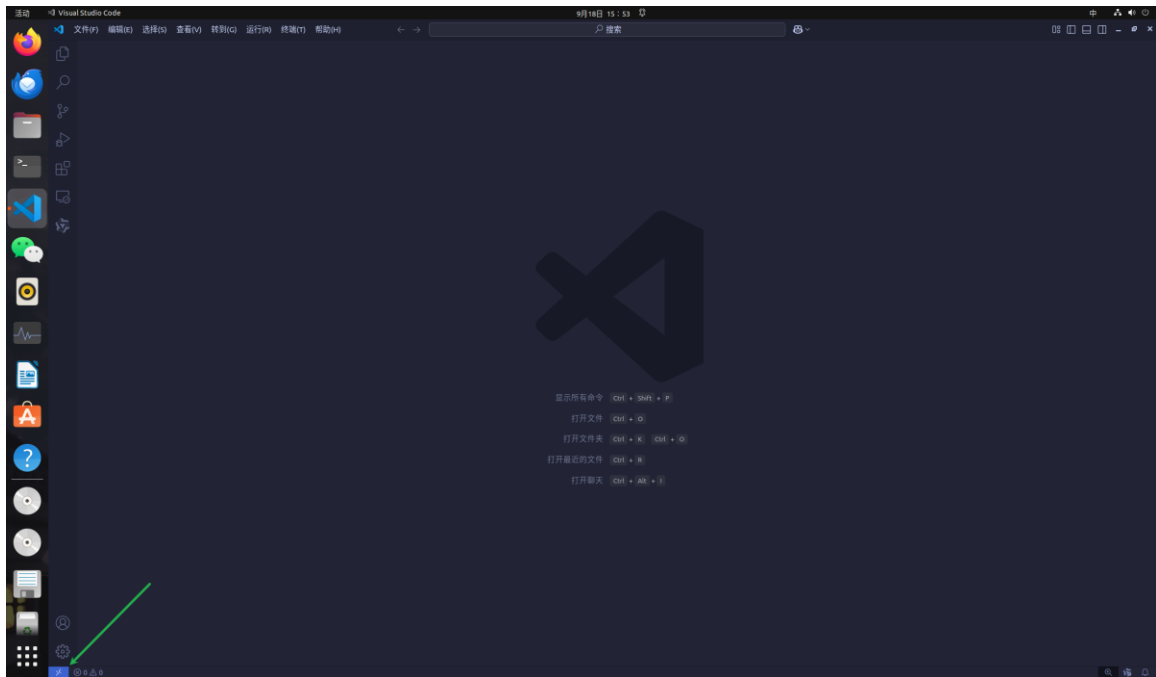


等待上传完成即可，此外，我们还需要准备 yolov8 或者 yolov5 所需要的字体（链接：[Arial.ttf](#) 需要将这个文件移动到/root/.config/Ultralytics/位置，后面配置环境会提到）

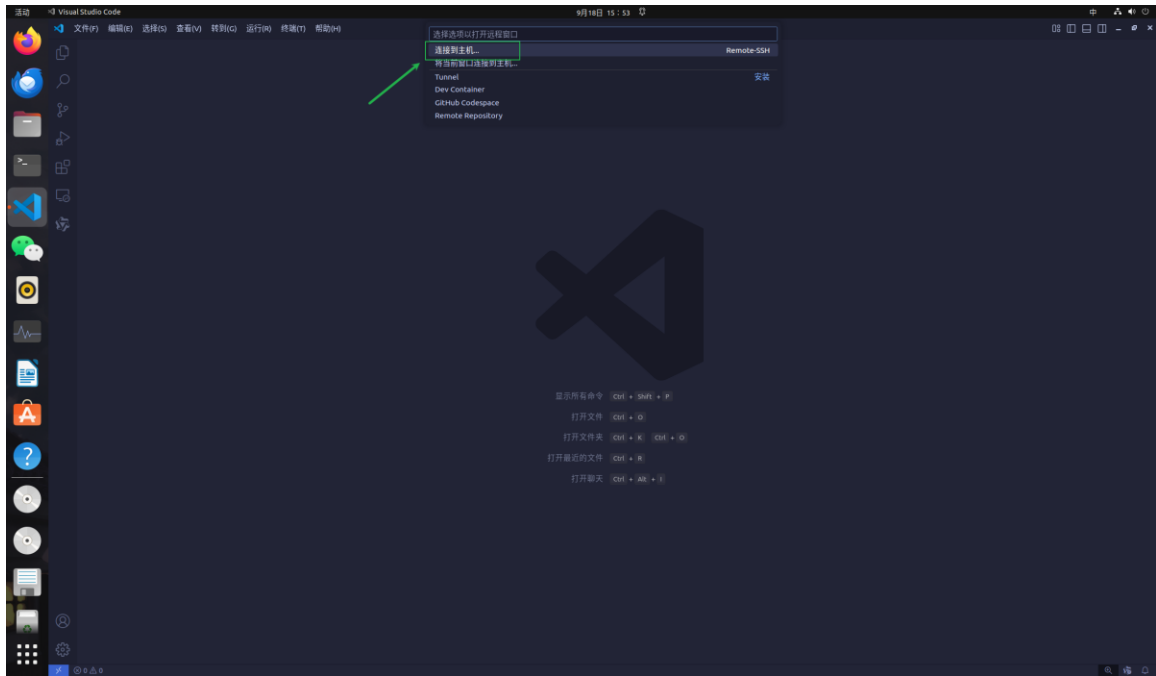
方式一：使用小鱼一键安装的 Vscode，老版本 ssh 存储方式更新版本不一样，新版本要会在你远程连接的设备下载东西 100MB，会导致输入密码还是无法建立连接。Vscode 下载插件 Remote - SSH、Remote - SSH: Editing Configuration Files、Remote Explorer



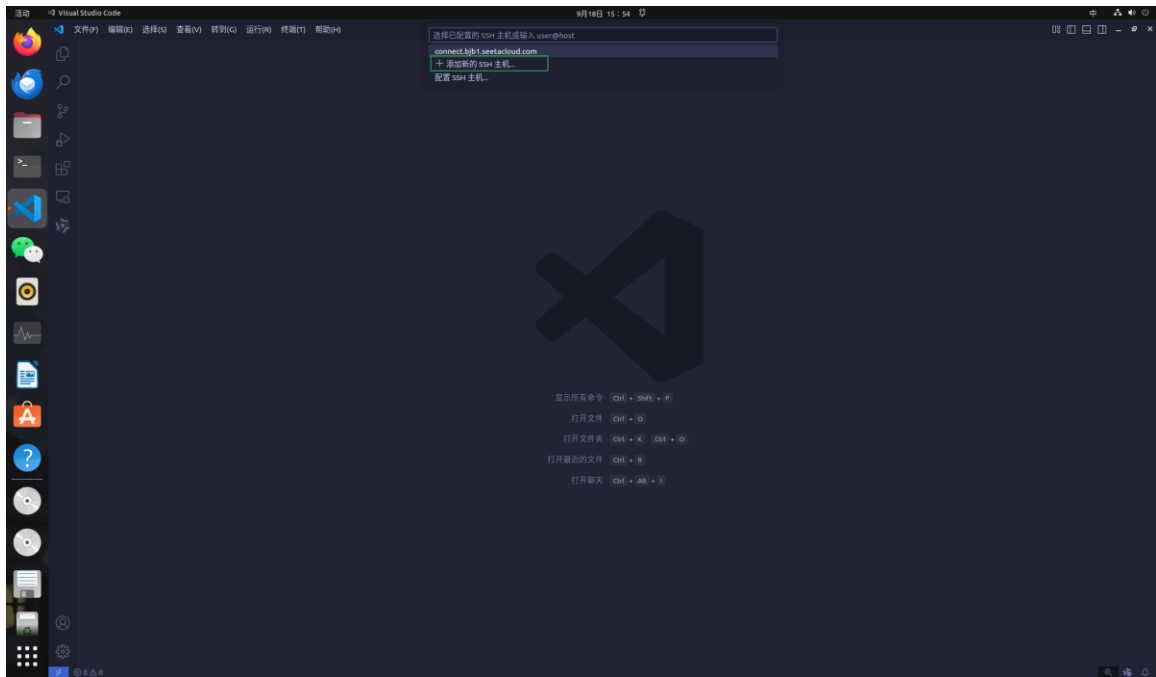
ssh 远程连接：点击下面的按钮



选择连接到主机



添加新的 ssh 主机

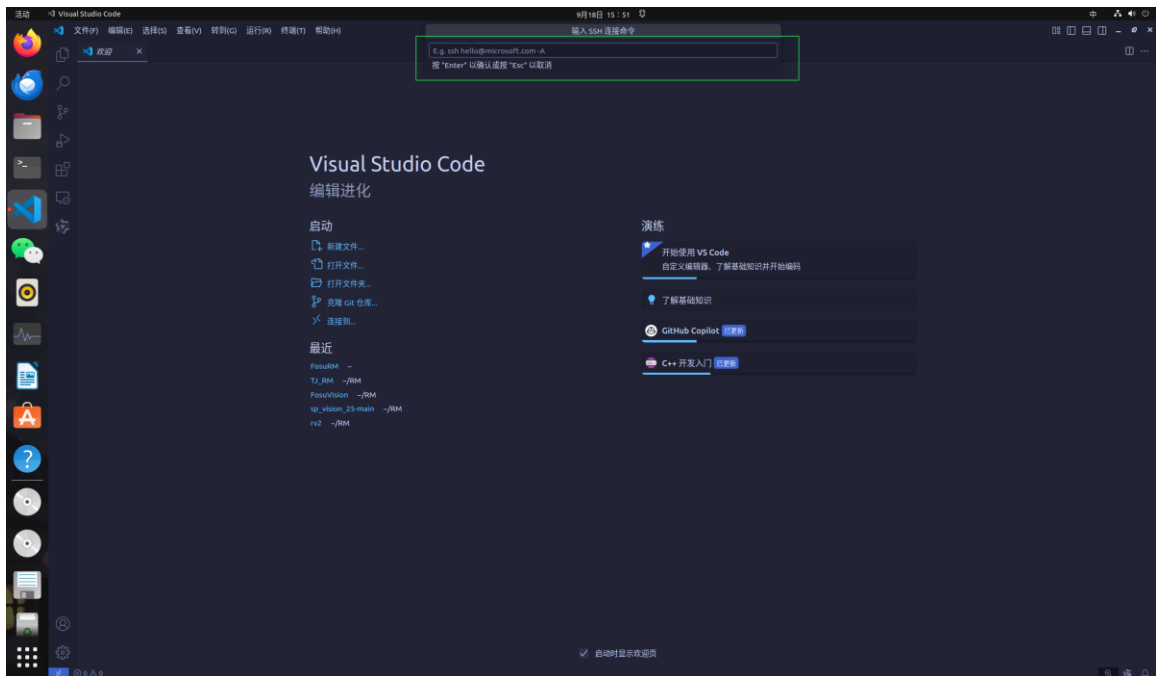


回到容器实例，复制 ssh 账号、密码

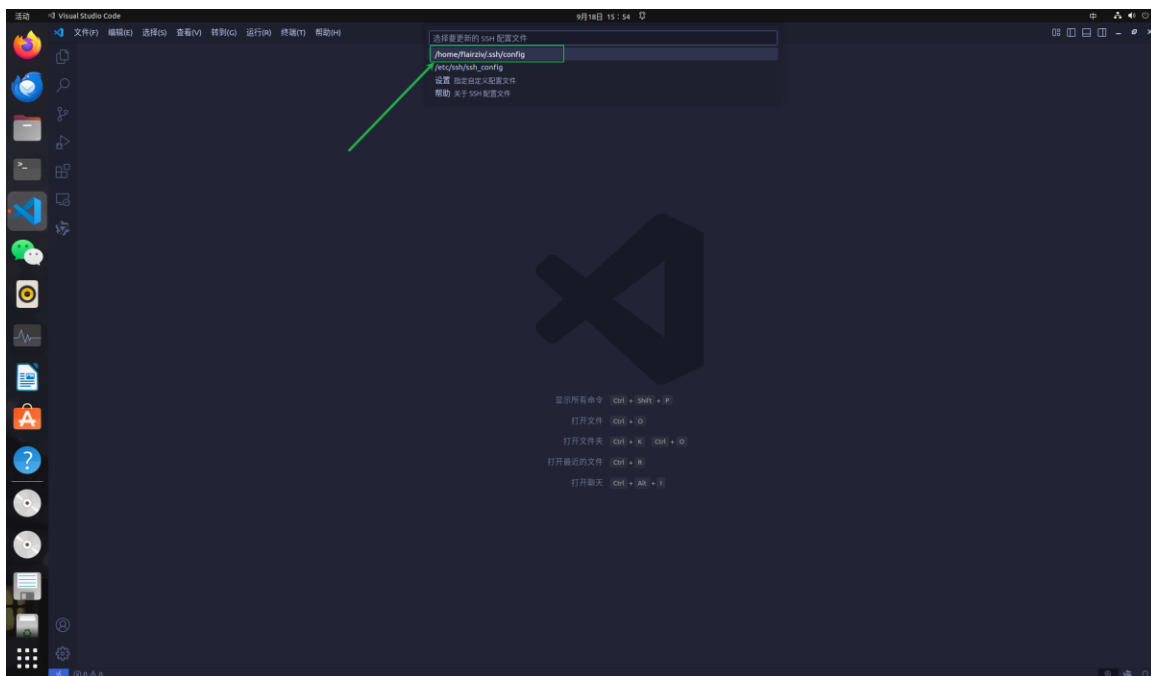




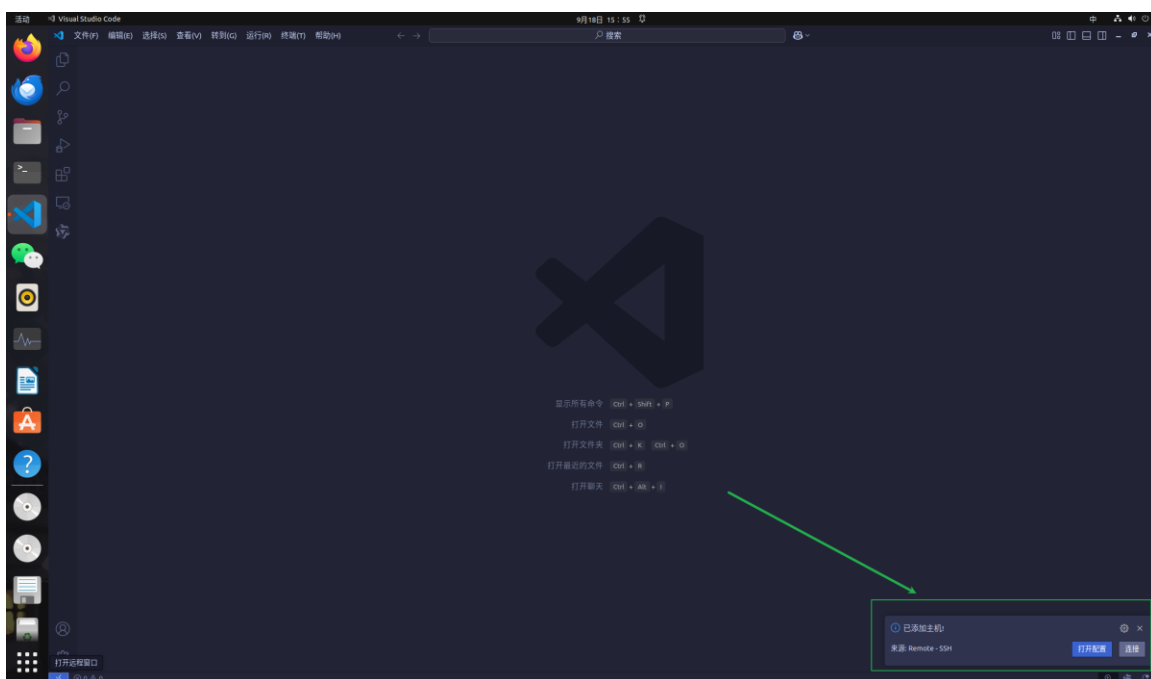
将复制的账号粘贴并且回车



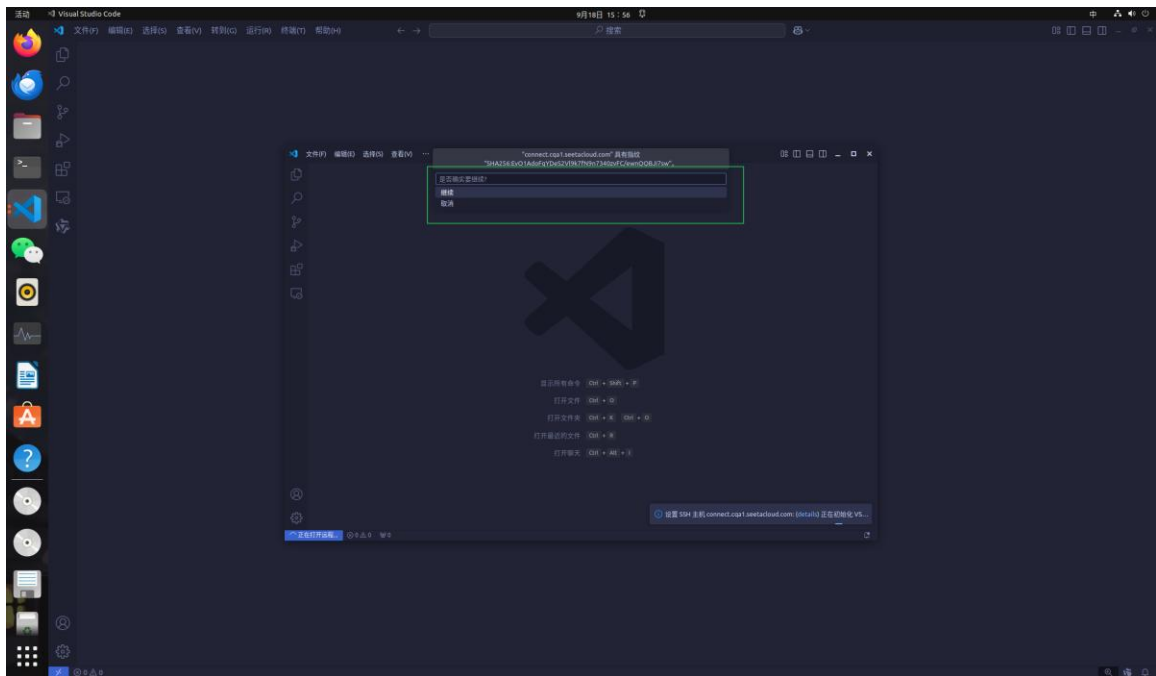
选择默认的就好



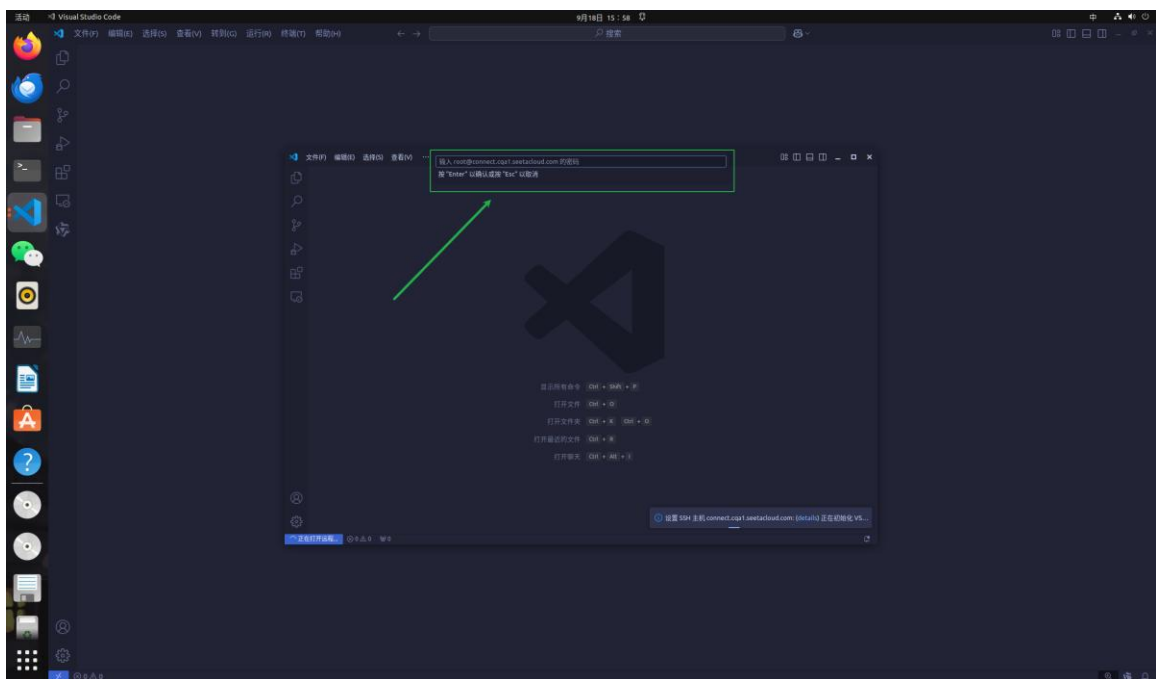
右下角会出现连接，点击连接即可



点击新窗口的继续



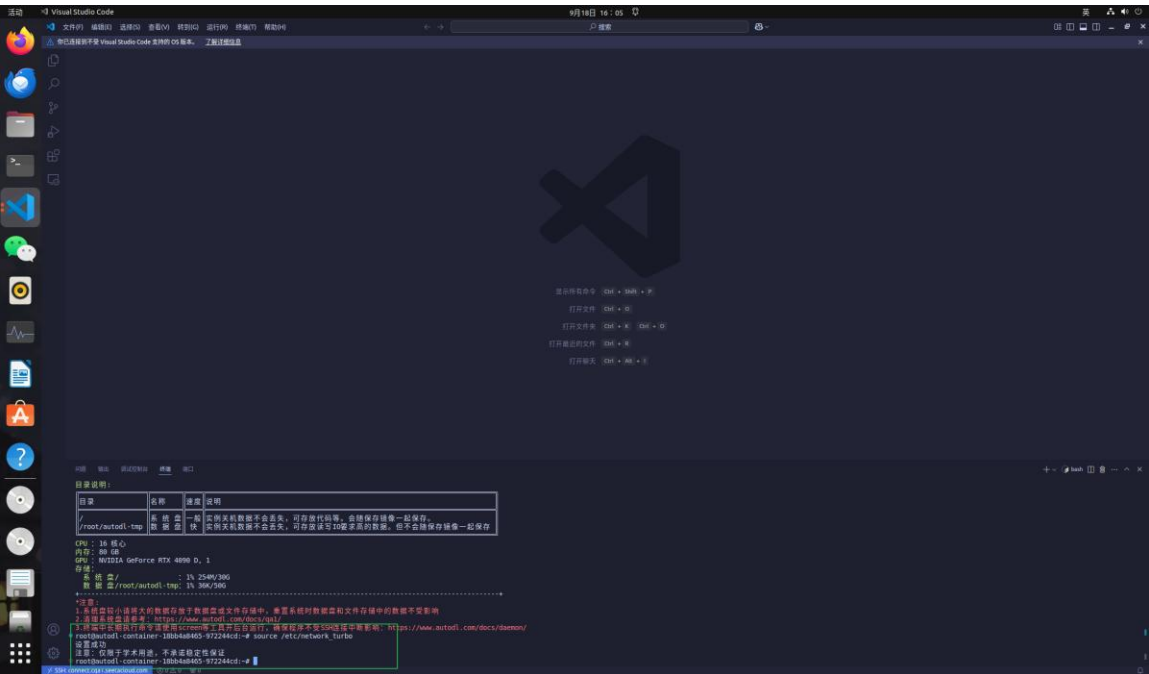
粘贴 ssh 的密码然后回车



等待就可以了

允许





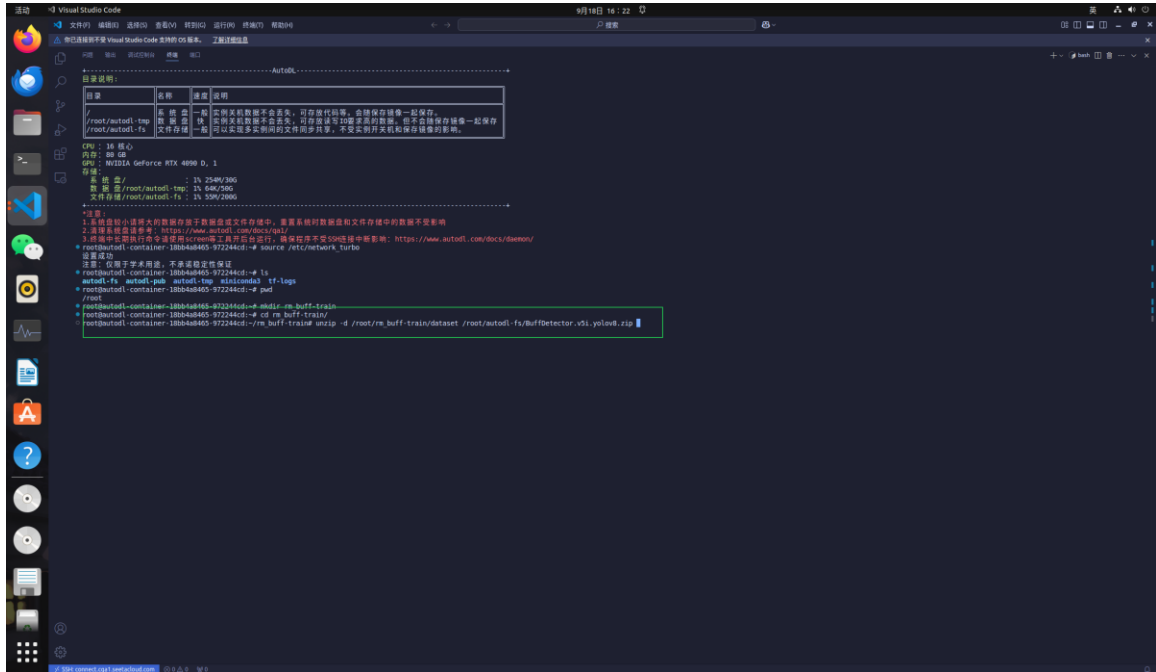
我们可以在/root/autodl-fs 这个目录找到上传的数据集



建立文件夹用来存放代码模型以及数据集

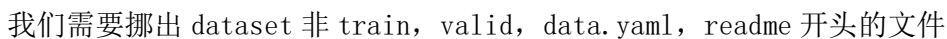
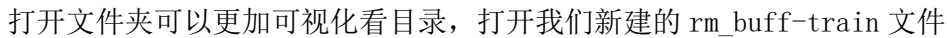
相关命令行: `mkdir rm_buff-train && cd rm_buff-train`

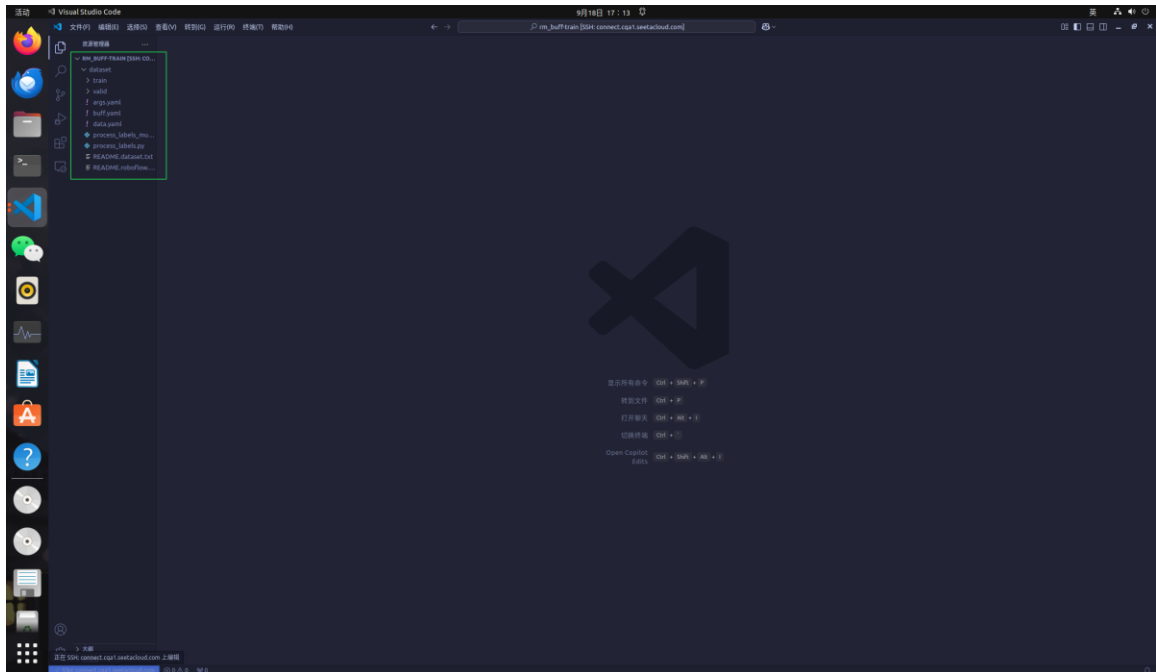
`unzip -d /root/rm_buff-train/dataset /root/autodl-fs/BufDetector.v5i.yolov8.zip`



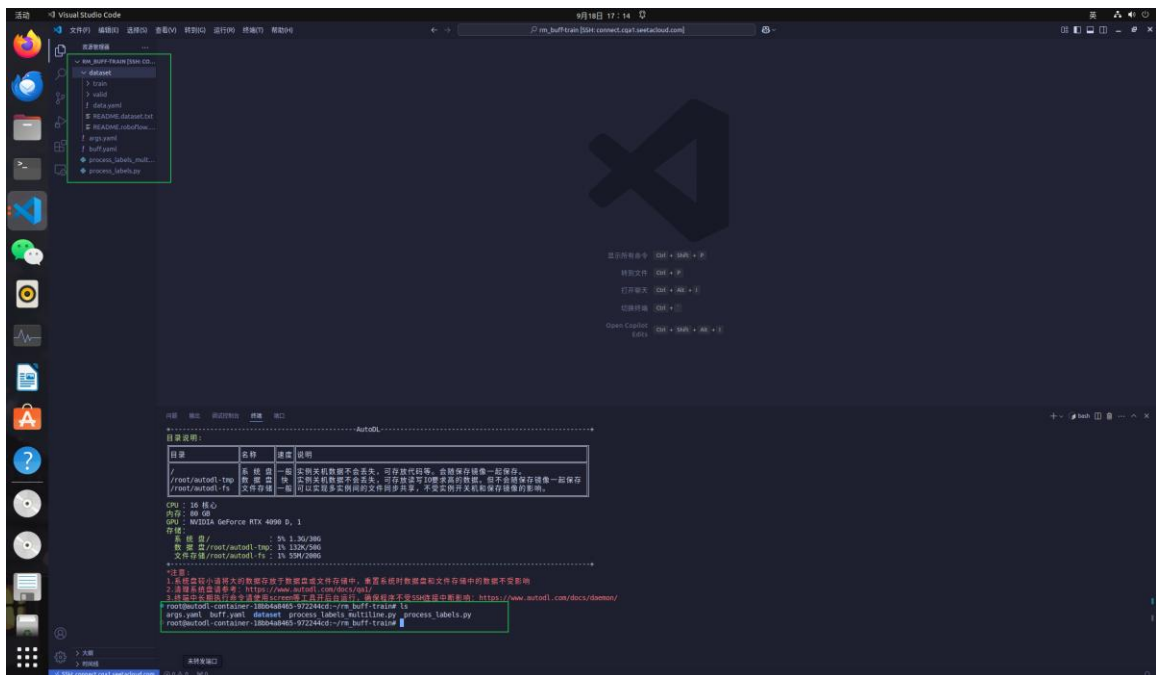
下载 yolov8 环境

下载 Ultralytics (pip install ultralytics)





项目结构跟我的一致即可



（下面将下载一个查看目录的工具 tree）

更新: `sudo apt-get update && sudo apt-get upgrade`

下载: `sudo apt-get install tree`

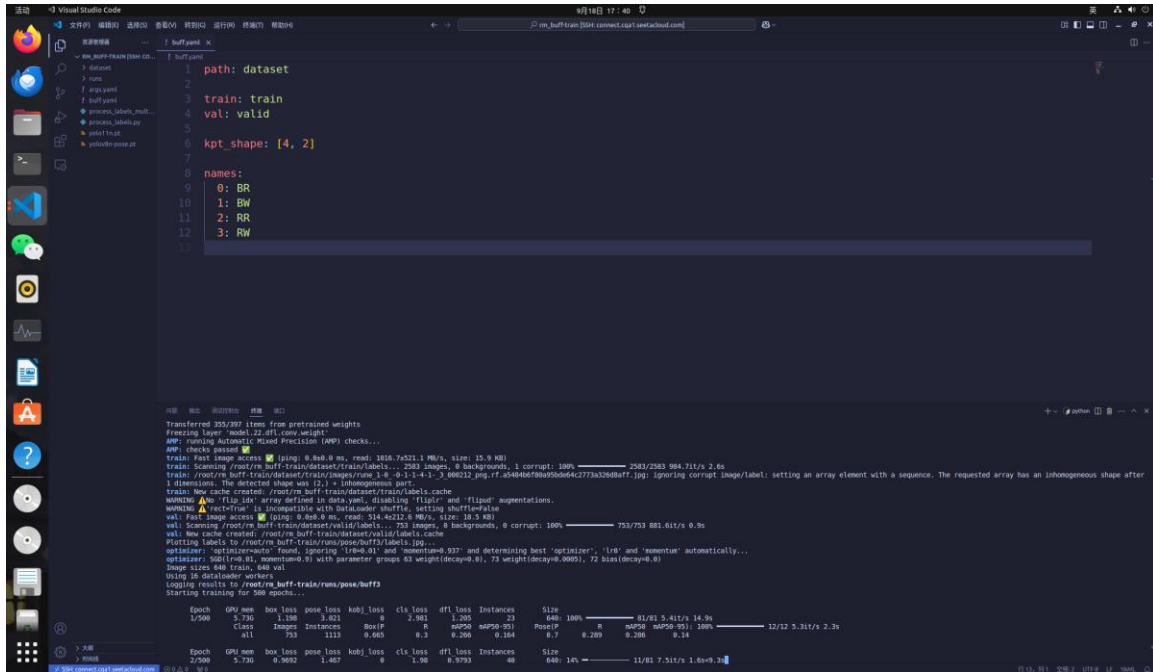




```
yolo pose train data=buff.yaml model=yolov8n-pose.pt epochs=500 batch=32  
imgsz=640 iou=0.7 max_det=10 kobj=10 rect=True name=buff workers=16
```

等待下载需要的模型，可能会警告，如果提示字体无法下载运行

```
mv /root/autodl-fs/Arial.ttf /root/.config/Ultralytics/
```



```
path: dataset  
train: train  
val: valid  
kpt_shape: [4, 2]  
names:  
0: BR  
1: BW  
2: AR  
3: RW  
Transferred 355/389 items from pretrained weights  
Freezing layer 'model.72-dfl.conv.weight'  
AMP: running Automatic Mixed Precision (AMP) checks...  
AMP checks passed  
train: Fast image access (img: 8.8e6 B/s, read: 1410.7521 MB/s, size: 15.9 KB)  
train: Scanning /root/.rm_buff-train/dataset/train/labels... 2503 images, 8 backgrounds, 1 corrupt: 100% 2503/2503 384.711/s 2.6s  
train: Scanning /root/.rm_buff-train/dataset/train/images/rune_1-8_0-1-1-4-1-3_000012.png.rf_a5484b6f8buv5b0d64c2773a326d8aff.jpg: ignoring corrupt image/label: setting an array element with a sequence. The requested array has an inhomogeneous shape after 1 dimension. The detected shape was (2,) + inhomogeneous part.  
train: Now cache created: /root/.rm_buff-train/dataset/train/labels.cache  
WARNING: The 'flip' key is defined in data.yaml, disabling 'flip' and 'flipud' augmentations.  
WARNING: 'rect=True' is incompatible with Dataloader shuffle, setting shuffle=False  
val: Fast image access (img: 8.8e6 B/s, read: 314.46312 MB/s, size: 18.9 KB)  
val: Scanning /root/.rm_buff-train/dataset/valid/labels... 753 images, 8 backgrounds, 0 corrupt: 100% 753/753 881.611/s 0.8s  
val: Now cache created: /root/.rm_buff-train/dataset/valid/labels.cache  
Plotting labels to /root/.rm_buff-train/runs/pose/buff/labels.jpg...  
optimizer: optim.AdamW found, ignoring 'lr=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr' and 'momentum' automatically...  
optimizer: SGD(lr=0.01, momentum=0.9) with parameter groups 43 weight(decay=0.0), 73 weight(decay=0.0005), 73 bias(decay=0.0)  
image sizes 640 train, 640 val  
Using 16 dataloader workers  
Logging results to /root/.rm_buff-train/runs/pose/buff3  
Starting training for 500 epochs...  


| Epoch | GPU mem | box loss | pose loss | kobj loss | loss   | cls loss | dfl loss | Instances | Size | Progress                | Time               |      |
|-------|---------|----------|-----------|-----------|--------|----------|----------|-----------|------|-------------------------|--------------------|------|
| 1/500 | 5.736   | 1.198    | 1.021     | 0         | 2.219  | 1.205    | 0.23     | 640       | 100% | 81/81 5.412/s 14.9s     | 12/12 5.311/s 2.3s |      |
|       |         | Class    | Images    | Instances | Box(P) | R        | mAP50    | mAP50-95  |      | Pose(P)                 | R                  |      |
|       |         | all      | 753       | 1113      | 0.065  | 0.3      | 0.266    | 0.184     | 0.7  | 0.289                   | 0.286              | 0.14 |
| Epoch | GPU mem | box loss | pose loss | kobj loss | loss   | cls loss | dfl loss | Instances | Size |                         |                    |      |
| 500   | 5.730   | 0.0002   | 1.487     | 0         | 1.488  | 0.0793   | 0        | 640       | 14%  | 11/81 7.511/s 1.6s@0.3s |                    |      |


```

等待训练即可

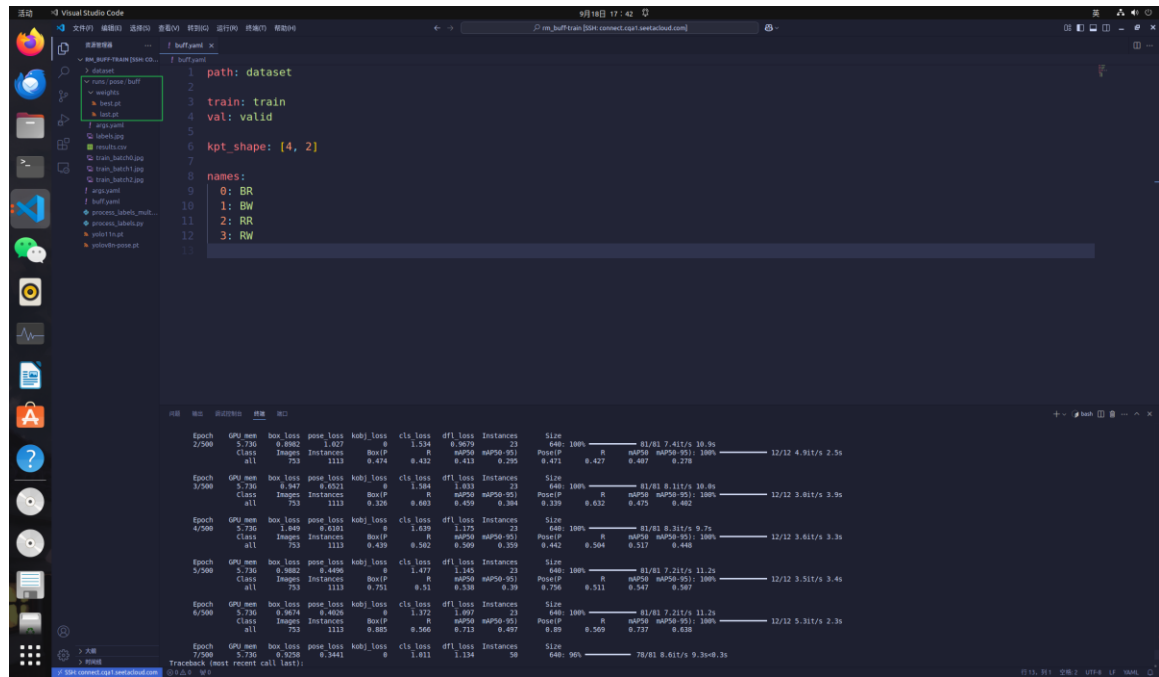
### 3. 模型训练

使用 YOLOv8 的 `pose` 模块进行关键点训练，命令如下：

```
```bash
```

```
yolo pose train data=buff.yaml model=yolov8n-pose.pt epochs=500 batch=32 imgsz=640
iou=0.7 max_det=10 kobj=10 rect=True name=buff workers=16
'''
```

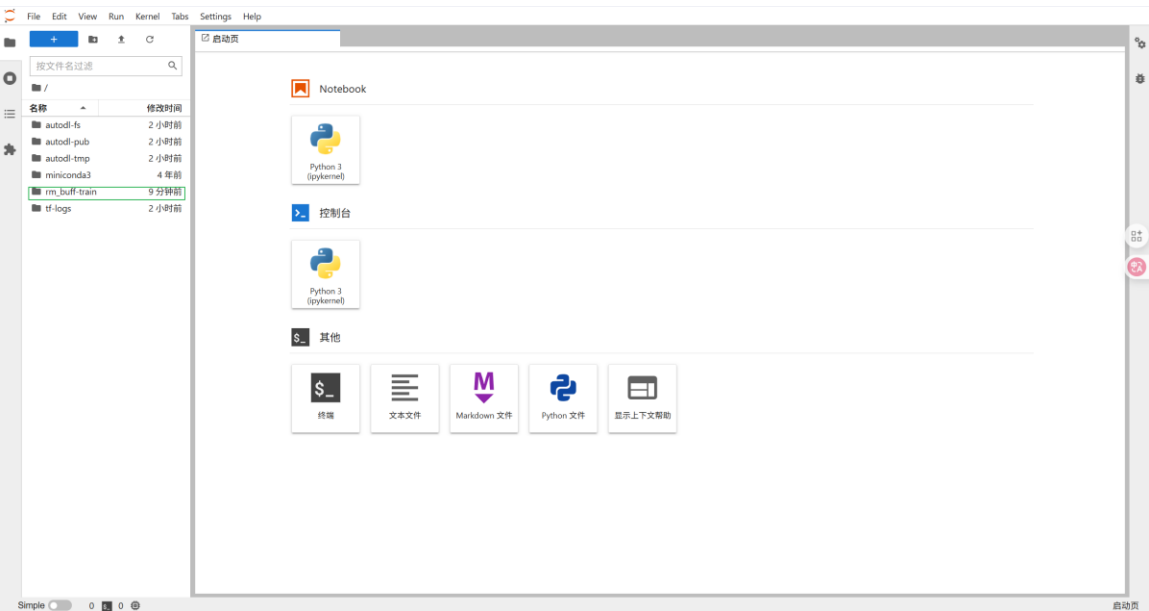
训练完成模型会保存在这里



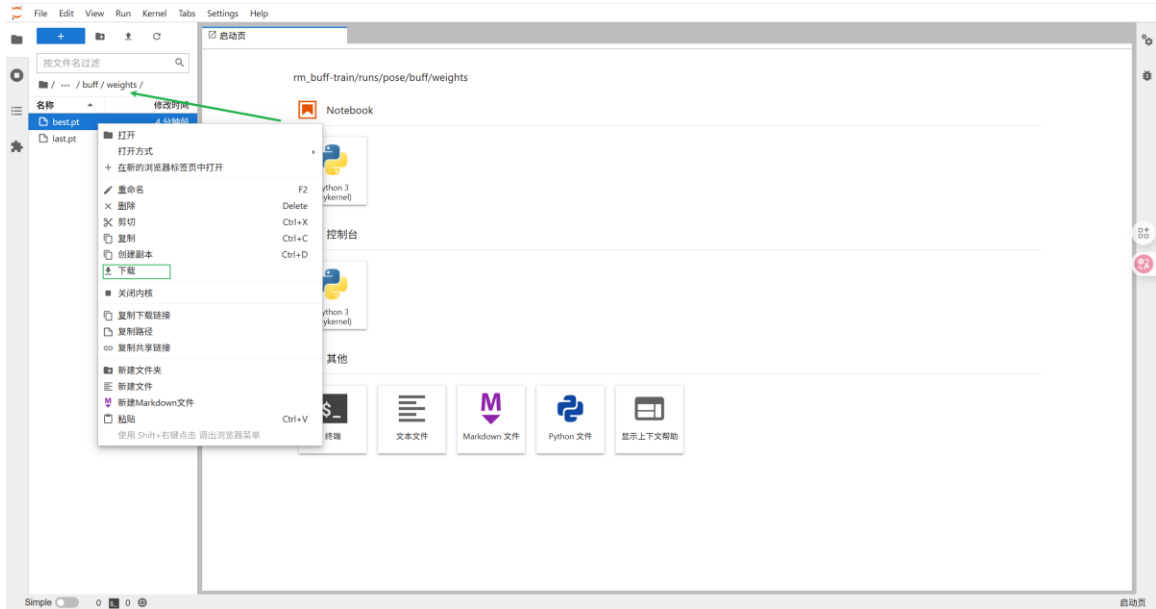
回到 AutoDL，打开 Jupyter 下载训练完的 best.pt



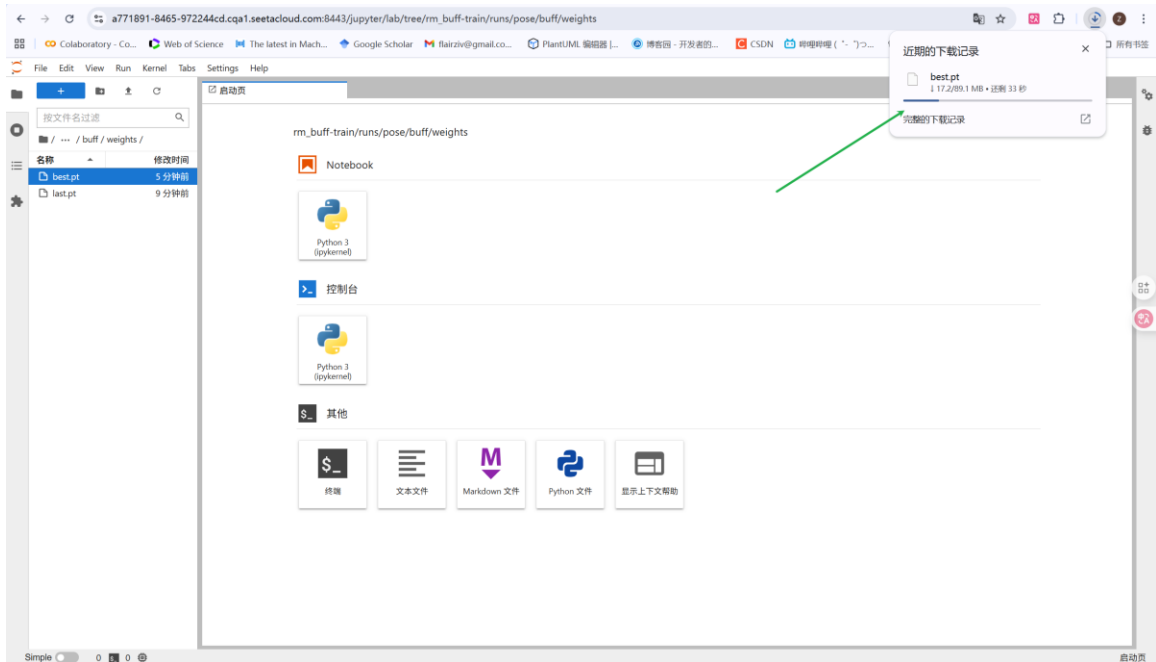
双击打开这个文件夹



找到这个目录，下载



等待下载即可



#### 4. 模型导出 : pt → onnx (这一步大家网上查)

训练完成后, 将 best.pt 导出为 ONNX 格式 :

```
``bash
yolo export model=./runs/pose/buff/weights/best.pt format=onnx dynamic=False
half=True simplify=True opset=13
``
```

#### 5. 模型转换 : onnx → OpenVINO IR (这一步大家网上查)

使用 OpenVINO Model Optimizer 工具进行模型转换 :

```
``bash
mo --input_model ./models/buff.onnx --output_dir ./models
``
```

#### 6. 总结

通过本次培训, 大家能够完成以下内容 :

1. 在 Ubuntu22.04 环境下搭建 YOLOv8 训练环境
2. 准备并转化 RoboMaster Buff 数据集
3. 使用 YOLOv8 完成关键点检测模型的训练
4. 将模型从 pt 格式导出到 onnx, 再转换为 OpenVINO IR 格式
5. 为后续部署与推理做好准备