逐飞科技
SEEKFREE

2024

MCX Vision 说明书--逐飞科技

目录

1.产品介绍

MCX Vision 是基于 NXP MCX 系列芯片做的视觉模块，搭载凌瞳彩色摄像头，30 帧采集 320*240 的彩色图像。提供源码，可以更高效的实现图像处理算法。芯片内部有 NPU，极大的提高神经网络运算速度，也就能更快运行模型

2.产品特性

1、处理器：采用 MCXN947 芯片，主频 150MHz，

2、芯片带有 512k 片内 SRAM、2M 片内 FLASH

3、芯片内部有 NPU，支持运行 TensorFlow Lite 神经网络模型

4、Type-C 接口及 SD 卡槽

5、预留专用串口接口(XH2.54 接口)；

6、预留 SWD 调试接口（带调试串口）；

7、使用 Keil 软件，C/C++语言开发环境；

3.技术参数

1、输入电压：5V

2、支持的图像格式：RGB565

3、支持的像素：QQVGA、QVGA

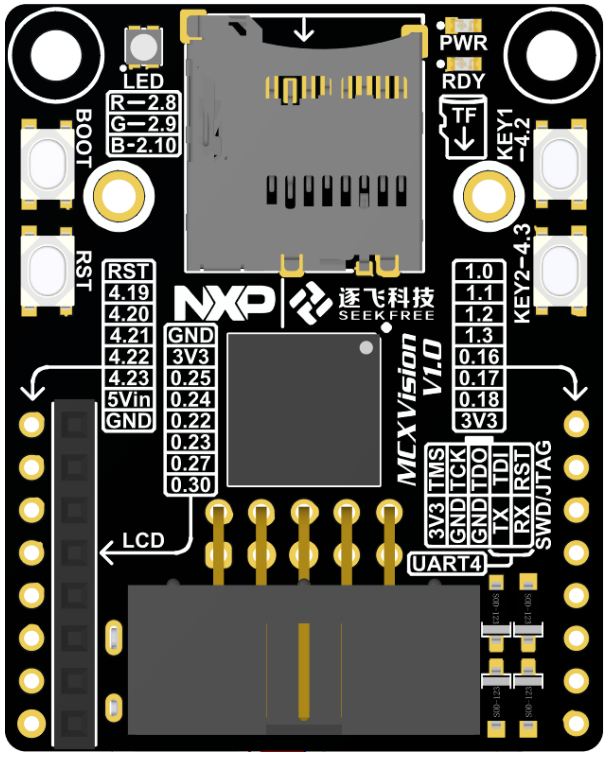
4、重量：

130°镜头：14.4 ± 0.2 g

90° 镜头：17.4 ± 0.2 g

5、功率：小于 0.5W（不带屏幕）

4.硬件说明



4.1.引脚说明

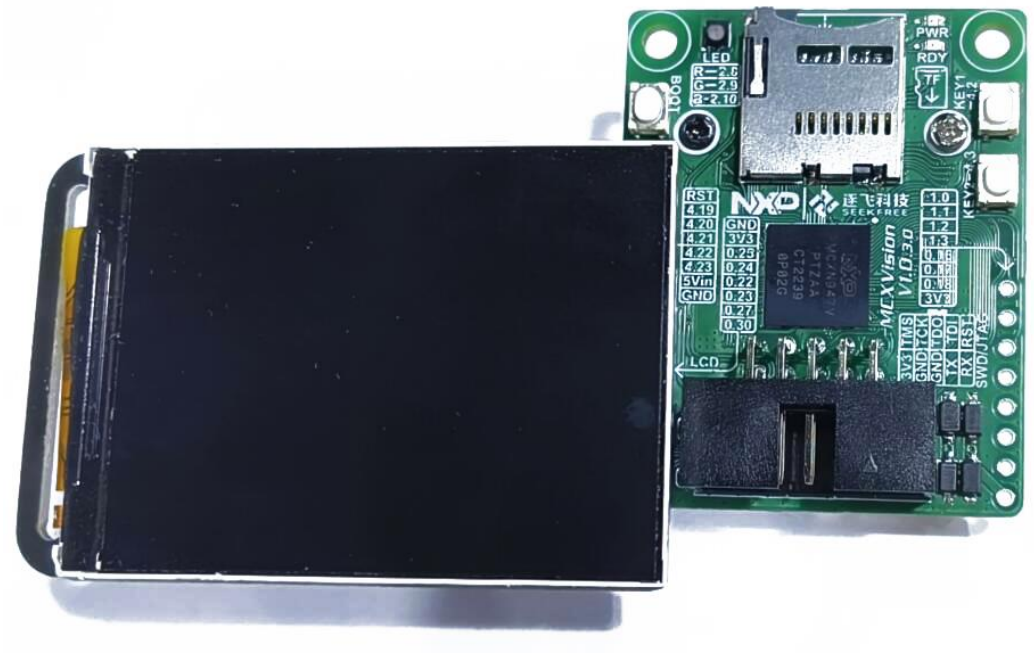
引脚名称	引脚定义	引脚名称	引脚定义
RST	Reset 引脚	1.0	FlexComm3_0/ GPIO
4.19	CT3_MAT3/ GPIO	1.1	FlexComm3_1/ GPIO

4.20	CT2_MAT0/GPIO	1.2	FlexComm3_2/ GPIO
4.21	CT2_MAT1/GPIO	1.3	FlexComm3_3/ / GPIO
4.22	CT2_MAT2/GPIO	0.16	CT0_MAT0/IIC_SDA/ GPIO
4.23	CT2_MAT3/GPIO	0.17	CT0_MAT1/IIC_SCL / GPIO
5V	5 V 电源（输入）	0.18	CT0_MAT2/GPIO
GND	电源地	3V3	3.3 V 电源（输出）

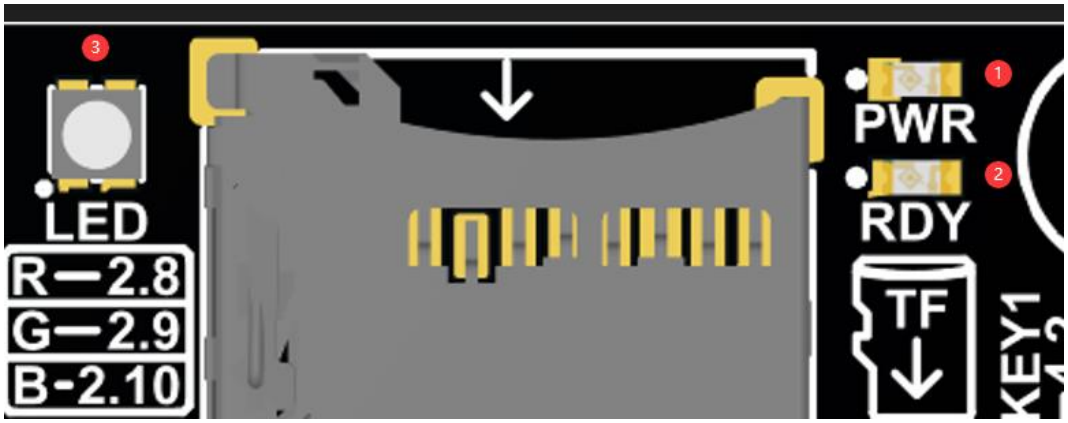
屏幕接口说明：

引脚名称	屏幕引脚定义
GND	GND
3V3	3V3
0.25	CLK
0.24	MOSI
0.22	RST
0.23	DC
0.27	CS
0.30	BLK

2 寸串口屏插到模块示意图如下：



4.2.指示灯说明



①为电源指示灯，此灯未亮，MCX Vision 可能未正确供电或者损坏；

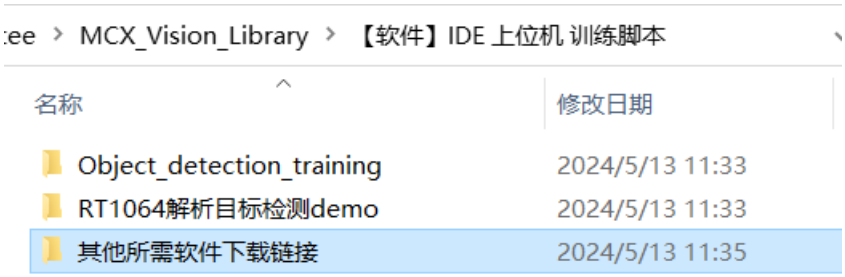
②为内核灯，上电后如果没有亮则说明内核没有启动，如果模块在运行过程中此灯闪烁则说明芯片程序异常进入 HardFault。

③为三色指示灯，三种颜色分别为红色、绿色、蓝色，分别连接芯片的三个引脚

④正面还有两个照明灯，对应芯片的一个引脚，在本文使用方式章节会给出控制方式的例程说明.

5.使用说明

此模块作为视觉模块开发板进行使用，资料中提供了多个基础示例例程，以及视觉处理例程，方便用户使用。打开资料中的其他所需软件下载链接，会进入百度网盘链接，下载 MDK，Python，eIQ 等软件。



5.1.开发环境

为避免低版本 IDE 打开高版本工程出现异常，请使用 MDK 5.38a 或更高的版本打开我们的库例程或进行二次开发。并且在使用之前，先安装芯片对应的 DFP PACK

测试无法使用 MDK 5.37 版本，会导致程序无法运行

5.2.例程介绍

1. E01_mcx_vision_opensource_library

此工程为空工程，用于写自己的程序。

2. E02_mcx_vision_led_key_demo

此例程用于板载 LED 和板载按键的使用测试，可以基于此历程做 GPIO 相关的输入输出控制。

3. E02_mcx_vision_uart_demo

此例程用于板载 DEBUG 串口和用户串口的通信测试。

4. E03_mcx_vision_sd_rw_demo

此例程用于板载 SD 卡的读写测试。

5. E05_mcx_vision_ips200_demo

此例程为测试屏幕显示，测试显示字符或全屏刷新功能

6. E06_mcx_vision_camera_qvga_demo

此例程为显示 QVGA 摄像头图像到显示屏上，可以基于此例程写摄像头处理算法。

7. E07_mcx_vision_camera_qqvga_demo

此例程为显示 QQVGA 摄像头图像到显示屏上，可以基于此例程写摄像头处理算法。

8. E08_mcx_vision_object_detection_demo

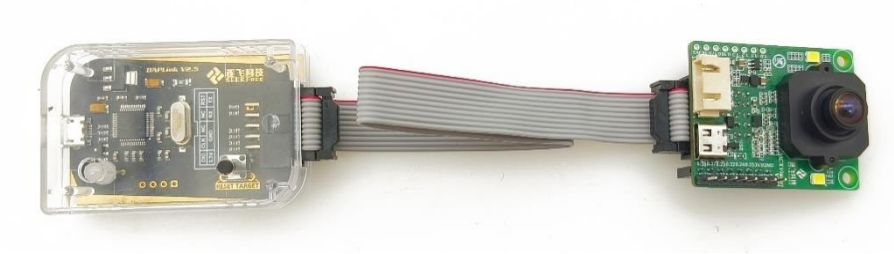
此例程为目标检测识别，并将结果通过串口发送出来。

9. E09_mcx_vision_color_trace_demo

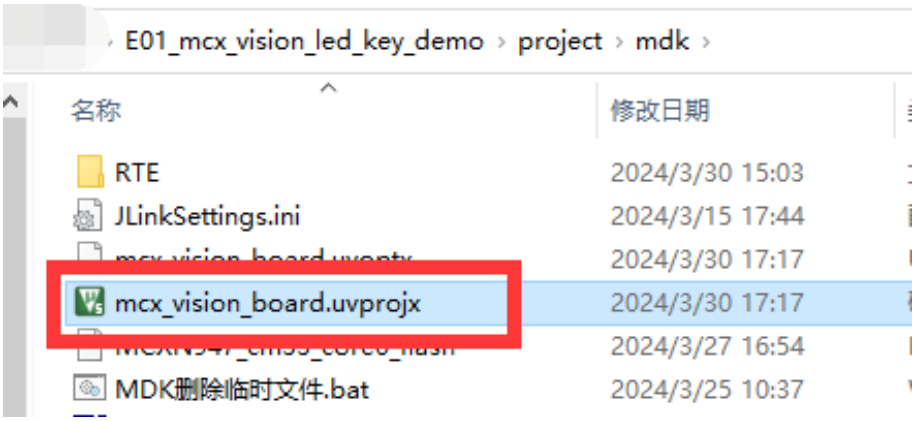
此例程为色块检测识别

5.3.例程下载

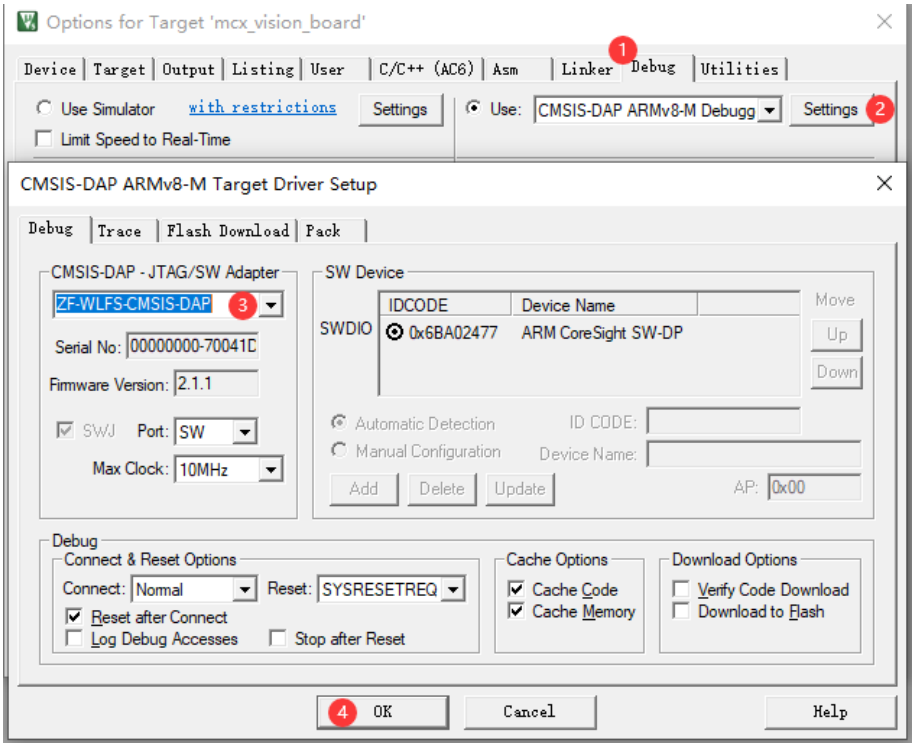
1. 使用 DAP 下载器连接 MCX Vision，并将下载器连接电脑



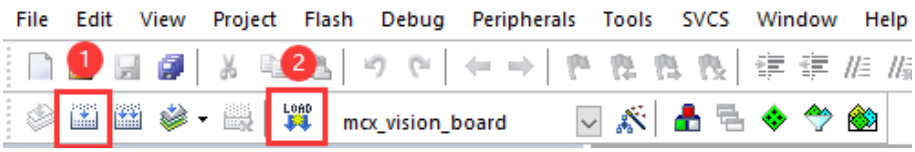
2. 打开例程中的 MDK 工程，位置如下所示：



3. 在设置中选择自己的下载器



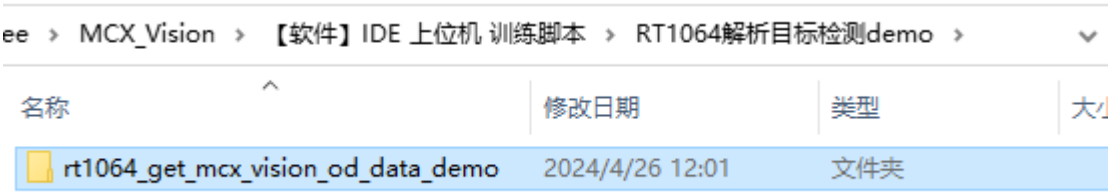
4. 右上角编译工程，然后下载



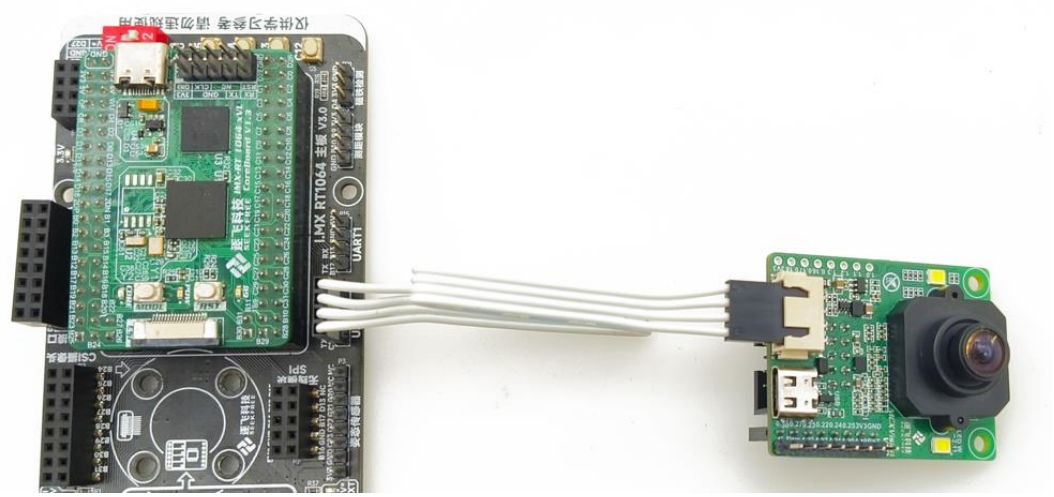
5. 下载完成后需要手动复位！

5.4.使用 RT1064 解析目标检测数据

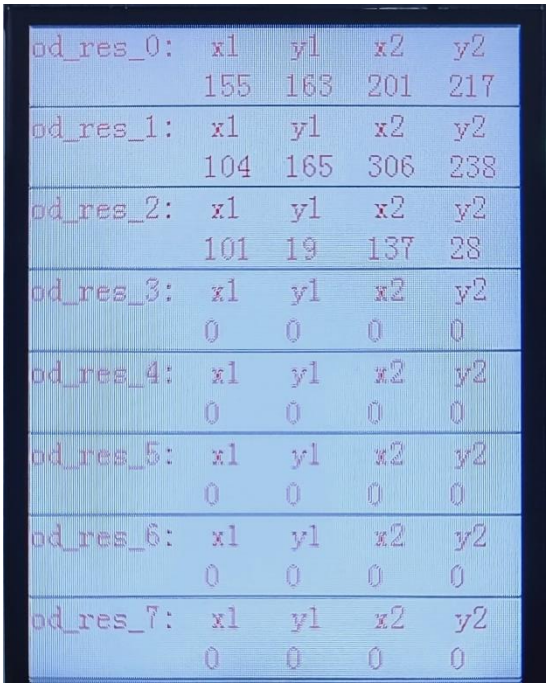
1. MCX Vision 模块下载好 E08_mcx_vision_object_detection_demo
2. RT1064 核心板下载好资料中的解析目标检测 demo



3. 将 MCX Vision 模块连接到 RT1064 主板上，MCX 的用户串口-串口 5 连接到 RT1064 的串口 4 接口上。如下图所示：



4. RT1064 插上 IPS2.0 寸屏幕，然后使用电池供电
5. 在屏幕上会显示目标检测的输出结果，包含每个目标的左上角和右下角的坐标信息，如果没有屏幕，可以打开 RT1064 的 demo 程序，进入调试查看对应变量信息。


od_res_0: x1 y1 x2 y2
155 163 201 217
od_res_1: x1 y1 x2 y2
104 165 306 238
od_res_2: x1 y1 x2 y2
101 19 137 28
od_res_3: x1 y1 x2 y2
0 0 0 0
od_res_4: x1 y1 x2 y2
0 0 0 0
od_res_5: x1 y1 x2 y2
0 0 0 0
od_res_6: x1 y1 x2 y2
0 0 0 0
od_res_7: x1 y1 x2 y2
0 0 0 0

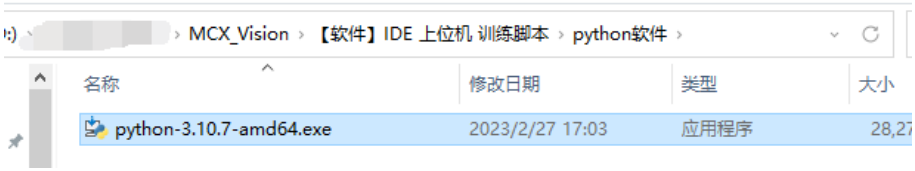
6.模型训练

本章会详细说明如何使用 MCX 的 NPU 来运行基于神经网络的目标检测模型。

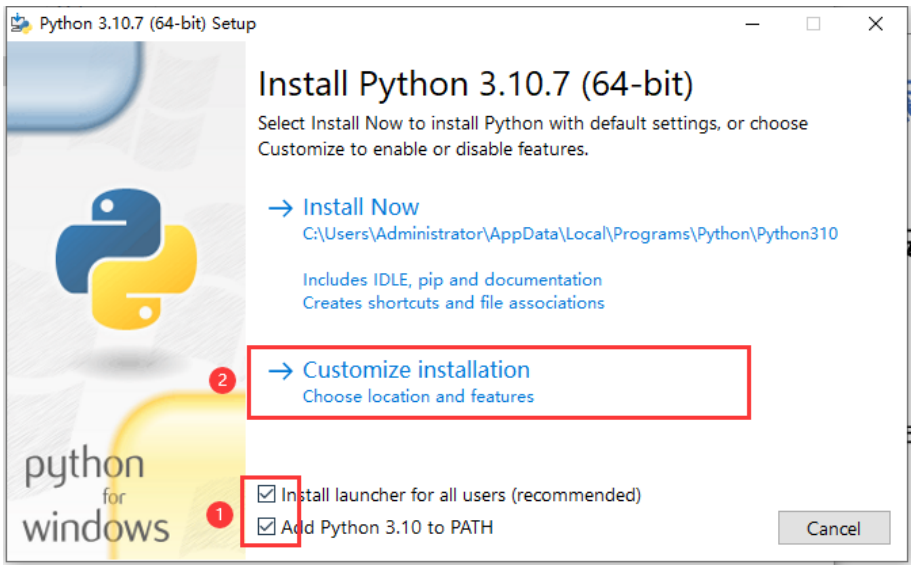
6.1.软件安装

6.1.1.Python 安装

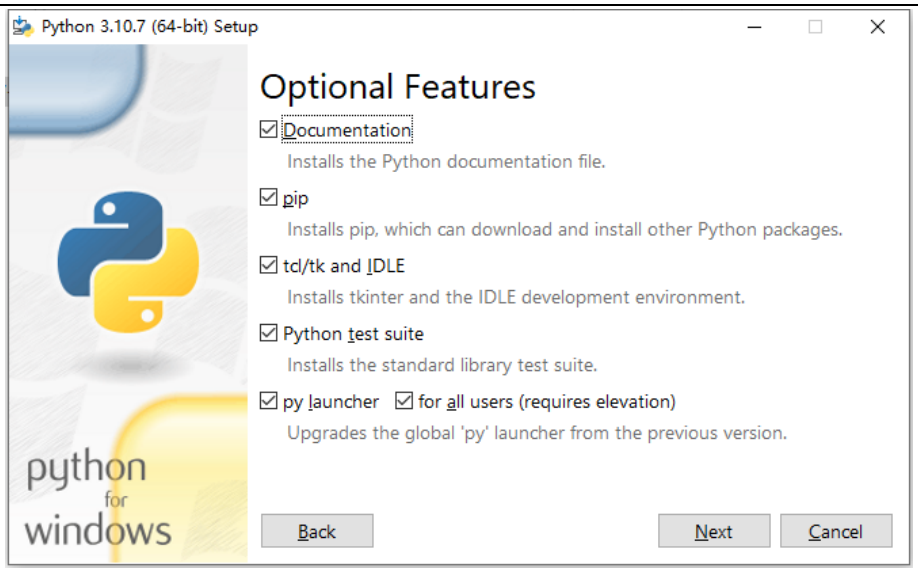
1. 打开从百度网盘下载的 python 安装包，双击进行安装，注意 python 版本必须使用提供的版本。



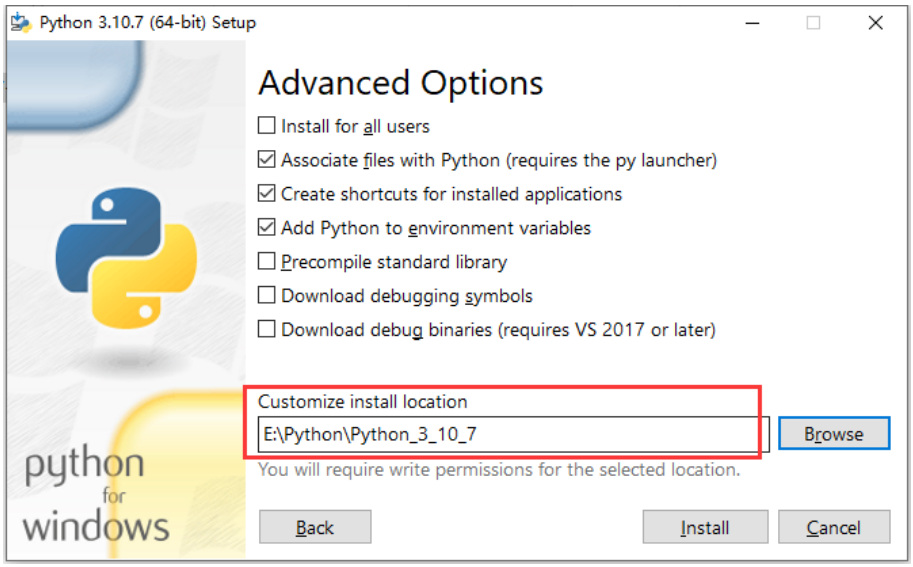
2. 勾选最下方两个框，然后点击自定义安装。



3. Next。

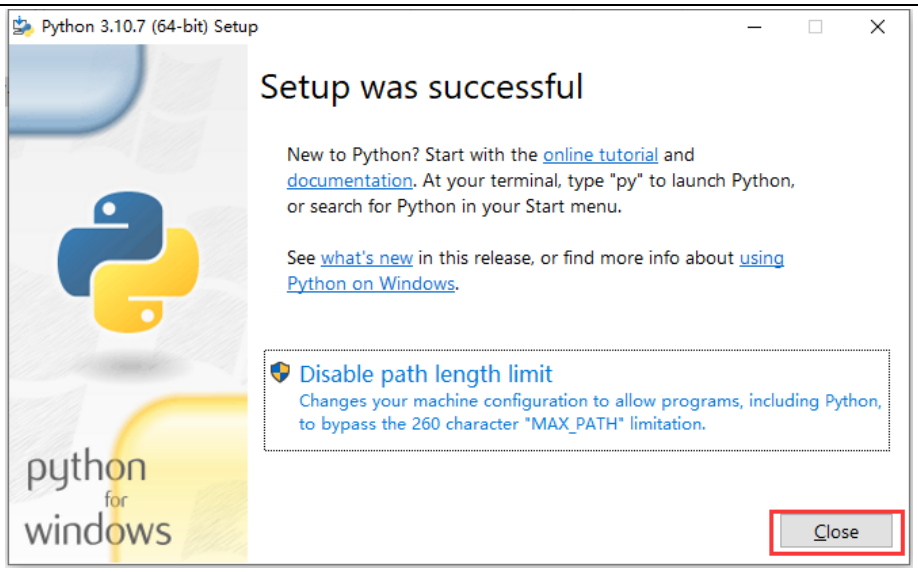


4. 修改路径，点击 Install。

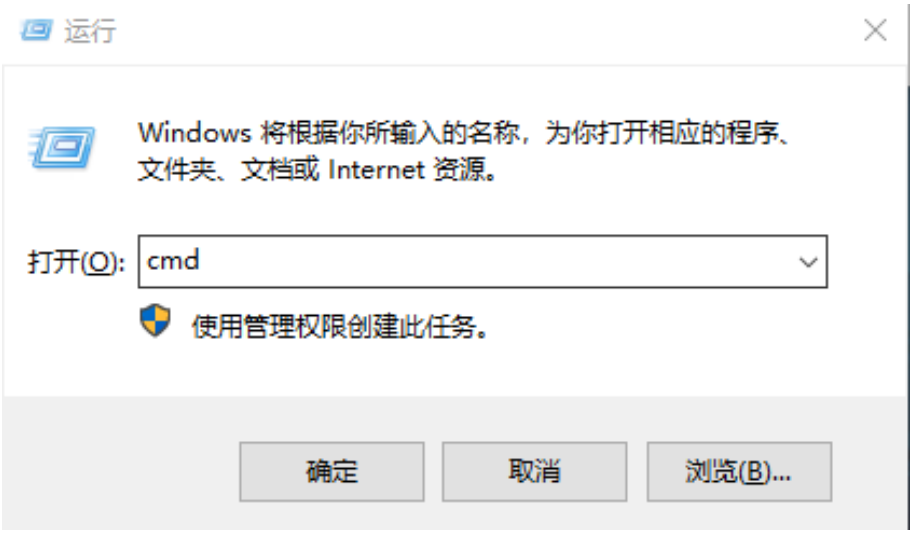


5. 等待安装。

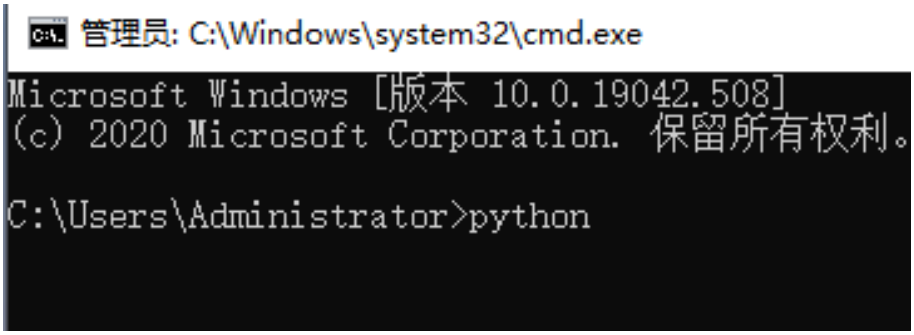
6. 安装完成后点击 Close。



7. 判断是否安装完成，win+r 打开运行窗口，输入 cmd 回车。



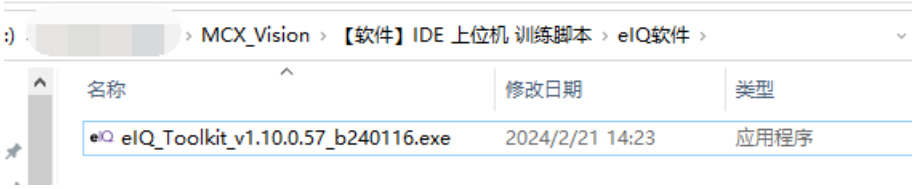
8. 输入 python，回车。



9. 如果看到显示 Python 3.10.7 则安装完成。

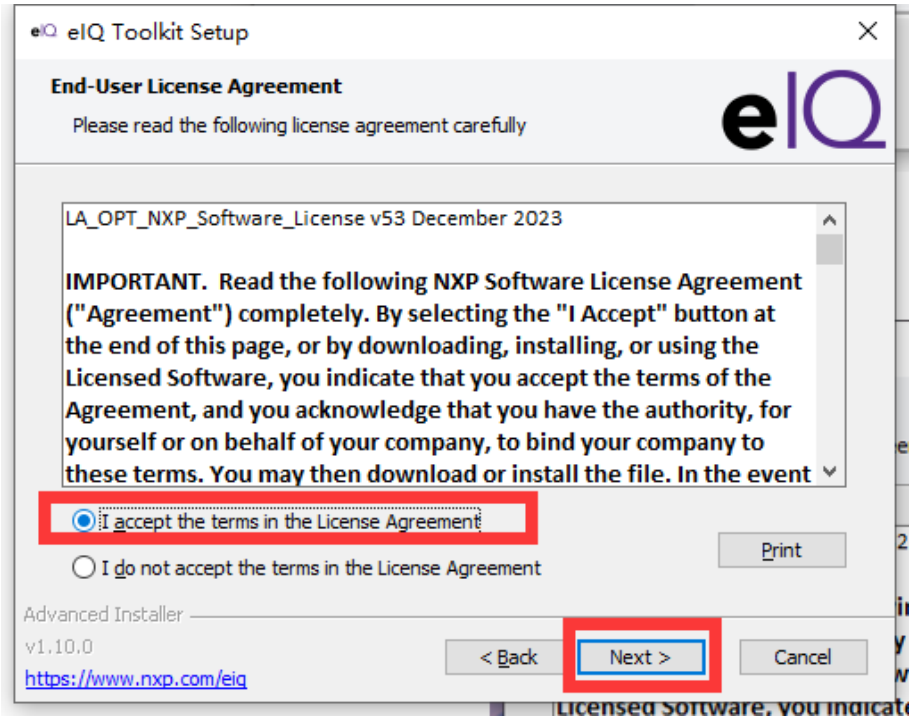
6.1.2.eIQ 安装

1. 打开从百度网盘下载的 eIQ 安装包，双击进行安装

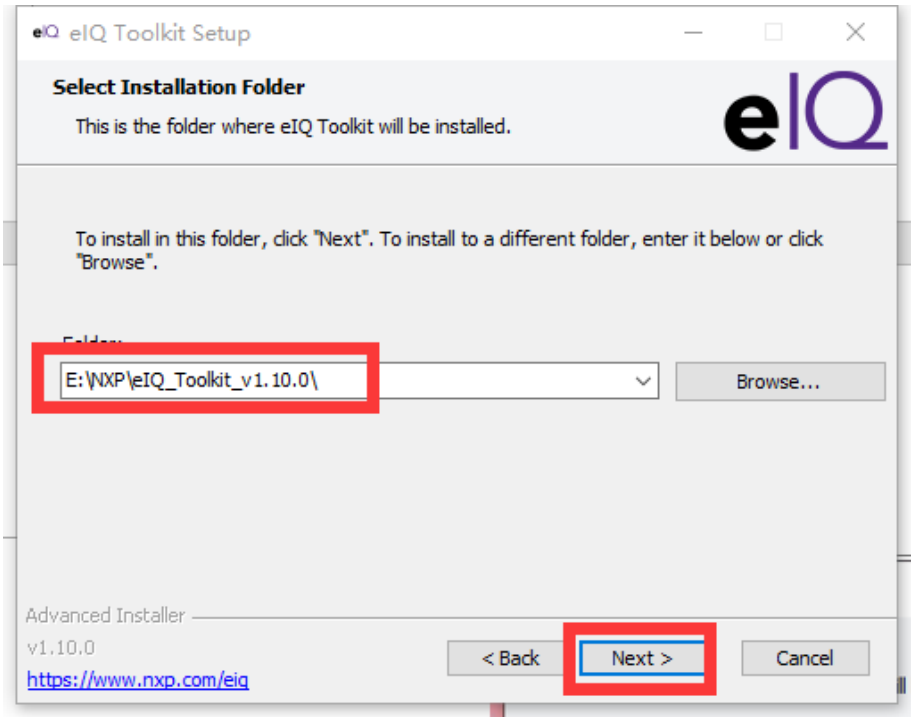


2. Next。

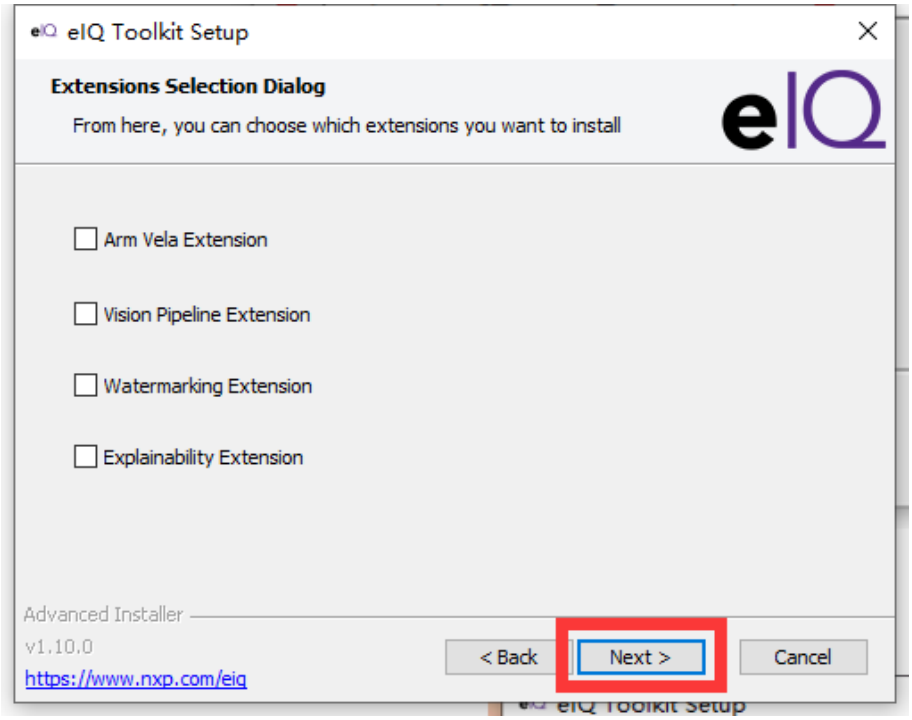
3. 选择 I accept，然后 Next。



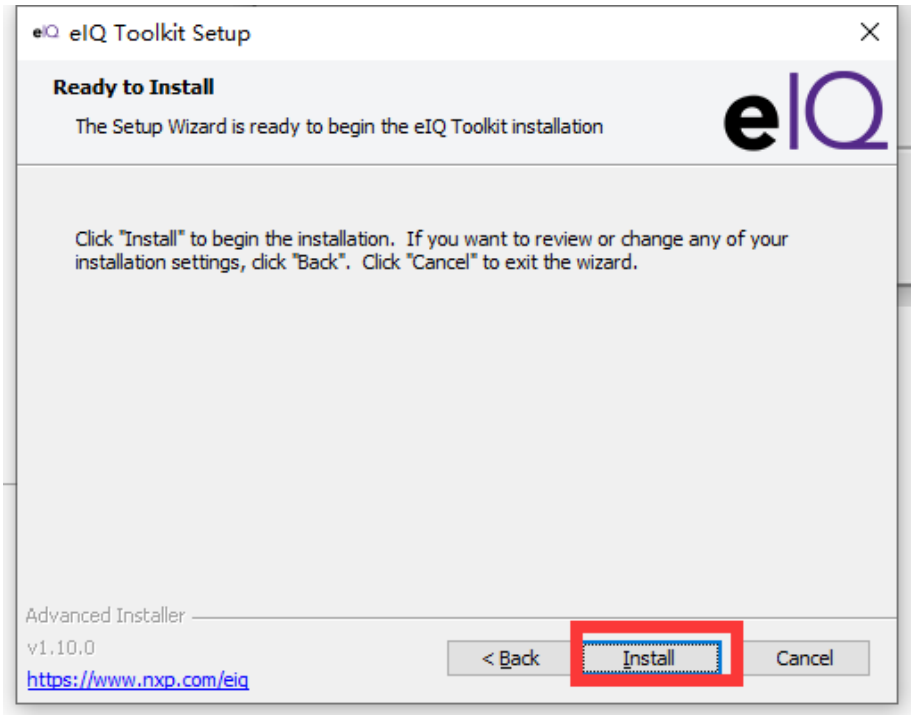
4. 设置路径（需要记住这个路径），Next，等待安装完成。



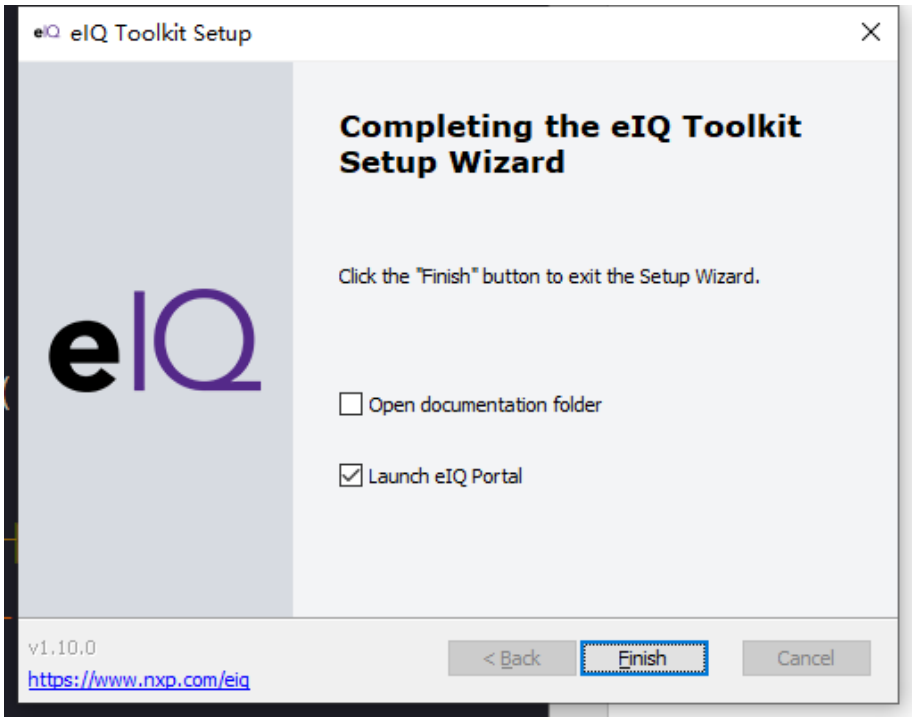
5. NEXT。



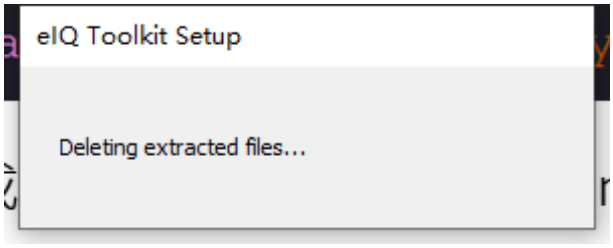
6. Install。



7. Finish。



8. 后续 eIQ 会自动删除安装提取的文件，不用管



6.2.图片获取

图片获取推荐使用 OpenART mini 模块来拍摄图片。OpenART mini 拍摄整张图片的示例程序如下：

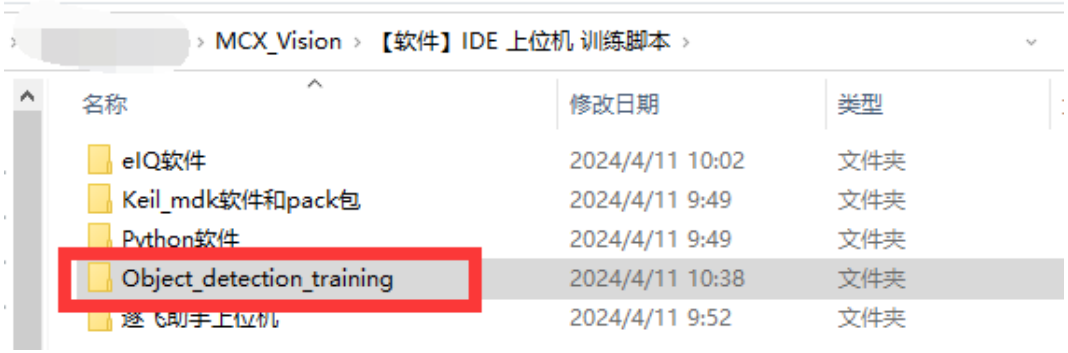
```
from machine import UART
import pyb
import sensor, image, time, math, tf
import os


sensor.reset()
sensor.set_pixformat(sensor.RGB565)      # 设置摄像头像素格式
sensor.set_framesize(sensor.QQVGA)       # 设置摄像头分辨率
sensor.set_brightness(1000)              # 设置摄像头亮度 越大越亮
sensor.set_auto_whitebal(True)
sensor.skip_frames(time = 20)

clock = time.clock()

save_img_num = 0;
while(True):
    img = sensor.snapshot()              # 获取一幅图像
    # 修改文件名称，准备保存
    save_img_num += 1;
    image_pat = "/sd/"+str(save_img_num)+".jpg"

    print(image_pat)
    # 将拷贝之后的图像保存到sd卡
    img.save(image_pat,quality=99)
```

拍摄完成后需要复位 OpenART mini 模块，并且重新打开 U 盘文件夹，或者刷新文件夹内容就能看到拍摄的图像。

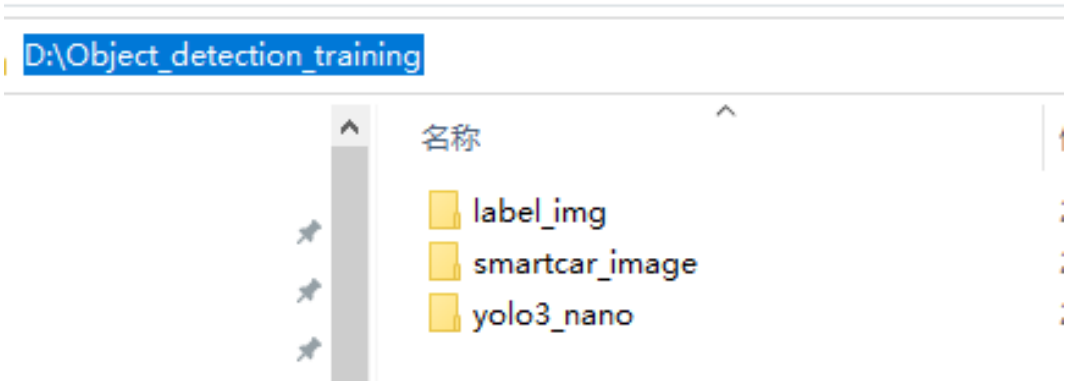
如果没有 OpenART mini 模块，使用其他的设备拍摄照片也可以。

6.3.图片标记

1. 移动目标检测的文件夹到全英文路径下



移动位置如下图所示，只要是全英文就行。



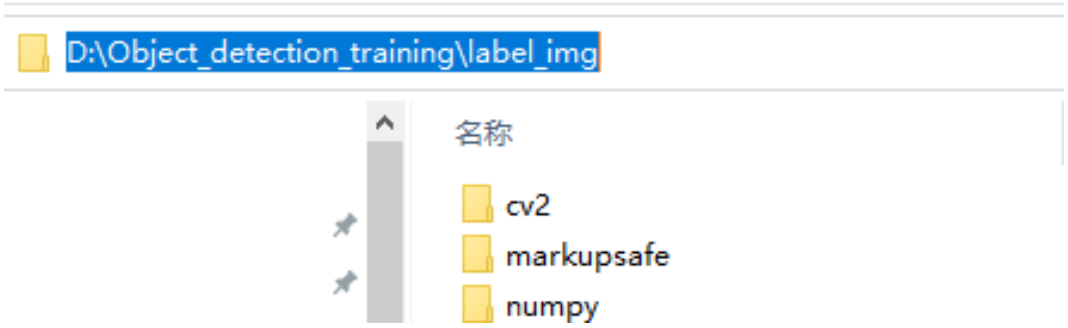
在文件夹中包含三个文件夹：

smartcar_image 为已标记好的数据集（用于测试，实际情况还需要自行拍摄和标记）。

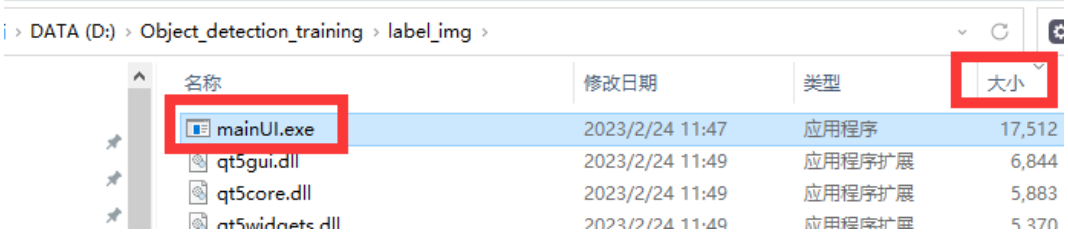
label_img 为标记图片的工具。

yolo3_nano 为模型训练和导出的脚本。

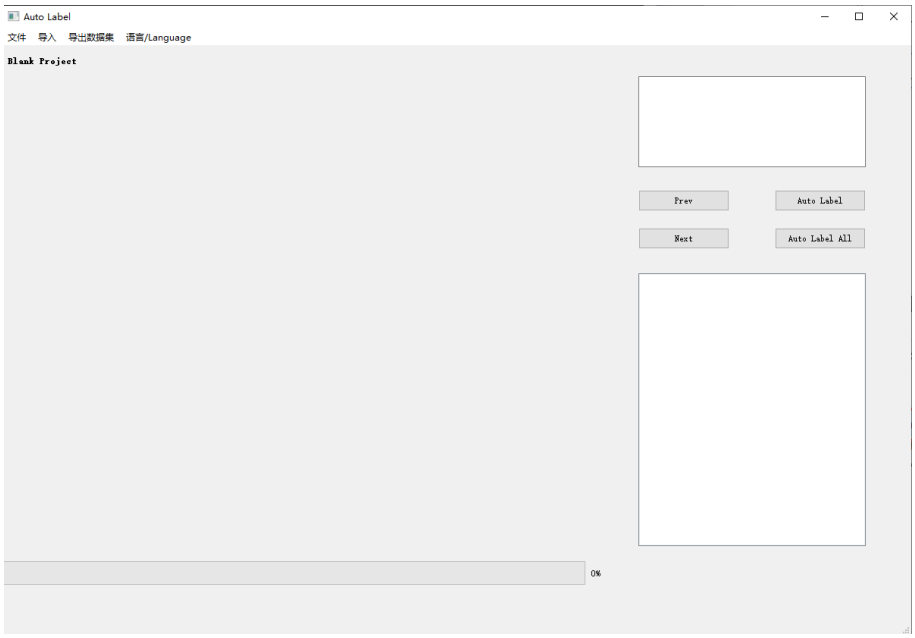
2. 进入 Object_detection_training 的 label_img 文件夹



3. 按照大小排序，找到 mainUI.exe 应用，双击运行。

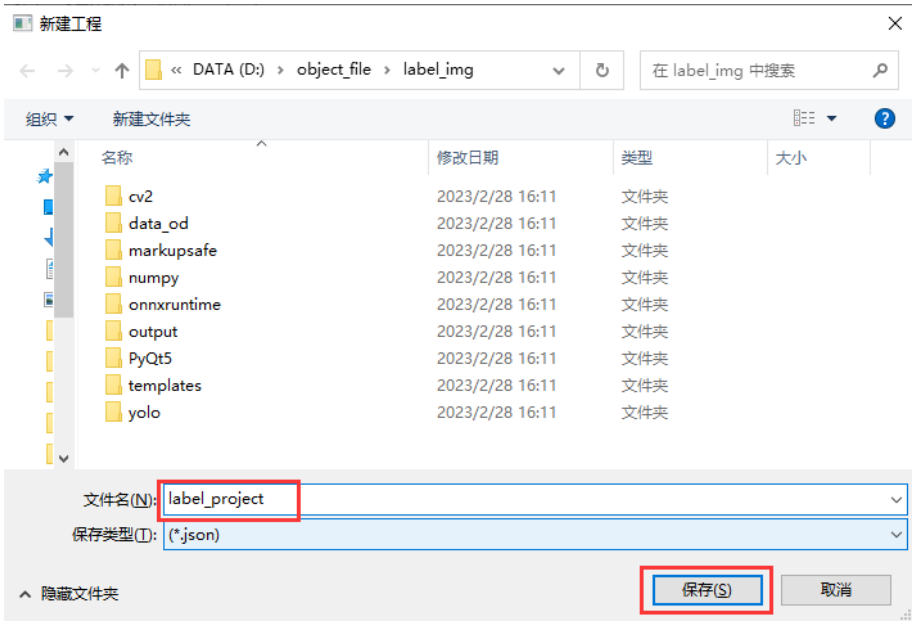


主页面如下图所示：



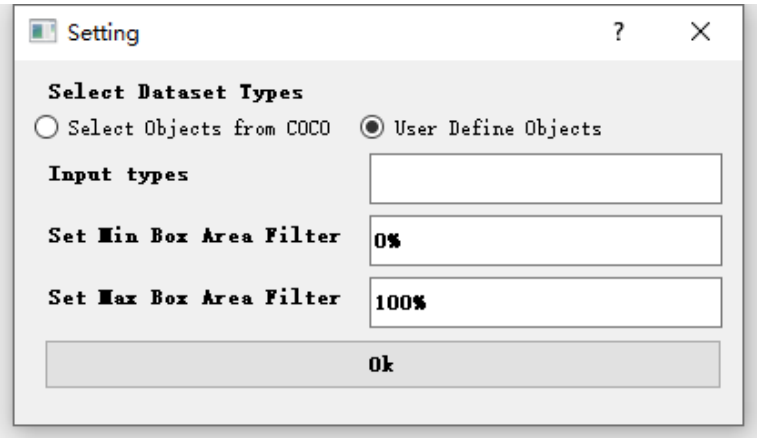
4. 在软件中点击左上角文件，选择新建工程。

5. 输入工程名，然后保存。

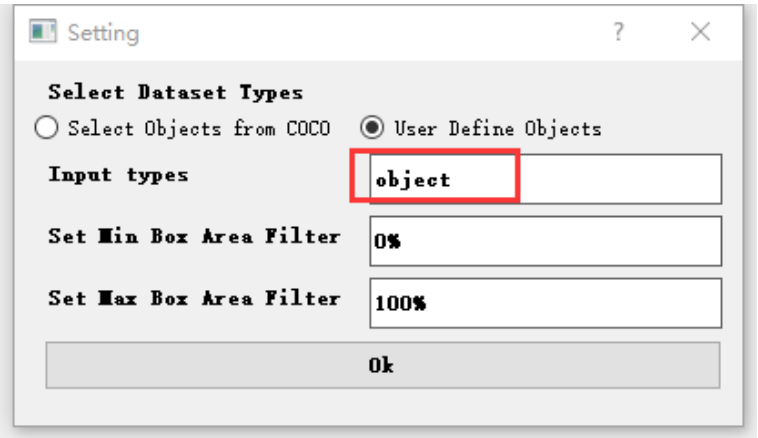


6. 保存后，会弹出一个设置窗口。

7. 这里选择右侧选项，User Define Objects。

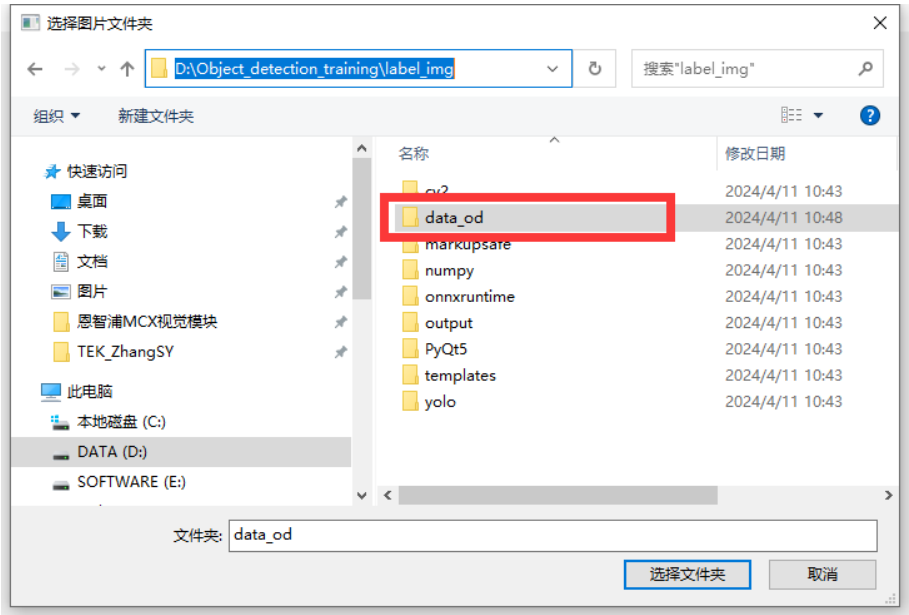


8. 输入目标类名，这里只能输入 object，否则后面训练会报错!!!其他默认，然后点击 Ok。

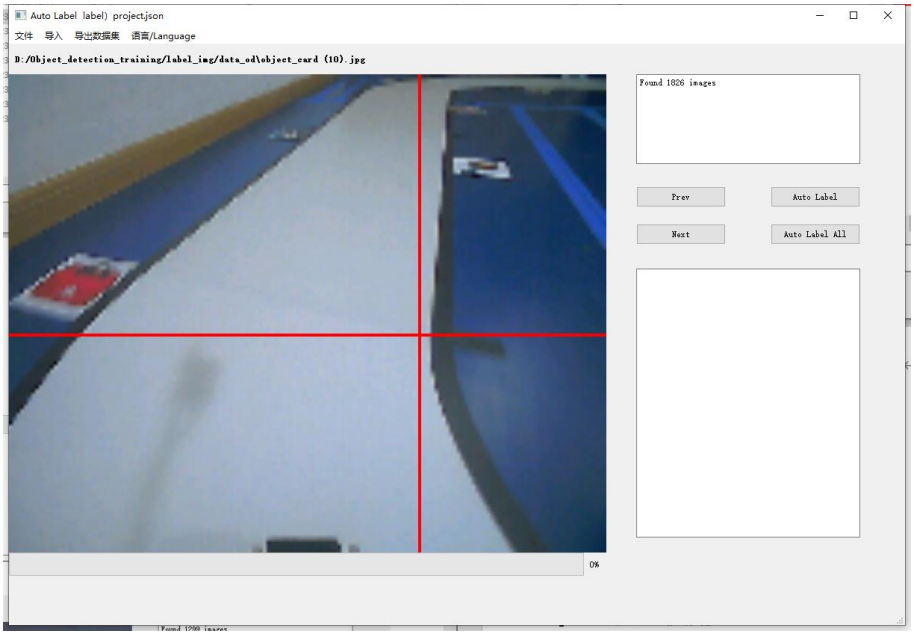


9. 在软件中点击左上角导入，选择打开图片文件夹。

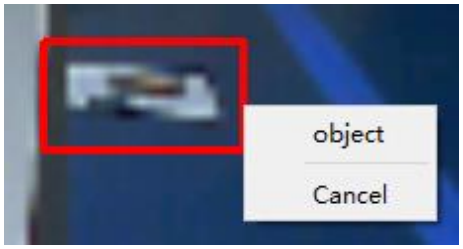
10. 选择软件目录下的 data_od 文件夹，这个文件夹是我们使用 OpenART mini 拍摄的一些实际场地图片，用于本次测试，实际还需要同学们自己拍摄。



11. 可以看到图片已经导入软件中。



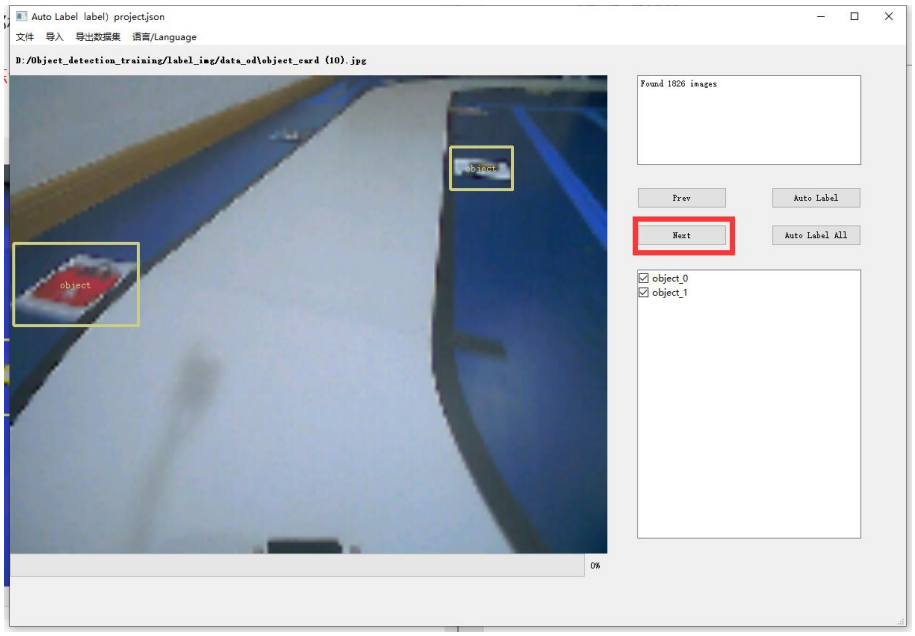
12. 使用鼠标左键来框选图片位置，先鼠标左键点击图片左上角位置，然后不松开左键拖动到图片右下角，松开鼠标左键，然后弹出类别提示。



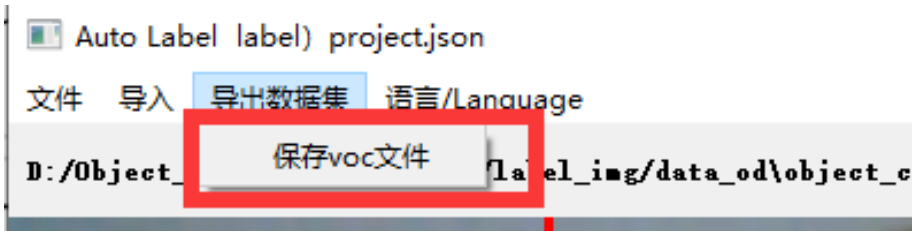
13. 选择 object，标记完成图片中第一个目标

14. 然后需要将图片中的较为突出的目标都标记上，随后点击下一张图片进行标记。

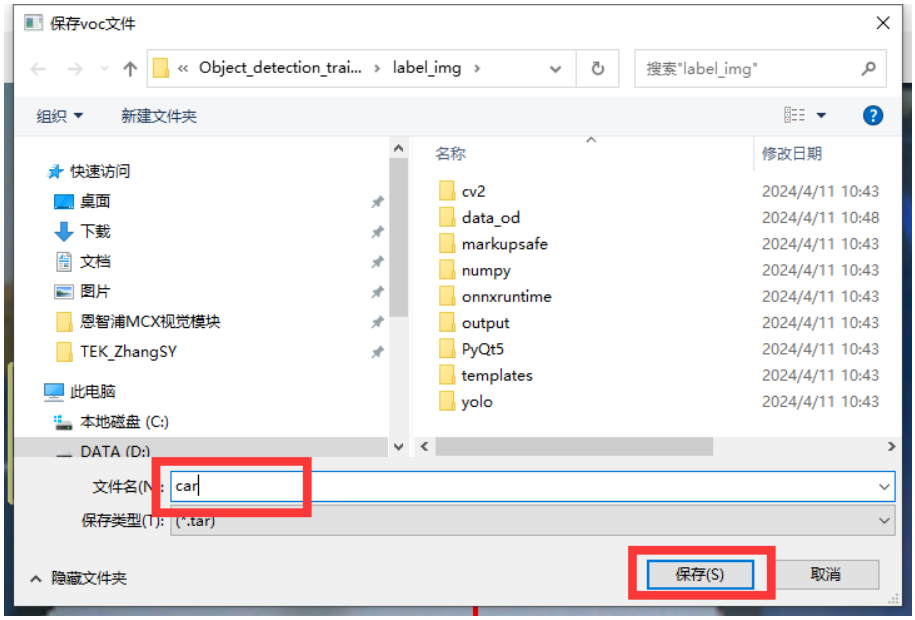
注意：建议标记 1000 张，否则可能训练失败！



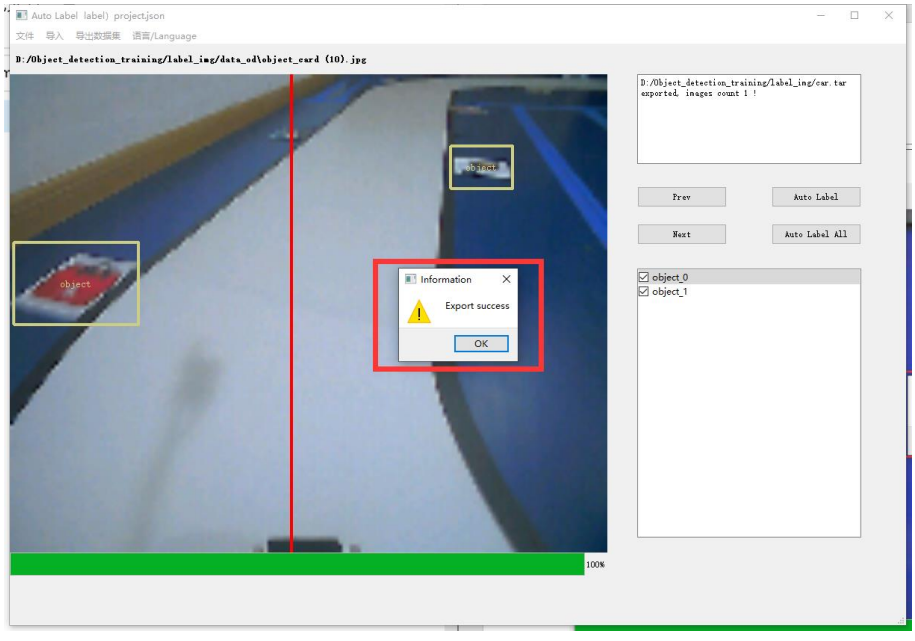
15. 图片标记完成后，点击右上角导出数据集，选择保存 voc 文件。



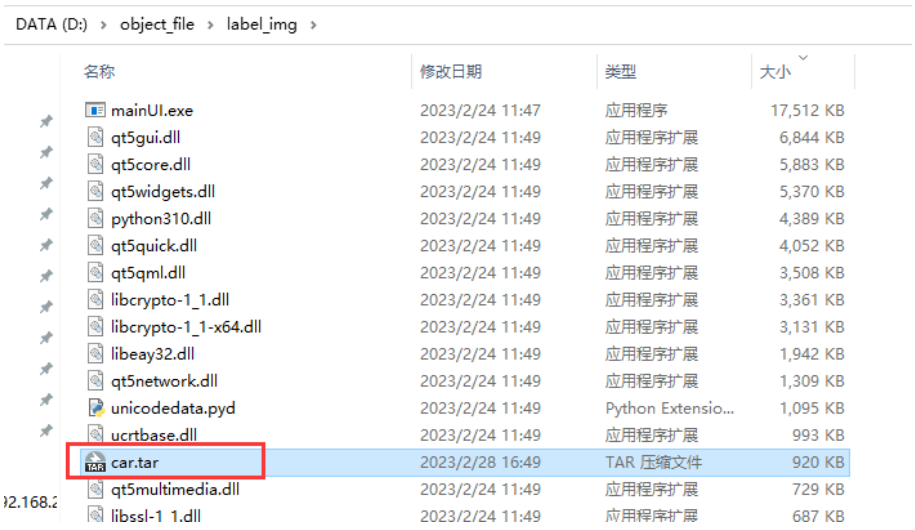
16. 输入文件名，然后保存，保存的是一个压缩包。



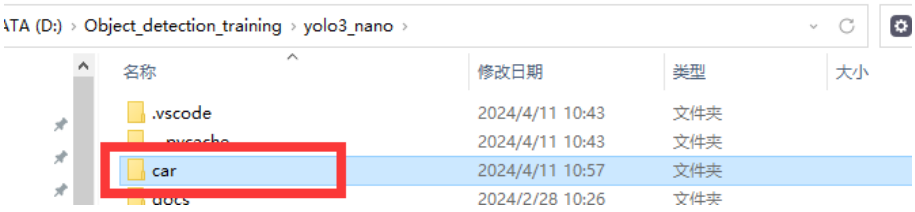
17. 等待保存完成，点击 OK 即可。



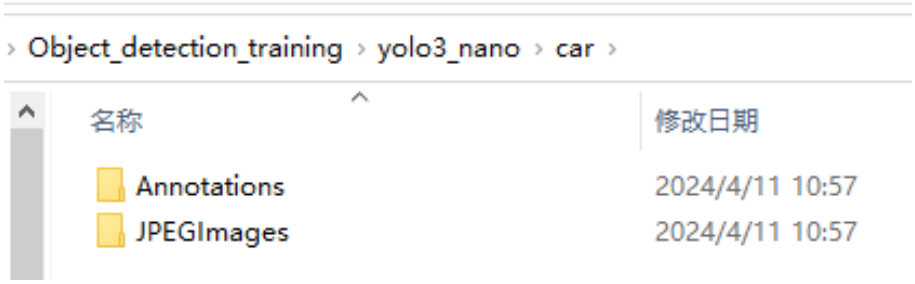
18. 我们可以在目录下看到这个压缩包。



19. 将这个压缩包解压到 Object_detection_training\yolo3_nano 文件夹中。

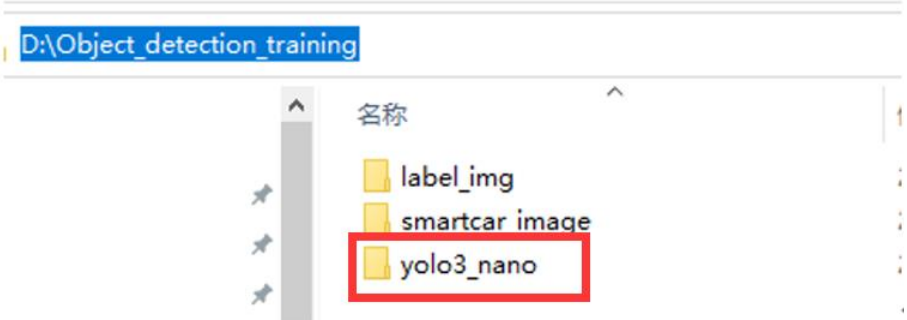


20. 解压后的文件夹中就包含原图像和标记文件。



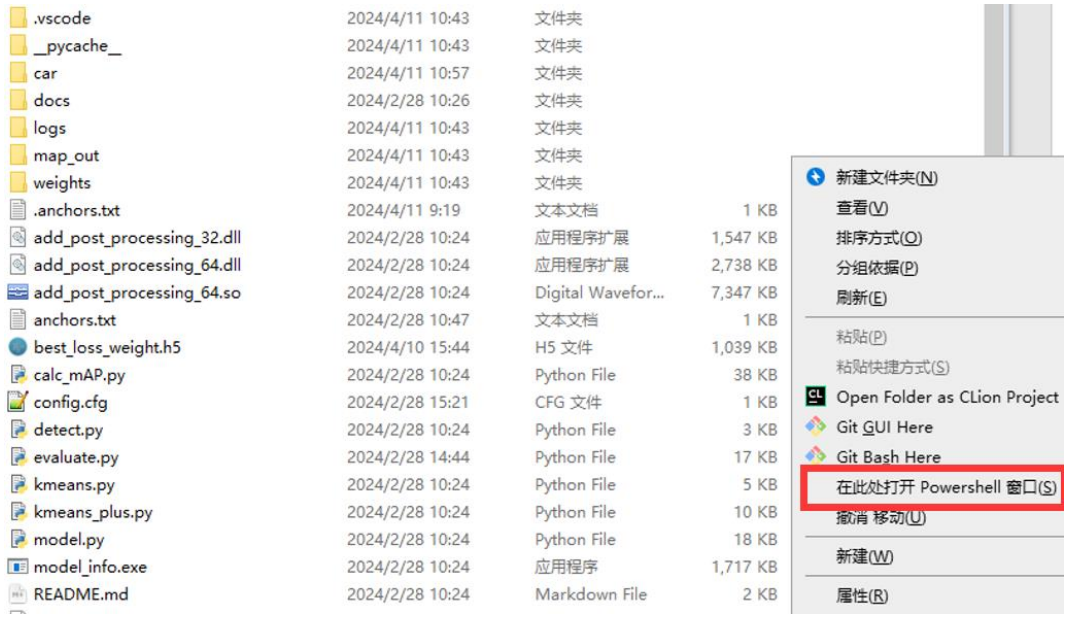
6.4.模型训练

1. 需要先将文件夹移动到全英文路径。



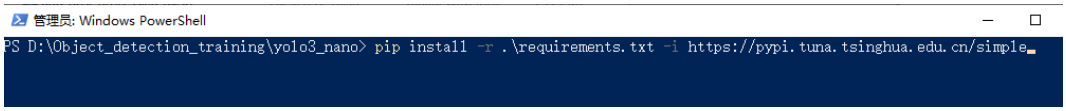
2. 进入 yolo3_nano 文件夹。

3. 按住 shift，鼠标右键点击文件夹空白位置,选择打开。Powershell 窗口



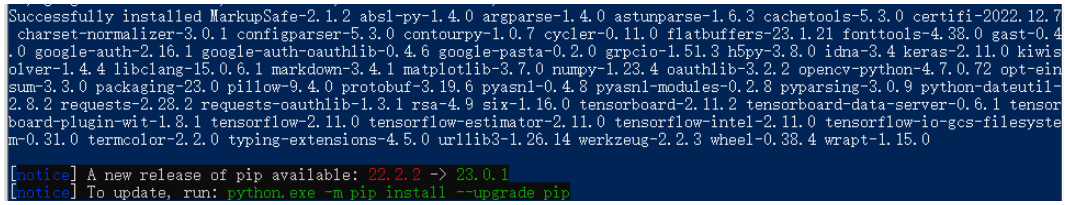
4. 在窗口中输入下面命令，并回车进行安装依赖库。

pip install -r .\requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple

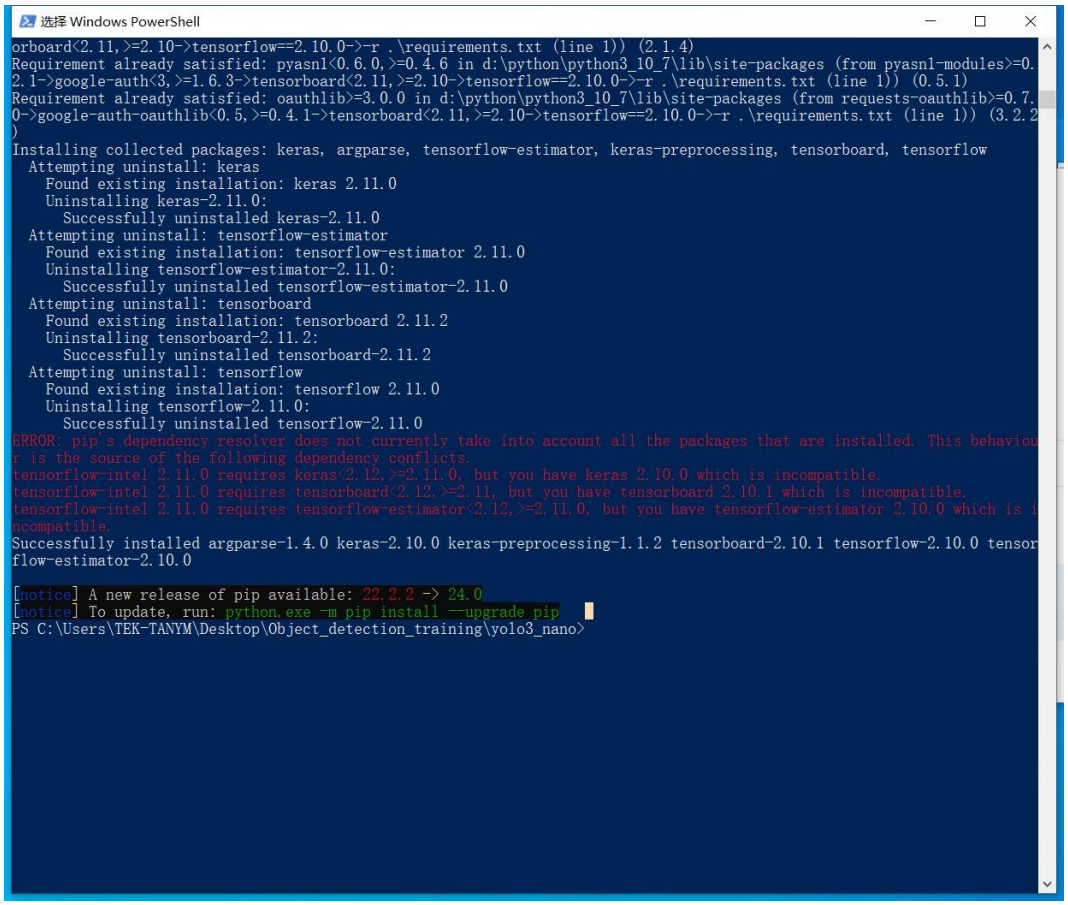


5. 等待安装完成。

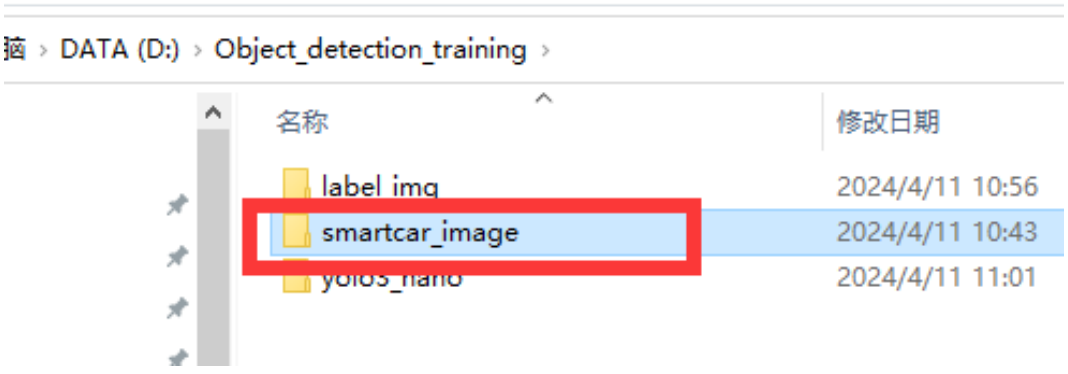
6. 看到以下信息表示安装完成。如果遇到 pip 报错，可以先尝试执行后面的步骤。



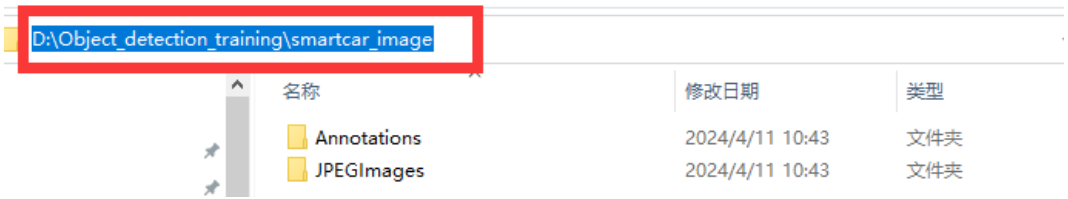
遇到下面这个错误不用管



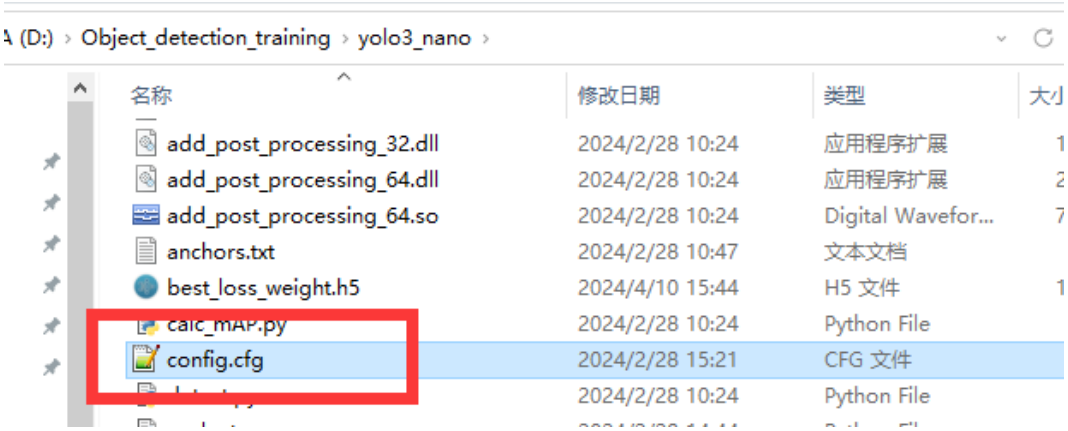
7. 我们的 demo 中提供了用于训练的图片和标记文件，里面的内容和上一章节标记的 car 文件夹相同，如果需要用自己的图片，替换这个文件夹即可，如下图所示：



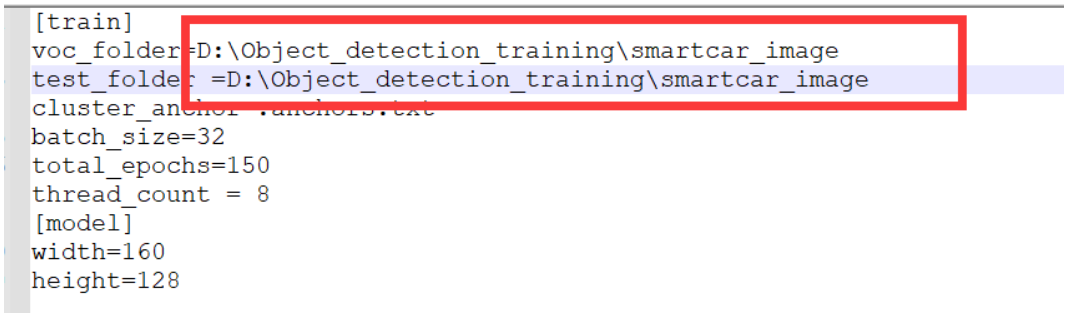
8. 进入文件夹，并复制路径 D:\Object_detection_training\smartcar_image



9. 进入 yolo3_nano 文件夹，找到 config.cfg 文件并打开

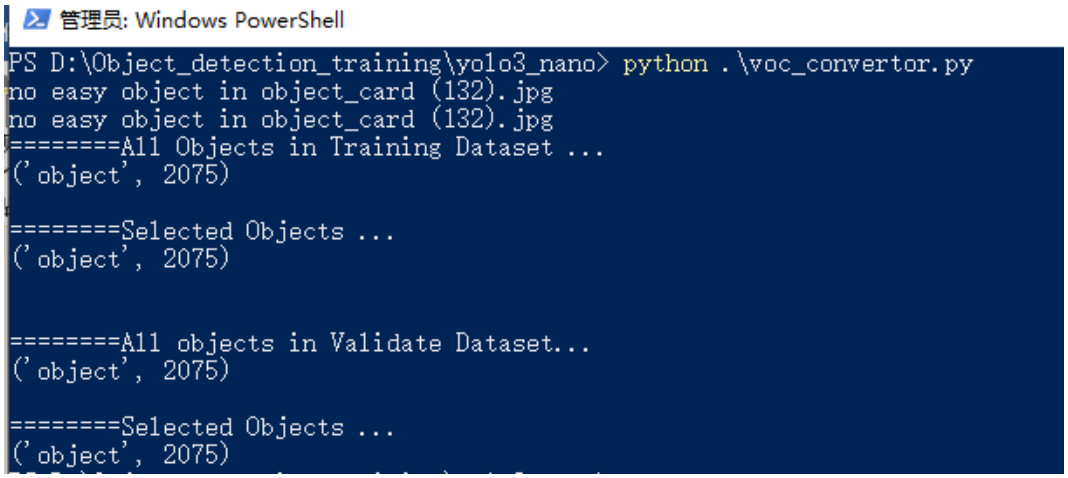


10. 文件中修改文件夹路径，将刚刚复制的粘贴到如下图所示位置，并保存。



11. 再次进入 Powershell 窗口。

12. 输入 python .\voc_convertor.py 命令，并回车。



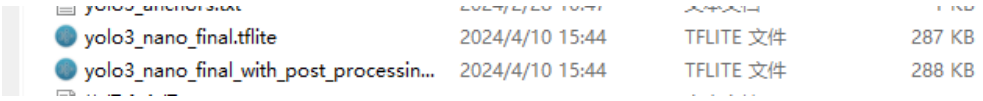
13. 输入 python .\kmeans.py 命令，并回车

```
PS D:\Object_detection_training\yolo3_nano> python .\kmeans.py
max = min : C:\Users\Administrator\Desktop\yolo3_nano\card_labe/JPEGImages/object_card (1712).jpg$117,39
10,83,0

max = min : C:\Users\Administrator\Desktop\yolo3_nano\card_labe/JPEGImages/object_card (1876).jpg$43,17,
108,0

max = min : C:\Users\Administrator\Desktop\yolo3_nano\card_labe/JPEGImages/object_card (706).jpg$10,72,9

box error count:3 in 1825 files
K anchors:
 [[ 9 18]
 [16 23]
 [18  8]
 [21 10]
 [23 29]
 [25 11]
 [29 14]
 [34 20]
 [45 39]]
(9, 2)
Avg Accuracy: 81.36%
 Accuracy: 53.47%
 Accuracy: 64.58%
 Accuracy: 62.65%
```

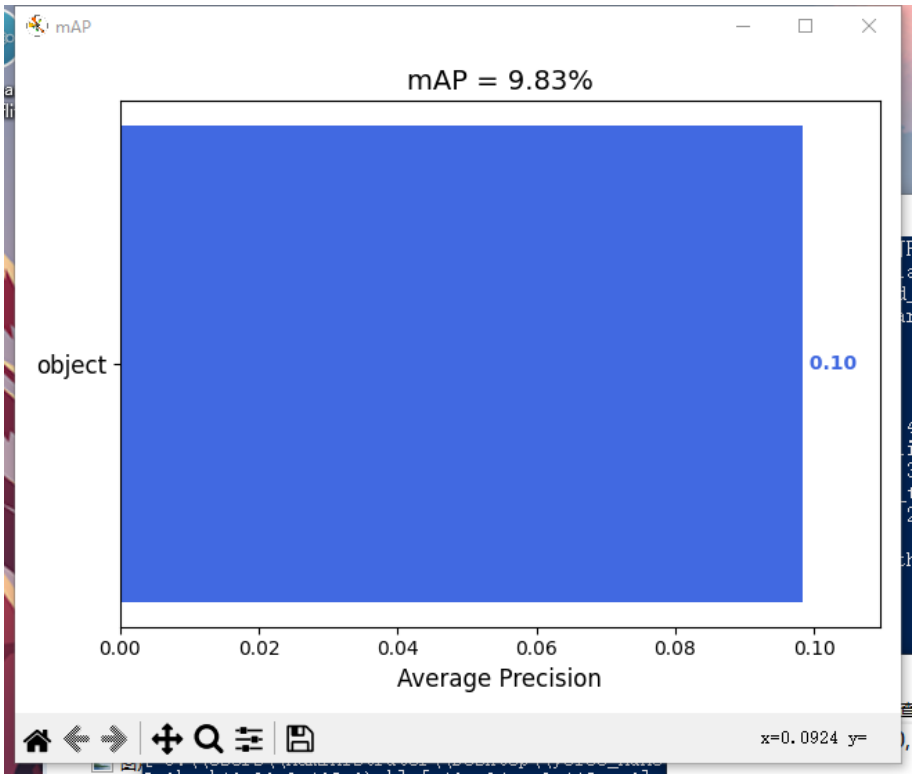
14. 输入 python .\train.py 命令，并回车，然后等待训练完成。开始运行后可能会等待一段时间，

```
PS D:\Object_detection_training\yolo3_nano> python .\train.py
2024-04-11 11:58:13.378494: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libr
ry 'cudart64_110.dll'; dlerror: cudart64_110.dll not found
2024-04-11 11:58:13.378618: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do
ot have a GPU set up on your machine.
Load weights ./weights/mbv1_0.25_128_128.h5.
2024-04-11 11:58:15.905334: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic libr
ry 'nvcuda.dll'; dlerror: nvcuda.dll not found
2024-04-11 11:58:15.905427: W tensorflow/stream_executor/cuda/cuda_driver.cc:263] failed call to cuInit: UNKNOWN ERROR
303)
2024-04-11 11:58:15.909325: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic infor
ation for host: TEK-ZhangSY
2024-04-11 11:58:15.909490: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: TEK-ZhangSY
2024-04-11 11:58:15.910093: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized wi
h oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:
 AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Create Tiny YOLOv3 model with 9 anchors and 1 classes.



E:\Python\Python_3_10_7\lib\site-packages\keras\optimizers\optimizer_v2\adam.py:114: UserWarning: The `lr` argument is
eprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
Model: "model_2"
__________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
 input_5 (InputLayer)        [(None, 128, 160, 3       0         []
                             )]

 conv1 (Conv2D)              (None, 64, 80, 8)         216       ['input_5[0][0]']
```

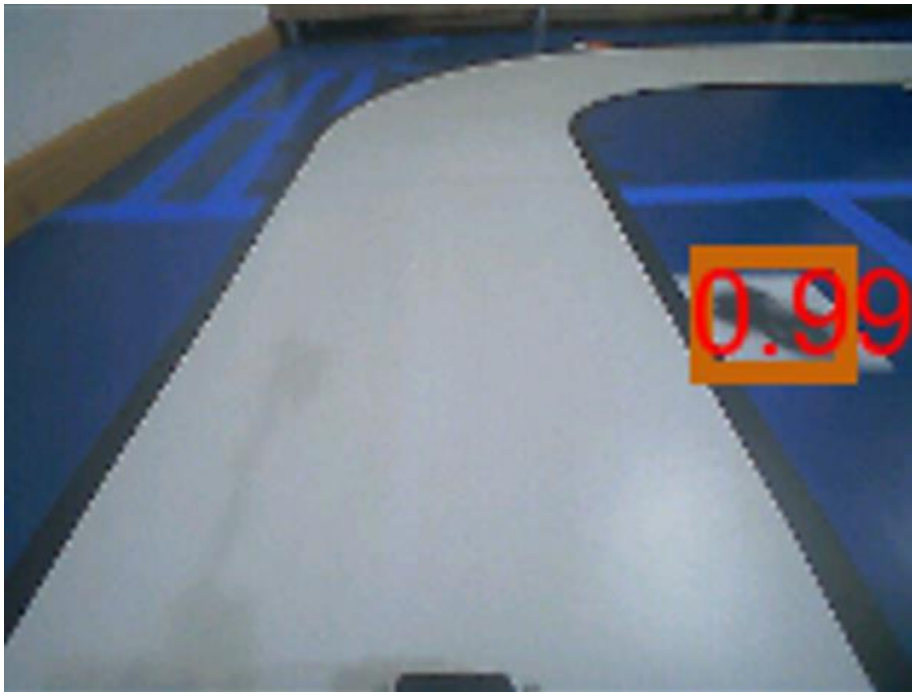
15. 最终会在当前文件夹下生成两个文件，如下图所示。



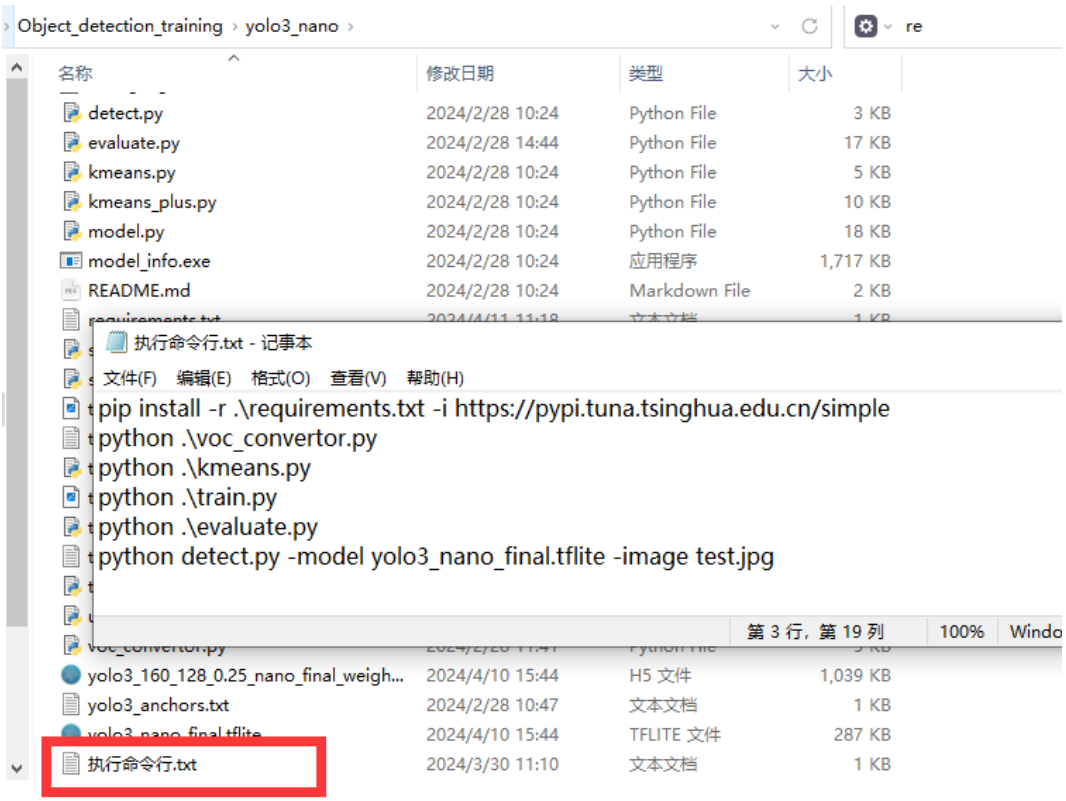
16. 输入 python .\evaluate.py 命令验证，会生成一个准确度的图，运行过程中会等稍长一点时间。



17. 输入 python detect.py -model yolo3_nano_final.tflite -image test.jpg 用于测试图的验证。



18. 如果命令记不到，可以打开文件夹下的 执行命令行.txt，里面有所用到的命令

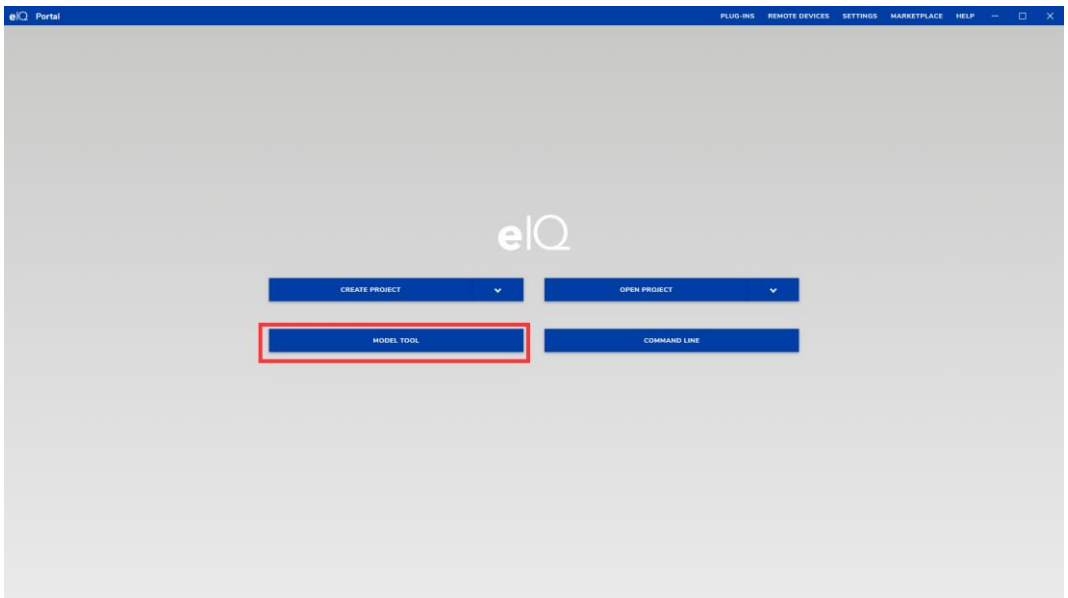


19. 训练完成后，模型还无法直接放到 MCX Vision 中进行使用，具体使用看下一节。

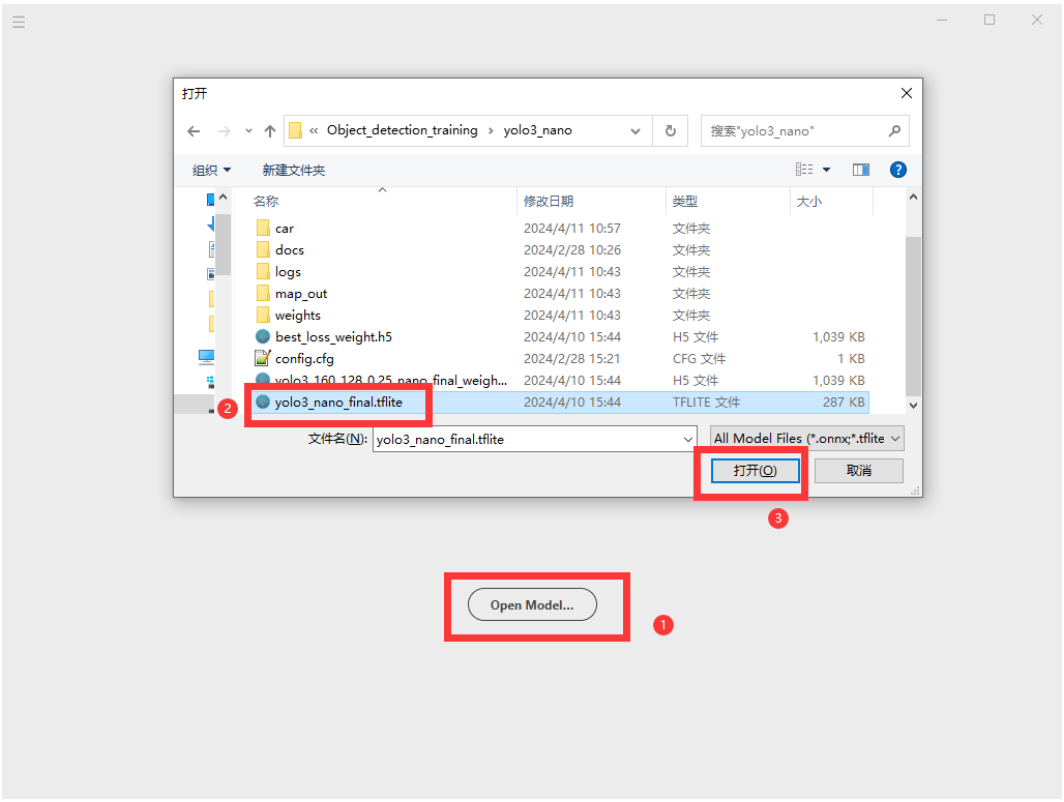
6.5.模型使用

1. 上一章节输出了两个模型，我们需要用到第一个模型。

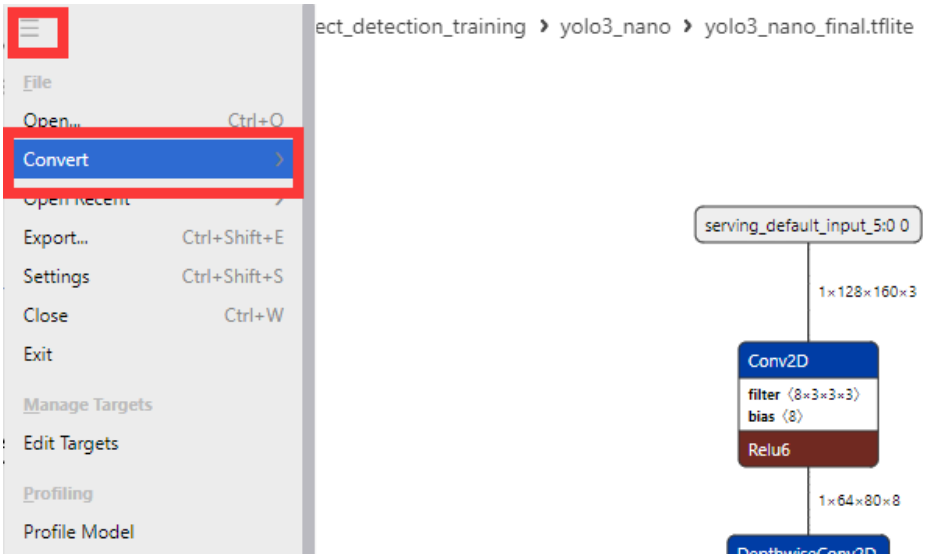
2. 打开 eIQ 软件，选择 model tool。



3. 打开模型

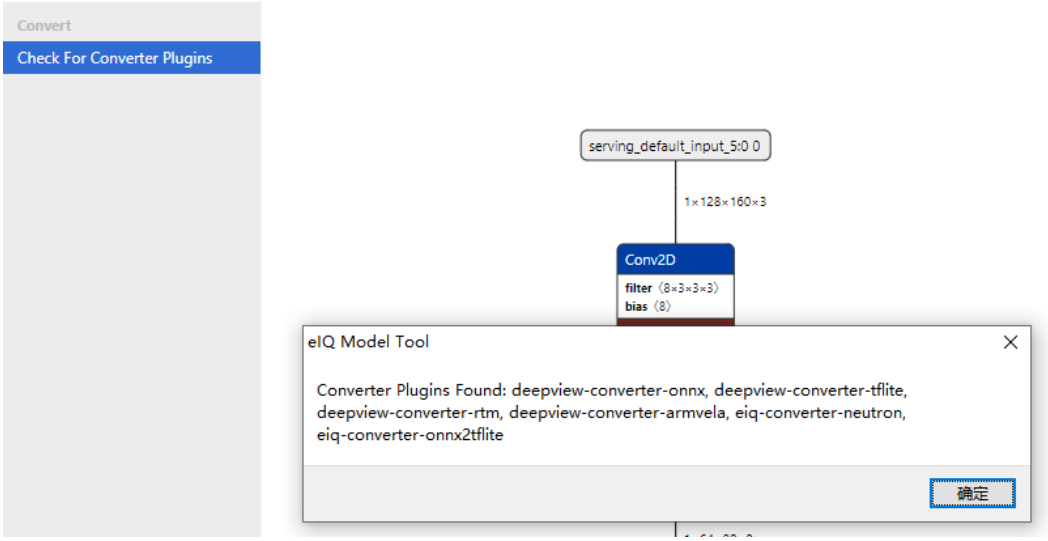


4. 点击左上角的三根杠，然后点击 Convert

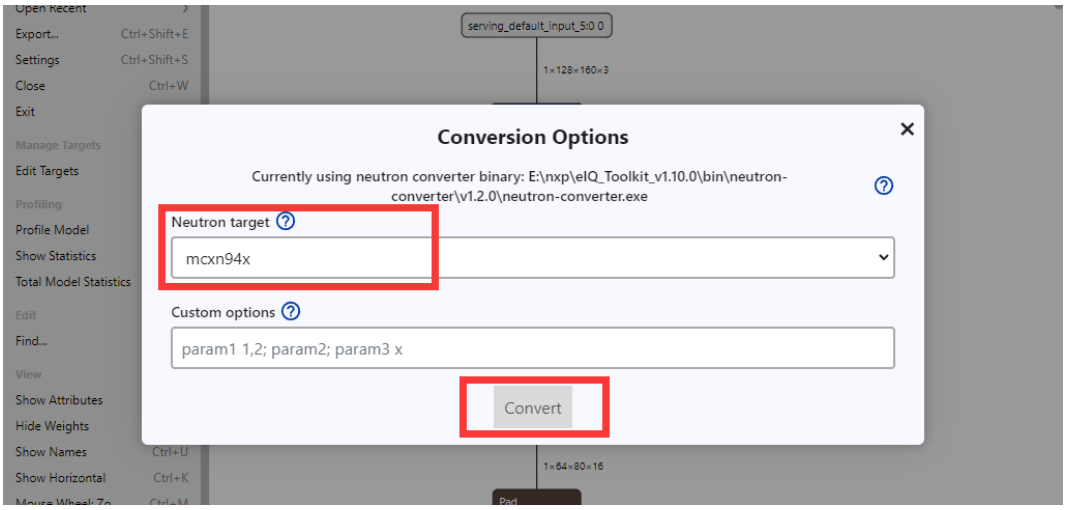


5. 选择 Tensor Flow Lite For Neutron

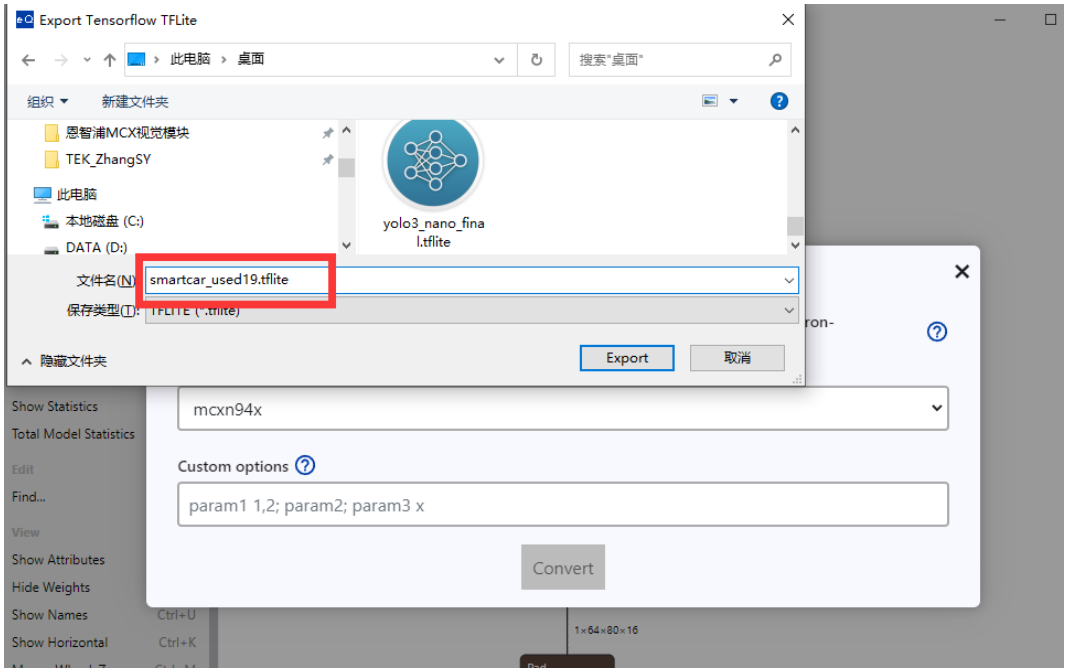
可能会出现如下情况，点击 Check，等待一段时间，跳出弹窗后重新执行上述操作。



6. 在弹出页面中选择 mcxn94x 选项，然后点击 Convert

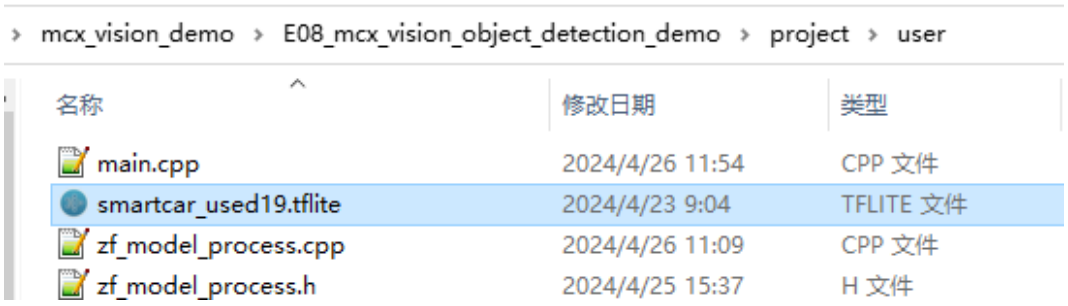


7. 更改导出文件名称为 smartcar_used19.tflite，导出到合适的位置，改名是为了后面方便使用。

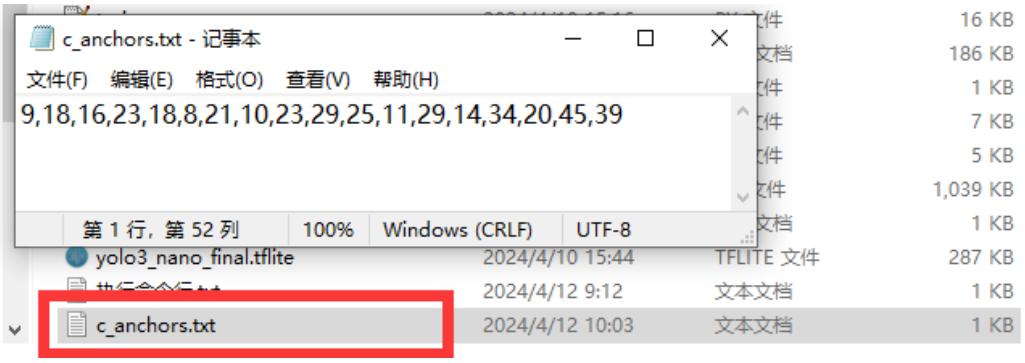


8. 打开 mcx vision 的模型使用例程。把模型放到 user 中，如果已存在就覆盖它，确保名称和这个名称完全相同。

E08_mcx_vision_object_detection_demo	目标检测例程

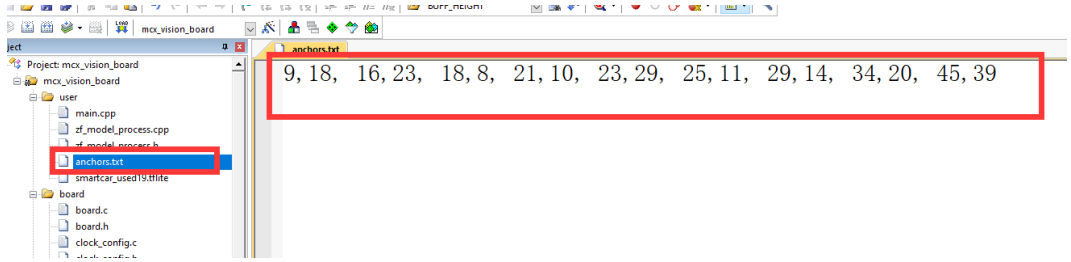


9. 打开之前的训练脚本路径，打开里面的 c_anchors.txt 文件



10. 将这一段数据复制。

11. 打开 mcx vision 的工程，也打开 anchors.txt 文件，并将刚刚复制的内容粘贴进去



12. 然后编译并下载程序即可运行训练好的模型，下载完程序后需要手动复位。

7.问题总结

7.1.编译报错 1

报错信息：error: use of undeclared identifier 'posix_memalign'

解决方案：

1. 更新 MDK 到 5.38a（资料中提供了）

2.重新去 gitee 下载例程，然后重新编译下载，因为之前的工程在低版本打开过，所以需要重新下载例程压缩包

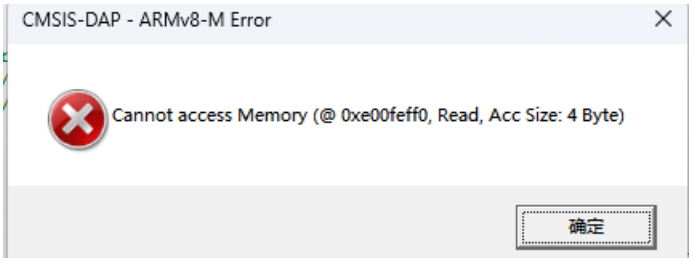
7.2.编译报错 2

报错信息：error: L6050E

解决方案：激活 keil

7.3.下载报错 1

报错信息：



解决方案：

1. 更新 MDK 到 5.38a（资料中提供了）

2. 将模块解锁：上电后按住 BOOT 按键不放，按一下 RST 按键，然后松开 RST 按键

3.重新去 gitee 下载例程，然后重新编译下载，因为之前的工程在低版本打开过，所以需要重新下载例程压缩包

7.4.下载报错 2

报错信息：

解决方案：

1. 原因是下载口有串口的引脚，芯片如果外设优先于电源上电，内核无法启动。

2. 断开下载器重新插入

7.5.无报错信息

下载成功无法运行代码。目前在 5.39 版本遇到

解决方案：

1. 更新 MDK 到 5.38a（资料中提供了）

2.重新去 gitee 下载例程，然后重新编译下载，因为之前的工程在低版本打开过，所以需要重新下载例程压缩包

8.文档版本

版本号	日期	作者	内容变更
V1.0	2024/4/26	ZSY	初始版本。
V1.1	2024/5/16	ZSY	增加报错信息说明