

-目录/Index

任务一：车道线识别_数据采集	4
任务二：车道线识别_数据处理	9
任务三：车道线识别_训练模型	11
任务四：沿车道线自主移动	14
任务五：标志物检测_数据采集	16
任务六：标志物检测_数据处理	18
任务七：标志物检测_训练模型	23

智能车简介

2019 年的特斯拉自动驾驶开放日上，特斯拉人工智能高级主管 Andrej Karpathy 强调特物理数据无法代替，对于依赖虚拟仿真自动驾驶，特斯拉更相信现实物理数据。也就是说，看图比雷达更真实。在发布会后环节中，马斯克也再次重申自己的态度，我们不用激光雷达，这就是态度。

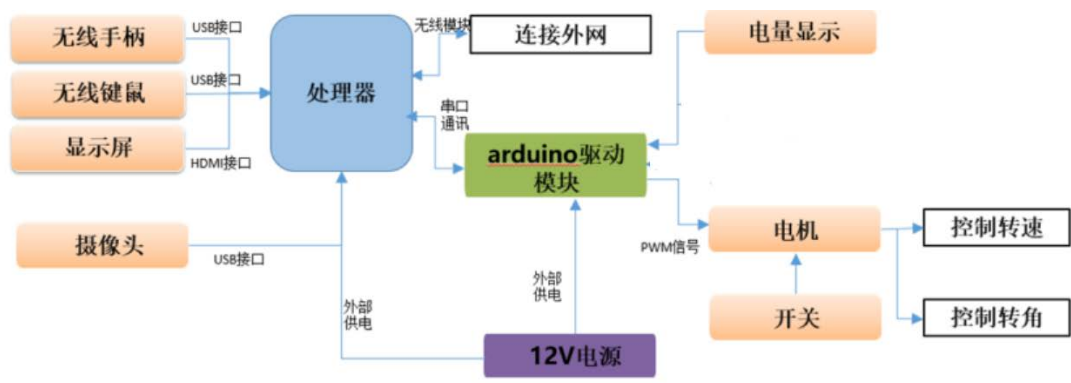
无人驾驶（深度学习）智能车采用 Python 编程语言，以深度学习开源框架百度飞桨 paddlepaddle 为基础，高度集成硬件驱动模块，分布式结构化软件设计框架，可实现数据采集、数据模型构建、自主识别弯道、无人驾驶验证等功能，是一套学习深度学习开发的最优平台。

PaddlePaddle，中文名称：飞桨；作为国内唯一功能完备的端对端开源深度学习平台，集深度学习训练和预测框架、模型库、工具组件、服务平台为一体，其兼具灵活和效率的开发机制、工业级应用效果的模型、超大规模并行深度学习能力、推理引擎一体化设计以及系统化的服务支持,致力于让深度学习技术的创新与应用更简单。

AI Studio 是基于深度学习平台飞桨的一站式 AI 开发平台，提供在线编程环境、免费 GPU 算力、海量开源算法和开放数据，帮助理发者快速创建和部署模型。



序号	类型	参数
1	底盘	四轮带编码器差速底盘。 尺寸:340mm*270mm*300mm (长宽高)
2	电机	G37-520B 编码器直流电机 12V 空载转速: 178rpm
3	处理器	主控: i5 系列内存 : 4G
4	摄像头 (2 个)	像素 720P 对角 70 度 水平 55 度 YUY2/10-15 帧/S



在采集数据的时候，采用手柄控制小车采取数据；

在自主运行时候，该车通过摄像头获取数据，接着在处理器中进行处理，进而控制小车移动。

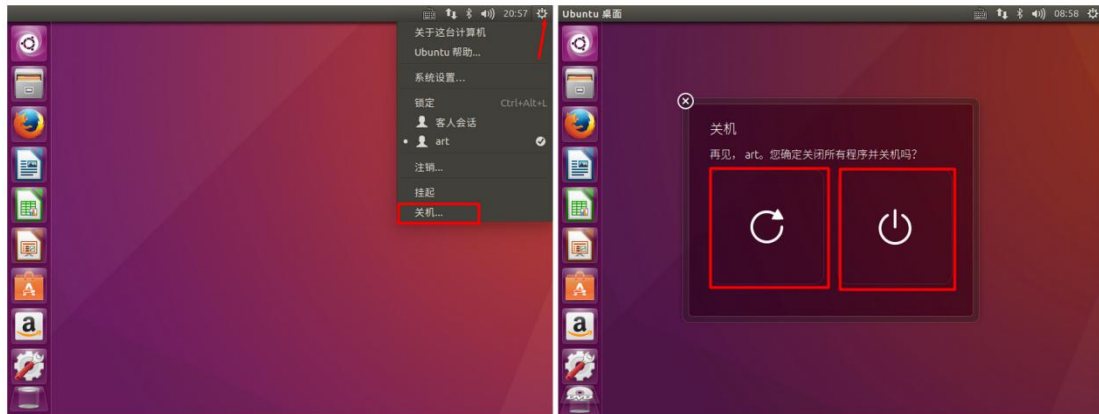
任务一：车道线识别_数据采集

深度学习训练之前需要获取跑道的图像集以及对应时刻的前轮转弯角度。本章通过手柄控制小车在跑道内移动，并通过摄像头1采集跑道图像以及转弯信息。



开机：直接按车载电脑开关，可直接打开电脑，此时显示屏上会显示 ubuntu16.04 的界面；**用户密码是：1（回车）**

关机：结束时可直接单击右上角的设置标志，单击“关机”，跳出的对话框中可以选择“重启”和“关机”，左侧为“重启”键，右侧为关机键。



ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/deeplearning_car/src/ 目录下,可以看到里面包含 9 个 python 脚本。其中 Data_Coll.py 实现的功能是获取数据。

```
2020-04-24 10:36:39 deep in ~  
→ cd ~/paddlepaddle/deeplearning_car/src/  
  
2020-04-24 10:36:51 deep in ~/paddlepaddle/deeplearning_car/src  
→ ls  
Auto_Driver_client2.py  cnn_model.py  Ing_Handle.py  Train_Model.py  
Auto_Driver_client.py  Create_Data_Liet.py  pycache  
Auto_Driver.py  Data_Coll.py  reader.py  
  
2020-04-24 10:37:33 deep in ~/paddlepaddle/deeplearning_car/src  
→
```

运行命令 `python3 Data_coll.py -h`，可以通过 `-h` 查看里面涉及到的参数：

```
终端
2020-04-24 10:39:16 deep in ~/paddlepaddle/deeplearning_car/src
python3 Data_Coll.py -h
python3 Data_Coll.py --vels=1600 --output=data.npy --serial=/dev/ttyUSB0 --camera=/dev/video0

2020-04-24 10:40:03 deep in ~/paddlepaddle/deeplearning_car/src
```

其中 vels 是当前速度参数，默认值为 1540；output 是输出文件格式；serial 是调用的串口；camera 是调用的摄像机参数；save_name 是保存的文件名。在输入时可以按照自己的需要对其几个参数进行调整，若不后加参数按默认值；输入 python3 Data_Coll.py 并回车执行该文件。

```
终端
2020-04-24 10:41:21 deep in ~
cd ~/paddlepaddle/deeplearning_car/src/

2020-04-24 10:41:33 deep in ~/paddlepaddle/deeplearning_car/src
python3 Data_Coll.py
available devices
/dev/input/js0
Open Serial port success!
fcntl=0
set done!
Wait Start!
```

接着将小车放在搭建好的跑道上，开启底盘开关，插上手柄接收头。通过手柄控制小车移动，首先将总开关调到 ON 档位，按上电按钮 START，再进行模式切换，指示灯红绿两个灯都亮表示模式切换正确。



按动启动键，小车将开始跑起来采集数据了，通过转向遥感控制小车左右转弯。

```
speed= 1560   angle= 1500
imgInd= 222
speed= 1560   angle= 1500
imgInd= 223
speed= 1560   angle= 1500
imgInd= 224
speed= 1560   angle= 1500
imgInd= 225
speed= 1560   angle= 1500
imgInd= 226
speed= 1560   angle= 1500
imgInd= 227
speed= 1560   angle= 1500
imgInd= 228
speed= 1560   angle= 1500
imgInd= 229
speed= 1560   angle= 1500
imgInd= 230
speed= 1560   angle= 1500
imgInd= 231
```

采集结束后，通过按 4 次结束键结束采集数据。


```
终端
speed= 1500    angle= 1132
imgInd= 1080
speed= 1500    angle= 1132
imgInd= 1081
speed= 1500    angle= 1132
imgInd= 1082
speed= 1500    angle= 1132
imgInd= 1083
Stop
speed= 1500    angle= 1132
imgInd= 1084
speed= 1500    angle= 1500
Stop
Stop
Stop
Stop
STOP CAR
TXT to npy
finally
Stop
car run finally

2020-04-24 10:52:28 deep in ~/paddlepaddle/deeplearning_car/src
```

采集的数据放到~/paddlepaddle/deeplearning_car/data 目录下，通过命令 `cd ~/paddlepaddle/deeplearning_car/ data` 切换到该目录下；通过命令 `ls` 显示采集过程中获取的图像集 `img` 文件夹以及数据 `data.npy`。

```
终端

2020-04-24 10:54:05 deep in ~
→cd ~/paddlepaddle/deeplearning_car/data/

2020-04-24 10:54:38 deep in ~/paddlepaddle/deeplearning_car/data
→ls
data.npy data.txt img

2020-04-24 10:54:42 deep in ~/paddlepaddle/deeplearning_car/data
→
```

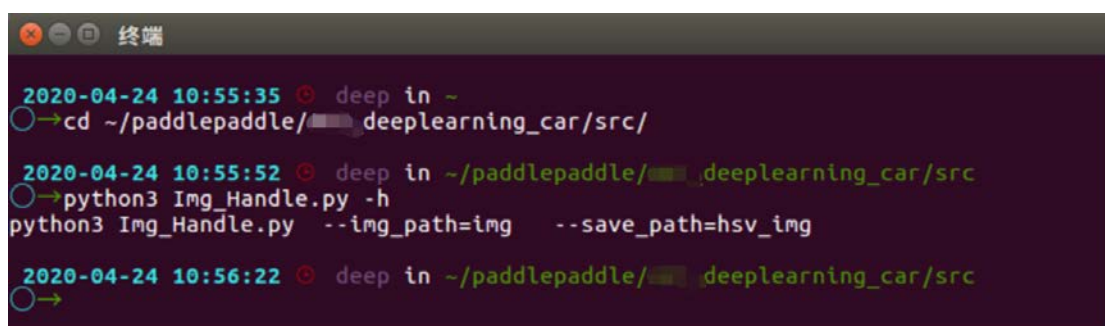
Ctrl+c 快捷键，关闭终端。

任务二：车道线识别_数据处理

由于获取的图像信息中没有用的数据信息很多，会影响到训练效果，本章对获取的图像集进行二值化处理，提取出对训练有用的跑道信息。

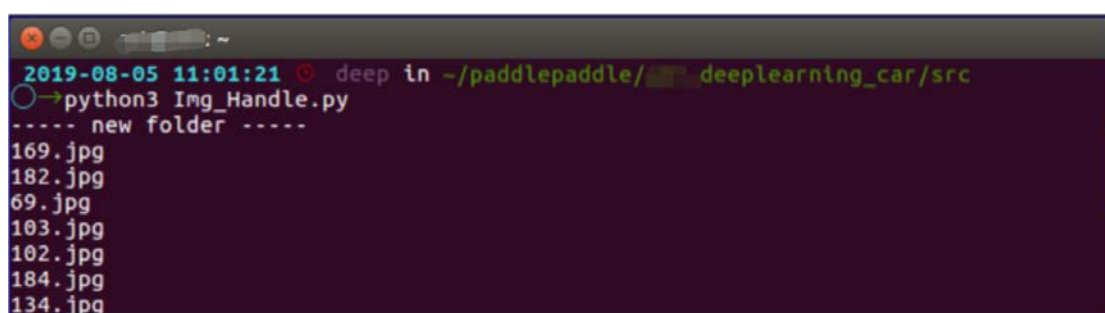
ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/deeplearning_car/src/ 目录下，其中 Img_Handle.py 实现对数据进行处理。

运行命令 `python3 Img_Handle.py -h`，可以通过 -h 查看里面涉及到的参数：

A terminal window titled '终端' (Terminal) with a dark background. It shows a series of commands and their outputs. The first command is 'cd ~/paddlepaddle/deeplearning_car/src/' which is executed successfully. The second command is 'python3 Img_Handle.py -h' which outputs 'python3 Img_Handle.py --img_path=img --save_path=hsv_img'. The third command is 'python3 Img_Handle.py --img_path=img --save_path=hsv_img' which is also executed successfully. The terminal shows the date and time for each command: 2020-04-24 10:55:35, 2020-04-24 10:55:52, and 2020-04-24 10:56:22.

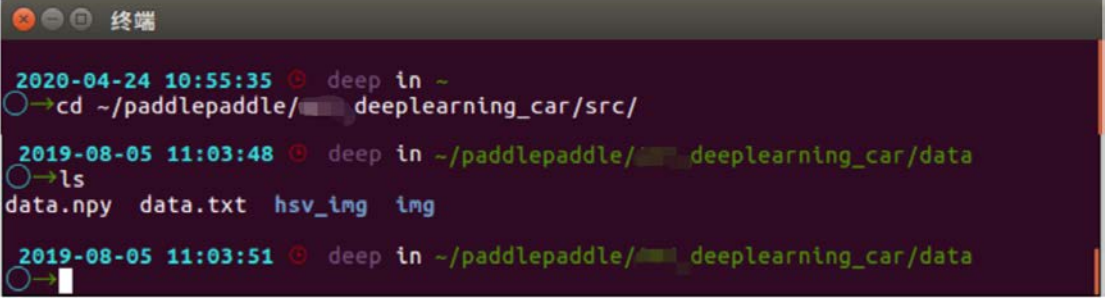
```
2020-04-24 10:55:35 deep in -  
→ cd ~/paddlepaddle/deeplearning_car/src/  
  
2020-04-24 10:55:52 deep in ~/paddlepaddle/deeplearning_car/src  
→ python3 Img_Handle.py -h  
python3 Img_Handle.py --img_path=img --save_path=hsv_img  
  
2020-04-24 10:56:22 deep in ~/paddlepaddle/deeplearning_car/src  
→
```

其中 `img_path` 是输入的图像集；`hsv_img` 是输出处理后的图像集。运行 `Img_Handle.py` 文件，如下图所示。

A terminal window showing the execution of 'python3 Img_Handle.py'. The output indicates that a new folder has been created and lists several image files: 169.jpg, 182.jpg, 69.jpg, 103.jpg, 102.jpg, 184.jpg, and 134.jpg. The terminal shows the date and time: 2019-08-05 11:01:21.

```
2019-08-05 11:01:21 deep in ~/paddlepaddle/deeplearning_car/src  
→ python3 Img_Handle.py  
----- new folder -----  
169.jpg  
182.jpg  
69.jpg  
103.jpg  
102.jpg  
184.jpg  
134.jpg
```

采集的数据放了~/paddlepaddle/deeplearning_car/data 目录下，通过命令 `cd ~/paddlepaddle/deeplearning_car/data` 切换到该目录下；通过命令 `ls` 可以看到文件夹中已经生成了一个 `hsv_img` 文件夹。



```
2020-04-24 10:55:35 deep in ~  
→ cd ~/paddlepaddle/deeplearning_car/src/  
  
2019-08-05 11:03:48 deep in ~/paddlepaddle/deeplearning_car/data  
→ ls  
data.npy data.txt hsv_img img  
  
2019-08-05 11:03:51 deep in ~/paddlepaddle/deeplearning_car/data  
→
```

Ctrl+c 结束命令。

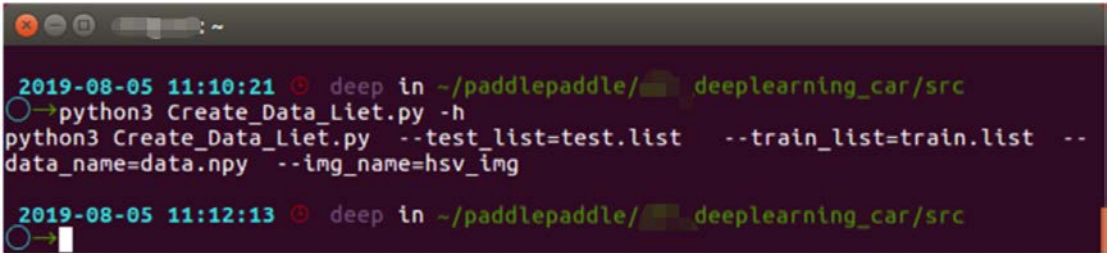
任务三：车道线识别_训练模型

数据处理好后，接下来对模型进行训练。本章对深度学习模型参数进行训练。其中有两种方式供选择，第一种方式，利用车载电脑 CPU 资源进行训练；第二种方式，利用线上 AIstudio 线上资源进行训练。

3.1 本地训练

ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/deeplearning_car/src/ 目录下，其中 Create_Data_List.py 获取训练数据列表和测试数据列表；Train_Model.py 实现对数据进行处理。

运行命令 python3 Create_Data_Liet.py -h，可以通过 -h 查看里面涉及到的参数：

A terminal window with a dark background and light-colored text. The prompt is 'deep in ~/paddlepaddle/deeplearning_car/src'. The user enters 'python3 Create_Data_Liet.py -h', which outputs 'python3 Create_Data_Liet.py --test_list=test.list --train_list=train.list --data_name=data.npy --img_name=hsv_img'. The user then enters another command, which is partially visible as 'python3 Train_Model.py --save_path=...'.

其中 test_list 是输入的测试列表文件；train_list 是输入的训练列表文件；save_path 是输出模型路径。运行 Train_Model.py 文件，如下图所示。

```
2020-04-24 11:01:15 deep in ~
→cd ~/paddlepaddle/deeplearning_car/src/

2020-04-24 11:01:31 deep in ~/paddlepaddle/deeplearning_car/src
→python3 Create_Data_List.py
----- there is this folder -----
loading image: /home/deep/paddlepaddle/deeplearning_car/src/../data/hsv_img
图像列表已生成

2020-04-24 11:02:04 deep in ~/paddlepaddle/deeplearning_car/src
→
```

采集的数据放到~/paddlepaddle/deeplearning_car/data 目录下，通过命令 `cd ~/paddlepaddle/deeplearning_car/data` 切换到该目录下；通过命令 `ls` 可以看到文件夹中已经生成了一个 `data.list` 和 `train.list`。

```
2020-04-24 11:02:40 deep in ~
→cd ~/paddlepaddle/deeplearning_car/data/

2020-04-24 11:02:59 deep in ~/paddlepaddle/deeplearning_car/data
→ls
data.npy data.txt hsv_img img test.list train.list

2020-04-24 11:03:10 deep in ~/paddlepaddle/deeplearning_car/data
→
```

运行命令 `python3 Train_Model.py -h`，可以通过 `-h` 查看里面涉及到的参数：

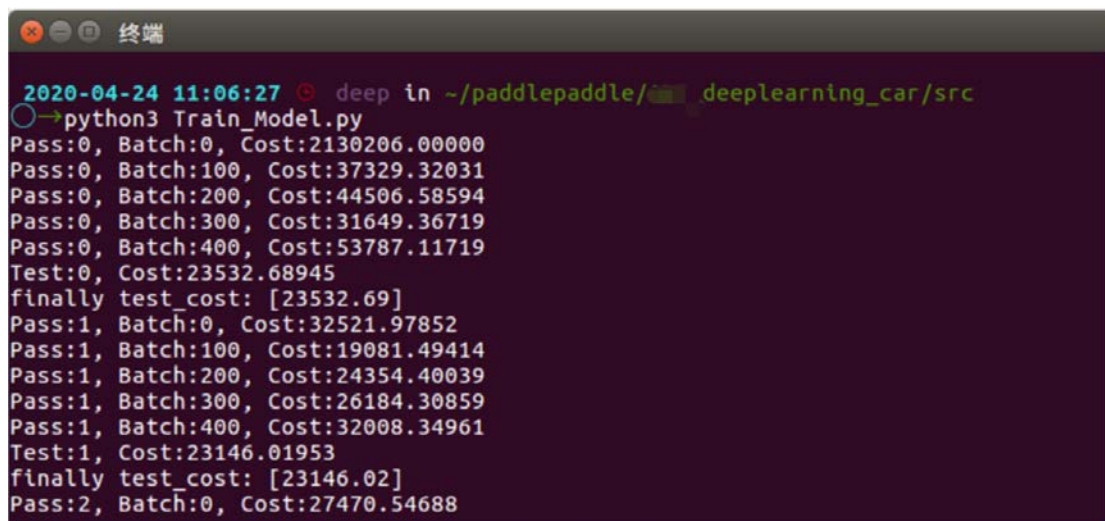
```
2020-04-24 11:05:02 deep in ~
→cd ~/paddlepaddle/deeplearning_car/src/

2020-04-24 11:05:19 deep in ~/paddlepaddle/deeplearning_car/src
→python3 Train_Model.py -h
python3 Train_Model.py --test_list=test.list --train_list=train.list --save_path=model_infer

2020-04-24 11:05:37 deep in ~/paddlepaddle/deeplearning_car/src
→
```

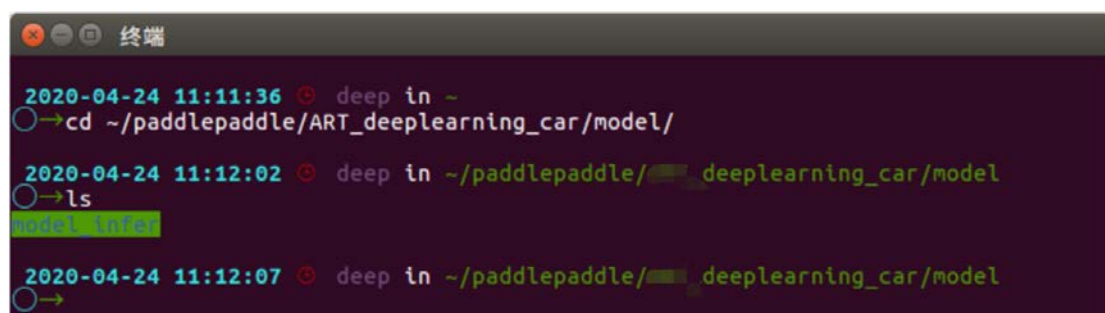
其中 `test_list` 是输入的测试数据列表文件；`train_list` 是输入的训练数据列表文件；`save_path` 是输出模型路径。运行

Train_Model.py 文件，如下图所示。



```
2020-04-24 11:06:27 deep in ~/paddlepaddle/ART_deeplearning_car/src
python3 Train_Model.py
Pass:0, Batch:0, Cost:2130206.00000
Pass:0, Batch:100, Cost:37329.32031
Pass:0, Batch:200, Cost:44506.58594
Pass:0, Batch:300, Cost:31649.36719
Pass:0, Batch:400, Cost:53787.11719
Test:0, Cost:23532.68945
finally test_cost: [23532.69]
Pass:1, Batch:0, Cost:32521.97852
Pass:1, Batch:100, Cost:19081.49414
Pass:1, Batch:200, Cost:24354.40039
Pass:1, Batch:300, Cost:26184.30859
Pass:1, Batch:400, Cost:32008.34961
Test:1, Cost:23146.01953
finally test_cost: [23146.02]
Pass:2, Batch:0, Cost:27470.54688
```

采集的数据放~/ paddlepaddle/deeplearning_car/model 目录下，通过命令 `cd ~/paddlepaddle/deeplearning_car/model` 切换到该目录下；通过命令 `ls` 可以看到文件夹中已经生成了一个模型文件。



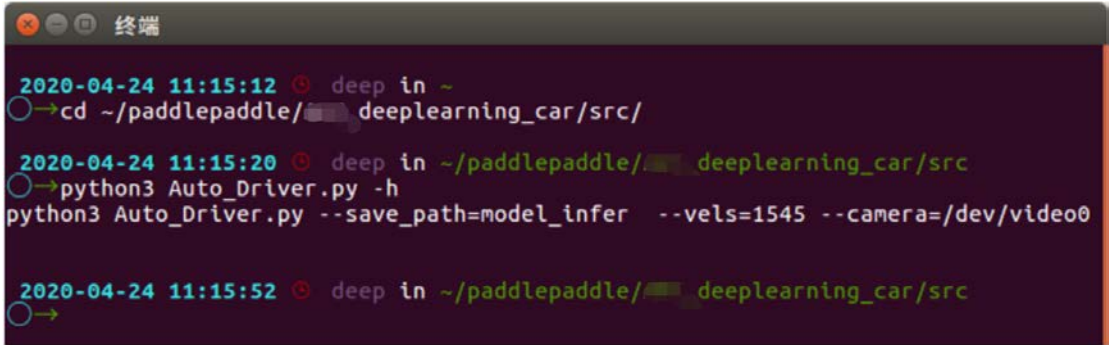
```
2020-04-24 11:11:36 deep in ~
cd ~/paddlepaddle/ART_deeplearning_car/model/
2020-04-24 11:12:02 deep in ~/paddlepaddle/ART_deeplearning_car/model
ls
model_infer
2020-04-24 11:12:07 deep in ~/paddlepaddle/ART_deeplearning_car/model
```

任务四：沿车道线自主移动

训练后，小车可以调用训练好的模型，进行自主移动。本章的内容主要介绍如何远程控制小车自主运行。

ctrl+alt+t 打开终端，使用 `cd` 命令切换目录到 `~/paddlepaddle/deeplearning_car/src/` 目录下，其中 `Auto_Driver.py` 实现小车自主移动。

运行命令 `python3 Auto_Driver.py -h`，可以通过 `-h` 查看里面涉及到的参数：



```
2020-04-24 11:15:12 deep in ~
→ cd ~/paddlepaddle/deeplearning_car/src/

2020-04-24 11:15:20 deep in ~/paddlepaddle/deeplearning_car/src
→ python3 Auto_Driver.py -h
python3 Auto_Driver.py --save_path=model_infer --vels=1545 --camera=/dev/video0

2020-04-24 11:15:52 deep in ~/paddlepaddle/deeplearning_car/src
→
```

其中 `vels` 是小车运行速度；`camera` 是调用的摄像机参数；`save_path` 是模型路径。

将小车放在跑道上，运行 `python3 Auto_Driver.py` 命令，如下图所示。

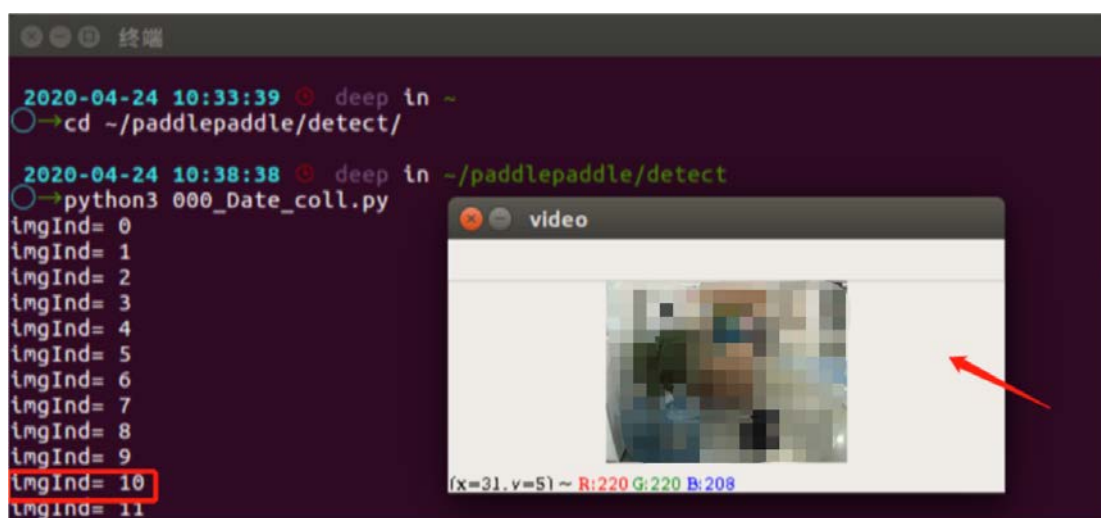

```
2019-08-05 11:26:45 deep in ~/paddlepaddle/deeplearning_car/src
python3 Auto_Driver.py
Open Serial port success!
fcntl=0
set done!
0
angle: 1500, throttle: 1560
1
angle: 1500, throttle: 1560
2
angle: 1500, throttle: 1560
3
angle: 1500, throttle: 1560
4
angle: 1500, throttle: 1560
5
angle: 1500, throttle: 1560
```

ctrl+c 结束当前命令；小车停止移动。

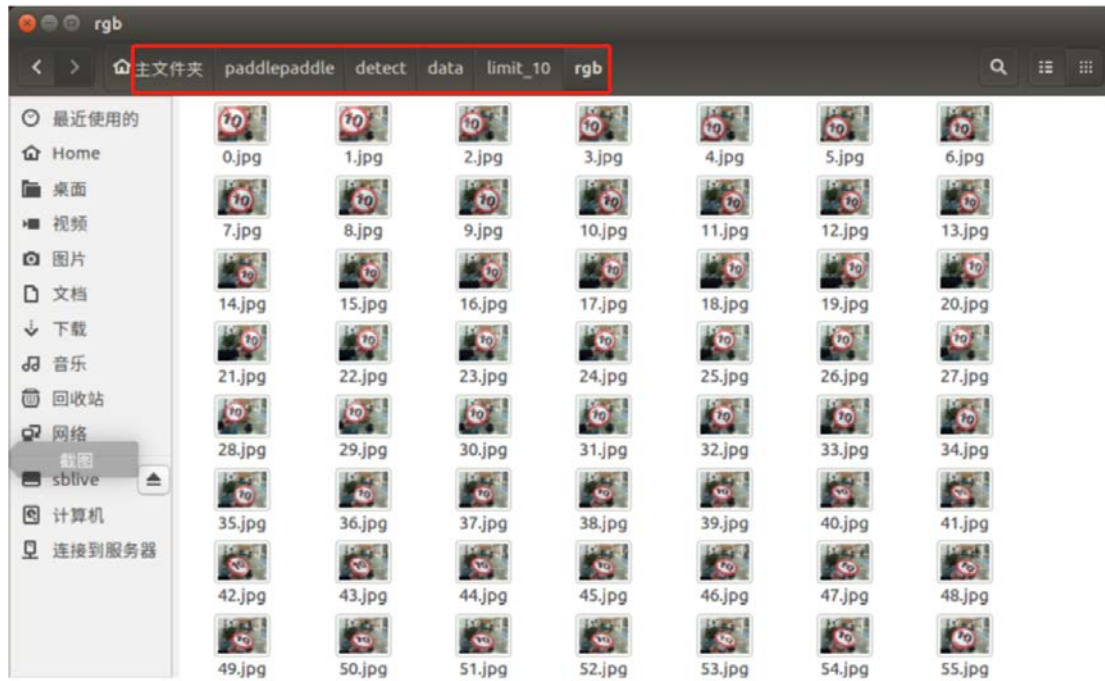
任务五：标志物检测_数据采集

标志物检测需要获取标志物的图像集以及对应的标签文件。
本章通过摄像头 2 采集标志物信息。

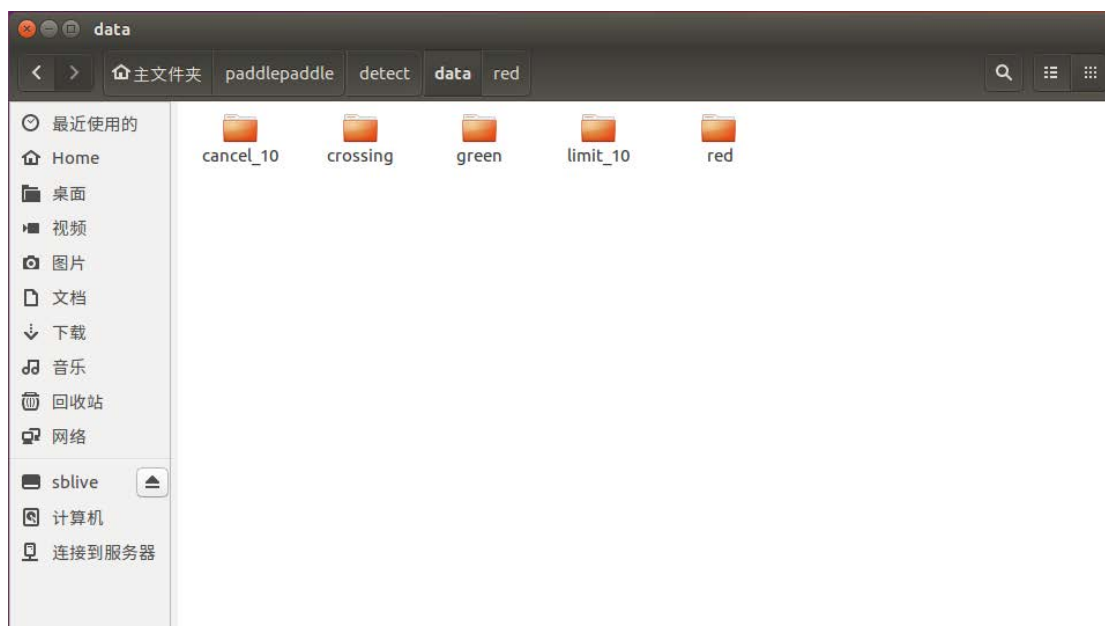
开机后,ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/detect/ 目录下,可以看到里面包含 5 个 python 脚本。
其中 000_Data_Coll.py 实现的功能是获取数据。



此时会弹出当前采集的图像，以及采集的个数，此时移动车体或移动标志物，获取不同视角下的包含有标志物的图像；当采集到合适数量图像后，单击显示图像并输入 ‘q’ 结束采集；保存的图像将会放在 ~/paddlepaddle/detect/data/img/rgb 目录下，可以到该目录下已经生成的图像数据；



将 **img** 文件夹名字改成标签名称；进而采集下一个标志物，采集完后~/paddlepaddle/detect/data 目录内容如图所示：



任务六：标志物检测_数据处理

获取图像信息后，需要对图像进行打标签，并按照一定格式打包数据集，本章利用 `labelimg` 软件实现打标签，并通过几个 `python` 脚本实现数据集打包整理。

6.1 打标签

软件安装：此处需要安装 `labelimg` 软件对每张图像进行标注；该软件可以安装在 `windows` 上，也可以安装在车载电脑 `ubuntu16.04` 系统上；

安装在 `windows` 上的软件版本连接可以从这里下载：链接：
<https://pan.baidu.com/s/1Z0Hui24a0B9O5Boax0JaeQ> 提取码：
5w09

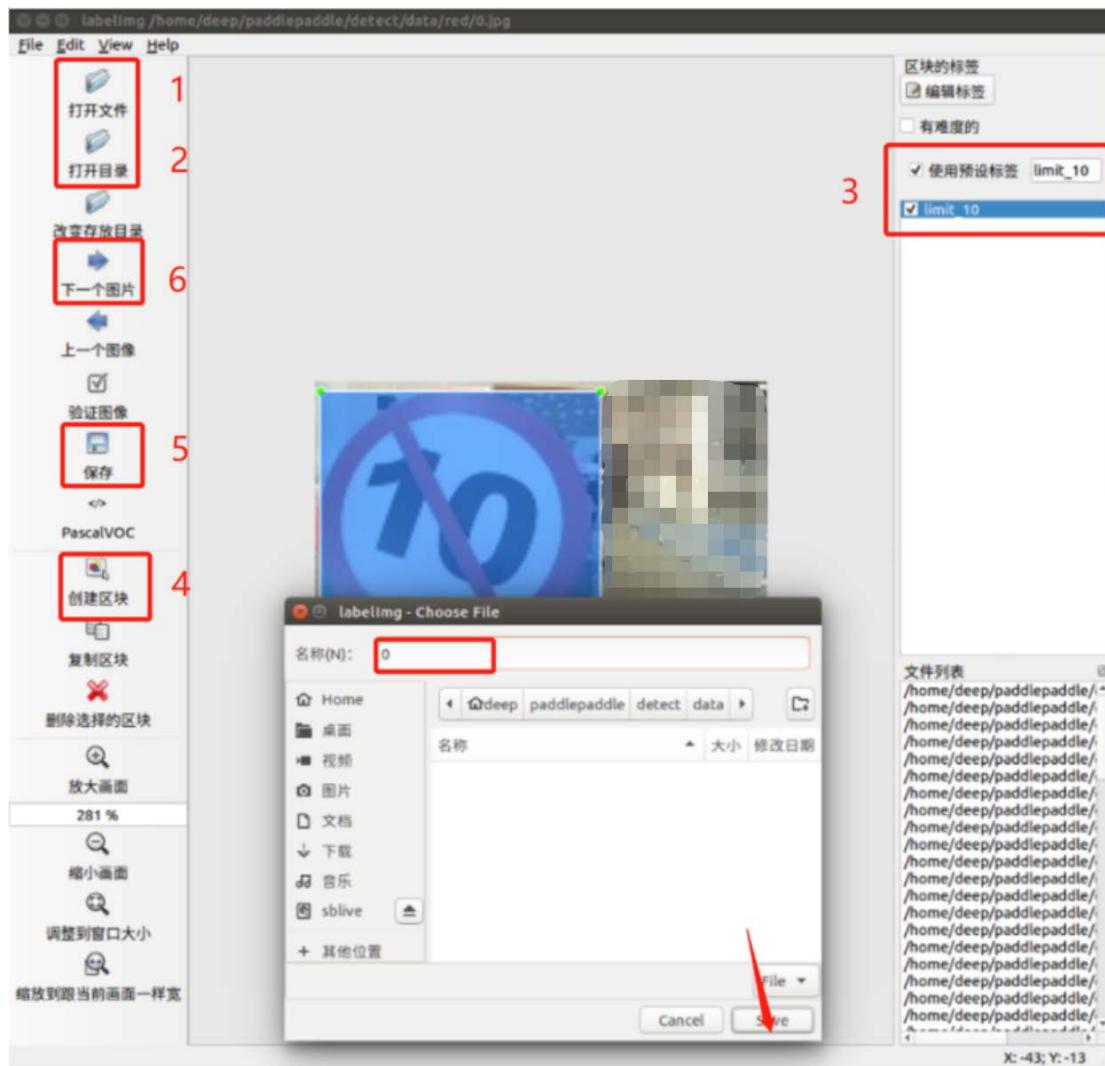
安装在 `ubuntu16.04` 上需要进行如下操作：1.通过网口或无线连接外网；2.`ctrl+alt+t` 打开终端；3.运行命令 `sudo pip3 install labelImg`；4.此时软件安装完成

打开软件：`ctrl+alt+t` 打开终端；运行命令 `sudo labelImg` 启动软件

```
2020-04-24 13:57:02 deep in ~
→sudo pip3 install labelImg
WARNING: The directory '/home/deep/.cache/pip/http' or its parent directory is not owned by the current user and the cache has been disabled. Please check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
WARNING: The directory '/home/deep/.cache/pip' or its parent directory is not owned by the current user and caching wheels has been disabled. check the permissions and owner of that directory. If executing pip with sudo, you may want sudo's -H flag.
Collecting labelImg
  Downloading https://files.pythonhosted.org/packages/91/04/3a48958976f3f82a1d21cdfcd26c87f8c753275a88578b9a9e4ca3a18a93/labelImg-1.8.3-py2.py3-none-any.whl (248kB)
    | 256kB 98kB/s
Requirement already satisfied: PyQt5 in /usr/local/lib/python3.5/dist-packages (from labelImg) (5.10.1)
Requirement already satisfied: lxml in /usr/local/lib/python3.5/dist-packages (from labelImg) (4.2.4)
Requirement already satisfied: sip<4.20,>=4.19.4 in /usr/local/lib/python3.5/dist-packages (from PyQt5->labelImg) (4.19.8)
Installing collected packages: labelImg
Successfully installed labelImg-1.8.3
WARNING: You are using pip version 19.1.1, however version 20.0.2 is available. You should consider upgrading via the 'pip install --upgrade pip' command.

2020-04-24 13:57:18 deep in ~
→labelImg
```

运行软件： 1.单击“打开文件”图标，选择图像目录 ~/paddlepaddle/detect/data/{标签名}/rgb；此时会显示目录下的图像； 2.单击“打开目录”图标，选择图像标签保存目录 ~/paddlepaddle/detect/data/{标签名}/xml； 3.在右侧“使用预设标签”前打勾并在后面输入标签名； 4.单击“创建区块”按钮给当前显示图像打标签； 5.单击“保存”保存当前标签文件，此时弹出对话框，“名称”处不用改动，单击“save”保存 6.然后单击“下一个图片”切换到下一张图片,循环执行 4、5、6 三步直至此文件夹标记完，然后按照切换到下一个文件夹下进行标注。



标记完后，目录结构如下

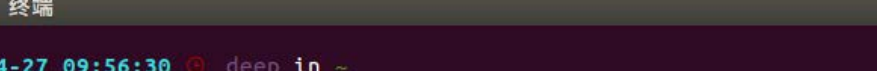
```

---data-----{ 标签 1 }----- rgb   #放标签 1 的图片
                |-----xml   #放标签 1 的标签
                |
                |-----{ 标签 2 }
                |...

```

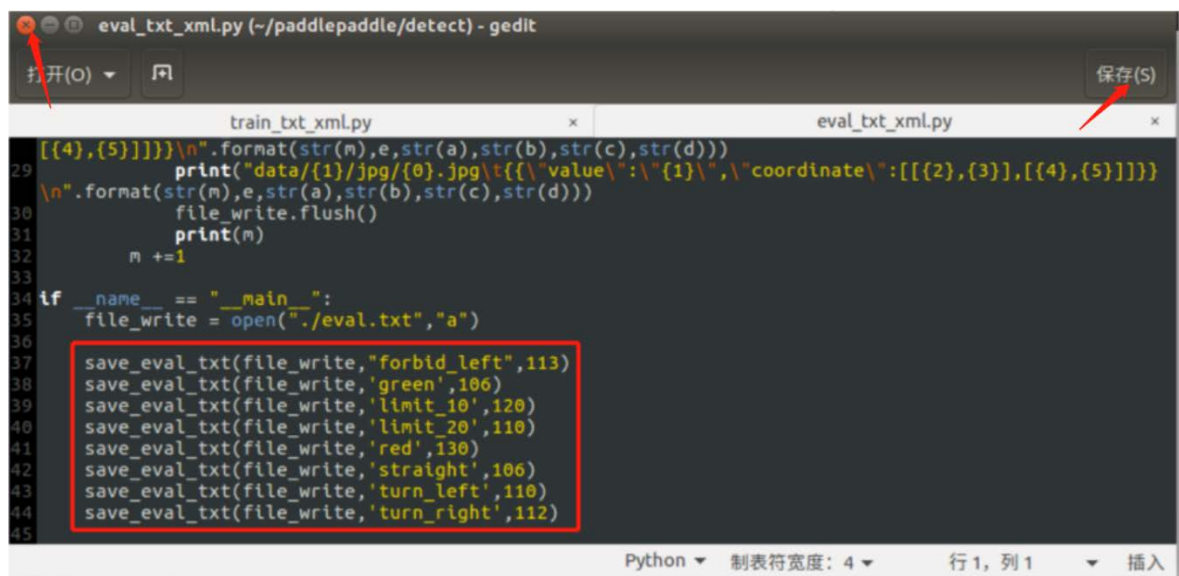
6.2 生成列表文件

ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/detect / 目录下,运行命令 gedit train_txt_xml.py 打开 train_txt_xml.py 脚本；

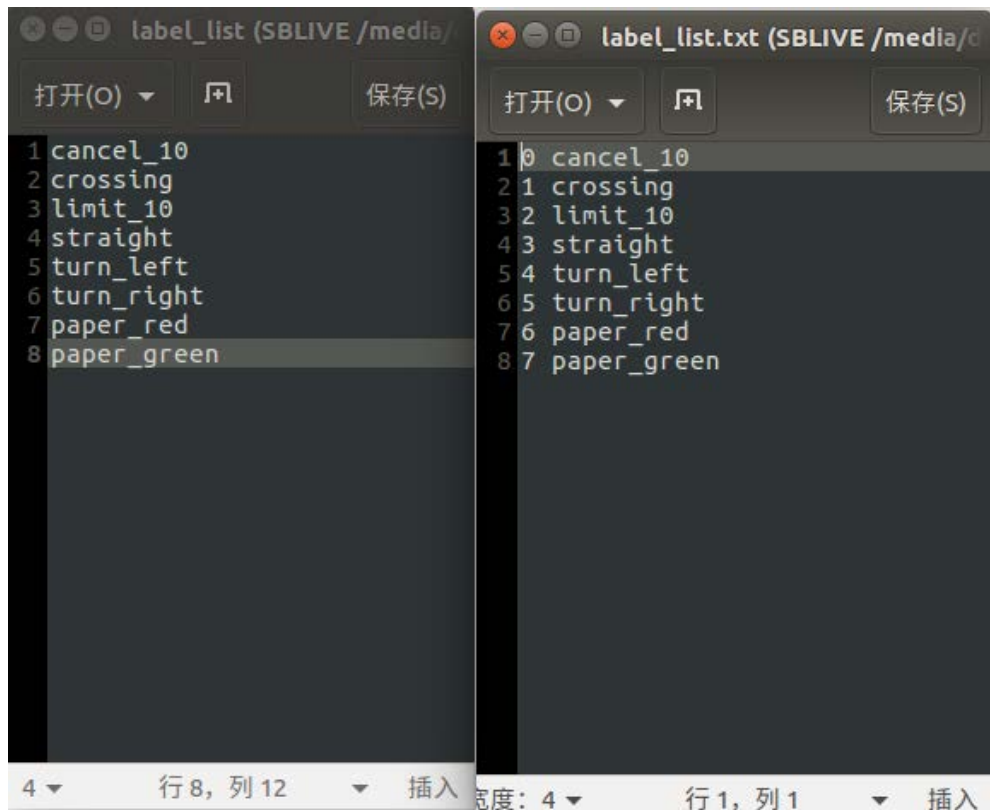


```
2020-04-27 09:56:30 deep in ~  
→cd ~/paddlepaddle/detect/  
2020-04-27 09:56:42 deep in ~/paddlepaddle/detect  
→gedit train_txt_xml.py
```

以 `save_train_txt(file_write.“{标签}”, {采集张数})` 格式，添加采集的数据，如图所示；然后单击“保存”以及 关闭键，结束修改。



ctrl+alt+t 打开终端，使用 cd 命令切换目录到 ~/paddlepaddle/detect/ 目录下，运行命令 python3 train_txt_xml.py 生成 train.txt、eval.txt 文件；在同目录下，运行命令 gedit laxbel.txt 打开 label.txt；将标签分别添加到里面，最后关闭此文件；在同目录下，运行命令 gedit label 打开 label；将标签分别添加到里面，最后关闭此文件；label.txt 和 label 格式如图所示。



最终得到的数据集格式如下所示：

```

---data-----{ 标签 1 }----- rgb          #放标签 1 的图片
                |-----xml          #放标签 1 的标签
                |-----{ 标签 2 }
                |...
                |-----{ 标签* }
                |-----train.txt
                |-----eval.txt
                |-----label
                |-----label.txt

```


任务七：标志物检测_训练模型

本章将利用 AIstudio 线上平台线上资源对模型进行训练；1. 首先需要构建 AIstudio 开源项目；2.上传数据集；3.然后关联数据集和项目；4.训练模型；5.下载训练好的模型。该部分内容需要连接外部网络，在车载电脑或实验室台式机上均可操作。

7.1 构建项目

此次标志物识别的开源项目地址为：
(<https://aistudio.baidu.com/aistudio/projectdetail/298734>) ;进入网址，单击右侧“fork”键，fork 到自己的项目里；



回到 AIstudio 首页，单击标签栏“项目”，然后单击“我的项目”，就可以看到刚才创建的新项目。



7.2 上传数据集

在 AIstudio 官网上，首先单击右上角的登录按钮，登录个人账户；接着单击标题栏的“数据集”；接着单击“创建数据集”，此时会弹出对话框，添加数据集名称，数据集简介；接着单击“上传文件”，接着单击“下一步”在弹出对话框下面选择“创建”，该数据集创建完成。



数据集介绍

12px

数据集背景:
主要包含哪些方面的信息,例如这是1905-2005年间全球电影的评分数据。

数据集内容:
数据集下各文件包含哪些列,每列格式是什么,以及这些内容的时间范围或者统计口径优势是什么。

数据集来源:
您是在哪里获取到这份数据集的。

其他说明:
其他人在使用该数据集时,是否需要注意什么事项。

DIV

上一步 创建 取消

7.3 打开项目

在 11.1 中我们创建了项目,单击创建好的项目;进入到项目中;单击“修改”,跳出编辑界面,单击 2 处删除已有的数据集,单击“添加数据集”选择 11.2 创建的数据集,最后单击“保存”“取消”;接着单击“启动环境”,此时跳出另外一个对话框,在对话框中选择使用 GPU 资源还是 CPU 资源;建议优先使用 GPU 资源,该项目默认代码是在 GPU 环境下运行的;最后单击“确定”。

我的项目 > 无人车教具, 目标检测

无人车教具, 目标检测

直接fork过来的

TobeWell Notebook 高级 计算机视觉 深度学习 公开 36 Python3 2020-03-03 15:18:23

删除 修改

版本内容 数据集 Fork记录 评论(0) 在线服务

启动环境 停止 部署



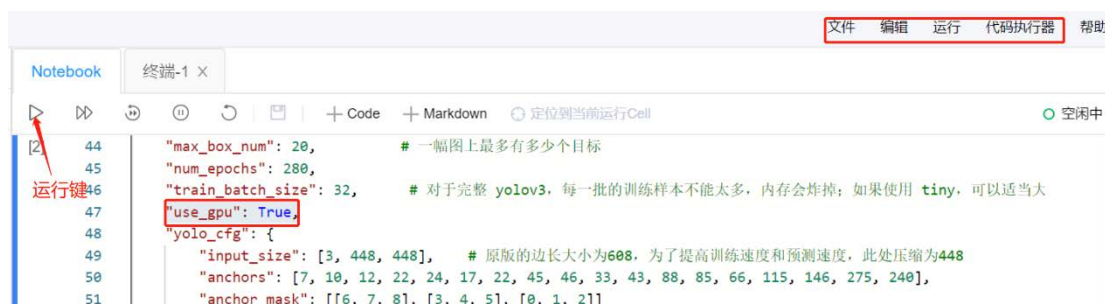
7.4 训练模型

打开项目后，首先在左侧看一下 **data** 目录下包含的数据集；
然后找到自己上传的数据集，将右侧相应位置进行修改；



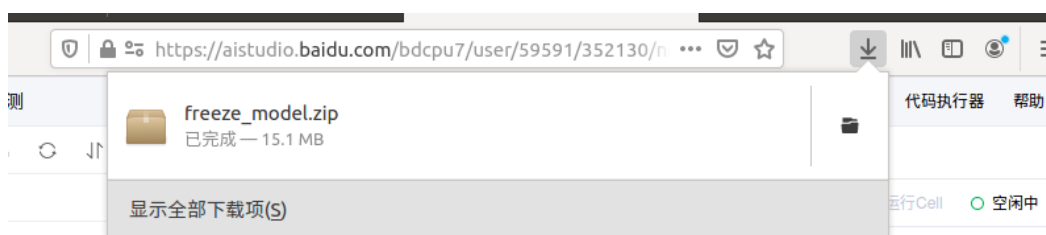
默认使用的是 **GPU** 资源，如果使用 **CPU** 资源，需要把参数改成 `"use_gpu":False`。接着按执行键从第一个开始，逐一运行代码；

如果中途出现问题，需要单击右上角“代码执行器”中的“重启执行器”重启执行器；然后单击“编辑”中的“清除所有输出”，然后重复以上操作。



7.5 下载模型

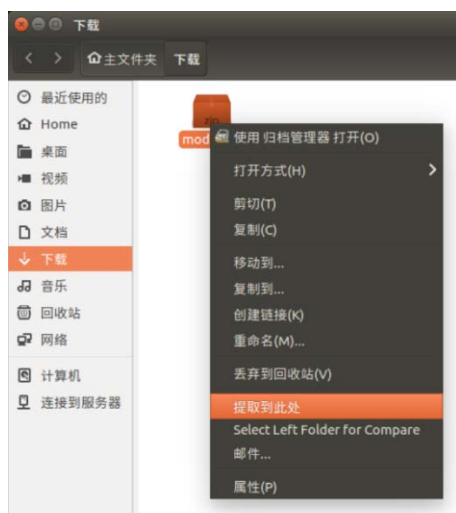
代码运行后在右侧会生成 freeze_model 文件夹；单击“下载按钮”按钮将文件下载到车载电脑上；选中“保存文件”并单击确定；此时保存到本地；通过单击“文件夹”图标可显示当前下载的 freeze_model.zip 文件。



右键压缩文件，单击“提取到此处”进行解压操作，然后将 ~/下载/home/aistudio/ 目录下的 freeze_model 文件复制粘贴到 ~/paddlepaddle/pd/ 目录下。

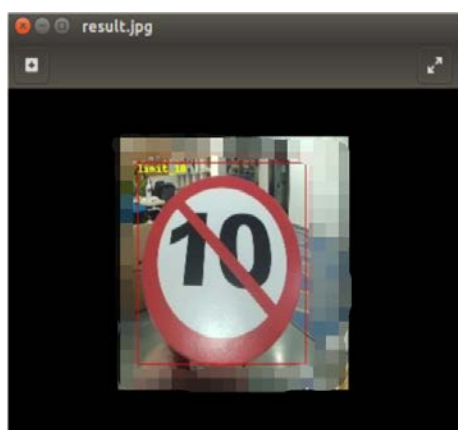
接下来还需要把 ~/paddlepaddle/pd/data 中的 label_list，

label_list.txt 文件进行更新。



7.6 运行结果

ctrl+alt+t 打开终端，使用 `cd` 命令切换目录到 `~/paddlepaddle/pd/` 目录下,其中 `test.py` 实现的功能是测试目标检测部分。运行代码 `python3 test.py` 运行该脚本；代码运行时，在 `~/paddlepaddle/pd/` 下的 `result.jpg` 图像会持续更新，打开该文件，可以看到运行效果；



```
终端
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

2020-04-28 10:04:57 deep in ~
→ cd ~/paddlepaddle/pd/

2020-04-28 10:05:04 deep in ~/paddlepaddle/pd
→ ls
1.jpg          data          freeze_model  result.jpg
1-result.jpg  detector.py   mmd.py        test.py

2020-04-28 10:05:06 deep in ~/paddlepaddle/pd
→ python3 test.py
data/train.txt
data/label_list
label_dict:{0: 'cancel_10', 1: 'crossing', 2: 'limit_10', 3: 'straight', 4: 'turn_left', 5: 'turn_right', 6: 'paper_red', 7: 'paper_green'} class dim:8
luuuu,freeze_model
predict cost time:0.10 sec
box:[]
label:[]
<PIL.JpegImagePlugin.JpegImageFile image mode=RGB size=224x224 at 0x7F6D9F47EF98>
>
send_data:null
predict cost time:0.06 sec
box:[]
```