深度学习入门笔记V1.0.0-2021.1.22

版本	作者	时间	备注
V1.0.0	Zhe Chen	2021.1.22	深度学习入门笔记V1
V1.1.0	Zhe Chen	2021.1.22	增加了训练自己的数据集

1 **RK-PC-001电脑配置**
2 **Ubuntu 下载 typora**
2.1 安装
2.2 导出 PDF, HTML, WORD
2.2.1 配置 pandoc
2.2.2 mathpix snipping tool 下载
2.3 打开typora
3 **Ubuntu 换源**
3.1 备份先前的ubuntu源
3.2 换源
3.2.1 阿里源
3.2.2 清华源
3.3 更新源
4 **Cmake安装与下载**
4.1 卸载
4.2 安装
5 **Nvidia 驱动下载**
6 **Cuda 10.1 下载与安装**
6.1 下载
6.2 安装
7 **Cudcnn 7.6下载**
7.1 Download
7.2 拷贝Cudcnn到cuda文件夹中
7.3 最后检测是否成功安装和查询安装版本
8 **Opencv 下载与安装**
8.1 Install
8.2 opencv环境配置
8.3 opencv-python下载安装
9 **Python3-pip3 下载安装和更新**
9.1 安装
9.2 查看pip3版本
9.3 更新pip3
9.4 pip安装第三方库方法
9.5 pip换源
10 **Pytorch下载安装**
11 **开始第一个深度学校项目**
11.1 Darknet 目标检测
11.1.1 环境配置
11.1.2 YOLOv4 - AlexeyAB
11.1.2.1 编译 make
11.1.2.2 下载权重

11.1.2.3 测试 YOLOv4
11.2 训练自己的数据集
11.2.1 制作自己的数据集----LabelImg

1 RK-PC-001电脑配置

可以通过如下命令获取电脑配置信息：

```
lshw  -short #简略
lshw  #详细
```

这是我的配置信息：

运行内存	处理器	显卡	系统磁盘	硬盘
32GB	Intel® Xeon(R) CPU E5-2678 v3 @ 2.50GHz × 24	GeForce RTX 2080 Ti/PCIe/SSE2（11GB*2）	512GB	4TB+4TB

2 Ubuntu 下载 typora

极力推荐typora文本编辑工具，方便做学习笔记。

2.1 安装

在终端输入以下命令：

```
wget -qO - https://typora.io/linux/public-key.asc | sudo apt-key add - # 添加公钥
sudo add-apt-repository 'deb https://typora.io/linux ./' # 添加typora仓库
sudo apt-get update
sudo apt-get install typora # 安装typora
```

2.2 导出 PDF, HTML, WORD

2.2.1 配置 pandoc

这个包能够输出 PDF, HTML, WORD等文件。

```
sudo apt-get install pandoc
```

2.2.2 mathpix snipping tool 下载

该工具能够截图自动识别获取数学公式：

```
sudo snap install mathpix-snipping-tool
```

2.3 打开typora

- **方法 1**: 可以在菜单中搜索typora
- **方法 2**: 也可以在终端中输入typora

3 Ubuntu 换源

3.1 备份先前的ubuntu源

```
sudo cp /etc/apt/sources.list /etc/apt/sources_init.list
```

将以前的源备份以下，以防以后可以用的。

3.2 换源

```
sudo gedit /etc/apt/sources.list
```

使用gedit打开文档，将下边的阿里源复制进去，然后点击保存关闭。

3.2.1 阿里源

阿里源（Ubuntu 18.04）：

```
deb http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-security main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-updates main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-proposed main restricted universe multiverse

deb http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ bionic-backports main restricted universe multiverse
```

3.2.2 清华源

清华源：

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic main restricted universe multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-updates main restricted universe
multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-backports main restricted universe
multiverse
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-security main restricted universe
multiverse

# 预发布软件源，不建议启用
# deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe multiverse
# deb-src https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ bionic-proposed main restricted universe
multiverse
```

3.3 更新源

更新源：

```
sudo apt-get update
```

修复损坏的软件包，尝试卸载出错的包，重新安装正确版本的。

```
sudo apt-get -f install
```

更新软件：

```
sudo apt-get upgrade
```

4 Cmake安装与下载

4.1 卸载

卸载已经安装的旧版Cmake：

```
sudo apt-get autoremove cmake
```

4.2 安装

下载cmake：

```
sudo wget https://cmake.org/files/v3.12/cmake-3.12.2-Linux-x86_64.tar.gz
```

解压文件：

```
sudo tar zxvf cmake-3.12.2-Linux-x86_64.tar.gz
```

查看解压后目录:

```
tree -L 2 cmake-3.12.2-Linux-x86_64
cmake-3.12.2-Linux-x86_64
├── bin
│   ├── ccmake
│   ├── cmake
│   ├── cmake-gui
│   ├── cpack
│   └── ctest
├── doc
│   └── cmake
├── man
│   ├── man1
│   └── man7
└── share
    ├── aclocal
    ├── applications
    ├── cmake-3.9
    ├── icons
    └── mime
12 directories, 5 files
```

创建软链接:

> 注: 文件路径是可以指定的, 一般选择在/opt 或 /usr 路径下, 这里选择/opt

```
mv cmake-3.12.2-Linux-x86_64 /opt/cmake-3.12.2
ln -sf /opt/cmake-3.12.2/bin/* /usr/bin/
```

然后执行命令检查一下：

```
cmake --version
cmake version 3.12.2
```

5 Nvidia 驱动下载

1. 删除先前的驱动

   ```
   sudo apt-get purge nvidia*
   sudo apt --purge remove "cublas*" "cuda*"
   ```

2. 添加nvidia驱动下载需要的ppa源

   ```
   sudo add-apt-repository ppa:graphics-drivers/ppa
   sudo apt update
   ```

3. 安装

```
ubuntu-drivers devices
#最后根据自己需求安装，这里我安装的是：
sudo apt-get install --no-install-recommends nvidia-driver-460
```

4. 重启然后检查是否安装成功

```
reboot
nvidia-smi
```

5. 如果出现以下情况，则安装成功

```
1. Wed Jan 20 18:58:40 2021
 +-----------------------------------------------------------------------------+
 | NVIDIA-SMI 460.32.03    Driver Version: 460.32.03    CUDA Version: 11.2     |
 |-------------------------------+----------------------+----------------------+
 | GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
 | Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
 |                               |                      |               MIG M. |
 |===============================+======================+======================|
 |   0  GeForce RTX 208...  Off  | 00000000:03:00.0  On |                  N/A |
 | 13%   34C    P8    8W / 257W |    268MiB / 11016MiB |      2%      Default |
 |                               |                      |                  N/A |
 +-------------------------------+----------------------+----------------------+
 |   1  GeForce RTX 208...  Off  | 00000000:04:00.0 Off |                  N/A |
 | 28%   30C    P8    9W / 257W |     10MiB / 11019MiB |      0%      Default |
 |                               |                      |                  N/A |
 +-------------------------------+----------------------+----------------------+

 +-----------------------------------------------------------------------------+
 | Processes:                                                                  |
 |  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
 |        ID   ID                                                   Usage      |
 |=============================================================================|
 |    0   N/A  N/A      1420      G   /usr/lib/xorg/Xorg                 18MiB |
 |    0   N/A  N/A      1505      G   /usr/bin/gnome-shell               72MiB |
 |    0   N/A  N/A      1741      G   /usr/lib/xorg/Xorg                111MiB |
 |    0   N/A  N/A      1875      G   /usr/bin/gnome-shell               35MiB |
 |    0   N/A  N/A      2334      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      2760      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      2806      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      2876      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      3175      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      3226      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      3376      G   /usr/lib/firefox/firefox            2MiB |
 |    0   N/A  N/A      3426      G   /usr/lib/firefox/firefox            2MiB |
 |    1   N/A  N/A      1420      G   /usr/lib/xorg/Xorg                  4MiB |
 |    1   N/A  N/A      1741      G   /usr/lib/xorg/Xorg                  4MiB |
 +-----------------------------------------------------------------------------+\
```

6 Cuda 10.1 下载与安装

6.1 下载

1. 根据自己的系统选择Cuda，这里我选择Cuda10.1，如果下载慢可以直接点击这里下载。
2. 用terminal命令行下载：

```
wget
http://developer.download.nvidia.com/compute/cuda/10.1/Prod/local_installers/cuda_10.1.243_418.87.00_linux.run
```

3. 百度网盘链接，提取码：6666 。(有配套的CUDA与CUDNN)

6.2 安装

1. 下载完`cuda_10.1.243_418.87.00_linux.run`之后`sudo sh cuda_10.1.243_418.87.00_linux.run`这里默认安装路径于：`/usr/local/cuda-10.1`。
2. 添加到bashrc让启动terminal可找到，也就是添加环境变量`sudo vim ~/.bashrc`
3. 在最后插入如下环境变量

```
#added by cuda10.1 installer
export CUDA_HOME=/usr/local/cuda-10.1
export PATH=$CUDA_HOME/bin:$PATH
export LD_LIBRARY_PATH=$CUDA_HOME/lib64:$LD_LIBRARY_PATH
```

4. 使用如下命令检查是否生效

```
source ~/.bashrc
nvcc -V
输出：
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Fri_Jan__8_19:08:17_CDT_2021
Cuda compilation tools, release 10.1, V10.1.105
```

7 Cudcnn 7.6下载

7.1 Download

1. 下载Cudacnn,这里你需要注册才能下载.
2. 百度网盘链接，提取码：6666 。(有配套的CUDA与CUDNN)

7.2 拷贝Cudcnn到cuda文件夹中

拷贝到Cuda文件夹:

```
sudo cp cuda/include/cudnn.h /usr/local/cuda/include/
sudo cp cuda/lib64/libcudnn* /usr/local/cuda/lib64/
sudo chmod a+r /usr/local/cuda/include/cudnn.h
sudo chmod a+r /usr/local/cuda/lib64/libcudnn*
```

7.3 最后检测是否成功安装和查询安装版本

```
cat /usr/local/cuda/include/cudnn.h | grep CUDNN_MAJOR -A 2
```

8 Opencv 下载与安装

8.1 Install

下载opencv 4.5.1和对应版本的opencv_contrib，这里一定要**下载对应版本**，不然很容易遇到错误。可以去opencv官网下载源码。

下载完后，进入opencv文件夹，安装cmake。

```
sudo apt-get install cmake
```

安装需要的依赖库：

```
sudo apt-get install build-essential libgtk2.0-dev libavcodec-dev libavformat-dev libjpeg.dev libtiff5.dev libswscale-dev libjasper-dev
```

创建编译文件夹：

```
mkdir build
```

进入文件夹进行配置：

```
cd build
```

执行cmake命令：

```
cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local ..
```

执行如下命令，编译过程可能会有点慢，耐心等待哦。 这里也可以使用`make -j`、`make -j4`、`make -j8`等命令速度会稍快一些，但如果电脑性能不佳，还是使用`make`命令较好。-j 后的的数字代表线程。

```
sudo make -j8
```

最后，执行命令：

```
sudo make install
```

8.2 opencv环境配置

1. **配置编译环境**
   将OpenCV的库添加到路径，这样的目的是可以让系统找到。

   ```
   sudo gedit /etc/ld.so.conf.d/opencv.conf
   ```

   执行命令后打开的可能是一个空白的文件，直接添加上下面这句代码:

   ```
   /usr/local/lib
   ```

执行下列命令使刚才的配置路径生效:

```
sudo ldconfig
```

配置bash:

```
sudo gedit /etc/bash.bashrc
```

把下列这两句代码，添加在文末处:

```
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig
export PKG_CONFIG_PATH
```

保存后，执行如下命令使配置生效:

```
source /etc/bash.bashrc
```

执行下列命令更新。至此，ubuntu18.04下opencv已经配置完成:

```
sudo updatedb
```

验证是否配置成功:

```
pkg-config --libs opencv4

-L/usr/local/opencv4/lib -lopencv_ml -lopencv_dnn -lopencv_video -lopencv_stitching -
lopencv_objdetect -lopencv_calib3d -lopencv_features2d -lopencv_highgui -lopencv_videoio -
lopencv_imgcodecs -lopencv_flann -lopencv_photo -lopencv_gapi -lopencv_imgproc -lopencv_core
```

8.3 opencv-python下载安装

现在我们需要在python里面安装opencv库

安装依赖项：

安装libopencv-dev依赖包，运行命令`sudo apt install libopencv-dev`，在出现的选项中输入y继续执行就行。

运行`sudo pip3 install opencv-python`命令就行

成功之后，运行python3，进入编译界面，导入库查看版本

```
python3

import cv2
print(cv2.__version__)
```

9 Python3-pip3 下载安装和更新

9.1 安装

```
sudo apt-get install python3-pip
```

9.2 查看pip3版本

```
pip3 --version
# or use `pip3 -V`
```

9.3 更新pip3

```
sudo apt-get install --upgrade pip
```

> **注**：用command安装的pip3包往往是最低版本的，所以一定要查看一下你的pip3包的版本，不然后续`pip3 install package`时，会出现各种问题。

9.4 pip安装第三方库方法

可以进入pip .whl文件离线下载官网，各大主流的库都在里面，这样比直接下载快很多。

9.5 pip换源

pip换源可以提高下载速度，其实这是拿到电脑后要做的第二件事儿。

1. 创建 .pip文件：`mkdir ~/.pip`
2. 进入文件：`cd ~/.pip`
3. 创建pip.conf文件：`touch pip.conf`
4. 编辑pip.conf文件：`sudo gedit ~/.pip/pip.conf`
5. 打开pip.conf文件窗口，将以下内容复制到文件中：

```
[global]
index-url = https://pypi.tuna.tsinghua.edu.cn/simple
[install]
trusted-host=pypi.tuna.tsinghua.edu.cn
```

> pip install -i http://mirrors.aliyun.com/pypi/simple 包名 阿里源
>
> pip install -i https://pypi.mirrors.ustc.edu.cn/simple/ 包名 中科院源
>
> pip install -i https://pypi.hustunique.com/ 包名 华科源

10 Pytorch下载安装

pytorch离线下载地址，选择对应的cuda版本、python版本、操作系统的.whl文件。

下载完.whl文件后，通过`pip3 install name.whl`可以快速安装。

11 开始第一个深度学校项目

11.1 Darknet 目标检测

11.1.1 环境配置

- **system**： Ubuntu 18.04
- **Python**： 3.6.9
- **Opencv**: 4.5.1
- **CUDA**: 10.1
- **GPU**: RTX 2080TI

11.1.2 YOLOv4 - AlexeyAB

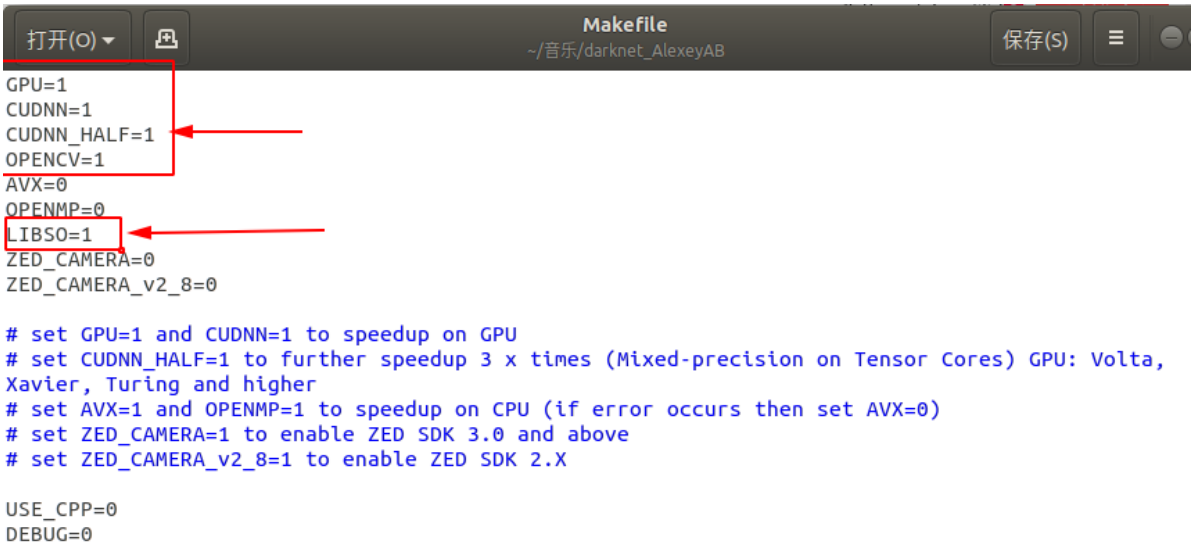
首先下载代码：

```
git clone https://github.com/AlexeyAB/darknet.git
```

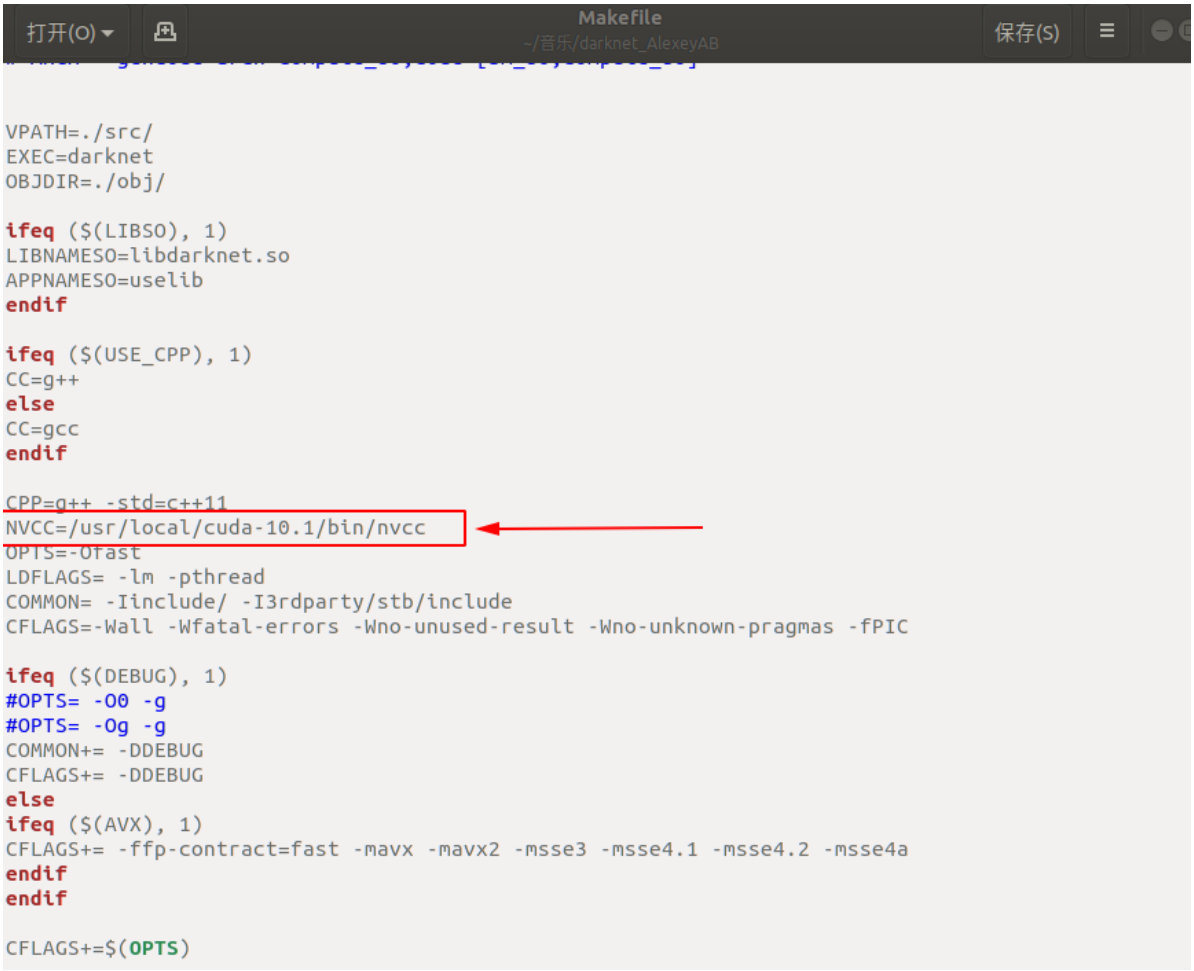
由于都是AlexeyAB大神的杰作，在使用上与YOLOv3使用过程几乎相同。

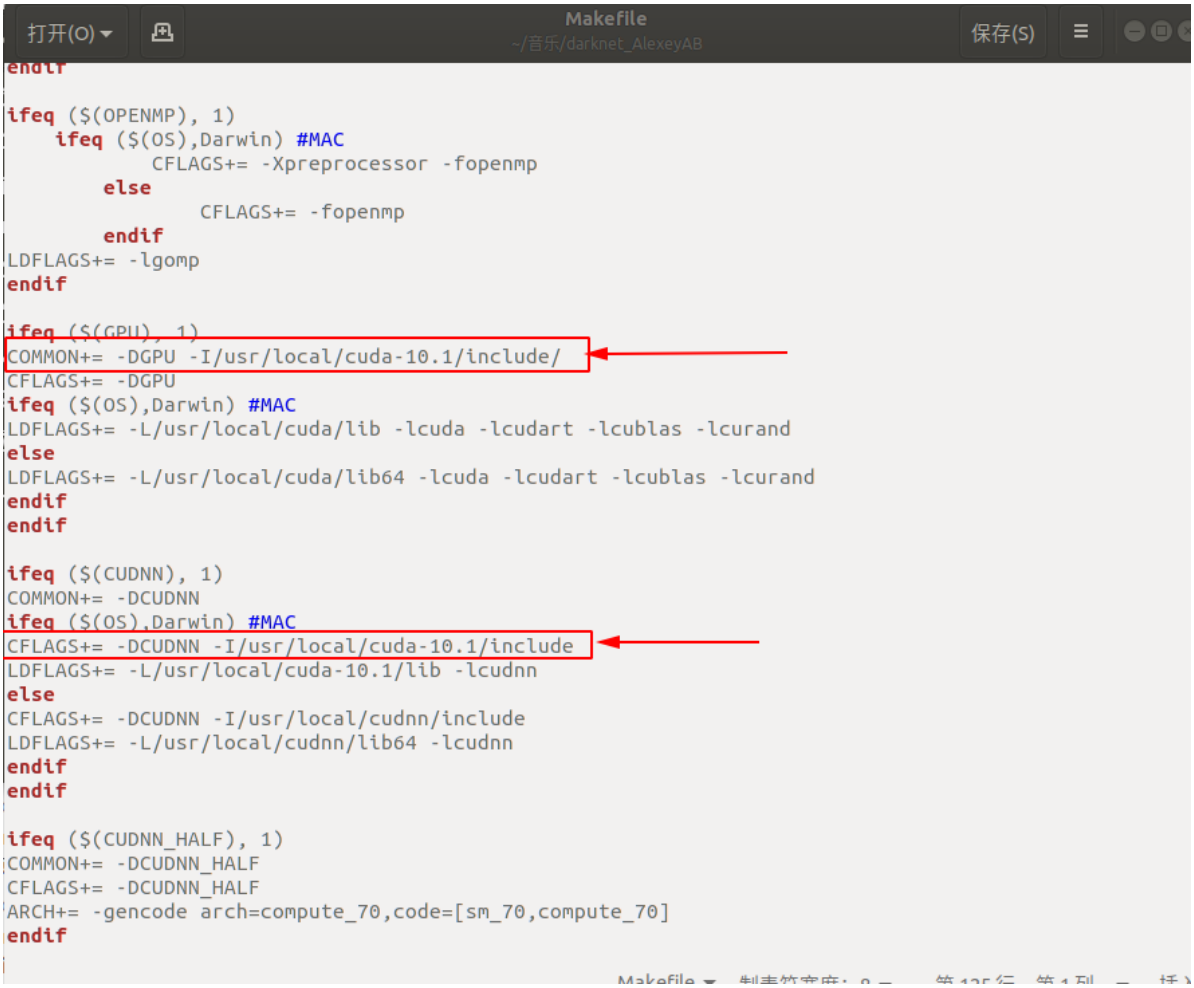
11.1.2.1 编译 make

如果硬件设备包含GPU加速，需要对makefile文件进行修改。训练肯定需要使用GPU加速，那么得打开项目里面的makefile文件修改一些参数的值，makefile文件前面几行：打开GPU 加速，打开opencv，打开libdarknet.so生成开关。



另外，还需要修改NVCC=你自己cuda对应的路径，以及CFLAGS和COMMON对应的CUDA路径。


打开(O)
Makefile
~/音乐/darknet_AlexeyAB
保存(S)
VPATH=./src/
EXEC=darknet
OBJDIR=./obj/
ifeq ($(LIBSO), 1)
LIBNAMESO=libdarknet.so
APPNAMESO=uselib
endif
ifeq ($(USE_CPP), 1)
CC=g++
else
CC=gcc
endif
CPP=g++ -std=c++11
NVCC=/usr/local/cuda-10.1/bin/nvcc
OPTS=-Ofast
LDFLAGS= -lm -pthread
COMMON= -Iinclude/ -I3rdparty/stb/include
CFLAGS=-Wall -Wfatal-errors -Wno-unused-result -Wno-unknown-pragmas -fPIC
ifeq ($(DEBUG), 1)
#OPTS= -O0 -g
#OPTS= -Og -g
COMMON+= -DDEBUG
CFLAGS+= -DDEBUG
else
ifeq ($(AVX), 1)
CFLAGS+= -ffp-contract=fast -mavx -mavx2 -msse3 -msse4.1 -msse4.2 -msse4a
endif
endif
CFLAGS+=$(OPTS)


打开(O)
Makefile
~/音乐/darknet_AlexeyAB
保存(S)
endif
ifeq ($(OPENMP), 1)
ifeq ($(OS),Darwin) #MAC
CFLAGS+= -Xpreprocessor -fopenmp
else
CFLAGS+= -fopenmp
endif
LDFLAGS+= -lgomp
endif
ifeq ($(GPU), 1)
COMMON+= -DGPU -I/usr/local/cuda-10.1/include/
CFLAGS+= -DGPU
ifeq ($(OS),Darwin) #MAC
LDFLAGS+= -L/usr/local/cuda/lib -lcuda -lcudart -lcublas -lcurand
else
LDFLAGS+= -L/usr/local/cuda/lib64 -lcuda -lcudart -lcublas -lcurand
endif
endif
ifeq ($(CUDNN), 1)
COMMON+= -DCUDNN
ifeq ($(OS),Darwin) #MAC
CFLAGS+= -DCUDNN -I/usr/local/cuda-10.1/include
LDFLAGS+= -L/usr/local/cuda-10.1/lib -lcudnn
else
CFLAGS+= -DCUDNN -I/usr/local/cudnn/include
LDFLAGS+= -L/usr/local/cudnn/lib64 -lcudnn
endif
endif
ifeq ($(CUDNN_HALF), 1)
COMMON+= -DCUDNN_HALF
CFLAGS+= -DCUDNN_HALF
ARCH+= -gencode arch=compute_70,code=[sm_70,compute_70]
endif

然后在终端进行编译：

```
# cd到darknet文件夹下：
make # 或make -j8
```

11.1.2.2 下载权重

yolov4.weights: https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights，**下载后放在主目录下**。

11.1.2.3 测试 YOLOv4

使用与训练的权重进行测试，这里我使用了USB摄像头进行cam调试：

```
./darknet detector demo cfg/coco.data cfg/yolov4.cfg yolov4.weights -c 0
```

以上是使用USB摄像头进行测试实时的目标检测，如果有兴趣可以去AlexeyAB作者github网址寻找webcam进行IP摄像头视频在线实时检测。

11.2 训练自己的数据集

11.2.1 制作自己的数据集----LabelImg

LabelImg github 地址：https://github.com/tzutalin/labelImg

Ubuntu Linux python3 +Qt5:

```
git clone https://github.com/tzutalin/labelImg
sudo apt-get install pyqt5-dev-tools
sudo pip3 install -r requirements/requirements-linux-python3.txt
make qt5py3
python3 labelImg.py
```