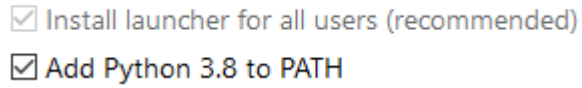
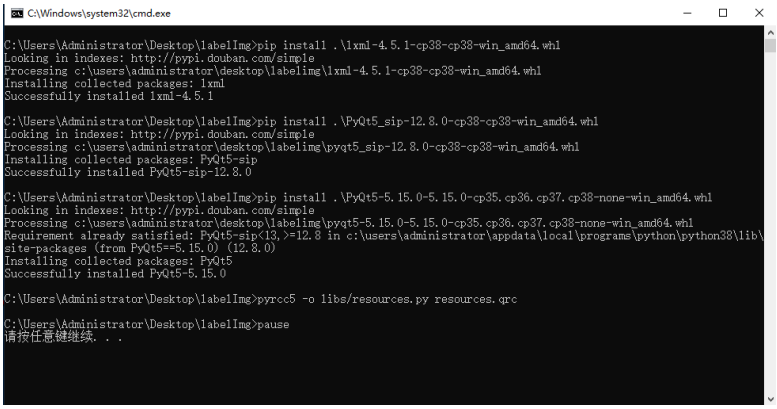
1 labelImg图片标注

1.1 安装

1. 拷贝 `labelImg.zip` 并解压至想要安装的位置
2. 双击 `python_install.exe` 安装 python 环境
3. 勾选 “Add Python 3.8 to path” 的选项



4. 点击 “Install Now”
5. 双击执行 `install.bat` 来安装和编译 labelImg(有的时候如果卡住就按一下回车, 是 windows 快速编辑模式的问题)

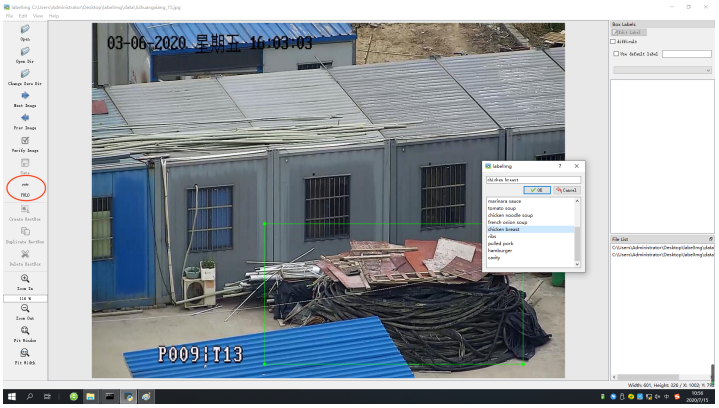


1.2 预处理

1. 在 `data\predefined_classes.txt` 提供预设类别
2. 预设类别不可以删除但可以增加, 因为删除会影响之前的次序

1.3 使用

1. 双击执行 `Run.bat`
2. 点 PascalVOC 切换成 YOLO 模式
3. Open Dir 打开图片所在文件夹



4. 快捷键

按键	作用
Ctrl s	保存
w	新建标注框

按键	作用
d	下一张图
a	上一张图
ctrl +	放大
ctrl -	缩小

5. 标注完成的图片会生成一个同名的`.txt` 后缀的文件
6. 其中第一个数字是类别 (从 0 开始), 后四个为框的坐标

2 Darknet图像识别

尽管原始作者是 pjreddie, 考虑到对 windows 的兼容性, 使用 AlexeyAB 的版本

2.1 安装

首先拷贝 darknet_en.zip 并解压至想要安装的位置, cn 版操作相同

2.1.1 Linux

1. 用编辑器打开 Makefile
2. `GPU=0, CUDNN=0, OPENCV=0` 这三个选项, 如果装了就改成 1
3. cd 到 `darknet_en` 下执行 `make`
4. 当前目录即是最终文件包, 二进制文件是`./darknet`

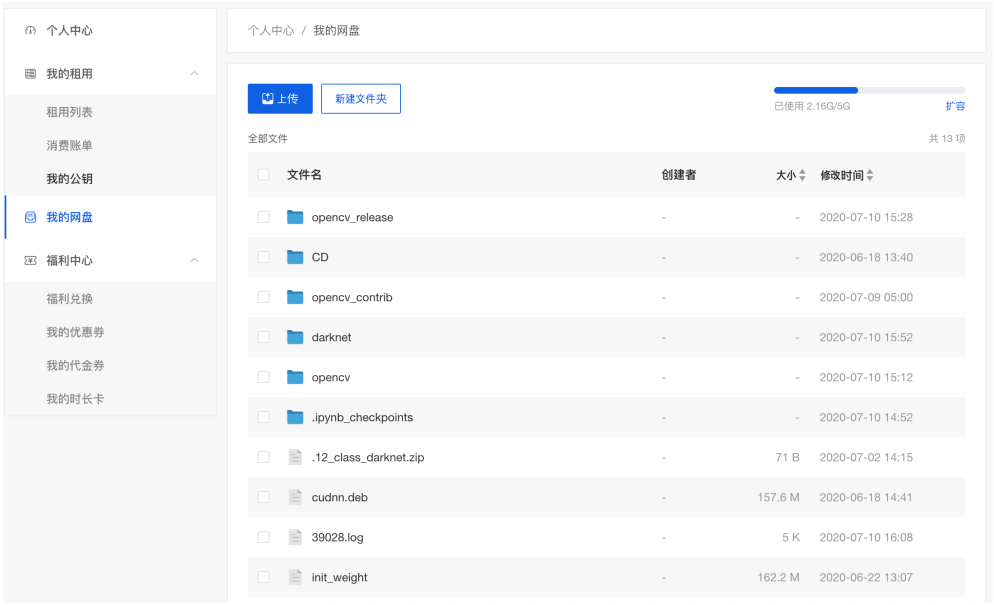
2.1.2 Windows

1. 安装 VisualStudio 2015 或更新版本
2. 用 VS 打开 `build\darknet\darknet_no_gpu.sln` 或者 `darknet.sln`(取决于有没有 GPU)
3. VS 菜单执行 `Build>Build darknet`
4. 编译完成, 最终文件包是 `build\darknet\x64`, 二进制文件是`.\darknet.exe` 或`.\darknet_no_gpu.exe`

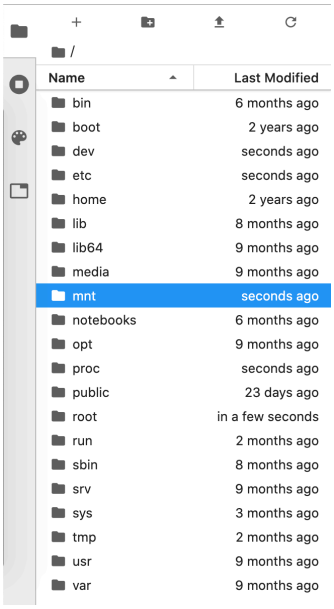
2.2 Matpool 使用

1. 注册 (使用我的二维码可以优惠)

2. 主机市场租用机器, 以小时计费
3. 预先将需要使用的 darknet 包以及初始 weight 放到我的网盘

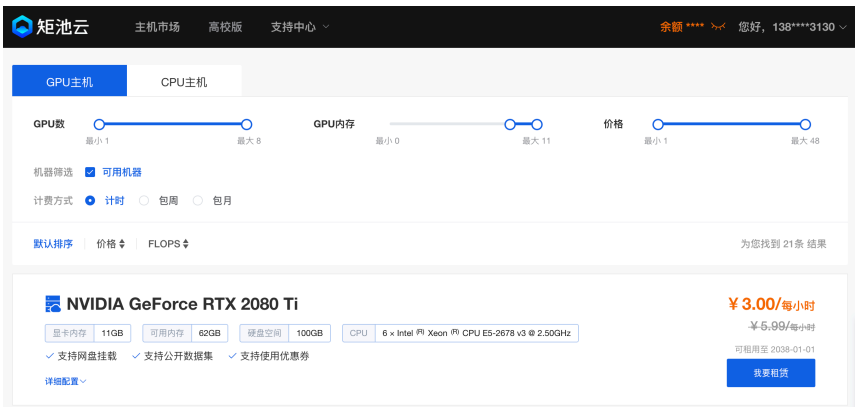


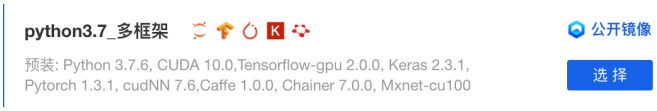
4. 每当租用机器时, 网盘会挂在到/mnt 的位置



2.3 训练 (GPU)

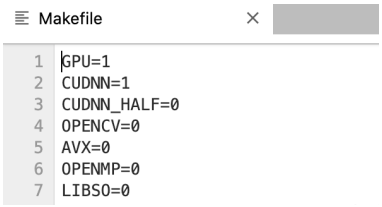
1. 租用 2080Ti, 镜像选择 “python3.7_ 多框架”



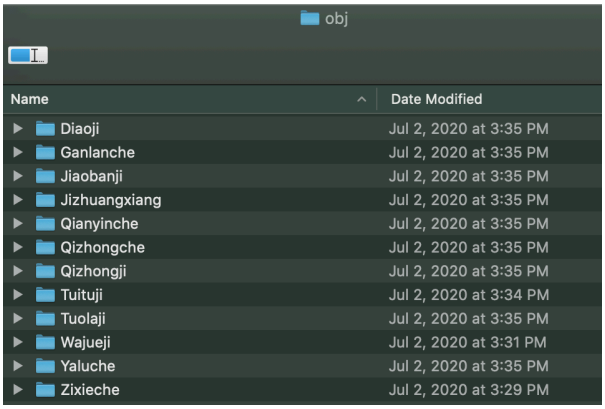


2. 机器启动后在浏览器打开 jupyterlab 的链接

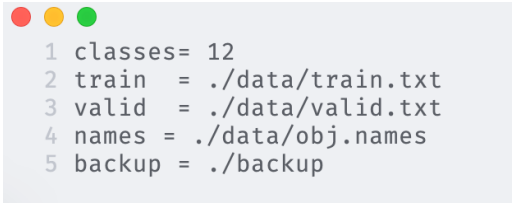
3. 编译 darknet(GPU 和 CUDNN 需要改成 1)



4. 将图片和标注数据按类别放在 `data/obj` 下, 没有被标注的图像将被忽略



4. 修改 `data/obj.data` 中的 `classes` 数



5. `data/obj.names` 应当与 labelImg 中的 `data\predefined_classes.txt` 相同
6. 修改 `data/obj.cfg`
   - `max_batches` 为 `classes`*2000
   - `steps` 为 80% 和 90% 的 `max_batches`

   - 寻找 3 个 `[yolo]` 组块修改 `classes` 数

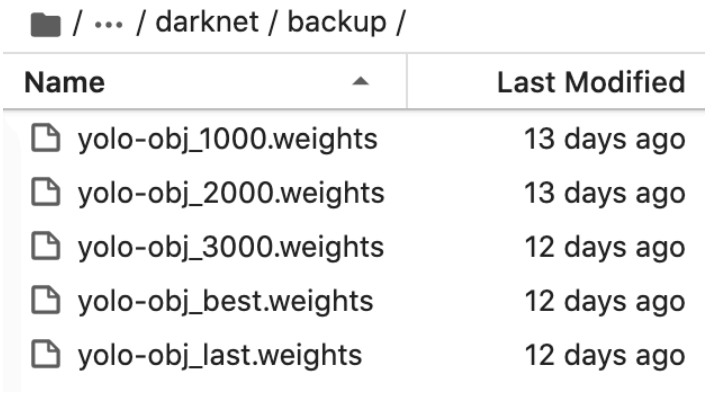
   - 仅在 **yolo** 上面的 **[convolutional]** 组块修改 `filters` 为 `(classes + 5)*3`

```
[convolutional]
size=1
stride=1
pad=1
filters=51
activation=linear


[yolo]
```

- 举例: classes=3, max_batches=6000, steps=4800,5400, filters=24

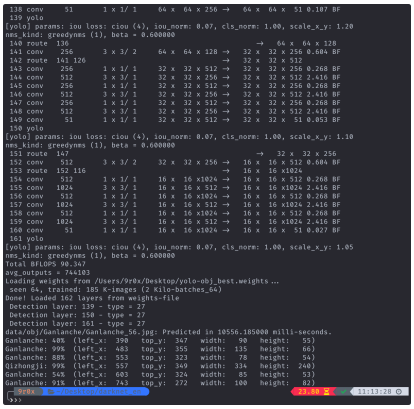
7. cd 到 `data` 下, 运行 `python helper.py`
8. 拷贝初始 weight `init_weight` 到 `darknet_en` 下
9. cd 到 `darknet_en` 下, 运行`./darknet detector train data/obj.data data/obj.cfg init_weight -map` 即开始训练
10. 每 1000batch, 最新 batch 以及结果最好的 batch 的权重会放在 `backup` 下



11. 1000batch 后每 100batch 进行 mAP 计算, 显示准确率 accuracy
12. 最优 accuracy 一般可以达到 60% 以上 (3000+batch)

2.4 预测

1. 安装对应系统的 darknet
2. 中文预测使用 `darknet_cn`, 需 cd 到 data/labels 下删除旧 label 并运行 `python make_label.py`(其中需手动安装 ImageMagick 的 convert 命令并自己设定字体路径 所以建议在 Linux 上完成). 生成 labels 之后拷贝到 windows 上亦可以使用
3. cd 到 `darknet_en` 下, 运行`./darknet detector test data/obj.data data/obj.cfg best.weights -thresh 0.2 -ext_output data/obj/Ganlanche/Ganlanche_56.jpg`



3. 命令格式为 `./darknet detector test [obj.data 的位置] [obj.cfg 的位置] [weights 的位置] -thresh [阈值] -ext_output [预测图片的位置]`
4. `-ext_output` 可以用来输出每个框的位置
5. 预测多张图片 `./darknet detector test [obj.data 的位置] [obj.cfg 的位置] [weights 的位置] -thresh [阈值] -ext_output -dont_show -out result.json < input.txt` 其中 `input.txt` 是包括一行一个图片路径的文本文件

2.5 注意

1. windows 的软件包在 `build\darknet\x64`, 所以上文涉及到 data/等的相对路径指的是在这个目 录下相对的 data 而不是 darknet 下的 data
2. `darknet_cn/data/data.names` 为 Linux/Mac 支持的 UTF-8 编码, 而 `darknet_cn/build/darknet/x64/data.names` 为支持 windows 采用了 GB2312 编码

3 Flask服务器端

3.1 安装 python 及 flask(视情况使用 sudo)

```
apt update && apt install -y python3.8 python3-pip
pip install flask
```

3.2 部署

1. 拷贝 darknet 和 weight 并修改 MakeFile 中的 `LIBSO=1`(其他参数按需修改)
2. 用 `make` 命令编译
3. 确认当前文件夹下有 libdark.so 或 libdarknet.so
4. 拷贝 app.py 至 darknet 下
5. 修改 app.py 11 行.so 文件的绝对路径
6. 运行

```
export FLASK_APP=app.py
flask run --host=0.0.0.0
```