

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

☒ Install launcher for all users (recommended)

☒ Add Python 3.8 to PATH

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)

```
C:\Windows\system32\cmd.exe
C:\Users\Administrator\Desktop\labelImg>pip install .\lxml-4.5.1-cp38-cp38-win_amd64.whl
Looking in indexes: http://pypi.douban.com/simple
Processing c:\users\administrator\desktop\labelimg\lxml-4.5.1-cp38-cp38-win_amd64.whl
Installing collected packages: lxml
Successfully installed lxml-4.5.1

C:\Users\Administrator\Desktop\labelImg>pip install .\PyQt5sip-12.8.0-cp38-cp38-win_amd64.whl
Looking in indexes: http://pypi.douban.com/simple
Processing c:\users\administrator\desktop\labelimg\pyqt5sip-12.8.0-cp38-cp38-win_amd64.whl
Installing collected packages: PyQt5sip
Successfully installed PyQt5sip-12.8.0

C:\Users\Administrator\Desktop\labelImg>pip install .\PyQt5-5.15.0-5.15.0-cp35.cp36.cp37.cp38-none-win_amd64.whl
Looking in indexes: http://pypi.douban.com/simple
Processing c:\users\administrator\desktop\labelimg\pyqt5-5.15.0-5.15.0-cp35.cp36.cp37.cp38-none-win_amd64.whl
Requirement already satisfied: PyQt5sip<13,>=12.8 in c:\users\administrator\appdata\local\programs\python\python38\lib\site-packages (from PyQt5==5.15.0) (12.8.0)
Installing collected packages: PyQt5
Successfully installed PyQt5-5.15.0

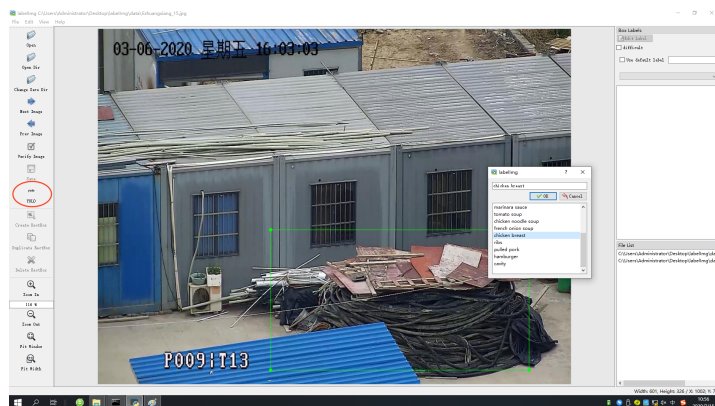
C:\Users\Administrator\Desktop\labelImg>pyrcc5 -o libs/resources.py resources.qrc
C:\Users\Administrator\Desktop\labelImg>pause
请按任意键继续. . .
```

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 切换到 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

- 标注完成的图片会生成一个同名的.txt 后缀的文件
- 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

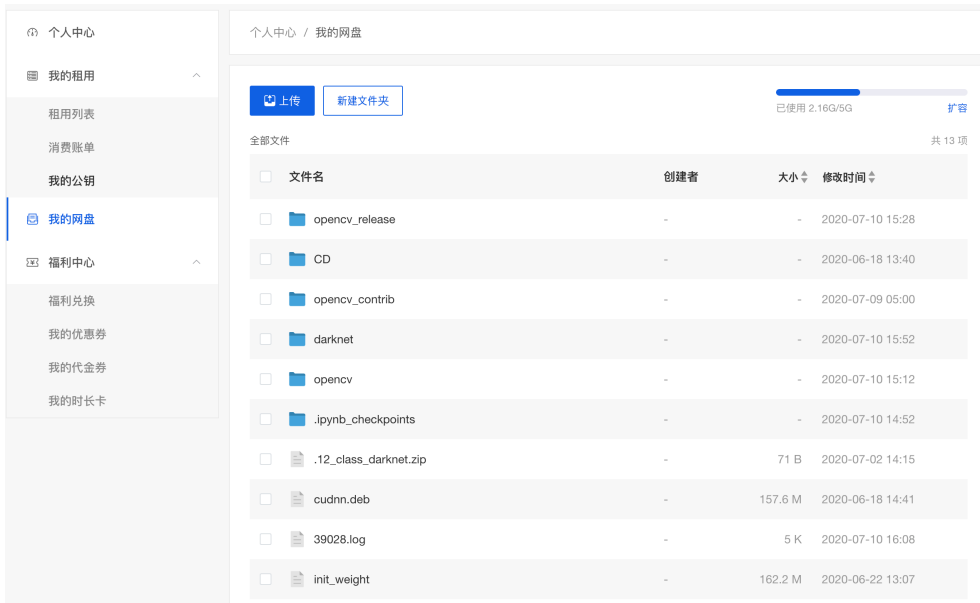
2.2 Matpool 使用

1. 注册 (使用我的二维码可以优惠)

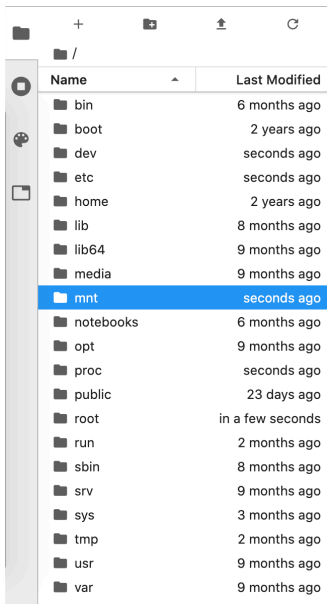


2. 主机市场租用机器, 以小时计费
3. 预先将需要使用的 darknet 包以及初始 weight 放到我的网盘



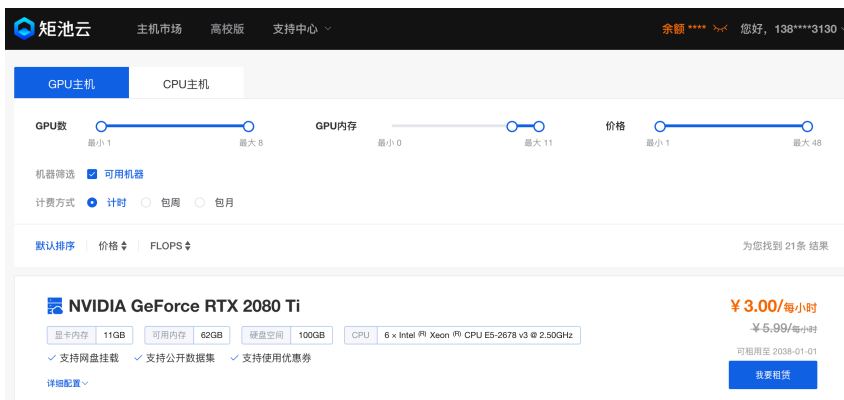


4. 每当租用机器时, 网盘会挂在到/mnt 的位置



2.3 训练 (GPU)

1. 租用 2080Ti, 镜像选择 “python3.7_ 多框架”



python3.7_多框架

公开镜像

预装: Python 3.7.6, CUDA 10.0, Tensorflow-gpu 2.0.0, Keras 2.3.1, Pytorch 1.3.1, cudNN 7.6, Caffe 1.0.0, Chainer 7.0.0, Mxnet-cu100

选择

2. 机器启动后在浏览器打开 jupyterlab 的链接

NVIDIA GeForce RTX 2080 Ti

运行中

停止并释放

长时间运行推荐使用SSH, 该链接非长期有效链接, 偶尔会因网络问题而变更, 请留意地址云通知。

SSH 链接: <ssh://hz-12.matpool.com:27432>

JupyterLab 链接: <https://hz-12.matpool.com:27124?token=GANF6uz9X>

3. 编译 darknet(GPU 和 CUDNN 需要改成 1)

Makefile

```
1 GPU=1
2 CUDNN=1
3 CUDNN_HALF=0
4 OPENCV=0
5 AVX=0
6 OPENMP=0
7 LIBS0=0
```

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

Name	Date Modified
▶ Diaoji	Jul 2, 2020 at 3:35 PM
▶ Ganlanche	Jul 2, 2020 at 3:35 PM
▶ Jiaobanji	Jul 2, 2020 at 3:35 PM
▶ Jizhuangxiang	Jul 2, 2020 at 3:35 PM
▶ Qianyinche	Jul 2, 2020 at 3:35 PM
▶ Qizhongche	Jul 2, 2020 at 3:35 PM
▶ Qizhongji	Jul 2, 2020 at 3:35 PM
▶ Tuituji	Jul 2, 2020 at 3:34 PM
▶ Tuolaji	Jul 2, 2020 at 3:35 PM
▶ Wajueji	Jul 2, 2020 at 3:31 PM
▶ Yaluche	Jul 2, 2020 at 3:35 PM
▶ Zixieche	Jul 2, 2020 at 3:29 PM

4. 修改 data/obj.data 中的 classes 数

```
1 classes= 12
2 train = ./data/train.txt
3 valid = ./data/valid.txt
4 names = ./data/obj.names
5 backup = ./backup
```

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

```
max_batches = 24000
policy=steps
steps=19200,21600
```

- 寻找 3 个 [yolo] 组块修改 classes 数

```
[yolo]
mask = 6,7,8
anchors = 12, 16, 19, 36, 40, 28, 36, 75, 76, 55, 72, 146, 142, 110, 192,
243, 459, 481
classes=10
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
```

- 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

📁 / ... / darknet / backup /

Name	Last Modified
📄 yolo-obj_1000.weights	13 days ago
📄 yolo-obj_2000.weights	13 days ago
📄 yolo-obj_3000.weights	12 days ago
📄 yolo-obj_best.weights	12 days ago
📄 yolo-obj_last.weights	12 days ago

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`



```

130 conv 51 1 x 3/1 64 x 64 x 256 → 64 x 64 x 51 0.187 BF
130 yolo
[total params: iou loss: cloc (4), iou_norm: 0.07, cls_norm: 1.00, scale_x_y: 1.20
msa_kind: greedyms (1), beta = 0.600000]
131 route 126
131 conv 256 3 x 3/2 64 x 64 x 128 → 32 x 32 x 256 0.604 BF
132 route 131 126
132 conv 256 1 x 3/1 32 x 32 x 512 → 32 x 32 x 256 0.268 BF
133 conv 512 3 x 3/1 32 x 32 x 256 → 32 x 32 x 512 2.416 BF
134 conv 256 1 x 3/1 32 x 32 x 512 → 32 x 32 x 256 0.368 BF
135 conv 512 3 x 3/1 32 x 32 x 256 → 32 x 32 x 512 2.416 BF
136 conv 256 1 x 3/1 32 x 32 x 512 → 32 x 32 x 256 0.368 BF
137 conv 512 3 x 3/1 32 x 32 x 256 → 32 x 32 x 512 2.416 BF
138 conv 51 1 x 3/1 32 x 32 x 512 → 32 x 32 x 51 0.403 BF
138 yolo
[total params: iou loss: cloc (4), iou_norm: 0.07, cls_norm: 1.00, scale_x_y: 1.20
msa_kind: greedyms (1), beta = 0.600000]
139 route 147
139 conv 512 3 x 3/2 32 x 32 x 256 → 16 x 16 x 512 0.604 BF
140 route 139 146
140 conv 512 1 x 3/1 16 x 16 x 1024 → 16 x 16 x 512 0.268 BF
141 conv 1024 3 x 3/1 16 x 16 x 512 → 16 x 16 x 1024 2.416 BF
142 conv 512 1 x 3/1 16 x 16 x 1024 → 16 x 16 x 512 0.268 BF
143 conv 1024 3 x 3/1 16 x 16 x 512 → 16 x 16 x 1024 2.416 BF
144 conv 512 1 x 3/1 16 x 16 x 1024 → 16 x 16 x 512 0.268 BF
145 conv 1024 3 x 3/1 16 x 16 x 512 → 16 x 16 x 1024 2.416 BF
146 conv 51 1 x 3/1 16 x 16 x 1024 → 16 x 16 x 51 0.407 BF
146 yolo
[total params: iou loss: cloc (4), iou_norm: 0.07, cls_norm: 1.00, scale_x_y: 1.05
msa_kind: greedyms (1), beta = 0.600000]
Total BIPS: 90.467
avg_outputs = 74183
loading weights from Users/Darknet/yolo-obj.weights...
seen 64, trained 185 K-images (2 kilo-batches_64)
None Loaded 102 layers from weights-file
Detection layer: 139 - type = 27
Detection layer: 138 - type = 27
Detection layer: 161 - type = 27
data2obj/yoloobj/Ganlanche_Sc.jpg Predicted in 18556.185000 milli-seconds:
Ganlanche 486 (left_x: 388 top_y: 347 width: 90 height: 55)
Ganlanche 994 (left_x: 480 top_y: 200 width: 150 height: 60)
Ganlanche 886 (left_x: 553 top_y: 323 width: 78 height: 94)
Qizhongji 994 (left_x: 557 top_y: 340 width: 134 height: 240)
Ganlanche 546 (left_x: 680 top_y: 224 width: 85 height: 53)
Ganlanche 916 (left_x: 743 top_y: 272 width: 100 height: 62)

```

- 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]`
- `-ext_output` 可以用来输出每个框的位置
- 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

- 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`
- `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 LIBS0=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
```