Win10 下 Darknet 和 Yolov4 的使用

--20192131077 张帆

一、环境准备

1.1 本机环境

本机使用的是 Rtx2060 Super 显卡进行训练，CPU 使用 AMD R5 3600

```
C:\Users\Fanz>nvidia-smi
Tue Jun 29 19:12:43 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 456.71       Driver Version: 456.71       CUDA Version: 11.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name            TCC/WDDM | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce RTX 206... WDDM  | 00000000:07:00.0  On |                  N/A |
| 39%   32C    P8    18W / 175W |    784MiB /  8192MiB |      4%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      1484    C+G   Insufficient Permissions        N/A      |
|    0   N/A  N/A      9184    C+G   ...w5n1h2txyewy\SearchUI.exe    N/A      |
|    0   N/A  N/A     26932    C+G   ...y\Shell1ExperienceHost.exe    N/A      |
|    0   N/A  N/A     31768    C+G   ...es.TextInput.InputApp.exe    N/A      |
|    0   N/A  N/A     33260    C+G   ...cw5n1h2txyewy\LockApp.exe    N/A      |
|    0   N/A  N/A     34060    C+G   C:\Windows\explorer.exe         N/A      |
|    0   N/A  N/A     34416    C+G   ...artMenuExperienceHost.exe    N/A      |
|    0   N/A  N/A     36916    C+G   ...ge\Application\msedge.exe    N/A      |
|    0   N/A  N/A     42692    C+G   ...wekyb3d8bbwe\Music.UI.exe    N/A      |
|    0   N/A  N/A     43192    C+G   ...kyb3d8bbwe\Calculator.exe    N/A      |
|    0   N/A  N/A     47528    C+G   ...e6\promecefpluginhost.exe    N/A      |
|    0   N/A  N/A     48548    C+G   ...wekyb3d8bbwe\Video.UI.exe    N/A      |
|    0   N/A  N/A     53052    C+G   ...8\office6\photolaunch.exe    N/A      |
|    0   N/A  N/A     59712    C+G   ...6\extracted\WeChatApp.exe    N/A      |
|    0   N/A  N/A     61584    C+G   ...tracted\WechatBrowser.exe    N/A      |
|    0   N/A  N/A     61952    C+G   ...1Panel\SystemSettings.exe    N/A      |
+-----------------------------------------------------------------------------+
```

1.2 安装 Cuda 和 Cudnn

因为之前已经配置好了，所以过程不多赘述

```
C:\Users\Fanz>nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2020 NVIDIA Corporation
Built on Wed_Jul_22_19:09:35_Pacific_Daylight_Time_2020
Cuda compilation tools, release 11.0, V11.0.221
Build cuda_11.0_bu.relgpu_drvr445TC445_37.28845127_0
```

安装 Cuda 地址：

https://developer.nvidia.com/cuda-downloads?target_os=Windows&target_arch=x86_64&target_version=10

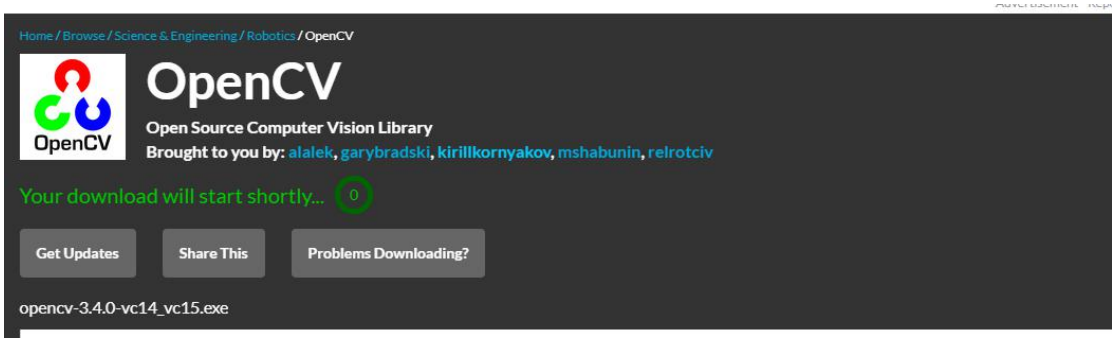
安装 Cudnn 地址：
https://developer.nvidia.com/cudnn

具体安装教程：
https://blog.csdn.net/u010618587/article/details/82940528?utm_medium=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-2.control&depth_1-utm_source=distribute.pc_relevant.none-task-blog-2%7Edefault%7EBlogCommendFromMachineLearnPai2%7Edefault-2.control

需要注意的是 Cuda 和 Cudnn 版本的对应

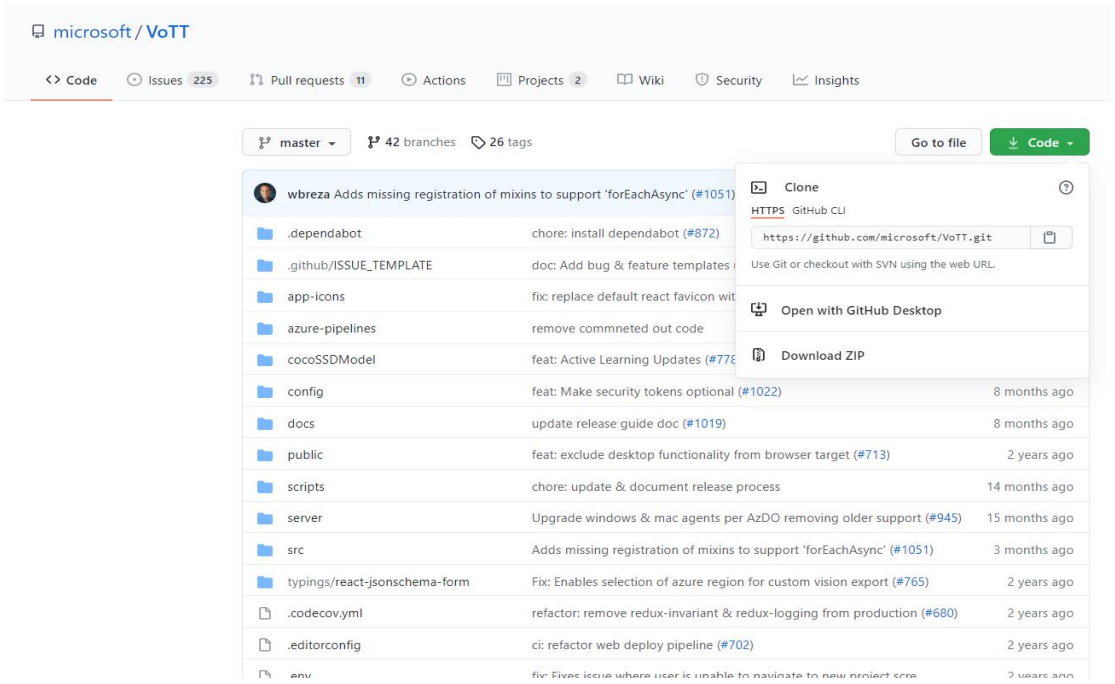
1.3 Opencv

我安装的是 opencv 4.5.2 最新版本



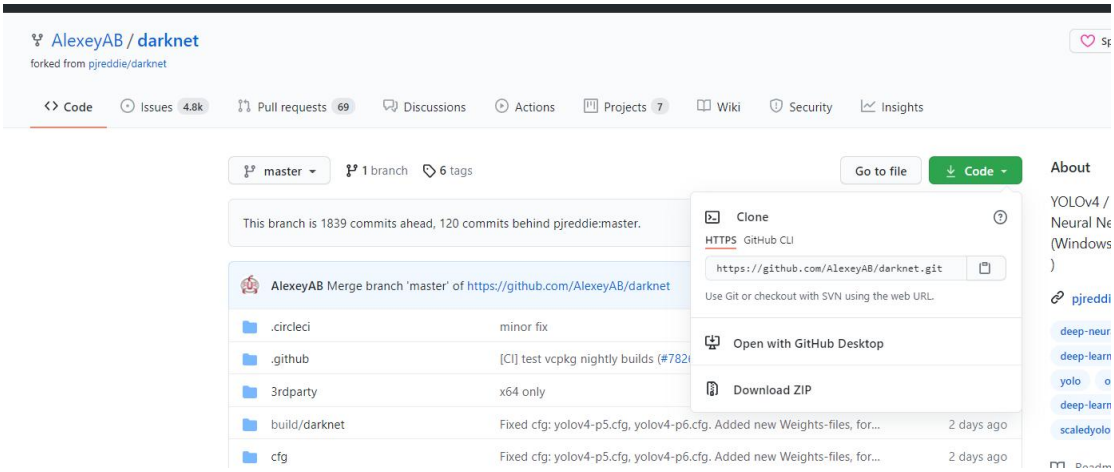
安装地址:
https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.4.0/opencv-3.4.0-vc14_vc15.exe/download

1.4 Vott （标注数据集）

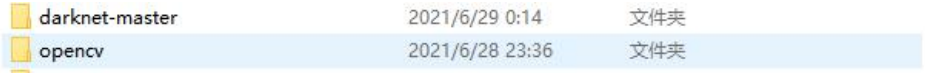


地址：https://github.com/microsoft/VoTT 直接 DownloadZip，解压后安装

1.5 Draknet



地址：https://github.com/AlexeyAB/darknet 也是下载下来，解压，最后我把它们两个都放在 D 盘



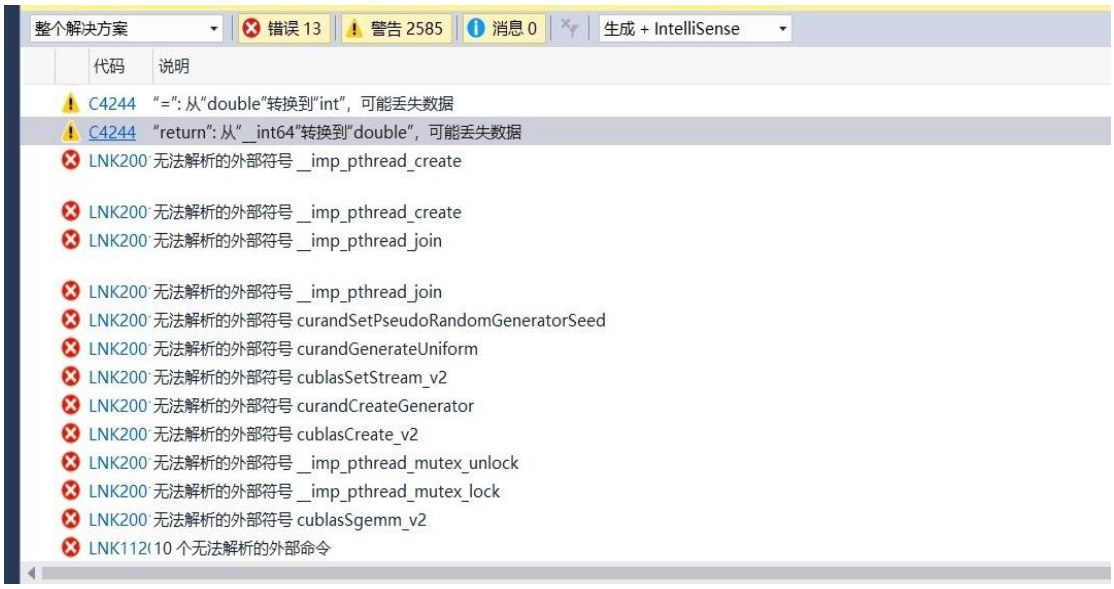
1.6 环境适配

首先打开 darknet-master 的 build 文件夹下的 darknet 文件夹，并用编辑器（sublime）打开 darknet.vcxproj，用搜索功能查找 CUDA （这里就是 CUDA 的版本号了），并修改为自己 CUDA 的版本号，我的 CUDA 版本是 11.0，所以修改为 CUDA 11.0（一共有两处需要修改的）并保存

剩下的步骤可以按着 https://zhuanlan.zhihu.com/p/45845454 继续照着做

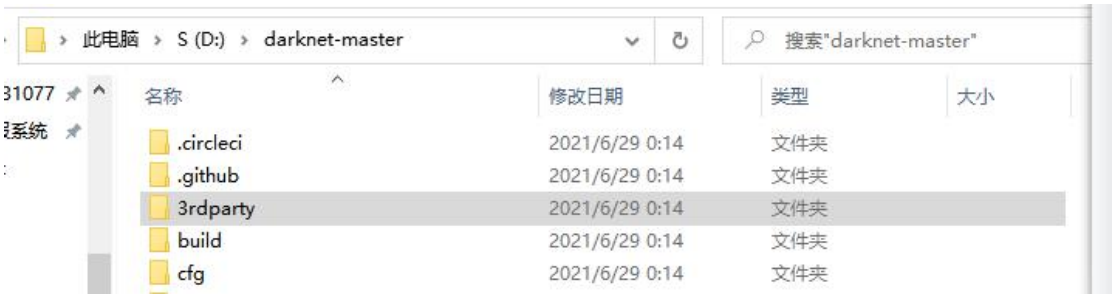
Tips （参考 https://www.it610.com/article/1281080563343572992.htm）

在 Darknet 生成过程中可能有非常多的问题，这些说一下我遇到的问题。

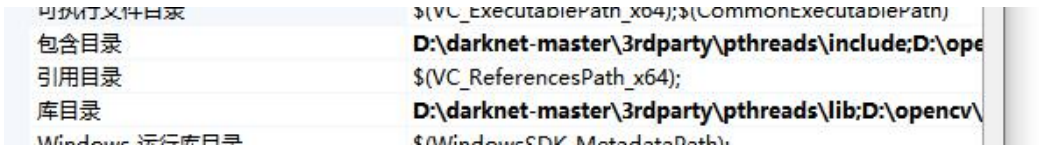


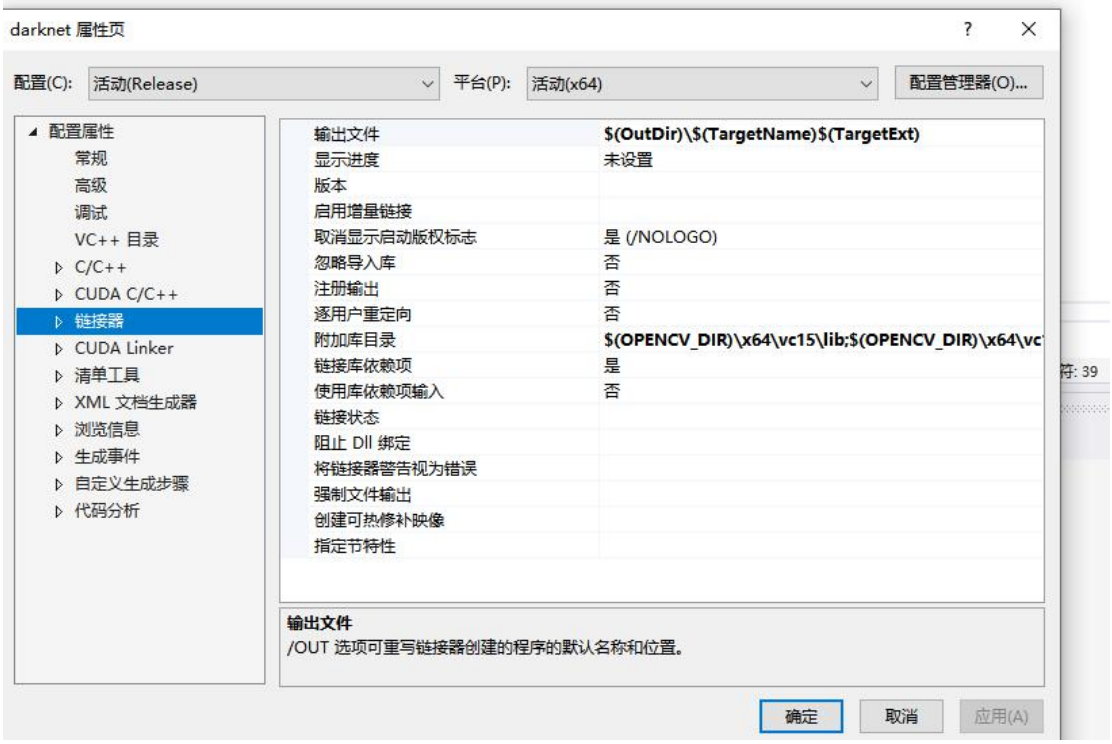
如上图

第一个是 imp__xxx 这种的问题，需要去下载 pthreadVC2 这个包， 下载完安装会在 darknet 里自动生成一个文件 3rdparty

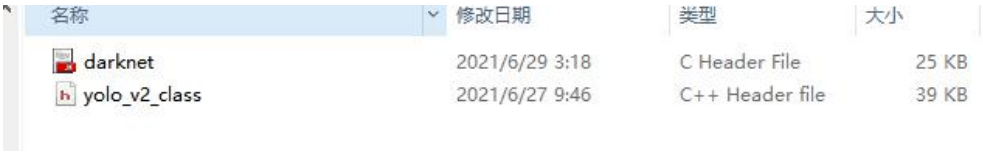


然后和配置 opencv 一样，把里面相关的 lib 等文件加载到 vs2019 中就行了





第二个就是 cublasxxx 和 curandxxx 相关的无法解析外部符号
我们需要在 draknet-master/include 文件夹下找到 darknet.h 文件，打开

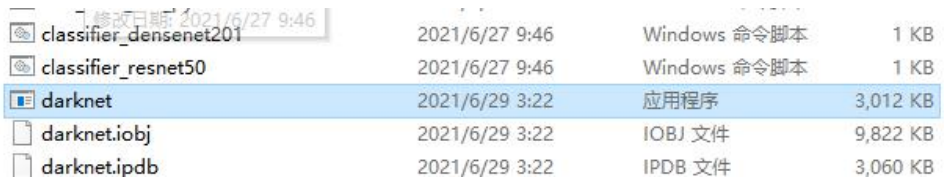


然后在里面加入

```
#pragma comment(lib, "cudart.lib")
#pragma comment(lib, "curand.lib")
#pragma comment(lib, "cublas.lib")
```

```
40
41  #include <cuda_runtime.h>
42  #include <curand.h>
43  #include <cublas_v2.h>
44  #pragma comment(lib, "cudart.lib")
45  #pragma comment(lib, "curand.lib")
46  #pragma comment(lib, "cublas.lib")
47
```

这三句语句，生成 darknet 就不会有问题了
做完以上的步骤之后,会成功生成 darknet.exe, (build\darknet\x64目录下）



二、测试与数据集准备

2.1 测试程序 darknet.exe

去 https://pjreddie.com/media/files/yolov3.weights 下载权重文件，把它放在和 darknet 一个文件夹，然后打开 cmd，用 cd 命令定位到此文件夹，然后运行命令

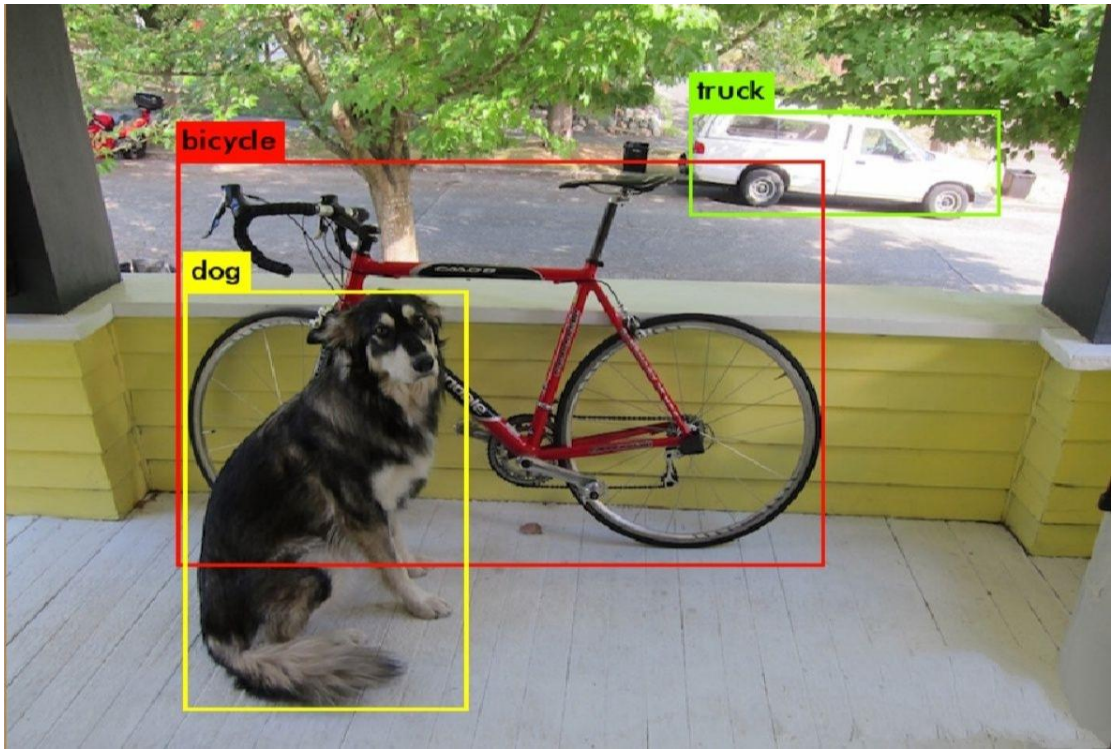
darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg

```
D:\darknet-master\build\darknet\x64>darknet detect cfg/yolov3.cfg yolov3.weights data/dog.jpg
 CUDA-version: 11000 (11010), cuDNN: 8.0.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 4.5.2
```

运行完后

```
 105 conv    255       1 x 1/ 1     52 x  52 x 256 ->   52 x  52 x 255 0.353 BF
 106 yolo
[yolo] params: iou loss: mse (2), iou_norm: 0.75, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.00
Total BFLOPS 65.879
avg_outputs = 532444
 Allocate additional workspace_size = 18.88 MB
Loading weights from yolov3.weights...
 seen 64, trained: 32013 K-images (500 Kilo-batches_64)
Done! Loaded 107 layers from weights-file
Cannot load image data/dog.jpg
 Detection layer: 82 - type = 28
 Detection layer: 94 - type = 28
 Detection layer: 106 - type = 28
data/dog.jpg: Predicted in 824.069000 milli-seconds.
```

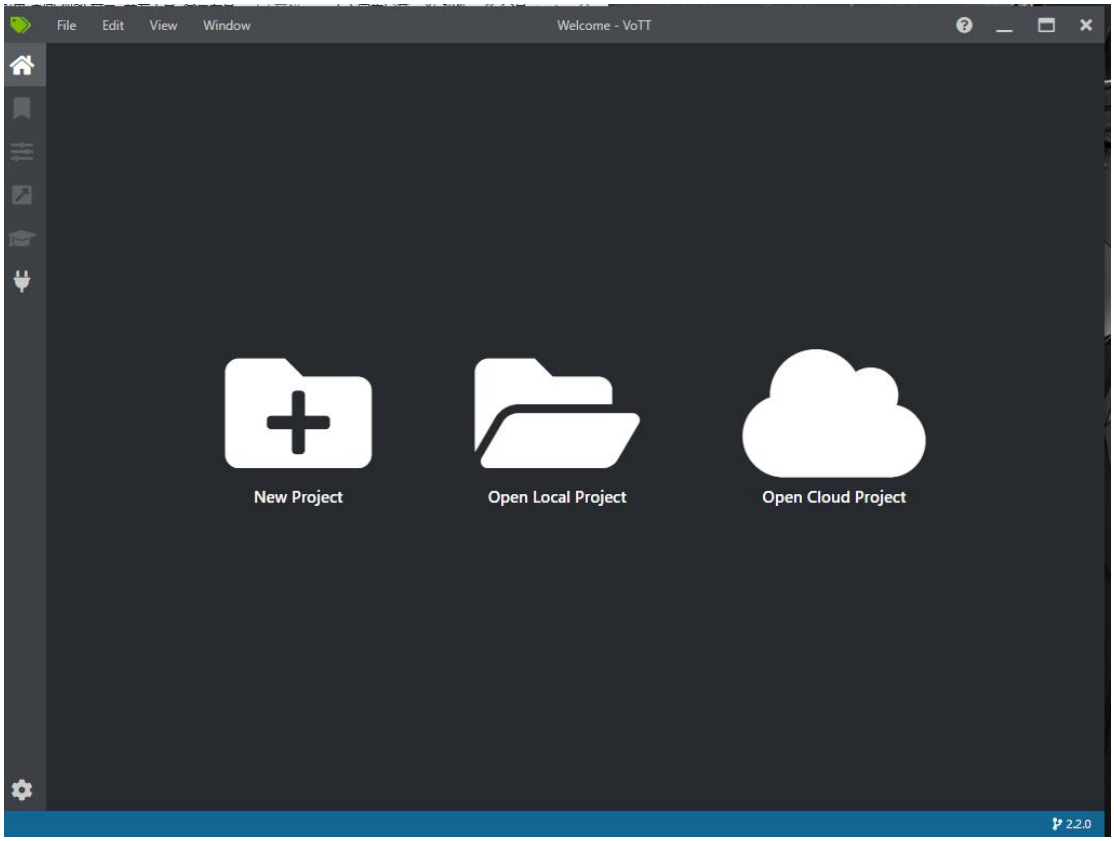
会出现



当然也可以用 eagle 和其它的图片进行测试

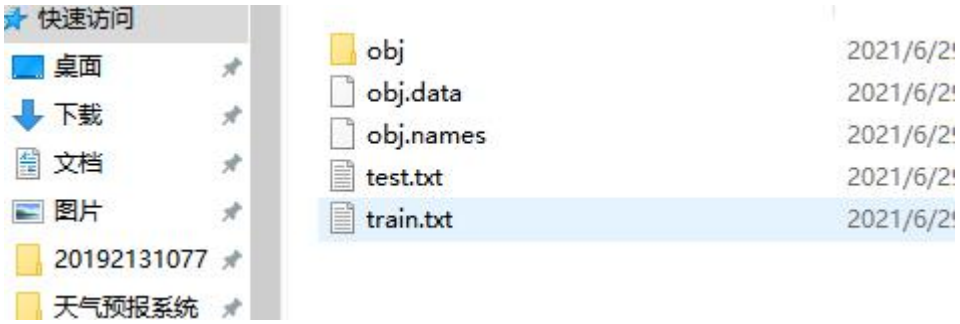
dog	2021/6/27 9:46	JPG 文件	160 KB
eagle	2021/6/27 9:46	JPG 文件	139 KB
giraffe	2021/6/27 9:46	JPG 文件	374 KB
goal	2021/6/27 9:46	文本文档	1 KB
horses	2021/6/27 9:46	JPG 文件	131 KB

2.2 Vott 打标签



输出数据（此文件夹命名为 datatest）

data 文件夹如下



三、训练

此部分主要是参考班上几位同学的过程，感谢
https://blog.csdn.net/weixin_43850253/article/details/118220395
http://course.gdou.com/file/source/YOLOv4_win10.pdf

将 darknet.exe 拷贝到 datat 文件夹下，，新建三个文件夹： backup, cfg, data, 还有 weights。下载一下权重文件放置到 weights 中, yolov4-tiny：
https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-tiny.conv.29
新建一个文件叫 yolov4-tiny.cfg, 复制下以下配置信息。

```
[net]
# Testing   #测试模式，测试时开启
#batch=1    #
#subdivisions=1 #
# Training  #训练模式，训练时开启，测试时注释
# 每批数量，根据配置设置，如果内存小，改小 batch 和 subdivisions， batch 和 subdivisions 越大，效果越好
# 要改
batch=64
# 要改
subdivisions=8
width=416
height=416
channels=3  # 输入图像 width height channels 长宽设置为 32 的倍数，因为下采样参数是 32，最小 320*320 最大 608*608
momentum=0.9    # 动量参数，影响梯度下降速度
decay=0.0005    # 权重衰减正则项，防止过拟合
angle=0 # 旋转
saturation = 1.5    # 饱和度扩增
exposure = 1.5  # 曝光度
hue=.1  # 色调

learning_rate=0.00261   # 学习率，权值更新速度
burn_in=1000    # 迭代次数小于 burn_in,学习率更新;大于 burn_in,采用 policy 更新
# 要改
max_batches = 12000 # 训练达到 max_batches 停止
policy=steps    # 学习率调整策略 policy:
constant, steps, exp, poly, step, sig, RANDOM
# 要改
steps=4800,5400 # 步长
scales=.1,.1    # 学习率变化比例
```

```
[convolutional]
batch_normalize=1
filters=32
size=3
stride=2
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=64
size=3
stride=2
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky
[route]
layers=-1
groups=2
group_id=1
[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky
[route]
layers = -1,-2
[convolutional]
batch_normalize=1
filters=64
```

```
size=1
stride=1
pad=1
activation=leaky
[route]
layers = -6,-1
[maxpool]
size=2
stride=2
[convolutional]
batch_normalize=1
filters=128
size=3
stride=1
pad=1
activation=leaky
[route]
layers=-1
groups=2
group_id=1
[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky
[route]
layers = -1,-2
[convolutional]
batch_normalize=1
filters=128
size=1
stride=1
pad=1
activation=leaky
```

```
[route]
layers = -6,-1
[maxpool]
size=2
stride=2
[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky
[route]
layers=-1
groups=2
group_id=1
[convolutional]
batch_normalize=1
filters=128
size=3
stride=1
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=128
size=3
stride=1
pad=1
activation=leaky
[route]
layers = -1,-2
[convolutional]
batch_normalize=1
filters=256
size=1
stride=1
pad=1
activation=leaky
[route]
layers = -6,-1
[maxpool]
size=2
stride=2
```

```
[convolutional]
batch_normalize=1
filters=512
size=3
stride=1
pad=1
activation=leaky
##################################
[convolutional]
batch_normalize=1
filters=256
size=1
stride=1
pad=1
activation=leaky
[convolutional]
batch_normalize=1
filters=512
size=3
stride=1
pad=1
activation=leaky
[convolutional]
size=1
stride=1
pad=1
# 要改    # 3*(classes+5)
filters=21
activation=linear
[yolo]
mask = 3,4,5
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319
# 要改    # 类别数，修改为实际需要数量
classes=2
num=6
jitter=.3
scale_x_y = 1.05
cls_normalizer=1.0
iou_normalizer=0.07
iou_loss=ciou
ignore_thresh = .7
truth_thresh = 1
random=0
resize=1.5
```

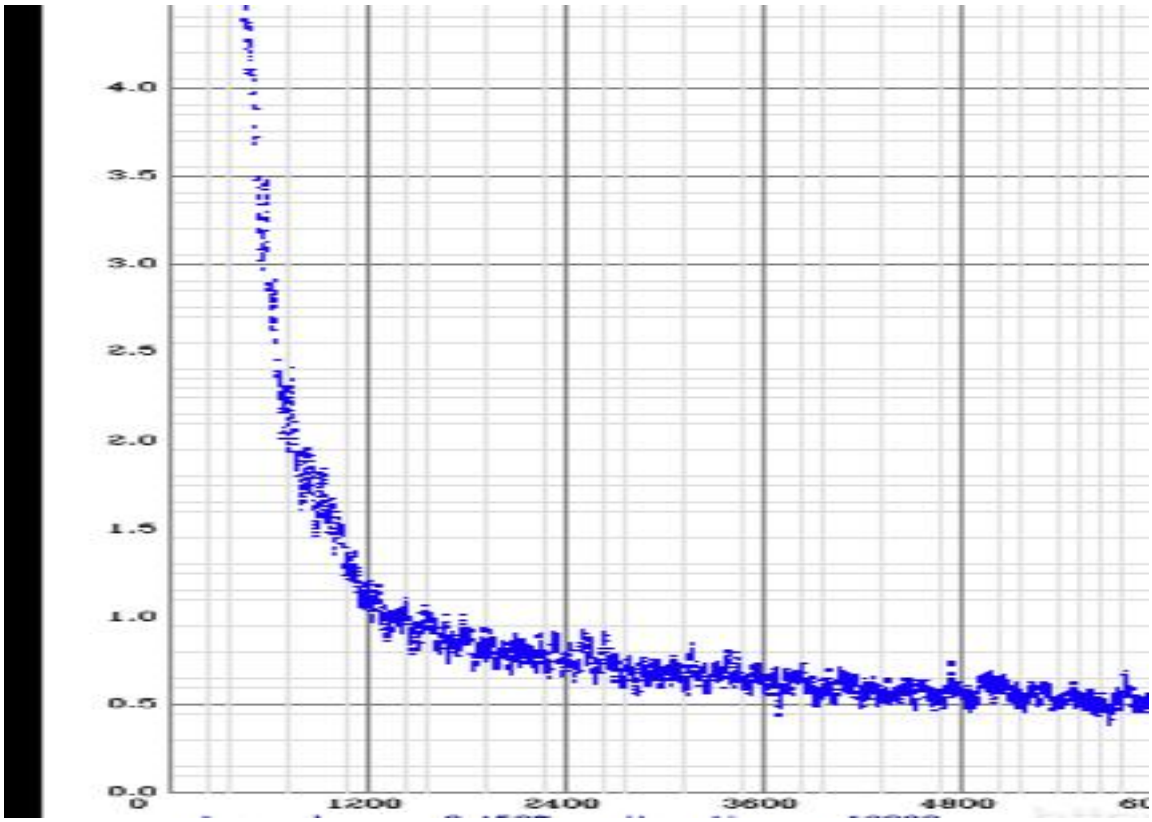
```
nms_kind=greedynms
beta_nms=0.6
[route]
layers = -4
[convolutional]
batch_normalize=1
filters=128
size=1
stride=1
pad=1
activation=leaky
[upsample]
stride=2
[route]
layers = -1, 23
[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky
[convolutional]
size=1
stride=1
pad=1
# 要改    # 3*(5+classes)
filters=21
activation=linear
[yolo]
mask = 1,2,3
anchors = 10,14,  23,27,  37,58,  81,82,  135,169,  344,319 #预测框的初始宽高，第一个是 w，第二个是 h，总数量是 num*2
# 要改    # 类别数，修改为实际需要数量
classes=2
num=6   # 每个 grid 预测的 BoundingBox 个数
jitter=.3   # # 利用数据抖动产生更多数据抑制过拟合.YOLOv2 中使用的是 crop，filp,以及 net 层的 angle,flip 是随机的,crop 就是 jitter 的参数,tiny-yolo-voc.cfg 中 jitter=.2，就是在 0~0.2 中进行 crop
scale_x_y = 1.05
cls_normalizer=1.0
iou_normalizer=0.07
iou_loss=ciou
```

```
ignore_thresh = .7  # # 决定是否需要计算 IOU 误差的参数，大于 thresh，IOU 误差不会夹在 cost function 中
truth_thresh = 1
random=0    #  如果为 1 每次迭代图片大小随机从 320 到 608，步长为 32，如果为 0，每次训练大小与输入大小一致
resize=1.5
nms_kind=greedynms
beta_nms=0.6
```

然后将 vott 标记好的数据拷贝到 data 中， 并将原来下载的 darknet 仓库中的 labels 文件夹拷贝进来,不拷贝的话可能会出现这种情况

```
OpenCV version: 4.5.2
Cannot load image data/labels/32_0.png
Cannot load image data/labels/33_0.png
Cannot load image data/labels/34_0.png
Cannot load image data/labels/35_0.png
Cannot load image data/labels/36_0.png
Cannot load image data/labels/37_0.png
Cannot load image data/labels/38_0.png
Cannot load image data/labels/39_0.png
Cannot load image data/labels/40_0.png
Cannot load image data/labels/41_0.png
Cannot load image data/labels/42_0.png
Cannot load image data/labels/43_0.png
Cannot load image data/labels/44_0.png
Cannot load image data/labels/45_0.png
Cannot load image data/labels/46_0.png
```

训练消耗了比较久，等训练才发现自己不知道参数设置错了，训练情况不是很好，次数也设置多了，但时间比较紧迫，因此就没有再试



四、计算 map

1.到这个网站下载这个项目的代码
2.把验证集中的测试图片放到 map\input \images-optional 目录下
3.把用 vott 输出的所有测试图片对应的.txt 文件放到 map\input\ground-truth 目录下
4.把 map\scripts\extra 目录下的 class_list.txt 里面的所有内容删除,第一行输入 gun，第二行输入 sword,然后保存文件.
5.把 yolov4-tiny-obj.cfg 文件里面的 subvision 和 batch 改为 1
6.把训练生成的文件夹下的 yolov4-tiny-obj_6000.weights 文件复制到 test 目录下
7.在 test 目录下运行 darknet.exe detector test data/obj.data yolov4-tiny-obj.cfg yolov4-tiny-obj_6000.weights -dont_show -ext_output <data/test.txt > result.txt -thresh 0.25
8.运行完成之后在 test 目录下得到一个 result.txt 文件,把这个文件复制粘贴到 map\input\detection_results 目录下
9.打开 map\scripts\extra 目录下的 convert_dr_yolo.py 文件,把里面的代码全部删掉，换为以下代码,然后保存文件,并且运行该文件.运行完成之后把上述第八个步骤得到的 result.txt 文件删除掉.

```
# mydarknet/separate_test.py
import os
import shutil
from tqdm import tqdm


# 删除一个文件夹下的内容
def remove_reg(top):
    for root, dirs, files in os.walk(top, topdown=False):
        for name in files:
            os.remove(os.path.join(root, name))
        for name in dirs:
            os.rmdir(os.path.join(root, name))
# 拷贝文件到文件夹
def copy_file_to_dir(file_lt, target_dir):
    for file_path in tqdm(file_lt):
        shutil.copy(file_path, target_dir)
    print('the files -> {} succeed, now the dir has {} files\n'.format(
        target_dir, len(os.listdir(target_dir))))
def separate_test(origin_dir, pic_dir, txt_dir, test_file):
    # 如果没有该文件夹，则创建该文件夹
    if not os.path.exists('test_dir'):
        os.mkdir('test_dir')
    if not os.path.exists(pic_dir):
        os.mkdir(pic_dir)
    if not os.path.exists('test_dir/test_txt'):
        os.mkdir('test_dir/test_txt')
```

```
    # 帮你清空文件夹下的图片/txt，这些上一次运行后的结果
    if len(os.listdir(pic_dir)) > 0:
        remove_reg(pic_dir)
    if len(os.listdir(txt_dir)) > 0:
        remove_reg(txt_dir)
    with open(test_file, 'r') as f:
        data = f.readlines()
    data_pic = [item.strip() for item in data]
    data_txt = [item.split('.')[0] + '.txt' for item in data]
    print('测试集的图片有 {} 张，它们为: '.format(len(data_pic)))
    print(data_pic)
    print('他们对应的 txt 有 {} 个文件，它们为: '.format(len(data_txt)))
    print(data_txt)
    print('\n')
    copy_file_to_dir(data_pic, pic_dir)
    copy_file_to_dir(data_txt, txt_dir)
if __name__ == '__main__':
    # 图片的路径
    origin_dir = 'work/data/obj'
    # 抽取出测试集的图片的路径
    pic_dir = 'test_dir/test_pic'
    # 抽取出测试集图片对应的 txt 图片的路径
    txt_dir = 'test_dir/test_txt'
    # 测试集 txt
    test_file = 'data/test.txt'
    separate_test(origin_dir, pic_dir, txt_dir, test_file)
```

10. 进入 map\scripts\extra\目录，打开其中的 convert_gt_yolo.py 文件,把其中 line62 和 line58 的图片路径改为 images-optional,然后运行该文件.

11. 运行 main.py 文件就能得到我们要的 map 图了。