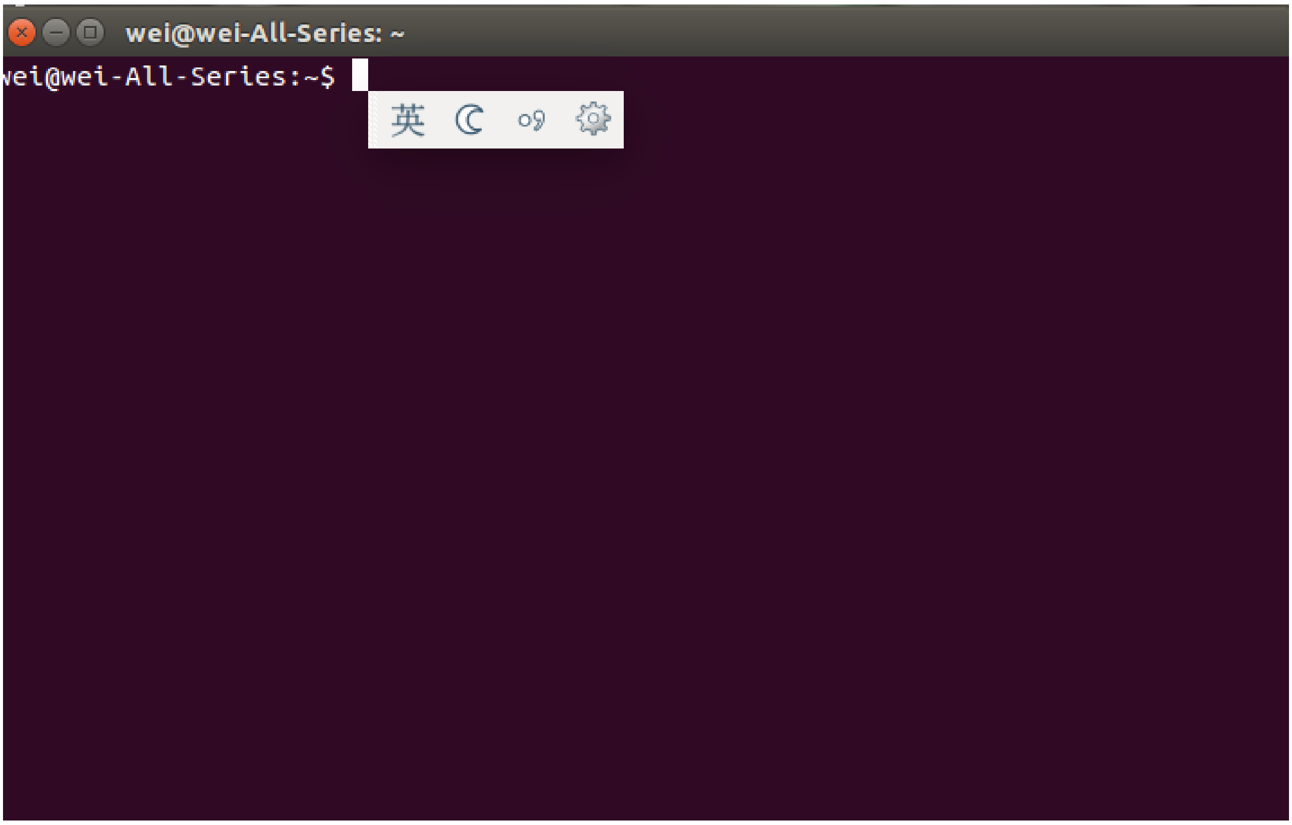
一，环境准备 ubuntu14.04　cuda8.0　cudnn v5.1　显卡 gtx1060

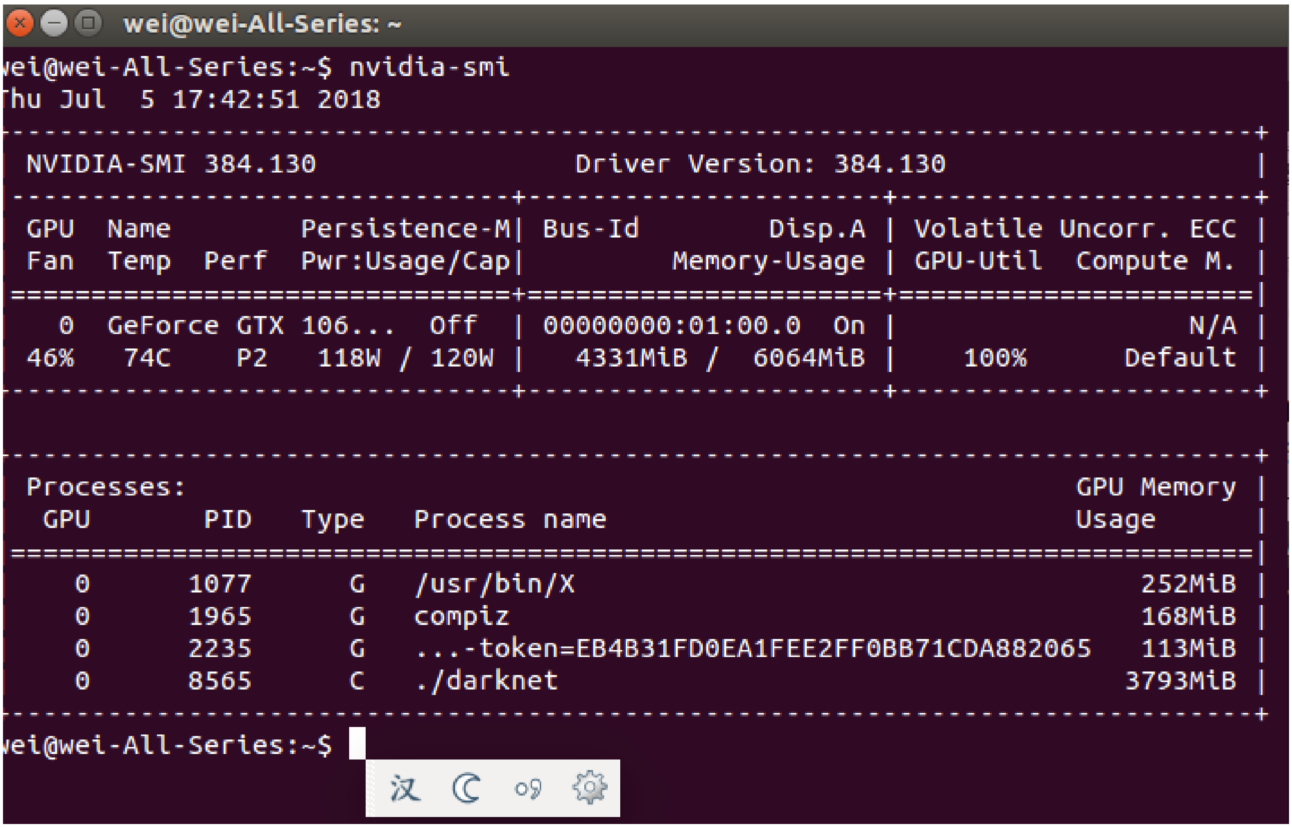
1 首先查看显卡驱动是否存在

同时按 ctrl+alt+t



```
wei@wei-All-Series:~$
```

粘贴 nvidia-smi 到终端

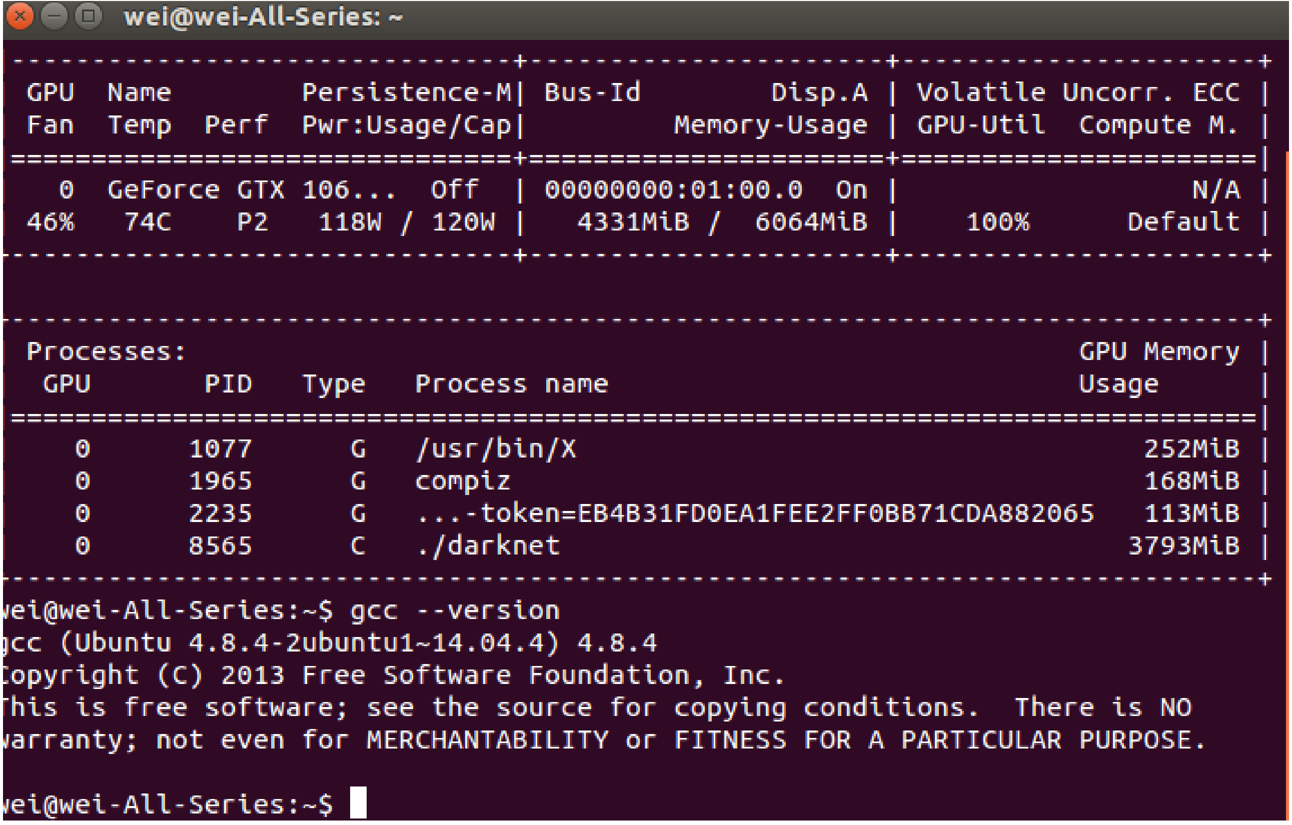


```
wei@wei-All-Series:~$ nvidia-smi
Thu Jul  5 17:42:51 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 384.130                Driver Version: 384.130                   |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 106...  Off  | 00000000:01:00.0  On |                  N/A |
| 46%   74C    P2   118W / 120W |   4331MiB /  6064MiB |    100%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      1077      G   /usr/bin/X                                   252MiB |
|    0      1965      G   compiz                                       168MiB |
|    0      2235      G   ...-token=EB4B31FD0EA1FEE2FF0BB71CDA882065   113MiB |
|    0      8565      C   ./darknet                                   3793MiB |
+-----------------------------------------------------------------------------+
wei@wei-All-Series:~$
```

如图显示则显卡驱动存在。

2 检查 cuda

输入 gcc --version

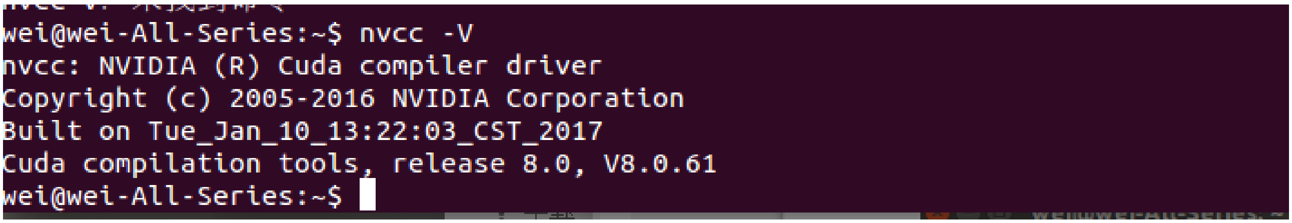


```
+-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 106...  Off  | 00000000:01:00.0  On |                  N/A |
| 46%   74C    P2   118W / 120W |   4331MiB /  6064MiB |    100%      Default |
+-------------------------------+----------------------+----------------------+


+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0      1077      G   /usr/bin/X                                   252MiB |
|    0      1965      G   compiz                                       168MiB |
|    0      2235      G   ...-token=EB4B31FD0EA1FEE2FF0BB71CDA882065   113MiB |
|    0      8565      C   ./darknet                                   3793MiB |
+-----------------------------------------------------------------------------+
wei@wei-All-Series:~$ gcc --version
gcc (Ubuntu 4.8.4-2ubuntu1~14.04.4) 4.8.4
Copyright (C) 2013 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

wei@wei-All-Series:~$ 
```

如图显示则成功

3 查看 cudnn

输入命令 nvcc -V



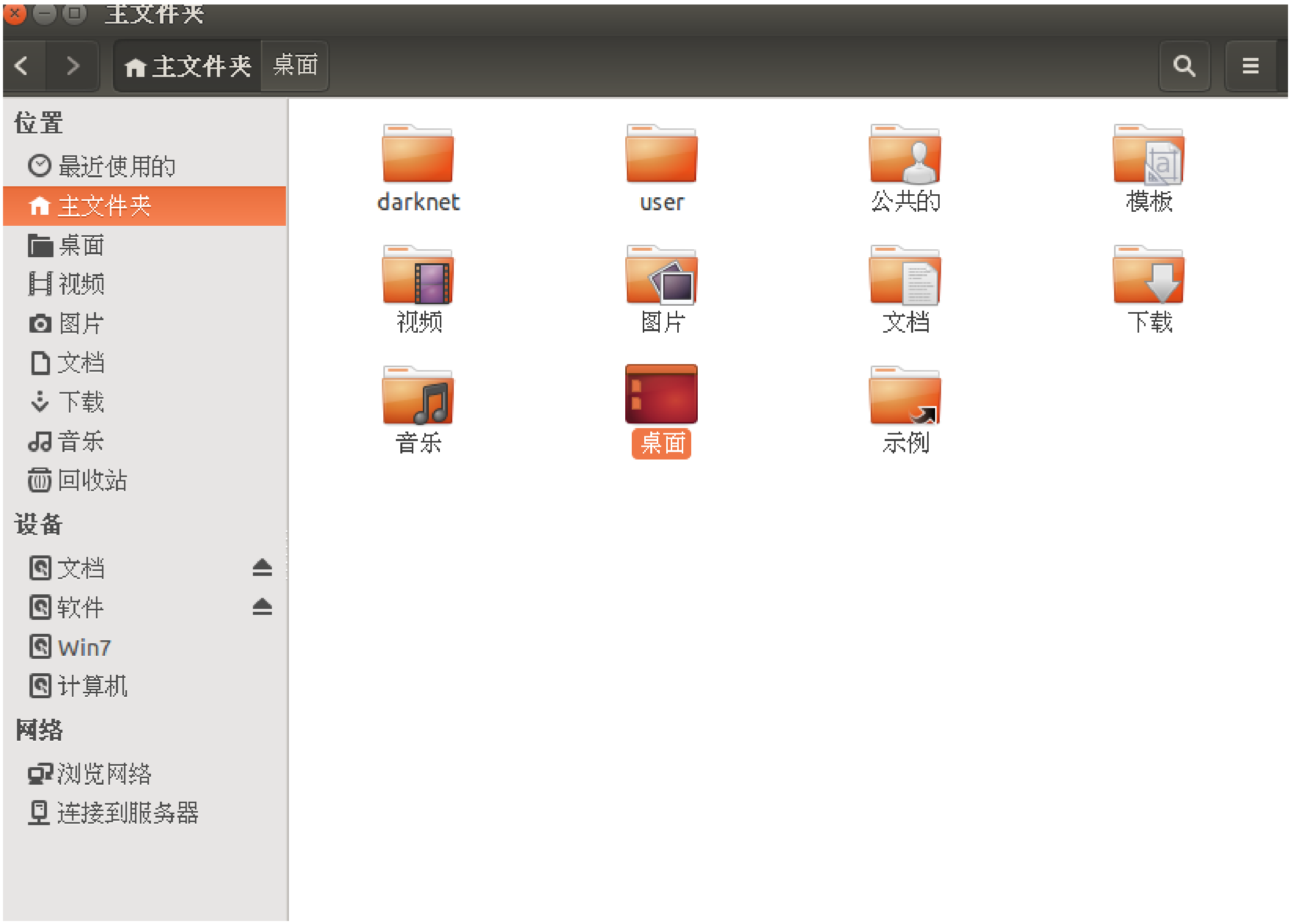
```
wei@wei-All-Series:~$ nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2016 NVIDIA Corporation
Built on Tue_Jan_10_13:22:03_CST_2017
Cuda compilation tools, release 8.0, V8.0.61
wei@wei-All-Series:~$ 
```

二，数据处理

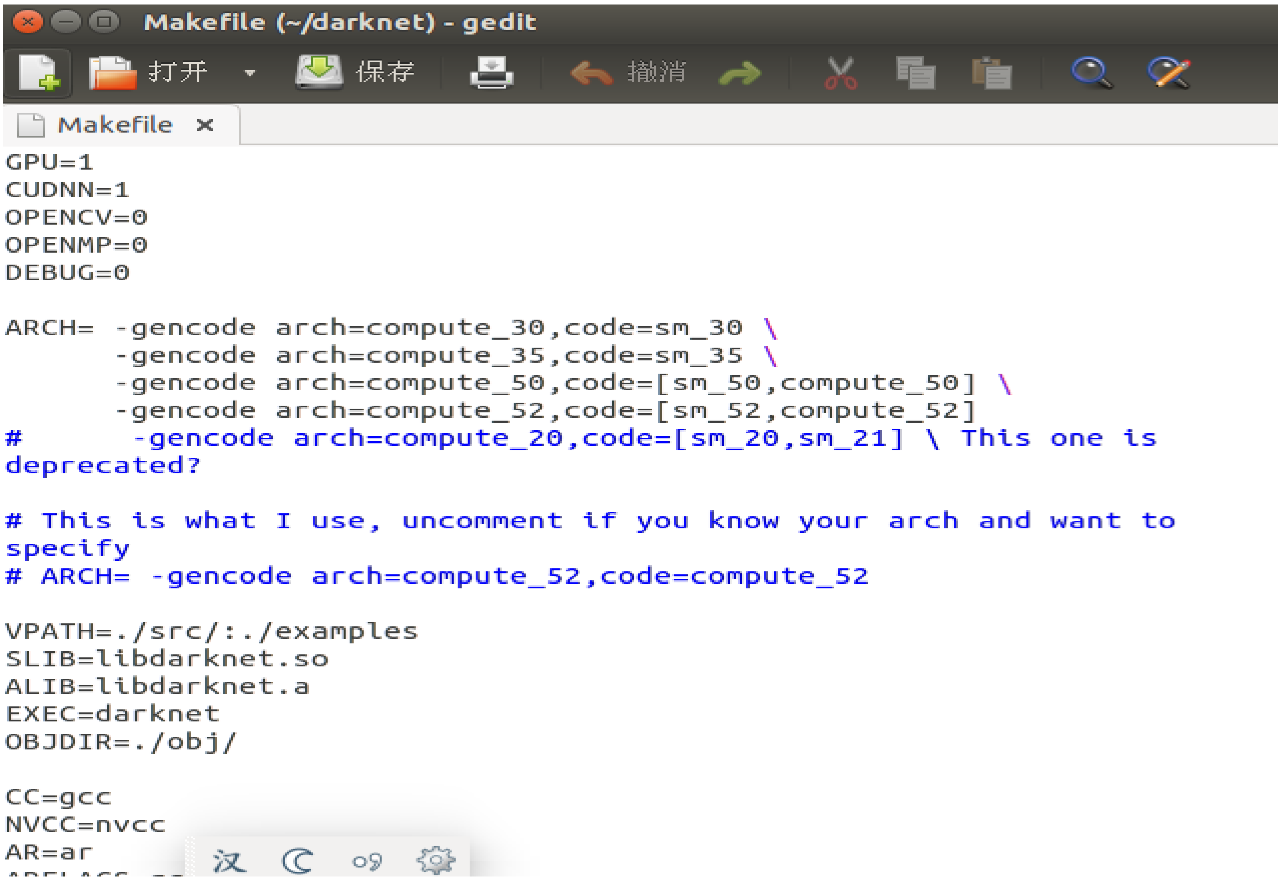
1 下载 darknet

输入 `git clone` https://github.com/pjreddie/darknet

下载 darknet 后，可以在看到 darknet 文件夹



2 双击 darknet 找到 makefile 文件双击打开



```
GPU=1
CUDNN=1
OPENCV=0
OPENMP=0
DEBUG=0

ARCH= -gencode arch=compute_30,code=sm_30 \
      -gencode arch=compute_35,code=sm_35 \
      -gencode arch=compute_50,code=[sm_50,compute_50] \
      -gencode arch=compute_52,code=[sm_52,compute_52]
#      -gencode arch=compute_20,code=[sm_20,sm_21] \ This one is
deprecated?

# This is what I use, uncomment if you know your arch and want to
specify
# ARCH= -gencode arch=compute_52,code=compute_52

VPATH=./src/:./examples
SLIB=libdarknet.so
ALIB=libdarknet.a
EXEC=darknet
OBJDIR=./obj/

CC=gcc
NVCC=nvcc
AR=ar
```

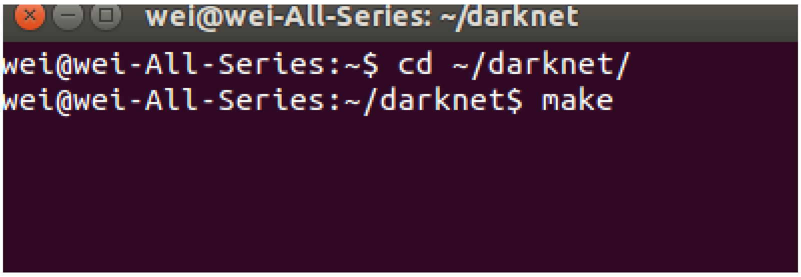
修改：

gpu=1

cudnn=1

3 切换到当前文件夹 cd ~/darknent/

在终端敲 make



```
wei@wei-All-Series:~$ cd ~/darknet/
wei@wei-All-Series:~/darknet$ make
```

4 双击 scripts 文件夹，双击 voc_label.py（这是 darknent 提供的 xml 转 txt 程序）

请按图修改。

修改

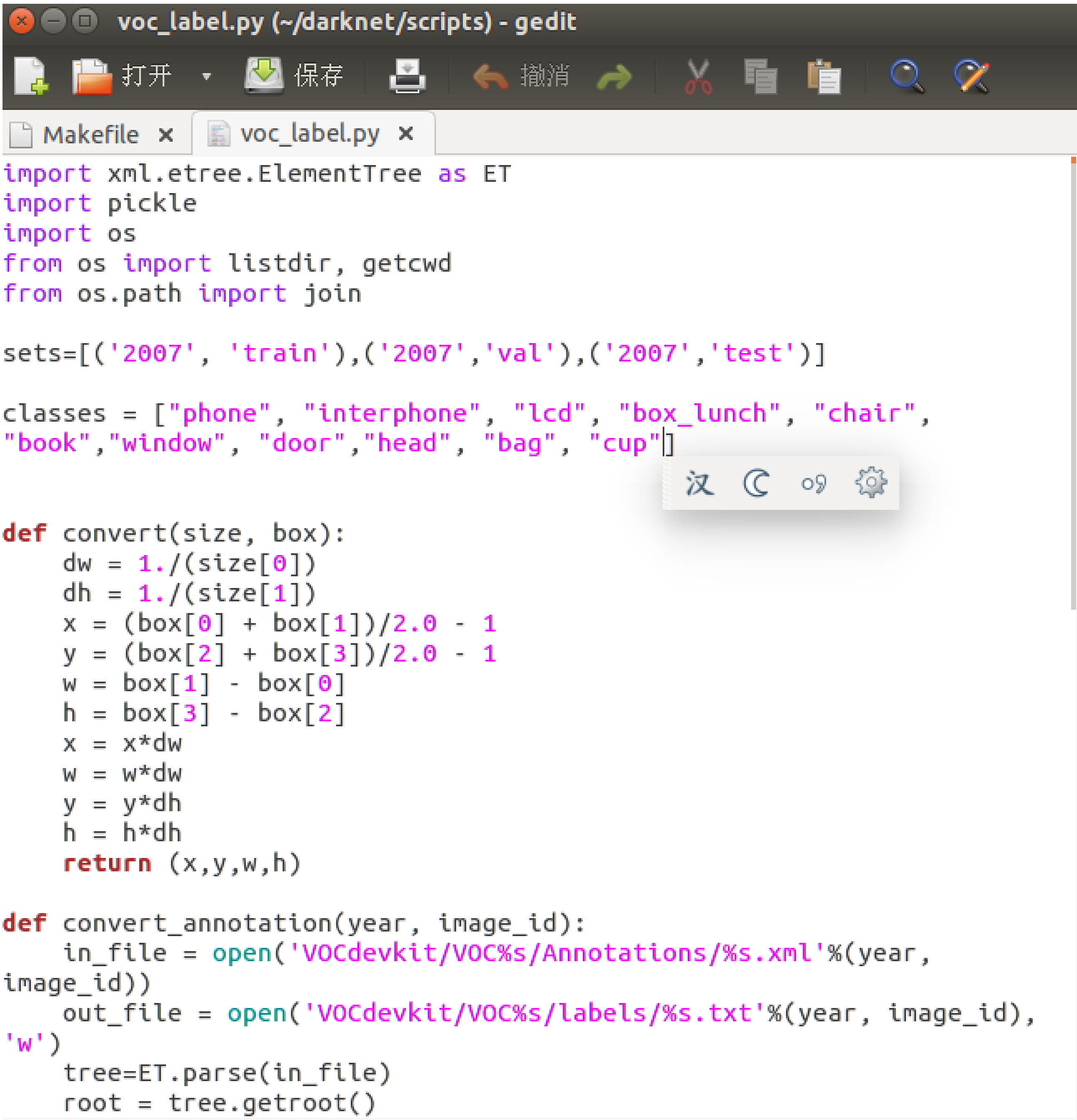
sets=[('2007', 'train'),('2007','val'),('2007','test')]

classes = ["phone", "interphone", "lcd", "box_lunch", "chair", "book","window", "door","head", "bag", "cup"]　这个要按照自己的分类来

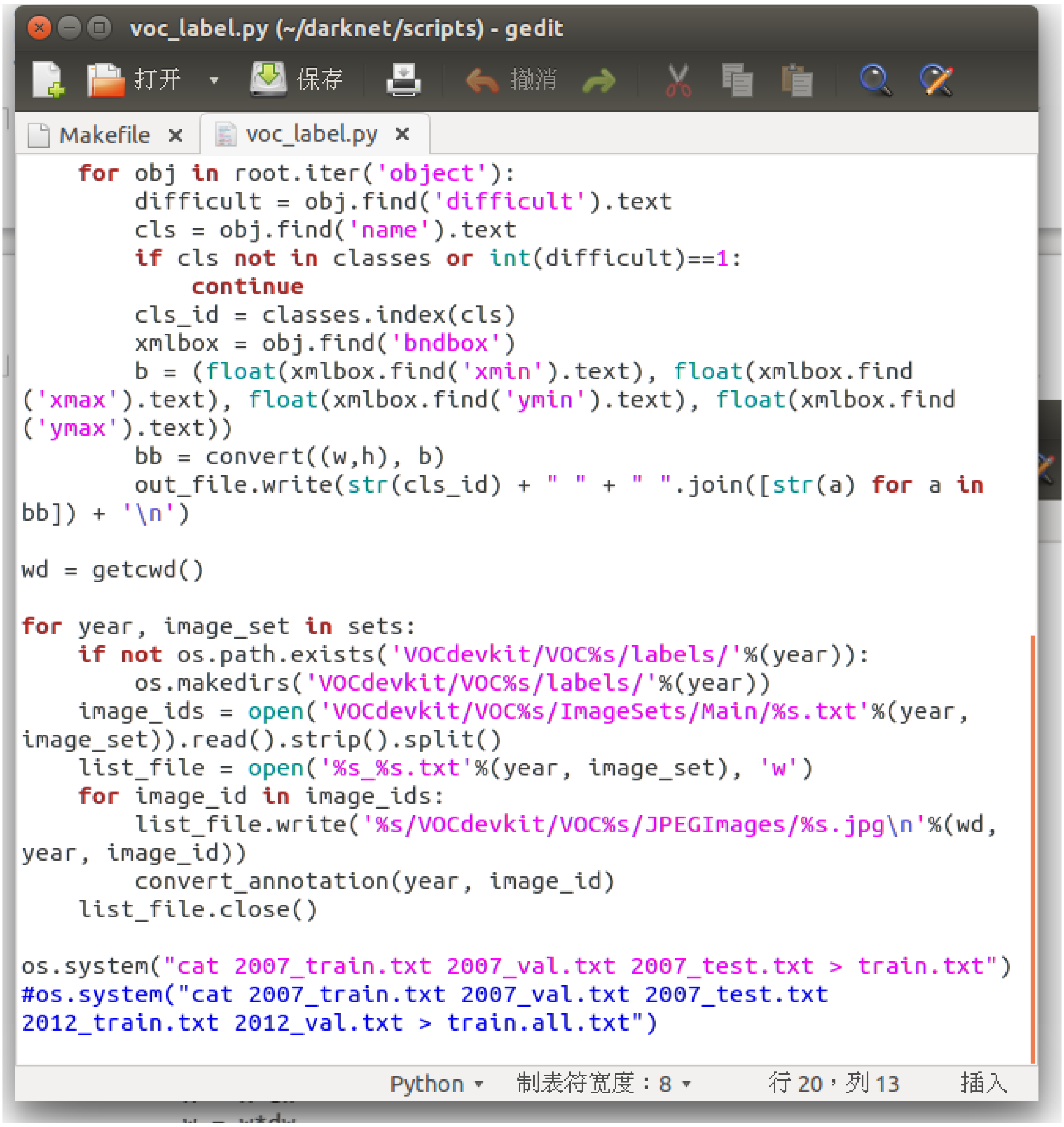
os.system("cat 2007_train.txt 2007_val.txt 2007_test.txt > train.txt")

#os.system("cat 2007_train.txt 2007_val.txt 2007_test.txt 2012_train.txt 2012_val.txt > train.all.txt")

注释最后一行。

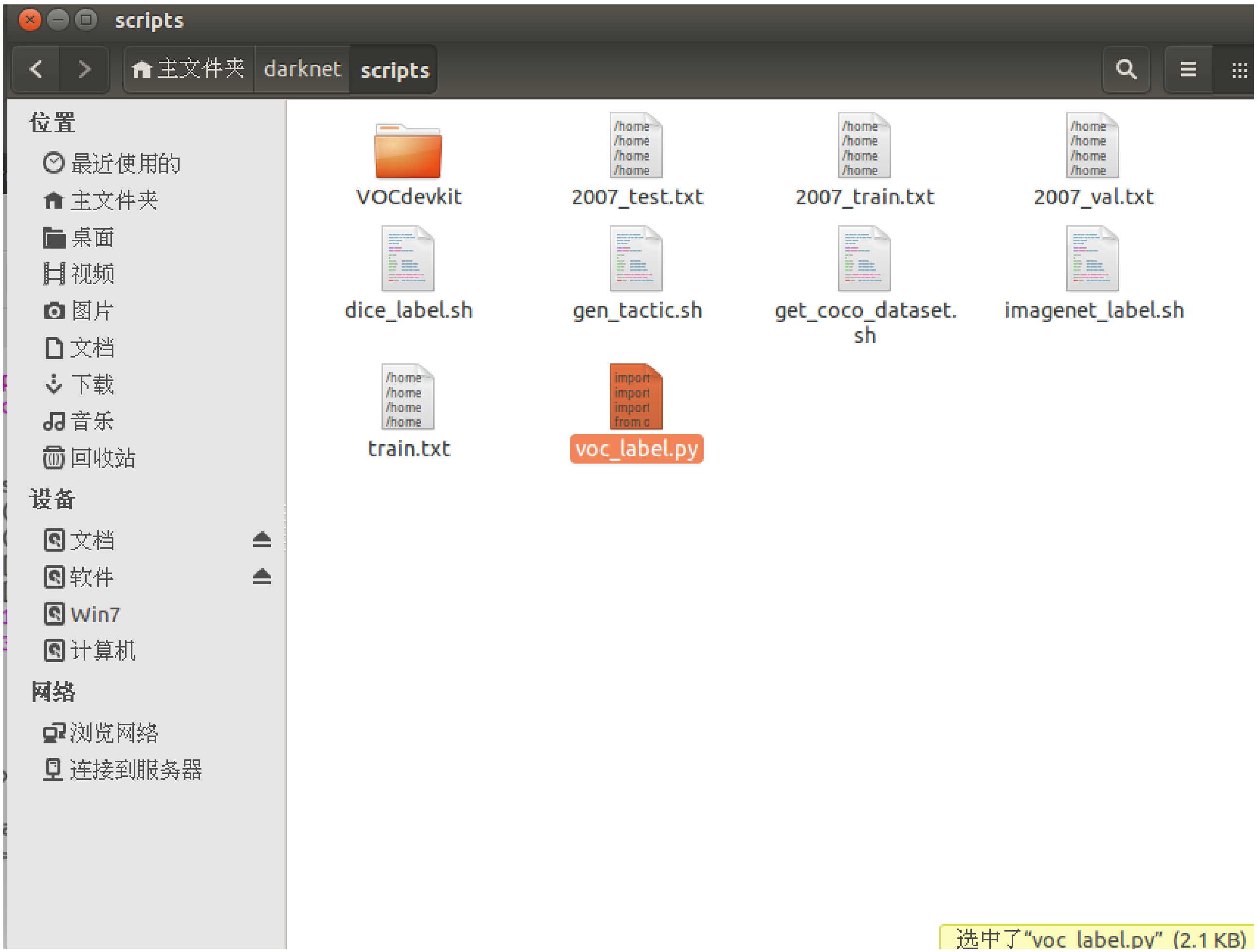


```
import xml.etree.ElementTree as ET
import pickle
import os
from os import listdir, getcwd
from os.path import join

sets=[('2007', 'train'),('2007','val'),('2007','test')]

classes = ["phone", "interphone", "lcd", "box_lunch", "chair",
"book","window", "door","head", "bag", "cup"]


def convert(size, box):
    dw = 1./(size[0])
    dh = 1./(size[1])
    x = (box[0] + box[1])/2.0 - 1
    y = (box[2] + box[3])/2.0 - 1
    w = box[1] - box[0]
    h = box[3] - box[2]
    x = x*dw
    w = w*dw
    y = y*dh
    h = h*dh
    return (x,y,w,h)

def convert_annotation(year, image_id):
    in_file = open('VOCdevkit/VOC%s/Annotations/%s.xml'%(year,
image_id))
    out_file = open('VOCdevkit/VOC%s/labels/%s.txt'%(year, image_id),
'w')
    tree=ET.parse(in_file)
    root = tree.getroot()
```



```python
    for obj in root.iter('object'):
        difficult = obj.find('difficult').text
        cls = obj.find('name').text
        if cls not in classes or int(difficult)==1:
            continue
        cls_id = classes.index(cls)
        xmlbox = obj.find('bndbox')
        b = (float(xmlbox.find('xmin').text), float(xmlbox.find('xmax').text), float(xmlbox.find('ymin').text), float(xmlbox.find('ymax').text))
        bb = convert((w,h), b)
        out_file.write(str(cls_id) + " " + " ".join([str(a) for a in bb]) + '\n')

wd = getcwd()

for year, image_set in sets:
    if not os.path.exists('VOCdevkit/VOC%s/labels/'%(year)):
        os.makedirs('VOCdevkit/VOC%s/labels/'%(year))
    image_ids = open('VOCdevkit/VOC%s/ImageSets/Main/%s.txt'%(year, image_set)).read().strip().split()
    list_file = open('%s_%s.txt'%(year, image_set), 'w')
    for image_id in image_ids:
        list_file.write('%s/VOCdevkit/VOC%s/JPEGImages/%s.jpg\n'%(wd, year, image_id))
        convert_annotation(year, image_id)
    list_file.close()

os.system("cat 2007_train.txt 2007_val.txt 2007_test.txt > train.txt")
#os.system("cat 2007_train.txt 2007_val.txt 2007_test.txt 2012_train.txt 2012_val.txt > train.all.txt")
```

5 双击进入 darknent/scripts 文件夹下新建一个名为 VOCdevkit 的文件夹

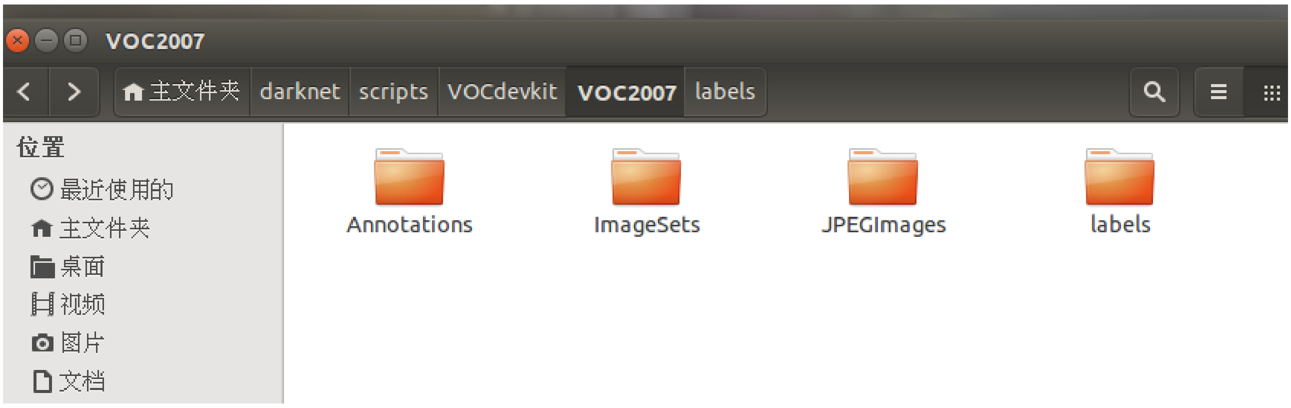


选中了"voc_label.py" (2.1 KB)

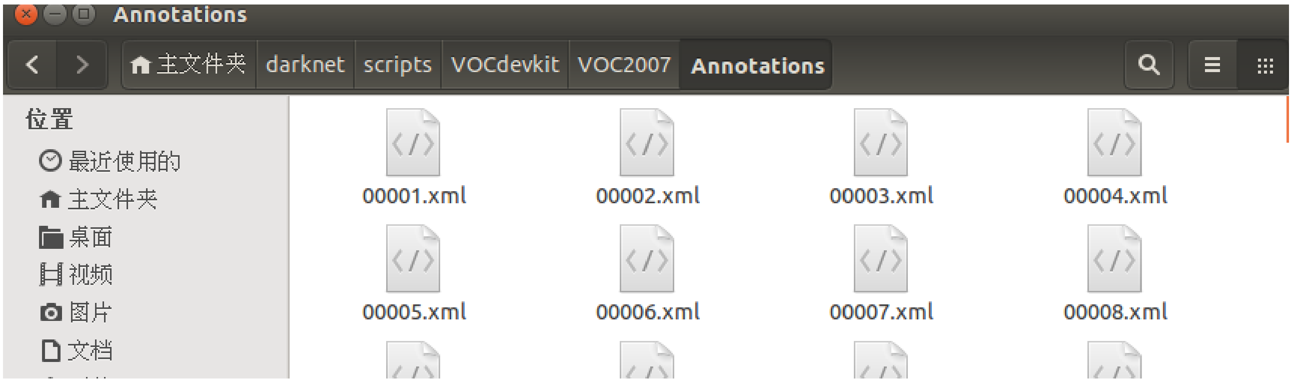
此时你比图片少 4 个*.txt 文件，不用担心，一会运行 voc_label.py 会帮你自动生成。

在 VOCdevkit 文件夹下新建 VOC2007 文件夹

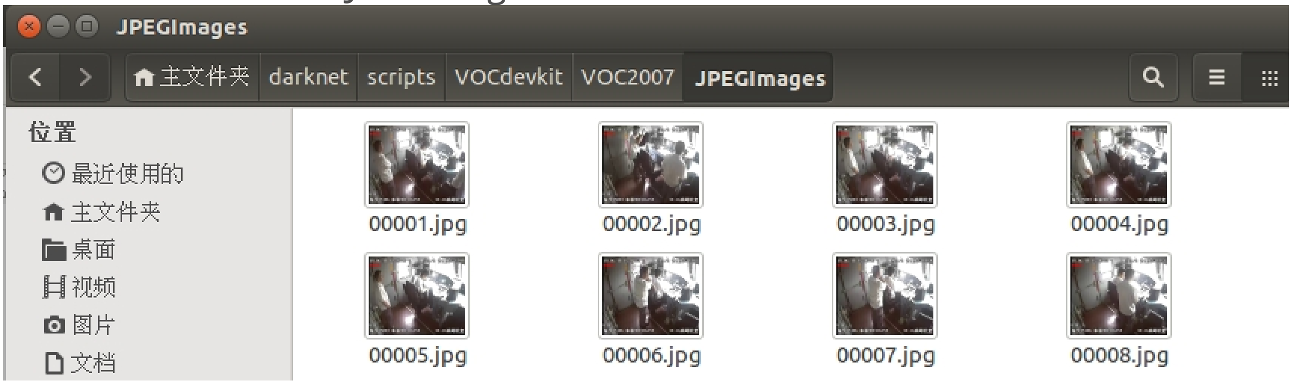
在 VOC2007 文件夹新建 Annotations；imageSets；JPEGimage；labels；四个子文件夹。



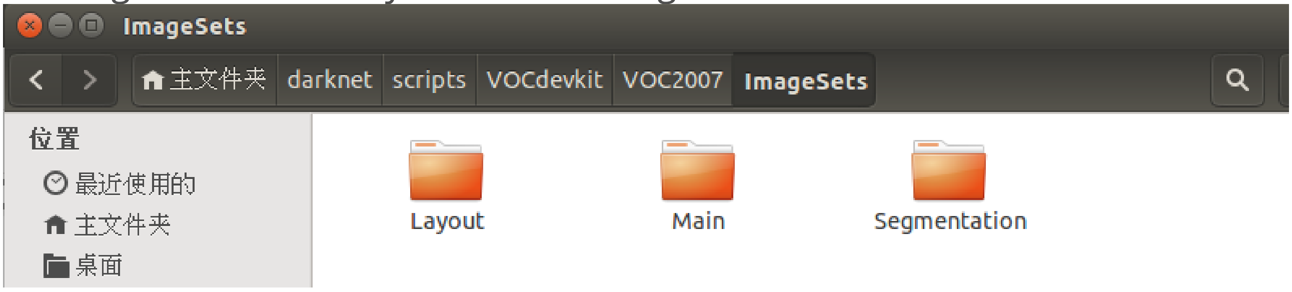
将我们之前标记生成的 xml 文件全部拷贝到 Annotations 下



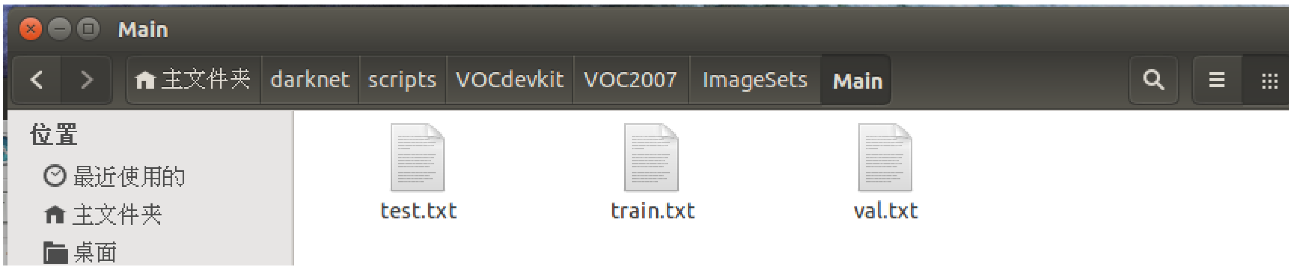
将所有的图片拷贝到 JPEGimages 下



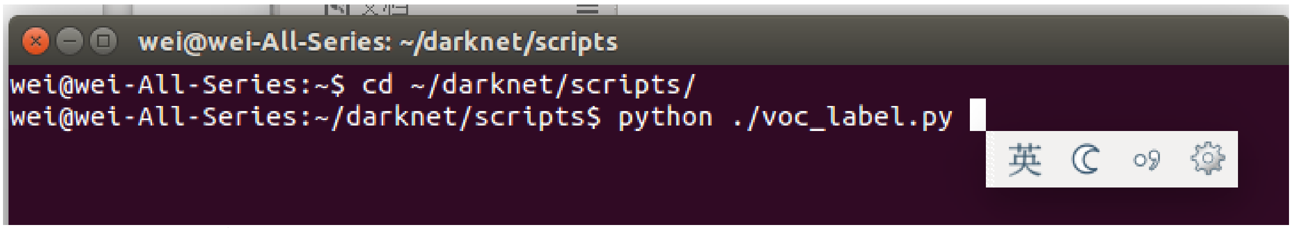
在 imageSets 下新建 Layout；Main；Segmentation；三个子文件夹



在 Main 文件夹下把上篇文章已经分好类的 3 个序号 TXT 拷进来



*****:~$ cd ~/darknet/scripts/
*****:~/darknet/scripts$ python ./voc_label.py(执行.py 文件则会生成之前你没有的 4 个 txt 数据文件，并且 vocdevkit/voc2007/labels 下也会生成 txt 文件)



```
wei@wei-All-Series: ~/darknet/scripts
wei@wei-All-Series:~$ cd ~/darknet/scripts/
wei@wei-All-Series:~/darknet/scripts$ python ./voc_label.py
```

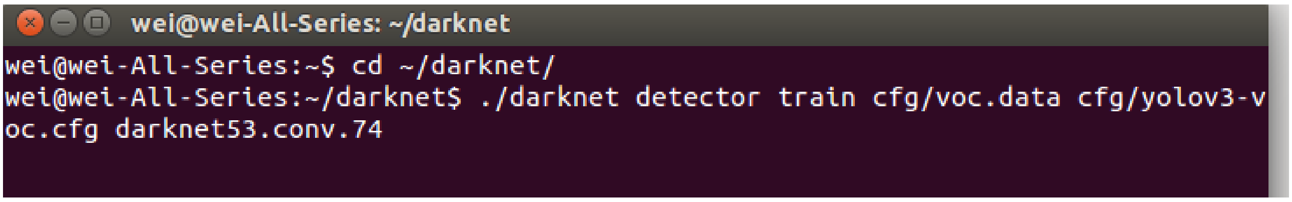
到此训练数据就放置成功了。

三.训练自己的数据

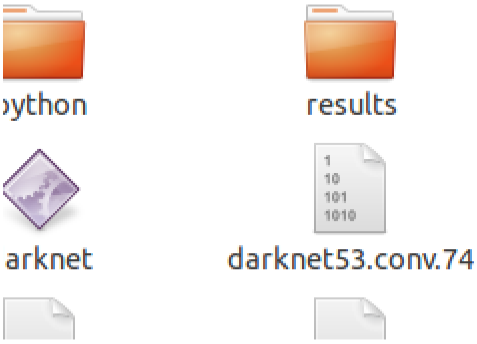
1 下载预训练模型

输入命令 cd ~/darknet/

输入命令`./darknet detector train cfg/voc.data cfg/yolov3-voc.cfg darknet53.conv.74`

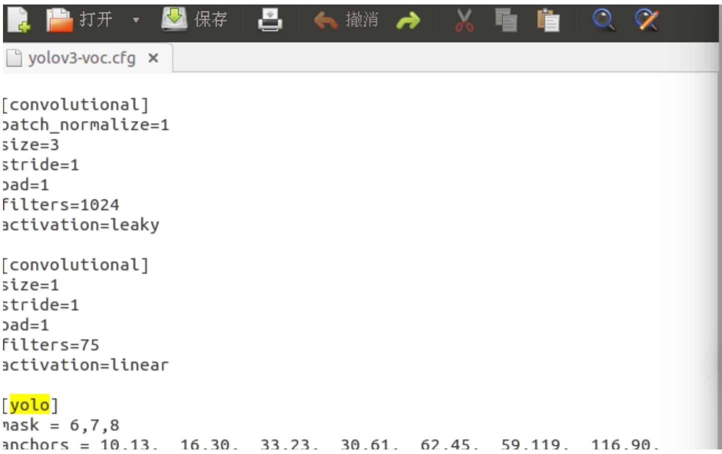


```
wei@wei-All-Series: ~/darknet
wei@wei-All-Series:~$ cd ~/darknet/
wei@wei-All-Series:~/darknet$ ./darknet detector train cfg/voc.data cfg/yolov3-v
oc.cfg darknet53.conv.74
```

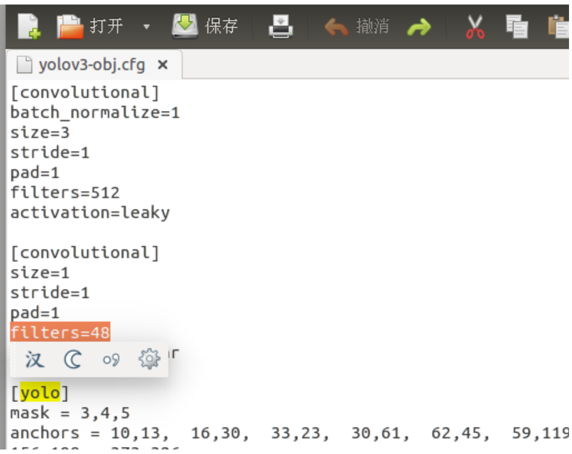


会在 darknet 文件夹下下载.74 文件。

2.在 cfg 文件夹下复制 yolov3-voc.cfg 重命名为 yolov3-obj.cfg
在 obj.cfg 中修改
修改 3 个【yolo】上面 filters=（分类数+5）*3
[yolo]修改 class 的数量

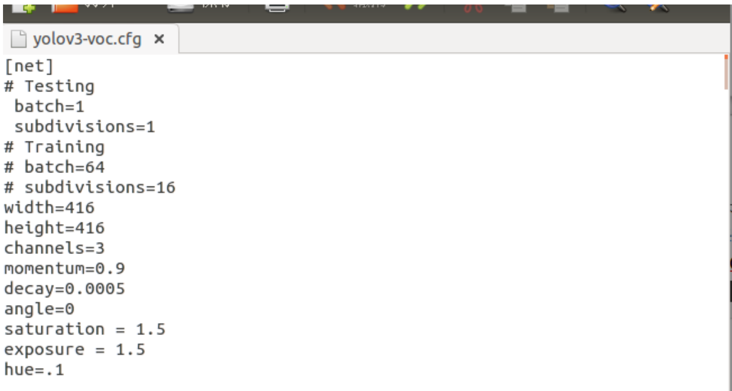


```
[convolutional]
batch_normalize=1
size=3
stride=1
pad=1
filters=1024
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=75
activation=linear

[yolo]
mask = 6,7,8
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,
```

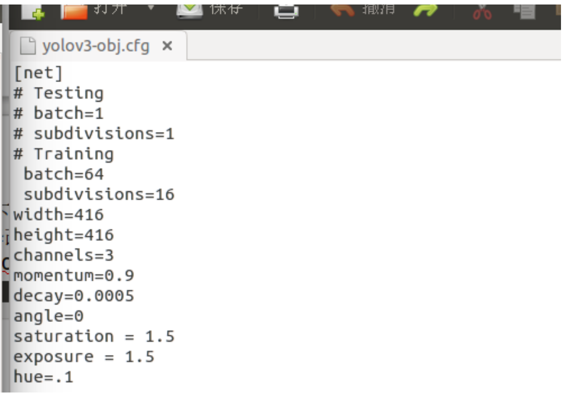


```
[convolutional]
batch_normalize=1
size=3
stride=1
pad=1
filters=512
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=48

[yolo]
mask = 3,4,5
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119
```

依然是 obj.cfg 仔细对照截图进行修改

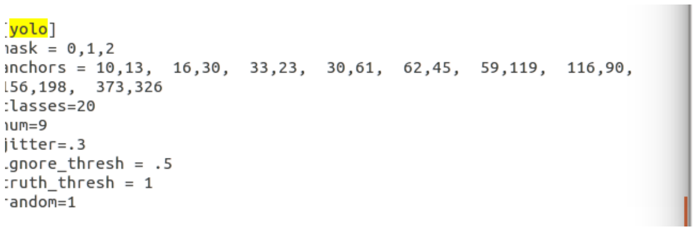


```
[net]
# Testing
 batch=1
 subdivisions=1
# Training
# batch=64
# subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1
```

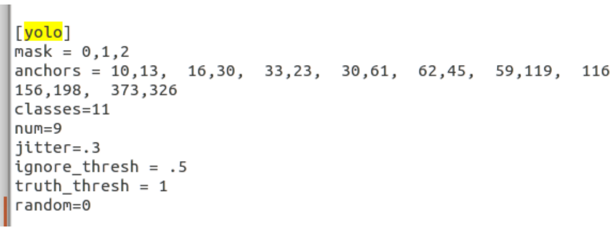


```
[net]
# Testing
# batch=1
# subdivisions=1
# Training
 batch=64
 subdivisions=16
width=416
height=416
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1
```

最后一行 random=0

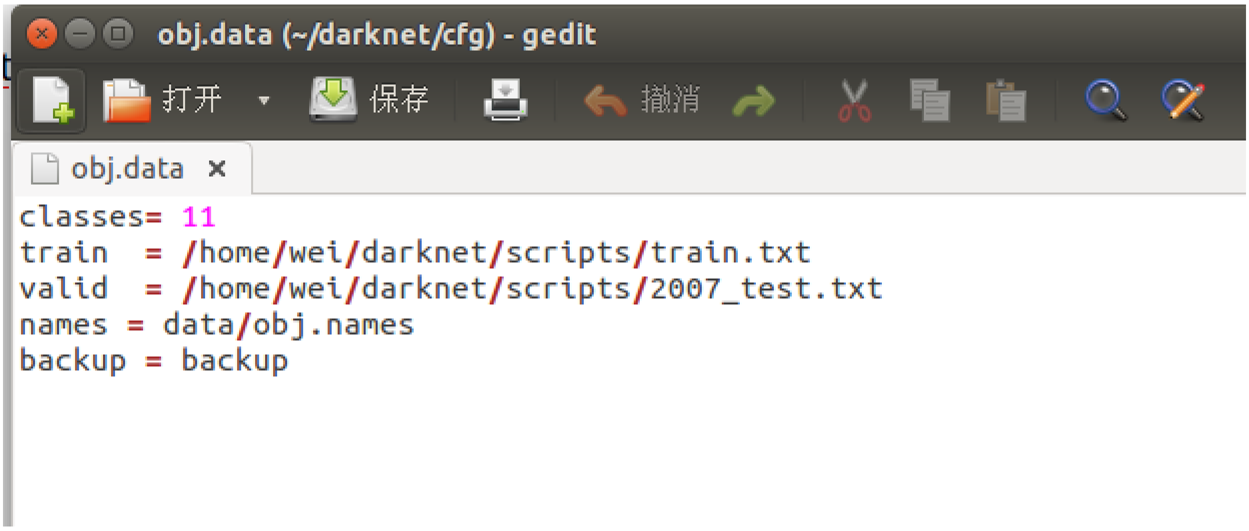


```
[yolo]
mask = 0,1,2
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,
156,198,  373,326
classes=20
num=9
jitter=.3
ignore_thresh = .5
truth_thresh = 1
random=1
```



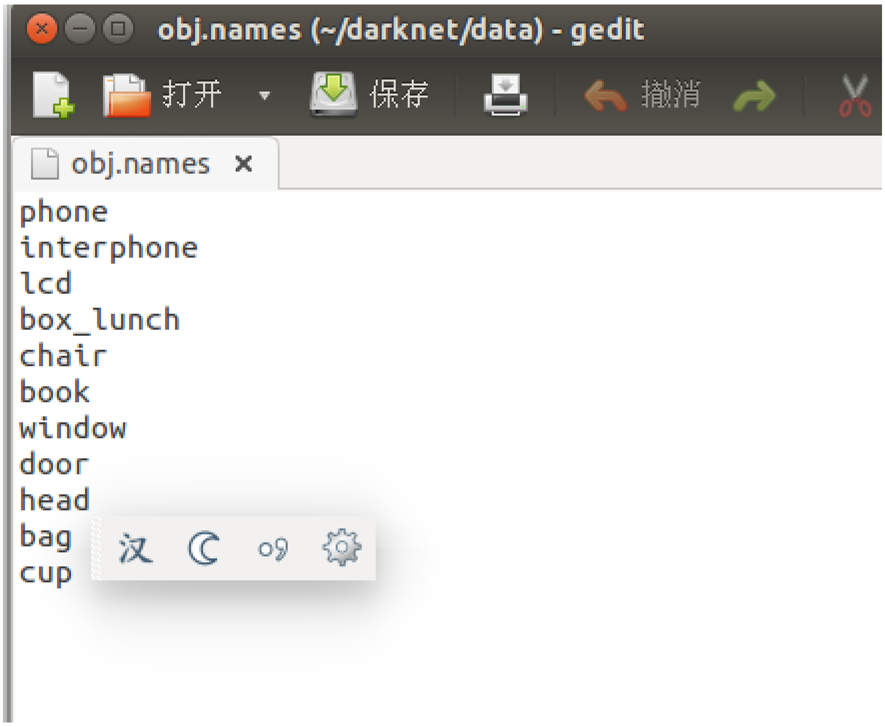
```
[yolo]
mask = 0,1,2
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116
156,198,  373,326
classes=11
num=9
jitter=.3
ignore_thresh = .5
truth_thresh = 1
random=0
```

3.在 darknet/cfg 文件夹下建立 obj.data，内容如图输入



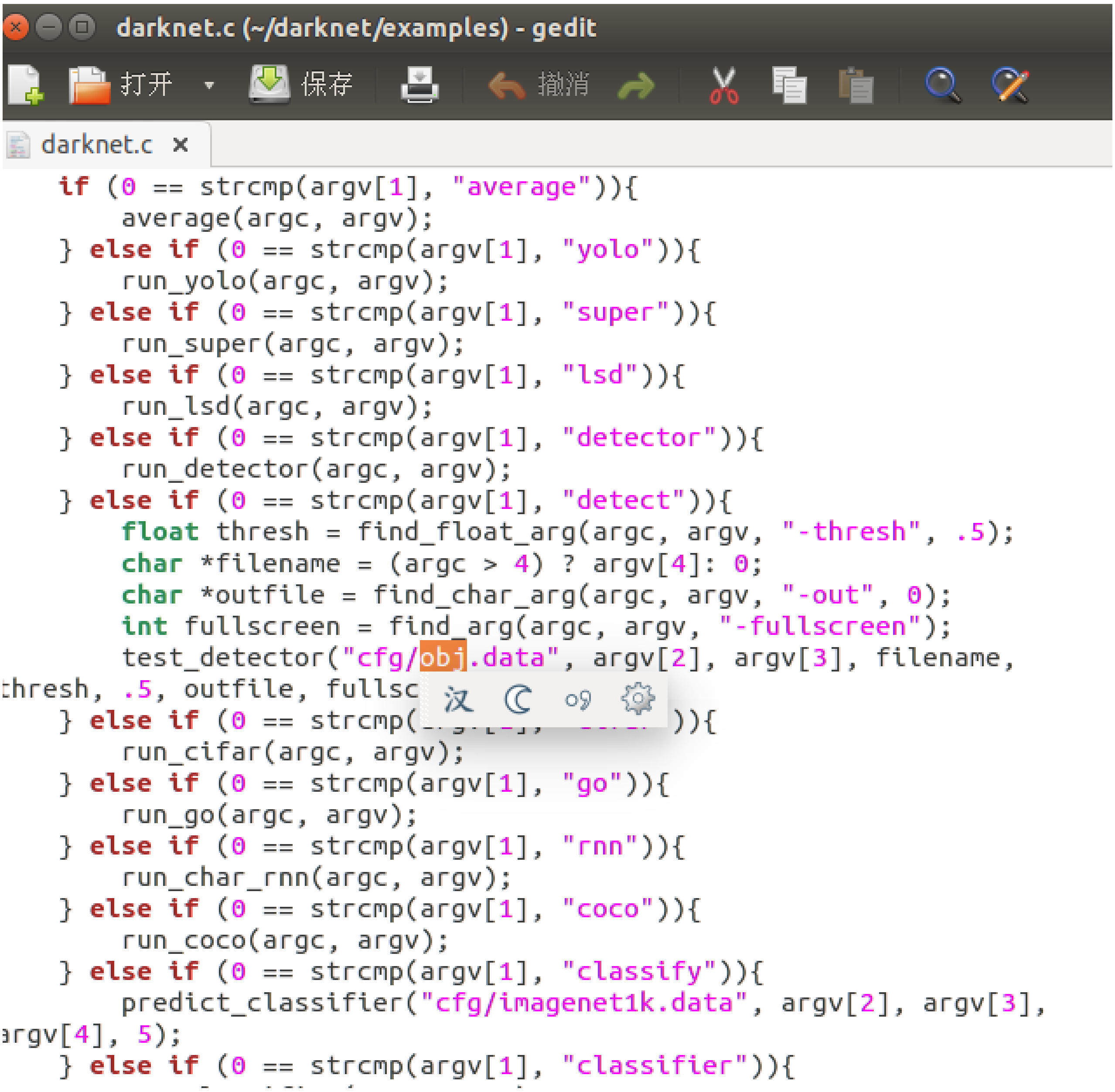
```
classes= 11
train  = /home/wei/darknet/scripts/train.txt
valid  = /home/wei/darknet/scripts/2007_test.txt
names = data/obj.names
backup = backup
```

4.在 darknet/data 文件夹下创建 obj.names，输入自己的分类。



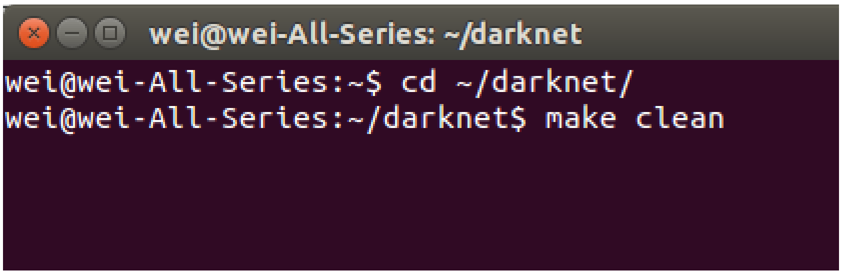
```
phone
interphone
lcd
box_lunch
chair
book
window
door
head
bag
cup
```

5.在 darknet/example 下找到 darknet.c 文件双击修改如图，将此处 coco 改为 obj



```
    if (0 == strcmp(argv[1], "average")){
        average(argc, argv);
    } else if (0 == strcmp(argv[1], "yolo")){
        run_yolo(argc, argv);
    } else if (0 == strcmp(argv[1], "super")){
        run_super(argc, argv);
    } else if (0 == strcmp(argv[1], "lsd")){
        run_lsd(argc, argv);
    } else if (0 == strcmp(argv[1], "detector")){
        run_detector(argc, argv);
    } else if (0 == strcmp(argv[1], "detect")){
        float thresh = find_float_arg(argc, argv, "-thresh", .5);
        char *filename = (argc > 4) ? argv[4]: 0;
        char *outfile = find_char_arg(argc, argv, "-out", 0);
        int fullscreen = find_arg(argc, argv, "-fullscreen");
        test_detector("cfg/obj.data", argv[2], argv[3], filename,
thresh, .5, outfile, fullsc
    } else if (0 == strcmp(                    )){
        run_cifar(argc, argv);
    } else if (0 == strcmp(argv[1], "go")){
        run_go(argc, argv);
    } else if (0 == strcmp(argv[1], "rnn")){
        run_char_rnn(argc, argv);
    } else if (0 == strcmp(argv[1], "coco")){
        run_coco(argc, argv);
    } else if (0 == strcmp(argv[1], "classify")){
        predict_classifier("cfg/imagenet1k.data", argv[2], argv[3],
argv[4], 5);
    } else if (0 == strcmp(argv[1], "classifier")){
```

6 打开终端
输入 cd ~/darknet/
输入 make clean

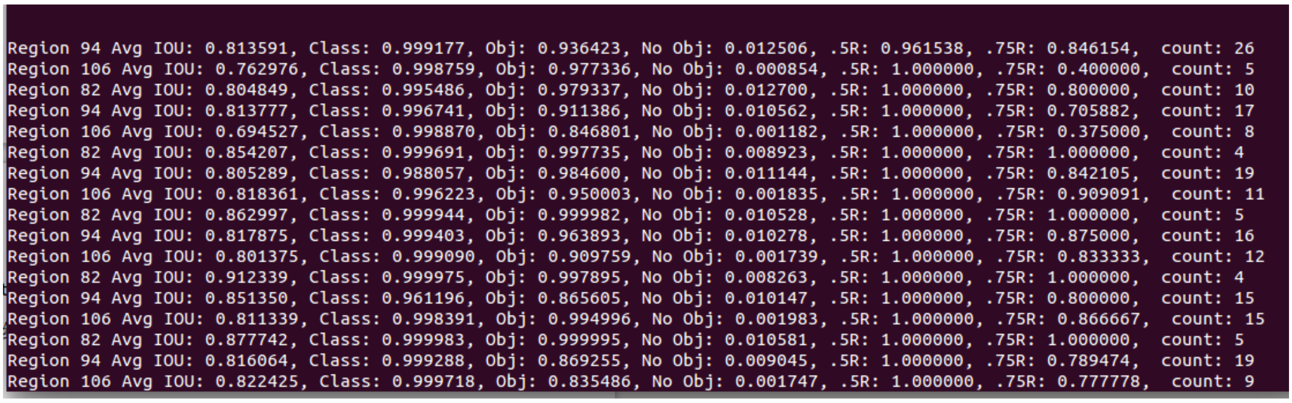


```
wei@wei-All-Series:~$ cd ~/darknet/
wei@wei-All-Series:~/darknet$ make clean
```

7 继续输入 make

8 继续输入`./darknet detector train cfg/obj.data cfg/yolov3-obj.cfg darknet53.conv.74 -gpus 0`

出现下图内容则成功训练。



```
Region 94 Avg IOU: 0.813591, Class: 0.999177, Obj: 0.936423, No Obj: 0.012506, .5R: 0.961538, .75R: 0.846154,  count: 26
Region 106 Avg IOU: 0.762976, Class: 0.998759, Obj: 0.977336, No Obj: 0.000854, .5R: 1.000000, .75R: 0.400000,  count: 5
Region 82 Avg IOU: 0.804849, Class: 0.995486, Obj: 0.979337, No Obj: 0.012700, .5R: 1.000000, .75R: 0.800000,  count: 10
Region 94 Avg IOU: 0.813777, Class: 0.996741, Obj: 0.911386, No Obj: 0.010562, .5R: 1.000000, .75R: 0.705882,  count: 17
Region 106 Avg IOU: 0.694527, Class: 0.998870, Obj: 0.846801, No Obj: 0.001182, .5R: 1.000000, .75R: 0.375000,  count: 8
Region 82 Avg IOU: 0.854207, Class: 0.999691, Obj: 0.997735, No Obj: 0.008923, .5R: 1.000000, .75R: 1.000000,  count: 4
Region 94 Avg IOU: 0.805289, Class: 0.988057, Obj: 0.984600, No Obj: 0.011144, .5R: 1.000000, .75R: 0.842105,  count: 19
Region 106 Avg IOU: 0.818361, Class: 0.996223, Obj: 0.950003, No Obj: 0.001835, .5R: 1.000000, .75R: 0.909091,  count: 11
Region 82 Avg IOU: 0.862997, Class: 0.999944, Obj: 0.999982, No Obj: 0.010528, .5R: 1.000000, .75R: 1.000000,  count: 5
Region 94 Avg IOU: 0.817875, Class: 0.999403, Obj: 0.963893, No Obj: 0.010278, .5R: 1.000000, .75R: 0.875000,  count: 16
Region 106 Avg IOU: 0.801375, Class: 0.999090, Obj: 0.909759, No Obj: 0.001739, .5R: 1.000000, .75R: 0.833333,  count: 12
Region 82 Avg IOU: 0.912339, Class: 0.999975, Obj: 0.997895, No Obj: 0.008263, .5R: 1.000000, .75R: 1.000000,  count: 4
Region 94 Avg IOU: 0.851350, Class: 0.961196, Obj: 0.865605, No Obj: 0.010147, .5R: 1.000000, .75R: 0.800000,  count: 15
Region 106 Avg IOU: 0.811339, Class: 0.998391, Obj: 0.994996, No Obj: 0.001983, .5R: 1.000000, .75R: 0.866667,  count: 15
Region 82 Avg IOU: 0.877742, Class: 0.999983, Obj: 0.999995, No Obj: 0.010581, .5R: 1.000000, .75R: 1.000000,  count: 5
Region 94 Avg IOU: 0.816064, Class: 0.999288, Obj: 0.869255, No Obj: 0.009045, .5R: 1.000000, .75R: 0.789474,  count: 19
Region 106 Avg IOU: 0.822425, Class: 0.999718, Obj: 0.835486, No Obj: 0.001747, .5R: 1.000000, .75R: 0.777778,  count: 9
```