Yolov5的检测和训练

检测

使用'detect.py'这个文件，需要修改其中的

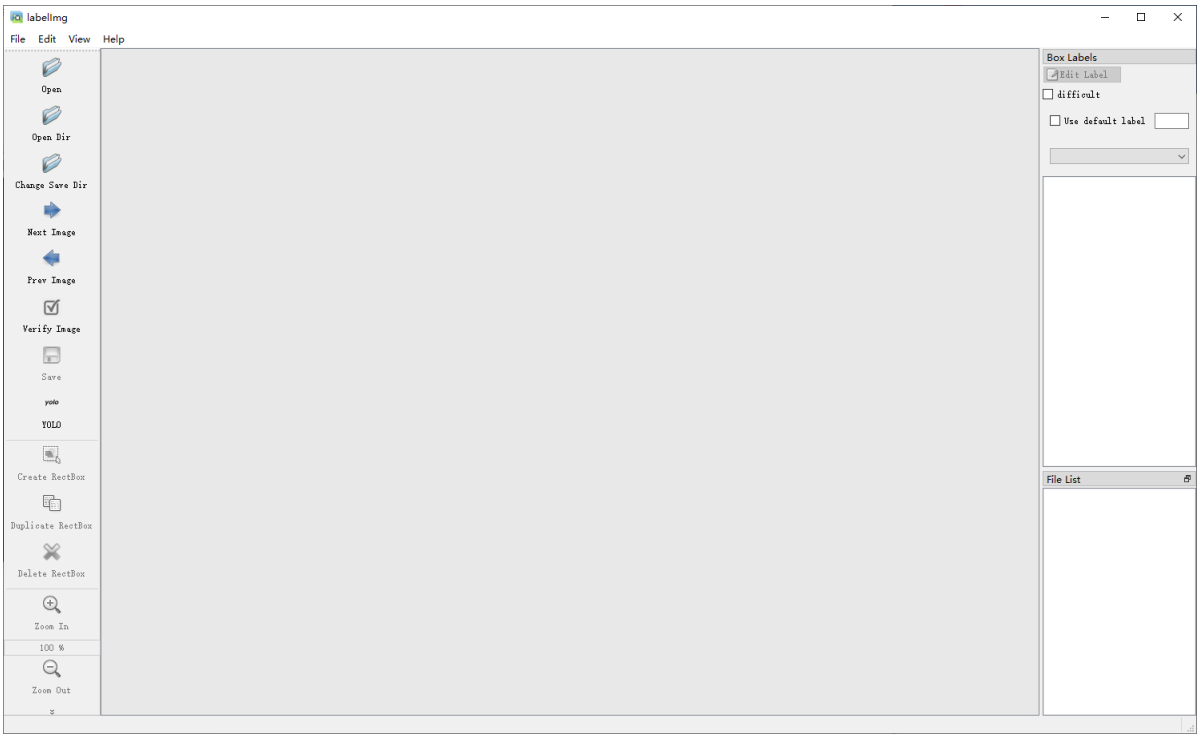
```
if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    # 存储权重的文件
    parser.add_argument('--weights', nargs='+', type=str, default='yolov5s.pt',
help='model.pt path(s)')
    # 检测图像的来源,使用 0 (摄像头) 或者 使用 1.mp4 (视频) 或者 xxx.jpg
    parser.add_argument('--source', type=str, default='data/images/bus.jpg',
help='source')  # file/folder, 0 for webcam
    # 图片的大小
    parser.add_argument('--img-size', type=int, default=640, help='inference size
(pixels)')
    parser.add_argument('--conf-thres', type=float, default=0.25, help='object
confidence threshold')
    parser.add_argument('--iou-thres', type=float, default=0.45, help='IOU
threshold for NMS')
    # 使用第几号cuda,gpu
    parser.add_argument('--device', default='0', help='cuda device, i.e. 0 or
0,1,2,3 or cpu')
```

其中：

```
weights  --权重文件的路径
source   --检测源，0代表本地的摄像头，可以是mp4视频，xxx.jpg图片等
img-size --图片的尺寸大小
device   --是否使用cuda，0表示cuda的0号设备，cpu表示只使用cup
```

训练

使用yolo格式的训练数据集，使用labelimg进行标注



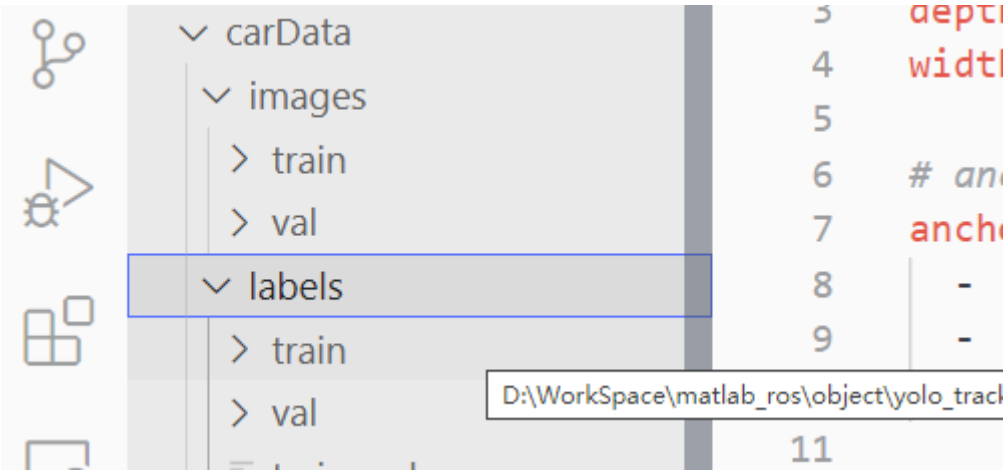

```
pip3 install labelimg python的版本不要太高
```

标注好以后会得到txt文件，如下所示

0表示类别，后面表示所在位置（已经归一化了）

文件存放格式

```
carData
|-images 存放图片
|------train
|------val
|-labels 存放txt文件
|------train
|------val
```



复制一份model文件夹下的yolov5s.yaml，重命名

然后修改其中的nc（number of classes）

```yaml
# parameters
nc: 2  # number of classes
depth_multiple: 0.33  # model depth multiple
width_multiple: 0.50  # layer channel multiple

# anchors
anchors:
  - [10,13, 16,30, 33,23]  # P3/8
  - [30,61, 62,45, 59,119]  # P4/16
  - [116,90, 156,198, 373,326]  # P5/32

# YOLOv5 backbone
backbone:
  # [from, number, module, args]
  [[-1, 1, Focus, [64, 3]],  # 0-P1/2
   [-1, 1, Conv, [128, 3, 2]],  # 1-P2/4
```

复制一份data文件夹下的coco128.yaml文件，重命名

```yaml
# download command/URL (optional)
download: https://github.com/ultralytics/yolov5/releases/download/v1.0/coco128.zip

# train and val data as 1) directory: path/images/, 2) file: path/images.txt, or 3) list: [path1/images/,
train: ../coco128/images/train2017/  # 128 images
val: ../coco128/images/train2017/  # 128 images

# number of classes
nc: 80

# class names
names: [ 'person', 'bicycle', 'car', 'motorcycle', 'airplane', 'bus', 'train', 'truck', 'boat', 'traffic l
         'fire hydrant', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog', 'horse', 'sheep', 'c
         'elephant', 'bear', 'zebra', 'giraffe', 'backpack', 'umbrella', 'handbag', 'tie', 'suitcase', 'fr
         'skis', 'snowboard', 'sports ball', 'kite', 'baseball bat', 'baseball glove', 'skateboard', 'surf
         'tennis racket', 'bottle', 'wine glass', 'cup', 'fork', 'knife', 'spoon', 'bowl', 'banana', 'appl
         'sandwich', 'orange', 'broccoli', 'carrot', 'hot dog', 'pizza', 'donut', 'cake', 'chair', 'couch'
         'potted plant', 'bed', 'dining table', 'toilet', 'tv', 'laptop', 'mouse', 'remote', 'keyboard', '
         'microwave', 'oven', 'toaster', 'sink', 'refrigerator', 'book', 'clock', 'vase', 'scissors', 'ted
         'hair drier', 'toothbrush' ]
```

修改成如下

data > ! car.yaml

```yaml
train: ./carData/images/train
val: ./carData/images/val
test: #

nc: 1
names:
  0: car
```

在train.py文件中修改

```python
if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    parser.add_argument('--weights', type=str, default='yolov5s.pt', help='initial weights path')
    # 修改配置文件
    parser.add_argument('--cfg', type=str, default='models/car.yaml', help='model.yaml path')
    parser.add_argument('--data', type=str, default='data/car.yaml', help='data.yaml path')
    parser.add_argument('--hyp', type=str, default='data/hyp.scratch.yaml', help='hyperparameters path')
    parser.add_argument('--epochs', type=int, default=300)
    parser.add_argument('--batch-size', type=int, default=8, help='total batch size for all GPUs')
    parser.add_argument('--img-size', nargs='+', type=int, default=[640, 640], help='[train, test] image s
    parser.add_argument('--rect', action='store_true', help='rectangular training')
    parser.add_argument('--resume', nargs='?', const=True, default=False, help='resume most recent trainin
    parser.add_argument('--nosave', action='store_true', help='only save final checkpoint')
    parser.add_argument('--notest', action='store_true', help='only test final epoch')
    parser.add_argument('--noautoanchor', action='store_true', help='disable autoanchor check')
    parser.add_argument('--evolve', action='store_true', help='evolve hyperparameters')
    parser.add_argument('--bucket', type=str, default='', help='gsutil bucket')
    parser.add_argument('--cache-images', action='store_true', help='cache images for faster training')
    parser.add_argument('--image-weights', action='store_true', help='use weighted image selection for tra
    parser.add_argument('--device', default='0', help='cuda device, i.e. 0 or 0,1,2,3 or cpu')
    parser.add_argument('--multi-scale', action='store_true', help='vary img-size +/- 50%%')
    parser.add_argument('--single-cls', action='store_true', help='train multi-class data as single-class'
    parser.add_argument('--adam', action='store_true', help='use torch.optim.Adam() optimizer')
    parser.add_argument('--sync-bn', action='store_true', help='use SyncBatchNorm, only available in DDP m
```

```
weights    --预训练权重
cfg        --使用的模型
data       --训练的数据来源
epochs     --训练的轮数
batch-size--图片的批量数
device     --是否使用cuda
```

之后执行train.py文件即可