**视频目标检测**

学生姓名：韩孟涛

学 号：3118311185

班 级：8095

**1实验内容**

利用 SSE4 和多核编程进行运算加速，实现目标检测

**2实验思路**

1. 下载YOLOv3数据集，进行模型训练
2. 利用opencv进行视频拆帧，对拆帧后的图像进行并行处理，多帧并行使用YOLO算法，每一帧都变成了一张图片，进行目标检测，最后将处理好的所有帧进行合并，处理成视频
3. 将视频的结果呈现处理。

**3实验环境**

1. 操作系统为win8 64位；
2. 环境：opencv 3.4,python 3.6,tensorflow,keras 2.1

3) 编程工具为pycharm

**4实验过程及代码实现**

**4.1加载图像，将图像大小进行调整**

为贴合多核思想，本实验所有图像先加载，然后经过调整等resize然后输出为（64,64,3）的图像，加载图像核心代码如下：

def process\_image(img):

image = cv2.resize(img, (416, 416),interpolation=cv2.INTER\_CUBIC)##调整缩小和展开图像

image = np.array(image, dtype='float32')

image /= 255.

image = np.expand\_dims(image, axis=0)

return image ##返回经过处理的图像（64,64,3）

**4.2获取图像的像素点**

在识别类时，,先进行迭代，找到对象检测的矩形框及其分类，其核心代码如下：

def draw(image, boxes, scores, classes, all\_classes):##在图像上画框（原始图像，盒子对象，对象的分数，对象的类，所有的类名）

for box, score, cl in zip(boxes, scores, classes):

x, y, w, h = box

##获取矩形点

top = max(0, np.floor(x + 0.5).astype(int))

left = max(0, np.floor(y + 0.5).astype(int))

right = min(image.shape[1], np.floor(x + w + 0.5).astype(int))

bottom = min(image.shape[0], np.floor(y + h + 0.5).astype(int))

cv2.rectangle(image, (top, left), (right, bottom), (255, 0, 0), 2)##画矩形

cv2.putText(image, '{0} {1:.2f}'.format(all\_classes[cl], score),

(top, left - 6),

cv2.FONT\_HERSHEY\_SIMPLEX,

0.6, (0, 0, 255), 1,

cv2.LINE\_AA)

print('class: {0}, score: {1:.2f}'.format(all\_classes[cl], score))

print('box coordinate x,y,w,h: {0}'.format(box))

print()

此方法主要是在检测图像时进行坐标点的实现和矩形框的绘制，主要是一个输出

**4.3识别某一图像，并进行识别，如分数，类名，框，然后，将这些参数分别传入draw进行图片上的显示**

核心代码如下：

def detect\_image(image, yolo, all\_classes):##使用YOLO v3检测图像（参数是：原始图像，YOLO之前训练好的模型，所有的类名）

pimage = process\_image(image)#加载图像

start = time.time()

boxes, classes, scores = yolo.predict(pimage, image.shape)##用yolo对image进行分类，得分，画框

end = time.time()

print('time: {0:.2f}s'.format(end - start))##总共使用时间

if boxes is not None:##就说明有对象，可以进行画框

draw(image, boxes, scores, classes, all\_classes)

return image

**4.4 视频拆帧**

主要对视频进行拆帧，在yolo中v3网络架构是横纵交叉的，看着卷积层多，其实很多channel的卷积层没有继承性，而且，虽然yolov3增加了anchor centroid，但是对ground truth的估计变得更加简单，每个ground truth只匹配一个先验框，而且每个尺度只预测3个框，增加了并行性，也降低了复杂度

def detect\_video(video, yolo, all\_classes):

"""用yolo v3检测视频

# 参数:

video: 视=视频文件名.

yolo: yolo训练模型.

all\_classes: 所有类名.

"""

video\_path = os.path.join("videos", "test", video)##video路径

camera = cv2.VideoCapture(video\_path)

cv2.namedWindow("detection", cv2.WINDOW\_AUTOSIZE)##识别

# 准备保存检测到的video

sz = (int(camera.get(cv2.CAP\_PROP\_FRAME\_WIDTH)),

int(camera.get(cv2.CAP\_PROP\_FRAME\_HEIGHT)))

fourcc = cv2.VideoWriter\_fourcc(\*'mpeg')

vout = cv2.VideoWriter()

vout.open(os.path.join("videos", "res", video), fourcc, 20, sz, True)

while True:

res, frame = camera.read()

if not res:

break

image = detect\_image(frame, yolo, all\_classes)

cv2.imshow("detection", image)

# 一帧一帧的保存图像

vout.write(image)

if cv2.waitKey(110) & 0xff == 27:

break

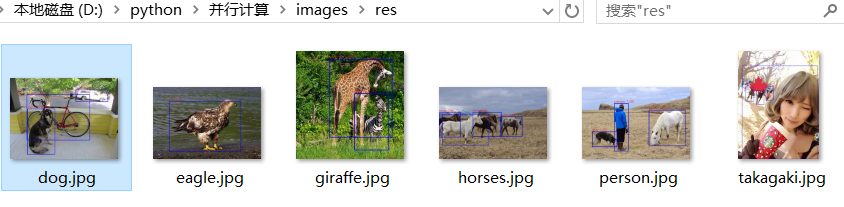
vout.release()

camera.release()

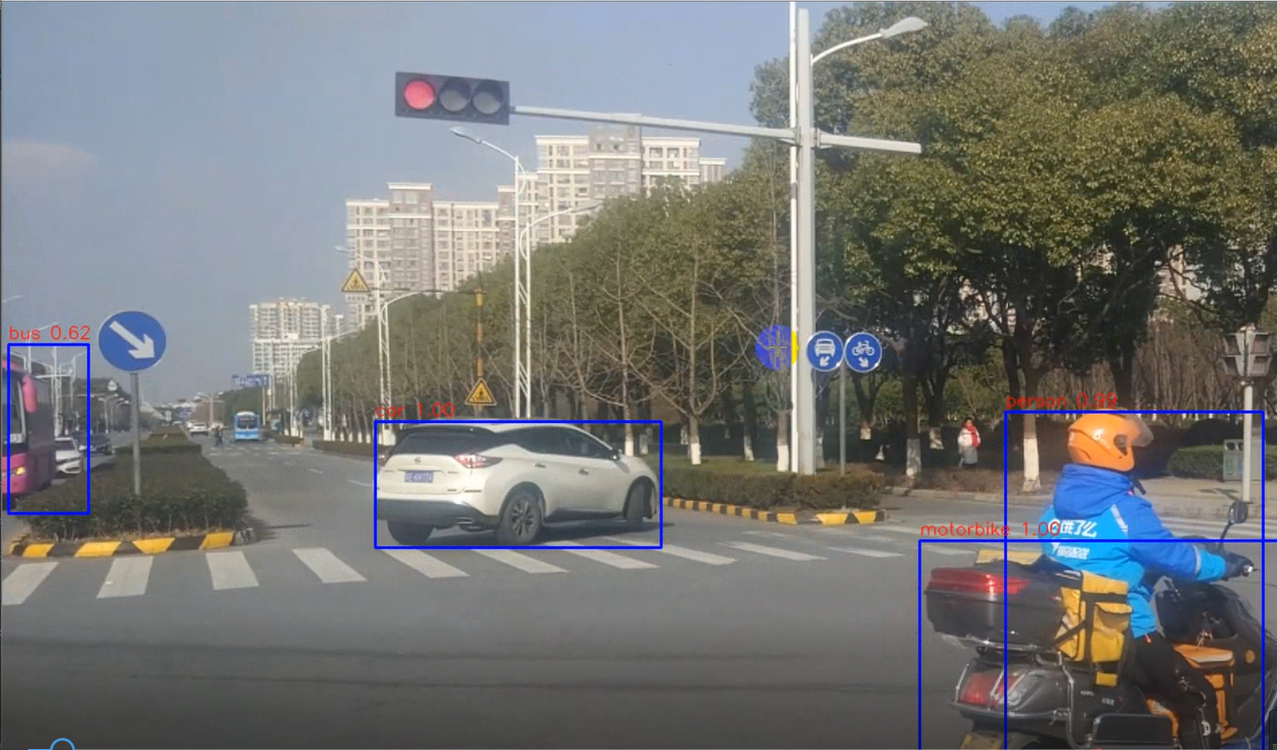
测试结果

图像测试结果和数值坐标结果都分别保存下来。





**5视频截图如下**



**6缺点及不足**

1）没有设计比较环节，无法清晰看出使用SSE4及多核编程的效果；

2）视频在保存的时候不流畅，程序总体较为简单