学生实验报告

学号	1120192933	学院	计算机学院
姓名	李桐	专业	人工智能

基于 json 的数据预处理流程

1 实验目的

（1）理解 json 文件在数据科学中的应用。
（2）掌握读写 json 文件的方法。
（3）掌握图像和视频的读取方法。

2 实验原理

（1）json 对象的格式与处理。
（2）argparse 的使用。
（3）opencv 和 PIL 库的使用。

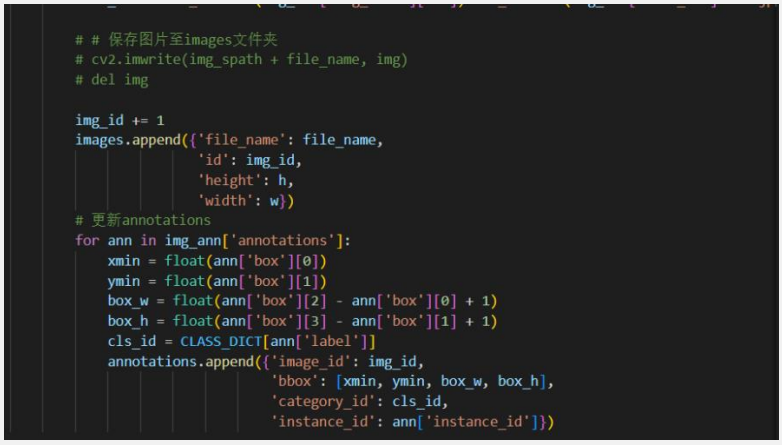
3 实验条件与环境

要求	名称	版本要求	备注
编程语言	python	3.6 以上	
开发环境	dsw	无要求	
第三方工具包/库/插件	opencv-python	4.5 以上	
第三方工具包/库/插件	tqdm	4.32	
其他工具	无	无要求	

硬件环境	台式机、笔记本均可	无要求	

4 实验步骤及操作

序号	步骤名称	步骤描述	代码
1	预处理数据	对训练数据进行预处理	
2	合并数据集	将多个数据集文件的标注信息合并	

步骤序号	1
步骤名称	预处理数据
步骤描述	(1)对图像库数据集进行标注文件的准备。 (2)对视频库直播切片进行标注文件的准备。
代码及讲解	 (1)对图像库数据集进行标注文件的准备 这一步的做法是先把图片读入进来，获取图像路径 ip 对应的标注路径 ap，再加上相应的标签，保存为 json 文件。 (2)对视频库直播切片进行标注文件的准备 获取视频路径 p 对应的标注路径 vap。之后的做法和上一步相似，不过处理的时候是对单个 frame 的操作。

```
# # 保存图片至images文件夹
# cv2.imwrite(img_spath + file_name, img)
# del img

img_id += 1
images.append({'file_name': file_name,
               'id': img_id,
               'height': h,
               'width': w})
# 更新annotations
for ann in img_ann['annotations']:
    xmin = float(ann['box'][0])
    ymin = float(ann['box'][1])
    box_w = float(ann['box'][2] - ann['box'][0] + 1)
    box_h = float(ann['box'][3] - ann['box'][1] + 1)
    cls_id = CLASS_DICT[ann['label']]
    annotations.append({'image_id': img_id,
                        'bbox': [xmin, ymin, box_w, box_h],
                        'category_id': cls_id,
                        'instance_id': ann['instance_id']})
```

步骤序号	2
步骤名称	合并数据集
步骤描述	将多个数据集文件的标注信息合并。

代码及讲解	

```
for vp in tqdm(video_paths):
    # 获取视频路径p对应的标注路径vap
    vap = vp.replace('video', 'video_annotation')
    vap = vap.replace('mp4', 'json')

    with open(vap, 'r') as json_f2:
        video_ann = json.load(json_f2)

    for frame in video_ann['frames']:
        # 如果单个frame下没有标注:
        if len(frame['annotations']) == 0:
            pass
        # 如果单个frame下有标注:
        else:
            frame_index = frame['frame_index']
            frame_img = get_frame_img(vp, frame_index)

            vh, vw, _ = frame_img.shape
            del frame_img
            # 更新images
            img_id += 1
            vfile_name = 'v_' + str(video_ann['video_id']) + '_' + str(frame_index) + '.jpg'
            images.append({'file_name': vfile_name,
```

因为数据集太大了，分了很多个压缩包，分别处理完每一个之后还需要进行合并。重要的代码如上图所示。

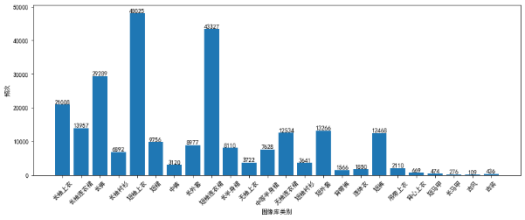
5 实验结果及分析讨论

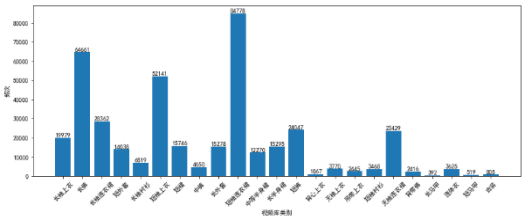
1.实验结果就是保存了 json 文件，文件里面是这个样子:

annotationstrainval.json

"}, {"id": 21, "name": "\u540a\u5e26\u4e0a\u8863"}, {"id": 21, "name": "\u540a\u5e26Top"}, {"id": 22, "name": "\u4e2d\u88e4"},

2.类别分布不怎么均匀，分布如下图所示:





‘长袖衬衫’，‘中裤’，‘背心上衣’，‘无袖上衣’，‘吊带上衣’，‘短袖衬衫’，‘背带裤’，‘连体衣’，‘长马甲’，‘短马甲’，‘古风’，‘古装‘在视频和图片里面数量都在 7000 以下。其他的类别的数量 10000-90000 之间。

3.每个衣服基本都有好几张图片，大部分是不同的姿势和颜色，感觉识别还是挺有难度的:


0.jpg


1.jpg

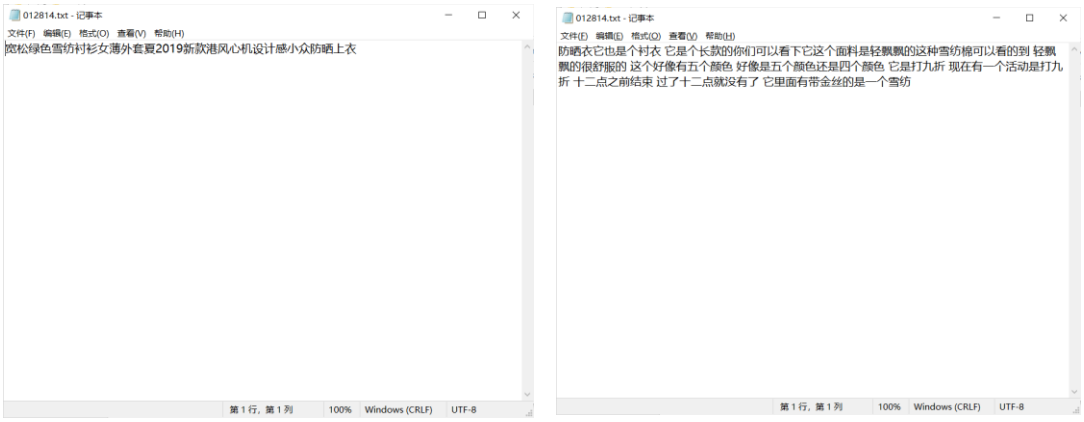

2.jpg


3.jpg


4.jpg


5.jpg

4.但是我看了一下文本，文本里面体现的类别比较清晰。图片的文本应该是商品的标题，视频的文本是视频对应的话。基本都清晰的体现了类别:



5.数据量很大，而且模型本身也不是很简单，预计需要训练好久。总之在第一个实验里面了解了数据集的组成、结构，学习了对视频和图片对应的处理方法，并且进行了预处理。

6.根据官方说法，定义了 23 类标注类别，在 148 万张标注图像上总计标注 236 万商品检测框。标注框统计分布如下：

Scale 分布：对于全部的商品检测框，按照标注框在整副图像中的面积占比划分为：large、moderate、small。其中面积占比小于 10%的定义为 small，面积占比介于 10%到 40%的定义为 moderate，面积占比大于 40%的定义为 large。scale 信息可以有效的应用于检测及特征训练，提升算法精度。

Viewpoint 分布：在商品框标注时标注了商品展示的视角（viewpoint: 0-正面，1-背面，2-左侧，3-右侧）信息。其中 86%的商品都是以正面展示为主。在商品识别阶段引入视角信息，避免不同视角的商品误匹配可以有效提升识别精度。

Display 分布：在商品框标注时标注了商品的展示方式（display: 0-纯商品展示，1-试穿展示）信息。纯商品展示至渲染商品图、主播手提展示等情况，试穿展示指模特或者主播试穿展示。

6 收获与体会

一直以来都不太会做 json 的操作。这个实验里面接触了比较多的 json 操作。其实主要就 4 个操作。

操作	作用
json.dumps()	将 Python 对象编码成 JSON 字符串
json.loads()	将已编码的 JSON 字符串解码为 Python 对象
json.dump()	将 Python 内置类型序列化为 json 对象后写入文件
json.load()	读取文件中 json 形式的字符串元素转化为 Python 类型

刚接触的时候一直有点抗拒，然后发现其实和字典的用法差不多。也没有很难。

7 备注及其他

无。