LabelImg标注的VOC格式xml标签与YOLO格式txt标签相互转换_wangmj_hdu的博客-程序员宝宝

技术标签： 深度学习 (/searchArticle?qc=深度学习&page=1) pytorch (/searchArticle?qc=pytorch&page=1) 神经网络 (/searchArticle?qc=神经网络&page=1)

1、VOC标签格式说明

VOC数据格式，会直接把每张图片标注的标签信息保存到一个xml文件中。

例如我在做仓储托盘检测的时候，需要对图片中的托盘进行标注，标注的标签信息会保存到一个跟图片对应的xml文件中（每张图片与每个xml文件一一对应），xml中的信息如下：

```
<annotation>
        <folder>Images</folder>
        <filename>1.jpg</filename>
        <path>/home/wangmj/pallet_data/Images/1.jpg</path>
        <source>
                <database>Unknown</database>
        </source>
        <size>
                <width>1920</width>
                <height>1080</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>forklift_pallet</name>
                <pose>Unspecified</pose>
                <truncated>0</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>437</xmin>
                        <ymin>557</ymin>
                        <xmax>1230</xmax>
                        <ymax>660</ymax>
                </bndbox>
        </object>
</annotation>
```

xml文件中的关键信息说明：

- 1.jpg，这是图片名称，则xml文件名为1.xml；
- /home/wangmj/pallet_data/Images/1.jpg，这是存放该图片的绝对路径；
- 1920 * 1080，这是图片分辨率，3代表三通道图片；
- forklift_pallet，这是类别名；
- xmin，ymin，xmax，ymax，定义了每个目标的标定框坐标：即左上角的坐标和右下角的坐标；

2、YOLO标签格式说明

YOLO标签格式，会直接把每张图片标注的标签信息保存到一个txt文件中。

我的图片名称为1.jpg，则对应的txt文件名称为1.txt。

同样例如我在做仓储托盘检测的时候，需要对图片中的托盘进行标注，标注的标签信息会保存到一个跟图片对应的txt文件中，txt中的信息如下：

```
0 0.433594 0.562037 0.409896 0.092593
```

txt文件中的关键信息说明：

- 每一行代表标注的一个目标，我这张图中只标注了一个目标，所以只有一行；
- 第一个数字0代表标注目标的类别；
- 后面四个数字代表标注框的中心坐标和标注框的相对宽和高（进行了归一化处理）；
- 五个数据从左到右依次为： class x_center y_center width height

同时还会生成一个classes.txt，里面内容如下：

```
forklift_pallet
```

我只有forklift_pallet这一种类型。

3、voc格式转化为yolo格式

标注好的voc格式的标签xml文件，主要信息为：

- 1.jpg，这是图片名称；
- /home/wangmj/pallet_data/Images/1.jpg，这是存放该图片的绝对路径；
- 1920 * 1080，这是图片分辨率，3代表三通道图片；
- forklift_pallet，这是类别名；
- xmin，ymin，xmax，ymax，定义了每个目标的标定框坐标：即左上角的坐标和右下角的坐标；

例如下面一张图：

- 原图大小为1920 * 1080
- 紫色框代表标注物体的框，紫色框的左上角的坐标为（xmin，ymin）=（372，518），右下角的坐标为（xmax，ymax）=（1344，674）

voc_to_yolo.py的目的就是把voc数据格式转换为yolo格式：

- voc格式标签：图片的实际宽高，标注框的左上角和右下角坐标；
- yolo格式标签：标注框的中心坐标（归一化），标注框的宽和高（归一化）。

voc格式转换为yolo格式计算公式：

- 框中心的实际坐标（x，y），一般可能还会在后面减1

$$x\_center = \frac{xmax + xmin}{2}$$

$$y\_center = \frac{ymax + ymin}{2}$$

- 归一化以后的中心坐标（x，y）

$$x = \frac{x\_center}{width}$$

$$y = \frac{y\_center}{height}$$

- 框的高和宽（归一化后）

$$w = \frac{xmax - xmin}{width}$$

$$h = \frac{ymax - ymin}{height}$$

voc格式的xml标签文件转化yolo格式的txt标签文件代码：voc_to_yolo.py

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

def convert(size, box):
    # size=(width, height)  b=(xmin, xmax, ymin, ymax)
    # x_center = (xmax+xmin)/2        y_center = (ymax+ymin)/2
    # x = x_center / width            y = y_center / height
    # w = (xmax-xmin) / width         h = (ymax-ymin) / height

    x_center = (box[0]+box[1])/2.0
    y_center = (box[2]+box[3])/2.0
    x = x_center / size[0]
    y = y_center / size[1]

    w = (box[1] - box[0]) / size[0]
    h = (box[3] - box[2]) / size[1]

    # print(x, y, w, h)
    return (x,y,w,h)

def convert_annotation(xml_files_path, save_txt_files_path, classes):
    xml_files = os.listdir(xml_files_path)
    # print(xml_files)
    for xml_name in xml_files:
        # print(xml_name)
        xml_file = os.path.join(xml_files_path, xml_name)
        out_txt_path = os.path.join(save_txt_files_path, xml_name.split('.')[0] + '.txt')
        out_txt_f = open(out_txt_path, 'w')
        tree=ET.parse(xml_file)
        root = tree.getroot()
        size = root.find('size')
        w = int(size.find('width').text)
        h = int(size.find('height').text)

        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(xm
lbox.find('ymin').text), float(xmlbox.find('ymax').text))
            # b=(xmin, xmax, ymin, ymax)
            # print(w, h, b)
            bb = convert((w,h), b)
            out_txt_f.write(str(cls_id) + " " + " ".join([str(a) for a in bb]) + '\n')


if __name__ == "__main__":
    # 把forklift_pallet的voc的xml标签文件转化为yolo的txt标签文件
    # 1、需要转化的类别
```

```
classes = ['forklift_pallet']
# 2、voc格式的xml标签文件路径
xml_files1 = r'/home/wangmj/pallet_data/Annotations'
# 3、转化为yolo格式的txt标签文件存储路径
save_txt_files1 = r'/home/wangmj/pallet_data/test'

convert_annotation(xml_files1, save_txt_files1, classes)
```

4、yolo格式转化为voc格式

voc格式中保存的信息为：xmin，ymin，xmax，ymax，所以只要根据上面的公式，就可以推导出这四个值。

yolo格式的txt标签文件转化voc格式的xml标签文件代码：yolo_to_voc.py

```
import os
import xml.etree.ElementTree as ET
from xml.dom.minidom import Document
import cv2

'''
import xml
xml.dom.minidom.Document().writexml()
def writexml(self,
             writer: Any,
             indent: str = "",
             addindent: str = "",
             newl: str = "",
             encoding: Any = None) -> None
'''

class YOLO2VOCConvert:
    def __init__(self, txts_path, xmls_path, imgs_path):
        self.txts_path = txts_path   # 标注的yolo格式标签文件路径
        self.xmls_path = xmls_path   # 转化为voc格式标签之后保存路径
        self.imgs_path = imgs_path   # 读取读片的路径各图片名字，存储到xml标签文件中
        self.classes = ['forklift_pallet']

    # 从所有的txt文件中提取出所有的类别， yolo格式的标签格式类别为数字 0,1,...
    # writer为True时，把提取的类别保存到'./Annotations/classes.txt'文件中
    def search_all_classes(self, writer=False):
        # 读取每一个txt标签文件，取出每个目标的标注信息
        all_names = set()
        txts = os.listdir(self.txts_path)
        # 使用列表生成式过滤出只有后缀名为txt的标签文件
        txts = [txt for txt in txts if txt.split('.')[-1] == 'txt']
        print(len(txts), txts)
        # 11 ['0002030.txt', '0002031.txt', ... '0002039.txt', '0002040.txt']
        for txt in txts:
            txt_file = os.path.join(self.txts_path, txt)
            with open(txt_file, 'r') as f:
                objects = f.readlines()
                for object in objects:
                    object = object.strip().split(' ')
                    print(object)  # ['2', '0.506667', '0.553333', '0.490667', '0.658667']
                    all_names.add(int(object[0]))
            # print(objects)  # ['2 0.506667 0.553333 0.490667 0.658667\n', '0 0.496000 0.2
85333 0.133333 0.096000\n', '8 0.501333 0.412000 0.074667 0.237333\n']

        print("所有的类别标签：", all_names, "共标注数据集：%d张" % len(txts))

        return list(all_names)

    def yolo2voc(self):
        # 创建一个保存xml标签文件的文件夹
        if not os.path.exists(self.xmls_path):
            os.mkdir(self.xmls_path)

        # 把上面的两个循环改写成为一个循环：
        imgs = os.listdir(self.imgs_path)
```

```
        txts = os.listdir(self.txts_path)
        txts = [txt for txt in txts if not txt.split('.')[0] == "classes"]  # 过滤掉classes.txt文件
        print(txts)
        # 注意，这里保持图片的数量和标签txt文件数量相等，且要保证名字是一一对应的   (后面改进，通过判断txt文件名是否在imgs中即可)
        if len(imgs) == len(txts):   # 注意：./Annotation_txt 不要把classes.txt文件放进去
            map_imgs_txts = [(img, txt) for img, txt in zip(imgs, txts)]
            txts = [txt for txt in txts if txt.split('.')[-1] == 'txt']
            print(len(txts), txts)
            for img_name, txt_name in map_imgs_txts:
                # 读取图片的尺度信息
                print("读取图片：", img_name)
                img = cv2.imread(os.path.join(self.imgs_path, img_name))
                height_img, width_img, depth_img = img.shape
                print(height_img, width_img, depth_img)   # h 就是多少行（对应图片的高度），w就是多少列（对应图片的宽度）

                # 获取标注文件txt中的标注信息
                all_objects = []
                txt_file = os.path.join(self.txts_path, txt_name)
                with open(txt_file, 'r') as f:
                    objects = f.readlines()
                    for object in objects:
                        object = object.strip().split(' ')
                        all_objects.append(object)
                        print(object)  # ['2', '0.506667', '0.553333', '0.490667', '0.658667']

                # 创建xml标签文件中的标签
                xmlBuilder = Document()
                # 创建annotation标签，也是根标签
                annotation = xmlBuilder.createElement("annotation")

                # 给标签annotation添加一个子标签
                xmlBuilder.appendChild(annotation)

                # 创建子标签folder
                folder = xmlBuilder.createElement("folder")
                # 给子标签folder中存入内容，folder标签中的内容是存放图片的文件夹，例如：JPEGImages
                folderContent = xmlBuilder.createTextNode(self.imgs_path.split('/')[-1])  # 标签内存
                folder.appendChild(folderContent)  # 把内容存入标签
                annotation.appendChild(folder)   # 把存好内容的folder标签放到 annotation根标签下

                # 创建子标签filename
                filename = xmlBuilder.createElement("filename")
                # 给子标签filename中存入内容，filename标签中的内容是图片的名字，例如：000250.jpg
                filenameContent = xmlBuilder.createTextNode(txt_name.split('.')[0] + '.jpg')  # 标签内容
                filename.appendChild(filenameContent)
                annotation.appendChild(filename)
```

```
                # 把图片的shape存入xml标签中
                size = xmlBuilder.createElement("size")
                # 给size标签创建子标签width
                width = xmlBuilder.createElement("width")  # size子标签width
                widthContent = xmlBuilder.createTextNode(str(width_img))
                width.appendChild(widthContent)
                size.appendChild(width)   # 把width添加为size的子标签
                # 给size标签创建子标签height
                height = xmlBuilder.createElement("height")  # size子标签height
                heightContent = xmlBuilder.createTextNode(str(height_img))  # xml标签中存入
的内容都是字符串
                height.appendChild(heightContent)
                size.appendChild(height)  # 把width添加为size的子标签
                # 给size标签创建子标签depth
                depth = xmlBuilder.createElement("depth")  # size子标签width
                depthContent = xmlBuilder.createTextNode(str(depth_img))
                depth.appendChild(depthContent)
                size.appendChild(depth)  # 把width添加为size的子标签
                annotation.appendChild(size)   # 把size添加为annotation的子标签

                # 每一个object中存储的都是['2', '0.506667', '0.553333', '0.490667', '0.65866
7']一个标注目标
                for object_info in all_objects:
                    # 开始创建标注目标的label信息的标签
                    object = xmlBuilder.createElement("object")  # 创建object标签
                    # 创建label类别标签
                    # 创建name标签
                    imgName = xmlBuilder.createElement("name")  # 创建name标签
                    imgNameContent = xmlBuilder.createTextNode(self.classes[int(object_info
[0])])
                    imgName.appendChild(imgNameContent)
                    object.appendChild(imgName)  # 把name添加为object的子标签

                    # 创建pose标签
                    pose = xmlBuilder.createElement("pose")
                    poseContent = xmlBuilder.createTextNode("Unspecified")
                    pose.appendChild(poseContent)
                    object.appendChild(pose)  # 把pose添加为object的标签

                    # 创建truncated标签
                    truncated = xmlBuilder.createElement("truncated")
                    truncatedContent = xmlBuilder.createTextNode("0")
                    truncated.appendChild(truncatedContent)
                    object.appendChild(truncated)

                    # 创建difficult标签
                    difficult = xmlBuilder.createElement("difficult")
                    difficultContent = xmlBuilder.createTextNode("0")
                    difficult.appendChild(difficultContent)
                    object.appendChild(difficult)

                    # 先转换一下坐标
                    # (objx_center, objy_center, obj_width, obj_height)->(xmin，ymin, xmax,
ymax)
                    x_center = float(object_info[1])*width_img + 1
                    y_center = float(object_info[2])*height_img + 1
```

```
                    xminVal = int(x_center - 0.5*float(object_info[3])*width_img)   # objec
t_info列表中的元素都是字符串类型
                    yminVal = int(y_center - 0.5*float(object_info[4])*height_img)
                    xmaxVal = int(x_center + 0.5*float(object_info[3])*width_img)
                    ymaxVal = int(y_center + 0.5*float(object_info[4])*height_img)

                    # 创建bndbox标签(三级标签)
                    bndbox = xmlBuilder.createElement("bndbox")
                    # 在bndbox标签下再创建四个子标签(xmin，ymin，xmax,ymax) 即标注物体的坐标
和宽高信息
                    # 在voc格式中，标注信息：左上角坐标（xmin，ymin） （xmax，ymax）右下角坐
标
                    # 1、创建xmin标签
                    xmin = xmlBuilder.createElement("xmin")  # 创建xmin标签（四级标签）
                    xminContent = xmlBuilder.createTextNode(str(xminVal))
                    xmin.appendChild(xminContent)
                    bndbox.appendChild(xmin)
                    # 2、创建ymin标签
                    ymin = xmlBuilder.createElement("ymin")  # 创建ymin标签（四级标签）
                    yminContent = xmlBuilder.createTextNode(str(yminVal))
                    ymin.appendChild(yminContent)
                    bndbox.appendChild(ymin)
                    # 3、创建xmax标签
                    xmax = xmlBuilder.createElement("xmax")  # 创建xmax标签（四级标签）
                    xmaxContent = xmlBuilder.createTextNode(str(xmaxVal))
                    xmax.appendChild(xmaxContent)
                    bndbox.appendChild(xmax)
                    # 4、创建ymax标签
                    ymax = xmlBuilder.createElement("ymax")  # 创建ymax标签（四级标签）
                    ymaxContent = xmlBuilder.createTextNode(str(ymaxVal))
                    ymax.appendChild(ymaxContent)
                    bndbox.appendChild(ymax)

                    object.appendChild(bndbox)
                    annotation.appendChild(object)  # 把object添加为annotation的子标签
                f = open(os.path.join(self.xmls_path, txt_name.split('.')[0]+'.xml'), 'w')
                xmlBuilder.writexml(f, indent='\t', newl='\n', addindent='\t', encoding='ut
f-8')
                f.close()

if __name__ == '__main__':
    # 把yolo的txt标签文件转化为voc格式的xml标签文件
    # yolo格式txt标签文件相对路径
    txts_path1 = './test_txt'
    # 转化为voc格式xml标签文件存储的相对路径
    xmls_path1 = './test_xml'
    # 存放图片的相对路径
    imgs_path1 = './Images'

    yolo2voc_obj1 = YOLO2VOCConvert(txts_path1, xmls_path1, imgs_path1)
    labels = yolo2voc_obj1.search_all_classes()
    print('labels: ', labels)
    yolo2voc_obj1.yolo2voc()
```

如何使用Labelimg工具标注图片 (https://blog.csdn.net/wangmj_hdu/article/details/116992986)

智能推荐

05《Go语言入门》使用GoLand创建、运行和调试Hello, World程序_干志雄的博客-程序员宝宝 (/article/oHaiKuoTianKong1682/6983578)

目录Go语言开发用什么IDE使用GoLand创建Hello, WorldRunDebugRun/Debug 快捷键Go语言开发用什么IDEGo语言开发能使用的IDE可多了，如Eclipse、VSCode、Atom等，甚至有不少Gopher使用Vim。但是，我还是建议初学者使用GoLand作为Go语言开发的IDE，理由很简单，其他IDE需要安装Go插件，GoLand开箱即用，且功能更强大。唯一缺点就是使用GoLand需要付费。虽然网上有需要共享的License或破解方法，但是还是推荐大家使用正版。」

ue4 大场景优化_一星程序饲养员-程序员宝宝_ue4场景优化 (/article/ys_ys_y/6983570)

1.PrecomputedVisibilityVolume预计算遮挡，在遮挡较多的地方可以有效提高渲染效率2.CullDistanceVolume根据尺寸裁剪模型，可以有效减少渲染量，缺点是裁剪相关参数需要考究

树结构练习——排序二叉树的中序遍历 sdut oj(2128)_rain_snowing的博客-程序员宝宝 (/article/rain_snowing/6983557)

插入元素建二叉排序树

unity3d android获取app 包名_思玉-程序员宝宝_unity 获取安卓包名 (/article/qqo_aa/6983552)

unity3d android获取app 包名废话不多直接上代码 public static string GetPackageName() { string _strPackageName = "null"; if (string.IsNullOrEmpty(_strPackageName)) { _strPackageName = "null";#if (UNITY_ANDROID && !UN

linux安装hadoop2.7.2_我的座右铭-程序员宝宝 (/article/boy317/6983551)

安装：配置互信 配置互信链接hadoop下载 hadoop官方下载二进制压缩包解压压缩包tar -xzvf hadoop-2.7.2.tar.gz 将其安装...安装位置...中在...安装位置.../hadoop-2.7.2下创建数据存放的文件夹。tmp、hdfs、hdfs/data、hdfs/name修改...安装位置.../hadoop-2.7.2/etc/hadoop下的core-site.xm

---

leetcode_3.31- （python-简单）_weixin_42648803的博客-程序员宝宝 (/article/weixin_42648803/6983543)

leetcode_3.31- （python-简单）—>Two Sum这个，没看明白。参考暴力解法，完成第一个leetcode。基础知识回顾：return须在def下，jupyter下，格式是：nums = [2, 7, 11, 15]target = 9而不是nums = [2, 7, 11, 15]，target = 9https://www.jianshu.com...

随便推点

UE4蓝图案例_风里有诗句哈的博客-程序员宝宝 (/article/Vansal/112371530)

物体会自动来回移动，且被子弹打中后会变颜色，1秒后恢复原先的颜色人物加速跑和瞄准二倍镜效果

2021-07-19 王道 数据结构 p17 第1题_one day_190304的博客-程序员宝宝 (/article/qq_44801423/118913327)

#王道 数据结构 p17 第1题从顺序表中删除具有最小值的元素（假设唯一）并由函数返回被删元素的值。空出的位置由最后一个元素填补，若顺序表为空，则显示出错信息并退出运行。#include<iostream>#define InitSize 100#define arrLen 5using namespace std;typedef struct { //声明一个动态分配内存的顺序表 int *data; int len,MaxSize;}SeqList;void ini

---

Fragment相关_Kelvin_Zou的博客-程序员宝宝 (/article/Kelvin_Zou/86648084)

Fragment的生命周期官网对生命周期方法的介绍The core series of lifecycle methods that are called to bring a fragment up to resumed state (interacting with the user) are:onAttach(Activity) called once the fragment i...

UE4 打包_DuGuYiZhao的博客-程序员宝宝_ue4 打包 (/article/DuGuYiZhao/121961076)

Unreal Engine4 打包

MySQL - ROUND 函数真能“完全”保留小数位数吗？_牧码的博客-程序员宝宝_mysql round函数保留0 (/article/Dream_Weave/118335951)

select ROUND(1.001, 2); // 1select ROUND(1.00, 2); // 1是不是有点不可思议，哈哈哈~所以我后面用 FORMAT 函数或 DECIMAL 函数!

python金融计算：浮点数不能转化为整型_Zhoudian_bufu的博客-程序员宝宝 (/article/Zhoudian_bufu/103905856)

python金融计算的时候定义了公式然后遇到下面这个问题：python2的视频，我的代码应该没错，用的python3，应该怎么解决呢，求指导)**def bsm_mcs_valuation(strike):import numpy as npS0 = 100.;T=1.0;r=0.05;vola=0.2M=50;I=20000dt = T/Mrand = np.random.stan...

推荐文章

使用mocha编写pomelo项目的单元测试_baidu20008的专栏-程序员宝宝 (/article/baidu20008/45871537)

数据孤岛问题_hxxjxw的博客-程序员宝宝_数据孤岛问题 (/article/hxxjxw/108193018)

jwt生成/加密/解密token_一影清歌的博客-程序员宝宝_jwt加密解密 (/article/weixin_42941045/106478372)

微信小程序JavaScript判断值是否为空工具类方法_小歲月、太着急-程序员宝宝_微信小程序判断字符串为空 (/article/Small_years/106482477)

oracle数据库的表数据导出为csv文件_eigo的聊哥儿-程序员宝宝_toad 导出csv (/article/eigo/1571032)

tableview 自动移动到某个cell_过期的码农-程序员宝宝 (/article/or7rccl/18803151)

macOS Monterey 12.0.1 (21A559) 虚拟机 IOS 镜像_happyG_G的博客-程序员宝宝_mac虚拟机镜像 (/article/qq_28735663/120986665)

以太网数据帧（802.3）最大与最小长度_farmwang的专栏-程序员宝宝_以太网帧的最小长度和最大长度 (/article/farmwang/64131318)

热门文章

Mesos和Marathon下容器无法正常部署可能的原因_学无止境-程序员宝宝 (/article/qq_33121481/111043073)

Android Studio&源码混淆配置及其调试注意事项_空白的泡的博客-程序员宝宝_android studio 混淆配置 (/article/kongbaidepao/88743987)

转：Emacs org-mode: 最好的文档编辑利器，没有之一_jueshu的博客-程序员宝宝 (/article/jueshu/84821637)

哈夫曼编码和译码的实现_ Pink_floyd的博客-程序员宝宝_哈夫曼编码译码算法的实现 (/article/qq_42932834/94639190)

Windows server 2008关机提示取消方法_网吧系统-程序员宝宝_server2008关机提示 (/article/wb_system/47003387)

在整型有序数组中查找想要的数字.（折半查找）_ambiguous__的博客-程序员宝宝 (/article/ambiguous__/89281161)

git id_rsa are too open的解决办法_Gabriel的专栏-程序员宝宝 (/article/gf771115/49928821)

相似三角形 oj_七九河开的博客-程序员宝宝 (/article/sdut_jk17_zhangming/79100260)

相关标签

深度学习 (/searchArticle?qc=深度学习&page=1)

pytorch (/searchArticle?qc=pytorch&page=1)

神经网络 (/searchArticle?qc=神经网络&page=1)