第一, 对比tensorflow ssd项目的300和640的区别; 整理出coco,voc2007等数据集本身的尺寸(是否是统一尺寸的, 找出两张图片查看一下)

第二, 分析出pascal\_voc.py含义.

第三. 分析出gen\_bbox??.py的含义. 如果能单独测试并添加log辅助理解更好了.

第四, 解决caffe ssd编译出现的问题. 不行的话,就换caffe版本.

第五, 尝试从上面的理解来看怎么做128\*128的ssd的调整.

一 数据集尺寸理解

* 1. 考察voc2007

考察voc2007的两张图片:

006644.jpg

其对应的分辨率是:500\*375

中间的人的对应的人体bbox:

Width=87, height=290

中间头bbox:

26, 32

占比为:

考察008305.jpg



尺寸是 500\*406

Bbox是299\*150

|  |  |
| --- | --- |
| 006644.jpg  500\*375  中间的人的对应的人体bbox:  Width=87, height=290  中间头bbox:  26, 32 | 008305.jpg  尺寸是 500\*406  Bbox是299\*150 |

* 1. tf300和640的区别

|  |  |
| --- | --- |
| image\_resizer {  fixed\_shape\_resizer {  height: **640**  width: **640**  }  } | image\_resizer {  fixed\_shape\_resizer {  height: **300**  width: **300**  }  } |

主要是imgresize不一样.

这里tf采用了不同的人resizer

如下所示.

|  |
| --- |
| **def** \_tf\_resize\_method**(**resize\_method**):**  """Maps image resize method from enumeration type to TensorFlow.  Args:  resize\_method: The resize\_method attribute of keep\_aspect\_ratio\_resizer or  fixed\_shape\_resizer.  Returns:  method: The corresponding TensorFlow ResizeMethod.  Raises:  ValueError: if `resize\_method` is of unknown type.  """  dict\_method **=** **{**  **image\_resizer\_pb2.BILINEAR:**  **tf.image.ResizeMethod.BILINEAR,**  **image\_resizer\_pb2.NEAREST\_NEIGHBOR:**  **tf.image.ResizeMethod.NEAREST\_NEIGHBOR,**  **image\_resizer\_pb2.BICUBIC:**  **tf.image.ResizeMethod.BICUBIC,**  **image\_resizer\_pb2.AREA:**  **tf.image.ResizeMethod.AREA**  **}**  **if** resize\_method **in** dict\_method**:**  **return** dict\_method**[**resize\_method**]**  **else:**  **raise** ValueError**(**'Unknown resize\_method'**)** |

1.3 fid的尺寸

|  |  |
| --- | --- |
| voc头,但实际没有这个类  26/500 = 5%  32/375 = 8.53%  实际有类别的:  87/500 = 17.4%  290/375 = 77.3% | fid数据的Head bbox大小  87/2688 = 3.2%  109/1520 = 7.17%  如果缩放成128\*128的,其bbox会为  W’=87\*128/2688=4.14  H’=109\*128/1520=9.17  **则只有36个像素.这样肯定效果特别差!**  **负样本的loss会很小, 正样本的Loss就下不来.** |

* 1. pascal\_voc.py

xml是voc格式的.

1. Xmin,ymin,xmax,ymax组合成一个object
   1. hd\_gen\_positives.py

就是把pascal或其他格式的xml读出来,把raw的bbox做成(resize成)一个矩形框.然后输出一个文件(比较精炼的), 一行是一个sample.

照片路径,1,bbox位置(依次排放).

* 1. 尝试从上面的理解来做128\*128的

缩放肯定是不成的.

只有做patches.

如果做patches?

Ssd的输入图是128的. 这个不必须说原bbox左右扩城128,这样负样本反而少了.

**适当在原bbox附件扣出一个300\*300的图.**

**然后缩放到128\*128,感觉应该是可以的.**

**那应该一个原来的sample会变成N个sample(每个框对应一个sample).**

先做这一步了.

分解动作:

1. 抠图demo代码
2. Resize demo
3. 生成对应的xml文件.
4. 批量生成VOC路径代码.

Is#0 baseline部分. 找lifeng的densebox作为baseline.

Gpu805上: /ssd/hnren/0\_DATA/head\_detection/fid\_0

Gpu803上:

root@80dfe1e72f76:/ssd/hnren/tf/1sd/caffe# echo $PYTHONPATH

/ssd/hnren/tf/tf\_base/refineDet\_caffe/python

Loss是0, gt框太小了.导致像素太少.

数据集环境整理

|  |
| --- |
| [hnren@gpu803 5\_patches300to128\_INT]$ ll  4\_hd\_gen\_patches\_voc\_andresize.py  HeadVocFormat  pascal\_voc.py  pascal\_voc.pyc    [hnren@gpu803 HeadVocFormat]$ ll  FID\_DID\_HEAD\_CLEAN\_0  FID\_DID\_HEAD\_CLEAN\_0\_patches\_int    [hnren@gpu803 FID\_DID\_HEAD\_CLEAN\_0\_patches\_int]$ ll  Annotations  auto\_create\_.sh  cls.py  create\_data.sh  create\_list.sh  ImageSets  JPEGImages  labelmap\_voc.prototxt  test\_name\_size.txt  test.txt  trainval.txt  1. 在"Annotations"和"JPEGImages"同级目录下建立ImageSets/Main的目录结构.  2. 修改cls.py的目录到ImageSets/Main的上级目录.  3. 运行cls.py以便在ImageSets/Main中产生分割好的数据集(训练集,验证集,测试集)(以样本名,图片名,保存)  4. 修改create\_list.sh中目录,并执行.  5. 修改create\_data.sh中目录,并执行. |