# 警示：每个工程维护一个工程文档

# 维护一个环境文档，在更改环境的时候要在环境文档中进行记录

**工程进度记录：**

2019/4/1

记录：

上周末发现代码不能运行的问题，是一个输入不匹配（input0:[1,128,128,256] != input 1:[128\*128\*256]）尝试过重新拷贝原MASK-RCNN中的model文件，发现不能解决问题，使用关键字不匹配进行查询发现可能是TensorFlow和python版本不匹配的问题，因为之前发生了一些事情（忘接了）导致我重新安装了另一个版本的TensorFlow，于是重新安装了一个环境，

**语句：**pip install --upgrade --ignore-installed tensorflow

pip install keras

pip install scikit-image

代码已可以运行

之前的总结：

1. 生成数据集的时候，由于代码中使用的是 [polygon](http://www.baidu.com/link?url=c_Gf9ccUSgdX12uttPaE5jSpd11SFBdab0tkZoC1w5TZ_CgHThdI5GcxloUuyRimunWcYWm-nQupBI72levq5201nf2HhO1LqnRajUeDdna)函数，故而对每一个区域而言，起始点必须添加两次，第二次作为结束点以生成封闭图形。
2. 这次制作数据集，直接将三类也添加进数据集中。

2019/4/24

代码又能运行了

中间出现过下标不能为bool值的错误，代码如下：

\_idx = np.sum(mask, axis=(0, 1)) > 0

mask = mask[:, :, \_idx]

class\_ids = class\_ids[\_idx]

后来发现是因为class\_ids我传的是list，而array是可以进行这种取值操作的。

修改完成之后就开始训练了。

训练时发现速度比较慢，打开任务管理器发现使用的是CPU进行训练，花了一些时间进行重装环境。

**经验总结：**

## 数据入口：

在train文件中自定义dataset类，在基类里共有image\_id，image\_info,class\_info,source\_class\_id四个内容，随后使用add\_image函数添加图像信息，同时可以添加其他的字典型附加信息，add\_class添加类别信息。在自定义的数据集类中添加load\_datasetname()函数用以txt，json等文件中添加上述两个内容，随后，定义load\_mask（）（必须同名）返回mask和对应的class\_id。在data\_generator()函数中使用load\_image\_gt()函数返回对应的所需内容：

image, image\_meta, gt\_class\_ids, gt\_boxes, gt\_masks = \

load\_image\_gt(dataset, config, image\_id, augment=augment,

augmentation=augmentation,

use\_mini\_mask=config.USE\_MINI\_MASK)

keras可以使用generator来进行数据的读取，标志语句：

yield inputs, outputs

**以上即为基本数据流程。基本上整个流程以np.array作为数据载体。**

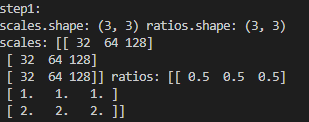
## anchor生成

在util的generate\_anchor（）函数中，使用了meshgrid函数，该函数创建点矩阵，返回x和y轴的所有坐标点的矩阵，例子：

scales = [32, 64, 128]

ratios = [0.5, 1, 2]

scales, ratios = np.meshgrid(np.array(scales), np.array(ratios))

结果：

假设点阵为（2\*3），那么最终函数的输出为四个数组，分别为



**如上图所示的四个数组。其维度是2\*3\*len(scales)\*len(ratios)，左边代表了每个anchor的宽或者高，右边代表了每个anchor中心点的横或者纵坐标。**