



OpenR8

Image-Segmentation-MaskRCNN-Keras

版本 20190808

LeaderG

立达软件科技



目录

一、	Image-Segmentation-MaskRCNN-Keras 介绍	4
二、	Image-Segmentation-MaskRCNN-Keras 文件夹介绍	5
三、	准备训练样本图 + 标记类别	7
四、	运行 1_train.py 开始训练	13
五、	运行 2_inference.py 看训练结果.....	16
六、	参数介绍.....	19

图目录

图 1.	标记、训练、测试样本的流程.....	4
图 2.	Image-Segmentation-MaskRCNN-Keras 位置	5
图 3.	标记网站接口.....	7
图 4.	按下 Add Files 新增档案	8
图 5.	选择与开启要训练的图档.....	8
图 6.	使用多边形框选药丸.....	9
图 7.	使用多边形框选药丸.....	9
图 8.	使用多边形框选药丸.....	10
图 9.	新增名称.....	10
图 10.	输入类别名称.....	11
图 11.	输出标记 json 文件	12
图 12.	将输出标记 json 文件放到 data\pill\train 文件夹	12
图 13.	开启 OpenR8 程序	13
图 14.	选择 1_train.py	14
图 15.	开启 1_train.py	14
图 16.	设置 dataset_Path	14
图 17.	选择 2_inference.py.....	16
图 18.	开启 2_inference.py.....	16
图 19.	填要测试的样本路径.....	17
图 20.	填入要训练的 h5 样本路径.....	17
图 21.	2_inference.py 的测试结果.....	18
图 22.	在 surgery.py 中更改读取 h5 檔檔名	19
图 23.	在 “data\predefined_classes.txt” 中设置类别名称.....	19
图 24.	在 surgery.py 中更改读取类别 json 名称	20
图 25.	在 surgery.py 中增加 GPU 数量设置.....	20

表目录

表 1. Image-Segmentation-MaskRCNN-Keras 文件夹介绍	6
--	---

一、Image-Segmentation-MaskRCNN-Keras 介绍

Mask R-CNN 为 Faster R-CNN 的延伸应用，比 Faster R-CNN 多增加一个分支，在检测目标物的同时，将目标像素分割出来。

此解决方案使用 Mask R-CNN 来判断药丸的种类与位置，标记时，目标物使用多边形描绘出对象轮廓，标上其类别。

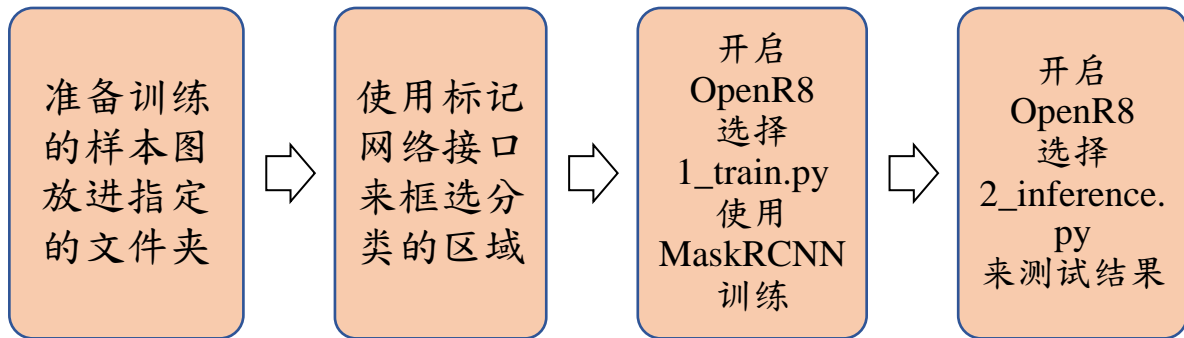


图1. 标记、训练、测试样本的流程

二、Image-Segmentation-MaskRCNN-Keras 文件夹介绍

Image-Segmentation-MaskRCNN-Keras 位于 OpenR8 的 solution 文件夹内，其中包含：

1. 文件夹：【data 文件夹】、【src 文件夹】、【tool 文件夹】。
2. py 档案：【1_train.py】、【2_inference.py】。

※初次使用者，建议先只改动 data 内 pill 文件夹的档案内容，等熟悉后，再自行更动至想要的位置。

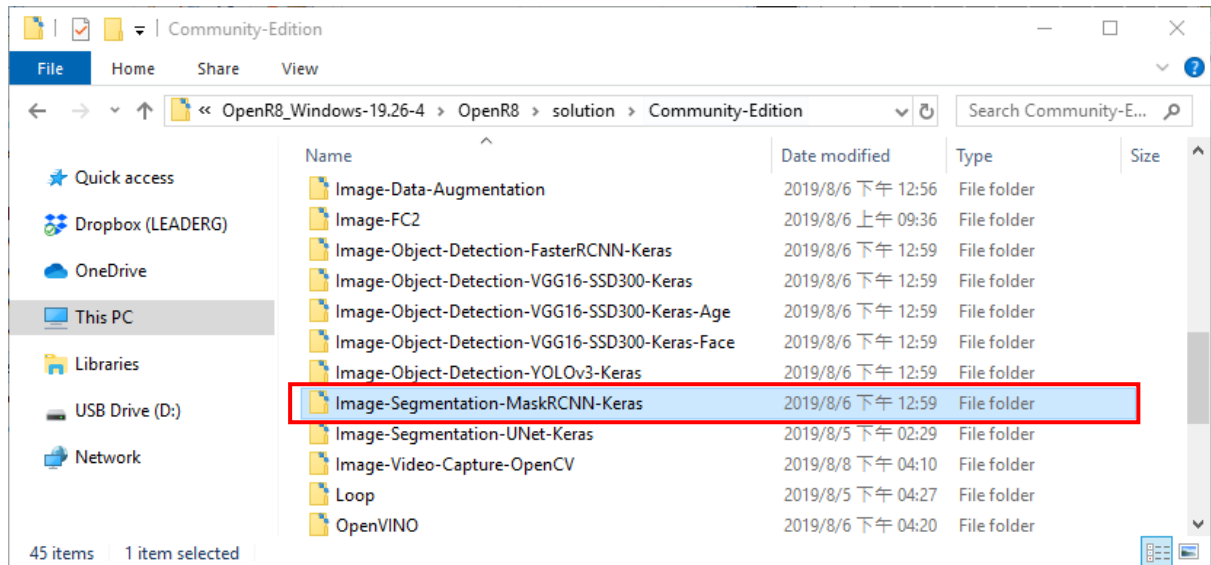


图2. Image-Segmentation-MaskRCNN-Keras 位置

名称	用途与功能	内容
data 文件夹	存放训练样本图、类别；测试图、类别；model 檔、存放训练完成后的 model 档案。。	<p>pill\test、</p> <p>pill\train、</p> <p>pill\val、</p> <p>mask_rcnn_coco.h5(用来当样本 model 训练(第一次使用请勿删除))</p> <p>mask20181217T0933\</p> <p>events.out.tfevents.1545010450.S3</p> <p>mask20181217T0933\</p> <p>mask_rcnn_mask_0020.h5(训练好的 model 档案)</p>

src 文件夹	训练与测试时会用到的 python 檔，其中 surgery.py 主要为训练与测试用。	real_time_detection.py split_dataset.py setup.py surgery.py
tool 文件夹	标记图文件所用的网页。	via-2.0.2\via.html
1_train.py	训练样本的解决方案。	
2_inference.py	测试样本的解决方案。	

表 1. Image-Segmentation-MaskRCNN-Keras 文件夹介绍

三、准备训练样本图 + 标记类别

我们要训练之前时，要先决定好方向，以此文件为例，我们想检测药丸种类与所在的位置，所以我们将样本图片一一标示它们的类别(药丸)。

第一步：开启标记网站接口

开启 tool 文件夹内 via-2.0.2 文件夹中 via.html 网页来标记我们想训练的样本类别。

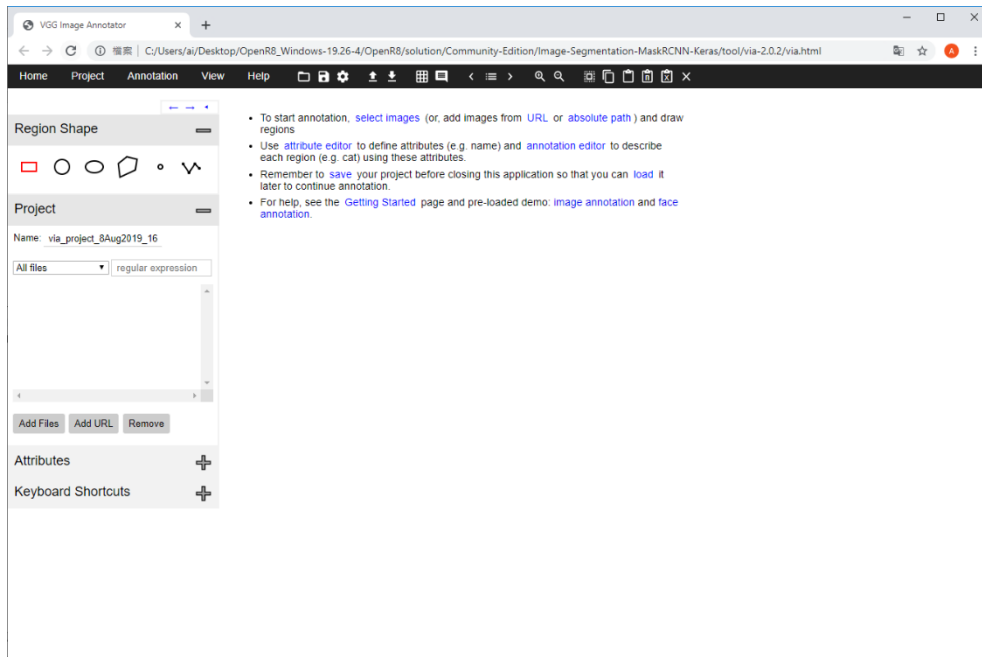


图3. 标记网站接口

第二步：选择样本图片存放文件夹

点选 **Open Dir** 来开启图片样本所放的文件夹位置，以这里的解决方案为例，要训练图片放在 data\pill\train，于是按下“Add Files”来准备标记图片，如图 4、图 5。

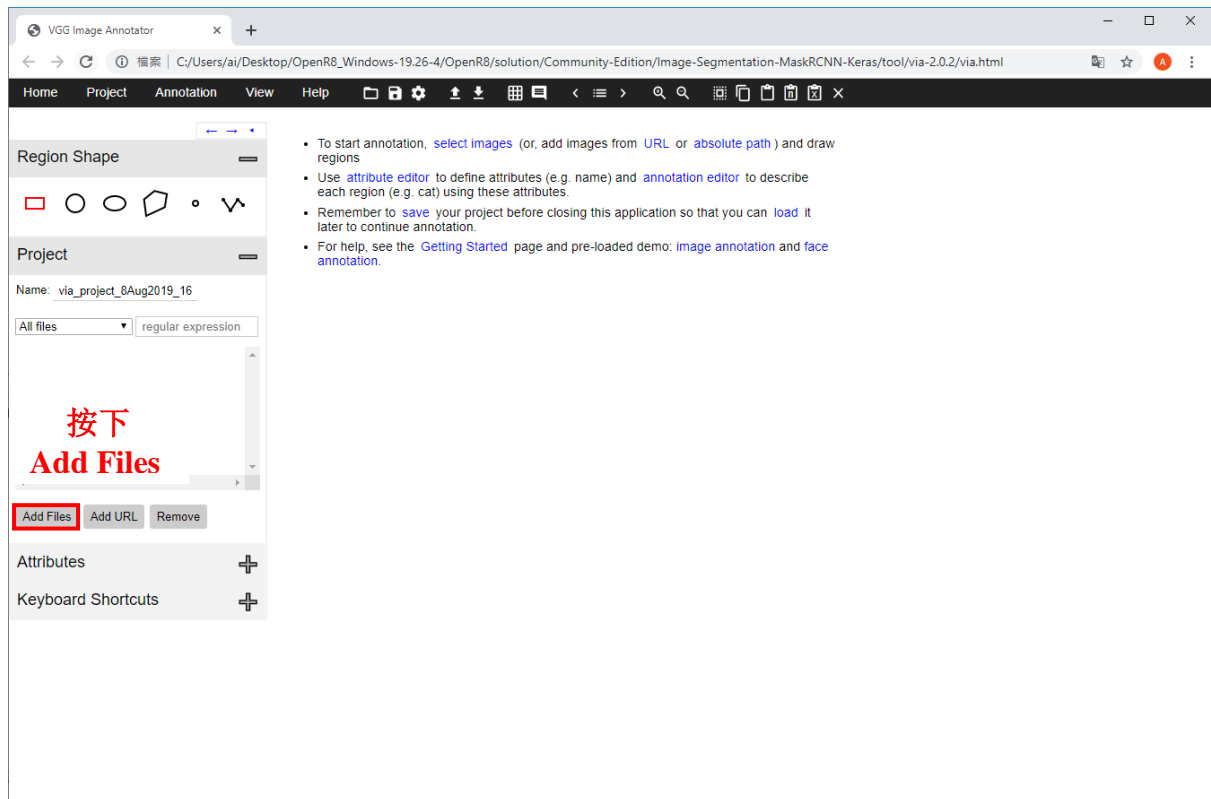


图4. 按下 Add Files 新增档案

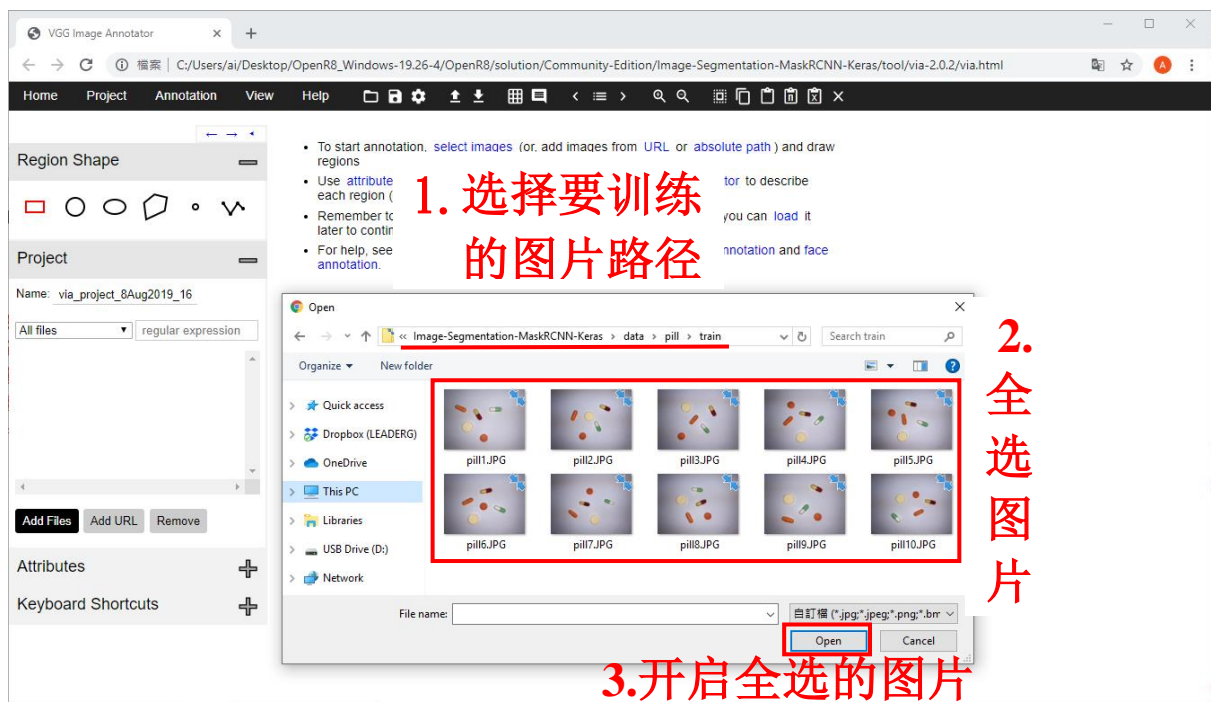


图5. 选择与开启要训练的图档

第三步：框选类别

使用多边形来描绘想辨识的区域。



图6. 使用多边形框选药丸

如果使用多边形框完后，双击左键，即可结束多边形框选，如图7、图8。



图7. 使用多边形框选药丸

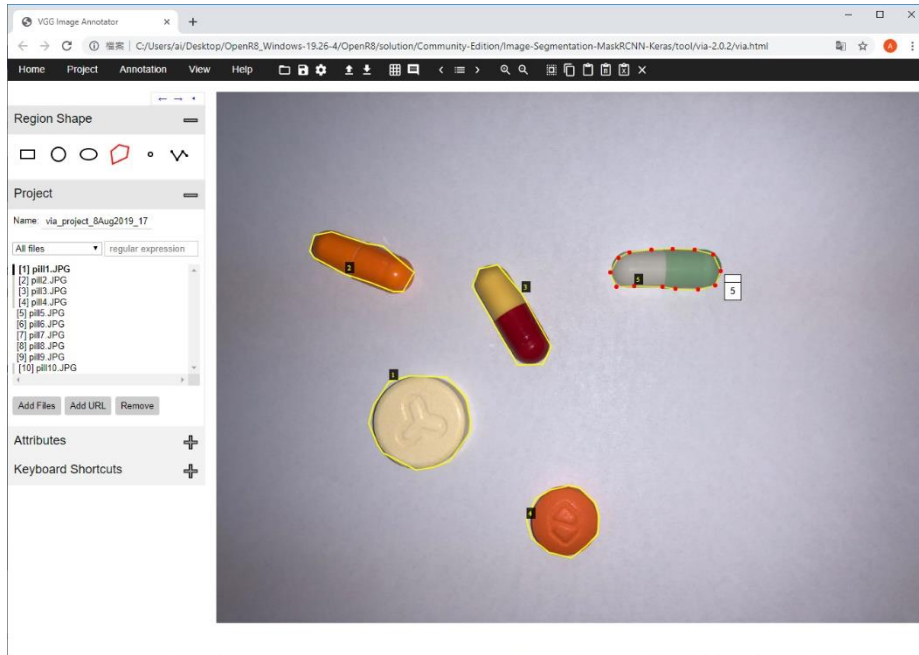


图8. 使用多边形框选药丸

第四步：框选样本图片并标记类别

如图 9，在“Attributes 的 attribute name”字段输入 name，接着如图 10，根据框的编号填入该类别名称，以本文件为例，是判断药丸，由于有不同种类药丸，于是填“pill1”、“pill2”、“pill3”等等，填完即可按 X 关闭。

继续框选下一张样本图片，直到所有样本图片皆标记好类别为止。



图9. 新增名称



图10. 输入类别名称

第五步：输出标记类别文件。

在全部标记完毕后，要输出标记的档案，如图 11，按下在上方 Annotation 中的 Export Annotations (as json) 输出标记所有图片类别的 json 文件。

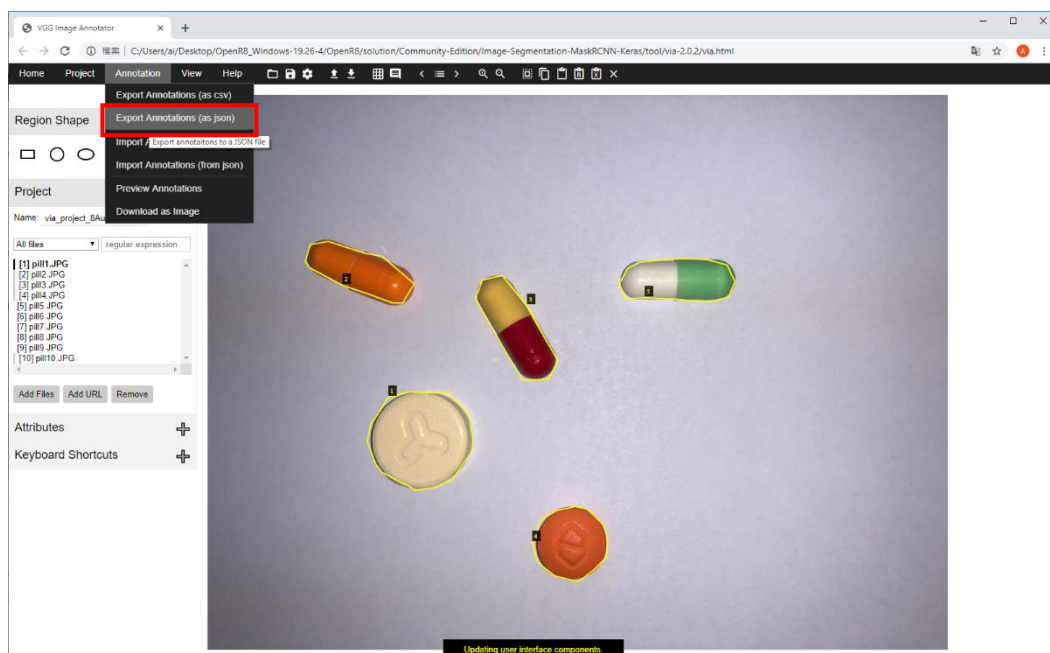


图 11. 输出标记 json 文件

第六步：将输出 json 檔放到 data\pill\train 文件夹内

将刚刚输出的 json 檔 放到 data\pill\train 文件夹内，并确认档名是否为

“via_region_data.json”，如果不是，请改名成 “via_region_data.json”，如图 12。

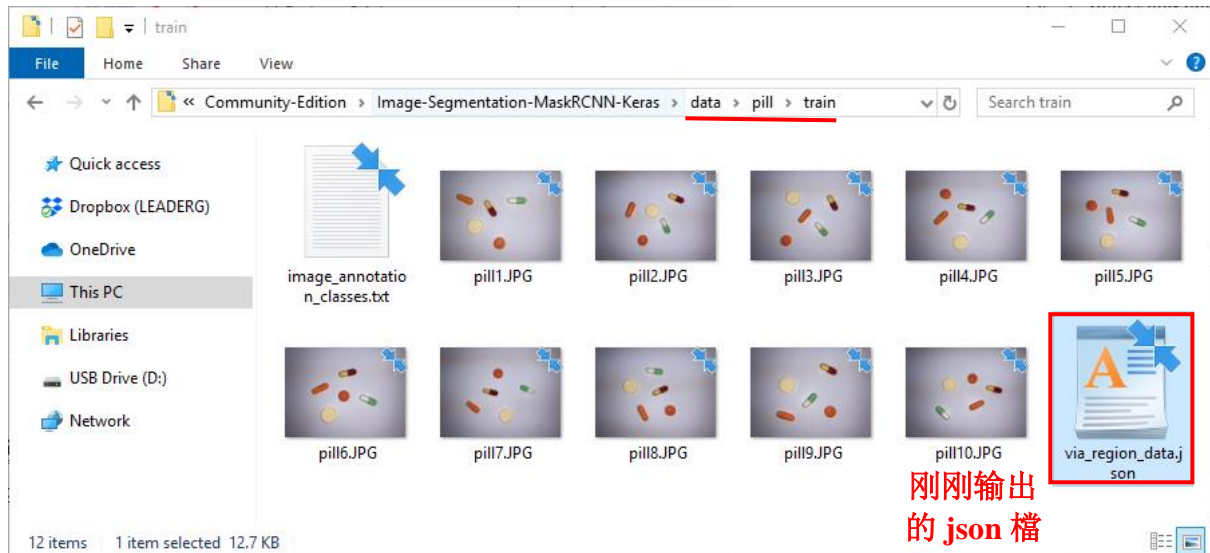


图 12. 将输出标记 json 文件放到 data\pill\train 文件夹

※ 要测试的样本 data\pill\val 文件夹内，一样要做第一步到第六步。

四、运行 1_train.py 开始训练

一开始请开启【OpenR8 程序】，如果计算机有安装显示适配器，请点选【R8_Python3.6_GPU.bat】运行档，没有则点选【R8_Python3.6_CPU.bat】运行档，如图 13。开启完【OpenR8 程序】后，请点选【档案】=>【开启】=>【进入到 OpenR8 底下的 solution 文件夹】=>【选择 Image-Segmentation-MaskRCNN-Keras 文件夹】=>【选择 1_train.py 开启】，如图 14、图 15。

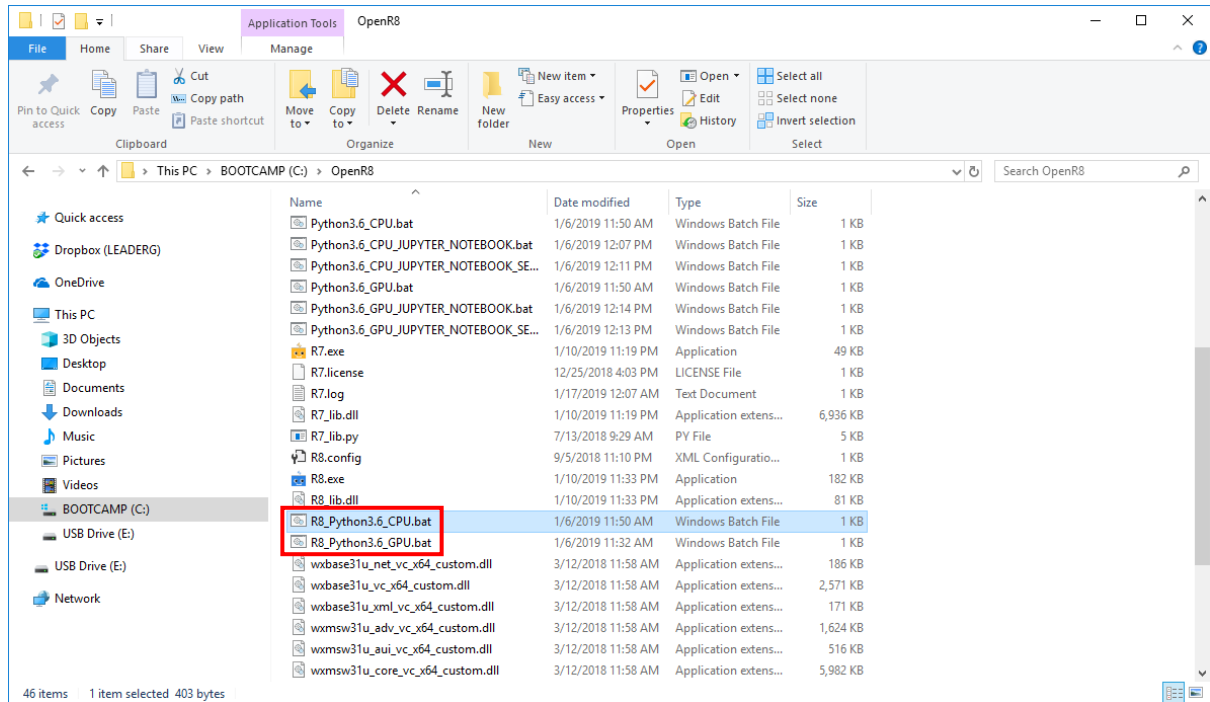


图 13. 开启 OpenR8 程序

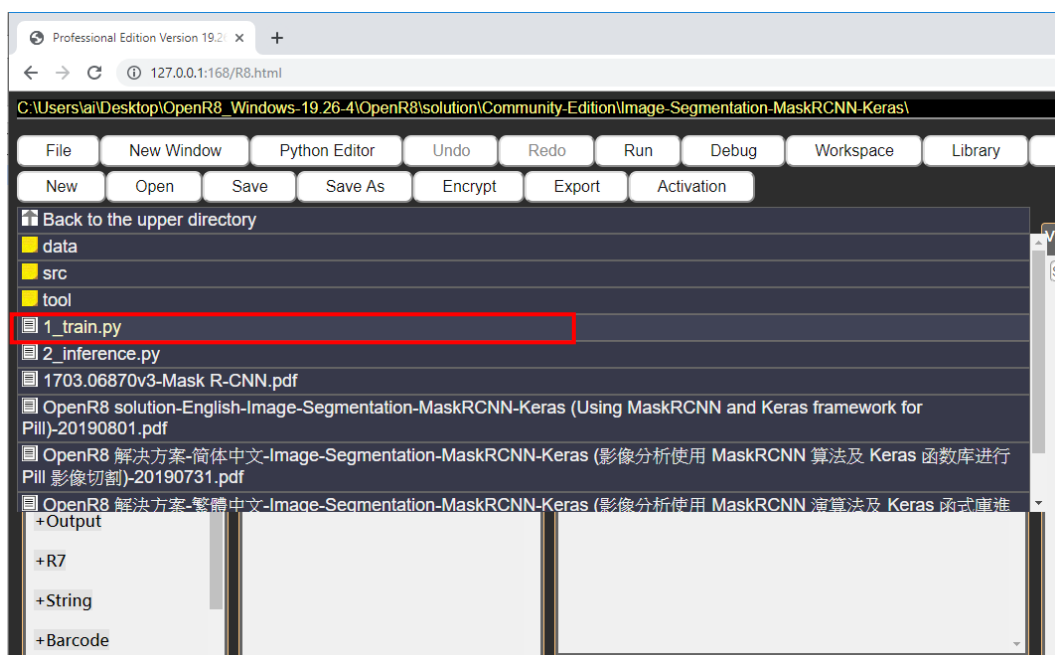


图 14. 选择 1_train.py

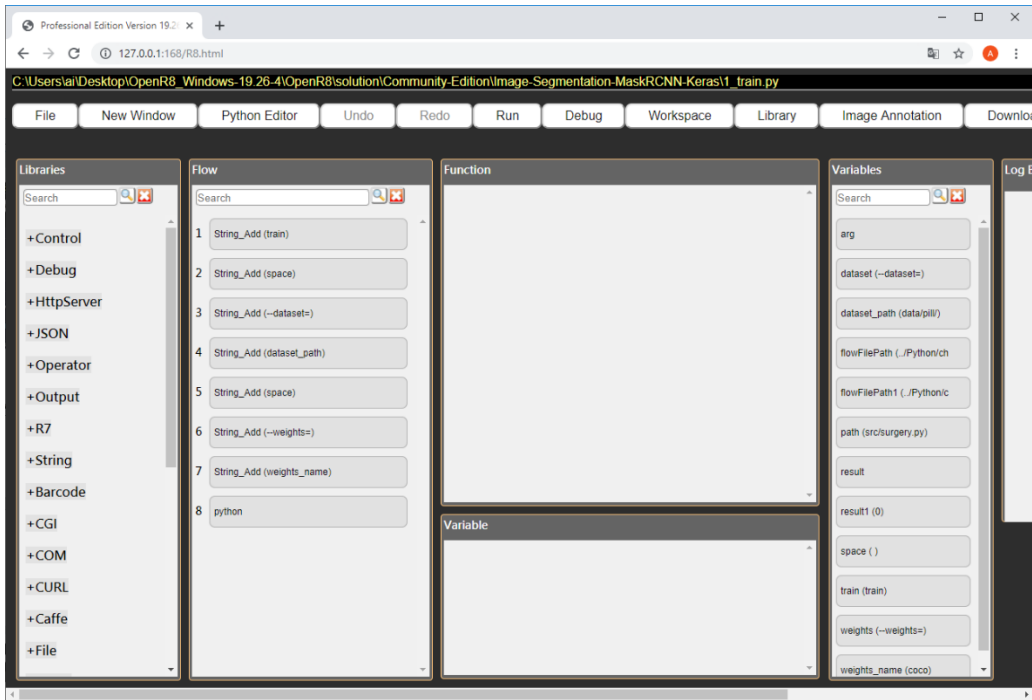


图 15. 开启 1_train.py

※如果样本图没有放在“data\pill”里面的话，需额外设置 dataset_Path，如图 16。

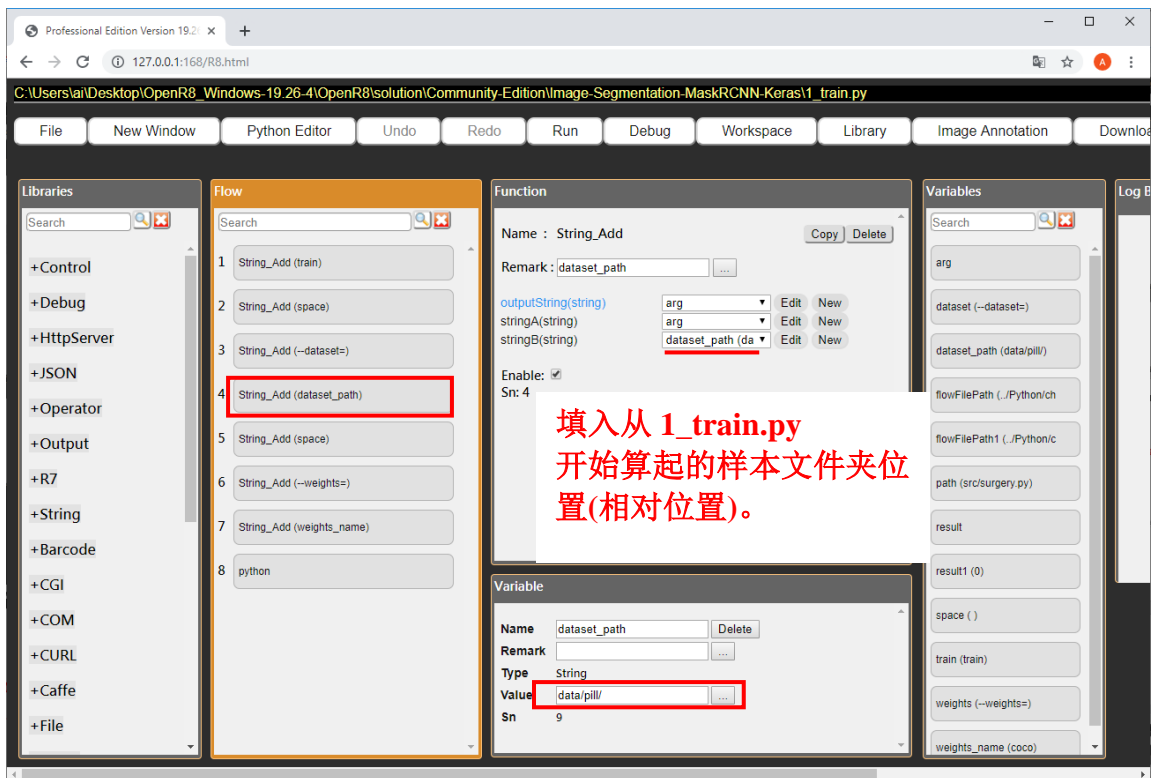


图 16. 设置 dataset_Path

※在运行前，如果没有要沿用之前的 **model**，请删除所有 **h5** 档案(但保留 **mask_rcnn_coco.h5**)，不熟悉者建议都不删除。

※在运行前，如果想改变“训练模型名称”、“训练次数”、“分类类别”等参数设置，请看第六章 — 参数介绍。

按下运行开始训练样本，直到跳出「Press any key to continue...」。

五、运行 2_inference.py 看训练结果

在运行完 1_train.py 训练结束后，开启 2_inference.py 来测试图片，如图 17、图 18。

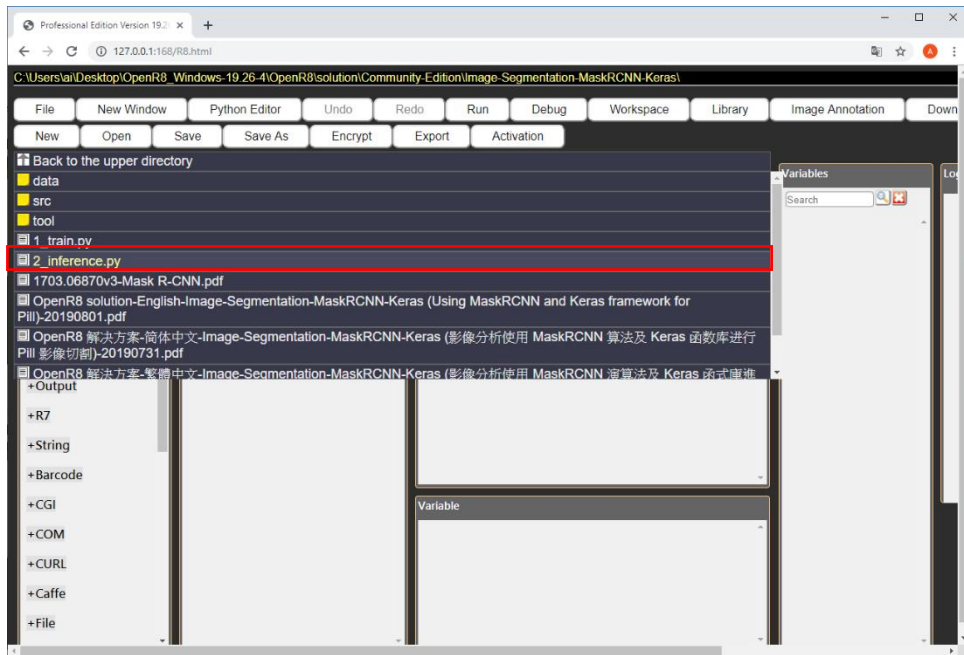


图 17. 选择 2_inference.py

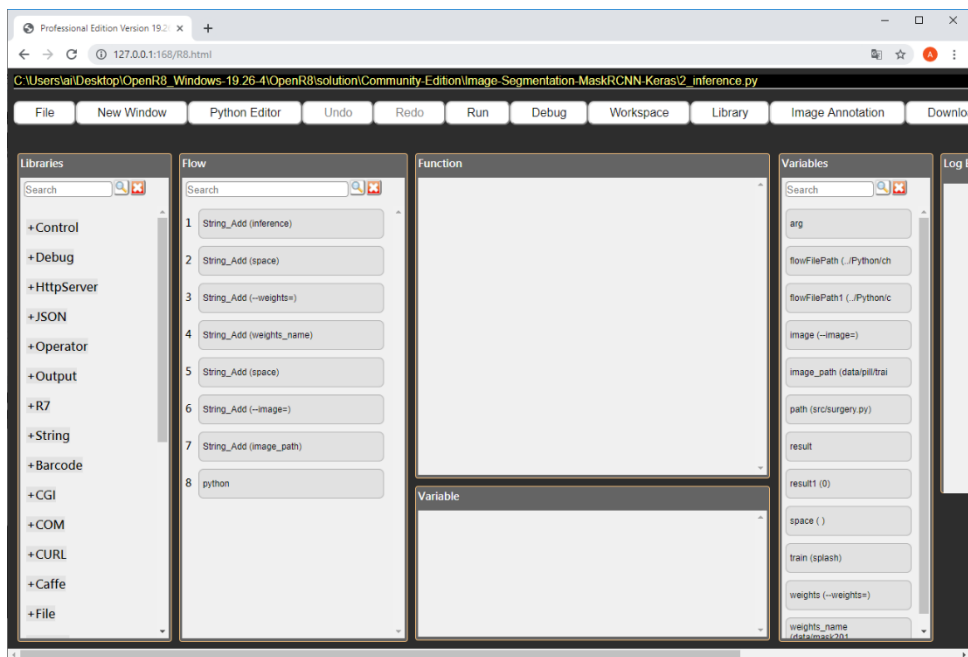


图 18. 开启 2_inference.py

填入要测试的样本路径与训练完的 h5 文件路径，如图 19、图 20。

※如果有运行过 1_train.py 且成功训练出 model 者，**务必**确认图 20 的 h5 档名称是否一致。

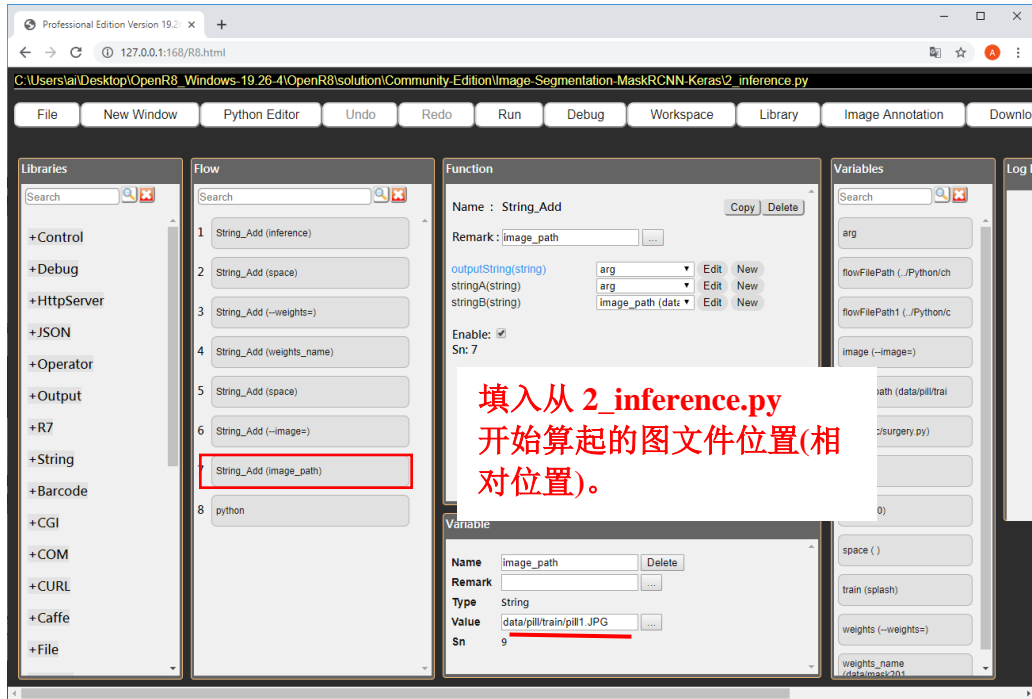


图19. 填入要测试的样本路径

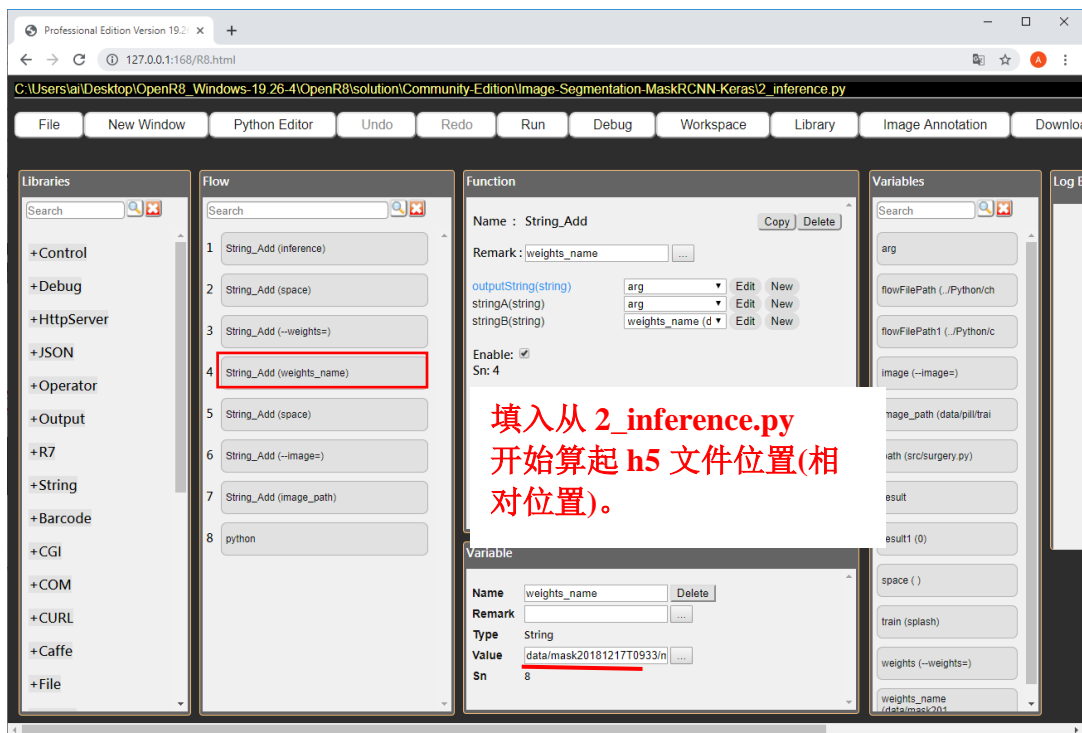


图20. 填入要训练的 h5 样本路径

按下运行看结果，Mask_R_CNN 和其他显示结果的方式不太一样，如果有判断到类别时，那个区域会标记成一种颜色并框起来显示类别及相似度，反之，如果甚么都没抓到就会没有标记颜色，如图 21，在药丸的位置分别标记成不同颜色，代表有被抓出。

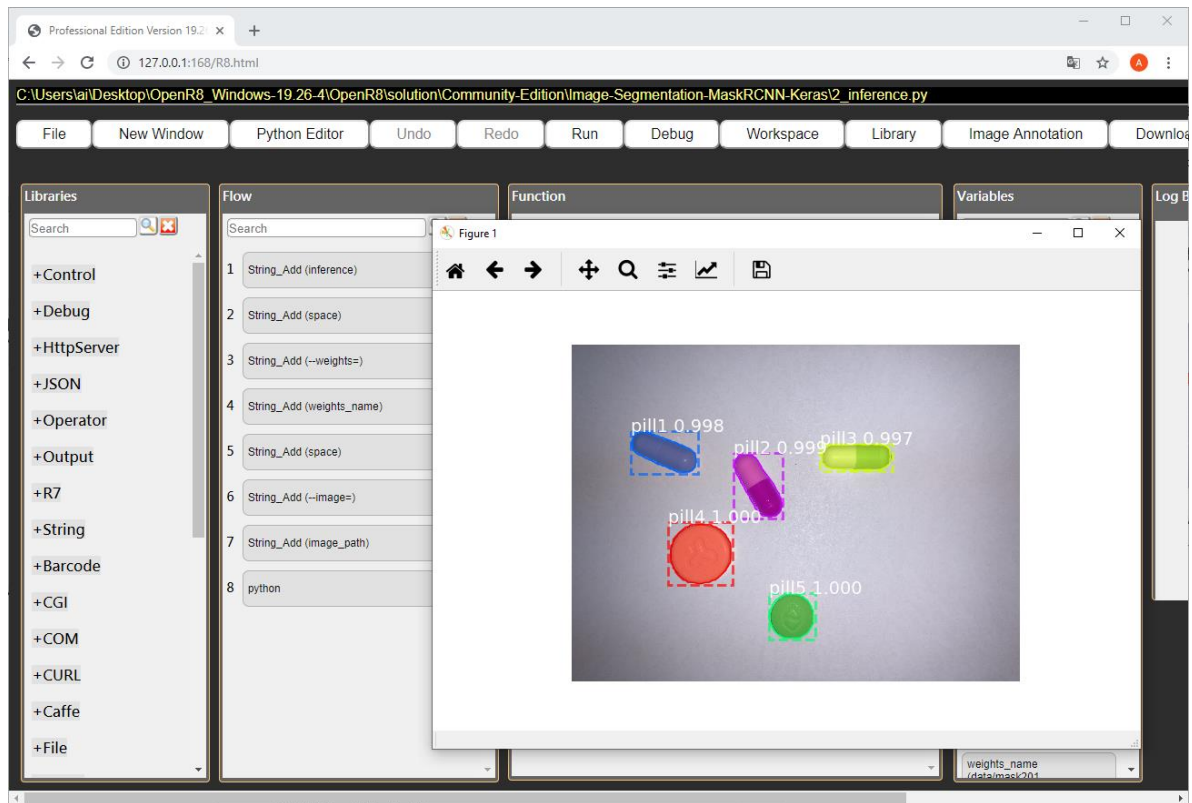


图21. 2_inference.py 的测试结果

六、参数介绍

- ※ 更改读取 h5 档檔名：图 22。
- ※ 在 “data\predefined_classes.txt” 设置类别名称：图 23。
- ※ 更改 json 名称：图 24。
- ※ 设置 GPU 数量：图 25。

```

31
32 import os
33 import sys
34 import json
35 import datetime
36 import numpy as np
37 import codecs
38 import skimage.draw
39 from matplotlib import pyplot as plt
40 # Root directory of the project
41 #ROOT_DIR = os.path.abspath("../..")
42 ROOT_DIR = os.getcwd()
43 print("ROOT_DIR = " + str(ROOT_DIR))
44
45 # Import Mask RCNN
46 sys.path.append(ROOT_DIR) # To find local version of the library
47 from mrcnn.config import Config
48 from mrcnn import model as modellib, utils
49 from mrcnn import visualize
50 # Path to trained weights file
51 COCO_WEIGHTS_PATH = os.path.join(ROOT_DIR, "data/mask_rcnn_coco.h5")
52
53 # Directory to save logs and model checkpoints, if not provided
54 # through the command line argument --logs
55 DEFAULT_LOGS_DIR = os.path.join(ROOT_DIR, "data")
56
57 # Path to predefined classes file
58 PREDEFINED_CLASSES_PATH = os.path.join(ROOT_DIR, "data/predefined_classes.txt")
59
60 # predefined classes

```

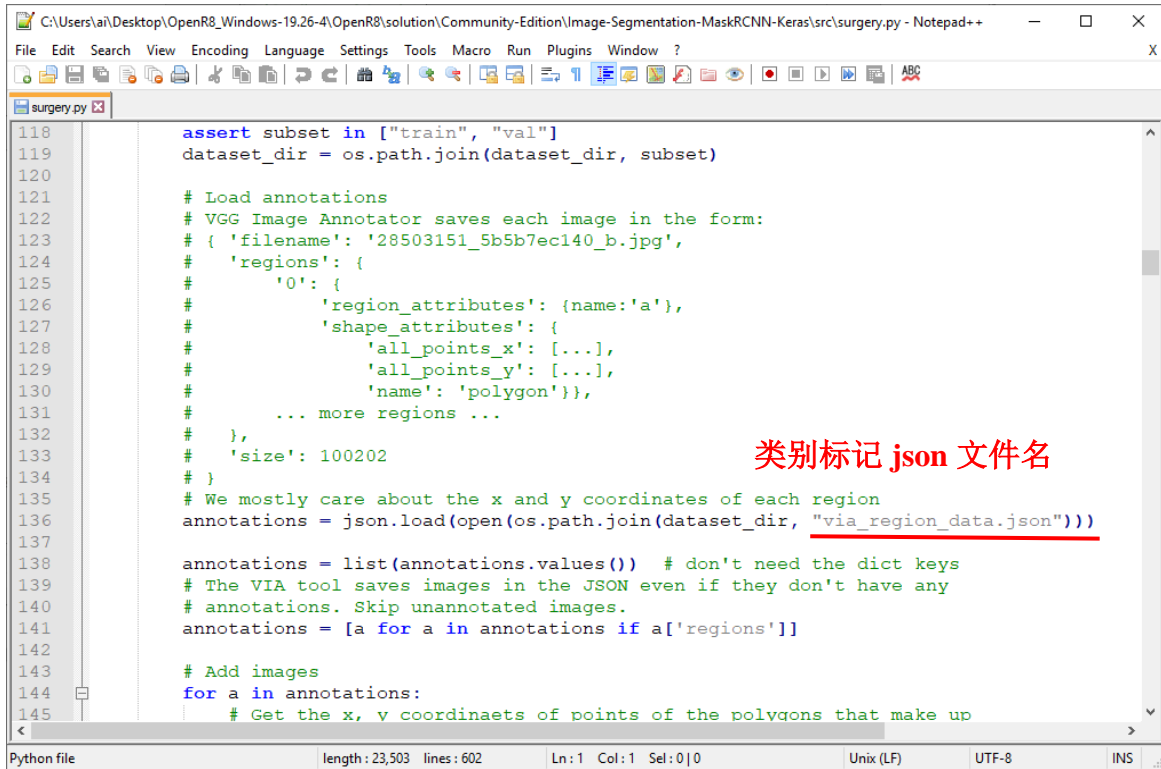
图22. 在 surgery.py 中更改读取 h5 档檔名

```

1 background
2 pill1
3 pill2
4 pill3
5 pill4
6 pill5

```

图23. 在 “data\predefined_classes.txt” 中设置类别名称

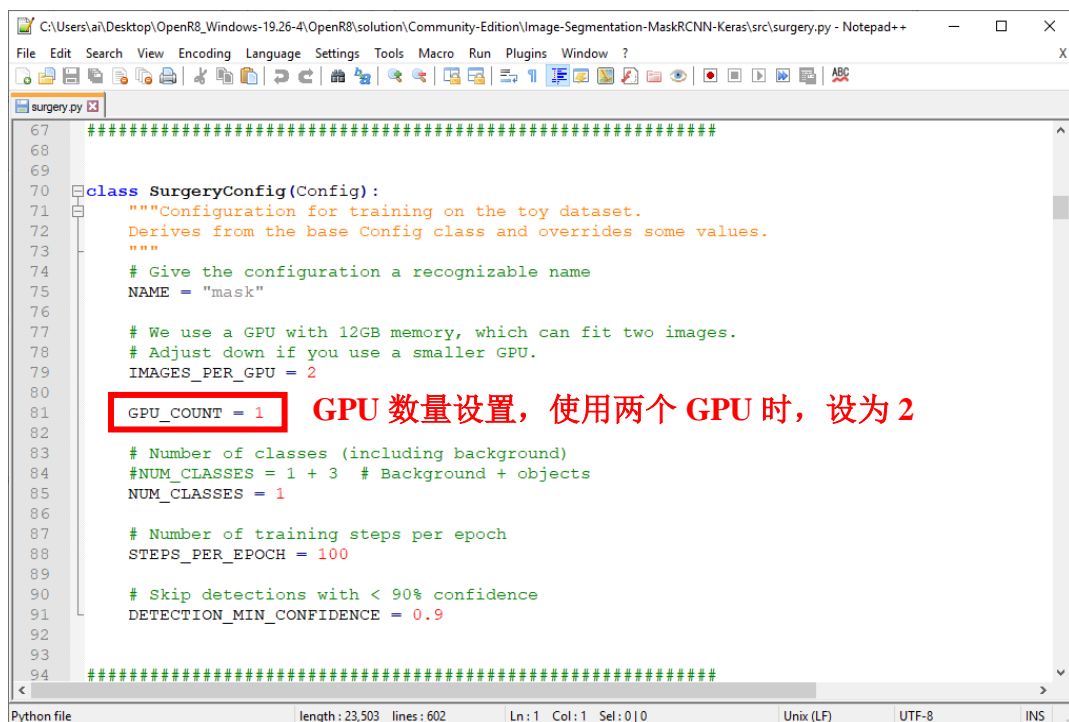


```

118 assert subset in ["train", "val"]
119 dataset_dir = os.path.join(dataset_dir, subset)
120
121 # Load annotations
122 # VGG Image Annotator saves each image in the form:
123 # { 'filename': '28503151_5b5b7ec140_b.jpg',
124 #   'regions': {
125 #     '0': {
126 #       'region_attributes': {name:'a'},
127 #       'shape_attributes': {
128 #         'all_points_x': [...],
129 #         'all_points_y': [...],
130 #         'name': 'polygon'}},
131 #     ... more regions ...
132 #   },
133 #   'size': 100202
134 # }
135 # We mostly care about the x and y coordinates of each region
136 annotations = json.load(open(os.path.join(dataset_dir, via_region_data.json)))
137
138 annotations = list(annotations.values()) # don't need the dict keys
139 # The VIA tool saves images in the JSON even if they don't have any
140 # annotations. Skip unannotated images.
141 annotations = [a for a in annotations if a['regions']]
142
143 # Add images
144 for a in annotations:
145     # Get the x, y coordinaets of points of the polygons that make up

```

图24. 在 surgery.py 中更改读取类别 json 名称



```

67 #####
68
69
70 class SurgeryConfig(Config):
71     """Configuration for training on the toy dataset.
72     Derives from the base Config class and overrides some values.
73     """
74     # Give the configuration a recognizable name
75     NAME = "mask"
76
77     # We use a GPU with 12GB memory, which can fit two images.
78     # Adjust down if you use a smaller GPU.
79     IMAGES_PER_GPU = 2
80
81     GPU_COUNT = 1
82
83     # Number of classes (including background)
84     # NUM_CLASSES = 1 + 3 # Background + objects
85     NUM_CLASSES = 1
86
87     # Number of training steps per epoch
88     STEPS_PER_EPOCH = 100
89
90     # Skip detections with < 90% confidence
91     DETECTION_MIN_CONFIDENCE = 0.9
92
93
94 #####

```

图25. 在 surgery.py 中增加 GPU 数量设置