



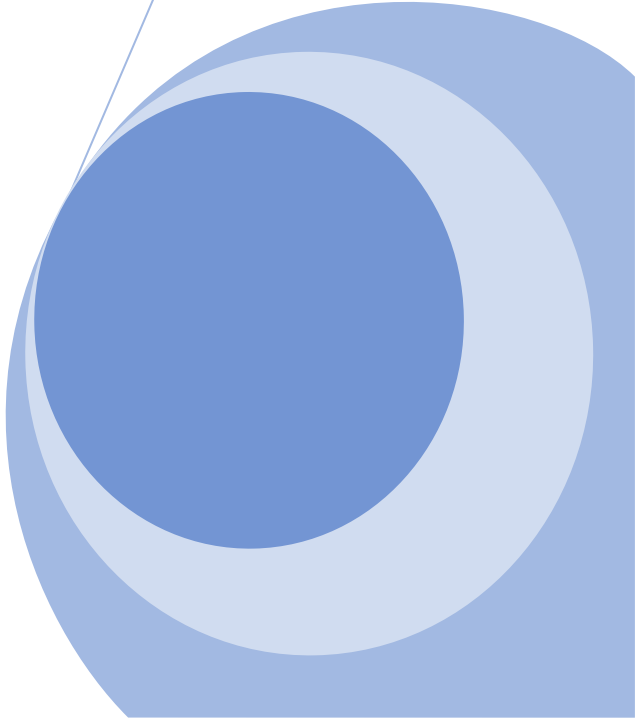
OpenR8

Image-Segmentation-MaskRCNN-Keras

版本 20190808

LeaderG

立達軟體科技



目錄

一、	Image-Segmentation-MaskRCNN-Keras 介紹	4
二、	Image-Segmentation-MaskRCNN-Keras 資料夾介紹	5
三、	準備訓練樣本圖 + 標記類別	7
四、	執行 1_train.py 開始訓練	13
五、	執行 2_inference.py 看訓練結果.....	16
六、	參數介紹.....	19

圖目錄

圖 1. 標記、訓練、測試樣本的流程.....	4
圖 2. Image-Segmentation-MaskRCNN-Keras 位置	5
圖 3. 標記網站介面.....	7
圖 4. 按下 Add Files 新增檔案	8
圖 5. 選擇與開啟要訓練的圖檔.....	8
圖 6. 使用多邊形框選藥丸.....	9
圖 7. 使用多邊形框選藥丸.....	9
圖 8. 使用多邊形框選藥丸.....	10
圖 9. 新增名稱.....	10
圖 10. 輸入類別名稱.....	11
圖 11. 輸出標記 json 檔	12
圖 12. 將輸出標記 json 檔放到 data\pill\train 資料夾	12
圖 13. 開啟 OpenR8 程式	13
圖 14. 選擇 1_train.py	14
圖 15. 開啟 1_train.py	14
圖 16. 設定 dataset_Path	14
圖 17. 選擇 2_inference.py.....	16
圖 18. 開啟 2_inference.py.....	16
圖 19. 填要測試的樣本路徑.....	17
圖 20. 填入要訓練的 h5 樣本路徑.....	17
圖 21. 2_inference.py 的測試結果.....	18
圖 22. 在 surgery.py 中更改讀取 h5 檔檔名	19
圖 23. 在 “data\predefined_classes.txt” 中設定類別名稱.....	19
圖 24. 在 surgery.py 中更改讀取類別 json 名稱	20
圖 25. 在 surgery.py 中增加 GPU 數量設定	20

表目錄

表 1. Image-Segmentation-MaskRCNN-Keras 資料夾介紹	6
--	---

一、Image-Segmentation-MaskRCNN-Keras 介紹

Mask R-CNN 為 Faster R-CNN 的延伸應用，比 Faster R-CNN 多增加一個分支，在檢測目標物的同時，將目標像素分割出來。

此解決方案使用 Mask R-CNN 來判斷藥丸的種類與位置，標記時，目標物使用多邊形描繪出物件輪廓，標上其類別。

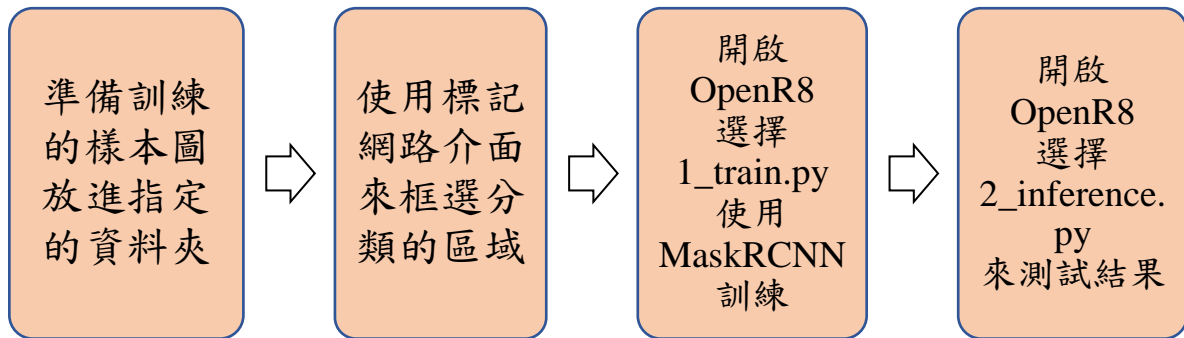


圖1. 標記、訓練、測試樣本的流程

二、Image-Segmentation-MaskRCNN-Keras 資料夾介紹

Image-Segmentation-MaskRCNN-Keras 位於 OpenR8 的 solution 資料夾內，其中包含：

1. 資料夾：【data 資料夾】、【src 資料夾】、【tool 資料夾】。
2. py 檔案：【1_train.py】、【2_inference.py】。

※初次使用者，建議先只改動 data 內 pill 資料夾的檔案內容，等熟悉後，再自行更動至想要的位置。

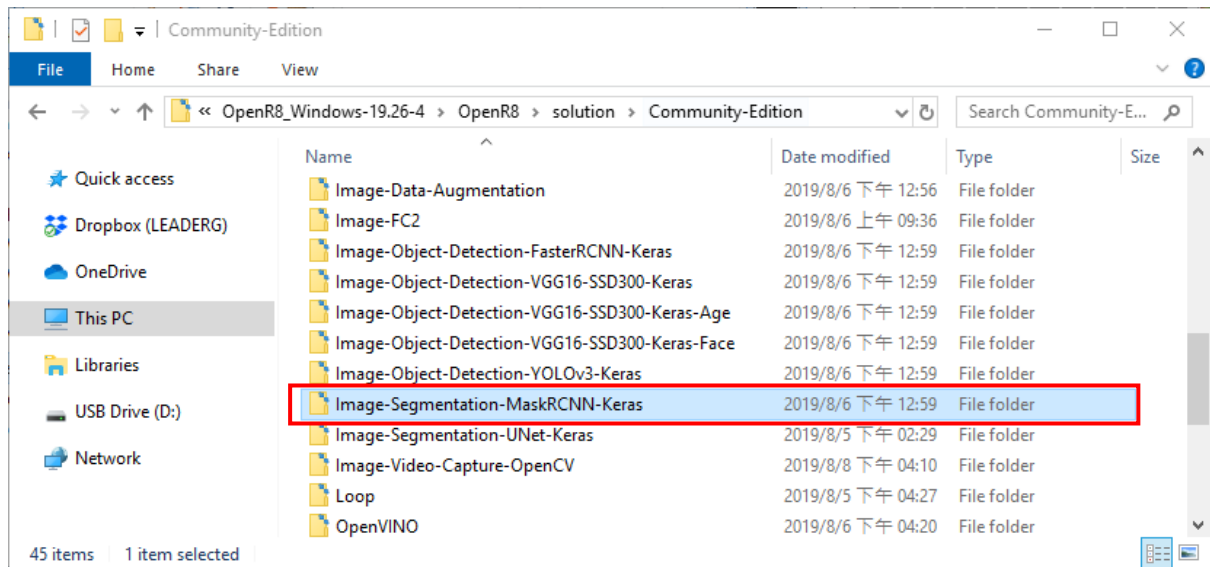


圖2. Image-Segmentation-MaskRCNN-Keras 位置

名稱	用途與功能	內容
data 資料夾	存放訓練樣本圖、類別；測試圖、類別；model 檔、存放訓練完成後的 model 檔案。	<p>pill\test、</p> <p>pill\train、</p> <p>pill\val、</p> <p>mask_rcnn_coco.h5(用來當樣本 model 訓練(第一次使用請勿刪除))</p> <p>mask20181217T0933\</p> <p>events.out.tfevents.1545010450.S3</p> <p>mask20181217T0933\</p> <p>mask_rcnn_mask_0020.h5(訓練好的 model 檔案)</p>

src 資料夾	訓練與測試時會用到的 python 檔，其中 surgery.py 主要為訓練與測 試用。	real_time_detection.py split_dataset.py setup.py surgery.py
tool 資料夾	標記圖檔所用的網頁。	via-2.0.2\via.html
1_train.py	訓練樣本的解決方案。	
2_inference.py	測試樣本的解決方案。	

表 1. Image-Segmentation-MaskRCNN-Keras 資料夾介紹

三、準備訓練樣本圖 + 標記類別

我們要訓練之前時，要先決定好方向，以此文件為例，我們想檢測藥丸種類與所在的位置，所以我們將樣本圖片一一標示它們的類別(藥丸)。

第一步：開啟標記網站介面

開啟 tool 資料夾內 via-2.0.2 資料夾中 via.html 網頁來標記我們想訓練的樣本類別。

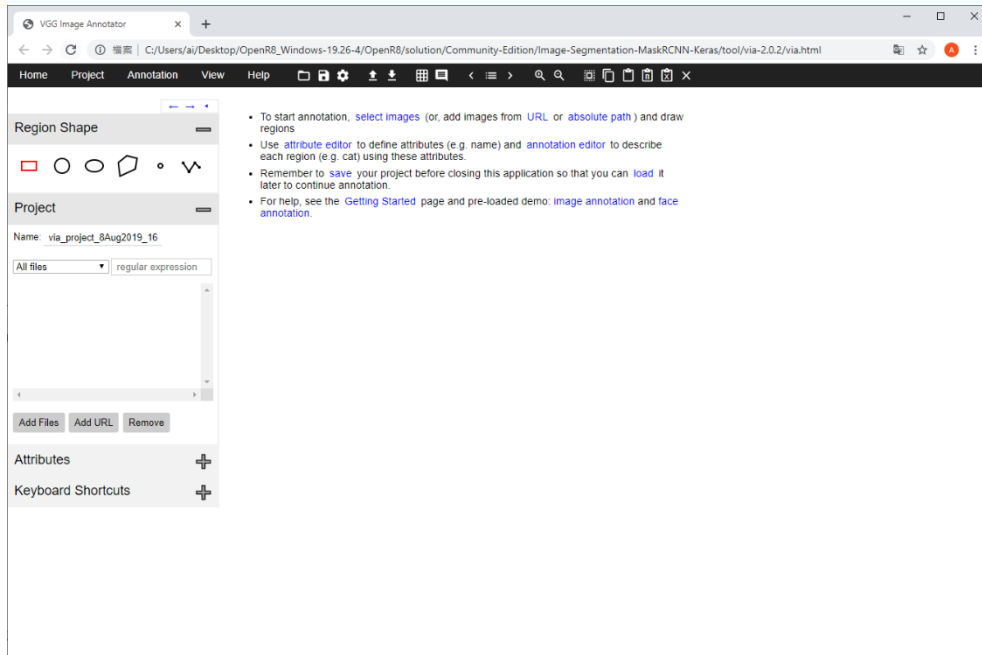


圖3. 標記網站介面

第二步：選擇樣本圖片存放資料夾

點選 Open Dir 來開啟圖片樣本所放的資料夾位置，以這裡的解決方案為例，要訓練圖片放在 data\pill\train，於是按下“Add Files”來準備標記圖片，如圖 4、圖 5。

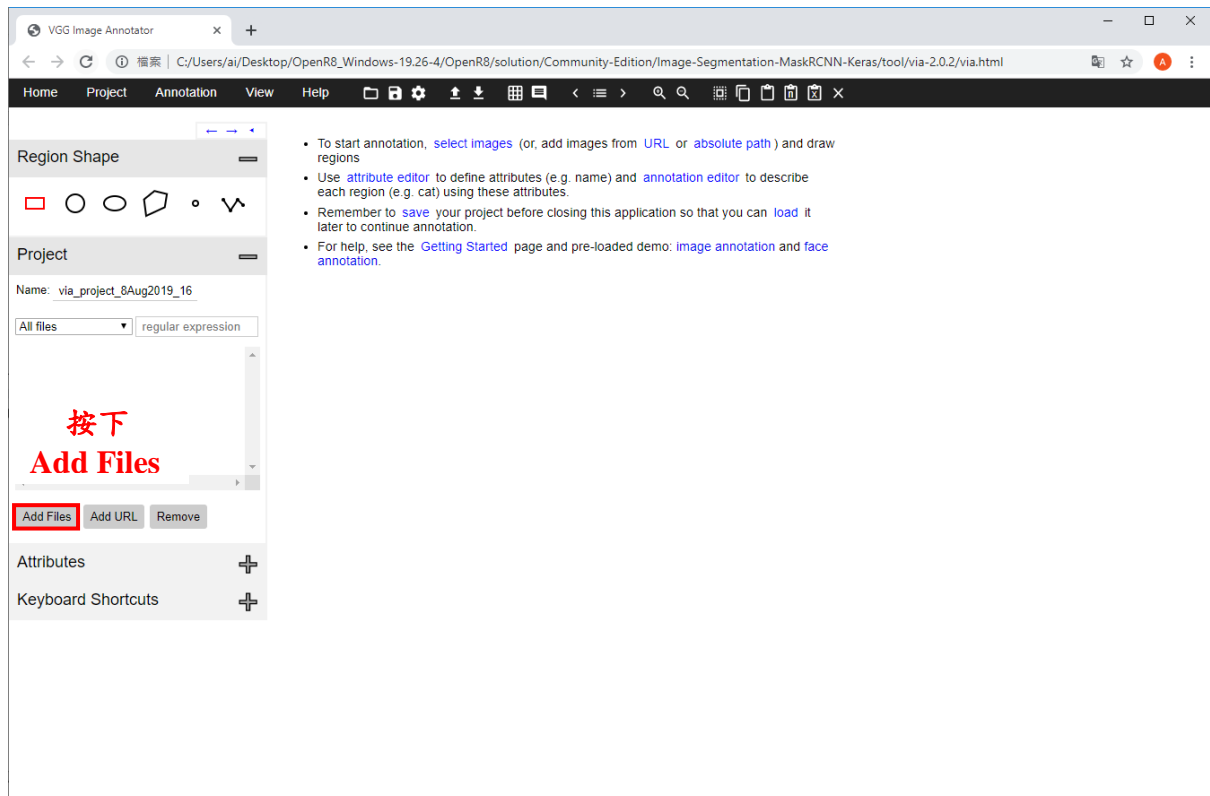


圖4. 按下 Add Files 新增檔案

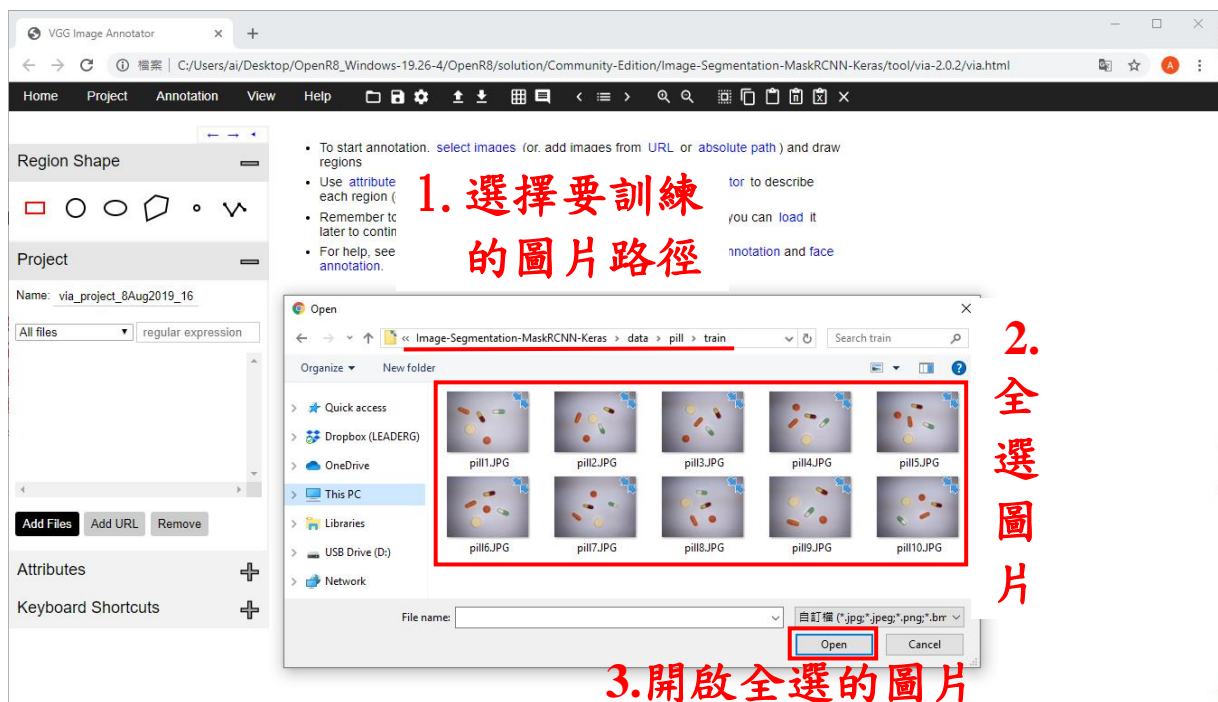


圖5. 選擇與開啟要訓練的圖檔

第三步：框選類別

使用多邊形來描繪想辨識的區域。



圖6. 使用多邊形框選藥丸

如果使用多邊形框完後，雙擊左鍵，即可結束多邊形框選，如圖 7、圖 8。



圖7. 使用多邊形框選藥丸

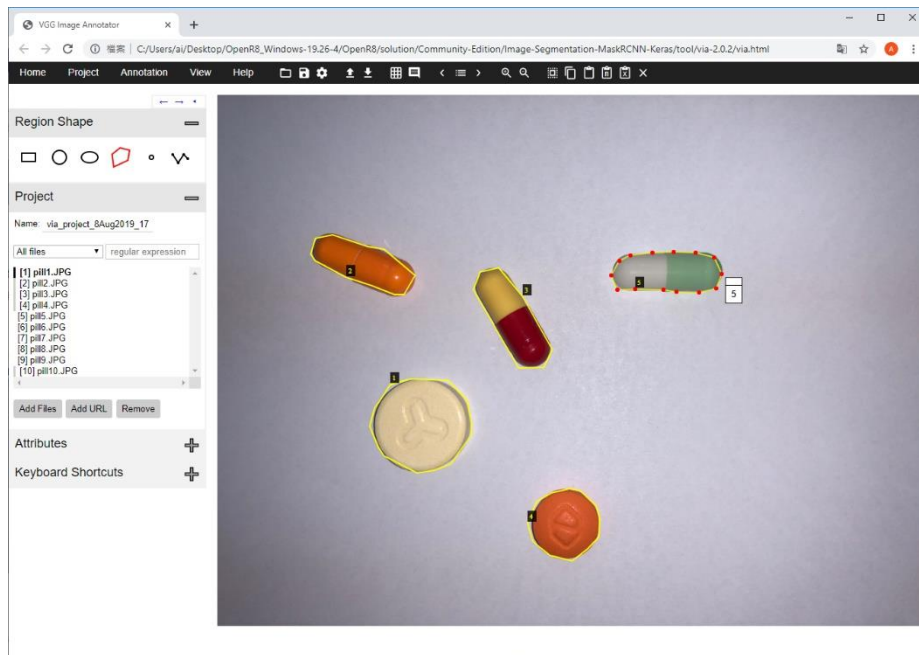


圖8. 使用多邊形框選藥丸

第四步：框選樣本圖片並標記類別

如圖 9，在“Attributes 的 attribute name”欄位輸入 name，接著如圖 10，根據框的編號填入該類別名稱，以本文件為例，是判斷藥丸，由於有不同種類藥丸，於是填“pill1”、“pill2”、“pill3”等等，填完即可按 X 關閉。

繼續框選下一張樣本圖片，直到所有樣本圖片皆標記好類別為止。



圖9. 新增名稱

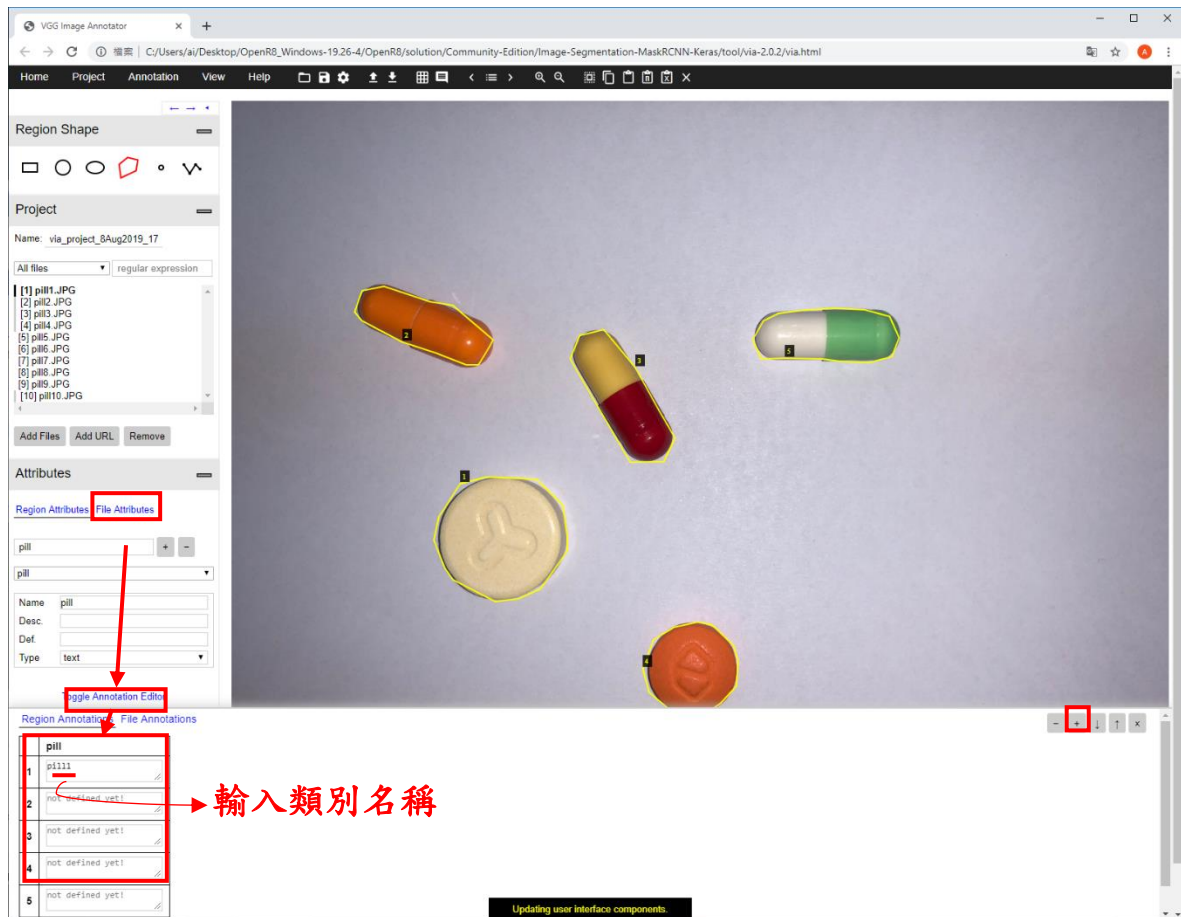


圖10. 輸入類別名稱

第五步：輸出標記類別檔。

在全部標記完畢後，要輸出標記的檔案，如圖 11，按下在上方 Annotation 中的 Export Annotations (as json) 輸出標記所有圖片類別的 json 檔。

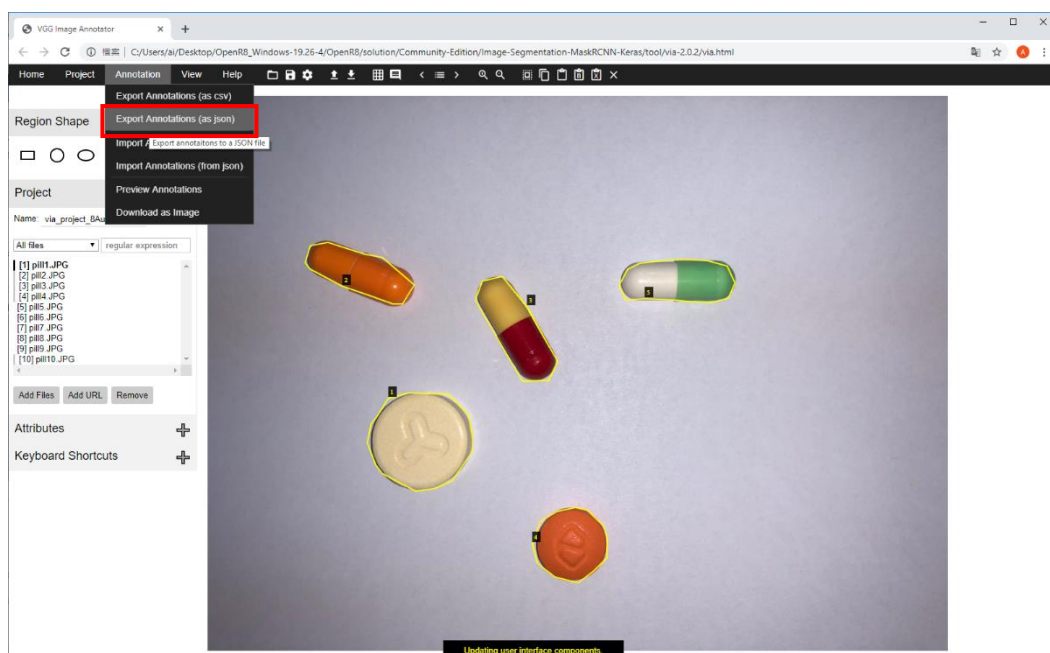


圖11. 輸出標記 json 檔

第六步：將輸出 json 檔放到 data\pill\train 資料夾內

將剛剛輸出的 json 檔 放到 data\pill\train 資料夾內，並確認檔名是否為

“via_region_data.json”，如果不是，請改名成 “via_region_data.json”，如圖 12。

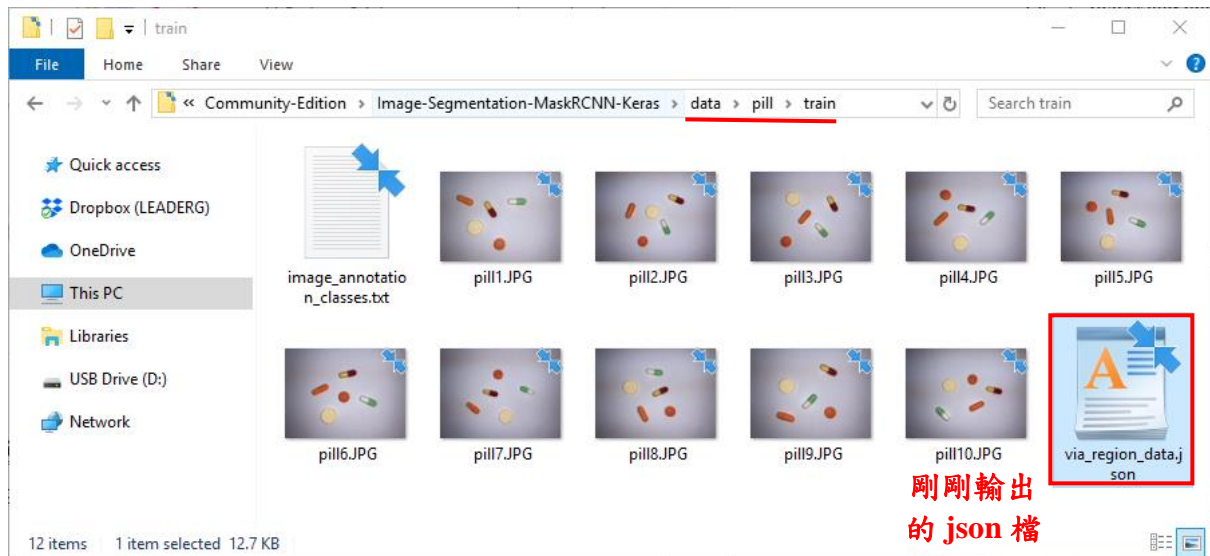


圖12. 將輸出標記 json 檔放到 data\pill\train 資料夾

※ 要測試的樣本 data\pill\val 資料夾內，一樣要做第一步到第六步。

四、執行 1_train.py 開始訓練

一開始請開啟【OpenR8 程式】，如果電腦有安裝顯示卡，請點選

【R8_Python3.6_GPU.bat】執行檔，沒有則點選【R8_Python3.6_CPU.bat】執行檔，如圖 13。開啟完【OpenR8 程式】後，請點選【檔案】=>【開啟】=>【進入到 OpenR8 底下的 solution 資料夾】=>【選擇 Image-Segmentation-MaskRCNN-Keras 資料夾】=>【選擇 1_train.py 開啟】，如圖 14、圖 15。

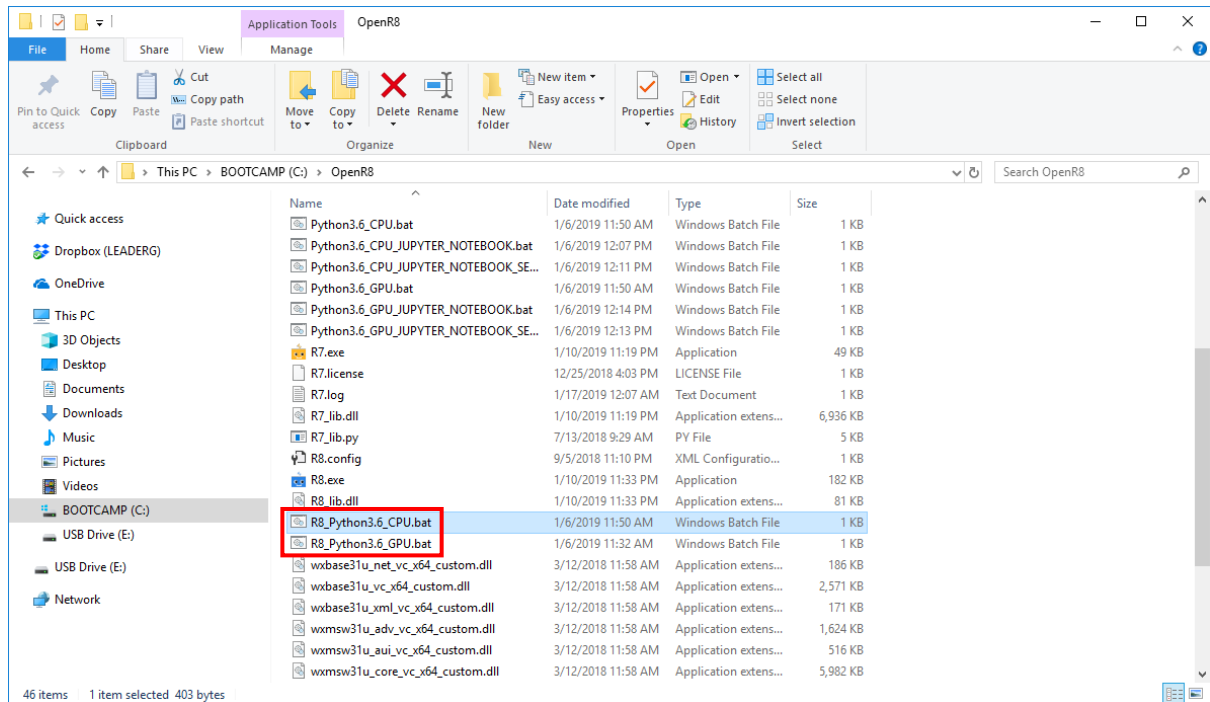


圖13. 開啟 OpenR8 程式

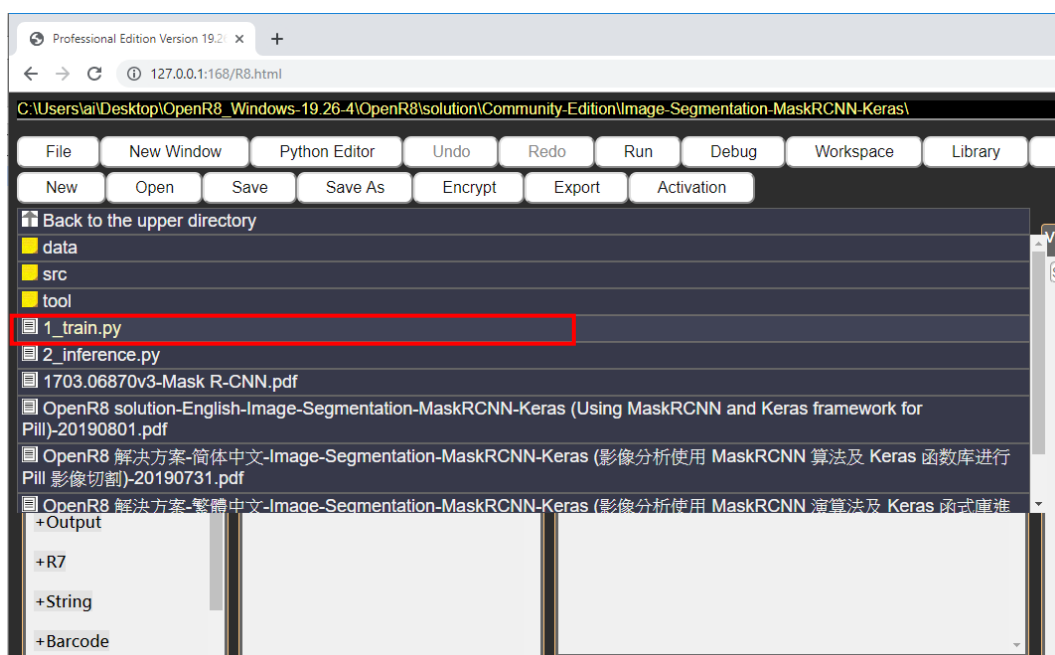


圖14. 選擇 1_train.py

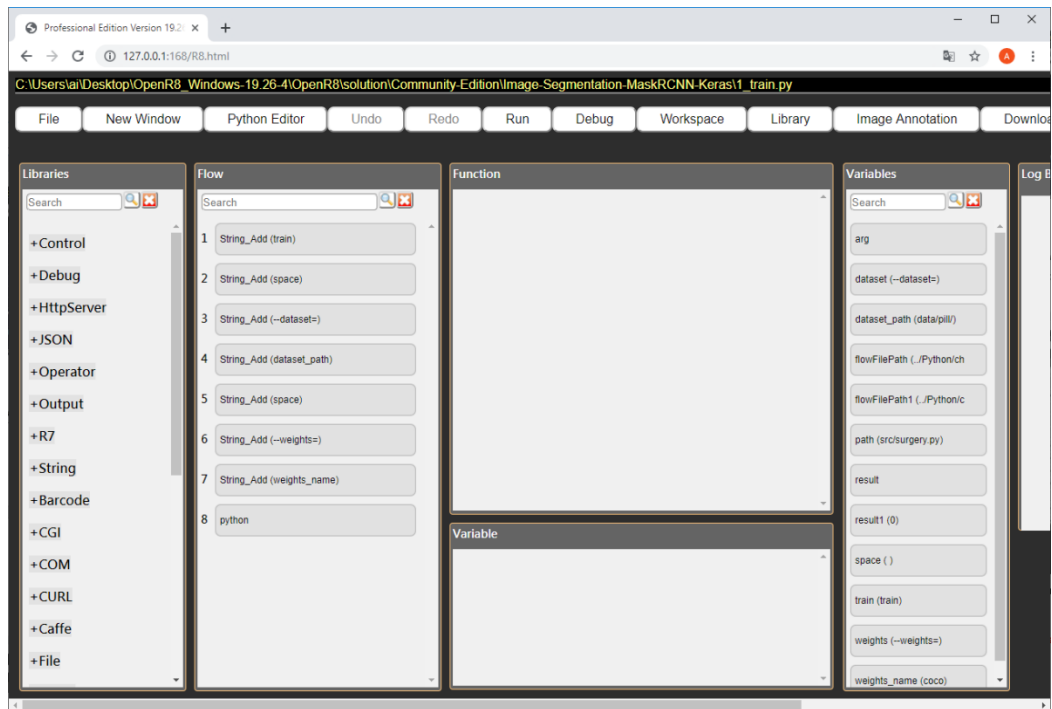


圖15. 開啟 1_train.py

※如果樣本圖沒有放在“data\pill”裡面的話，需額外設定 dataset_Path，如圖 16。

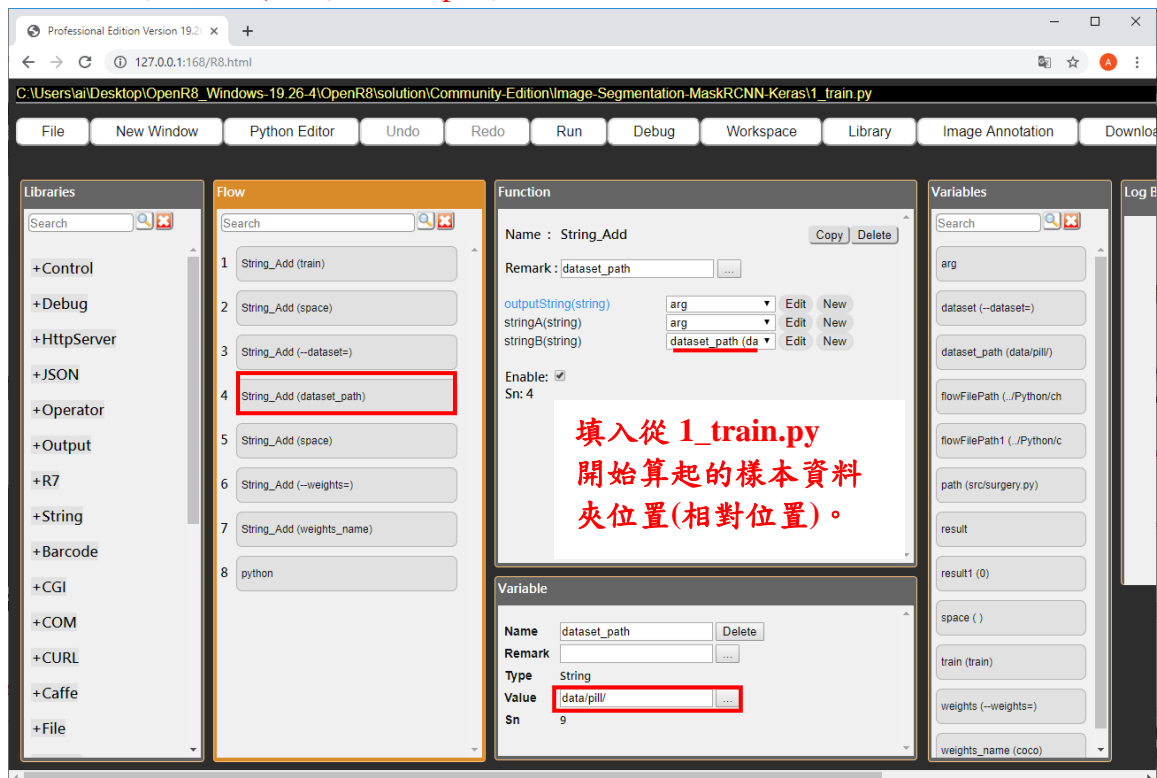


圖16. 設定 dataset_Path

※在運行前，如果沒有要沿用之前的 model，請刪除所有 h5 檔案(但保留 mask_rcnn_coco.h5)，不熟悉者建議都不刪除。

※在執行前，如果想改變“訓練模型名稱”、“訓練次數”、“分類類別”……等參數設定，請看第六章 — 參數介紹。

按下執行開始訓練樣本，直到跳出「Press any key to continue...」。

五、執行 2_inference.py 看訓練結果

在執行完 1_train.py 訓練結束後，開啟 2_inference.py 來測試圖片，如圖 17、圖 18。

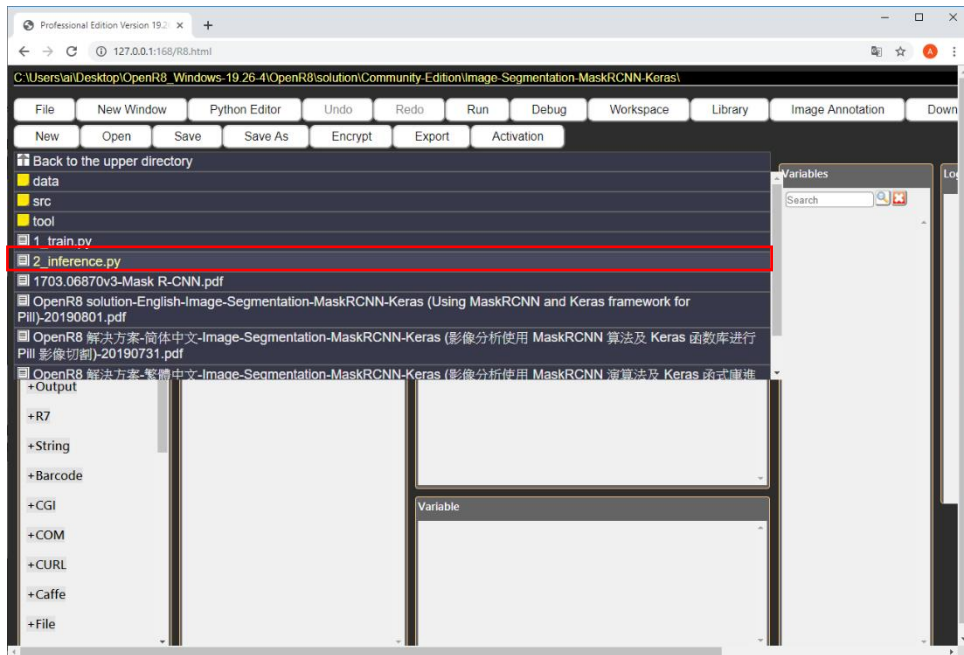


圖17. 選擇 2_inference.py

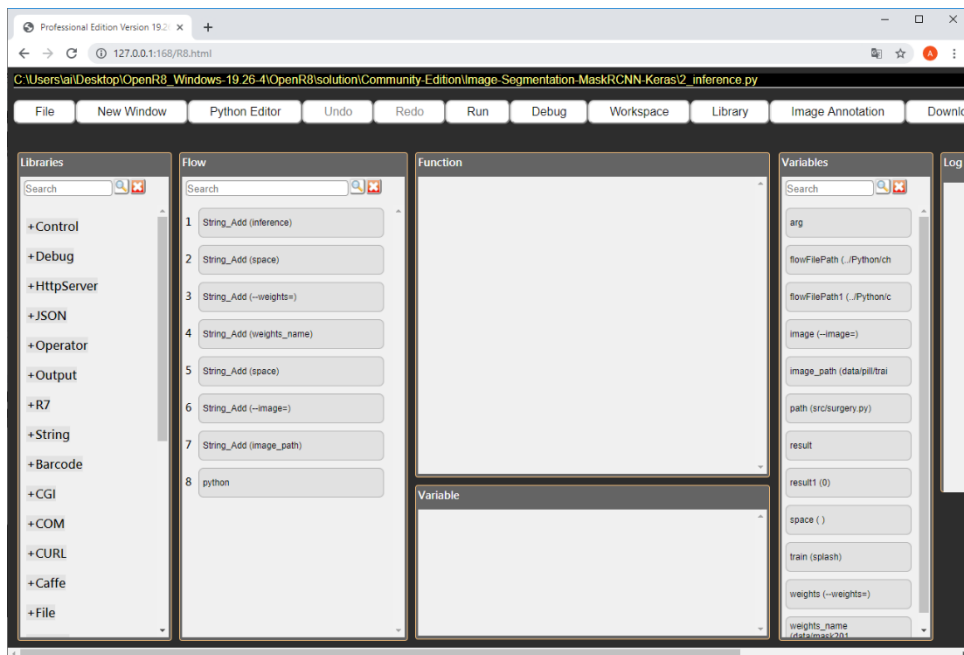


圖18. 開啟 2_inference.py

填入要測試的樣本路徑與訓練完的 h5 檔路徑，如圖 19、圖 20。

※如果有執行過 1_train.py 且成功訓練出 model 者，**務必**確認圖 20 的 h5 檔名稱是否一致。

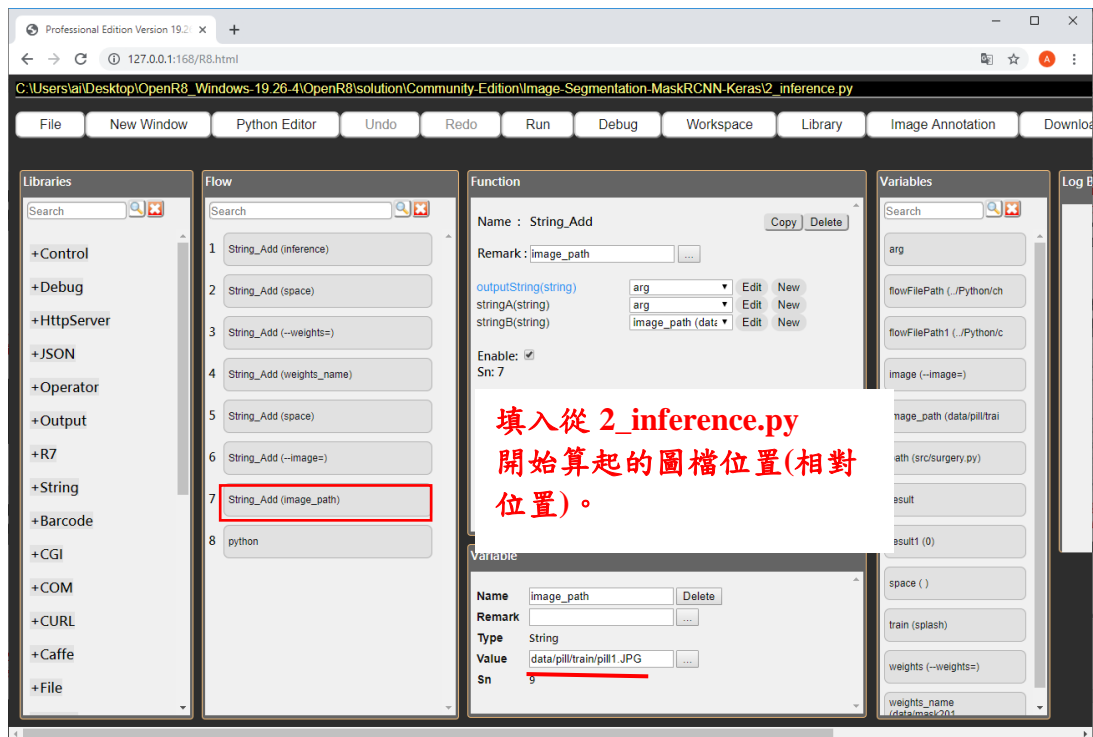


圖19. 填入要測試的樣本路徑

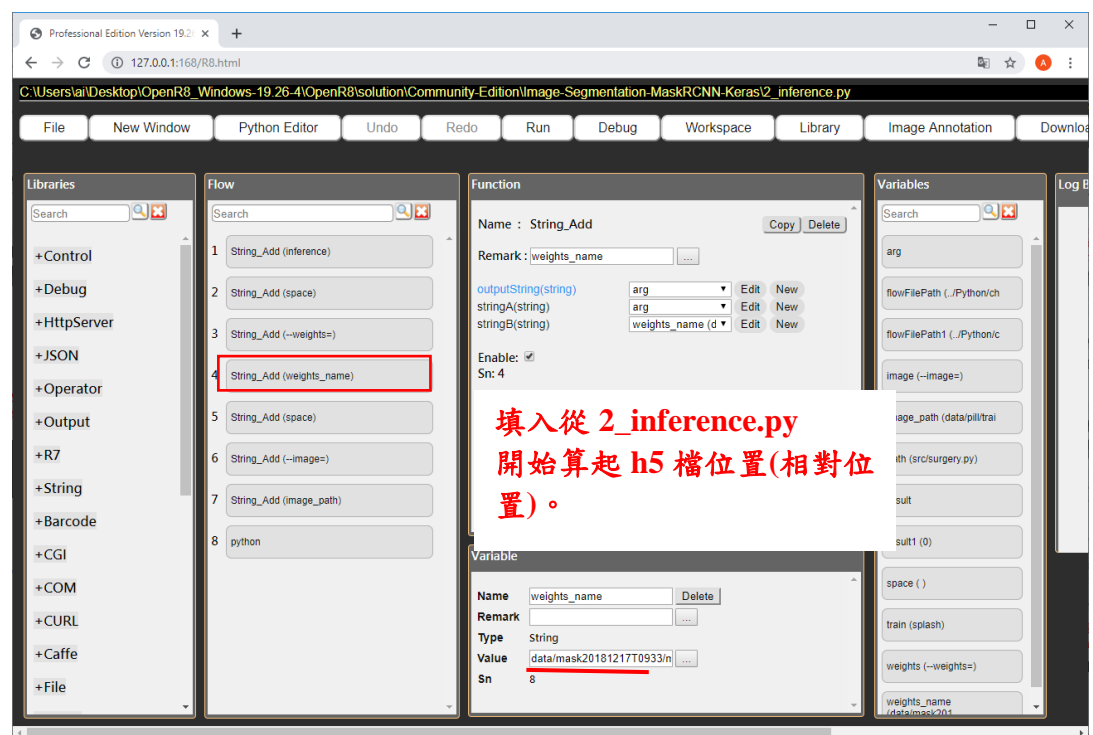


圖20. 填入要訓練的 h5 樣本路徑

按下執行看結果，Mask_R_CNN 和其他顯示結果的方式不太一樣，如果有判斷到類別時，那個區域會標記成一種顏色並框起來顯示類別及相似度，反之，如果甚麼都沒抓到就會沒有標記顏色，如圖 21，在藥丸的位置分別標記成不同顏色，代表有被抓出。

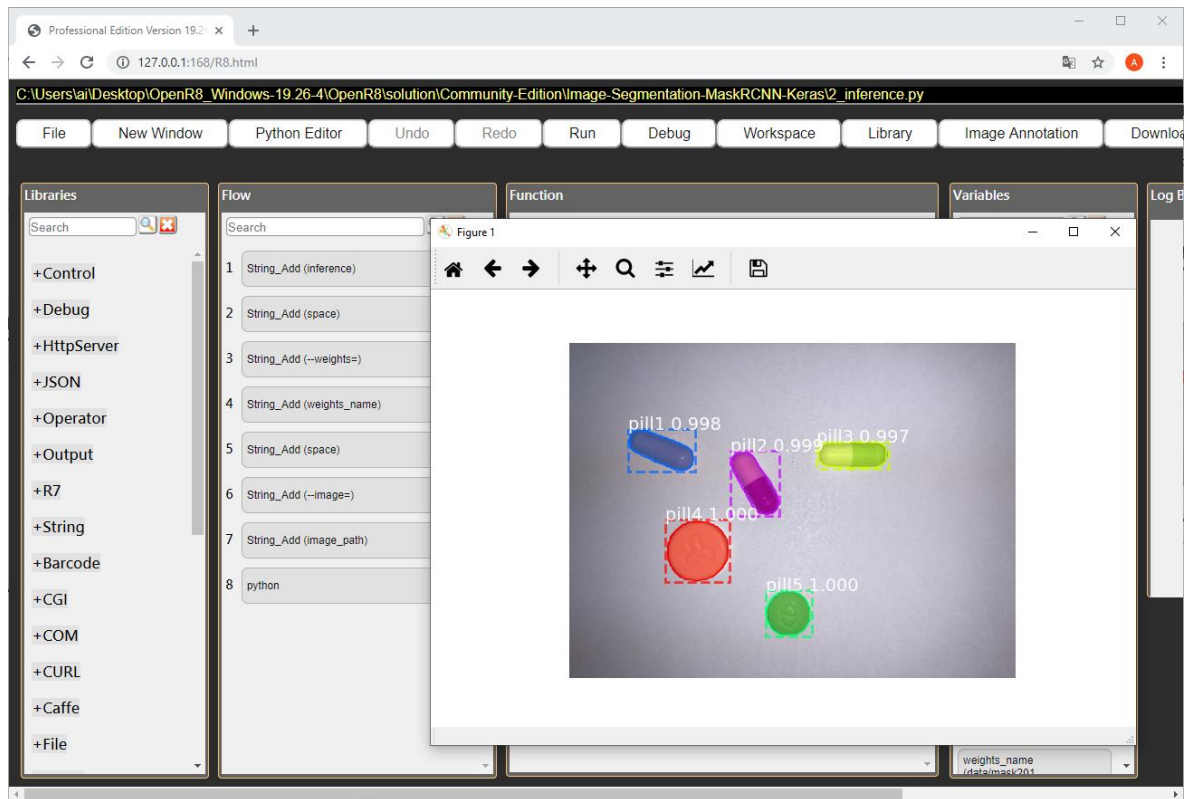


圖21. 2_inference.py 的測試結果

六、參數介紹

※ 更改讀取 h5 檔檔名：圖 22。

※ 在 “data\predefined_classes.txt” 設定類別名稱：圖 23。

※ 更改 json 名稱：圖 24。

※ 設置 GPU 數量：圖 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

```

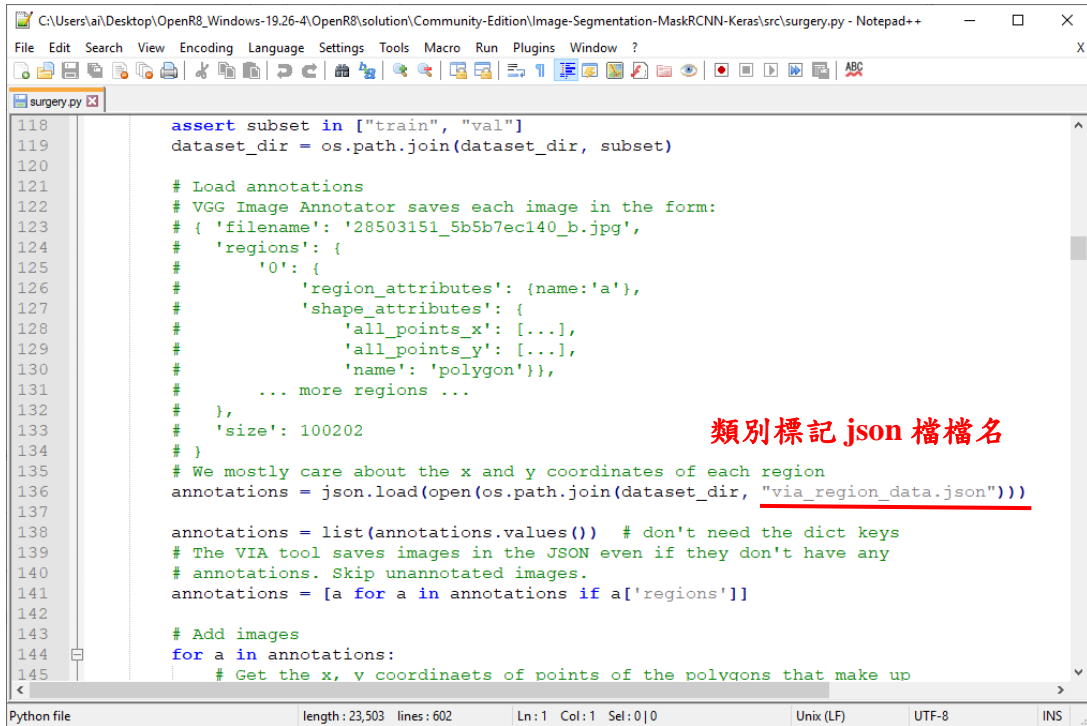
圖22. 在 surgery.py 中更改讀取 h5 檔檔名

```

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

```

圖23. 在 “data\predefined_classes.txt” 中設定類別名稱



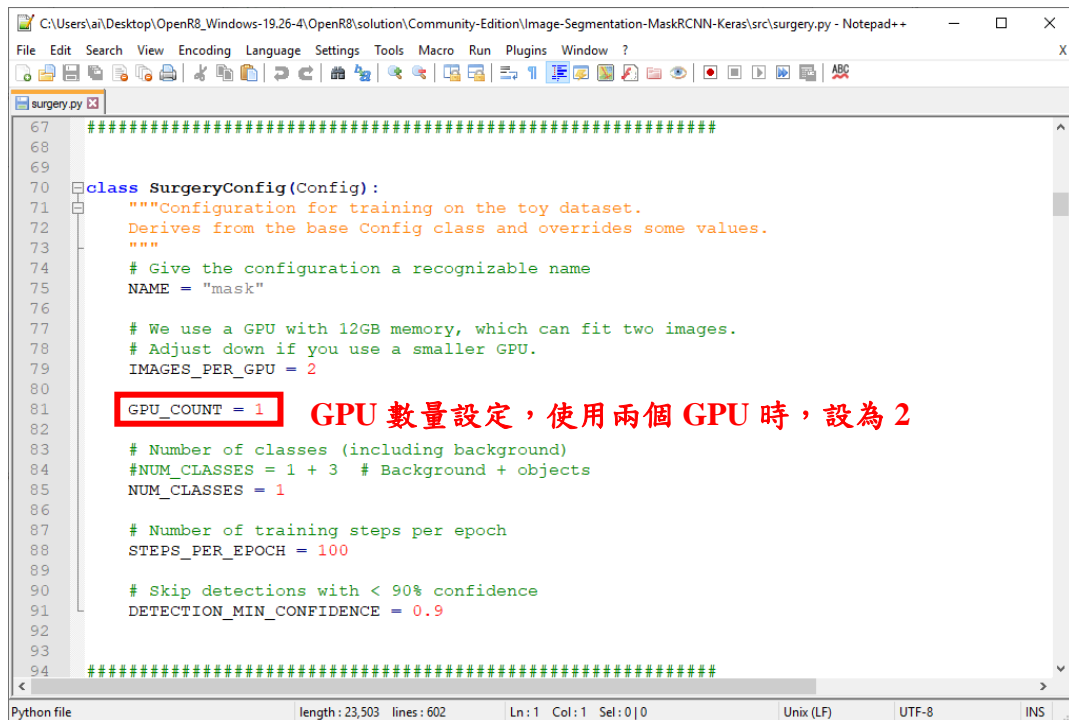
```

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

```

類別標記 json 檔檔名

圖24. 在 surgery.py 中更改讀取類別 json 名稱



```

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     #####

```

GPU 數量設定，使用兩個 GPU 時，設為 2

圖25. 在 surgery.py 中增加 GPU 數量設定