LAB6-EfficientDet

國立雲林科技大學
夏世昌 特聘教授 / 電子工程系
王斯弘 助理教授 / 前瞻學位學士學程

2022, Fall Semester

目錄

1. 環境
2. 網路、程式
3. label

目錄

1. 環境
2. 網路、程式
3. label

今天LAB請在 colab 跑
在es508 的 cuda(11.1) 不相容

要在自己電腦跑推薦 cuda10.2

在自己電腦處理資料環節會遇到的問題 (callback)

找出你的資料夾路徑: Python37/site-packages/tensorflow_examples/lite/model_maker/core/task/model_spec/object_detector_spec.py

找不到直接到conda搜尋這個檔

手動添加程式，為了繪製訓練曲線

```
    TB = TensorBoard(
        log_dir='./logs',
        histogram_freq=0,
        write_graph=True,
        write_images=False,
        update_freq="epoch",
        embeddings_freq=0,
        embeddings_metadata=None,
        )
    callbacks = train_lib.get_callbacks(config.as_dict(), val_dataset)
    callbacks.append(TB)
    print("callbacks", callbacks)
    model.fit(
        train_dataset,
        epochs=epochs,
        steps_per_epoch=steps_per_epoch,
        callbacks=callbacks,
        validation_data=val_dataset,
        validation_steps=validation_steps)
    return model
```

所需套件

安裝指令:pip install tflite_model_maker

TFLite 模型製作器的公共 API，一個用於訓練自定義 TFLite 模型的遷移學習庫。

Reference: https://www.tensorflow.org/lite/api_docs/python/tflite_model_maker

目錄

1. 環境
2. 資料集、網路、程式
3. label

dataset

雲端上的 efficient_dataset 壓縮檔

圖片

標籤

匯入函式庫

```
import numpy as np
import os
import pycocotools
import tqdm as notebook_tqdm
import tensorflow as tf
assert tf.__version__.startswith('2')

import tflite_model_maker

from tflite_model_maker.config import ExportFormat
from tflite_model_maker import model_spec
from tflite_model_maker import object_detector

tf.get_logger().setLevel('ERROR')
from absl import logging
logging.set_verbosity(logging.ERROR)

print(tf.__version__)
print(tflite_model_maker.__version__)
```

```
2.7.0
0.3.2
```

設定資料夾路徑

```
images_in = './oimage/'
annotations_in = './xml/'
```

舉例: 這是我在colab連結到我的雲端 我把dataset 放在雲端內的effcient_dataset中

```
images_in  =  '/content/drive/MyDrive/efficient_dataset/oimage/'
annotations_in  =  '/content/drive/MyDrive/efficient_dataset/xml/'
```

資料處理

```
import os
import random
import shutil

def split_dataset(images_path, annotations_path, val_split, test_split, out_path):


    """將已排序的圖像/註釋目錄拆分為訓練、驗證和測試集。
```

資料處理

```
train_dir, val_dir, test_dir = split_dataset(images_in, annotations_in,
                                             val_split=0.2, test_split=0.2,
                                             out_path='new_split-dataset_2')
```

注意圖片副檔名需要 .jpg

資料處理

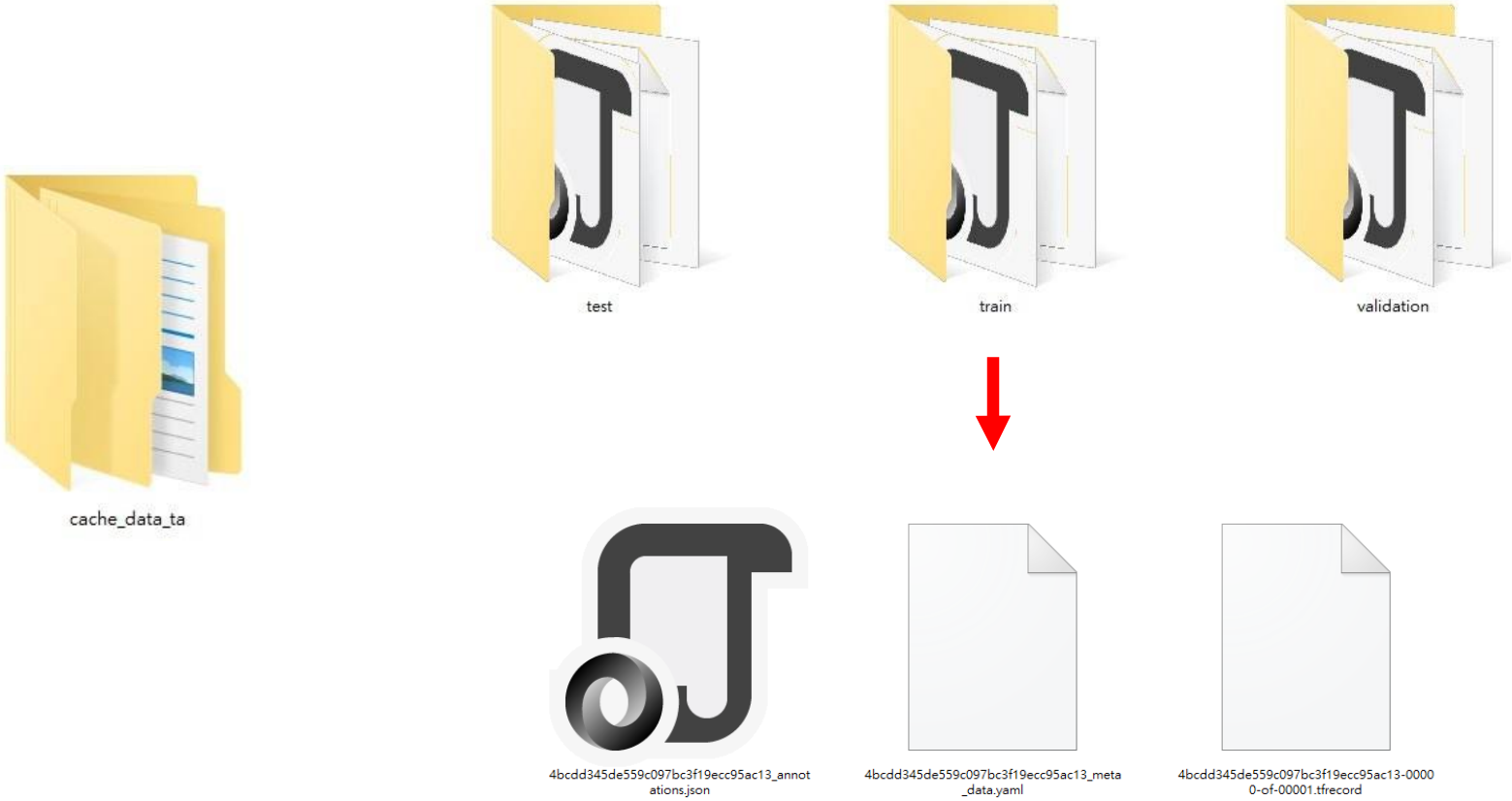
label_map需要新增你建立的標籤名稱

```
label_map= ["M", "K", "S"]
train_images_dir = './new_split-dataset_2/train/images/'
train_annotations_dir = './new_split-dataset_2/train/annotations'
val_images_dir = './new_split-dataset_2/validation/images/'
val_annotations_dir = './new_split-dataset_2/validation/annotations'
test_images_dir = './new_split-dataset_2/test/images/'
test_annotations_dir = './new_split-dataset_2/test/annotations'
```

資料處理

```
train_data = object_detector.DataLoader.from_pascal_voc(
    train_images_dir, train_annotations_dir, label_map=label_map, cache_dir="./cache_data_ta/train",

validation_data = object_detector.DataLoader.from_pascal_voc(
    val_images_dir, val_annotations_dir, label_map=label_map, cache_dir="./cache_data_ta/validation"

test_data = object_detector.DataLoader.from_pascal_voc(
   test_images_dir, test_annotations_dir, label_map=label_map, cache_dir="./cache_data_ta/test",num_
```

資料處理


test
train
validation
cache_data_ta
4bcdd345de559c097bc3f19ecc95ac13_annotations.json
4bcdd345de559c097bc3f19ecc95ac13_meta_data.yaml
4bcdd345de559c097bc3f19ecc95ac13-00000-of-00001.tfrecord

資料處理

```
train_data = object_detector.DataLoader.from_cache('./cache_data_ta/train/4bcdd345de559c097bc3f19ecc95ac13')
validation_data = object_detector.DataLoader.from_cache('./cache_data_ta/validation/4bcdd345de559c097bc3f19ecc95ac13
test_data = object_detector.DataLoader.from_cache('./cache_data_ta/test/4bcdd345de559c097bc3f19ecc95ac13')
```

```
print(f'train count: {len(train_data)}')
print(f'validation count: {len(validation_data)}')
print(f'test count: {len(test_data)}')
```

```
train count: 291
validation count: 97
test count: 97
```

訓練

```
spec = object_detector.EfficientDetLite0Spec()
```

```
model = object_detector.create(train_data=train_data,
                               model_spec=spec,
                               validation_data=validation_data,
                               epochs=200,
                               batch_size=16,
                               train_whole_model=True,
                               do_train=True)
```

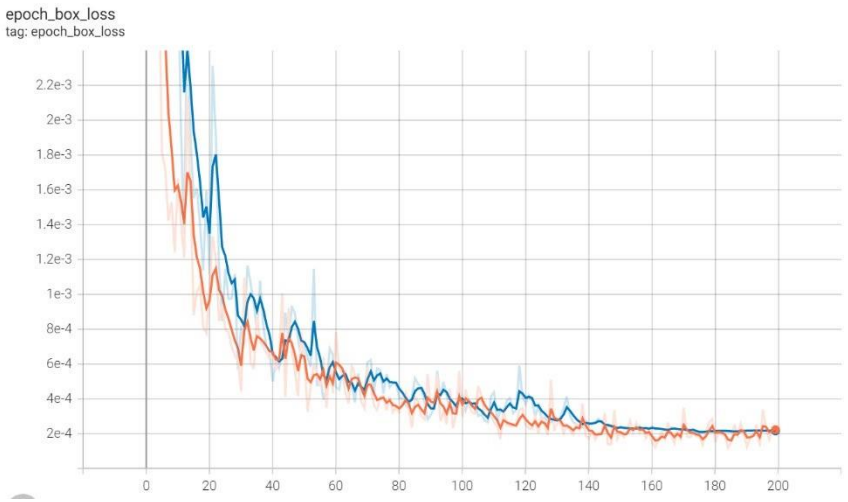
- Epochs、batch_size可以適當調整

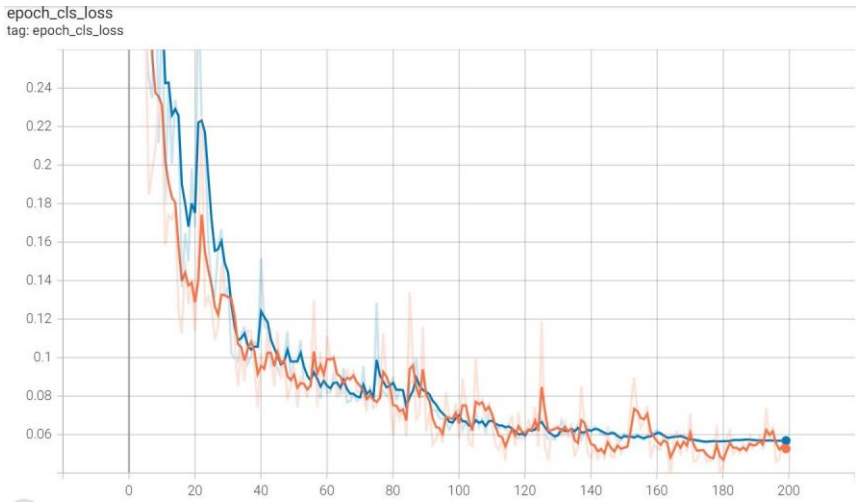
```
callbacks [<keras.callbacks.ModelCheckpoint object at 0x0000025987893348>, <keras.callbacks.TensorBoard object at
0x0000025989A41588>, <tensorflow_examples.lite.model_maker.third_party.efficientdet.keras.train_lib.COCOCallback o
bject at 0x0000025989A39AC8>, <keras.callbacks.TensorBoard object at 0x00000257C26591C8>]
Epoch 1/200
18/18 [==============================] - 43s 657ms/step - det_loss: 1.7116 - cls_loss: 1.1293 - box_loss: 0.0116 -
reg_l2_loss: 0.0634 - loss: 1.7750 - learning_rate: 0.0140 - gradient_norm: 1.1063 - val_det_loss: 1.7066 - val_cl
s_loss: 1.1288 - val_box_loss: 0.0116 - val_reg_l2_loss: 0.0634 - val_loss: 1.7700
Epoch 2/200
18/18 [==============================] - 10s 536ms/step - det_loss: 1.2997 - cls_loss: 0.8761 - box_loss: 0.0085 -
reg_l2_loss: 0.0634 - loss: 1.3631 - learning_rate: 0.0200 - gradient_norm: 1.7163 - val_det_loss: 1.1881 - val_cl
s_loss: 0.8238 - val_box_loss: 0.0073 - val_reg_l2_loss: 0.0634 - val_loss: 1.2515
Epoch 3/200
18/18 [==============================] - 9s 521ms/step - det_loss: 0.8016 - cls_loss: 0.5283 - box_loss: 0.0055 -
reg_l2_loss: 0.0635 - loss: 0.8650 - learning_rate: 0.0200 - gradient_norm: 1.8455 - val_det_loss: 1.1127 - val_cl
s_loss: 0.7659 - val_box_loss: 0.0069 - val_reg_l2_loss: 0.0635 - val_loss: 1.1762
Epoch 4/200
18/18 [==============================] - 9s 518ms/step - det_loss: 0.6037 - cls_loss: 0.4209 - box_loss: 0.0037 -
reg_l2_loss: 0.0635 - loss: 0.6672 - learning_rate: 0.0200 - gradient_norm: 1.9602 - val_det_loss: 0.9970 - val_cl
s_loss: 0.6466 - val_box_loss: 0.0070 - val_reg_l2_loss: 0.0636 - val_loss: 1.0605
```

驗證

終端機上 &> tensorboard --logdir=.\logs\

沒有跳出網頁請把終端機輸出的網頁複製到瀏覽器





驗證

Evaluate the model

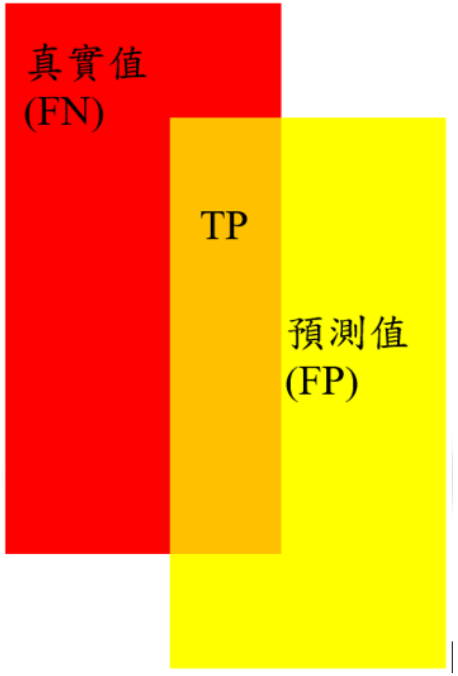
```
model.evaluate(test_data)
```

```
2/2 [==============================] - 12s 5s/step
```

```
{'AP': 0.94376695,
 'AP50': 1.0,
 'AP75': 1.0,
 'APs': -1.0,
 'APm': -1.0,
 'APl': 0.943777,
 'ARmax1': 0.9462973,
 'ARmax10': 0.9634138,
 'ARmax100': 0.9634138,
 'ARs': -1.0,
 'ARm': -1.0,
 'ARl': 0.9634138,
 'AP_/M': 0.938481,
 'AP_/K': 0.9886693,
 'AP_/S': 0.9790732,
 'AP_/P': 0.8688444}
```

驗證

$$\mathrm{MIoU} = \frac{\mathrm{TP}}{\mathrm{FP} + \mathrm{FN} + \mathrm{TP}}$$



- AP: IoU[0.5:0.95], Stride:0.05 , (mean Average Precision)
- AP50: IoU > 50% (mean Average Precision)
- AP75: IoU> 75% (mean Average Precision)

匯出權重檔

```
TFLITE_FILENAME = 'efficientdet.tflite'
LABELS_FILENAME = 'labels.txt'
SAVED_FILENAME = 'model'
```

```
model.export(export_dir='./export', tflite_filename=TFLITE_FILENAME, label_filename=LABELS_FILENAME, saved_model_fil
             export_format=[ExportFormat.TFLITE, ExportFormat.LABEL, ExportFormat.SAVED_MODEL])
```

測試

TFLite_test.ipynb

```
import tensorflow as tf
import cv2 as cv
import numpy as np
import glob

tf.compat.v1.reset_default_graph()
interpreter = tf.lite.Interpreter('./export/'+'efficientdet.tflite') # 讀取 .tflite 模型
interpreter.allocate_tensors()                                        # 初始化模型
```

測試

TFLite_test.ipynb

```
import tensorflow as tf
import cv2 as cv
import numpy as np
import glob

tf.compat.v1.reset_default_graph()
interpreter = tf.lite.Interpreter('./export/'+'efficientdet.tflite') # 讀取 .tflite 模型
interpreter.allocate_tensors()                                        # 初始化模型
```

測試

```
w,h,c=input_details[0]['shape'][1:4]            # 獲取輸入圖片大小
img_file='./new_split-dataset_2/test/images/20220425_230252_006.jpg' # 一個圖片路徑
imgc = 0
Pred = 0


img=cv.imread(img_file)                         # 讀取圖片
img=img.astype(np.uint8)                        # 轉整數
img=cv.resize(img,(w,h))                        # 調整圖片大小
img=cv.cvtColor(img,cv.COLOR_BGR2RGB)           # BGR to RGB
img=img.reshape((w,h,c))                        # 調整陣列大小
img=np.expand_dims(img,0)                       # 轉 4D 陣列

interpreter.set_tensor(input_details[0]['index'], img)                # 輸入資料到模型
interpreter.invoke()                                                  # 模型計算輸出

detection_scores = interpreter.get_tensor(output_details[0]['index']) # 獲取輸出各框的得分
detection_boxes = interpreter.get_tensor(output_details[1]['index']) # 獲取輸出框座標（百分比）
num_boxes = interpreter.get_tensor(output_details[2]['index'])  # 獲取輸出總框數量
detection_classes = interpreter.get_tensor(output_details[3]['index']) # 輸出框類別
```

測試

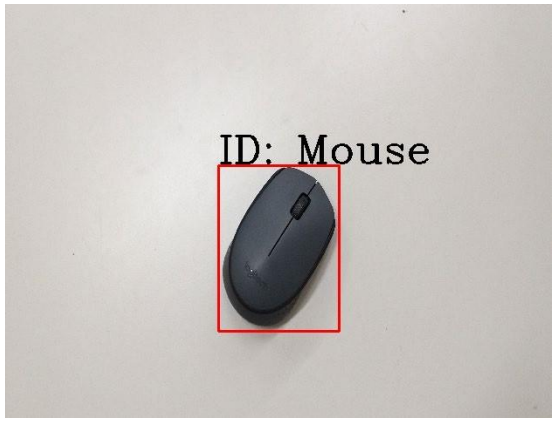
```
outimg=cv.imread(img_file)
for i in range(0,num_boxes):
    # 把百分比座標轉為實際圖片座標
    detection_boxes[i,0]=int(detection_boxes[i,0]*outimg.shape[0])
    detection_boxes[i,1]=int(detection_boxes[i,1]*outimg.shape[1])
    detection_boxes[i,2]=int(detection_boxes[i,2]*outimg.shape[0])
    detection_boxes[i,3]=int(detection_boxes[i,3]*outimg.shape[1])
    # 得分大於 0.5 (可自行調整) 才畫出框及文字敘述

    if detection_scores[i]>0.5:
        outimg=cv.rectangle(outimg, (int(detection_boxes[i,1]), int(detection_boxes[i,0])), (int(detection_boxes[i,3
        if detection_classes[i] ==1:
            ID = 'Mouse'
        elif detection_classes[i] ==2:
            ID = 'Keyboard'
        elif detection_classes[i] ==3:
            ID = 'Screen'
        outimg=cv.putText(outimg,'ID: %s'%(ID), (int(detection_boxes[i,1]),int(detection_boxes[i,0]-3)),cv.FONT_HERS
cv.imshow('out', outimg)
key = cv.waitKey(0)
if key>0:
    cv.destroyAllWindows()
```

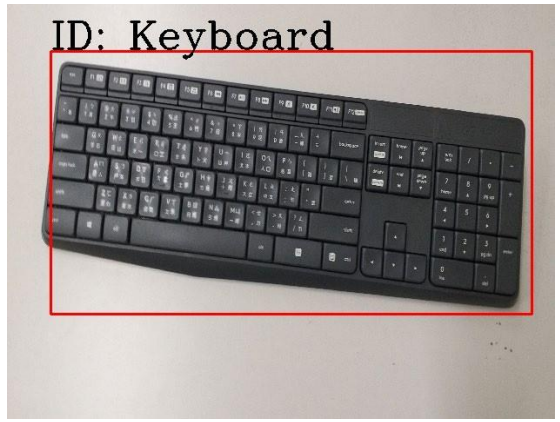
測試

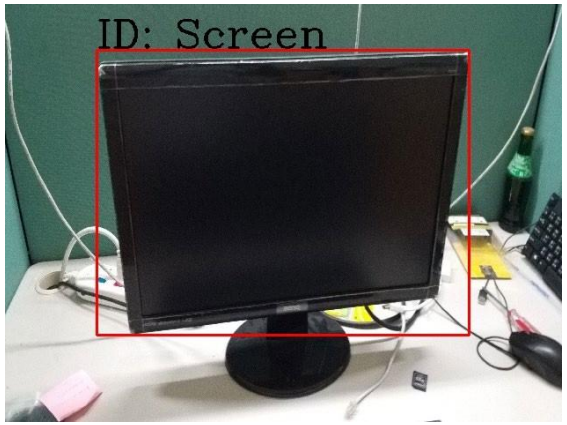
Colab無法使用imshow 改成存圖片

```
cv.imwrite('out.jpg',outimg)
'''
cv.imshow('out',  outimg)
key  =  cv.waitKey(0)
if  key>0:
        cv.destroyAllWindows()
'''
```

測試







目錄

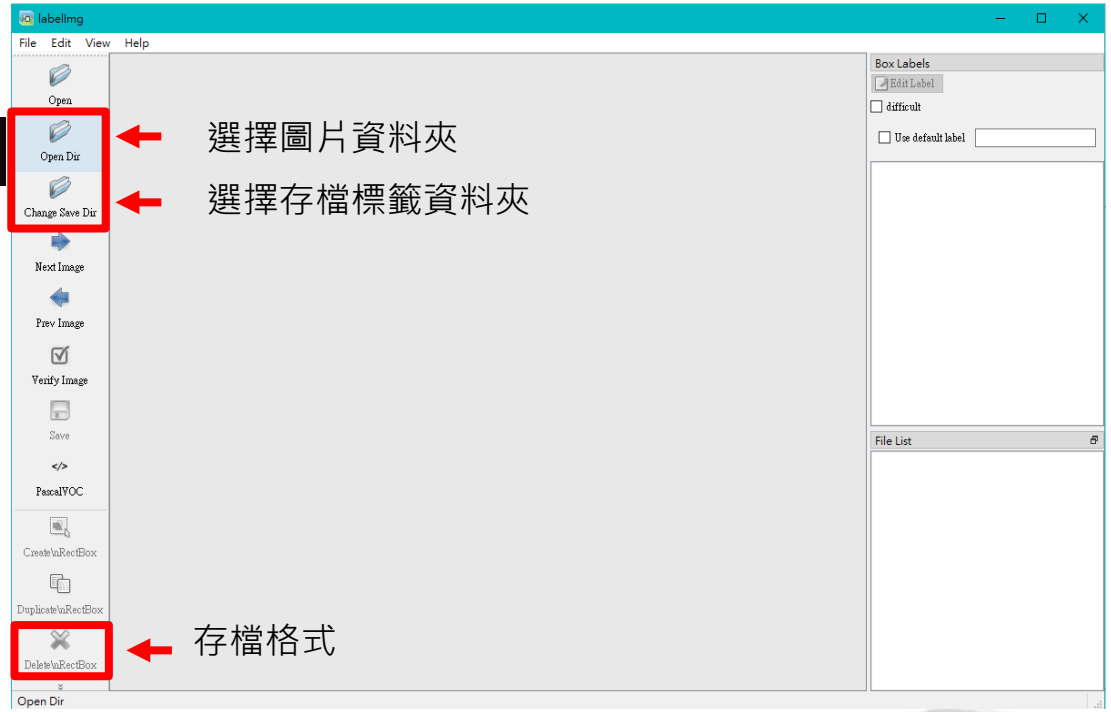
1. 環境
2. 資料集、網路、程式
3. label

Label

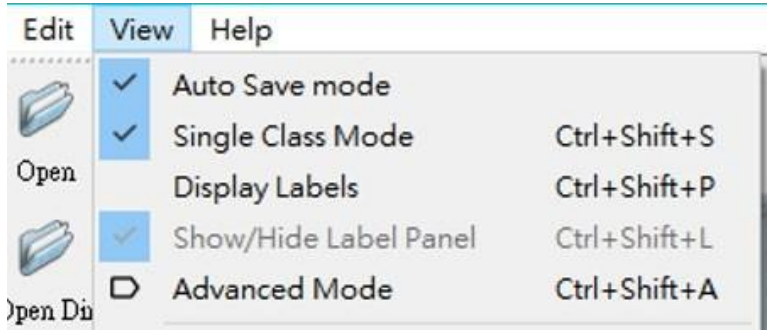
pip install **labelimg**

開啟**labelimg**

```
(keras) C:\Users\es402\Desktop>labelimg
```



Label


Edit
View
Help
Open
Auto Save mode
Single Class Mode Ctrl+Shift+S
Display Labels Ctrl+Shift+P
Show/Hide Label Panel Ctrl+Shift+L
Advanced Mode Ctrl+Shift+A

Label

手機請盡量調低相機像素、並打橫著拍，我的圖片是3264x1836

W 框選

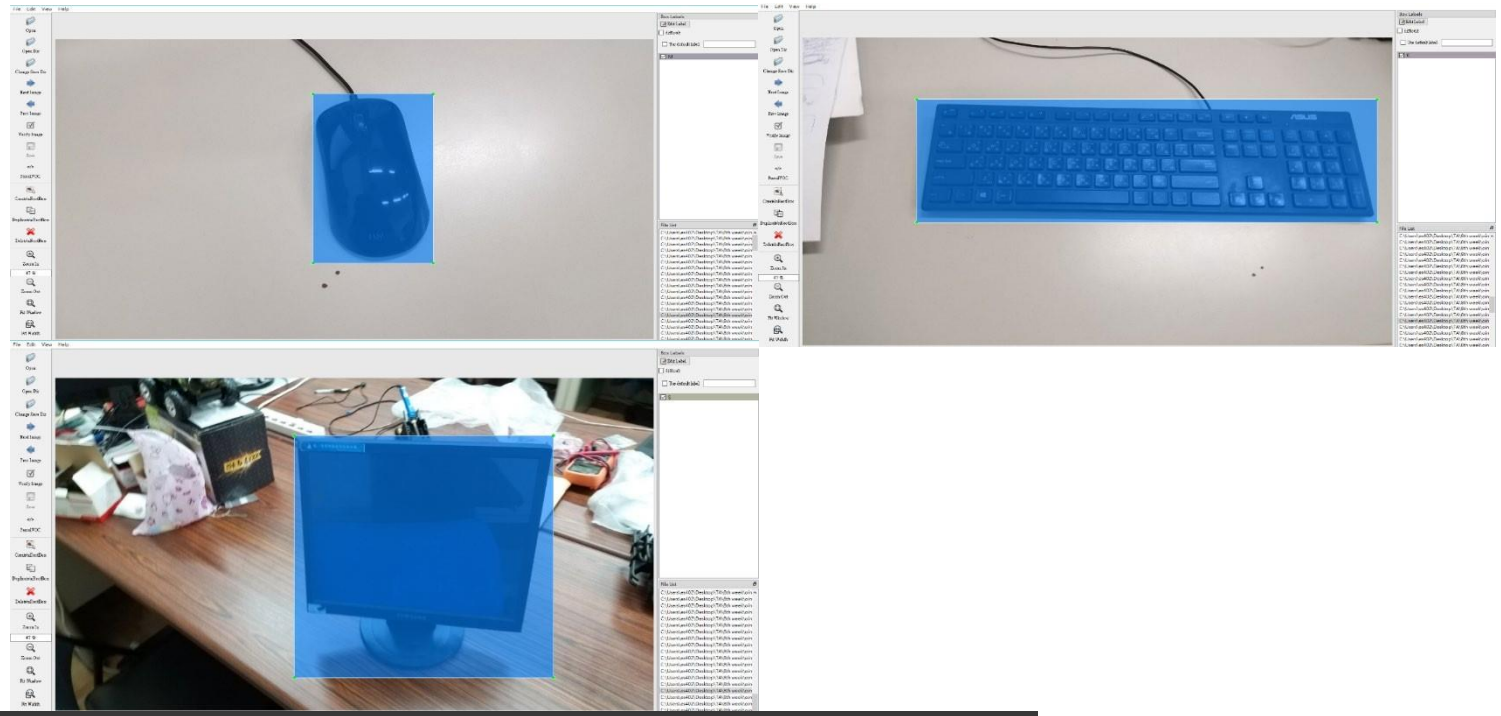
A 上一張

D 下一張

Delete 刪除框選

Ctrl+S 存檔

Label



Label:
M:滑鼠
K:鍵盤
S:螢幕

```
label_map= ["M", "K", "S"]
train_images_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/train/images/'
train_annotations_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/train/annotations'
val_images_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/validation/images/'
val_annotations_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/validation/annotations'
test_images_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/test/images/'
test_annotations_dir = './content/drive/MyDrive/Colab Notebooks/efficientdet/test/annotations'
```

END