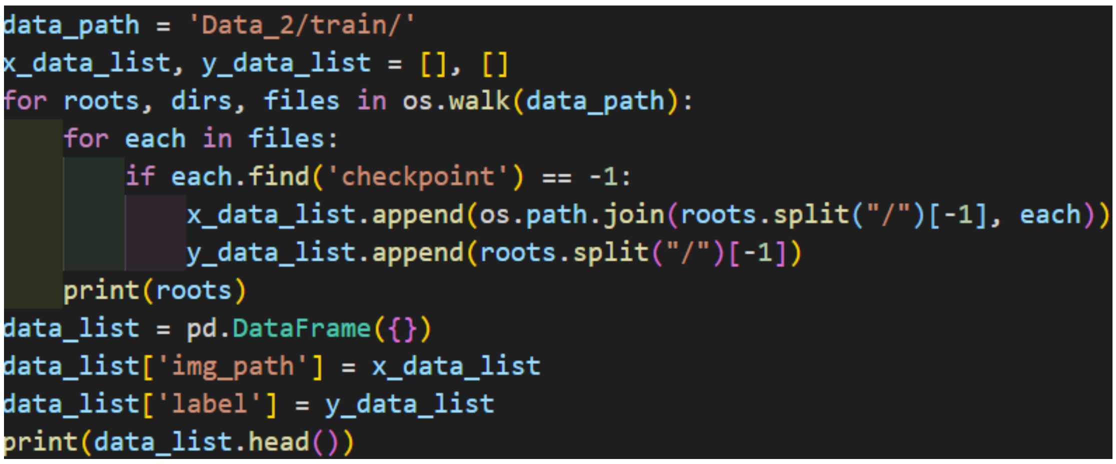
Final Programming Exam. (part 2)
Deep Learning, 2022

姓名：吳佳展　　學號：110318517

此題的答案的整體架構為幾大步驟分別為：a.讀取資料 b.切分 train data 和 test data c.設計模型架構 d.訓練模型 e.劃出訓練模型的 loss f.使用模型進行產出預測圖片分類 g.將其測試集資料和分類值合併匯出 csv 檔案，詳細步驟下方將會逐一說明各項過程。

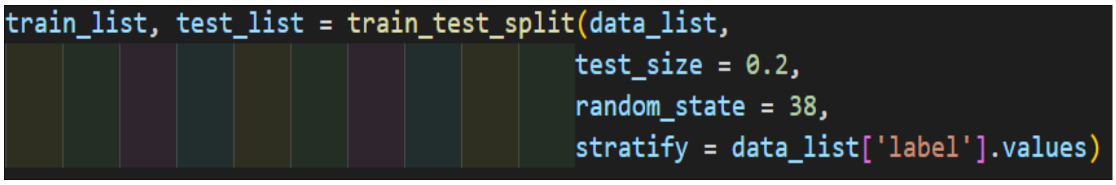
整體架構圖：

1. 第 1 步先讀取各個資料夾的檔案照片當作 data。



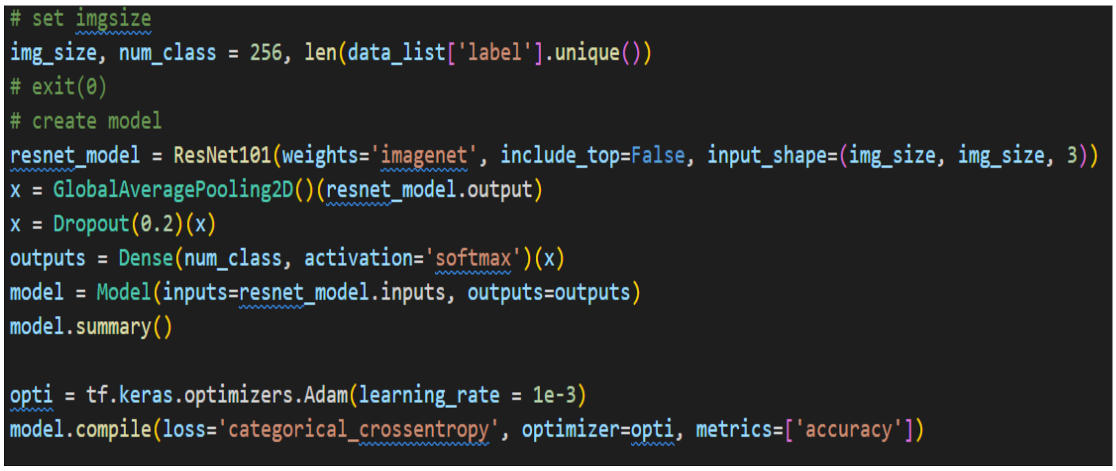
```
data_path = 'Data_2/train/'
x_data_list, y_data_list = [], []
for roots, dirs, files in os.walk(data_path):
    for each in files:
        if each.find('checkpoint') == -1:
            x_data_list.append(os.path.join(roots.split("/")[-1], each))
            y_data_list.append(roots.split("/")[-1])
    print(roots)
data_list = pd.DataFrame({})
data_list['img_path'] = x_data_list
data_list['label'] = y_data_list
print(data_list.head())
```

2. 第 2 步驟分割成 train data 和 test data



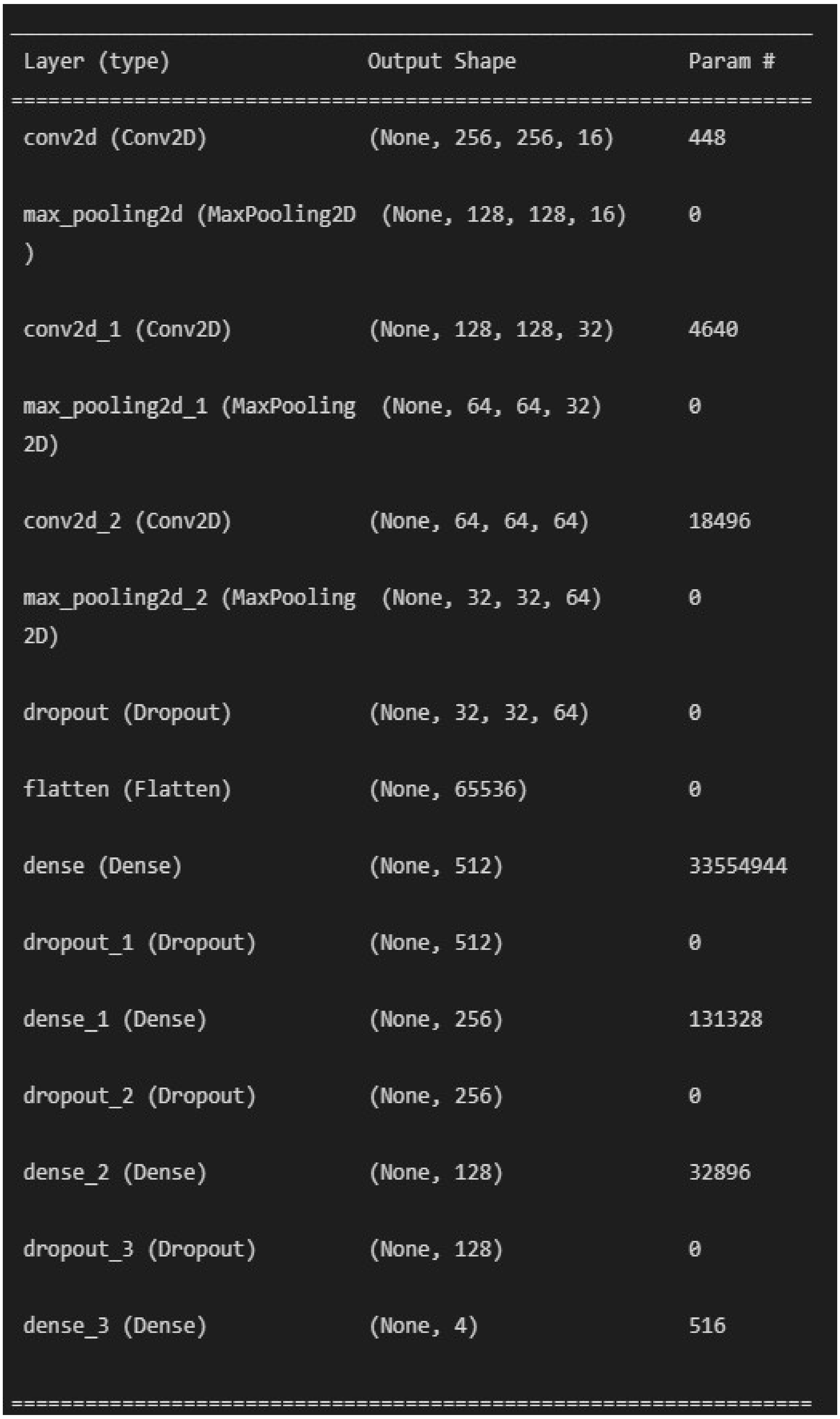
```
train_list, test_list = train_test_split(data_list,
                                         test_size = 0.2,
                                         random_state = 38,
                                         stratify = data_list['label'].values)
```

3. 第三步驟設計 imgsize 大小並 create model。



```
# set imgsize
img_size, num_class = 256, len(data_list['label'].unique())
# exit(0)
# create model
resnet_model = ResNet101(weights='imagenet', include_top=False, input_shape=(img_size, img_size, 3))
x = GlobalAveragePooling2D()(resnet_model.output)
x = Dropout(0.2)(x)
outputs = Dense(num_class, activation='softmax')(x)
model = Model(inputs=resnet_model.inputs, outputs=outputs)
model.summary()

opti = tf.keras.optimizers.Adam(learning_rate = 1e-3)
model.compile(loss='categorical_crossentropy', optimizer=opti, metrics=['accuracy'])
```

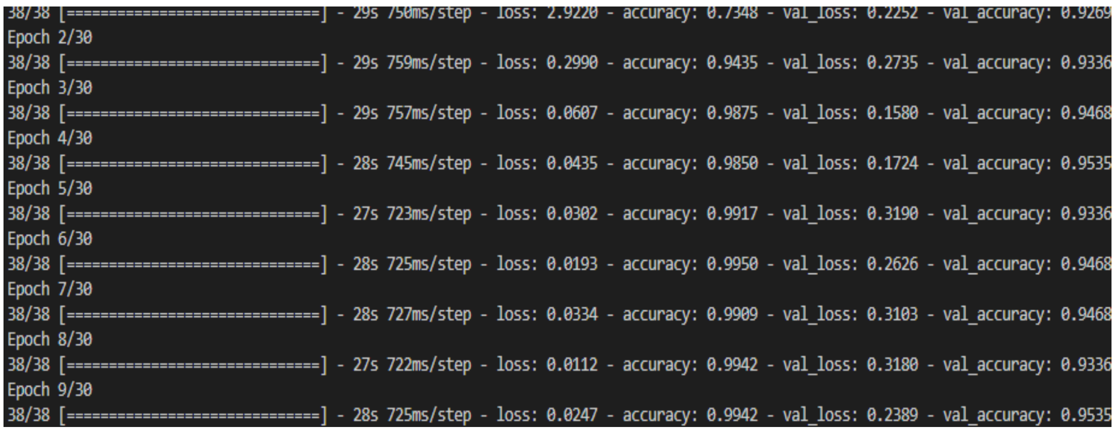
模型架構及參數如下圖所示



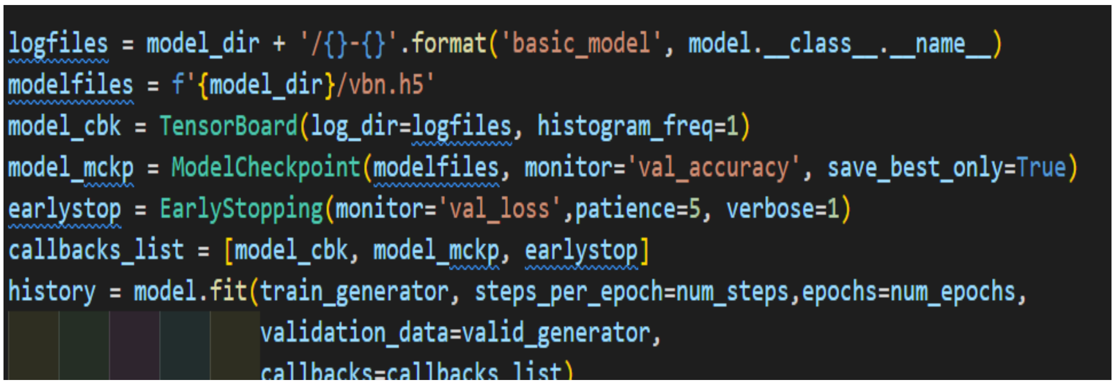
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 16)      448

 max_pooling2d (MaxPooling2D  (None, 128, 128, 16)     0
 )

 conv2d_1 (Conv2D)           (None, 128, 128, 32)      4640

 max_pooling2d_1 (MaxPooling  (None, 64, 64, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 64, 64, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 32, 32, 64)       0
 2D)

 dropout (Dropout)           (None, 32, 32, 64)        0

 flatten (Flatten)           (None, 65536)             0

 dense (Dense)               (None, 512)               33554944

 dropout_1 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 256)               131328

 dropout_2 (Dropout)         (None, 256)               0

 dense_2 (Dense)             (None, 128)               32896

 dropout_3 (Dropout)         (None, 128)               0

 dense_3 (Dense)             (None, 4)                 516

=================================================================
```

4. 第四步驟將 TRAINING 資料放入模型訓練，並將模型存取，而模型訓練的參數如下表所示。

參數名稱	Loss	Optimizer	Epochs	Batch_size
所使用參數	categorical_crossentropy	adam	100	30

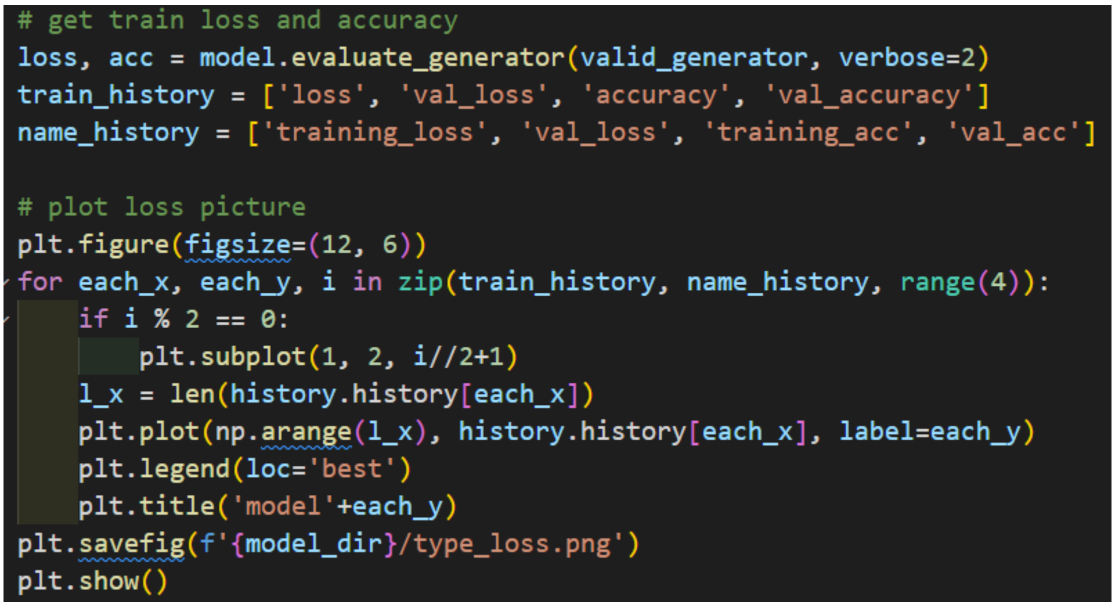


```
38/38 [==============================] - 29s 750ms/step - loss: 2.9220 - accuracy: 0.7348 - val_loss: 0.2252 - val_accuracy: 0.9269
Epoch 2/30
38/38 [==============================] - 29s 759ms/step - loss: 0.2990 - accuracy: 0.9435 - val_loss: 0.2735 - val_accuracy: 0.9336
Epoch 3/30
38/38 [==============================] - 29s 757ms/step - loss: 0.0607 - accuracy: 0.9875 - val_loss: 0.1580 - val_accuracy: 0.9468
Epoch 4/30
38/38 [==============================] - 28s 745ms/step - loss: 0.0435 - accuracy: 0.9850 - val_loss: 0.1724 - val_accuracy: 0.9535
Epoch 5/30
38/38 [==============================] - 27s 723ms/step - loss: 0.0302 - accuracy: 0.9917 - val_loss: 0.3190 - val_accuracy: 0.9336
Epoch 6/30
38/38 [==============================] - 28s 725ms/step - loss: 0.0193 - accuracy: 0.9950 - val_loss: 0.2626 - val_accuracy: 0.9468
Epoch 7/30
38/38 [==============================] - 28s 727ms/step - loss: 0.0334 - accuracy: 0.9909 - val_loss: 0.3103 - val_accuracy: 0.9468
Epoch 8/30
38/38 [==============================] - 27s 722ms/step - loss: 0.0112 - accuracy: 0.9942 - val_loss: 0.3180 - val_accuracy: 0.9336
Epoch 9/30
38/38 [==============================] - 28s 725ms/step - loss: 0.0247 - accuracy: 0.9942 - val_loss: 0.2389 - val_accuracy: 0.9535
```

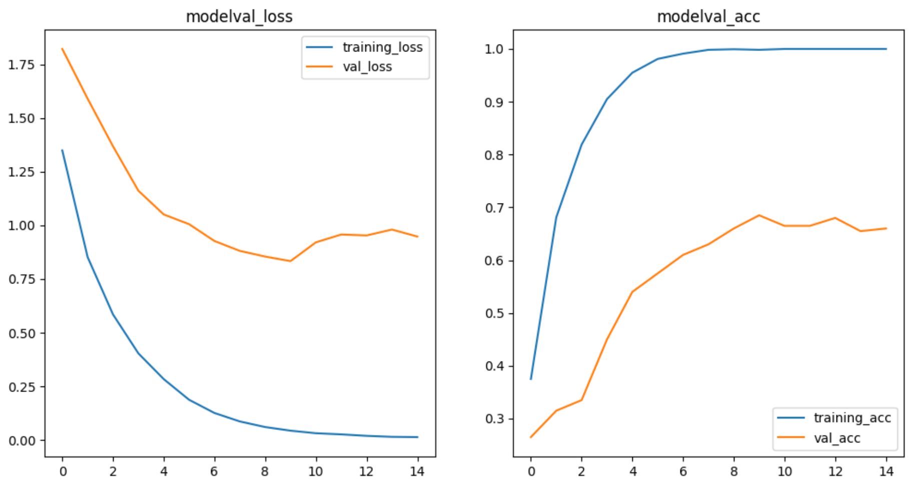


```python
logfiles = model_dir + '/{}-{}'.format('basic_model', model.__class__.__name__)
modelfiles = f'{model_dir}/vbn.h5'
model_cbk = TensorBoard(log_dir=logfiles, histogram_freq=1)
model_mckp = ModelCheckpoint(modelfiles, monitor='val_accuracy', save_best_only=True)
earlystop = EarlyStopping(monitor='val_loss',patience=5, verbose=1)
callbacks_list = [model_cbk, model_mckp, earlystop]
history = model.fit(train_generator, steps_per_epoch=num_steps,epochs=num_epochs,
                    validation_data=valid_generator,
                    callbacks=callbacks_list)
```

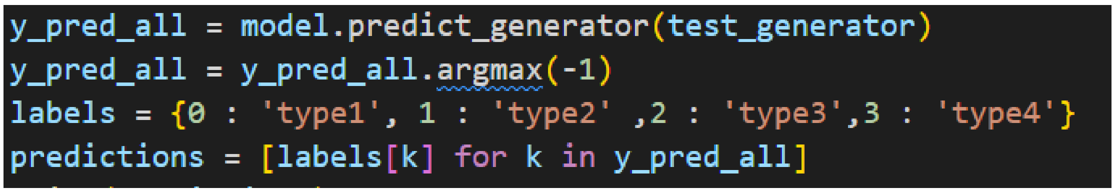
5. 第五步驟為將所訓練過程的 learn error 用圖表所示，判斷是否有 overfitting 或是 underfitting 的狀態。



```python
# get train loss and accuracy
loss, acc = model.evaluate_generator(valid_generator, verbose=2)
train_history = ['loss', 'val_loss', 'accuracy', 'val_accuracy']
name_history = ['training_loss', 'val_loss', 'training_acc', 'val_acc']

# plot loss picture
plt.figure(figsize=(12, 6))
for each_x, each_y, i in zip(train_history, name_history, range(4)):
    if i % 2 == 0:
        plt.subplot(1, 2, i//2+1)
    l_x = len(history.history[each_x])
    plt.plot(np.arange(l_x), history.history[each_x], label=each_y)
    plt.legend(loc='best')
    plt.title('model'+each_y)
plt.savefig(f'{model_dir}/type_loss.png')
plt.show()
```

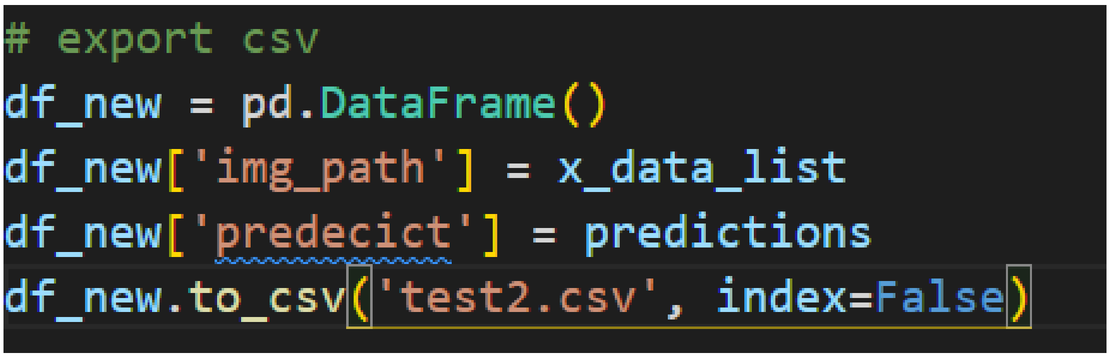


6. 第六步驟將 test_data 使用 test.py 進行預測，進而產生出預測分類。



```
y_pred_all = model.predict_generator(test_generator)
y_pred_all = y_pred_all.argmax(-1)
labels = {0 : 'type1', 1 : 'type2' ,2 : 'type3',3 : 'type4'}
predictions = [labels[k] for k in y_pred_all]
```

7. 第七步驟為將 test_data 和預測值組合起來輸出成“test2.csv”。



```
# export csv
df_new = pd.DataFrame()
df_new['img_path'] = x_data_list
df_new['predecict'] = predictions
df_new.to_csv('test2.csv', index=False)
```