利用CNN辨識水果名稱

109034549 溫芳苓

指導教授：邱銘傳 博士

目錄

1

研究背景

研究背景

- 現今水果業者大多採用人工結帳方式，其依水果外型、顏色、表皮紋理等特徵來逐一辨識水果名稱，不僅耗時且可能辨識錯誤。

- 當消費者於水果攤販不了解某項水果名稱時，其可能需耗費時間尋找或等待服務人員，以取得資訊。

研究背景

- 本研究利用CNN 將九種水果進行分類，分別為蘋果、酪梨、香蕉、櫻桃、芭樂、奇異果、檸檬、芒果、橘子，並將本研究之模型提供予水果業者或消費者，以協助其辨識水果名稱。

- 本研究之模型可提升水果業者結帳速度並降低人工辨識錯誤率，亦可減少消費者尋找或等待服務人員所耗費的時間。

5W1H

項目	內容
What	消費者於水果攤販可能不了解某水果名稱，而需尋找或等待服務人員以取得資訊；水果業者販售水果時需耗費時間逐一辨識水果名稱，且可能辨識錯誤造成財物損失。
Where	販售水果之店面（水果攤販）。
Who	欲了解某水果名稱之消費者、欲提升結帳速度或降低人工辨識錯誤率之水果業者。
When	當消費者欲了解某水果名稱時、水果業者欲提升結帳速度或降低人工辨識錯誤率時。
Why	建構水果辨識之模型，以減少消費者尋找或等待服務人員所耗費之時間，或提升水果業者結帳速度並降低人工辨識錯誤率。
How	九種水果圖片的資料前處理與CNN 模型訓練，以協助消費者或水果業者辨識水果名稱。

2

研究過程

卷積神經網路

- CNN 是一種前饋神經網路，其由一個或多個卷積層、池化層和全連接層組成。
- CNN之結構使其能夠利用輸入資料的二維結構，且其對於大型圖像處理有出色的表現。
- CNN需要考量的參數較少，使其成為一種頗具吸引力的深度學習結構。

資料說明

- 本研究由Kaggle公開數據集取得九種水果的圖像資料，而圖像皆由RGB三原色所組成。 30%為驗證集，70%為訓練集

水果名稱	訓練集圖像資料數	測試集圖像資料數	圖像資料總數
蘋果（Apple）	492	164	656
酪梨（Avocado）	427	143	570
香蕉（Banana）	490	166	656
櫻桃（Cherry）	492	164	656
芭樂（Guava）	490	166	656
奇異果（Kiwi）	466	156	622
檸檬（Lemon）	492	164	656
芒果（Mango）	490	166	656
橘子（Orange）	479	160	639
總和	4,318	1,449	5,767

資料前處理

- 圖像大小皆為100*100，且將圖像像素值標準化（圖像像素值/255），以等比例縮放至0~1，並利用One-hot encoding將圖像標籤進行有效編碼。

```
#normalization and reshaping
X_train=X_train.reshape(-1,img_size,img_size,3)
X_train=X_train/255
X_test=X_test.reshape(-1,img_size,img_size,3)
X_test=X_test/255
print("shape of X_train= ",X_train.shape)
print("shape of X_test= ",X_test.shape)
```

```
shape of X_train= (4318, 100, 100, 3)
shape of X_test= (1449, 100, 100, 3)
```

```
from keras.utils import to_categorical
Y_train=to_categorical(Y_train,num_classes=9)
Y_test=to_categorical(Y_test,num_classes=9)
```

資料擴增

- 利用Data Augmentation 將圖像進行縮放、移動、旋轉、翻轉，以避免過擬合情況發生。

```
datagen=ImageDataGenerator(featurewise_center=False,  #set input mean to 0
                           samplewise_center=False,    #set each sample mean to 0
                           featurewise_std_normalization=False,  #divide input datas to std
                           samplewise_std_normalization=False,    #divide each datas to own std
                           zca_whitening=False,    #dimension reduction
                           rotation_range=0.5,          #rotate 5 degree
                           zoom_range=0.5,              #zoom in-out 5%
                           width_shift_range=0.5,  #shift 5%
                           height_shift_range=0.5,
                           horizontal_flip=False,    #randomly flip images
                           vertical_flip=False,
                           )

datagen.fit(x_train)
```

旋轉：0.5

縮放：0.5

改變寬度：0.5

改變高度：0.5

水平翻轉：False

垂直翻轉：False

建立模型

- 利用Sequential 模型，其中包含2層卷積層、2層池化層、1層扁平層、2層全連接層；為了避免過擬合情況發生亦加上Dropout 層。

```
model=Sequential()
model.add(Conv2D(filters=8,kernel_size=(3,3),padding="Same",activation="relu",input_shape=(100,100,3)))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(0.35))

model.add(Conv2D(filters=16,kernel_size=(3,3),padding="Same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Dropout(0.35))


model.add(Flatten())
model.add(Dense(512,activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(9,activation="softmax"))
```

建立模型

```
#defining  optimizer
optimizer=Adam(lr=0.0025,beta_1=0.9,beta_2=0.999)
#compile  the  model
model.compile(optimizer=optimizer,loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()


epochs=10
batch_size=70
```

3

研究結果

模型校度分析

- 準確率

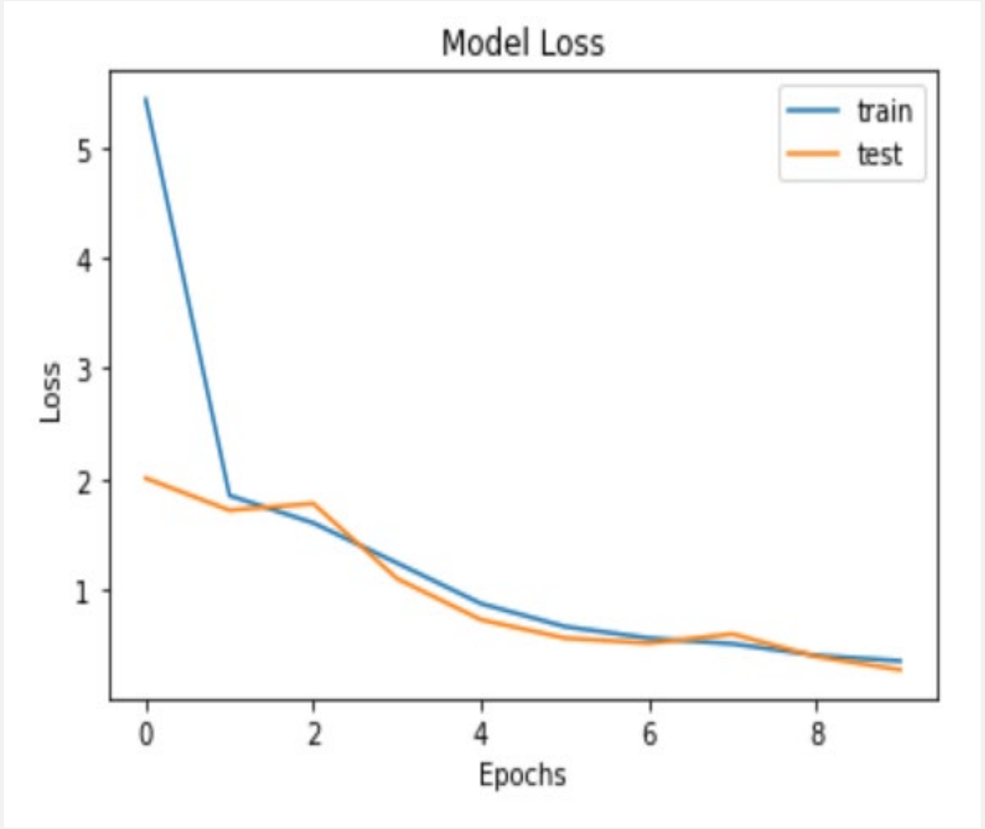
```
train_loss, train_acc = model.evaluate(x_train, y_train, batch_size=batch_size)
test_loss, test_acc = model.evaluate(x_val, y_val, batch_size=batch_size)
print('Train accuracy:', train_acc)
print('Test accuracy:', test_acc)
```

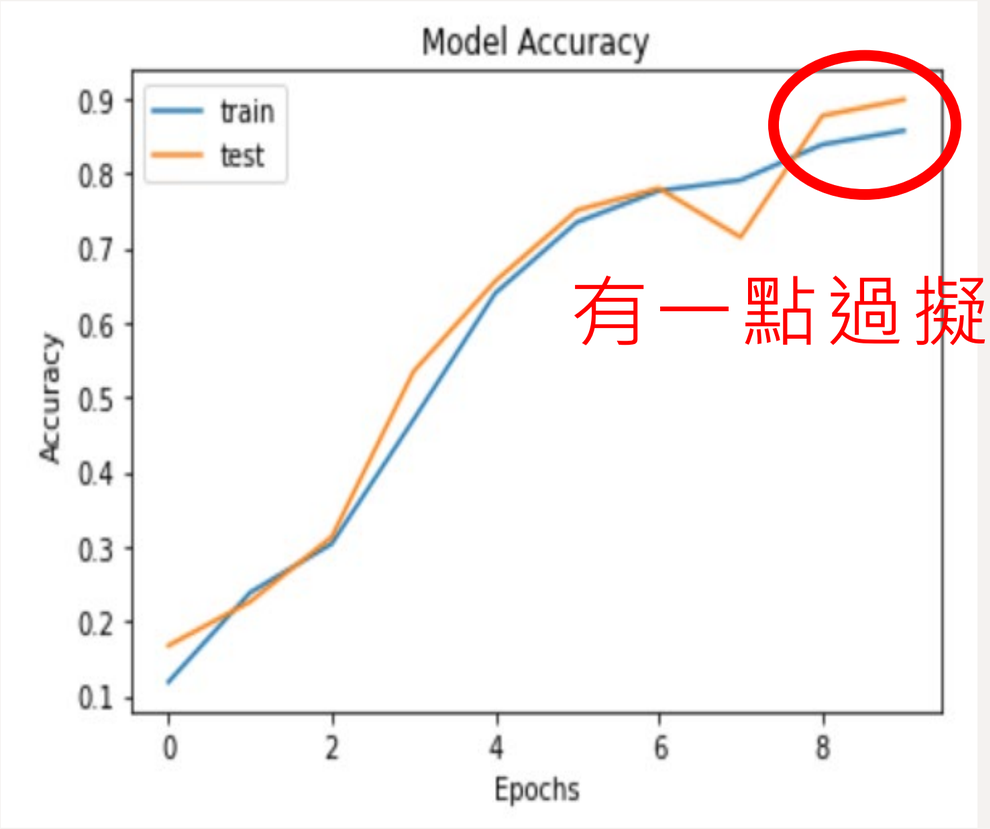
```
44/44 [==============================] - 6s 135ms/step - loss: 0.2369 - accuracy: 0.9136
19/19 [==============================] - 3s 133ms/step - loss: 0.2680 - accuracy: 0.8989
Train accuracy: 0.9136333465576172
Test accuracy: 0.8989197611808777
```

約90％

模型校度分析

- 繪製Loss、Accuracy與Epochs之關係圖

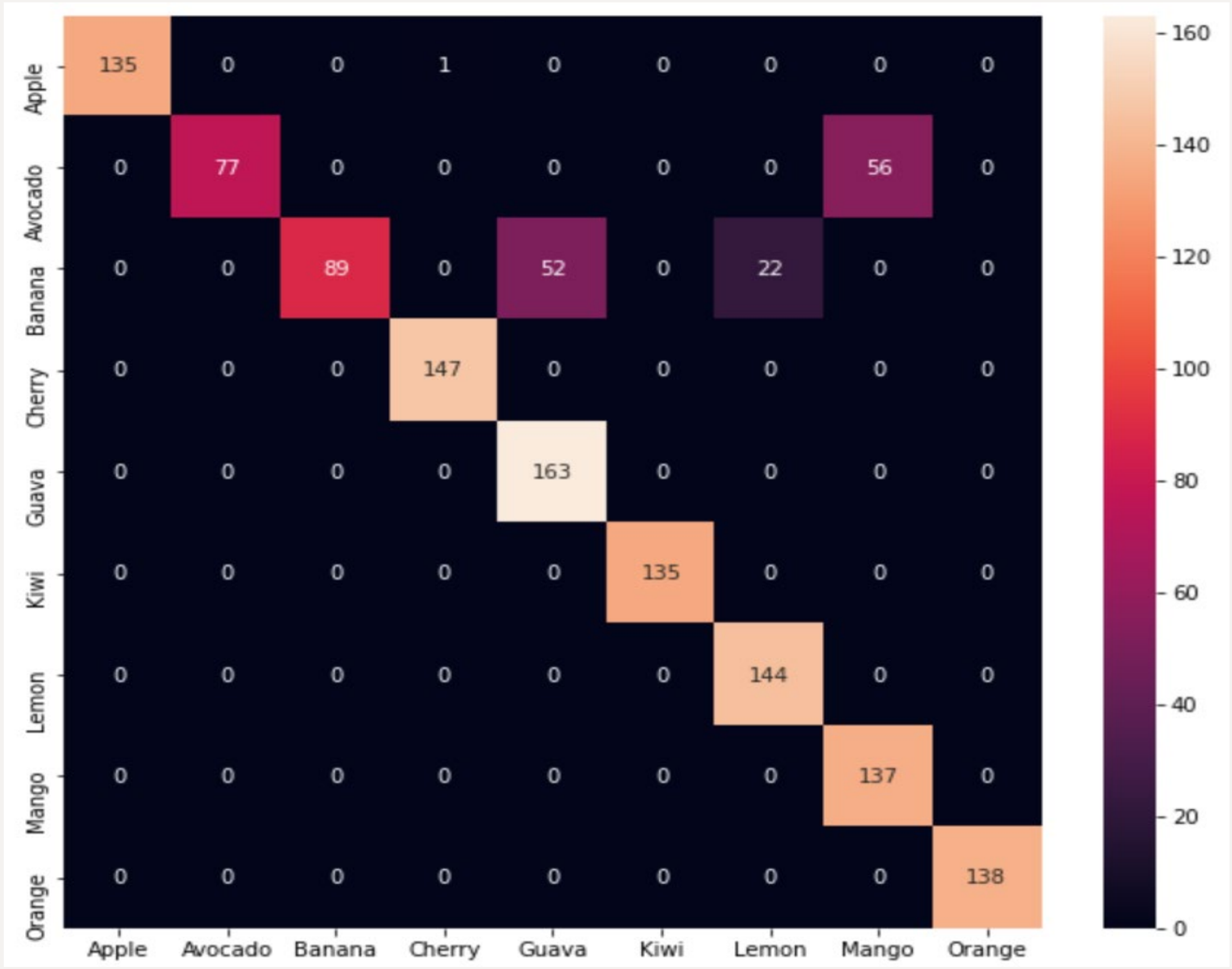




有一點過擬合情況

模型校度分析

- 驗證集之混淆矩陣

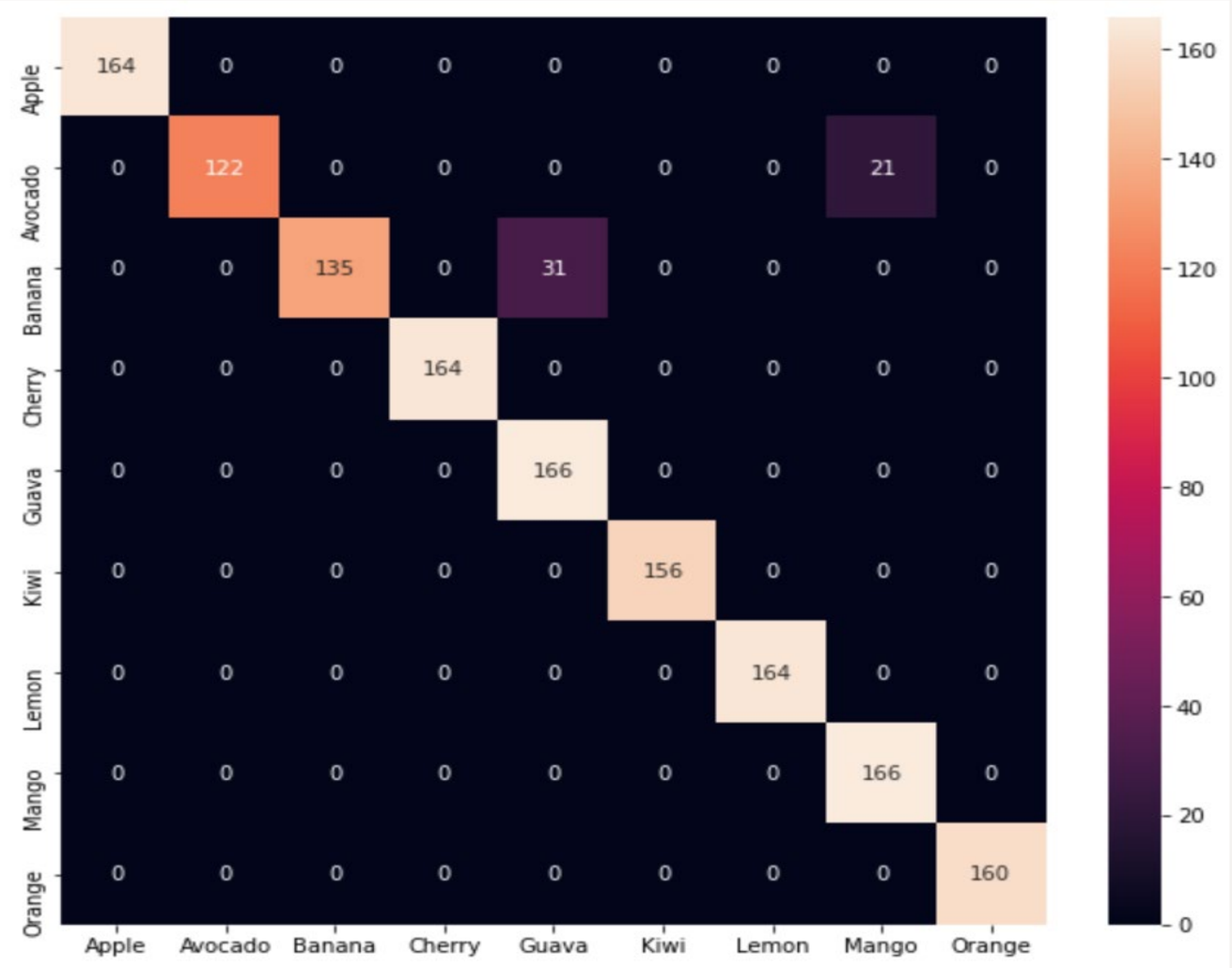


（橫軸表預測水果名稱，縱軸表實際水果名稱）

共1,296張驗證圖片
1,165張準確辨識
131張辨識錯誤

模型校度分析

- 測試集之混淆矩陣



共1,449張測試圖片
1,397張準確辨識
52張辨識錯誤

（橫軸表預測水果名稱，縱軸表實際水果名稱）

參數優化

- 本研究嘗試調整Batch_size、學習率、Layers、激活函數，以取得更高之準確率。（迭代次數Epochs固定為10）

	Batch_size	學習率	Layers	激活函數	準確率
Origin	70	0.25%	2層卷積層（Filters為8、16） 2層池化層	Relu	90%
Test 1	50	0.3%	3層卷積層（Filters為8、16、32） 3層池化層	Relu	73%
Test 2	30	0.2%	4層卷積層（Filters為8、16、32、64） 4層池化層	Relu	88%
Test 3	20	0.1%	3層卷積層（Filters為8、16、32） 3層池化層	Sigmoid	53%
Test 4	15	0.1%	3層卷積層（Filters為8、16、32） 3層池化層	Relu	99%
Test 5	15	0.05%	4層卷積層（Filters為8、16、32、64） 4層池化層	Relu	80%

參數優化結果

- 準確率

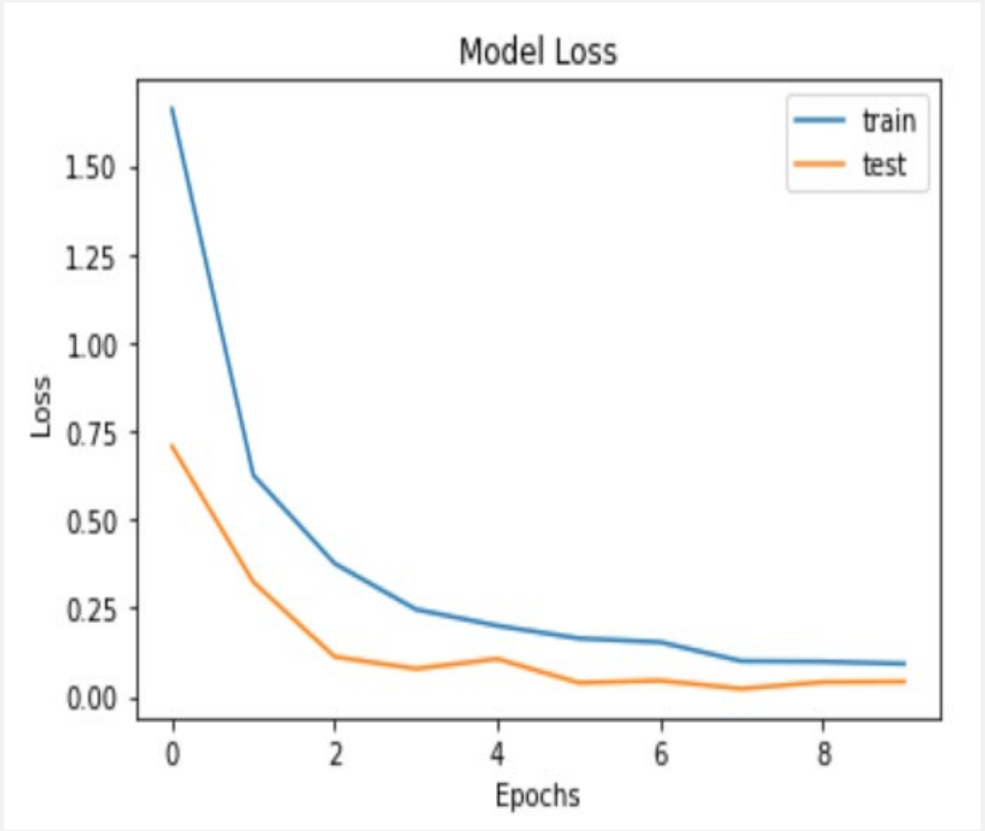
```
train_loss, train_acc = model.evaluate(x_train, y_train, batch_size=batch_size)
test_loss, test_acc = model.evaluate(x_val, y_val, batch_size=batch_size)
print('Train accuracy:', train_acc)
print('Test accuracy:', test_acc)
```

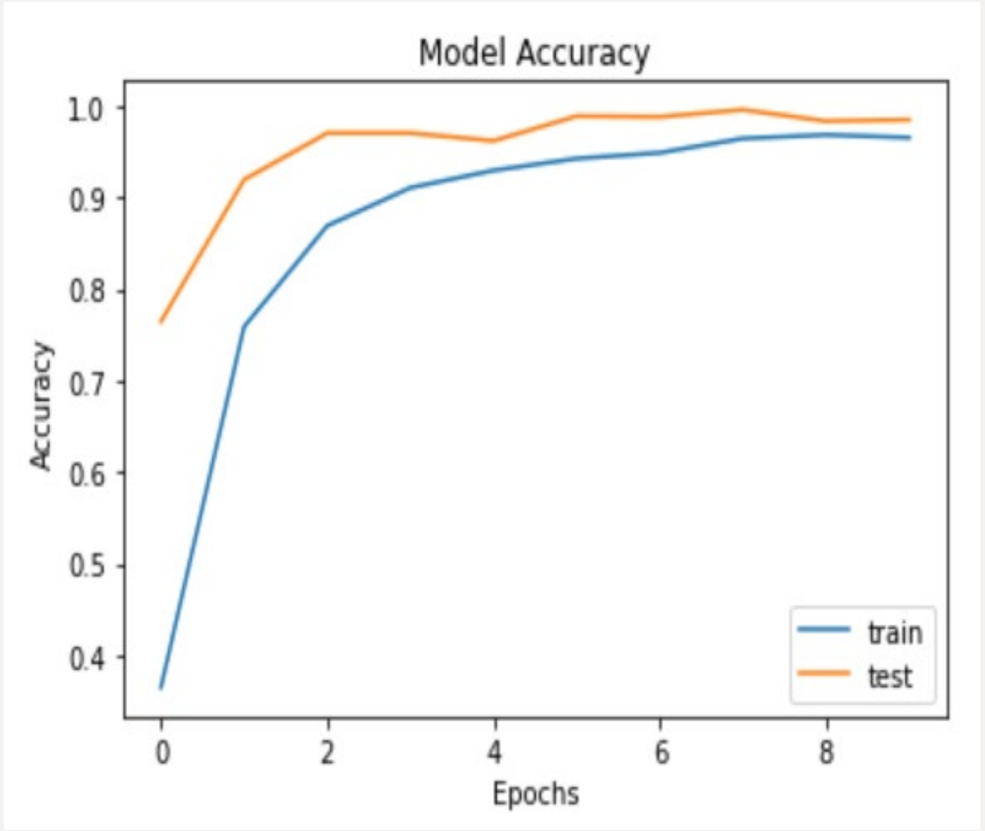
```
202/202 [==============================] - 6s 31ms/step - loss: 0.0387 - accuracy: 0.9871
87/87 [==============================] - 3s 31ms/step - loss: 0.0421 - accuracy: 0.9853
Train accuracy: 0.9870946407318115
Test accuracy: 0.9853395223617554
```

約99%，相較於Origin 提升了9%

參數優化結果

- 繪製Loss、Accuracy與Epochs之關係圖



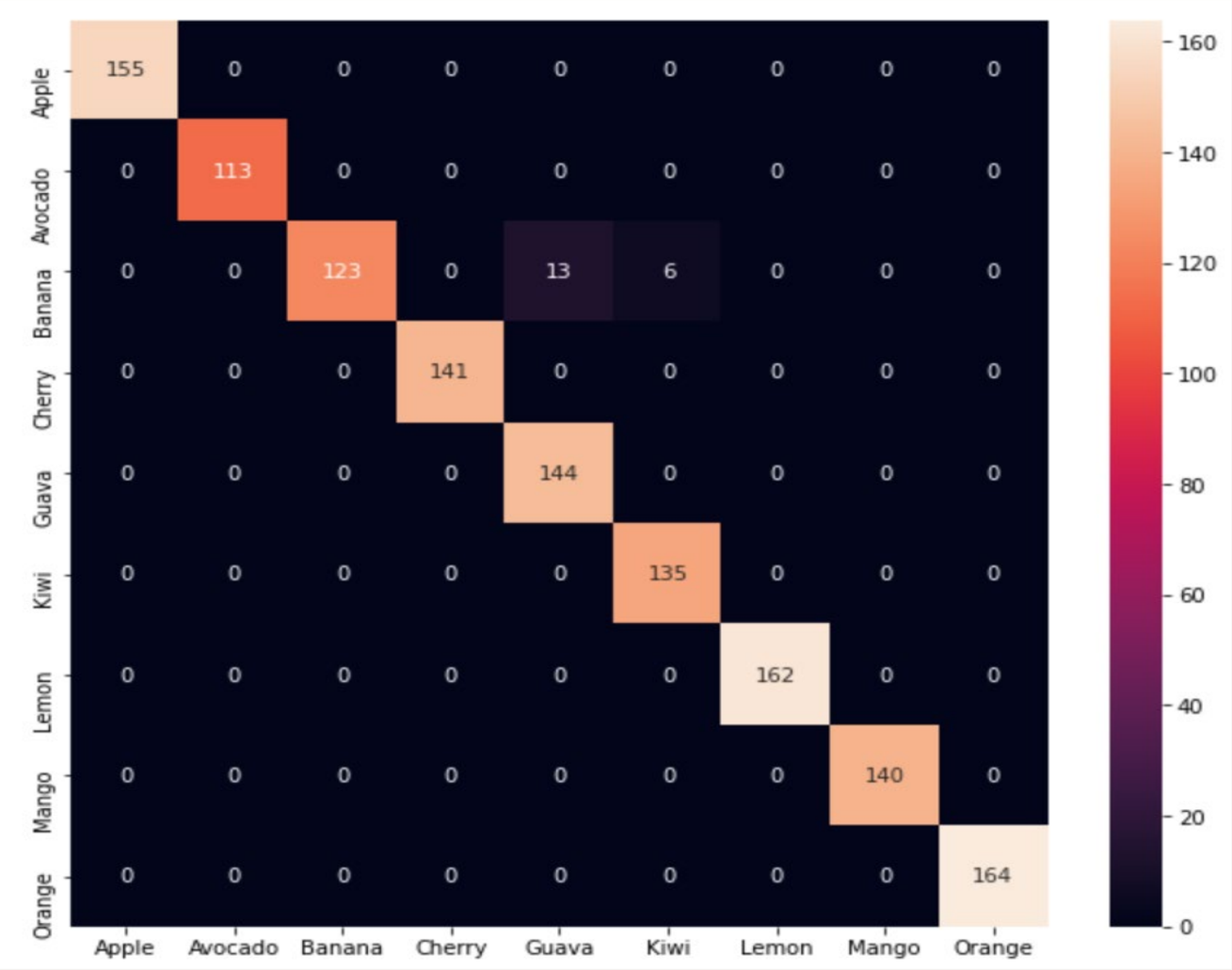


參數優化結果

- 驗證集之混淆矩陣

相較於Origi n，
增加112張準確辨識圖片

共1,296張驗證圖片
1,277張準確辨識
19張辨識錯誤



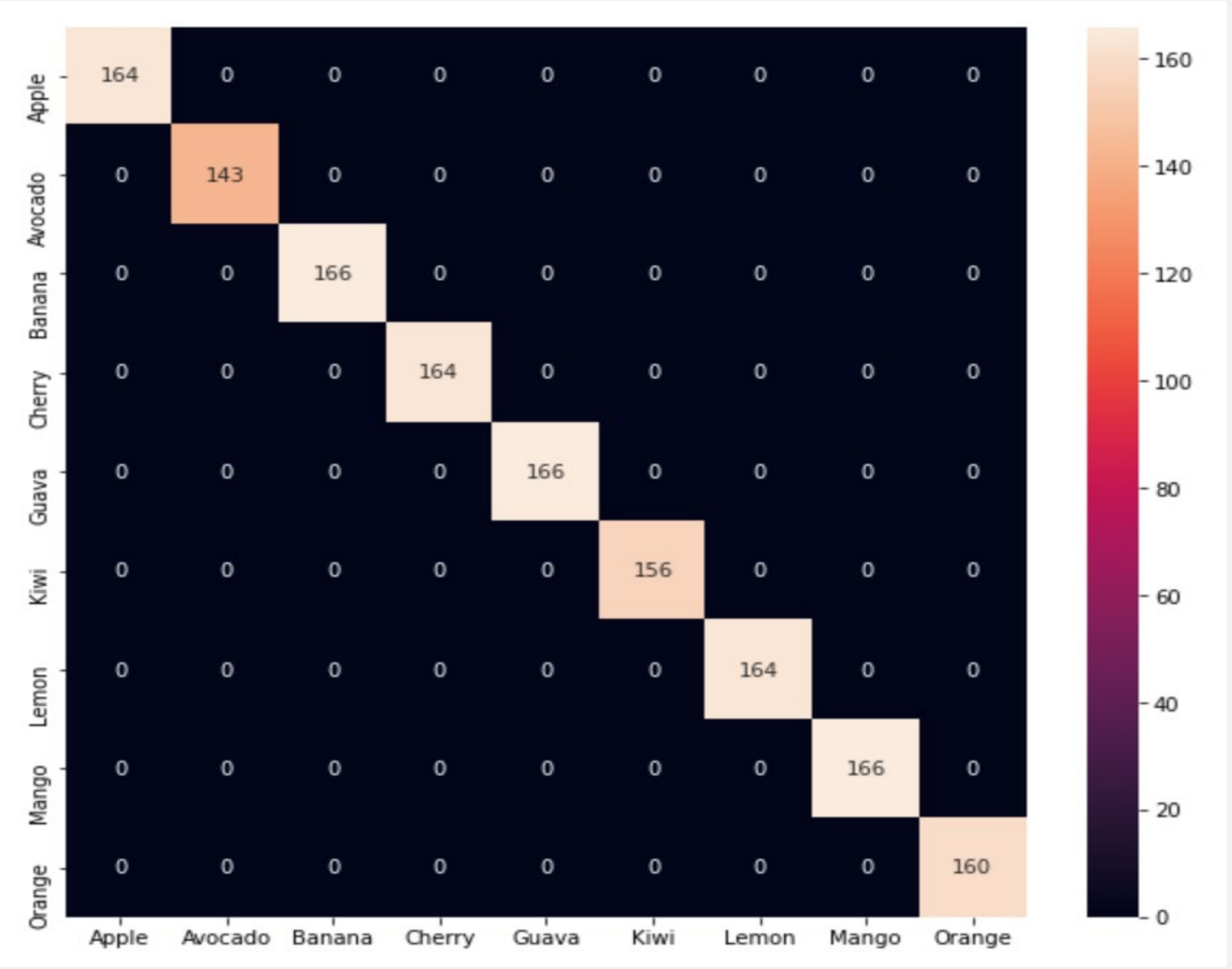
（橫軸表預測水果名稱，縱軸表實際水果名稱）

參數優化結果

- 測試集之混淆矩陣

相較於Origi n，
增加52張準確辨識圖片

共1,449張測試圖片
1,449張準確辨識
0張辨識錯誤



（橫軸表預測水果名稱，縱軸表實際水果名稱）

4

結論
與
未來展望

結論

- 本研究利用CNN 將九種水果進行分類，且透過多次參數調整使得準確率可達99%。
- 本研究欲提供CNN 模型予水果業者或消費者，以協助其辨識水果名稱。
- 本研究之模型可提升水果業者結帳速度並降低人工辨識錯誤率，亦可減少消費者尋找或等待服務人員所耗費的時間。

未來展望

- 本研究之模型僅針對九種水果進行研究，另外還有其他相當多種水果，且一種水果根據品種、熟度可能可再分為多種顏色、表皮紋理。

- 未來可增加更豐富多樣之水果圖像並加以訓練，以建構一個完整的水果辨識模型。

- 未來可利用不同的模型來訓練資料，並透過不同的資料集來驗證成果。

參考資料

- https :// www .kaggle .com/etatbak/cnn -fruit - classification/data?select=fruits -360
- https :// www .kaggle .com/moltean/fruits
- https ://zh .wikipedia .org/wiki/ %E5%8D%B7%E7%A7%AF %E7%A5%9E%E7%BB%8F%E7%BD%91%E7%BB%9C
- https :// ithelp .ithome .com.tw/articles/ 10192028

Thanks!