

The background is a deep blue gradient. On the left side, there is a large, semi-transparent circular scale with degree markings from 40 to 260. Several 3D cubes are scattered across the scene, some appearing to float or move. Faint, glowing lines and dots form a network-like pattern on the right side, suggesting a digital or scientific theme.

自然圖像辨識

07360532 陳威諭

07360726 劉沛綸

探討CNN圖像辨識的深淺層數差異



我的動機何在

製作動機

- 模型架構採用 CNN
- 使用 Google Colab 實作
- 資料集來自 Kaggle

使用工具

- 模型建立
- 訓練模型
- 訓練曲線
- 準確率與損失率
- 預測結果

實作目錄

模型建立

```
[ ] 1 model = Sequential()
    2
    3 model.add(Conv2D(64, (3, 3), padding='same', input_shape=x_train.shape[1:]))
    4 model.add(Activation('relu'))
    5 model.add(BatchNormalization())
    6
    7 model.add(Conv2D(64, (3, 3)))
    8 model.add(Activation('relu'))
    9 model.add(MaxPooling2D(pool_size=(2, 2)))
   10 model.add(BatchNormalization())
   11 model.add(Dropout(0.35))
   12
   13 model.add(Conv2D(64, (3, 3), padding='same'))
   14 model.add(Activation('relu'))
   15 model.add(BatchNormalization())
   16
   17 model.add(Conv2D(64, (3, 3)))
   18 model.add(Activation('relu'))
   19 model.add(MaxPooling2D(pool_size=(2, 2)))
   20 model.add(BatchNormalization())
   21 model.add(Dropout(0.35)) #64 -> 42
   22
   23 model.add(Conv2D(64, (3, 3), padding='same'))
   24 model.add(Activation('relu'))
   25 model.add(BatchNormalization())
   26
   27 model.add(Flatten())
   28 model.add(Dropout(0.5))
   29 model.add(Dense(512))
   30 model.add(Activation('relu'))
   31 model.add(BatchNormalization())
   32 model.add(Dense(8))
   33 model.add(Activation('softmax'))
   34
   35 model.summary()
```

模型建立

```
1 model = Sequential()
2
3 model.add(Conv2D(64, (3, 3), padding='same', input_shape=x_train.shape[1:]))
4 model.add(Activation('relu'))
5 model.add(BatchNormalization())
6
7 model.add(Conv2D(64, (3, 3)))
8 model.add(Activation('relu'))
9 model.add(MaxPooling2D(pool_size=(2, 2)))
10 model.add(BatchNormalization())
11 model.add(Dropout(0.35))
12
13 model.add(Conv2D(64, (3, 3), padding='same'))
14 model.add(Activation('relu'))
15 model.add(BatchNormalization())
16
17 model.add(Conv2D(64, (3, 3)))
18 model.add(Activation('relu'))
19 model.add(MaxPooling2D(pool_size=(2, 2)))
20 model.add(BatchNormalization())
21 model.add(Dropout(0.35)) #64 -> 42
22
23 model.add(Conv2D(64, (3, 3), padding='same'))
24 model.add(Activation('relu'))
25 model.add(BatchNormalization())
26
27 model.add(Conv2D(64, (3, 3), padding='same'))
28 model.add(Activation('relu'))
29 model.add(MaxPooling2D(pool_size=(2, 2)))
30 model.add(BatchNormalization())
31 model.add(Dropout(0.35))
32
33 model.add(Conv2D(64, (3, 3), padding='same'))
34 model.add(Activation('relu'))
35 model.add(BatchNormalization())
36
37 model.add(Conv2D(64, (3, 3), padding='same'))
38 model.add(Activation('relu'))
39 model.add(MaxPooling2D(pool_size=(2, 2)))
40 model.add(BatchNormalization())
41 model.add(Dropout(0.35))
42
43 model.add(Conv2D(64, (3, 3), padding='same'))
44 model.add(Activation('relu'))
45 model.add(BatchNormalization())
46
47 model.add(Flatten())
48 model.add(Dropout(0.5))
49 model.add(Dense(512))
50 model.add(Activation('relu'))
51 model.add(BatchNormalization())
52 model.add(Dense(8))
53 model.add(Activation('softmax'))
54
```

```
[ ] 1 model.compile(
      2     loss='categorical_crossentropy',
      3     optimizer='adam',
      4     metrics=['accuracy'])
```

訓練模型

```
[ ] 1 EPOCHS=25
    2 history = model.fit(x_train, y_train, epochs=EPOCHS, validation_split=0.2)
    3 model.save(drive_path + "5model_" + str(EPOCHS) + ".h5")
```

```
[ ] 1 EPOCHS=100
    2 history = model.fit(x_train, y_train, epochs=EPOCHS, validation_split=0.2)
    3 model.save(drive_path + "5model_" + str(EPOCHS) + ".h5")
```

```
[ ] 1 EPOCHS=25
    2 history = model.fit(x_train, y_train, epochs=EPOCHS, validation_split=0.2)
    3 model.save(drive_path + "9model_" + str(EPOCHS) + ".h5")
```

```
▶ 1 EPOCHS=100
   2 history = model.fit(x_train, y_train, epochs=EPOCHS, validation_split=0.2)
   3 model.save(drive_path + "9model_" + str(EPOCHS) + ".h5")
```

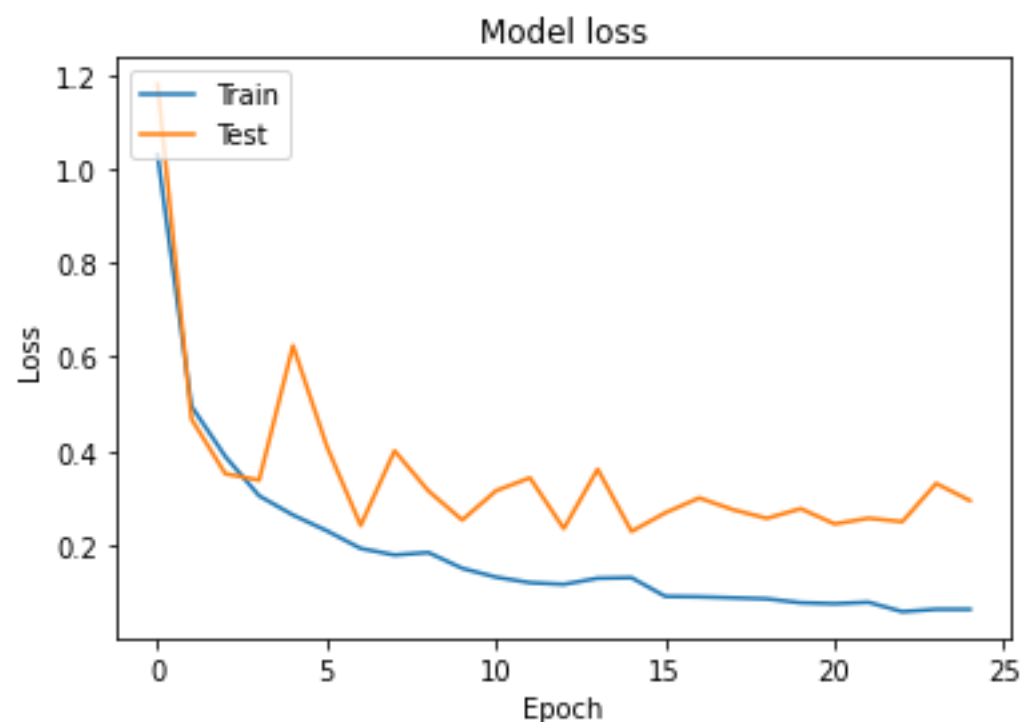
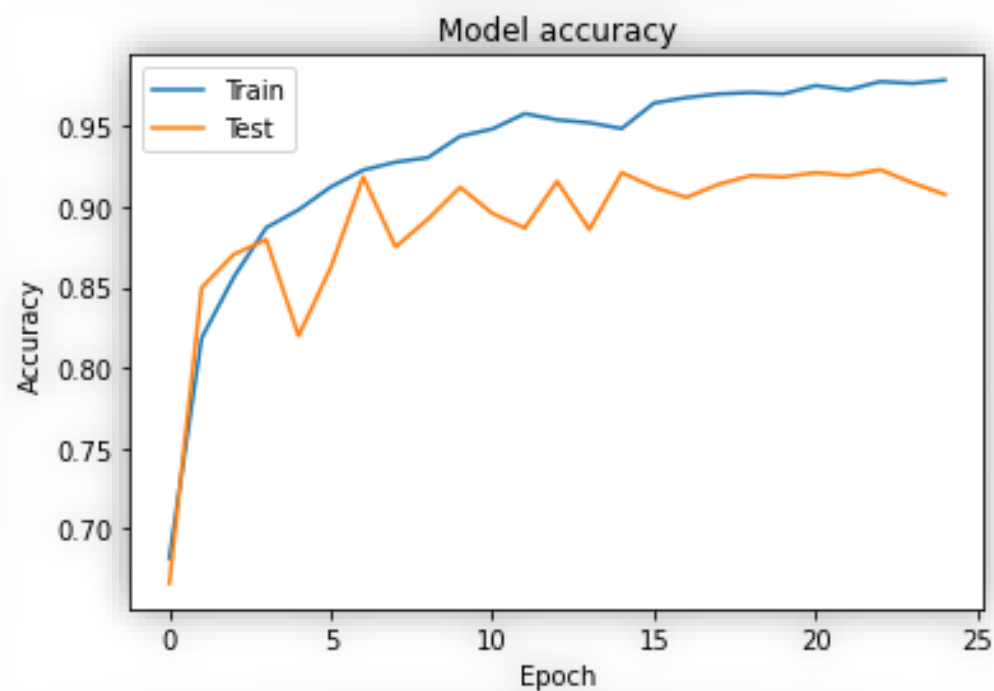
EPOCH 7/100

訓練曲線

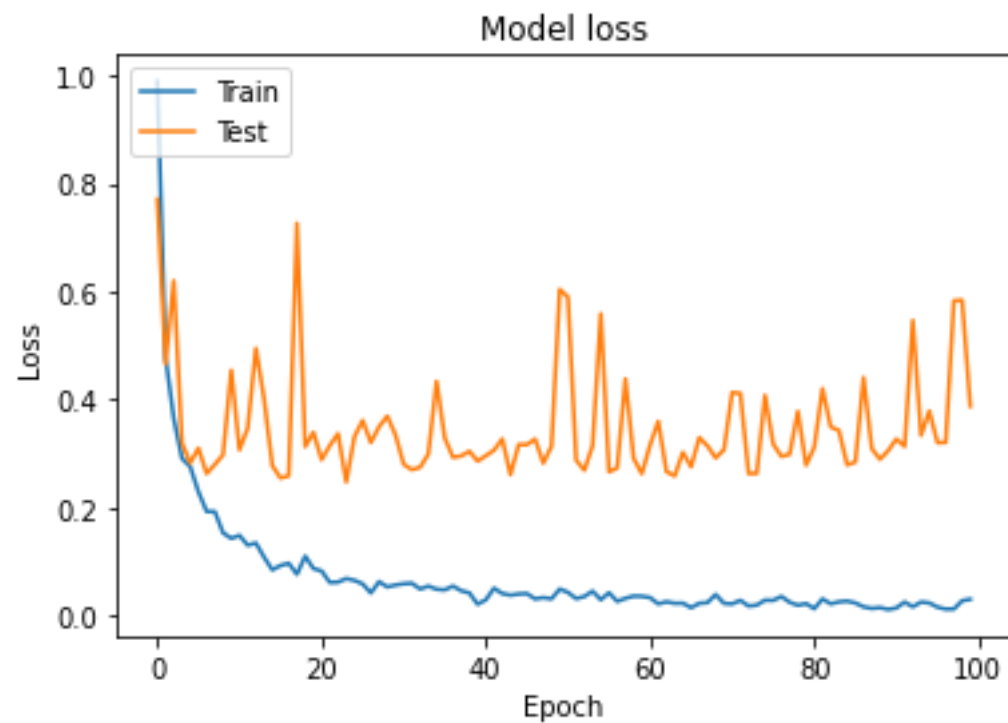
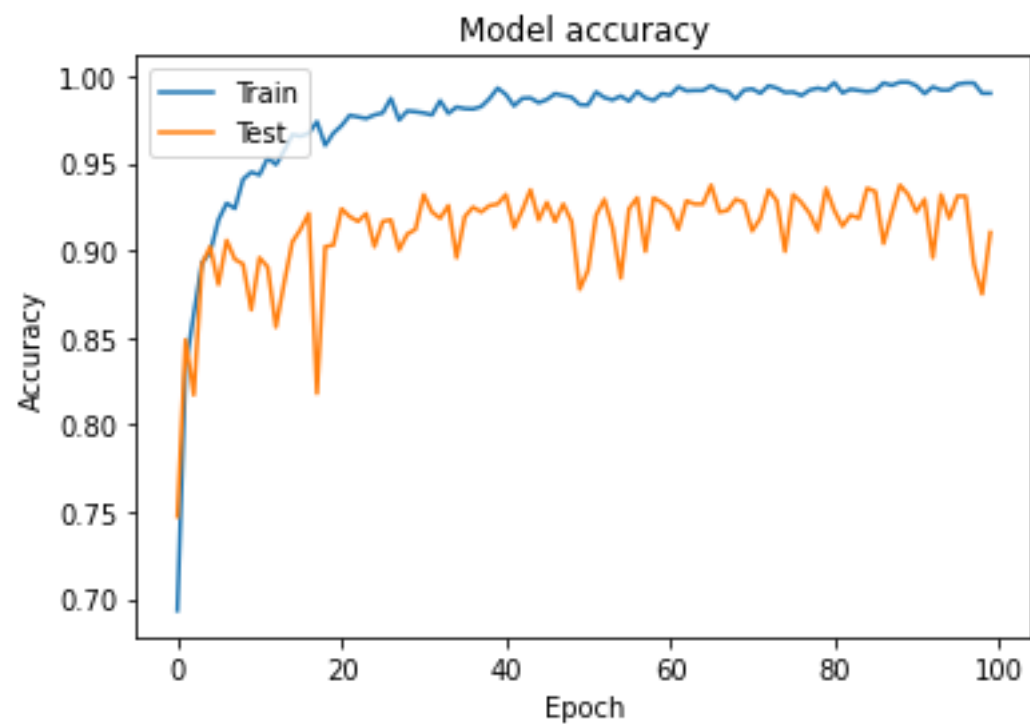


```
1 plt.plot(history.history['accuracy'])
2 plt.plot(history.history['val_accuracy'])
3 plt.title('Model accuracy')
4 plt.ylabel('Accuracy')
5 plt.xlabel('Epoch')
6 plt.legend(['Train', 'Test'], loc='upper left')
7 plt.show()
8
9 plt.plot(history.history['loss'])
10 plt.plot(history.history['val_loss'])
11 plt.title('Model loss')
12 plt.ylabel('Loss')
13 plt.xlabel('Epoch')
14 plt.legend(['Train', 'Test'], loc='upper left')
15 plt.show()
```

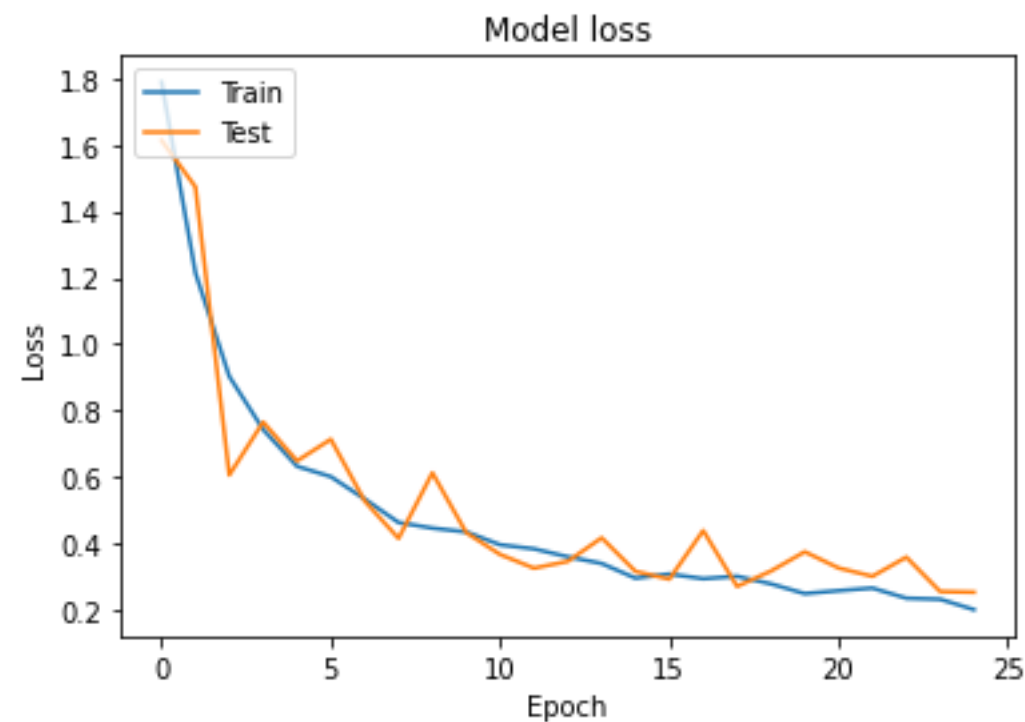
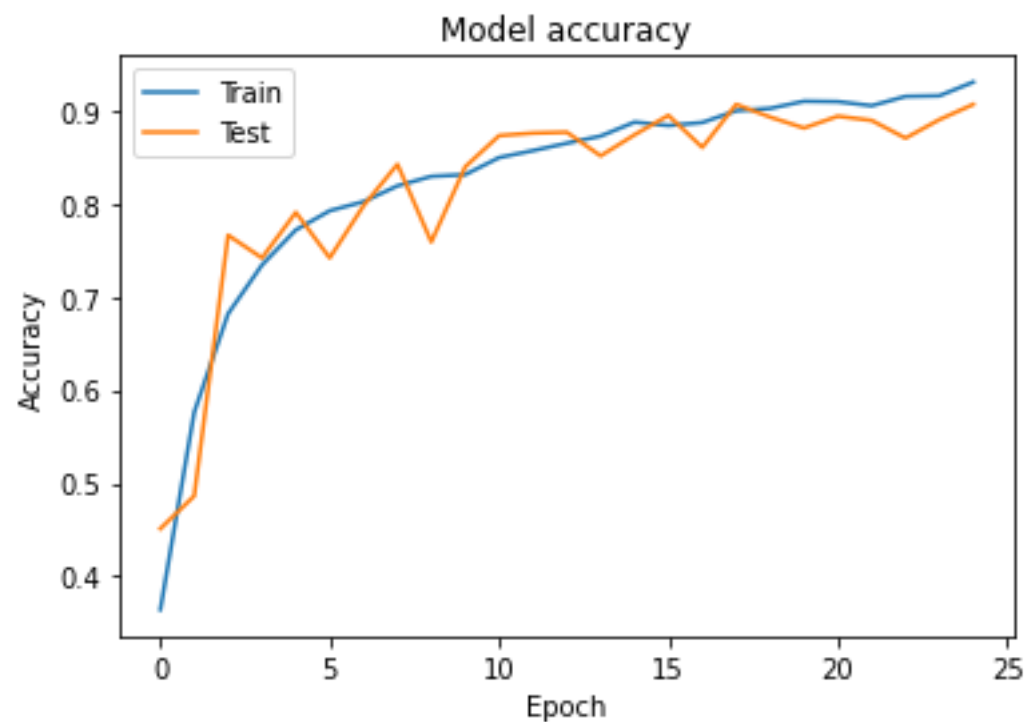

訓練曲線（5層CNN網路 訓練25圈）



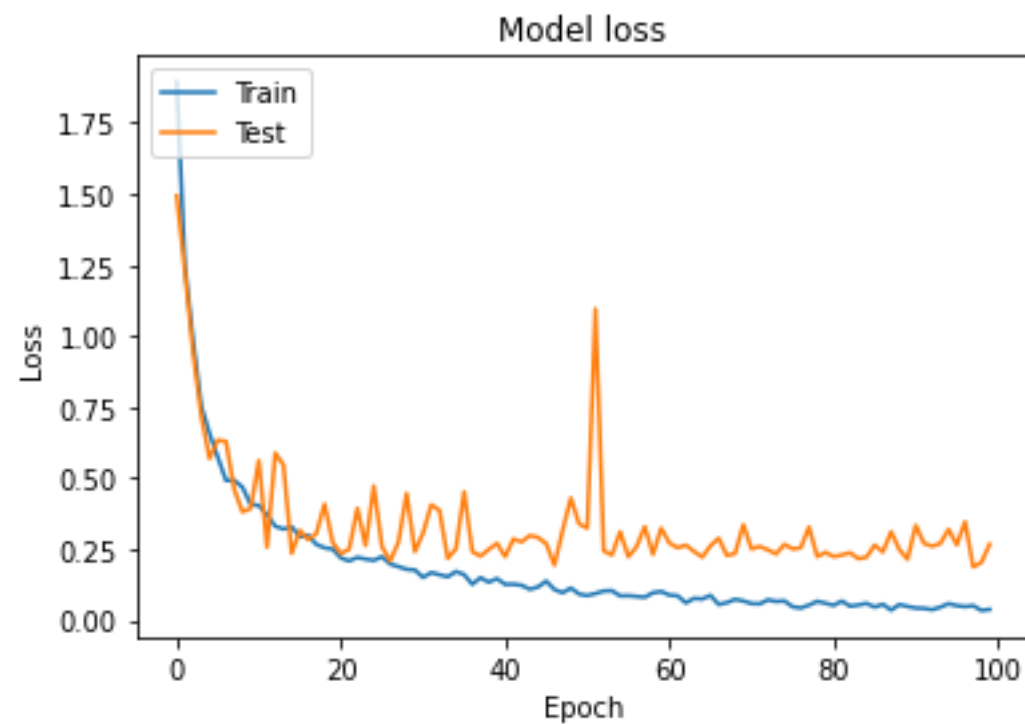
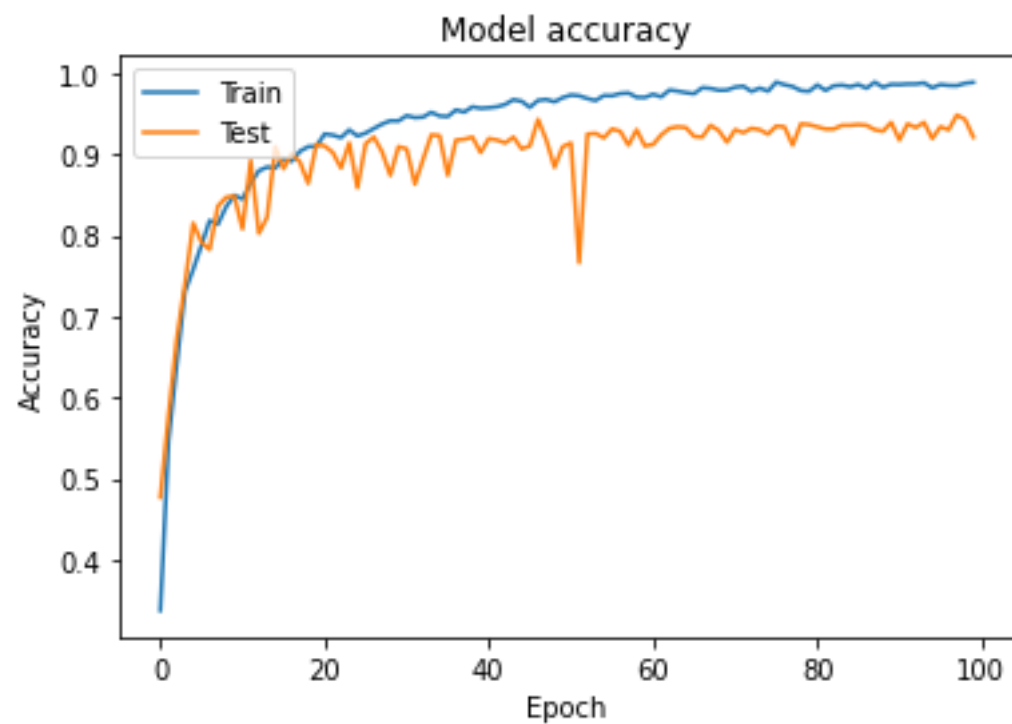
訓練曲線（5層CNN網路 訓練100圈）



訓練曲線（9層CNN網路 訓練25圈）



訓練曲線（9層CNN網路 訓練100圈）



準確率與損失率



```
1 print("train")
2 train_loss_acc = model.evaluate(x_train, y_train)
3 print("test")
4 test_loss_acc = model.evaluate(x_test, y_test)
```

```
train
173/173 [=====] - 2s 9ms/step - loss: 0.0860 - accuracy: 0.9714
test
44/44 [=====] - 0s 10ms/step - loss: 0.3227 - accuracy: 0.8993
```

5層CNN網路 訓練25圈

```
train
173/173 [=====] - 2s 12ms/step - loss: 0.1068 - accuracy: 0.9735
test
44/44 [=====] - 1s 14ms/step - loss: 0.3456 - accuracy: 0.9188
```

5層CNN網路 訓練100圈

```
train
173/173 [=====] - 2s 10ms/step - loss: 0.1319 - accuracy: 0.9514
test
44/44 [=====] - 1s 12ms/step - loss: 0.2157 - accuracy: 0.9174
```

9層CNN網路 訓練25圈

```
train
173/173 [=====] - 3s 15ms/step - loss: 0.0703 - accuracy: 0.9779
test
44/44 [=====] - 1s 19ms/step - loss: 0.2819 - accuracy: 0.9268
```

9層CNN網路 訓練100圈

預測結果

```
[ ] 1 def plot_images_labels_prediction(images, labels, prediction, idx, num=10):
2     fig = plt.gcf()
3     fig.set_size_inches(12,14)                                #設定影像的大小
4     if num>25 : num =25                                       #設定顯示最大項數
5     for i in range(0,num):                                     #for回圈畫出num個數字影像
6         ax=plt.subplot(5,5,i+1)
7         ax.imshow(images[idx], cmap='binary')                 #建立subgraph子圖形為五行五列
8         for j in range(0, len(labels[i])):
9             if labels[i][j] == 1:
10                break
11            title = "label="+label_dict[j]+"\\n"               #設定子圖形title, 顯示標籤欄位
12            for k in range(0, len(prediction[i])):
13                if prediction[i][k] == 1:
14                    break
15            if len(prediction)>0:                                #如果傳入了預測結果
16                title+="prediction="+label_dict[k]             #標題
17
18            ax.set_title(title,fontsize=10)                     #設定子圖形的標題
19            ax.set_xticks([]):ax.set_yticks([])                 #設定不顯示刻度
20            idx+=1                                              #讀取下一項
21        plt.show()
22
23 label_dict={0:"airplane",1:"car",2:"cat",3:"dog",4:"flower",5:"fruit",6:"motorbike",7:"person"}
```

```
[ ] 1 predictions = (model.predict(x_test) > 0.5).astype("int32")
2 plot_images_labels_prediction(x_test,y_test,predictions,0,10)
```

預測結果（5層CNN網路 訓練25圈）

label=flower
prediction=flower



label=dog
prediction=dog



label=airplane
prediction=airplane



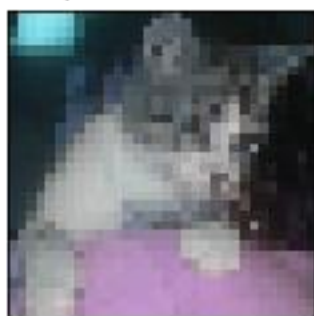
label=person
prediction=person



label=flower
prediction=flower



label=cat
prediction=cat



label=flower
prediction=flower



label=dog
prediction=dog



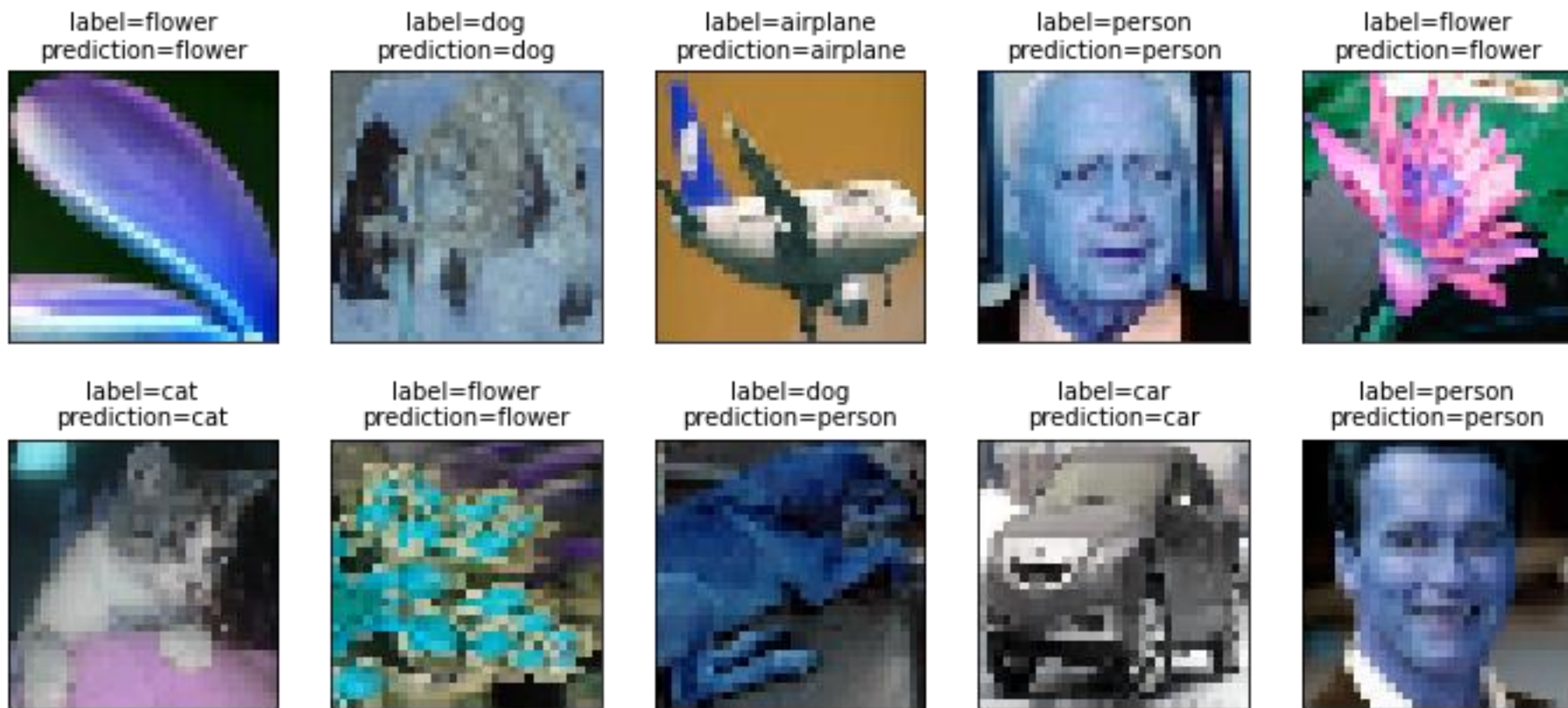
label=car
prediction=car



label=person
prediction=person



預測結果（5層CNN網路 訓練100圈）



預測結果（9層CNN網路 訓練25圈）

label=flower
prediction=flower



label=dog
prediction=dog



label=airplane
prediction=airplane



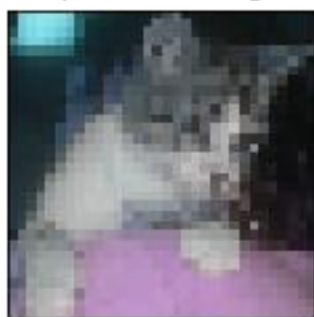
label=person
prediction=person



label=flower
prediction=flower



label=cat
prediction=dog



label=flower
prediction=flower



label=dog
prediction=dog



label=car
prediction=car



label=person
prediction=person



預測結果（9層CNN網路 訓練100圈）

label=flower
prediction=flower



label=dog
prediction=dog



label=airplane
prediction=airplane



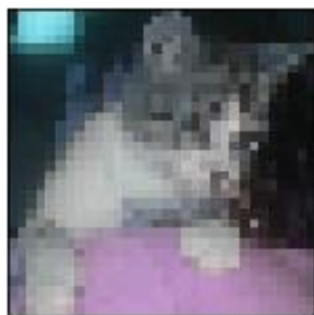
label=person
prediction=person



label=flower
prediction=flower



label=cat
prediction=dog



label=flower
prediction=flower



label=dog
prediction=car



label=car
prediction=car



label=person
prediction=person



測試過相同圈數與不同層數的比較後，我們認為較多層的CNN模型能夠提高準確率。同時發現到在卷基層較多時，8種類別中僅在貓狗辨識的部分出錯，其餘6種類別的辨識效果良好，我們認為是由於增加了池化層後減少了特徵擷取，因此對於較接近的物件上特徵數不足而造成了辨識的困難。

結論

大家一起做

07360532 陳威諭

07360726 劉沛綸

分工表

