人工智慧期末報告

自然圖像辨識

07360532 陳威諭 07360726 劉沛綸

中華民國　一一一年　一　月　十七　日

目錄

動機

探討 CNN 圖像辨識的深淺層數差異

成果展示

```
[ ]  1 labels = os.listdir('/content/drive/MyDrive/natural_images/')
     2 print(labels)

     ['motorbike', 'cat', 'airplane', 'dog', 'fruit', 'car', 'person', 'flower']

     1 x_data =[]
     2 y_data = []
     3 import cv2
     4 for label in labels:
     5         path = '/content/drive/MyDrive/natural_images/{0}/'.format(label)
     6         folder_data = os.listdir(path)
     7         for image_path in folder_data:
     8                 image = cv2.imread(path+image_path)
     9                 image_resized = cv2.resize(image, (32,32))
    10                 x_data.append(np.array(image_resized))
    11                 y_data.append(label)

[ ]  1 x_data = np.array(x_data)
     2 y_data = np.array(y_data)

[ ]  1 #converting the y_data into categorical:
     2 y_encoded = LabelEncoder().fit_transform(y_data)
     3 y_categorical = to_categorical(y_encoded)

[ ]  1 r = np.arange(x_data.shape[0])
     2 np.random.seed(42)
     3 np.random.shuffle(r)
     4 X = x_data[r]
     5 Y = y_categorical[r]

[ ]  1 X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2, random_state=42)
     2 saveTestTrainData(drive_path + "x_train.npy", X_train)
     3 saveTestTrainData(drive_path + "x_test.npy", X_test)
     4 saveTestTrainData(drive_path + "y_train.npy", Y_train)
     5 saveTestTrainData(drive_path + "y_test.npy", Y_test)
```

將資料集切割為訓練資料與測試資料

```
[ ]  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()
```

建立 CNN 模型（5 層）

```
model = Sequential()

model.add(Conv2D(64, (3, 3), padding='same',input_shape=x_train.shape[1:]))
model.add(Activation('relu'))
model.add(BatchNormalization())

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.35))

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(BatchNormalization())

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.35))  #64 --> 42

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(BatchNormalization())

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.35))

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(BatchNormalization())

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.35))

model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(BatchNormalization())

model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(512))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Dense(8))
model.add(Activation('softmax'))
```

建立 CNN 模型（9 層）

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

損失函數選用多分類型的 Categorical Cross Entropy，優化器選用 Adam

```
[ ]  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")
```

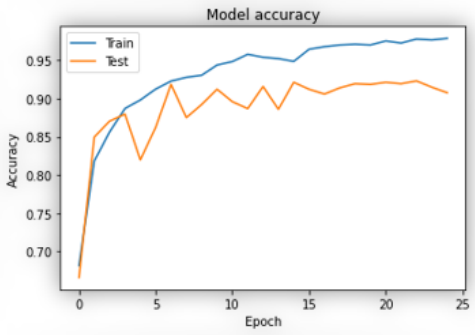
```
     1 EPOCHS=100
     2 history = model.fit(x_train, y_train, epochs=EPOCHS, validation_split=0.2)
     3 model.save(drive_path + "9model_" + str(EPOCHS) + ".h5")
```

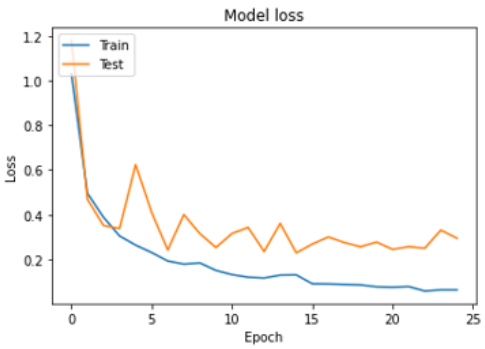
分別將 5 層與 9 層的版本做 25 圈與 100 圈訓練用於比較

```
 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()
```

畫出訓練曲線圖

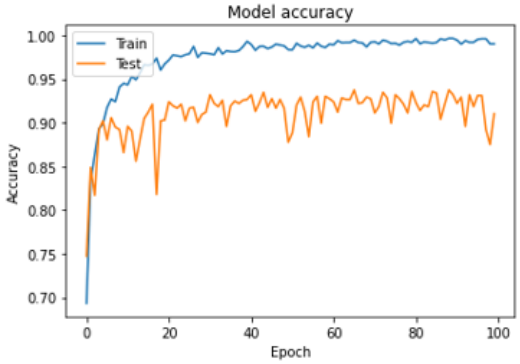
訓練曲線（5層CNN網路 訓練25圈）

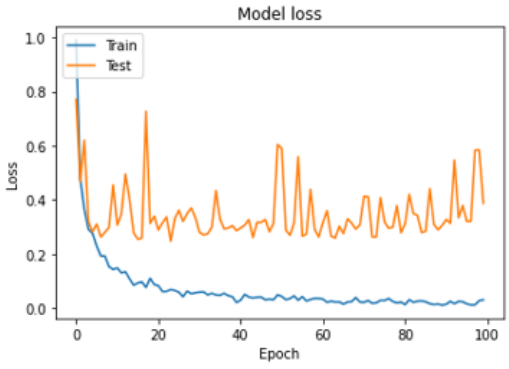





```
train
173/173 [==============================] - 2s 9ms/step - loss: 0.0860 - accuracy: 0.9714
test
44/44 [==============================] - 0s 10ms/step - loss: 0.3227 - accuracy: 0.8993
```

訓練曲線（5層CNN網路 訓練100圈）

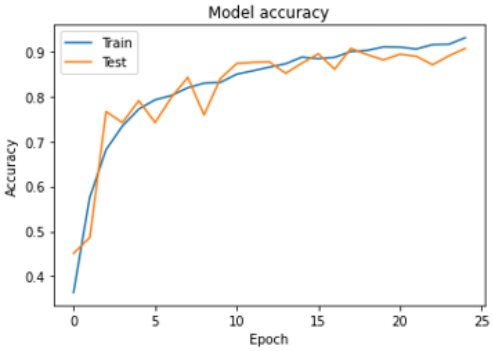


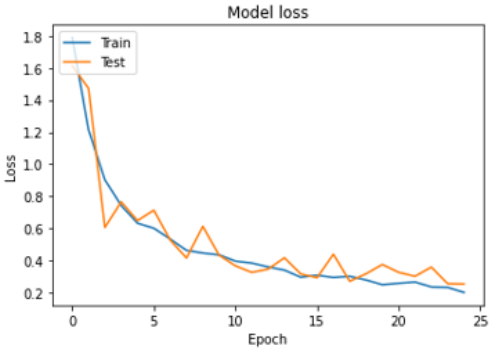



```
train
173/173 [==============================] - 2s 12ms/step - loss: 0.1068 - accuracy: 0.9735
test
44/44 [==============================] - 1s 14ms/step - loss: 0.3456 - accuracy: 0.9188
```

5 層的時候收斂程度很差，過擬和情況嚴重，同時測試準確率低。

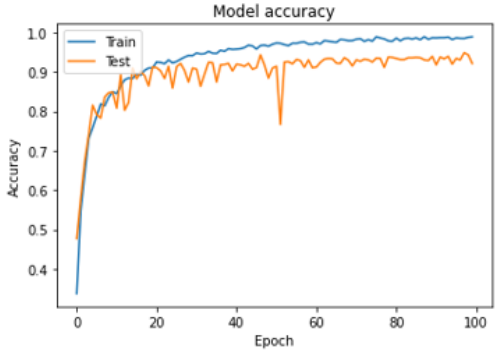
訓練曲線（9層CNN網路 訓練25圈）

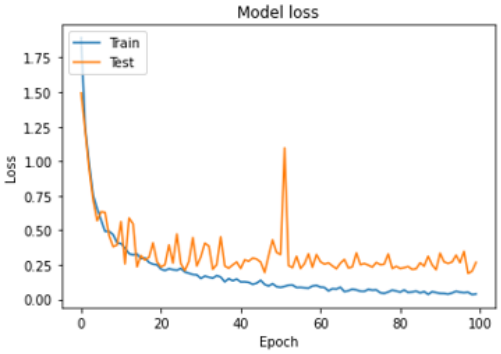





```
train
173/173 [==============================] - 2s 10ms/step - loss: 0.1319 - accuracy: 0.9514
test
44/44 [==============================] - 1s 12ms/step - loss: 0.2157 - accuracy: 0.9174
```

訓練曲線（9層CNN網路 訓練100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
```

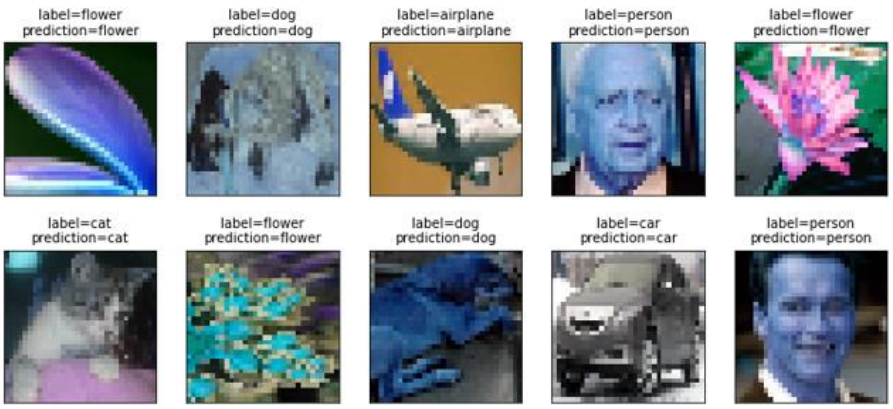
9 層時收斂狀況佳，同時測試準確率較高。

```
[ ]  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，顯示標簽欄位
    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)
```

印出預測結果

預測結果（5層CNN網路 訓練25圈）



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）

預測結果（9層CNN網路 訓練100圈）

資料集中前 10 張圖片在各個模型中的預測結果

心得

陳威諭：

這次的課程很有趣也很實用，透過實作讓我知道了不同的學習得使用方式以及其中的差別，同時在期末專題中的過程，讓我對於不同的學習也有更多的了解，是一堂非常好的課程。

劉沛綸：

這次的人工智慧讓我學到了許多東西，在專研時我們也是在做人工智慧相關的研究，不過我們對於人工智慧的瞭解僅限於題目，不太懂機器學習以及深度學習之間的關係，這次的課程讓我對於這兩種學習方式有了了解，並且也知道了要如何使用，非常的有趣且實用。

參考資料

1. Natural Images 檢自 url:

   https://www.kaggle.com/prasunroy/natural-images?fbclid=IwAR2RkYDKQyTB7e1QqhnvX4G2304yeZWzfGF-c0JaN4LLqzR_EEyLXrl0hNo, 2022/01

2. natural image classification 檢自 url:

   https://www.kaggle.com/yasinsoylu123/natural-image-classification?fbclid=IwAR2RkYDKQyTB7e1QqhnvX4G2304yeZWzfGF-c0JaN4LLqzR_EEyLXrl0hNo, 2022/01