Deep Learning 圖像辨識-Flower Recognition

個人 Project
學生:楊恆軒
指導老師:邱銘傳老師
2019/01/01

壹、摘要

利用 CNN Keras 對收集到的 Dataset 進行花的種類辨別，種類有: 1.向日葵 2.鬱金香 3.雛菊 4.玫瑰 5.蒲公英。

貳、簡介&文獻綜述

作為一個新手 Deep Learning 玩家，正因為能力有限所以上網尋找了許多文章，發現圖像辨識最為簡單，因此去了 Kaggle 網站，此網站為提供一大堆資料集而讓網站上的人可以利用這些資料集去做一些機器學習或深度學習的競賽。而我剛好看上了一個收集好的 Flowers Recognition Dataset，裡面分別分類了 ：

1. 雛菊 ：769 張照片。
2. 蒲公英 ：1055 張照片。
3. 玫瑰 ：784 張照片。
4. 向日葵 ：734 張照片。
5. 鬱金香 ：984 張照片。

參、研究方法

利用 Deep Learning Neural Networks 對蒐集到 Dataset 進行花朵種類辨別，透過 Tensorflow Keras API 建立並訓練模型。

肆、案例研究

一、使用工具和套件

1. 首先先安裝 OpenCV，OpenCV 是一個基於 BSD 許可（開源）發行的跨平台計算機視覺庫，可以運行在 Linux、Windows、Android 和 Mac OS 作業系統上。同時提供了 Python、Ruby、MATLAB 等語言的接口，實現了圖像處理和計算機視覺方面的很多通用算法。

```
# OpenCV
!apt-get -qq install -y libsm6 libxext6 && pip install -q -U opencv-python
```

2. Tensorflow： 一個開源軟體庫，用於各種感知和語言理解任務的機器學習。

3. Keras： 一個開放原始碼，高階深度學習程式庫，使用 Python 編寫，能夠運行在 TensorFlow 或 Theano 之上。

4. Python os： 提供了一個統一的操作系統接口函數, 這些接口函數通常是平台指定的，os 模塊能在不同操作系統平台（如 nt 或 posix）中的特定函數間自動切換,從而能實現跨平台操作。

5.Python numpy： 提供維度陣列與矩陣的運算。

6. Pythonmatplotlib:是 Python 程式語言及其數值數學擴展包 numPy 的可視化操作界面。

7. Python pandas： 提供數據操作和分析的 data frame 結構。

8.Python random： 返回隨機生成的一個實數，它在[0,1)範圍內。

```
# TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import datasets, layers, models

# Helper libraries
import os
from os import listdir
from os.path import join
import numpy as np
import matplotlib.pyplot as plt
import cv2
from google.colab import drive
import pandas
import random
```

9. 授權讓他使用我的 Google drive。

```
# Accessing My Google Drive
drive.mount('/content/drive')
```

10. 設定路徑和定位圖片位置和設置輸入文件夾的路徑，設定完畢後觀看裡面的資料夾，顯示 dandelion、tulip、daisy、sunflower、rose 可以確定資料夾的路徑設定沒有問題。

```
# defining global variable path
# Location of my dataset on My Google Drive
image_path = 'drive/My Drive/flowers'

def loadImages(path):
    '''Put files into lists and return them as one list with all images
     in the folder'''
    image_files = sorted([os.path.join(path, 'train', file)
                          for file in os.listdir(path + '/train')
                          if file.endswith('.png')])
    return image_files
```

```
# Set the path of the input folder
data = 'drive/My Drive/flowers'

# List out the directories inside the main input folder
folders = os.listdir(data)
print(folders)
```

```
['dandelion', 'tulip', 'daisy', 'sunflower', 'rose']
```

二、資料前處理

1. 將全部圖片調整為 64*64 尺寸後在逐一給上自己的標籤。

```
# Import the images and resize them to a 64*64 size
# Also generate the corresponding labels

image_names = []
train_labels = []
train_images = []

size = 64,64

for folder in folders:
    for file in os.listdir(os.path.join(data,folder)):
        if file.endswith("jpg"):
            image_names.append(os.path.join(data,folder,file))
            train_labels.append(folder)
            img = cv2.imread(os.path.join(data,folder,file))
            im = cv2.resize(img,size)
            train_images.append(im)
        else:
            continue
```

2. 將圖片轉換成矩陣，可以看到總共 4323 張圖片，尺寸為 64*64，因為圖片皆為彩色圖片所以維度為 3，確認沒問題後將圖片全部除以 255 以標準化。

```
# Transform the image array to a numpy type
train = np.array(train_images)
train.shape
```

```
(4323, 64, 64, 3)
```

```
# Reduce the RGB values between 0 and 1
train = train.astype('float32') / 255.0
```

3. 提取標籤並給定編號，可以看到 dandelio 為 1、tulip 為 4、daisy 為 0、sunflower 為 3、rose 為 2。

```
# Extract the labels
label_dummies = pandas.get_dummies(train_labels)
labels = label_dummies.values.argmax(1)
```

```
pandas.unique(train_labels)
```

```
array(['dandelion', 'tulip', 'daisy', 'sunflower', 'rose'], dtype=object)
```

```
pandas.unique(labels)
```

```
array([1, 4, 0, 3, 2])
```

4. 隨機打亂圖像與編號以獲得更好的效果，在將改組後的列表轉換為 numpy 的數組類型。

```
# Shuffle the labels and images randomly for better results
union_list = list(zip(train, labels))
random.shuffle(union_list)
train,labels = zip(*union_list)

# Convert the shuffled list to numpy array type
train = np.array(train)
labels = np.array(labels)
```

三、模型建立

1. 發展模型，我使用的模型為 Sequential；第一層 Layer 的 filters 為 32，activation 為 relu；第二層 Layer 的 filters 增加至 64；第三層 Layer 的 filters 最後設定為 128，而 activation 皆設為 relu，原先我使用的 activation 為 sigmoid，但是

發現 relu 的效果比較好，這也通過不斷改變參數得到的結果。

```
# Develop a sequential model
model = models.Sequential()

# 1st Convolutional Layer
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)))
model.add(layers.MaxPooling2D((2, 2)))

# 2nd Convolutional Layer
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

# 3rd Convolutional Layer
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))


model.add(layers.Flatten())
model.add(layers.Dropout(0.3))
# Add output layer
model.add(layers.Dense(5, activation='softmax'))
```

2. 設定模型參數，我使用的 optimizer 為 adam，目標準確率為判讀圖片跟標籤有沒有吻合。

```
# Compute the model parameters

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

3. 訓練模型並計算準確率，epochs 為 5

```
# Train the model  with 5 epochs
model.fit(train,labels, epochs=5)
```

```
Train on 4323 samples
Epoch 1/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 1.2667 - acc: 0.4541
Epoch 2/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 1.0652 - acc: 0.5732
Epoch 3/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.9460 - acc: 0.6336
Epoch 4/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.8532 - acc: 0.6699
Epoch 5/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.8130 - acc: 0.6863
<tensorflow.python.keras.callbacks.History at 0x7fa85f9fdcf8>
```

```
 #get score acording to train datas
print("Test Accuracy: {0:.2f}%".format(model.evaluate(train,labels)[1]*100))
```

```
4323/4323 [==============================] - 9s 2ms/sample - loss: 0.8098 - acc: 0.6799
Test Accuracy: 67.99%
```

此準確率為訓練第一次的結果，但是隨著訓練次數上升準確率會逐漸提升，下圖為訓練第二次的結果。

```
# Train the model  with 5 epochs
model.fit(train,labels, epochs=5)
```

```
Train on 4323 samples
Epoch 1/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.7325 - acc: 0.7263
Epoch 2/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.6915 - acc: 0.7395
Epoch 3/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.6125 - acc: 0.7717
Epoch 4/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.5951 - acc: 0.7731
Epoch 5/5
4323/4323 [==============================] - 29s 7ms/sample - loss: 0.5297 - acc: 0.8048
<tensorflow.python.keras.callbacks.History at 0x7fa85f9ddba8>
```

```
 #get score acording to train datas
print("Test Accuracy: {0:.2f}%".format(model.evaluate(train,labels)[1]*100))
```

```
4323/4323 [==============================] - 9s 2ms/sample - loss: 0.4237 - acc: 0.8436
Test Accuracy: 84.36%
```

四、加入新圖片

為了確認我的模型如何，我上網搜尋了 10 張新的圖片並用已經建立好的模型訓練，看看他的預測結果如何。

1. 確認成功載入 10 張圖片。

```
test = np.array(test_images)
test.shape
```

```
(10, 64, 64, 3)
```

2. 加入模型訓練，epochs = 5，訓練過後計算準確率為 50%，第一次訓練就有 50%的準確率，所以我的模型應該算是訓練得不錯吧。

```
model.fit(test,labels, epochs=5)
```

```
Train on 10 samples
Epoch 1/5
10/10 [==============================] - 0s 7ms/sample - loss: 6.6935 - acc: 0.3000
Epoch 2/5
10/10 [==============================] - 0s 6ms/sample - loss: 5.3688 - acc: 0.3000
Epoch 3/5
10/10 [==============================] - 0s 7ms/sample - loss: 4.3871 - acc: 0.3000
Epoch 4/5
10/10 [==============================] - 0s 6ms/sample - loss: 3.3309 - acc: 0.3000
Epoch 5/5
10/10 [==============================] - 0s 6ms/sample - loss: 2.5127 - acc: 0.3000
<tensorflow.python.keras.callbacks.History at 0x7fb64814c5c0>
```

```
print("Test Accuracy: {0:.2f}%".format(model.evaluate(test,labels)[1]*100))
```

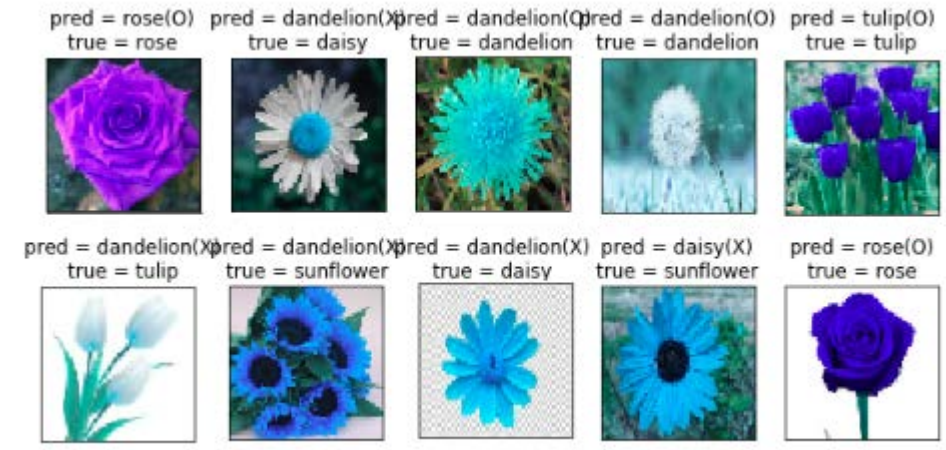
```
10/10 [==============================] - 0s 2ms/sample - loss: 1.8837 - acc: 0.5000
Test Accuracy: 50.00%
```

3. 載入工具 math，並用 plot 把預測結果和類別顯現出來，為了方便觀看，我特地標示了(O)&(X) ，而從結果來看可以知道蒲公英、雛菊、向日葵是最容易搞混的，所以若要改善準確率應該可以再多蒐集一些圖片，加強模型的精準度。。

```
import matplotlib.pyplot as plot
import math
def show_test_label_prediction( test, labels, predictions, indexList ) :

    num = len(indexList)
    plot.gcf().set_size_inches( 2*5, (2+0.4)*math.ceil(num/5) )

    loc = 0
    for i in indexList :
      loc += 1
      subp = plot.subplot( math.ceil(num/5), 5, loc )
      subp.imshow( test[i], cmap='binary' )
      if( len(predictions) > 0 ) :
            title = 'pred = ' + label_desc[ predictions[i] ]
            title += ('(O)' if predictions[i]==labels[i] else '(X)')
            title += '\ntrue = ' + label_desc[ labels[i] ]
      else :
            title = 'true = ' + label_desc[ labels[i] ]
      subp.set_title( title, fontsize=12 )
      subp.set_xticks( [] )
      subp.set_yticks( [] )
plot.show()
show_test_label_prediction( test, labels, prediction, range(0, 10) )
```



伍、結論

通過這次的 Project 讓我學到很多，不管是 Python 也好又或者是處理方法，經過這次歷練相信我自己的實力又有更進一步的提升，當然還有很多不足的地方可以讓我去學習的，這次的花朵種類分辨只有 5 個類別，在第一次訓練就有將近 68%的準確率可以說是相當不錯的，倘若我能自己搜尋更多圖片加到資料集裡面相信準確率可以有更高的提升模型也能夠更準確穩定。

未來可以改進的地方的話，我在網路上有看到有人可以結合手機，透過照相

然後及時上傳圖片，在讓你的模型辨別這是什麼種類的花朵；又或者是多增加一些花朵的種類，但是鑒於時間跟實力問題，這次我就沒有辦法做到像網路上這幾位大神一樣這麼厲害，不過我以前也沒有接觸過這些東西，對於我自己能做到這樣我已經非常滿足。

陸、參考資料

1. Kaggle 的資料集
https://www.kaggle.com/alxmamaev/flowers-recognition
2.授權使用 Google drive 的方法
https://datascience.stackexchange.com/questions/29480/uploading-images-folder-from-my-system-into-google-colab
3. Python 處理圖像錯誤的方法
https://blog.csdn.net/TwT520Ly/article/details/75040048
4.別人使用此資料集的方法
https://www.kaggle.com/rajmehra03/flower-recognition-cnn-keras/notebook
5.顯示圖片結果
https://tomkuo139.blogspot.com/2018/05/aicnn-cifar10.html