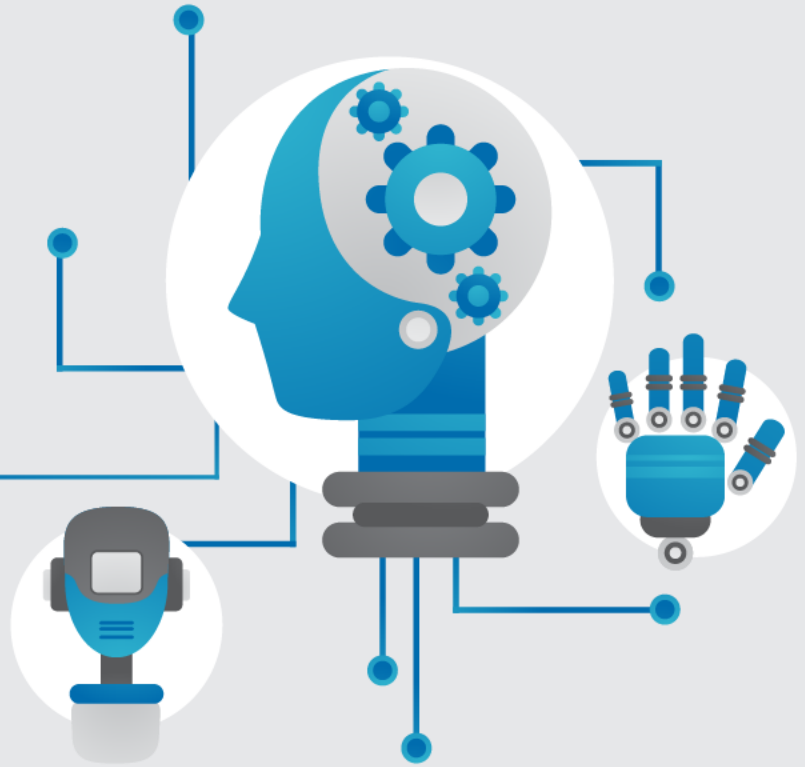
CNN模型架構



CNN模型的程式架構

模型
訓練

樣本
預測

模型訓練程式基本架構

參數設定

- batch size 設為128（使訓練速度快一點）
- number of classes 設定為10，因為有10個阿拉伯數字
- Epochs 設為20，代表資料訓練20回

```
10 batch_size = 128
11 num_classes = 10
12 epochs = 20
```

模型訓練模組 - 載入函示庫

- 載入 keras 模組
- 載入 keras 資料集中的 mnist 範例資料集
- 載入 keras 的循序模型（Sequential model）
- 載入 keras 的全連接層（Dense）、Flatten層
- 載入 keras 的卷積層（Conv2D）、池化層（MaxPooling2D）

```
1 # coding: utf-8
2
3 # 載入函式庫
4 import keras
5 from keras.models import Sequential
6 from keras.datasets import mnist
7 from keras.layers import Dense, Flatten
8 from keras.layers import Conv2D, MaxPooling2D
9
```

資料讀取和前處理

- 使用 reshape 將輸入資料轉換成二維陣列
- 將數值除以255進行正規化（所有值都介於0～1）
- 使用 to_categoricall 將類別數字轉換成獨熱編碼

```
14 # 下載mnist範例資料檔分兩部分：訓練資料集x_train, y_train（標記）和測試資料集x_test, y_test
15 (x_train, y_train), (x_test, y_test) = mnist.load_data()
16
17 # 資料格式化和正規化
18 x_train = x_train.reshape(60000, 28, 28, 1)
19 x_test = x_test.reshape(10000, 28, 28, 1)
20 x_train = x_train.astype('float32')
21 x_test = x_test.astype('float32')
22 x_train /= 255
23 x_test /= 255
24 print(x_train.shape[0], 'train samples')
25 print(x_test.shape[0], 'test samples')
26
27 # 類別轉換成onehot encoding
28 y_train = keras.utils.to_categorical(y_train, num_classes)
29 y_test = keras.utils.to_categorical(y_test, num_classes)
```

建立模型架構

- 使用 Keras 的循序模型（Sequential）為主要框架，循序模型是多個網路層的線性堆疊
- 依序加入卷積層、池化層、Flatten、全連接層和輸出層

```
31 # 模型架構
32 model = Sequential()
33 # 輸入卷積層
34 model.add(Conv2D(filters=16, kernel_size=(5,5), padding='same',
35                   input_shape=(28,28,1), activation='relu'))
36 # 池化層
37 model.add(MaxPooling2D(pool_size=(2, 2)))
38 # Flatten層
39 model.add(Flatten())
40 # 全連接層
41 model.add(Dense(128, activation='relu'))
42 # 輸出層
43 model.add(Dense(10,activation='softmax'))
44 # 模型架構摘要
45 print(model.summary())
```

模型架構摘要


```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 16)        416
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 16)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3136)              0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               401536
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1290
=================================================================
Total params: 403,242
Trainable params: 403,242
Non-trainable params: 0
_________________________________________________________________
```

- 卷積層參數個數 416 = (5 x 5 + 1) x 16
- dense_1層參數個數 401536 = (3136 + 1) x 128
- 輸出層參數個數 1290 = (128 + 1) x 10

模型訓練

- 在 Keras 中，model.compile 主要完成損失函數和優化器的配置
- model.fit 用來訓練模型，其中參數verbose用來控制是否要記錄訓練歷程，validation_split 是指切割訓練資料部分比例做為驗證（validation）用。

```
47 # 模型訓練
48 model.compile(loss='categorical_crossentropy',
49                               optimizer='adam',
50                               metrics=['accuracy'])
51
52 train_history=model.fit(x_train, y_train,
53                               batch_size=batch_size
54                               epochs=epochs,
55                               verbose=1,
56                               validation_split=0.2)
```

模型評估

- 在 Keras 中，model.evaluate 函數可用來測試模型的效率。其中的參數 x_test 和 y_test 分別是測試資料集和標記。

- model.evaluate 會回傳損失值和準確率。

```
53 # 模型評估
54 score = model.evaluate(x_test, y_test, verbose=0)
55 print('Test loss:', score[0])
56 print('Test accuracy:', score[1])
```

模型儲存

- 在Keras中，呼叫 model.save_weights 函數
  將訓練好的模型的參數記錄下來，供未來預測使用

```
58 # 儲存模型
59 try:
60     model.save_weights("mnist.h5")
61     print("success")
62 except:
63     print("error")
```

樣本預測程式基本架構

1.載入函示庫

2.建立模型架構

3.載入模型

4.讀取樣本

5.樣本前處理

6.樣本預測

樣本預測模組 - 載入函示庫

- 載入資料儲存用的模組numpy和繪圖用的模組matplotlib
- 載入python影像處理的模組 PIL
- 載入keras的循序模型（Sequential model）和卷積層、池化層、Flatten、全連接層

```
3 # 載入函式庫
4 import numpy as np
5 import matplotlib.pyplot as plt
6
7 from PIL import Image
8
9 from keras.models import Sequential
10 from keras.layers import Dense,Dropout,Flatten,Conv2D,MaxPooling2D
```

樣本預測模組 - 建立模型架構

- 要與訓練模組的模型架構相同

```
# 模型架構
model = Sequential()
model.add(Conv2D(filters=16, kernel_size=(5,5),
                 padding='same',input_shape=(28,28,1),
                 activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(10,activation='softmax'))
```

樣本預測模組 - 載入模型

- 呼叫 model.load_weights 函數從模型檔案中載入模型參數

```
20 # 載入模型
21 try:
22     model.load_weights("mnist.h5")
23     print("success")
24 except:
25     print("error")
```

樣本預測模組 - 讀取樣本

- 呼叫 Image 的 open() 讀取照片，convert (“L”) 將照片轉換成灰階，然後存成 nympy.array 的格式。

```
33 # 讀取樣本
34 img=np.array(Image.open('test.jpg').convert('L'))
```

樣本預測模組 - 樣本前處理

- 轉換成二維陣列，並且進行正規化

```
29 # 樣本前處理
30 x_Test4D = img.reshape(1,28,28,1).astype('float32')
31 x_Test4D_normalize = x_Test4D.astype('float32') / 255.0
```

樣本預測模組 - 樣本預測

- model.predict 輸出所有類別的機率，
  而model.predict_classes 則只輸出最有可能的類別編號

```
33 # 樣本預測
34 prediction=model.predict(x_Test4D_normalize)
35 print(prediction[0])
36
37 prediction=model.predict_classes(x_Test4D_normalize)
38 print(prediction[0])
```

- Output

```
1
[7.7321546e-22 1.0000000e+00 7.2617972e-16 2.2812512e-19 2.1435967e-12
 8.0995460e-23 1.3506902e-18 1.5143822e-13 3.5736862e-15 9.6129612e-20]
```