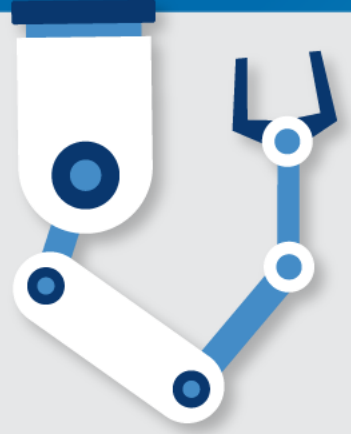
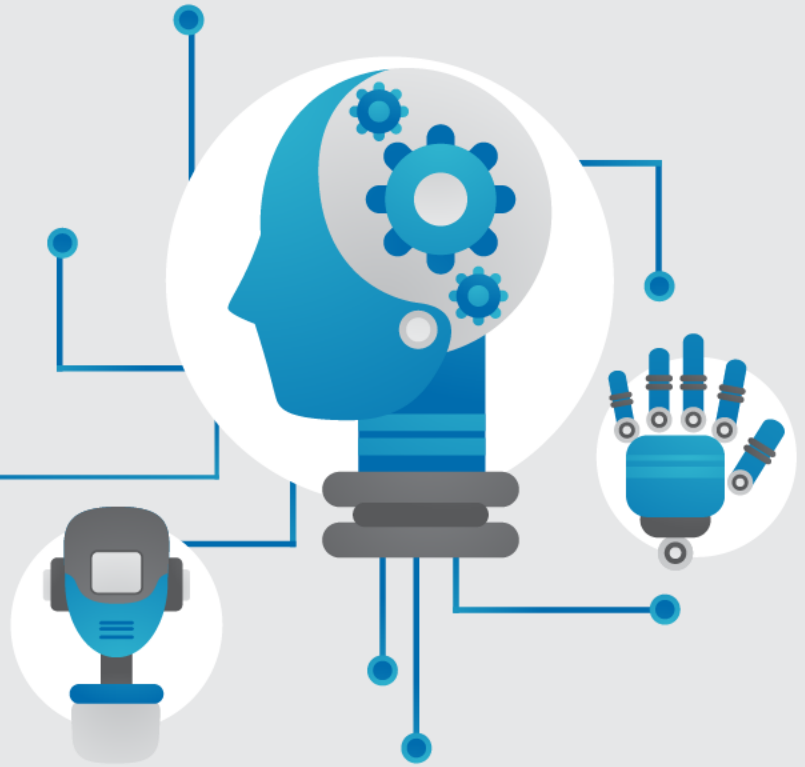


MLP模型架構



MLP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

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）
- 載入keras的優化器RMSprop

```
1 # coding=utf-8
2
3 # 載入函示庫
4 import keras
5 from keras.datasets import mnist
6 from keras.models import Sequential
7 from keras.layers import Dense
8 from keras.optimizers import RMSprop
```

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
```

資料讀取和前處理

- 使用reshape將輸入資料轉換成一維陣列
- 將數值除以255進行正規化（所有值都介於0～1）
- 使用to_categorical將類別數字轉換成獨熱編碼

```
# 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
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# 資料格式化和正規化
x_train = x_train.reshape(60000, 784)
x_test = x_test.reshape(10000, 784)
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
print(x_train.shape[0], 'train samples')
print(x_test.shape[0], 'test samples')

# 類別轉換成onehot encoding
y_train = keras.utils.to_categorical(y_train, num_classes)
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。
- 依序加入輸入層、隱藏層（一個全連接層）和一個輸出層

```
31 # 模型架構
32 model = Sequential()
33 # 輸入層
34 model.add(Dense(512, activation='relu', input_shape=(784,)))
35 # 隱藏層
36 model.add(Dense(512, activation='relu'))
37 # 輸出層
38 model.add(Dense(num_classes, activation='softmax'))
39 # 模型架構摘要
40 model.summary()
```

模型架構摘要


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 512)               401920
_________________________________________________________________
dense_2 (Dense)              (None, 512)               262656
_________________________________________________________________
dense_3 (Dense)              (None, 10)                5130
=================================================================
Total params: 669,706
Trainable params: 669,706
Non-trainable params: 0
_________________________________________________________________
```

第一層參數個數 401920 = (784 + 1) × 512

第二層參數個數 262656 = (512 + 1) × 512

第三層參數個數 5130 = (512 + 1) × 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，而validation_split是指切割訓練資料部分比例做為驗證（validation）用。

```
42 # 模型訓練
43 model.compile(loss='categorical_crossentropy',
44                optimizer=RMSprop(),
45                metrics=['accuracy'])
46
47 train_history = model.fit(x_train, y_train,
48                           batch_size=batch_size,
49                           epochs=epochs,
50                           verbose=1,
51                           validation_split=0.2)
```

模型評估

- 在Keras中，model.evaluate函數可以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

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）和全連接層（Dense）

```
3 # 載入函示庫
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
```

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

```
14 # 模型架構
15 model = Sequential()
16 model.add(Dense(512, activation='relu', input_shape=(784,)))
17 model.add(Dense(512, activation='relu'))
18 model.add(Dense(num_classes, 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numpy 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

- 轉換成一維陣列，並且進行正規化

```
37 # 樣本前處理
38 x_Test = img.reshape(1,784).astype('float32')
39 x_Test_normalize = x_Test.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

```
43 # 樣本預測
44 prediction=model.predict(x_Test_normalize)
45 print(prediction[0])
46
47 prediction=model.predict_classes(x_Test_normalize)
48 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]
```