



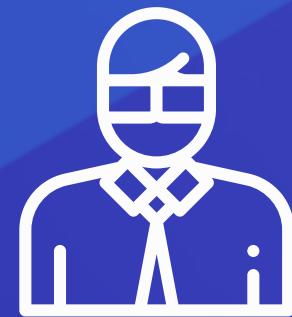
Day 86

初探深度學習使用 Keras

訓練神經網路的細節與技巧

使用 callbacks

函數儲存 model



游為翔

出題教練

知識地圖 深度學習訓練技巧

使用 callbacks 函數儲存 model

深度神經網路

Supervised Learning Deep Neural Network (DNN)

簡介 Introduction

套件介紹 Tools: Keras

組成概念 Concept

訓練技巧 Training Skill

應用案例 Application

卷積神經網路

Convolutional Neural Network (CNN)

簡介 introduction

套件練習 Practice with Keras

訓練技巧 Training Skill

電腦視覺 Computer Vision

深度學習訓練技巧

Training Skill of DNN

應注意的關鍵

防止過擬合 (Overfitting)

超參數 (Hyper-parameters)

學習率 (Learning Rate) 調整

相關訓練技巧

正規化
Regularization

批次標準化
Batch Normalization

回呼
Callback

隨機移除
Drop out

客製化損失函數
Customized Loss Function

提前終止
Early Stopping

本日知識點目標

- 了解如何在訓練過程中，保留最佳的模型權重
- 知道如何在 Keras 中，加入 ModelCheckPoint

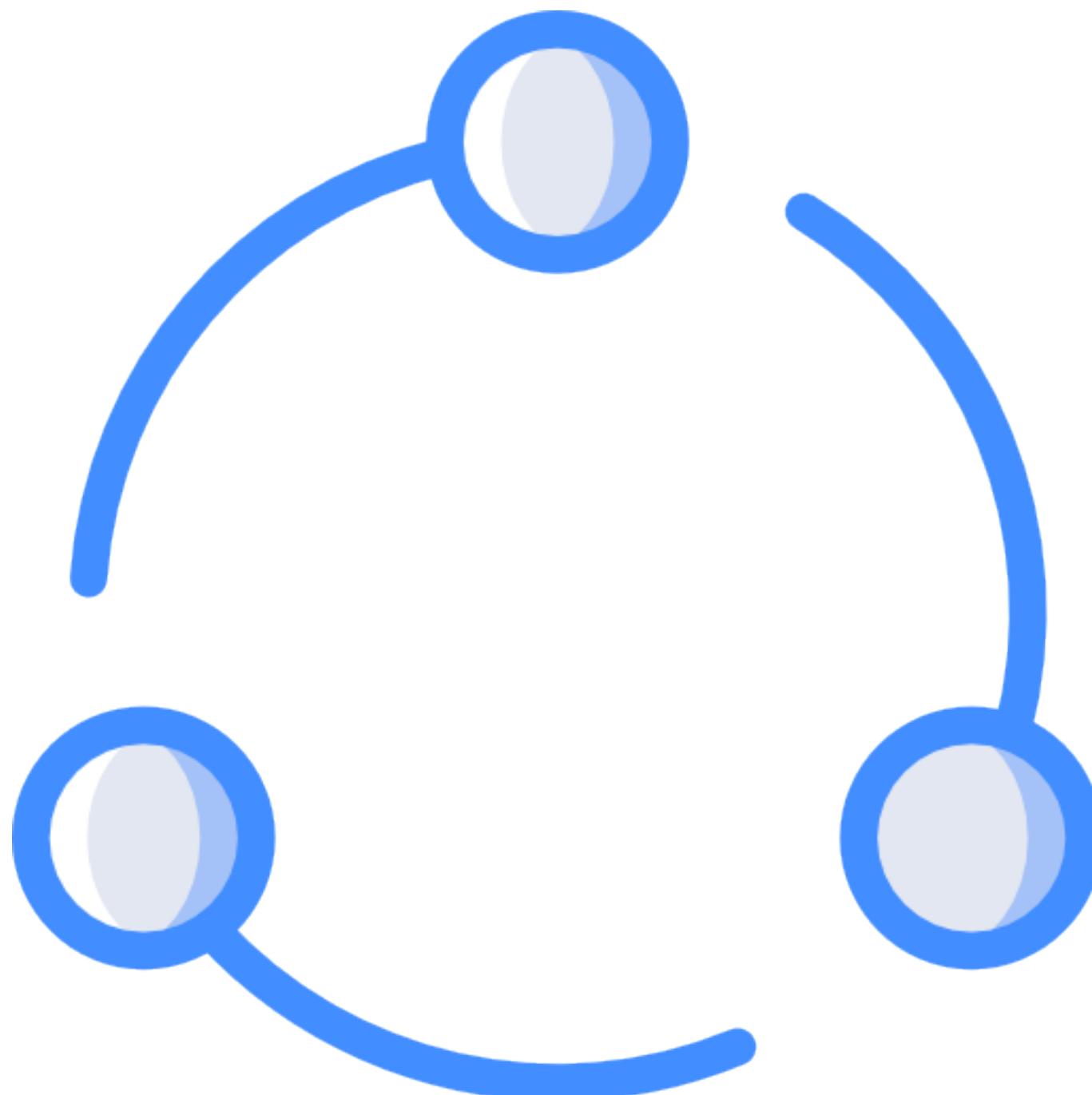
- 為何要使用 Model Check Point?
 - ModelCheckPoint : 自動將目前最佳的模型權重存下
- 假如電腦突然斷線、當機該怎麼辦？難道我只能重新開始？
 - 假如不幸斷線：可以重新自最佳的權重開始
 - 假如要做 Inference :可以保證使用的是對 monitor metric 最佳的權重

ModelCheckPoint in Keras

```
from keras.callbacks import ModelCheckpoint

checkpoint = ModelCheckpoint('model.h5', # path to save
                            monitor = 'val_loss', # target to monitor
                            verbose = 1, # print information
                            save_best_only = True, # save best checkpoint
                            )
model.fit(x_train, y_train,
          epochs=EPOCHS,
          batch_size=BATCH_SIZE,
          validation_data=(x_test, y_test),
          shuffle=True,
          callbacks=[checkpoint]
          )
```

重要知識點複習：



- Model checkpoint：根據狀況隨時將模型存下來，如此可以保證：
 - 假如不幸訓練意外中斷，前面的功夫不會白費。我們可以從最近的一次繼續重新開始。
 - 我們可以透過監控 validation loss 來保證所存下來的模型是在 validation set 表現最好的一個。



延伸 閱讀

除了每日知識點的基礎之外，推薦的延伸閱讀能補足學員們對該知識點的了解程度，建議您解完每日題目後，若有
多餘時間，可再補充延伸閱讀文章內容。

推薦延伸閱讀

莫煩 Python - 儲存與載回模型

● 儲存模型：前面的課程內容已經提過

- 載回模型：
 - 儲存整個模型的話
-> `keras.models.load_model(path_to_your_model)`

● 只儲存權重的話

- > `model = build_model(...)`
- > `model.load_weights(path_to_your_weight_file)`

● 參考連結

导入模型并应用

导入保存好的模型，再执行一遍预测，与之前预测的结果比较，可以发现结果是一样的。

```
# load
model = load_model('my_model.h5')
print('test after load: ', model.predict(X_test[0:2]))
#####
test after load:  [[ 1.87243938] [ 2.20500779]]
#####
```

另外还有其他保存模型并调用的方式，第一种是只保存权重而不保存模型的结构。

```
# save and load weights
model.save_weights('my_model_weights.h5')
model.load_weights('my_model_weights.h5')
```

第二种是用 `model.to_json` 保存完结构之后，然后再去加载这个 `json_string`。

```
# save and load fresh network without trained weights
from keras.models import model_from_json
json_string = model.to_json()
model = model_from_json(json_string)
```



解題時間

It's Your Turn

請跳出PDF至官網Sample Code & 作業
開始解題

