ANN

CH03 TensorFlow與Keras介紹

1

本章將介紹 TensorFlow 與 Keras 的概念與程式設計模式。TensorFlow 功能強大、執行效率高、支援各種平台。然而是屬於低階的深度學習程式庫，學習門檻高。所以本書先介紹 Keras 是高階的深度學習程式庫，對初學者學習門檻低，可以很容易地建立深度學習模型，並且進行訓練、預測。等讀者熟悉深度學習模型概念後，再來學習 TensorFlow，就比較輕鬆了。

- 深度學習的核心概念
  - 使用張量(矩陣)運算，模擬神經網路

- TensorFlow的設計目的就是讓矩陣運算達到最高效能，並能在不同平台執行
- TensorFlow是由Google開發，進行研究與產品開發
  - Gmail過濾垃圾信件、語音辨識、Google翻譯等
- 於2015年開放原始碼
  - 開源軟體(open source software, OSS)
  - 希望建立共同標準，擴展深度學習應用

3.1 TensorFlow 架構

TensorFlow 是較低階的應用編程介面 (Application Programming Interface, API)，需自行設計操作模型

對Python支援最好

分散式運算執行，縮短訓練時間

+雲端伺服器執行

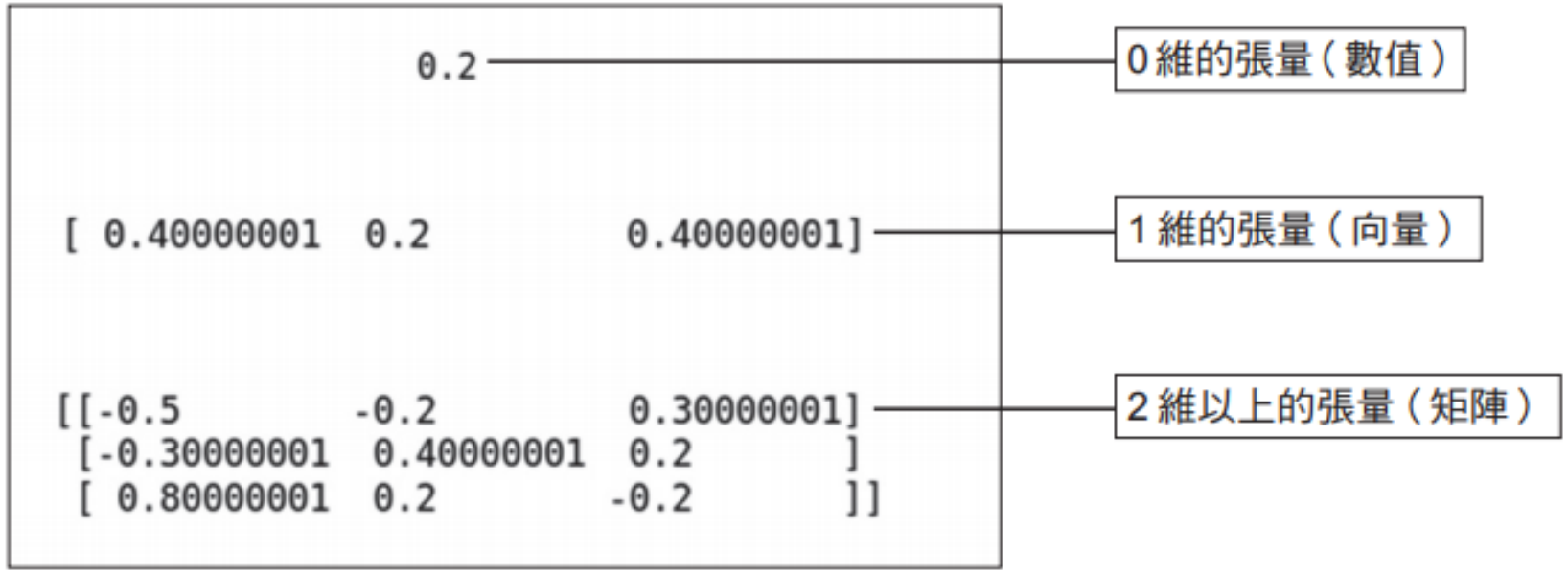
CPU：中央處理器
GPU：圖形處理器
TPU(Tensor Processing Unit)

3.2 TensorFlow 簡介

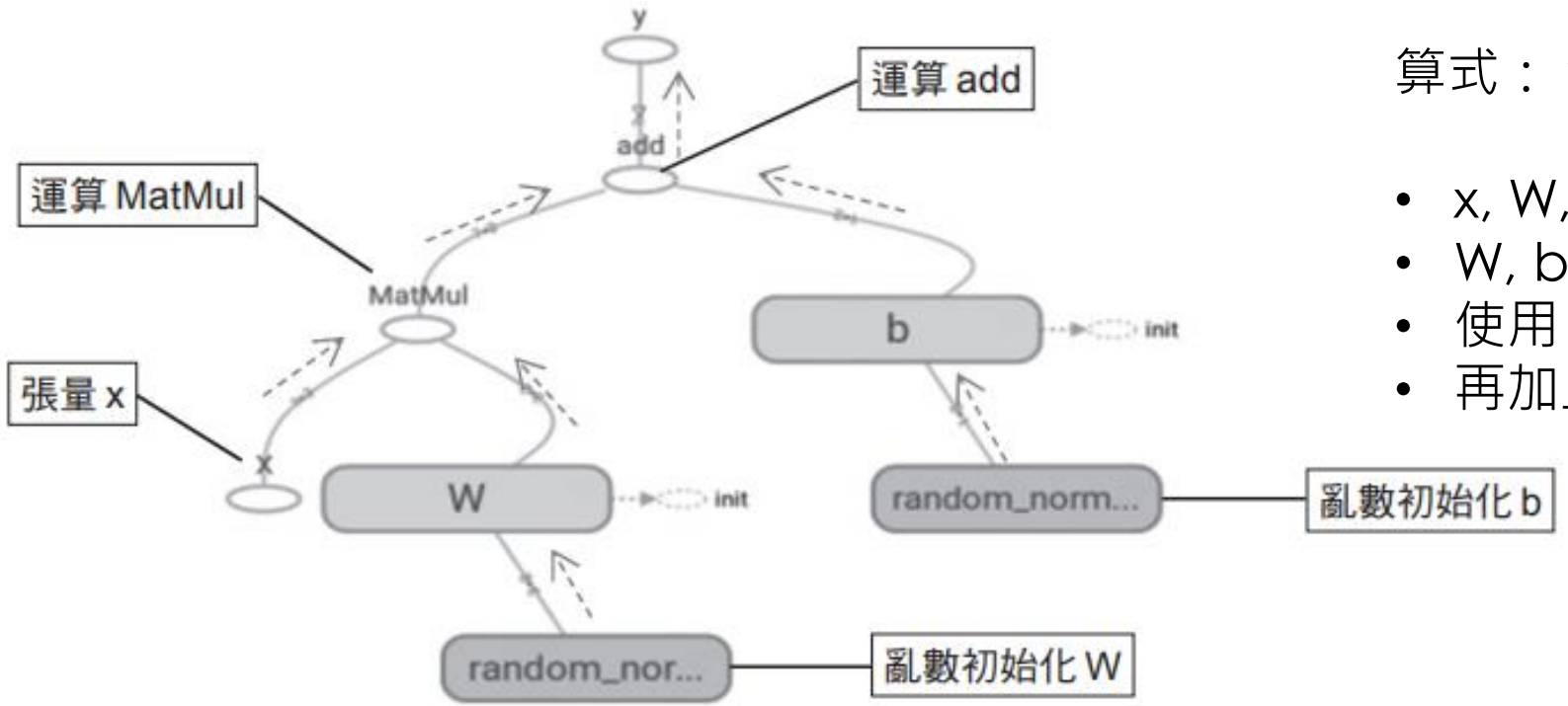
TensorFlow 是由 Tensor 與 Flow 組成

- Tensor 張量

在數學裡，張量是一種幾何實體，或是廣義上的「數量」，在此所謂的「數量」包含「純量、向量或矩陣」。0 維的張量爲純量，1 維的張量是向量，2 維以上的張量是矩陣。

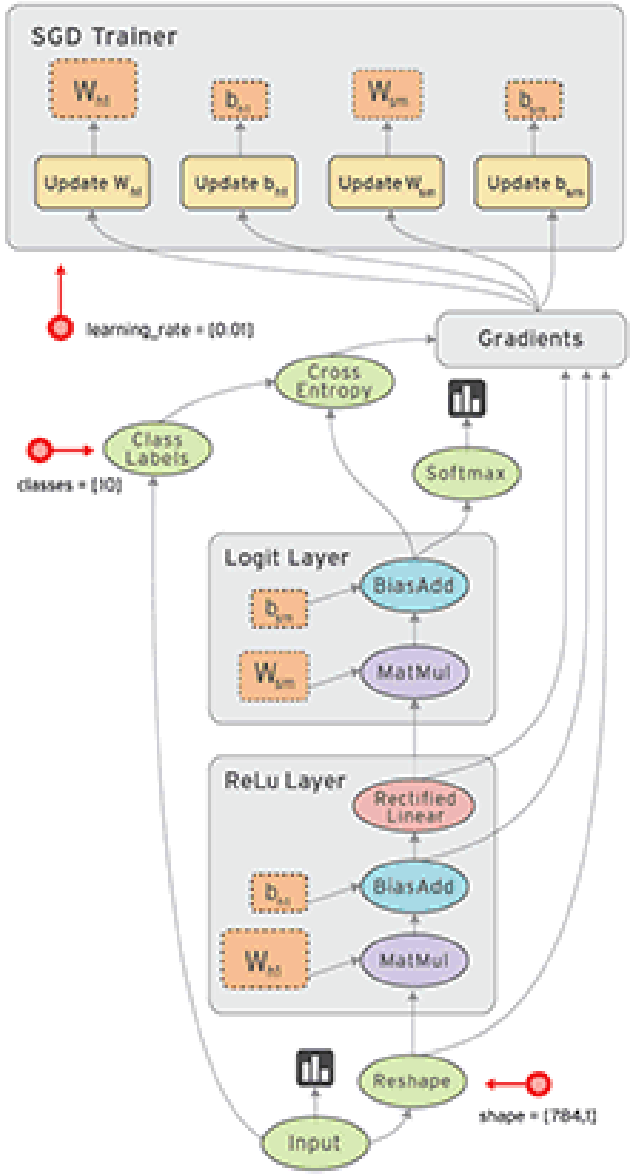


- Flow 資料流程
- 為讓 TensorFlow 支援不同的程式語言與平台
- TensorFlow 需先建立「計算圖」(Computational Graph)
  - 張量運算與資料處理流程



算式： y=MatMul(x, W)+b

- x, W, b 都是張量(矩陣)
- W, b 先用亂數初始化
- 使用 MatMul 進行張量乘積
- 再加上 b，可得結果 y



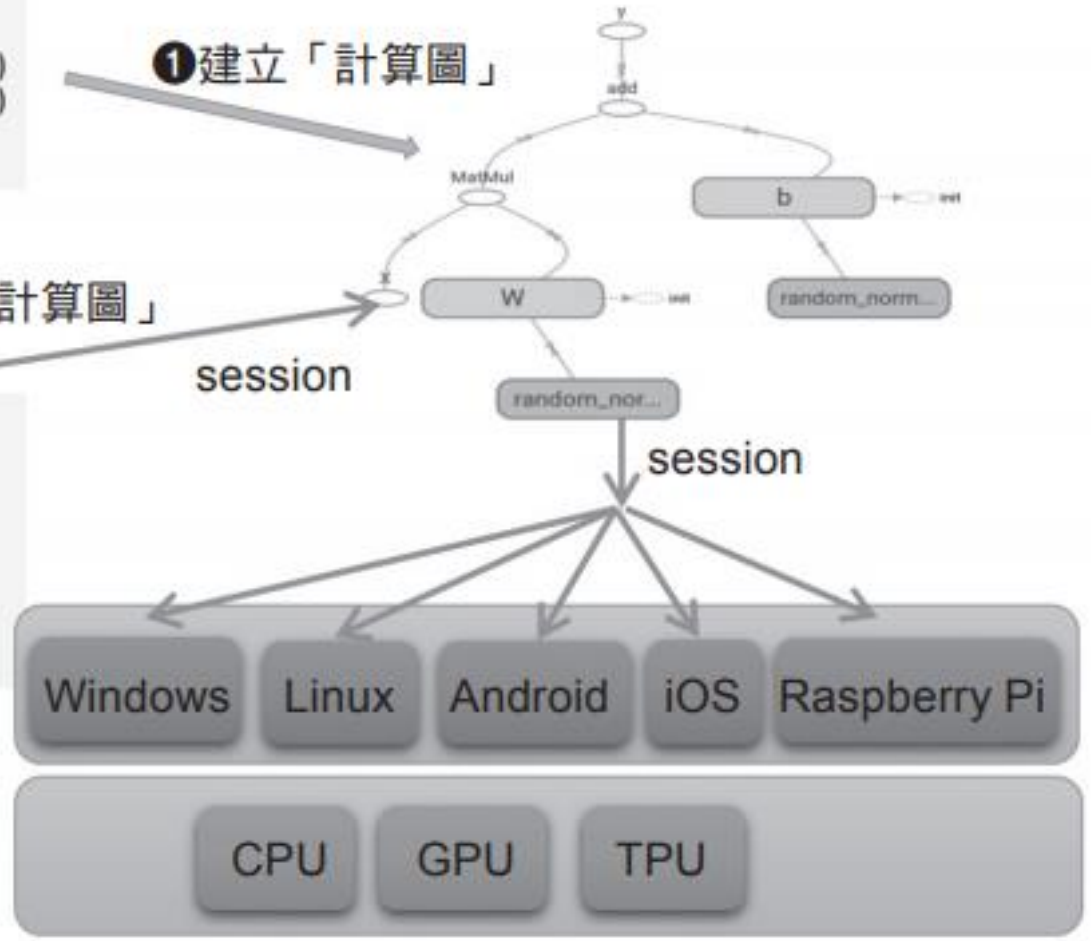
https://www.tensorflow.org/images/tensors_flowing.gif

3.3 TensorFlow 程式設計模式

```
import tensorflow as tf
import numpy as np
W = tf.Variable(tf.random_normal([3, 2]),name='W')
b = tf.Variable(tf.random_normal([1, 2]),name='b')
X = tf.placeholder("float", [None,3],name='X')
y=tf.nn.sigmoid(tf.matmul(X,W)+b,'y')
```

➢ **建立 Session (對談)**

```
with tf.Session() as sess:
    init = tf.global_variables_initializer()
    sess.run(init)
    X_array = np.array([[0.4,0.2 ,0.4],
                        [0.3,0.4 ,0.5],
                        [0.3,-0.4,0.5]])
    (_b,_W,_X,_y)=sess.run((b,W,X,y),
                           feed_dict={X:X_array})
```



補充 - 程式語言世代

第一代：機器語言(Machine language)

- 由0和1的邏輯狀態組成，CPU可直接執行
- 執行速度最快、開發難度高，可讀性低

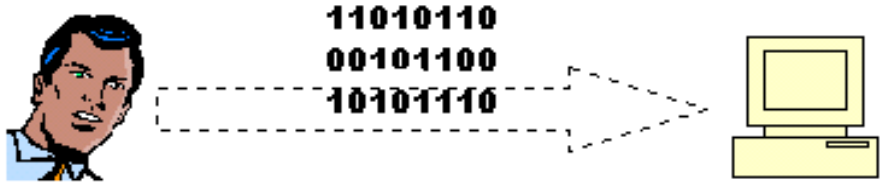


圖 2.5 人和電腦溝通示意圖(一)

第二代：組合語言

- 屬於低階語言，但可讀性較高
- 例如：ADD代表「相加」、LDA代表「載入」、MOV代表「搬移」
- 要使用組譯器(Assembler)編譯，才可由CPU執行

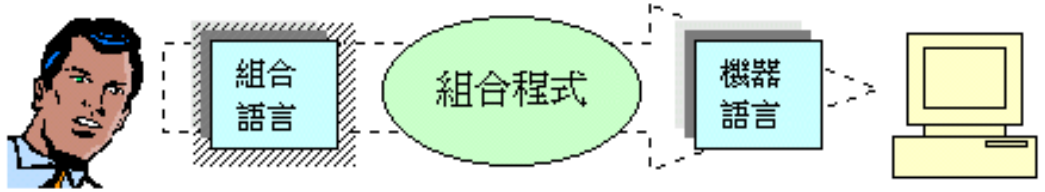


圖 2.6 人和電腦溝通示意圖(二)

補充 - 程式語言世代

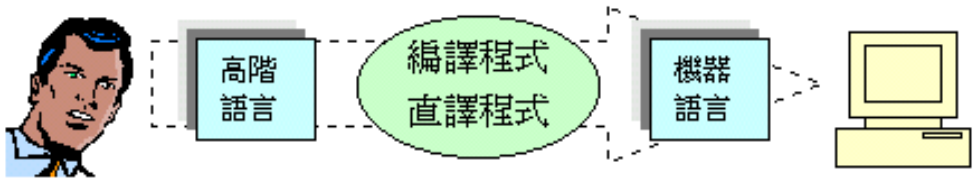


圖 2.7 人和電腦溝通示意圖(三)

第三代：高階語言(High level language)

- 用編譯程式(Compiler)來轉換成機器碼
- 早期依照指令邏輯順序執行，屬於程序導向語言(Procedure-Oriented Language)
- 物件導向程式(Object-Oriented Programming)
  - 把程式設計的概念具體化、物件化
  - 以**物件**的角度去分析和解決問題
  - 突破以往程序導向語言只能**循序單向**的設計缺失
  - 因物件簡便、維護容易及可重覆使用等特性，加快程式開發速度
- 常見C、C++、Java

補充 - 程式語言世代

第四代：查詢語言(Query Language)、非程序導向語言

- 語法較接近人類語言，只需將步驟寫出來，不必管電腦如何執行
- 常見如資料庫查詢語言：Oracle、SQL
  - SQL 可以使用select, from, order by 等指令查詢和排序
  - 「SELECT name FROM users WHERE age > 18」

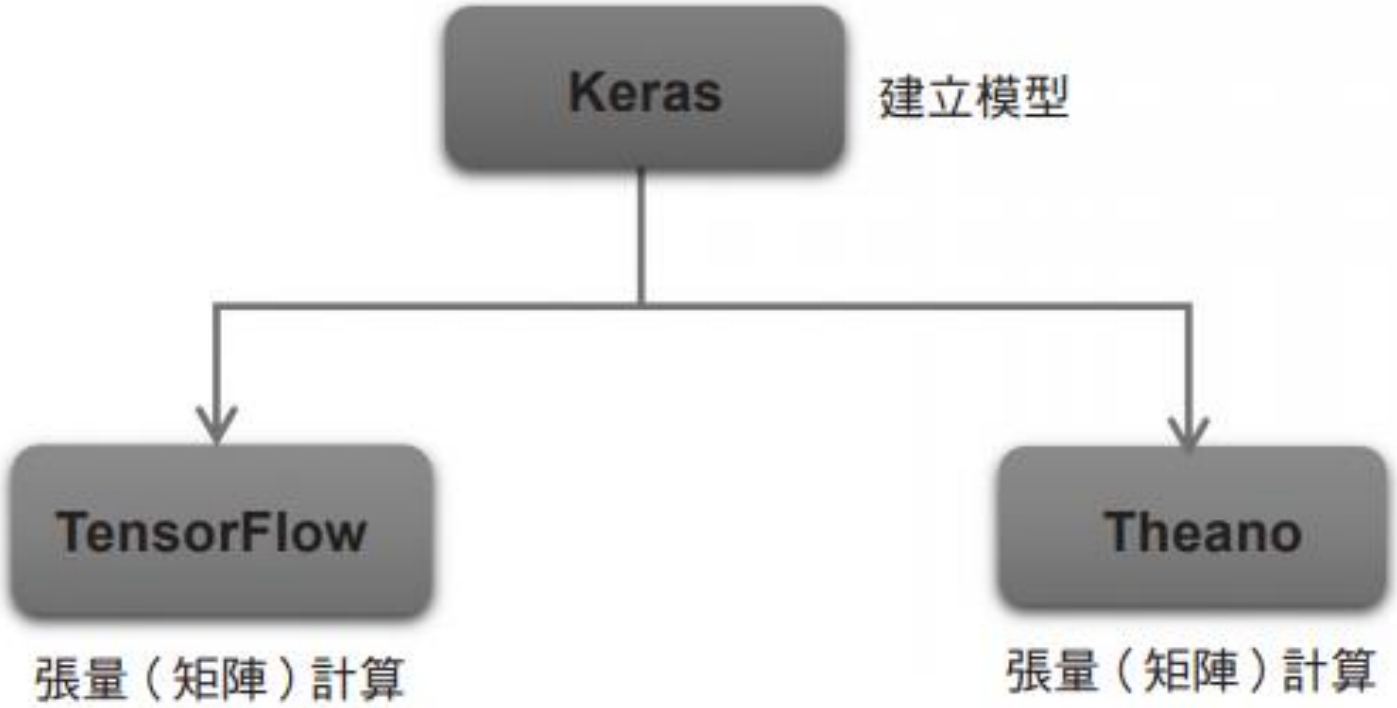
第五代：邏輯導向語言，又稱自然語言

- 目前主要用於AI研究領域
- 沒有特別語法，能夠讓電腦直接處理人類語言所寫的問題

3.4 Keras介紹

- Keras 是開放原始碼的「高階」深度學習「程式庫」
  - Tensorflow 相比較，相對「低階」

- Keras 快速方便運算的主要原因
  - 已建好訓練模型的輸入層、隱藏層、輸出層架構
  - 使用者只需加入並且填寫所需參數
    - 神經元個數、activation function...
  - 只處理模型的建立、訓練與運用，使用更少的程式碼
  - 運算交給後端引擎(backend engine)： TensorFlow 或 Theano



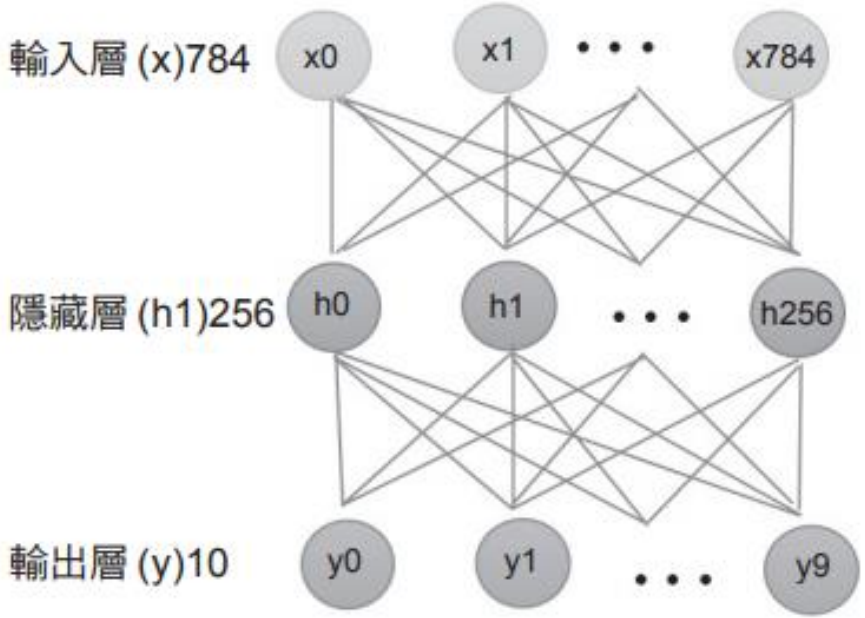
本書介紹的 Keras 範例，都是使用 TensorFlow 作爲後端引擎，因爲是以 TensorFlow 作爲底層，所以之前章節所介紹的 TensorFlow 的好處，例如：跨平台與執行效能，都可以享受到。

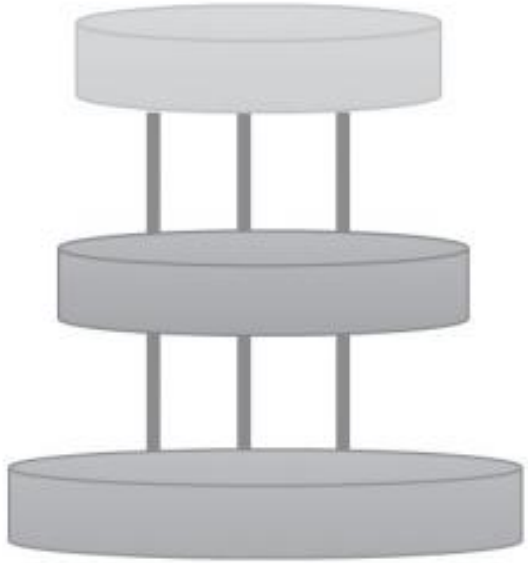
Keras 深度學習程式庫特色

- 簡單快速的建立原型 prototyping：keras 具備友善的使用者介面、模組化設計、可擴充性。
- 已經內建各式類神經網路層級，例如：卷積層 CNN、RNN，可以幫助您快速建立神經網路模型。
- 透過後端引擎（backend engine）：Theano 與 TensorFlow，可以在 CPU 與 GPU 運行。
- 以 Keras 開發的程式碼，更簡潔、更可讀性、更容易維護、更具生產力。
- Keras 的說明文件也非常齊全，官網上提供的範例，也非常淺顯易懂。

3.5 Keras 程式設計模式

- 用 Keras 建立深度學習模型就像做一個多層蛋糕
- 建立蛋糕架→選擇蛋糕層→指定裝飾種類→將蛋糕放到蛋糕架上
- 以建立 MLP 模型為例

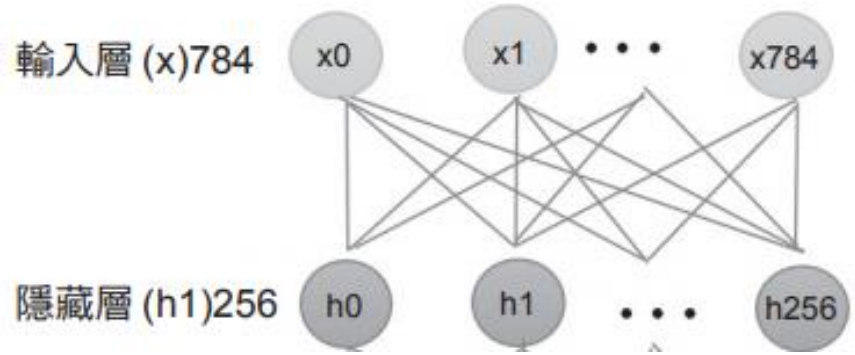




1. 建立 Sequential 模型(蛋糕架)

```
from keras.models import Sequential
model = Sequential()
```

- Sequential 是可堆疊多個神經網路層的模型



2. 加入輸入層、隱藏層到模型(加1、2層蛋糕)

```
model.add(Dense(units=256,input_dim=784,kernel_initializer='normal'
,activation='relu'))
```

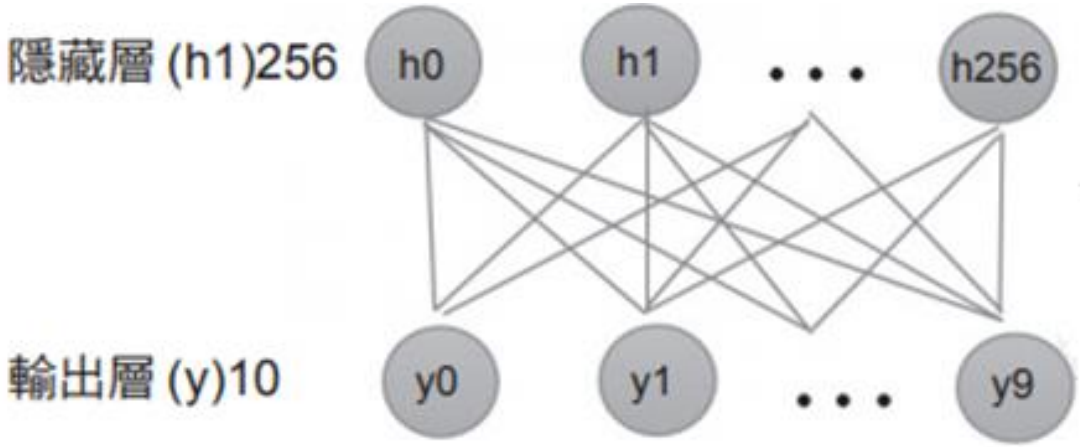
- Dense 是 Keras 內建的一種網路層

3. 加入輸出層到模型(加第3層蛋糕)

```
model.add(Dense(units=10,kernel_initializer='normal',activation='softmax'))
```

- 跟隱藏層語法相比較

```
model.add(Dense(units=256,input_dim=784,kernel_initializer='normal',activation='relu'))
```



	Keras	TensorFlow
學習難易度	簡單	比較困難
使用彈性	中等	高
開發生產力	高	中等
執行效能	高	高
適合使用者	初學者	進階使用者
張量(矩陣)運算	不需自行設計	需自行設計

- 先透過 Keras，快速簡單地建立深度學習模型
- 再學習 TensorFlow，針對性地建立適合的 ANN 模型