卷積神經網路基礎

2025/06/28

CREATED FOR INNOLUX

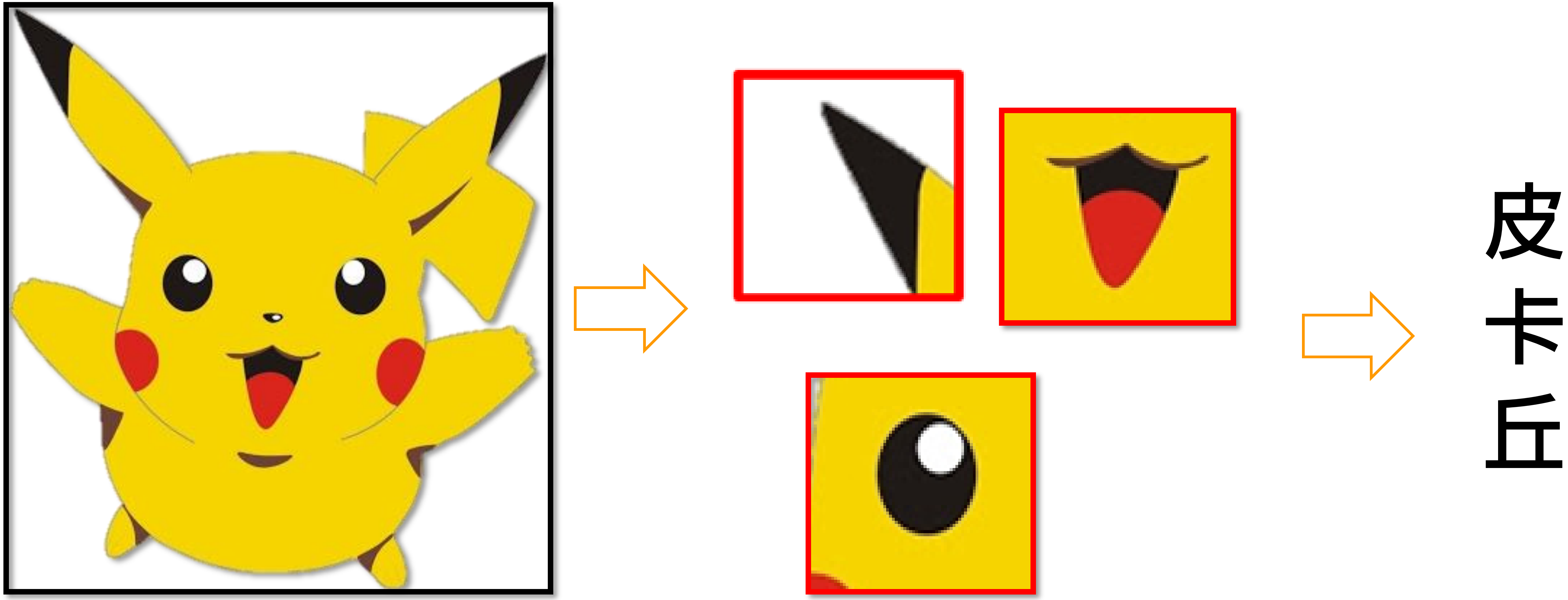
PRESENTED BY AI Foundation

目錄

1. 傳統電腦視覺與深度學習方法比較
2. 卷積神經網路介紹
3. 程式實作

傳統電腦視覺與深度學習方法比較

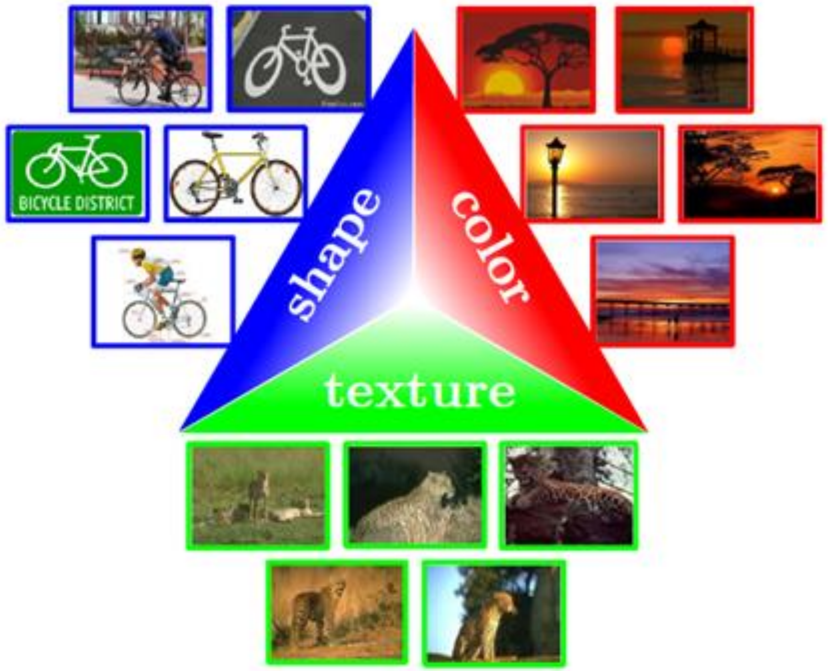
如何認識一張圖片

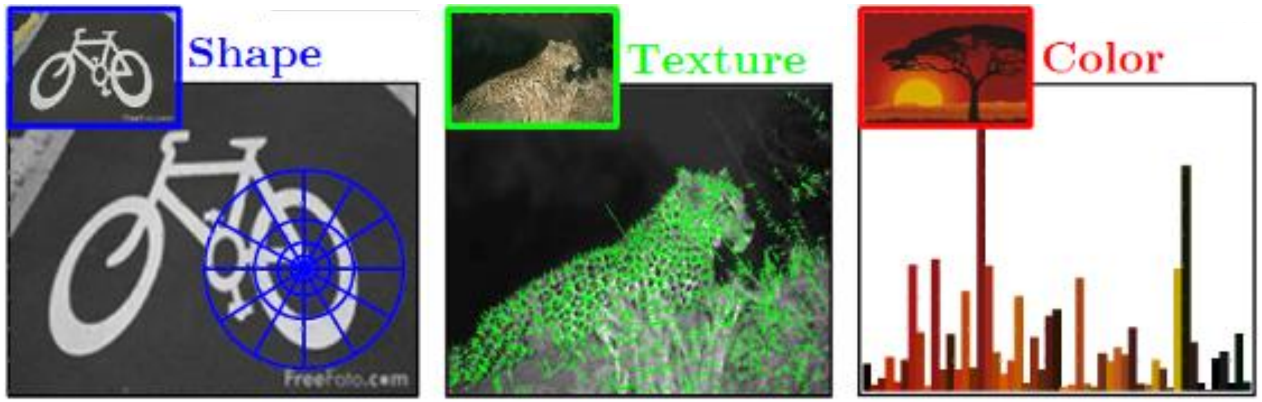


傳統電腦視覺

特徵萃取是電腦判斷圖片的關鍵

- 先前在電腦視覺分類作業的進展皆在於找到關鍵的特徵

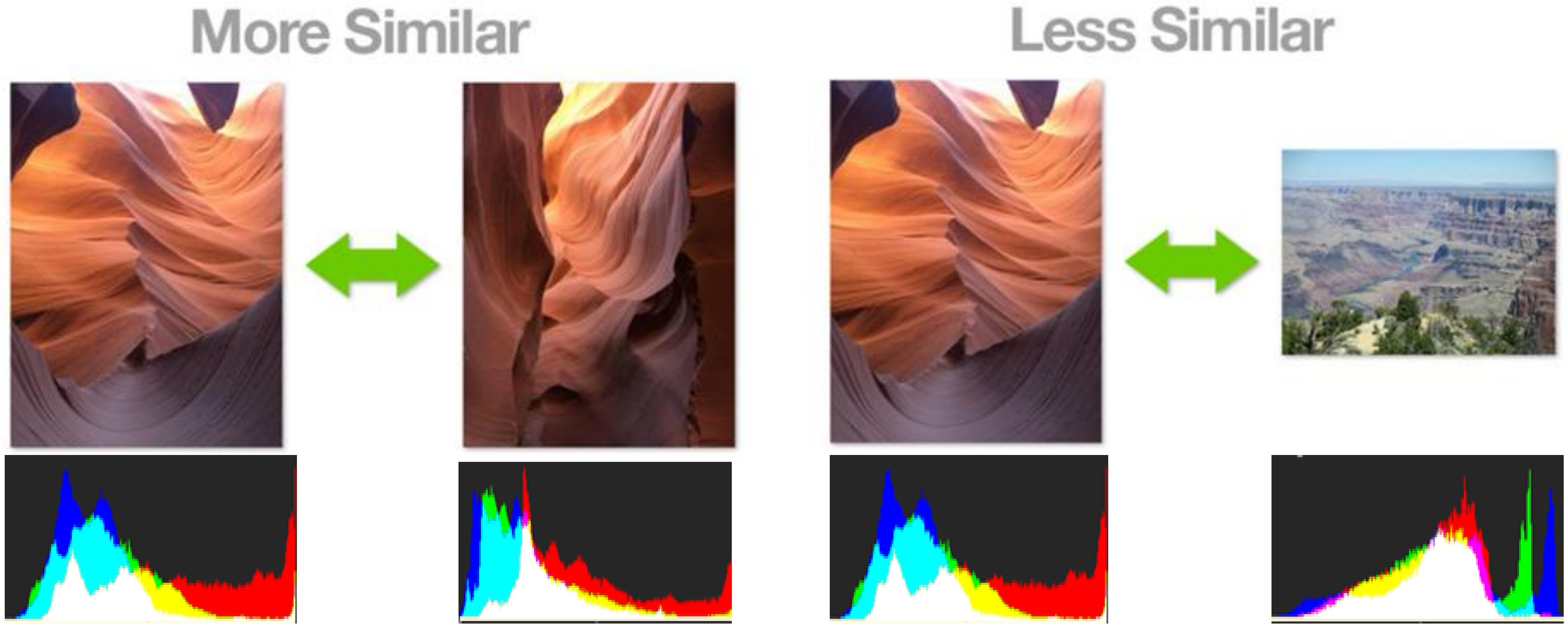




參考資料: Y LeCun

顏色資訊的萃取

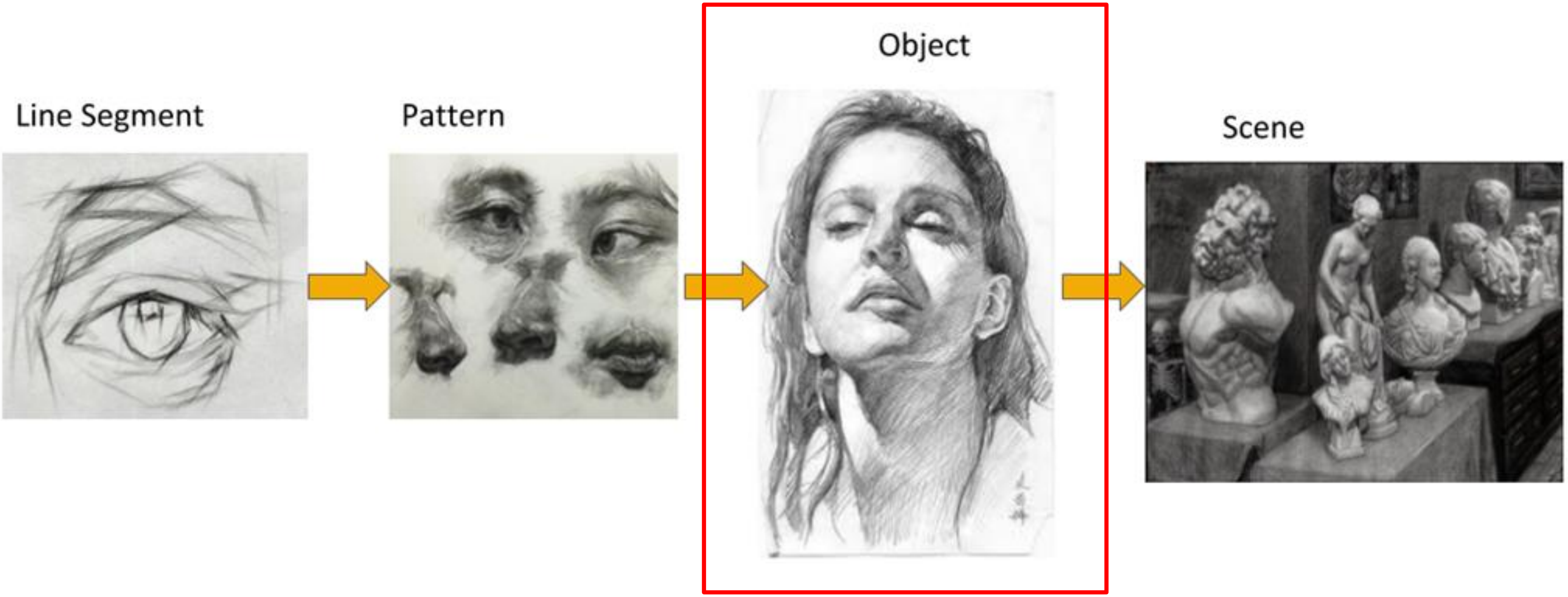
圖片在視覺上可以直接以顏色作出區分，如何能用量化描述一張圖片的顏色呢?



透過描述顏色的特徵向量，明顯可以看出左二兩張圖的色調較相近，皆為暖色調，右圖則為冷色調。

視覺外型的資訊萃取

- 圖的構成: 線條→圖案(pattern)→物件→場景



視覺外型的資訊萃取

Kernel = Filter

Input image

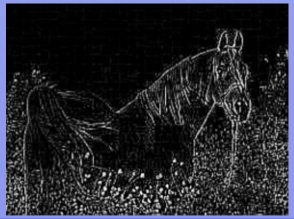
Convolution Kernel

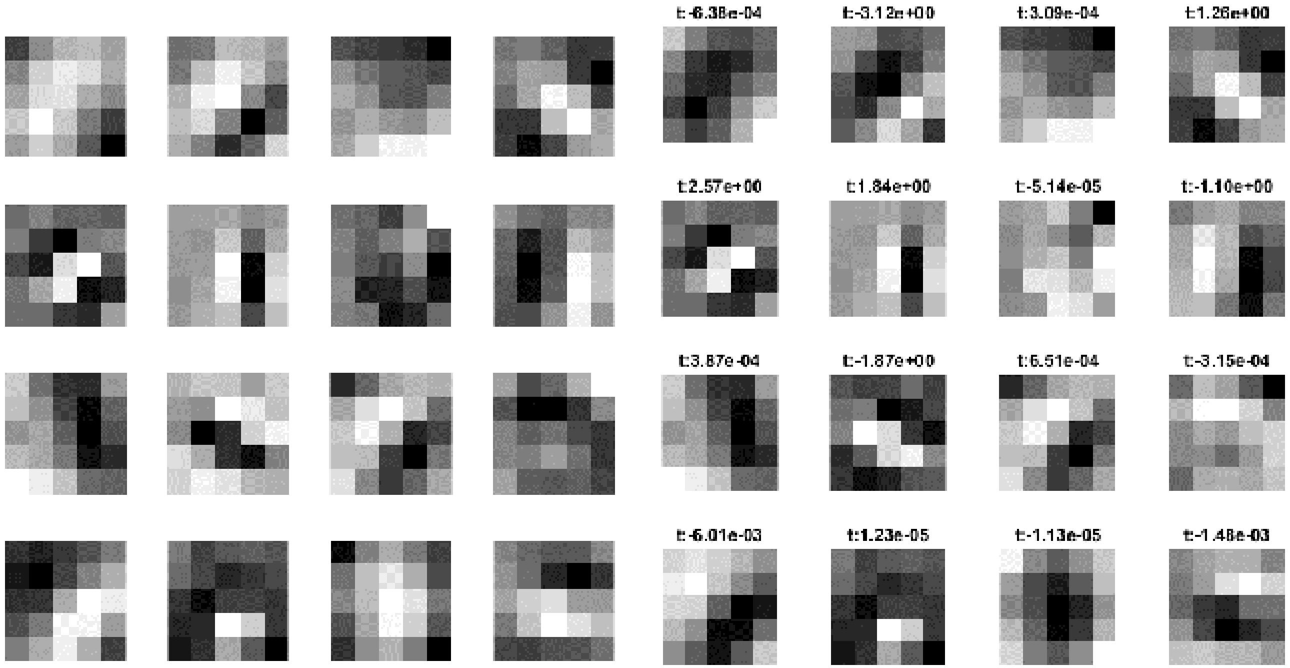
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Feature map

透過設計特別的運算，得以取得圖像當中的外型資訊。

資料來源: Convolution example

原圖	3 x 3 filters	卷積後結果	結果
	$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$		原圖不變
	$\begin{matrix} 1 & 1 & 1 \\ 1 & -7 & 1 \\ 1 & 1 & 1 \end{matrix}$		銳利化
	$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$		邊緣強化
	$\begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$		找水平特徵
	$\begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$		找垂直特徵

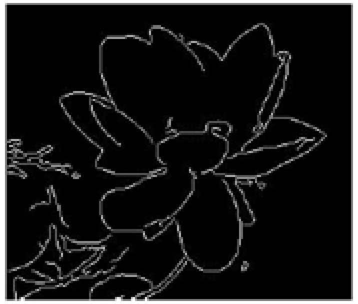


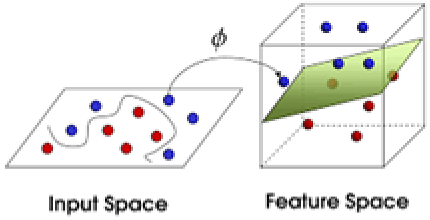
傳統電腦視覺的重點流程放在「hand crafted features」，設計不同用途的過濾器。

傳統電腦視覺方法

- 訓練階段

圖片收集 → 特徵萃取 → 模型訓練 → 模型預測



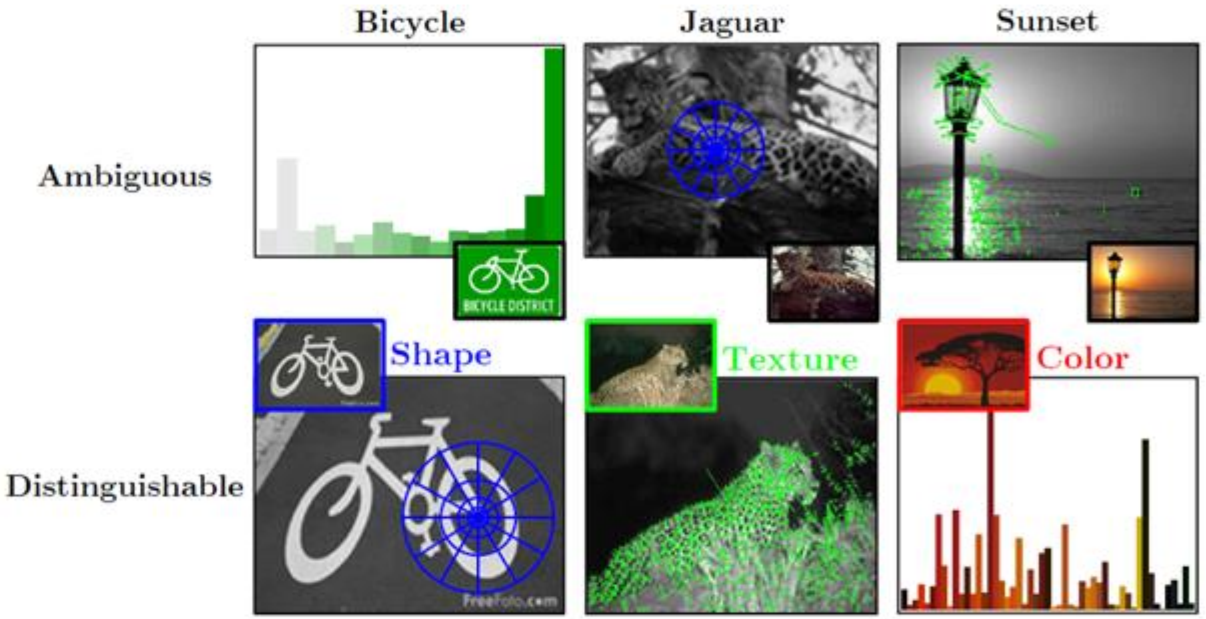


support vector machines
(SVMs)

dogs?	✖
flowers?	○
trains?	✖
persons?	✖

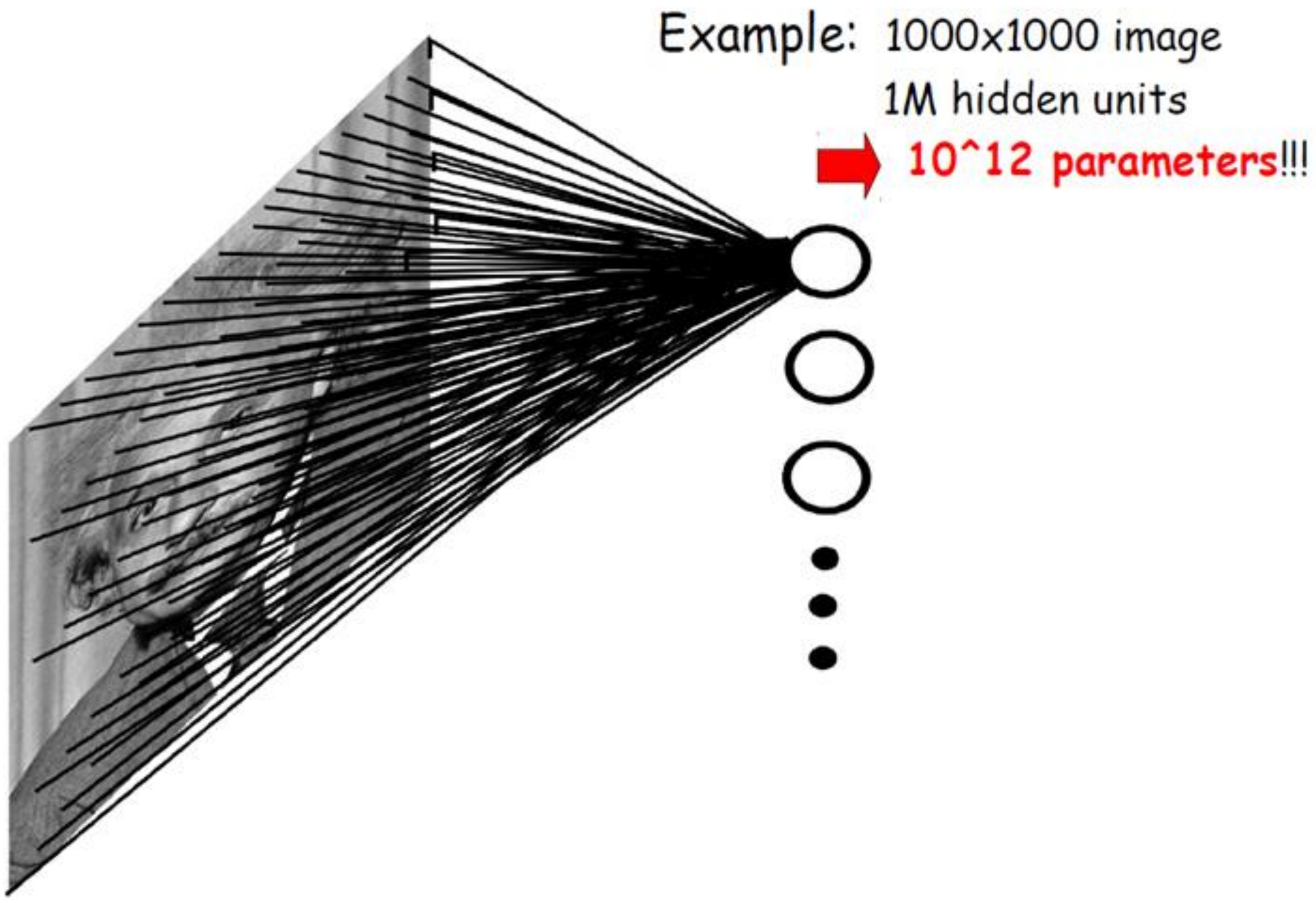
特徵萃取是電腦判斷圖片的關鍵

- 先前在電腦視覺分類作業的進展皆在於找到關鍵的特徵，但過去的做法可能有一些問題
  - 人為設計的特徵真的是最佳特徵嗎？
  - 最佳特徵可能因作業不同而有所不同



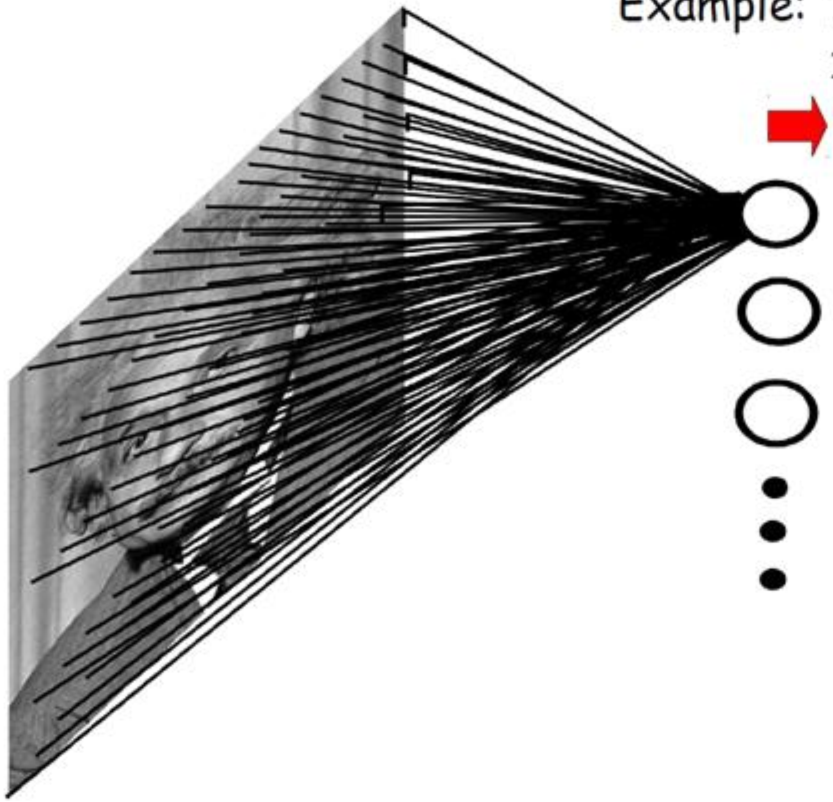
深度學習方法

DNN



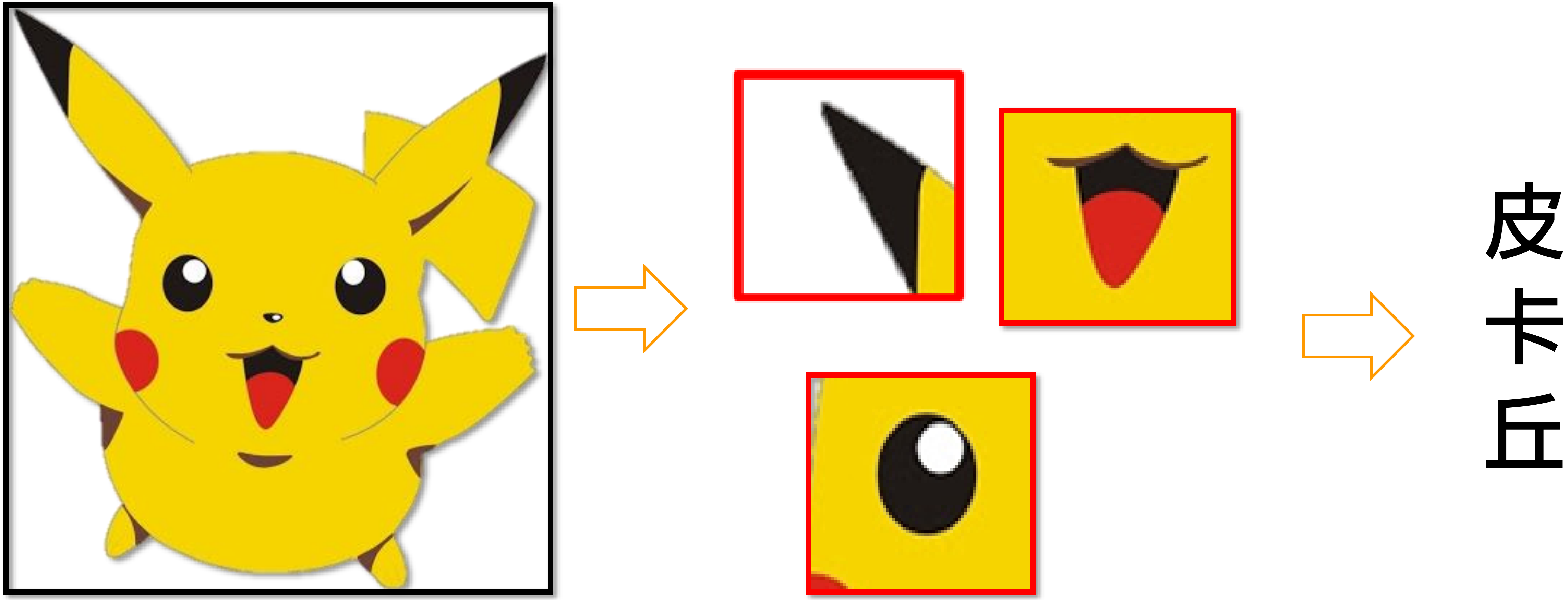
將DNN應用到圖片的問題

- 將原始圖片以全連接層連結
  - 參數量將急遽增加
  - 未考慮圖片資料特性



圖片來源: MA Ranzato

如何認識一張圖片



傳統電腦視覺與深度學習方法(CNN)比較

- 傳統電腦視覺
  - Fixed/engineered features + trainable classifier

Hand-crafted Feature Extractor → **Trainable** Classifier →

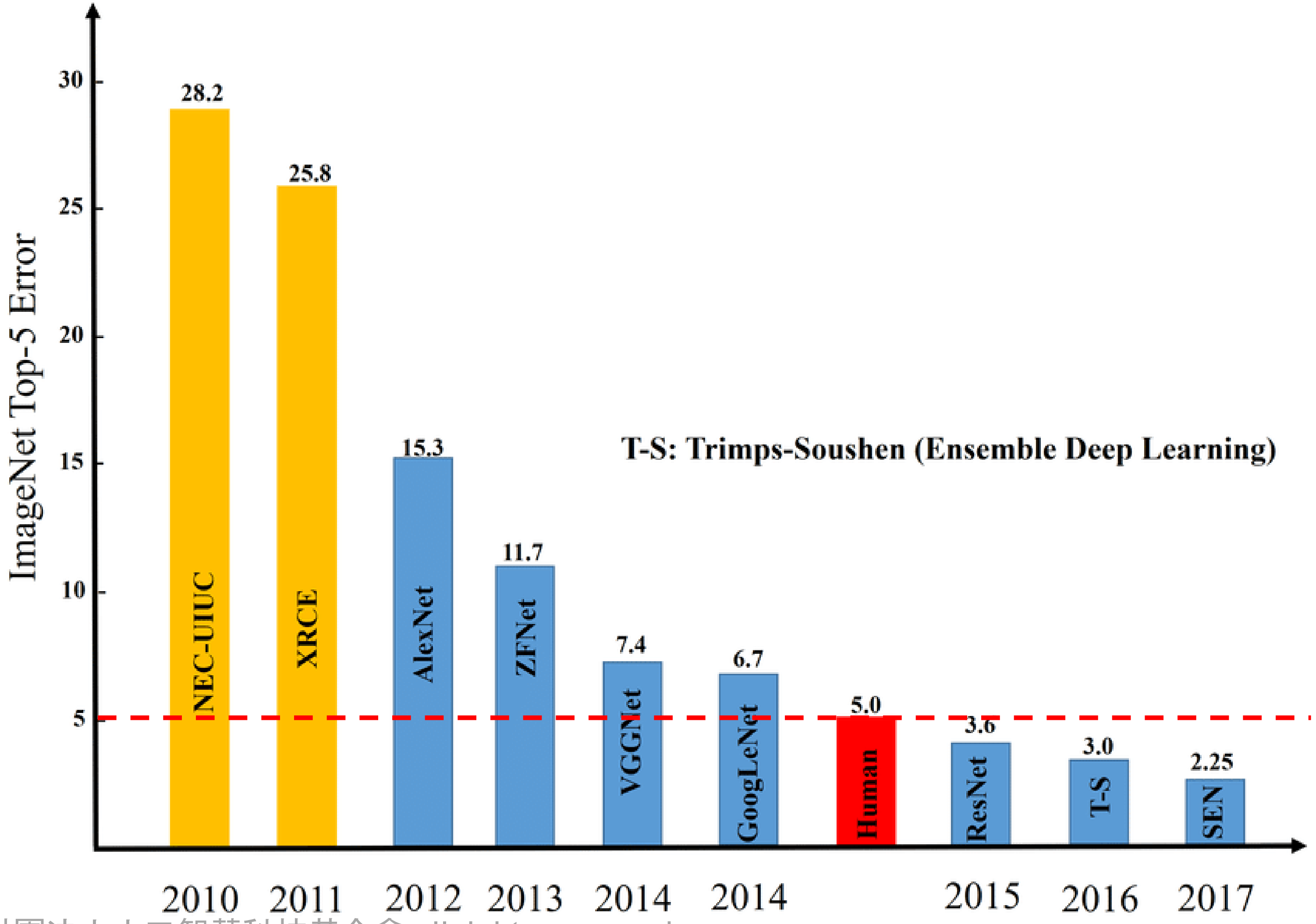
- 深度學習方法
  - Trainable features + trainable classifier

Trainable Feature Extractor → **Trainable** Classifier →

參考資料: Y LeCun & MA Ranzato

資料來源 ：Zaid Alyafeai & Lahouari Ghouti, Expert Systems with Applications 141, p112951 (2019)

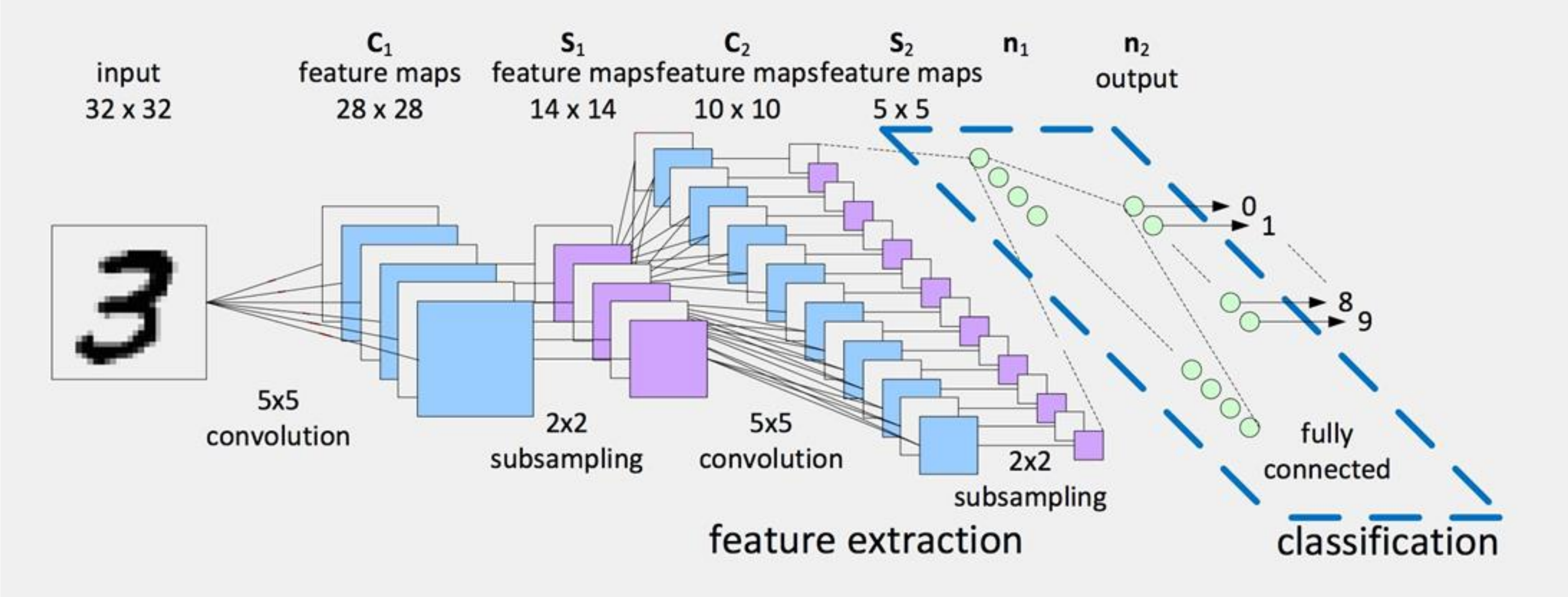
CNN on ILSVRC



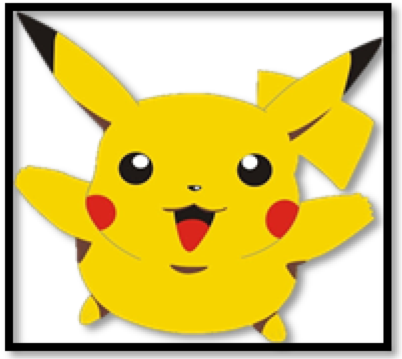
卷積神經網路介紹

Feature Extraction, Stride, Padding

卷積神經網路架構


input
32 x 32
C_1
feature maps
28 x 28
S_1
feature maps
14 x 14
C_2
feature maps
10 x 10
S_2
feature maps
5 x 5
n_1
n_2
output
0
1
8
9
5x5
convolution
2x2
subsampling
5x5
convolution
2x2
subsampling
fully
connected
feature extraction
classification

卷積神經網路架構



Convolutional Layer
&
Activation Layer

Pooling Layer

Convolutional Layer
&
Activation Layer

Pooling Layer

Can be performed many times

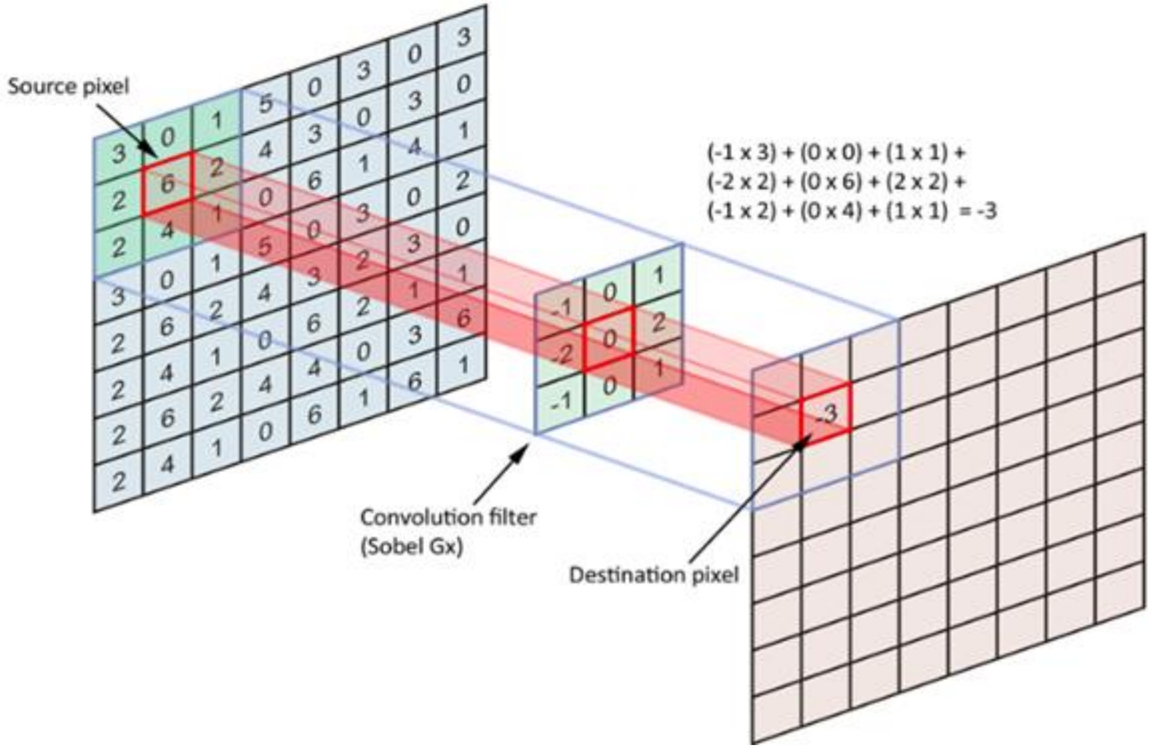
Flatten / GAP

Feedforward DNN

PIKACHU! !

卷積層(Convolutional Layer)

- 在CNN中，卷積層的用途在於提取特徵
- 每一組Filter內的數字即為神經網路中的權重
(將透過訓練資料做改變)



圖片來源：https://datascience.stackexchange.com/questions/23183/why-convolutions-always-use-odd-numbers-as-filter-size

卷積層(Convolutional Layer)

Layer 1

Image

0	0	0	0	0
0	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

*

Filter

1	0	0
0	**1**	**0**
0	**0**	**1**

=

Feature Map

1	2	2
1	2	3
3	1	3

這個步驟有那些地方是可以變化的呢?

Layer 2

1	2	2
1	2	3
3	1	3

*

1	0	0
0	1	0
0	0	1

=

6

Filter Size

- 3x3 filter

Image

0	0	0	0	0
0	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

*

Filter

1	0	0
0	1	0
0	0	1

=

Feature Map

1	2	2
1	2	3
3	1	3

Filter Size

- 5x5 filter

Image

0	0	0	0	0
0	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

*

Filter

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

=

3

Stride

- 使得卷積層的輸出大小降低
- 例：設定 stride 為 2

Image

0	0	0	0	0
0	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

*

Filter

1	0	0
0	1	0
0	0	1

=

Feature Map

1	2
3	3

遺失哪部分的資訊

Image

0	0	0	0	0
0	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

*

Filter

1	0	0
0	1	0
0	0	1

=

Feature map

1	2	2
1	2	3
3	1	3

Filter 觸及的次數

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

Zero-padding

- 若是邊界有較多資訊的圖片，使用 padding 會有比較好的效果

Image

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	1	1	0	0
0	1	1	1	1	1	0
0	0	1	0	1	1	0
0	0	0	1	0	1	0
0	0	0	0	0	0	0

*

Filter

1	0	0
0	1	0
0	0	1

=

Feature Map

0	1	1	0	0
1	1	2	2	0
2	1	2	3	2
0	3	1	3	2
0	0	2	0	2

Zero-padding 可以維持輸出與輸入的大小一致

Zero-padding

- Padding 要加的層數是取決於 Filter 的大小

Image

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0
0	0	1	1	1	1	1	0	0
0	0	0	1	0	1	1	0	0
0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

*

Filter

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

=

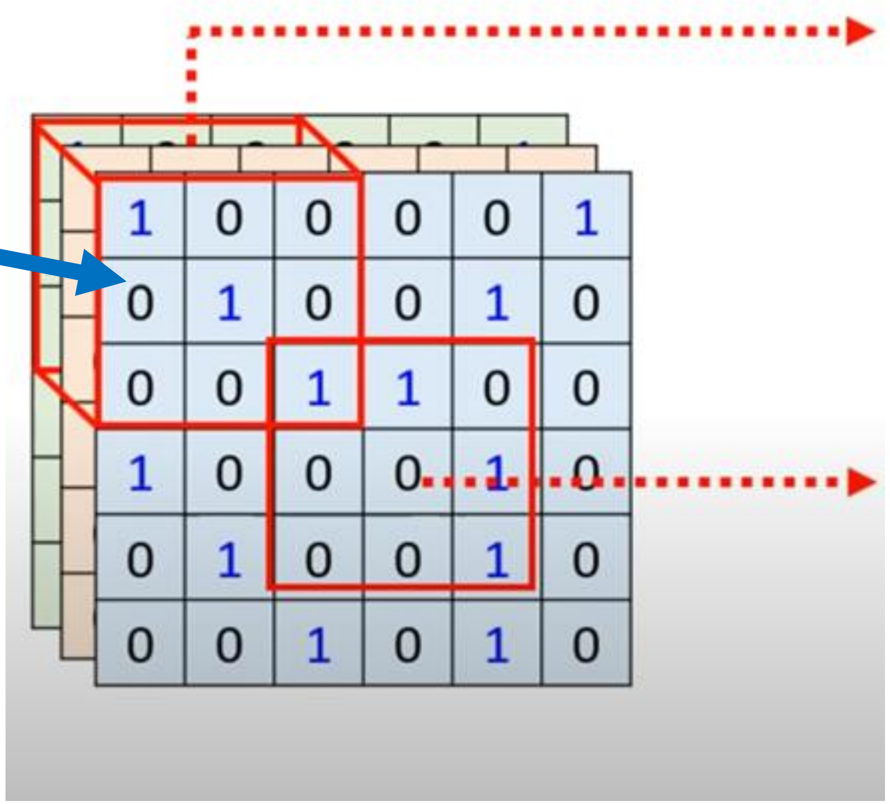
Feature map

1	2	2	0	0
1	2	3	2	0
3	1	3	3	2
0	3	1	3	3
0	0	3	1	3

卷積層的共享權重

- Convolution layer shared weights

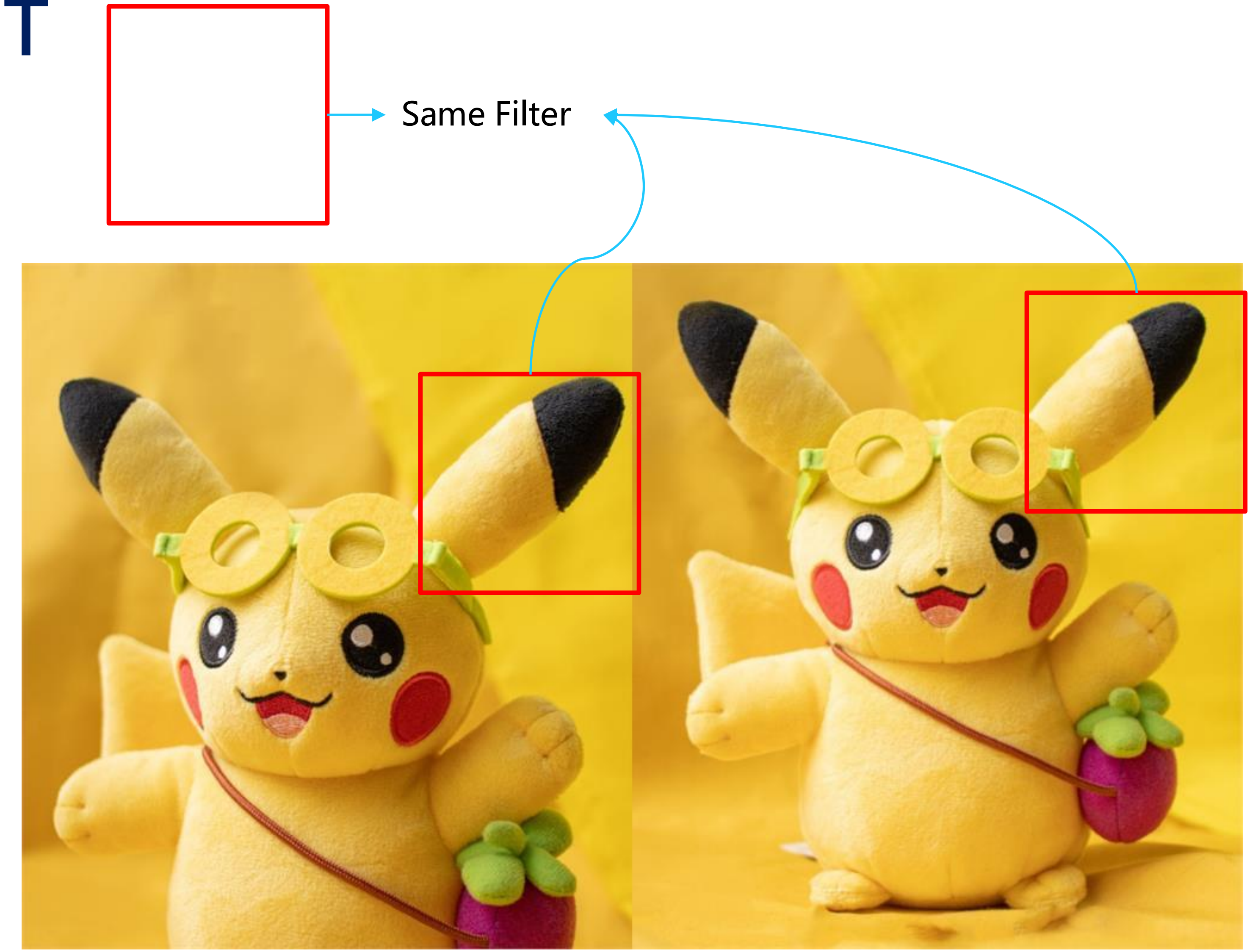
Share the same filter in a sliding



Parameters : (3*3*3+1)*1

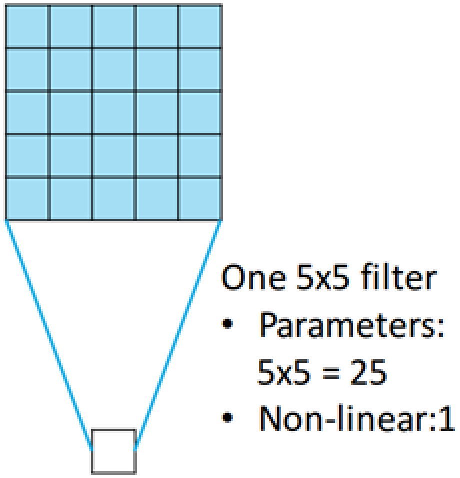
卷積演算法設計

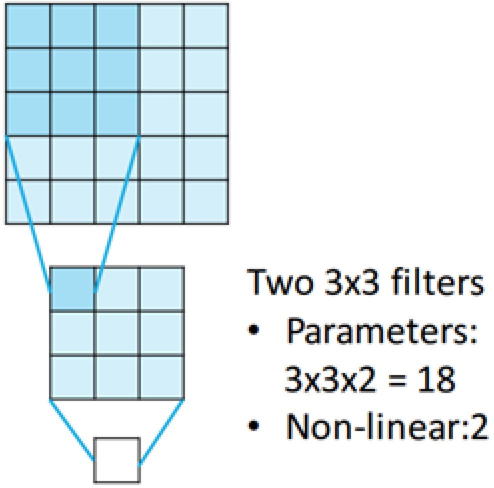
- 因應圖片特性
    - 相同權重的filter可應用在不同位置上



減少大卷積核的參數量

- 3x3 convolutional kernels – less parameters
  - Stacked convolutional layers have large receptive fields
  - More non-linearity
  - Less parameters to learn
  - More numbers of channels

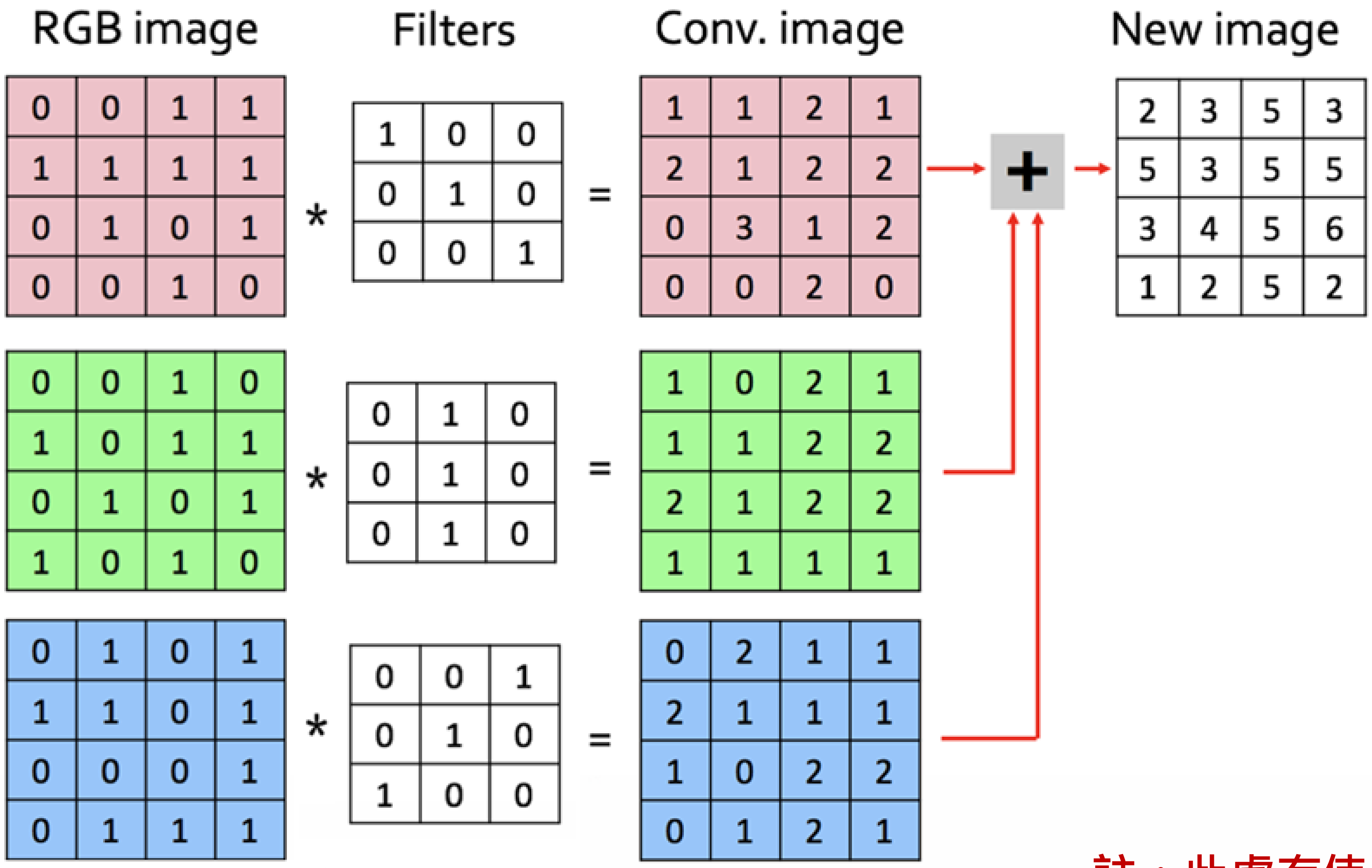




Parameters :
(5*5*1+1)*1 = 26

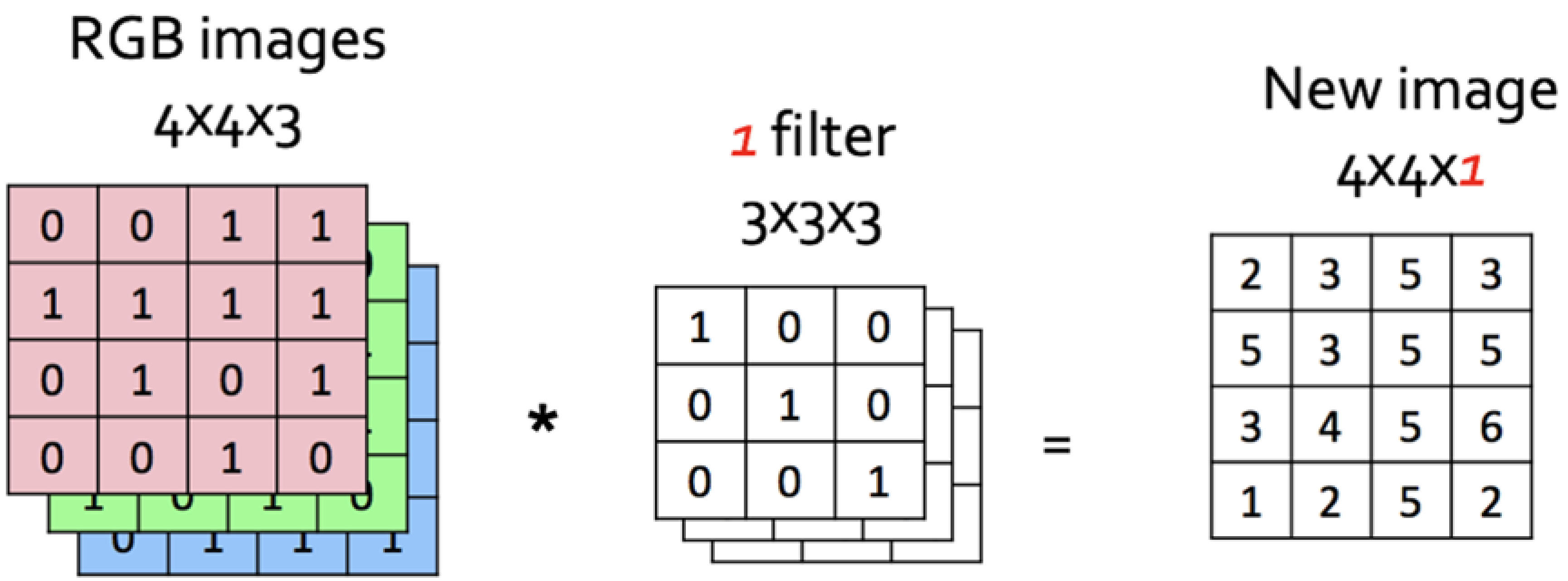
Parameters :
(3*3*1+1)*1
+(3*3*1+1)*1
=20

針對RGB影像進行卷積



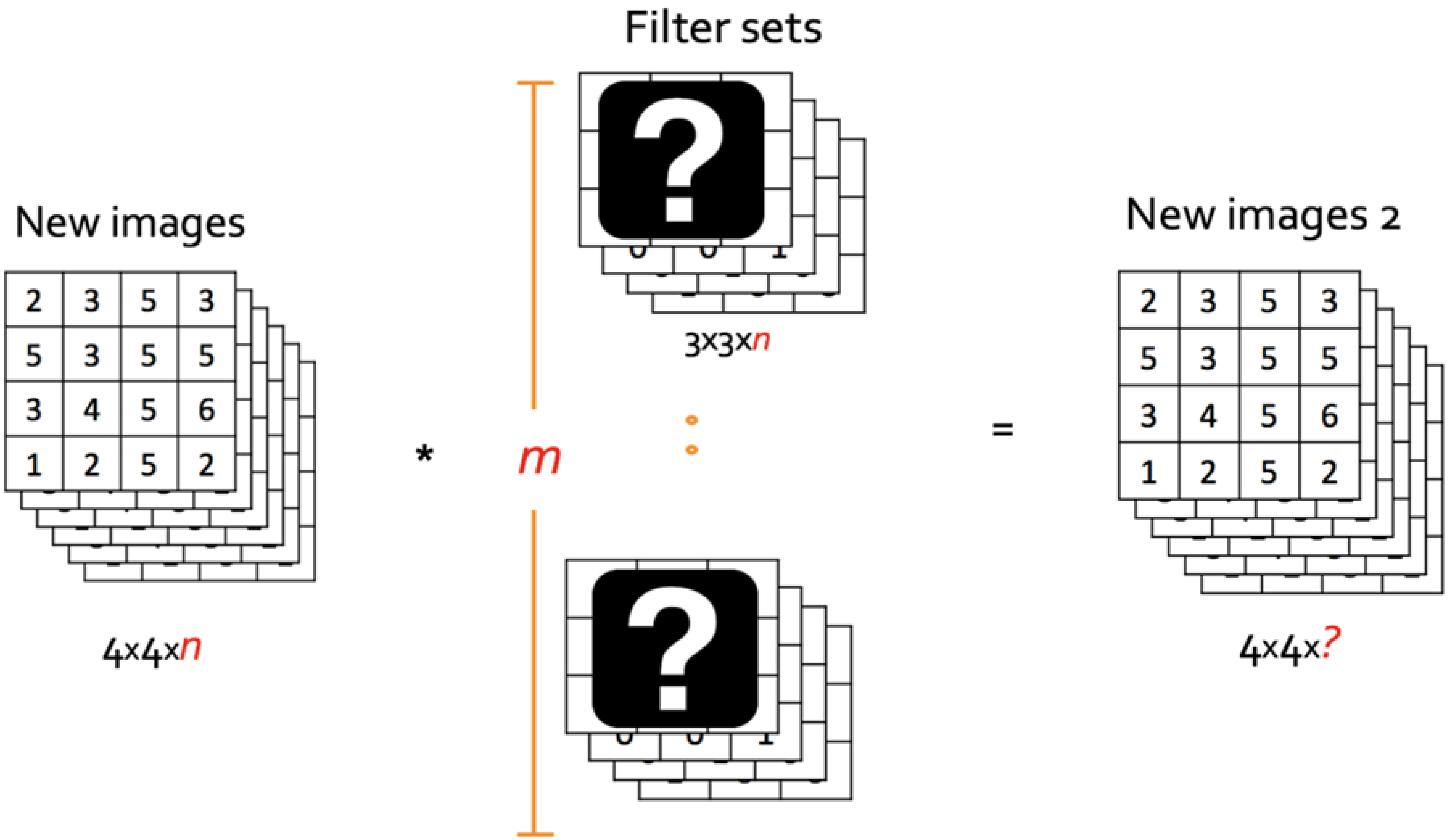
註：此處有使用Padding

Filter 數量與輸出維度的關聯

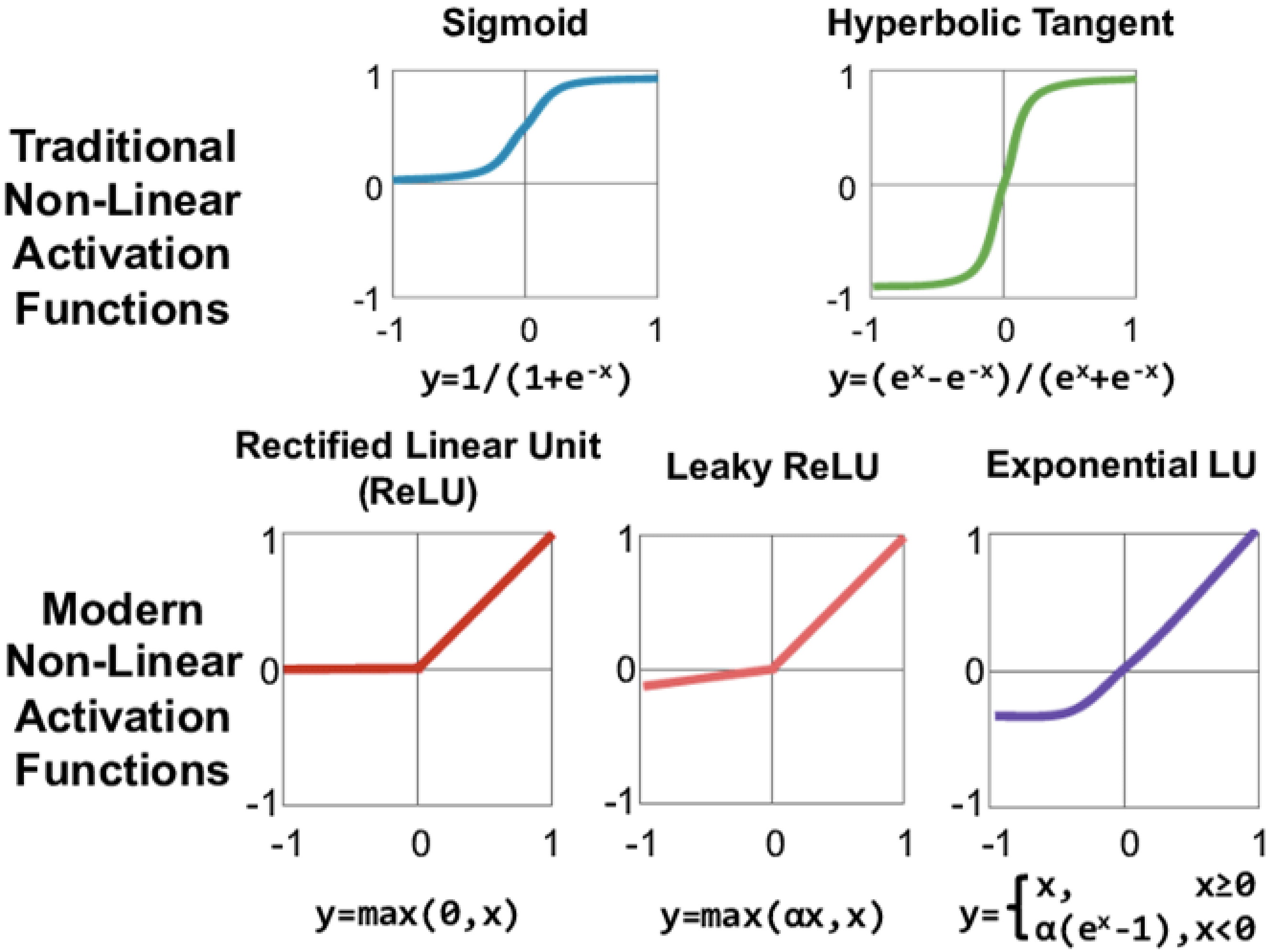


註：此處有使用Padding

動動腦時間：請問圖片中的？是多少



Activation Layer



在卷積神經網路中由於層數較多，需考量梯度消失與梯度爆炸問題，因此通常會使用ReLU類型的激活函數(Activation function)

圖片來源：https://www.researchgate.net/figure/Various-forms-of-non-linear-activation-functions-Figure-adopted-from-Caffe-Tutorial_fig3_315667264

Activation Layer

Feature map

0	0	4	-8	0
2	0	1	1	0
1	1	1	1	1
-3	1	0	1	1
-1	0	1	-2	1

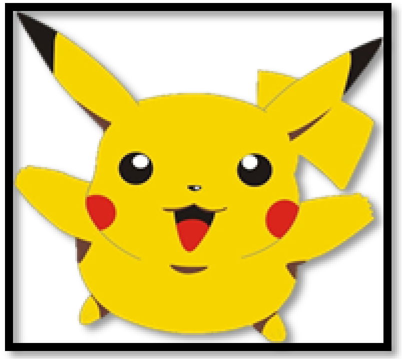
σ: Activation function

σ(features map)

Feature map after ReLU

0	0	4	0	0
2	0	1	1	0
1	1	1	1	1
0	1	0	1	1
0	0	1	0	1

卷積神經網路架構



Convolutional Layer & Activation Layer

Pooling Layer

Convolutional Layer & Activation Layer

Pooling Layer

Can be performed many times

Flatten / GAP

Feedforward DNN

PIKACHU!!

池化層(Pooling Layer)

- 池化層(Pooling Layer)作用
  - 減少模型參數
  - 防止模型過擬合(overfitting)
- Max pooling
  - 以區域內最大的數值作為代表
- Average pooling
  - 將區域內數值取平均

1	2	2	0
1	2	3	2
3	1	3	2
0	2	0	2

2	3
3	3

Max pooling

1.5	1.75
1.5	1.75

Average pooling

在池化層中無參數需要學習

池化層(Pooling Layer)

- Pool size

1	2	2	2	1	0
2	3	3	0	2	1
1	1	2	2	1	2
1	0	1	3	2	2
0	2	2	2	1	1
0	1	2	0	0	2

3	2
2	3

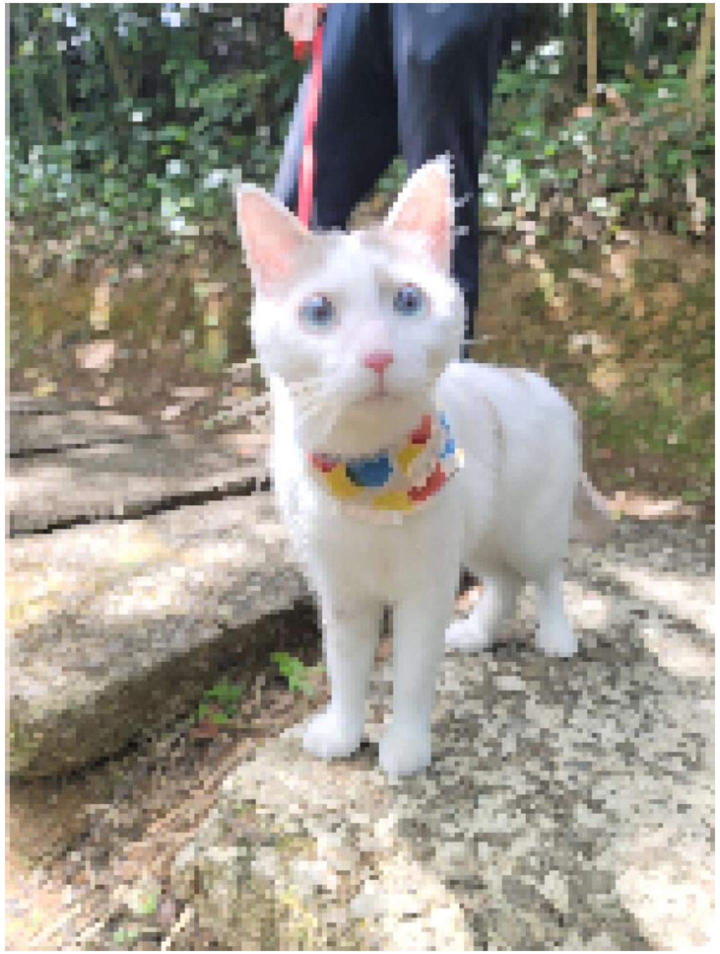
Pool size = (3, 3)

1	2	2	2	1	0
2	3	3	0	2	1
1	1	2	2	1	2
1	0	1	3	2	2
0	2	2	2	1	1
0	1	2	0	0	2

3	3	2
1	3	2
2	2	2

Pool size = (2, 2)

Max Pooling & Average Pooling



Max pooling

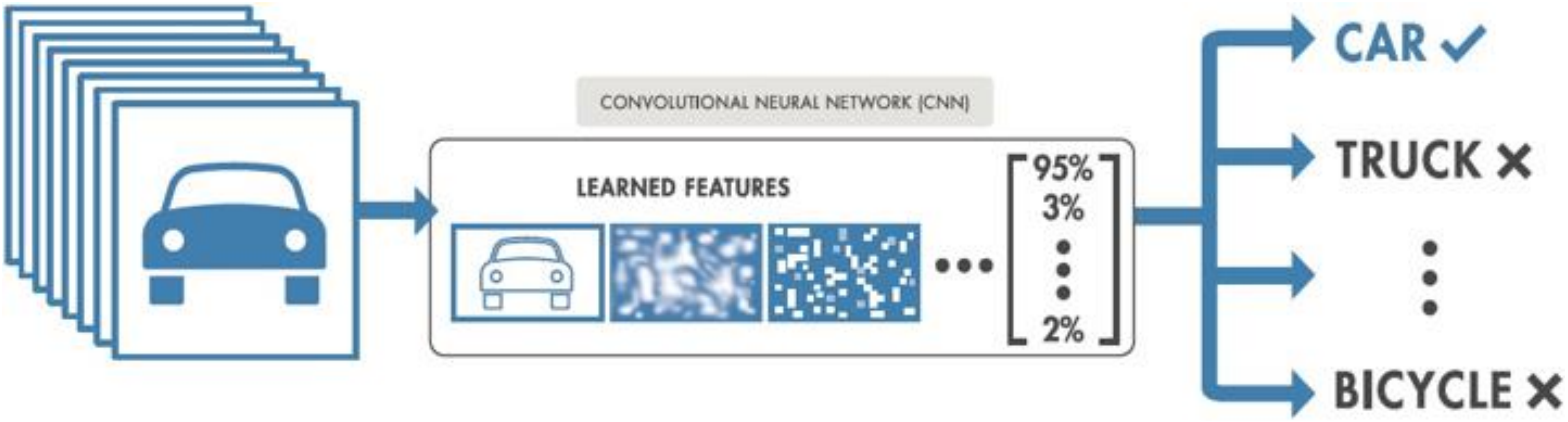
Average pooling

raw

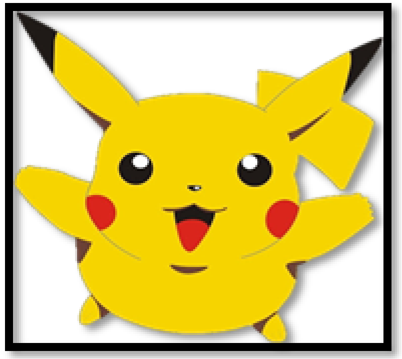
資料來源: Woody烏迪呆萌迪 (@woody0320meow)

小結

- 在卷積神經網路中主要使用卷積層與池化層作為特徵擷取的方式



卷積神經網路架構



Convolutional Layer
&
Activation Layer

Pooling Layer

Convolutional Layer
&
Activation Layer

Pooling Layer

Can be performed many times

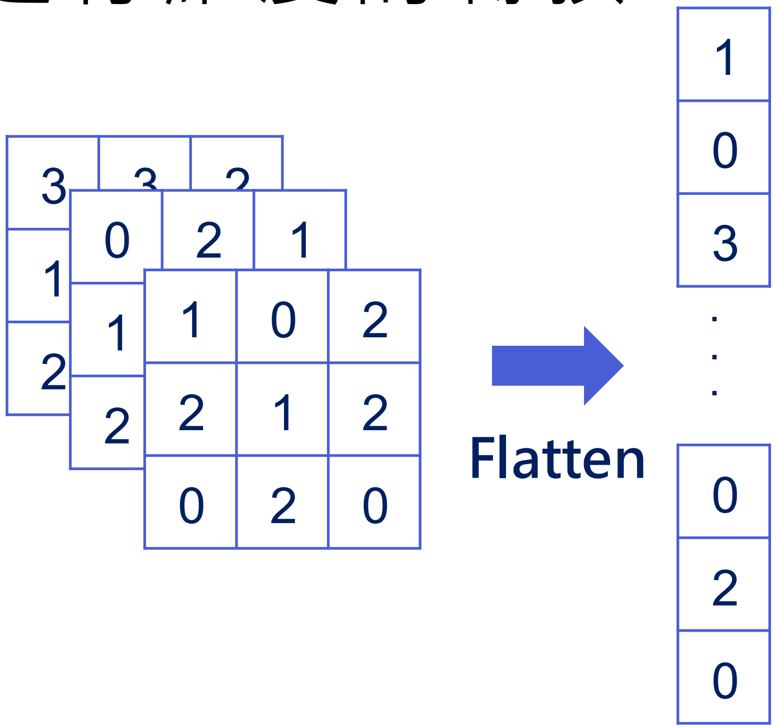
Flatten / GAP

Feedforward DNN

PIKACHU!!

攤平(Flatten)

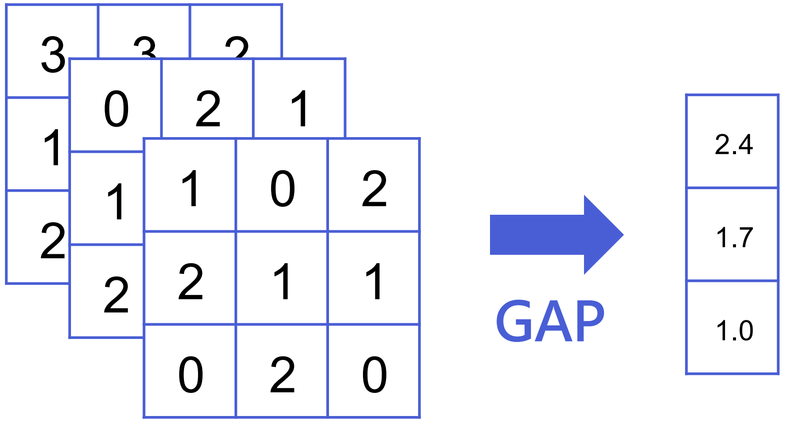
- 擔任在卷積層到全連接層之間的橋樑
- 將多維的輸入，攤平成一維輸出進行維度的轉換
- 過程中不需添加任何參數



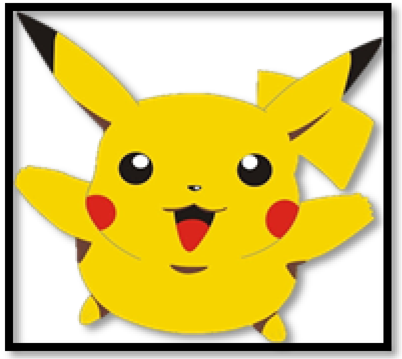
Global Average Pooling, GAP

- 攤平(Flatten)的方式同樣可能造成參數量較大，因此後續研究者提出以全局池化的方式將二維輸入轉成一維。

- 全局池化會針對每張特徵圖總結出一個數值，並且同樣不需額外參數
    - Global Max Pooling
    - Global Average Pooling



卷積神經網路架構



Convolutional Layer
&
Activation Layer

Pooling Layer

Convolutional Layer
&
Activation Layer

Pooling Layer

Can be performed many times

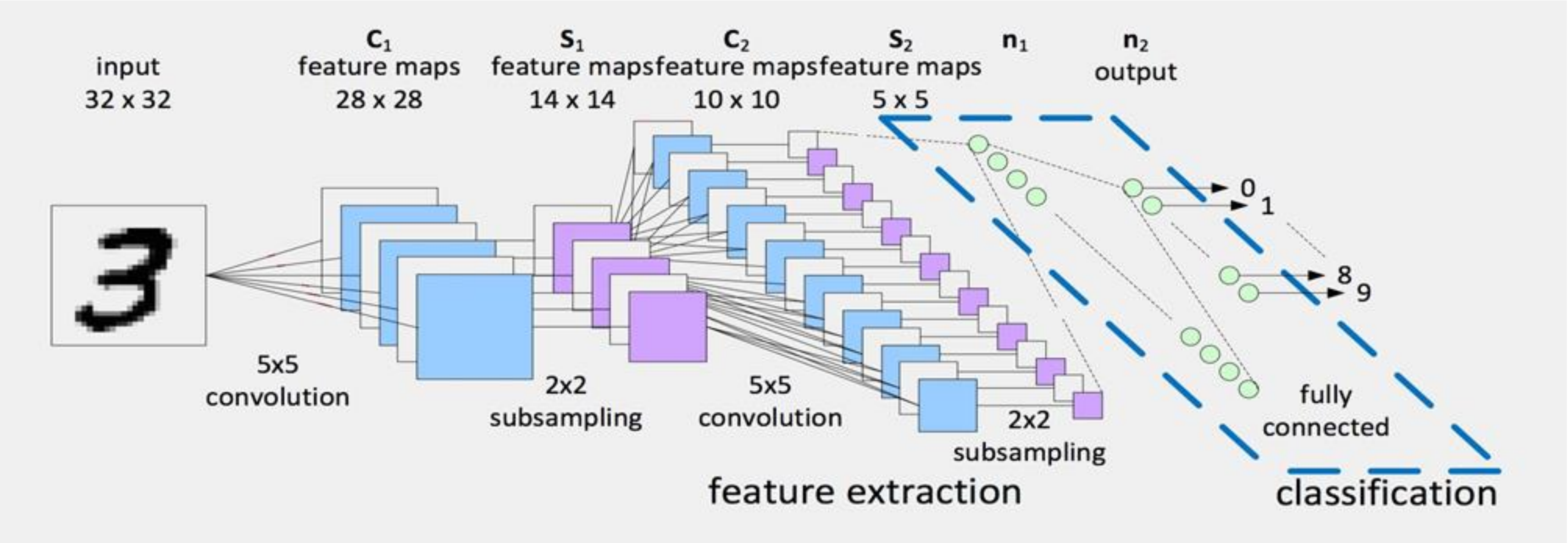
Flatten / GAP

Feedforward DNN

PIKACHU!!

全連接層(Fully Connected Layer)

- 因輸入維度已為一維，可直接以全連接層作為隱藏層或輸出層



程式實作練習

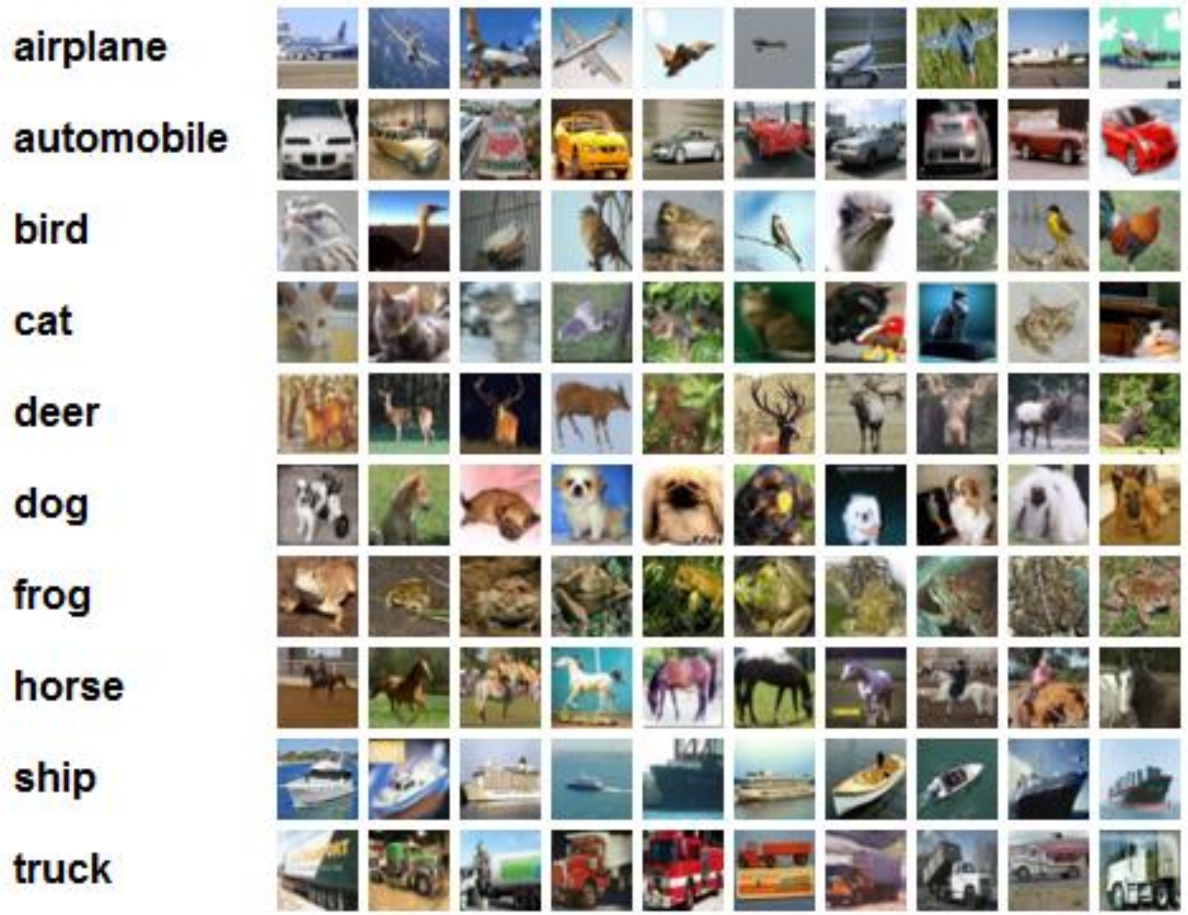
CIFAR-10 Dataset

- 60,000 (50,000 training + 10,000 testing) samples, 32x32 color images in 10 classes

- 10 classes
  - airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck

- Official website
  - https://www.cs.toronto.edu/~kriz/cifar.html

CIFAR-10 Dataset

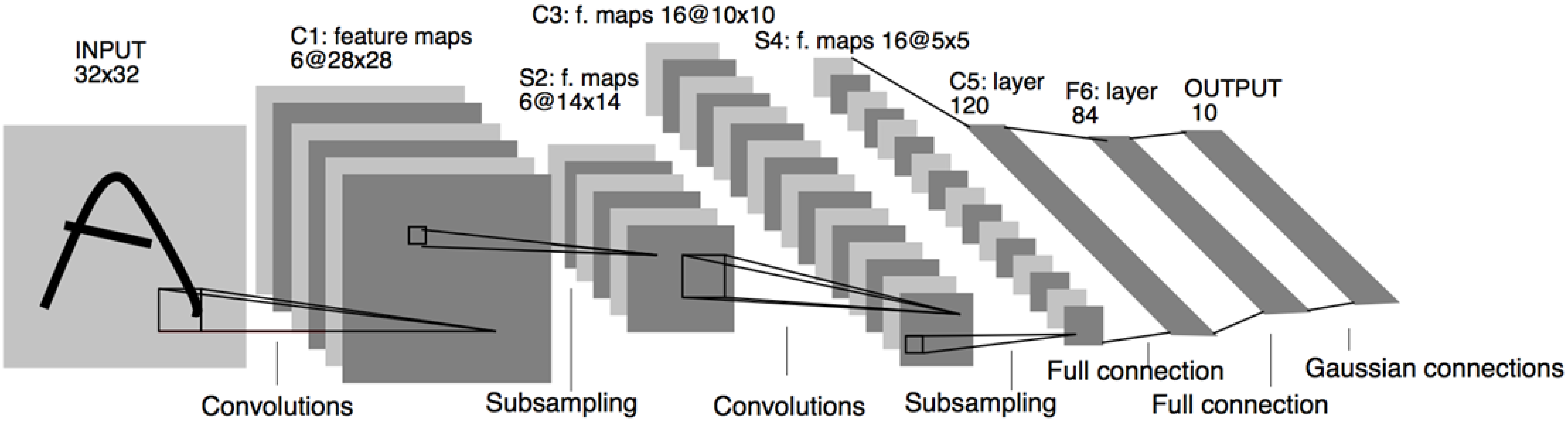


練習時間

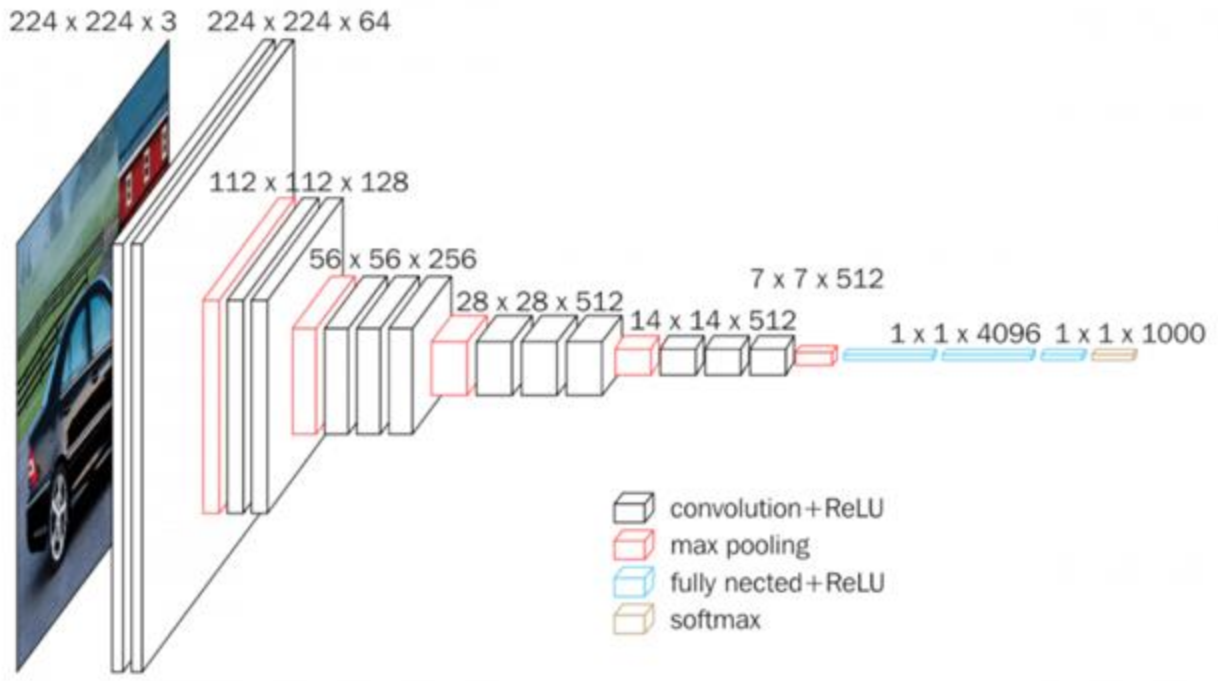
- 請使用 Cifar-10 資料集
- 建立 DNN 與 CNN 的模型
- 比較兩個模型的差異

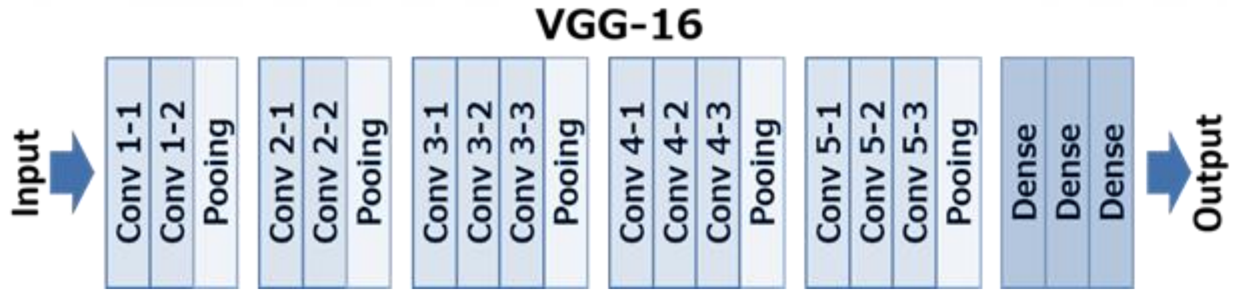
LENET-5

- 請以下面的LENET-5模型架構，建立一個與此相同的模型架構



VGG Net





- Research and Development (R&D) team
  - University of Oxford

- Architecture overview
  - Effect of CNN depths on accuracy
  - VGG16 and VGG19
  - Deeper than AlexNet
  - More accurate than AlexNet

資料來源：https://neurohive.io/en/popular-networks/vgg16/

Thank you