深度學習
Deep Learning

Convolutional Neural Networks (CNN)

Instructor: 林英嘉 (Ying-Jia Lin)
2025/10/08

Course GitHub

Slido # DL1008

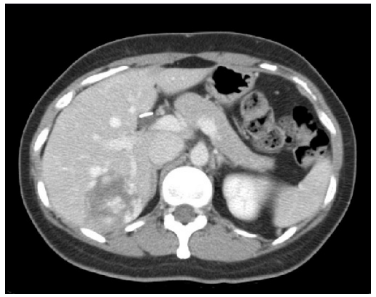
Outline

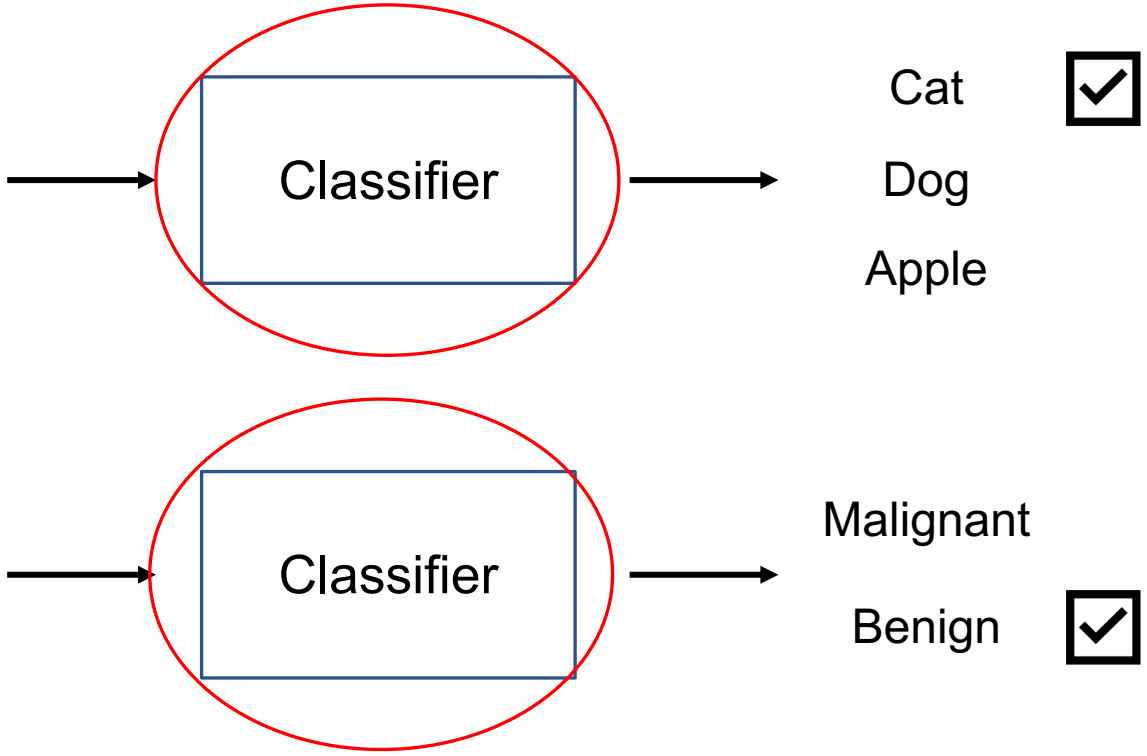
- Images [10 min]
- Convolutional Neural Networks [50 min]
- CV (Computer Vision) in PyTorch [40 min]
- HW2 [30 min]
- Quiz [20 min]

[Recep] 什麼是機器學習？(以電腦視覺為例)

We often use **CNN** for image classification!

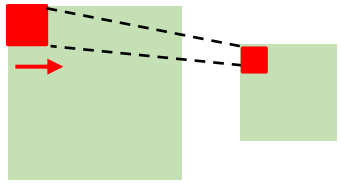
Machine Learning function!



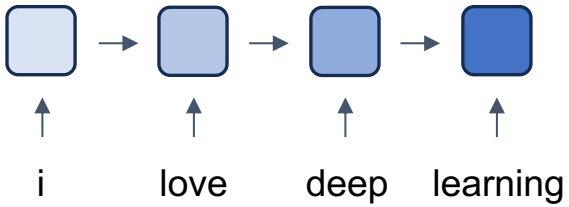


深度學習常見模型架構

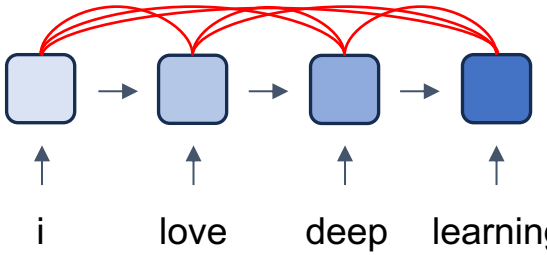
CNN
(著重局部資訊)



RNN
(著重記憶力)

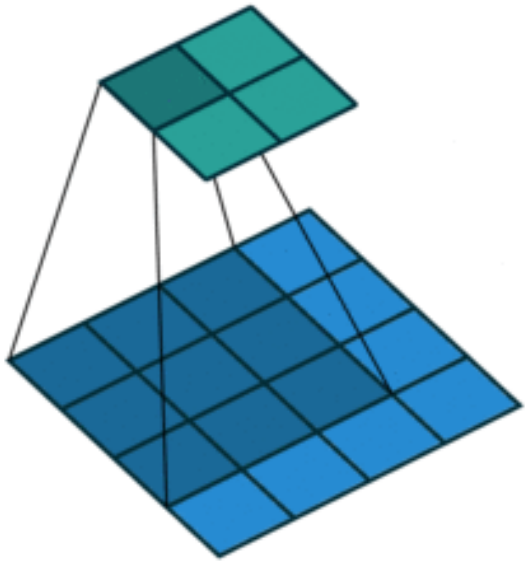


Transformer
(著重全局資訊與記憶力)

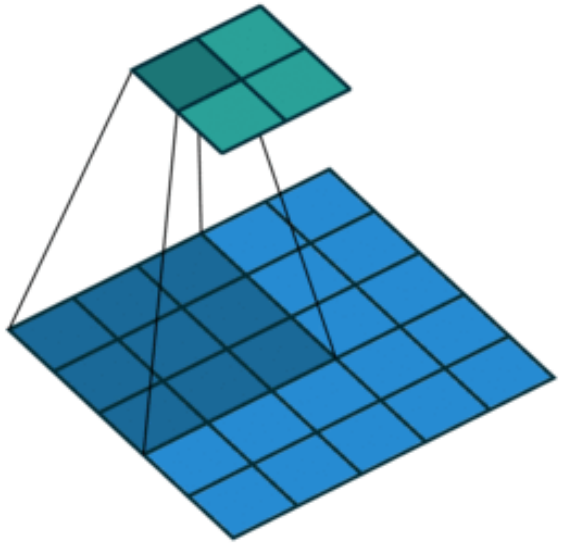


Examples of Convolutions

Padding = 0, stride = 1



Padding = 0, stride = 2



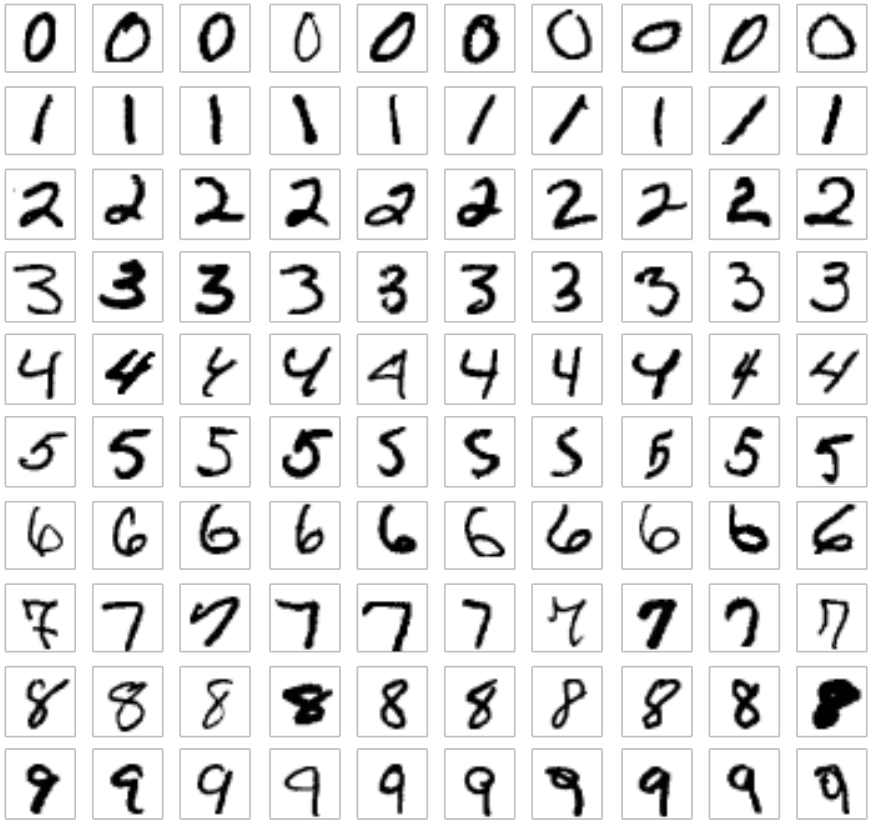
💡卷積 (convolution) 就是讓一個「小的矩陣」(filter) 在影像上滑動，進行局部加權平均 (局部區域的內積)，產生新的影像 (特徵圖，feature map)

Figure source:
https://hannibunny.github.io/mlbook/neuralnetworks/convolutionDemos.html

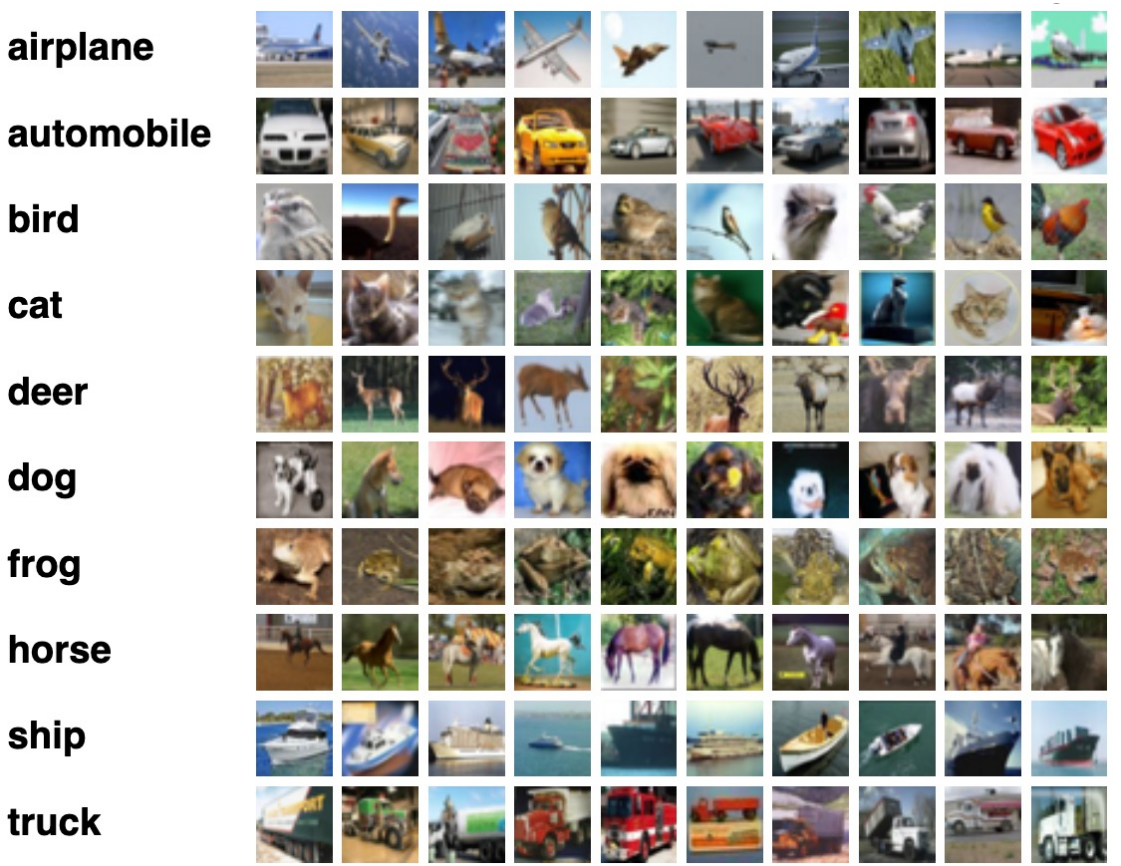
影像資料簡要介紹

Images

MNIST



CIFAR-10

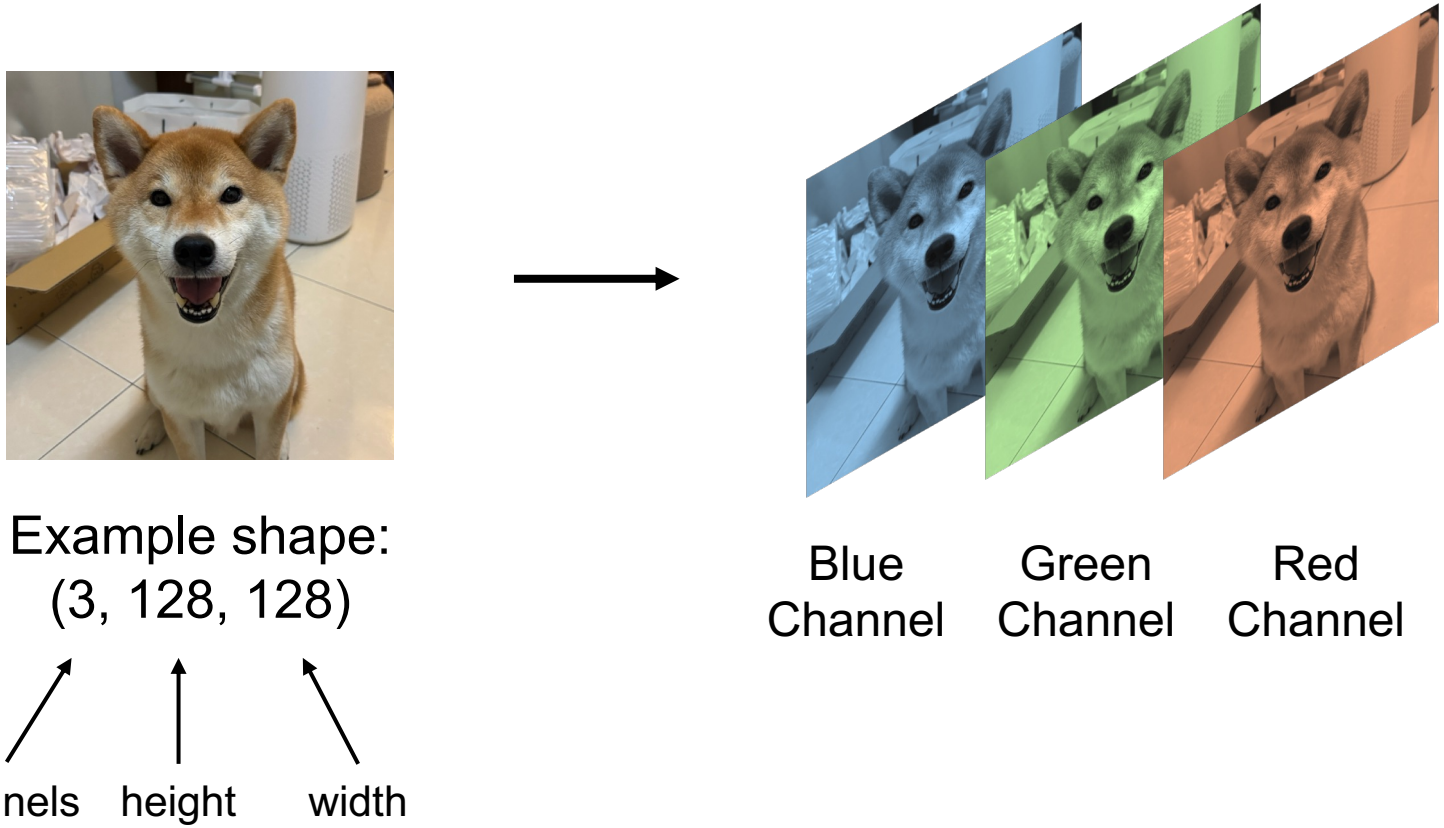


http://yann.lecun.com/exdb/mnist/

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

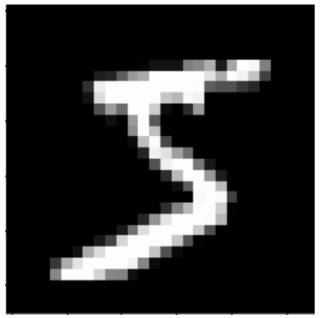
RGB Images (彩色影像)

- RGB 代表紅色 (Red)、綠色 (Green) 及藍色 (Blue)



Gray-scale Images (灰階影像)

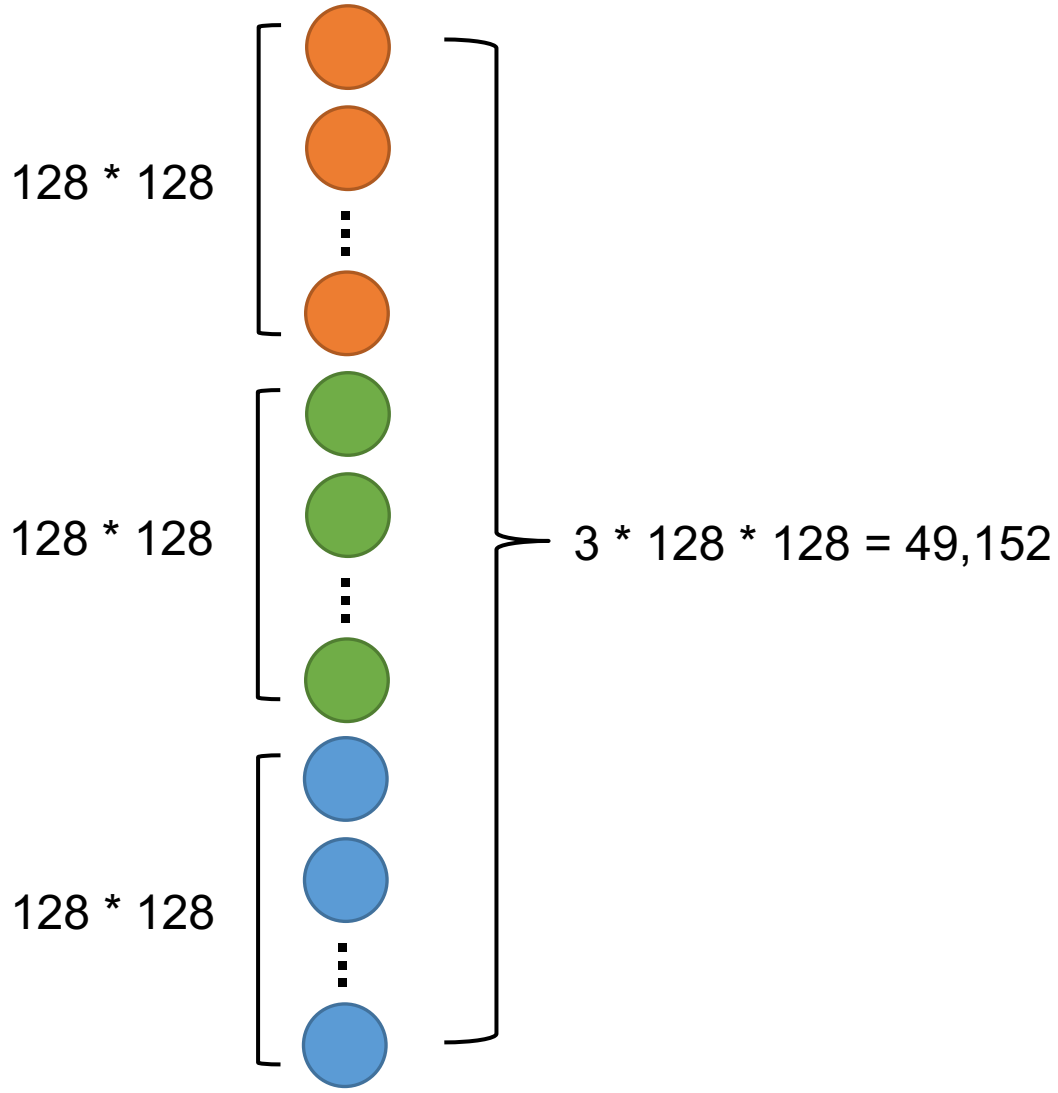
- 灰階影像只有一個 channel

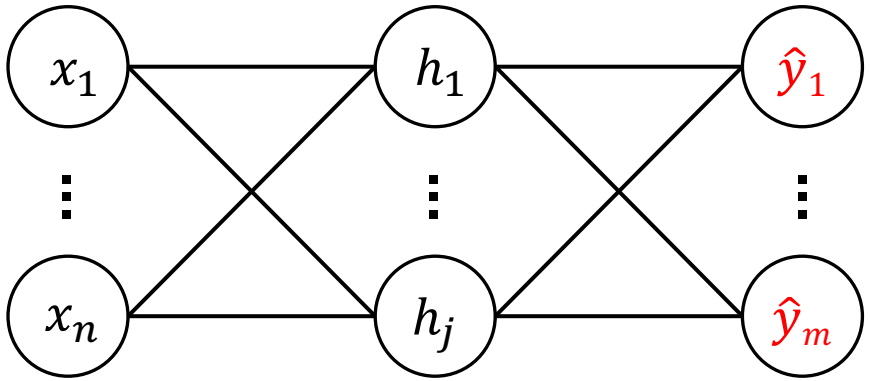


Example shape:
(1, 128, 128)

channels height width

Image inputs are too big for an MLP





n = 49,152

Convolutional Neural Networks

CNN

CNN Outline

- Convolutions (卷積層)
- Subsampling (池化層)
- Fully-connected Layer (全連接層)

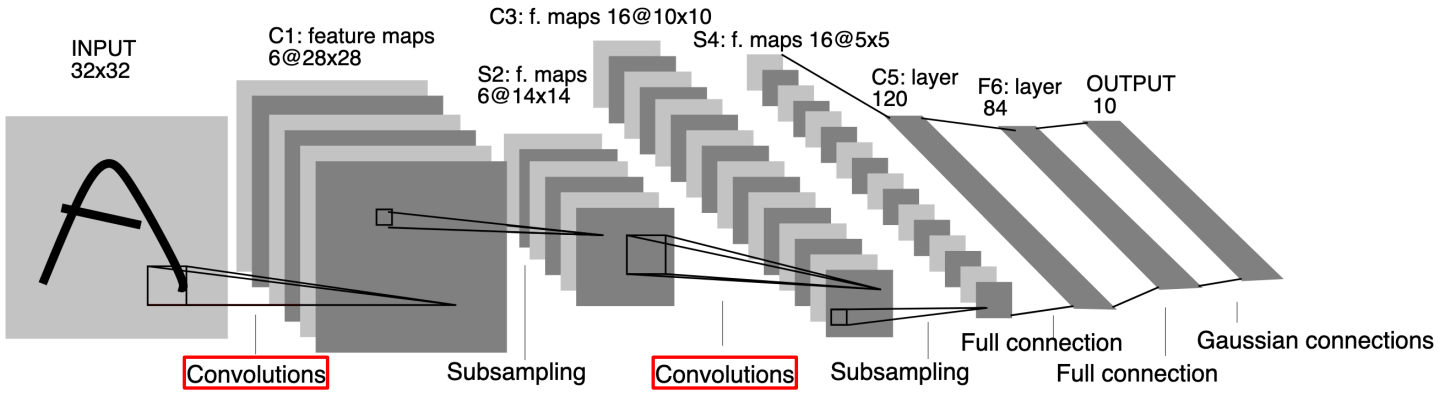


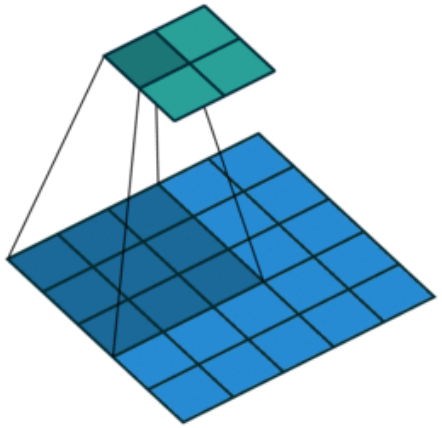
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Figure source: LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

卷積層有兩個重要的設定參數

https://docs.pytorch.org/docs/stable/generated/torch.nn.Conv2d.html

- stride (int): 步長，代表卷積一次走幾個像素點
- padding (int): 影像的四個邊要不要補 0



Padding = 0, stride = 2

Figure source:
https://hannibunny.github.io/mlbook/neuralnetworks/convolutionDemos.html

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

(白色代表原始影像區域，橘紅色代表padding)

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1	1

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1	1	2

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1	1	2
-1			

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1	1	2
-1	1		

Convolutions (stride = 1)

沒有 padding

Stride = 1

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1	1	2
-1	1	1	0
0	0	1	2
0	0	1	2

feature map

Convolutions (stride = 1)

沒有 padding

Stride = 2

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1

Convolutions (stride = 2)

沒有 padding

Stride = 2

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1

Convolutions (stride = 2)

沒有 padding

Stride = 2

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1
0	

Convolutions (stride = 2)

沒有 padding

Stride = 2

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

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

1	1
0	1

Convolutions (stride = 2, padding = 1)

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

有 padding

0		

Convolutions (stride = 2, padding = 1)

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

有 padding

0	1	

Convolutions (stride = 2, padding = 1)

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

有 padding

0	1	0

Convolutions (stride = 2, padding = 1)

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

有 padding

0	1	0
0		

Convolutions (stride = 2, padding = 1)

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

0	1	0
0	1	

Convolutions (stride = 2)

padding=1

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

0	1	0
0	1	0

Convolutions (stride = 2)

padding=1

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

0	1	0
0	1	0
0		

Convolutions (stride = 2)

padding=1

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

Padding

Stride = 2

1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

0	1	0
0	1	0
0	0	

Convolutions (stride = 2)

padding=1

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

Padding

Stride = 2

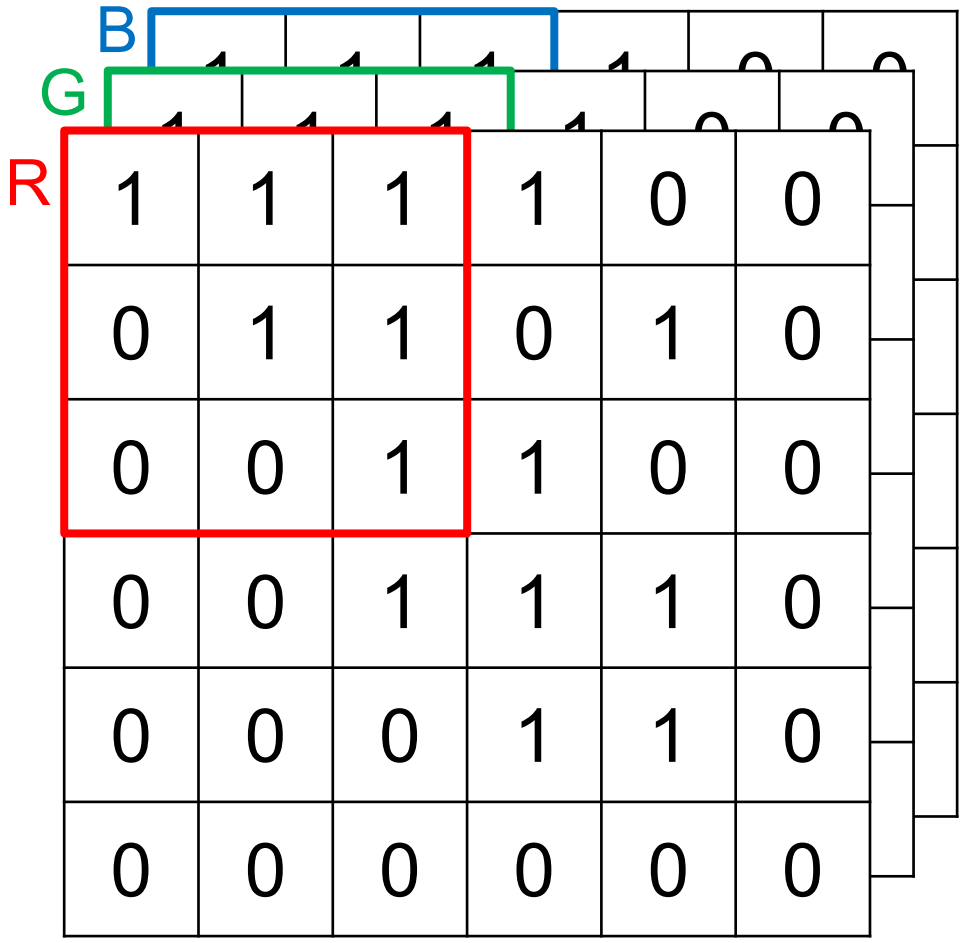
1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter

0	1	0
0	1	0
0	0	2

Convolutions (RGB)

filters 數值可能都是不同的！(為了方便計算，簡報使用一樣的數字)



1	0	0
0	1	0
0	0	-1

(R) Filter

1	0	0
0	1	0
0	0	-1

(G) Filter

1	0	0
0	1	0
0	0	-1

(B) Filter

1	1	1
0	1	1
0	0	1

1	0	1
0	1	1
0	0	1

1	0	0
0	1	1
1	0	1

先各自與Filter做內積後相加

3	0	0
0	3	0
0	0	-3

3

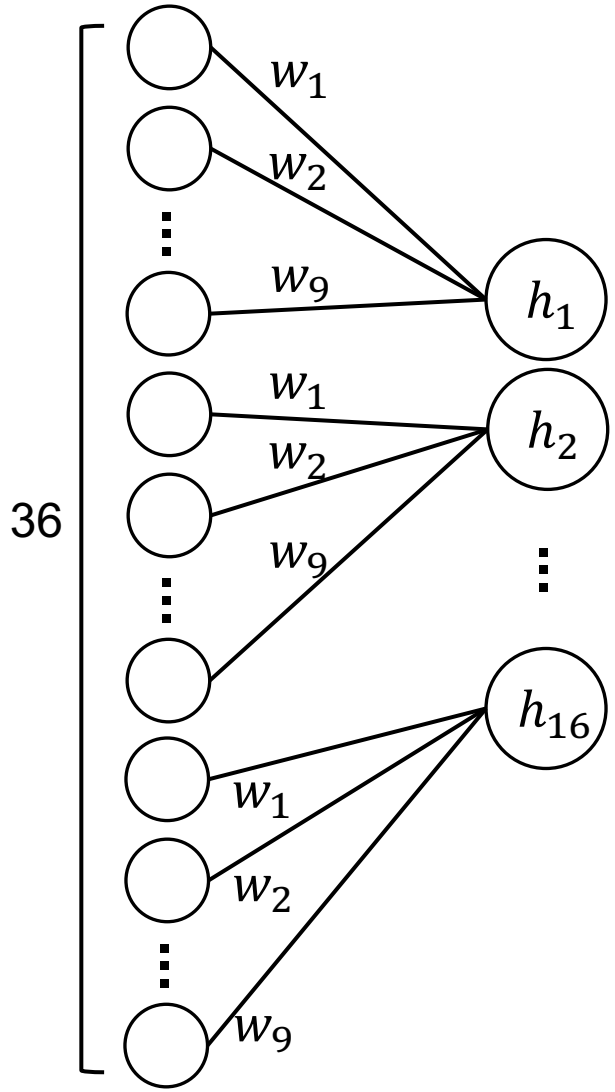
Filters 數目與 output_channels 的關係

- Filters 的數目可以調整
- 對輸入層來說，1 個 filter 照顧 3 個 channels 的資料，輸出 1 個 feature map
- 使用 64 個 filters，就會輸出 64 個 feature maps
- 可調整參數為: num_filters * input_channels * filter_size * filter_size

就是torch.nn.Conv2d的out_channels

⭐ Convolutional layer is not fully-connected

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



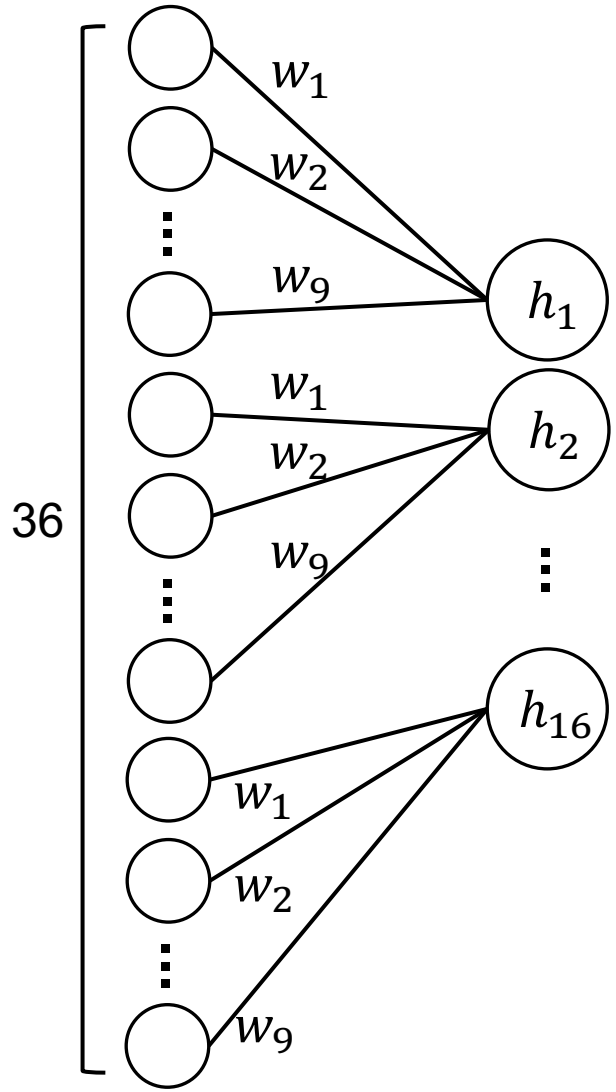
1	**0**	**0**
0	**1**	**0**
0	**0**	**-1**

Filter, stride=1

h_1	h_2	h_3	h_4
h_5	h_6	h_7	h_8
h_9	h_{10}	h_{11}	h_{12}
h_{13}	h_{14}	h_{15}	h_{16}

feature map

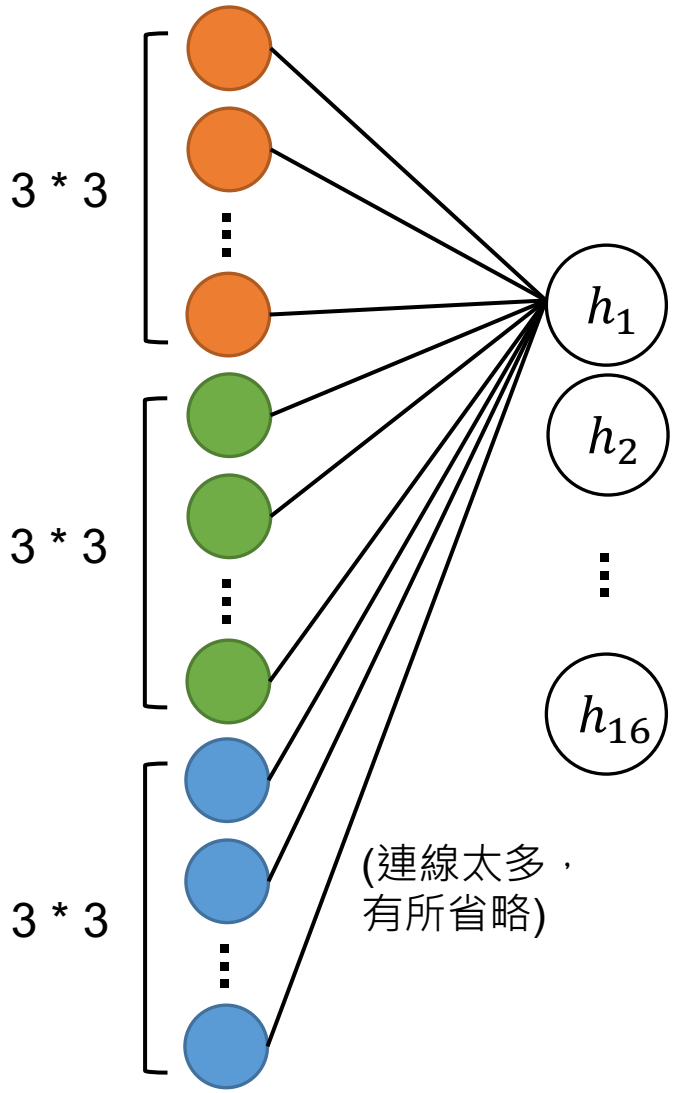
Convolutional layer vs. FC layer 參數量



FC: fully-connected

Grayscale images	參數量比較 (不算 bias 數)
FC layer	36 * 16 = 576
Conv layer	3 * 3 = 9

Convolutional layer vs. FC layer 參數量

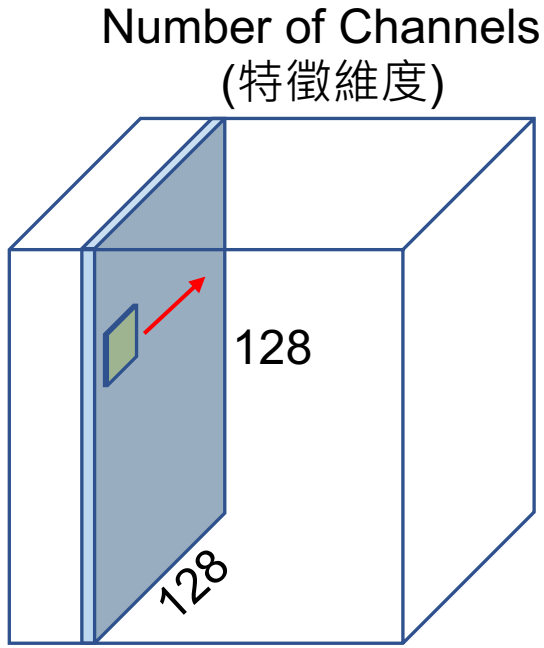


FC: fully-connected

RGB images	參數量比較 (不算 bias 數)
FC layer	3 * 128 * 128 * 16 = 786,432
Conv layer (num_filter = 1)	3 * 3 * 3 = 27
Conv layer (num_filter = 16)	3 * 3 * 3 * 16 = 432

Implementation in PyTorch

- 我們使用 `torch.nn.Conv2d`
- Why not 3d?

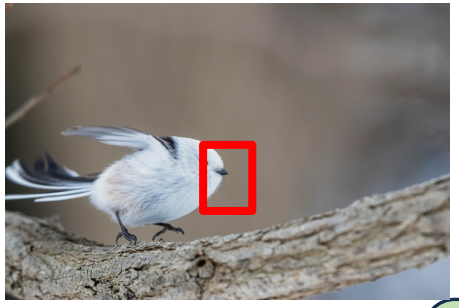


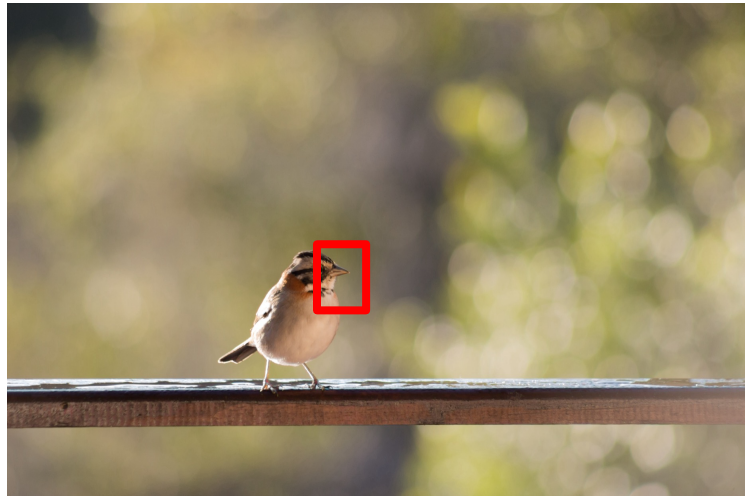
2d 卷積僅沿著空間維度 (Height, Width) 進行

[Recap] 深度學習可以減少 Feature Selection

找出重要
的特徵

Machine Learning
Classifier





透過 NN (例如卷積層) 找出重要的特徵

Deep Learning
Classifier

找不到鳥嘴 -> 不是小鳥

CNN Outline

- Convolutions (卷積層)
- Subsampling (池化層)
- Fully-connected Layer (全連接層)

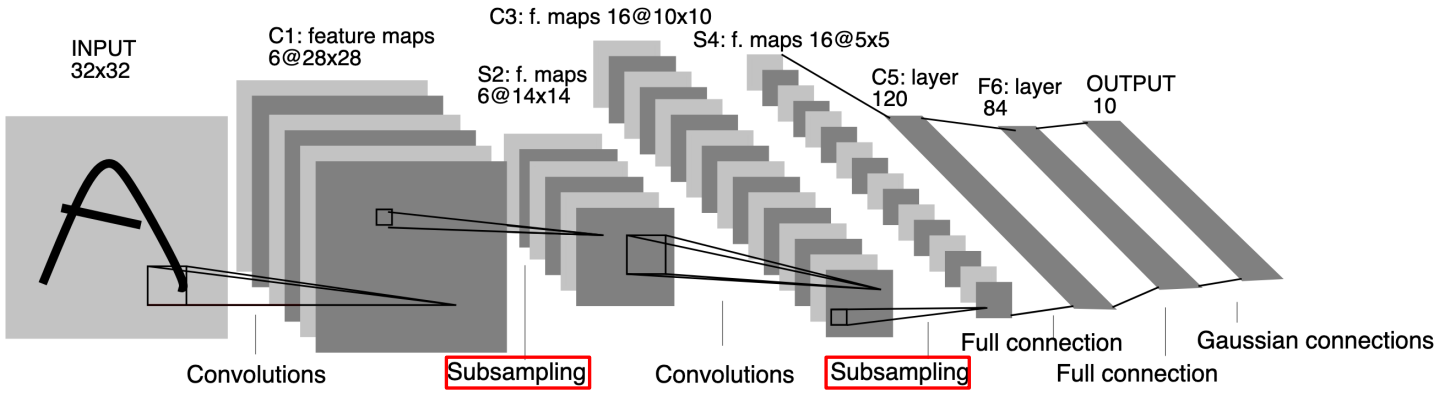


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Figure source: LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

Pooling (example of Max Pooling)

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

参數：

- kernel_size=2
- stride = 2

3	2
1	2

Pooling (example of Average Pooling)

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

參數：

- kernel_size=2
- stride = 2

1	1
0.25	1

Pooling (example of Max Pooling)

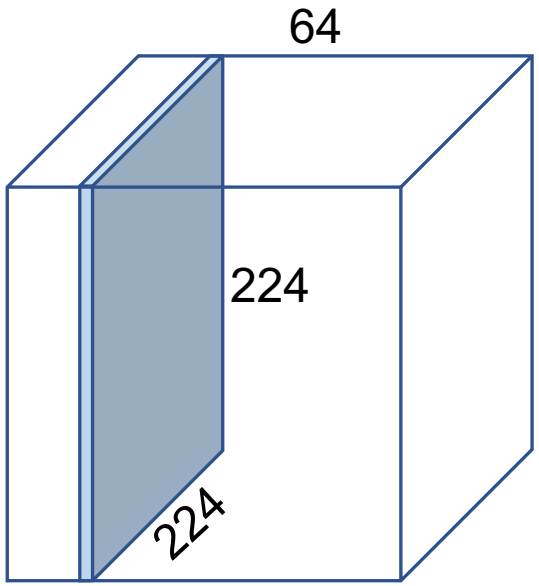
1	3	1	2
-1	1	1	0
0	1	1	0
0	0	1	2

参數：

- kernel_size=2
- stride = 1

3	3	2
1	1	1
1	1	2

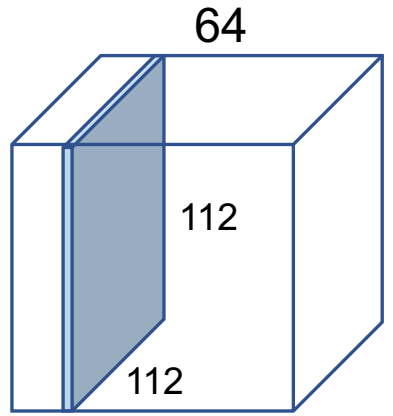
2D Pooling on RGB Images



224 x 224 x 64
(H x W x C)

参數：

- kernel_size=2
- stride = 2



112 x 112 x 64
(H x W x C)

https://docs.pytorch.org/docs/stable/generated/torch.nn.MaxPool2d.html

[Summary] Meaning of Pooling

- Pooling 目的在於：強制保留最重要的特徵
- 優點：加速 CNN 的計算
- 缺點：不保證模型效能真的會因此變好

CNN Outline

- Convolutions (卷積層)
- Subsampling (池化層)
- Fully-connected Layer (全連接層)

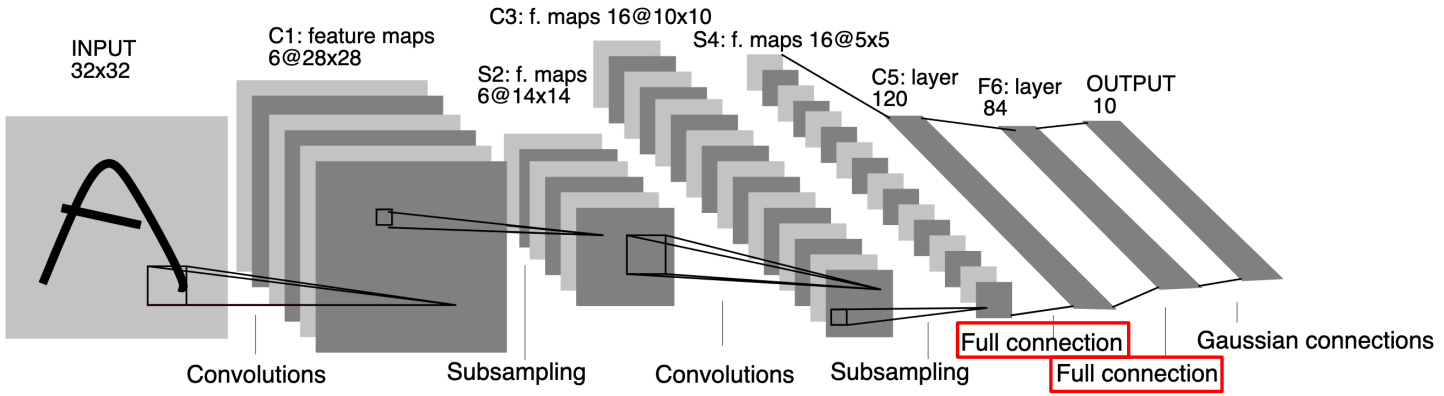


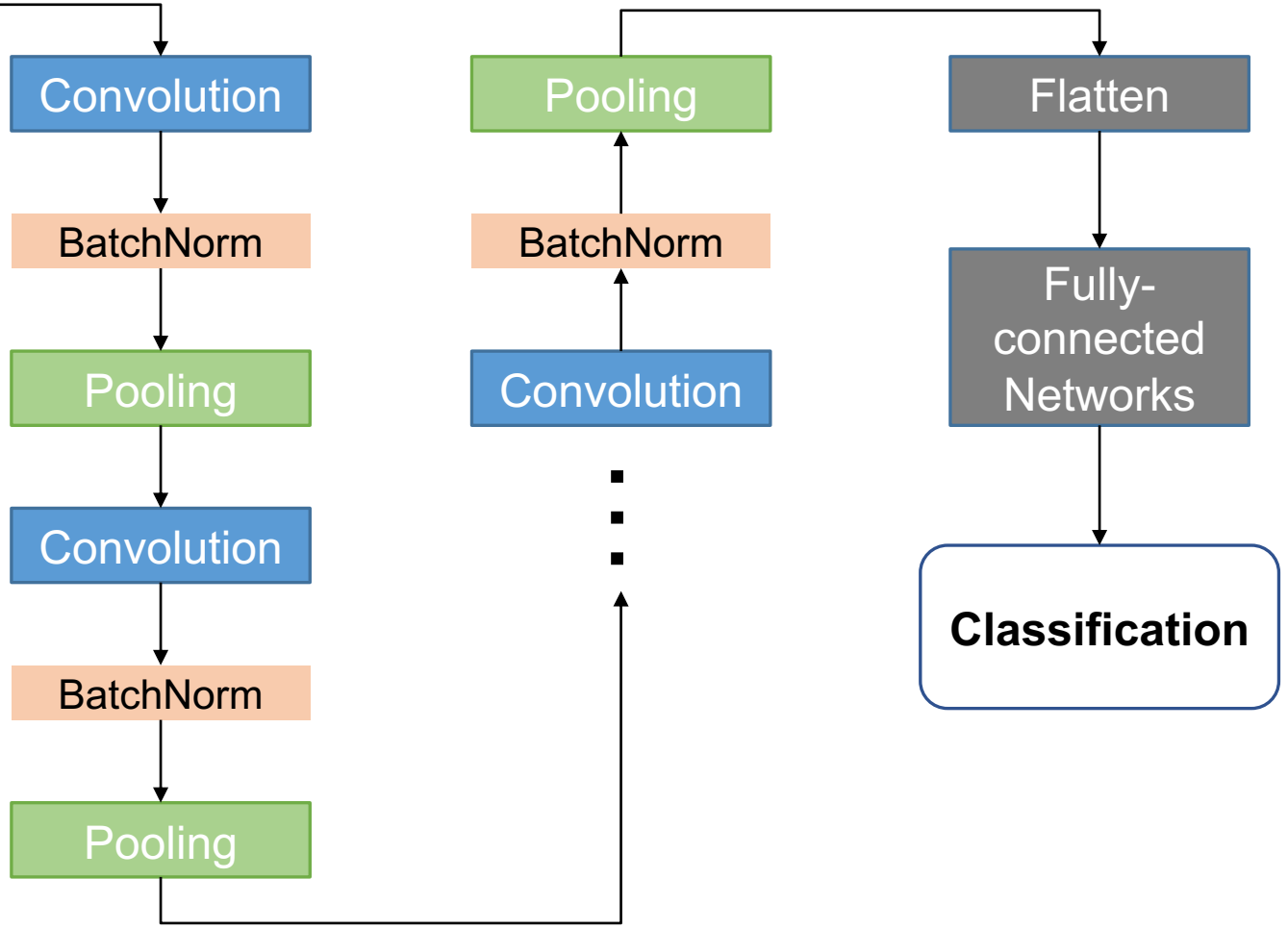
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Figure source: LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

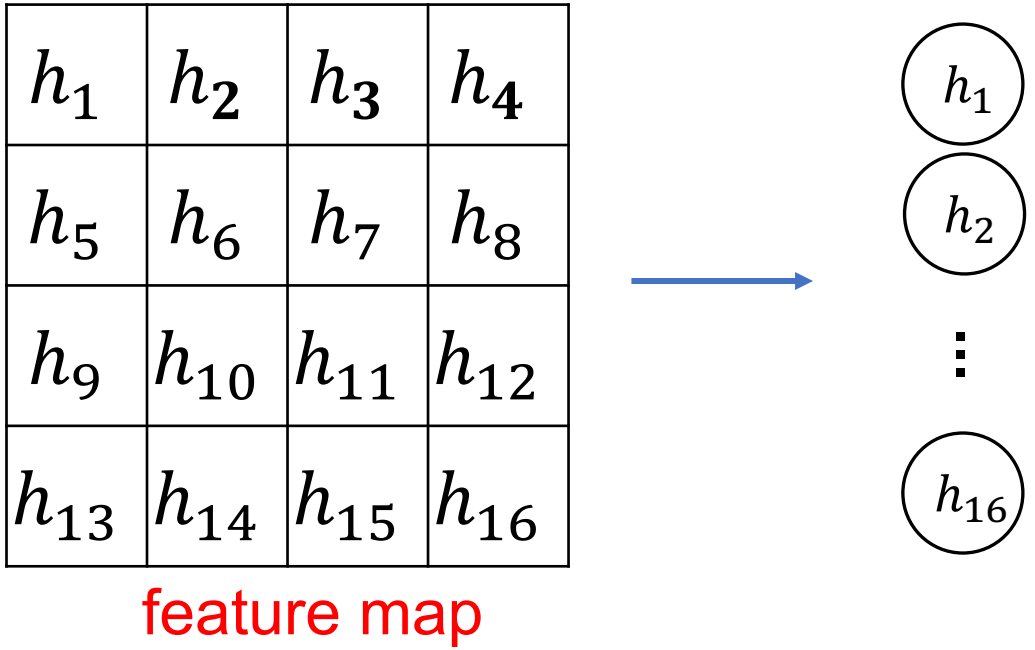
CNN 的結局

- 通常不會只有一層的 Convolutional Layer
- 隨著層數增加，通常 Feature map 會越來越小，因為：
  - Convolutional Layer 可能會縮小 feature map (如果沒有 padding)
  - Pooling 也會縮小 feature map
- 經過了多次的 Convolution + Pooling 後的 feature maps 為壓縮後的**重要資訊**

The whole CNN



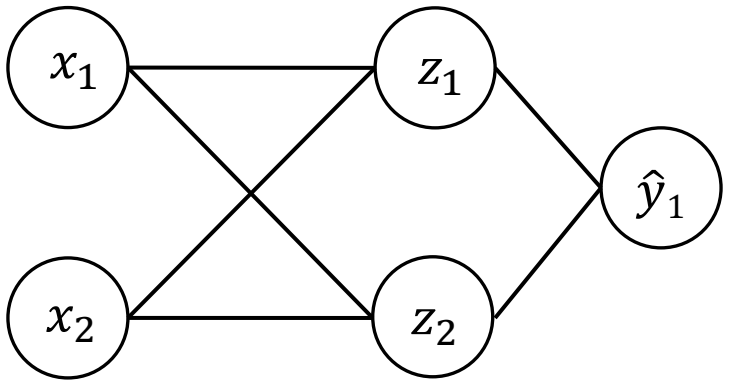
Flatten



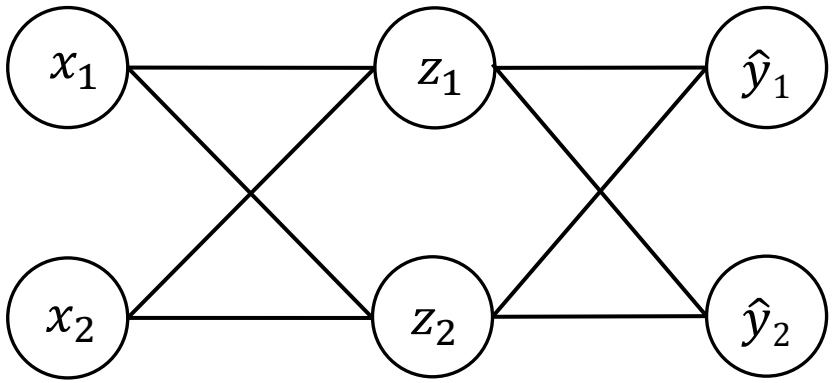
- Why Flatten? Feature maps 最後的大小通常不會等於類別數目，因此需要接上 fully-connected layer

Fully-connected Networks

- 常被稱為全連接層 (Fully-connected layers, FC)
- Multi-layer perceptron (MLP) 即是由 FC 所構成
- 注意! CNN 不是全連接層



Regression / Binary Classification



Classification

CNN Summary

- 為影像資料量身定做，用來捕捉空間資訊 (Spatial Information)
- Parameter sharing (參數共享機制)
  - 降低用 MLP 來進行影像任務的參數量

ConvNets are everywhere

Object Detection with YOLO

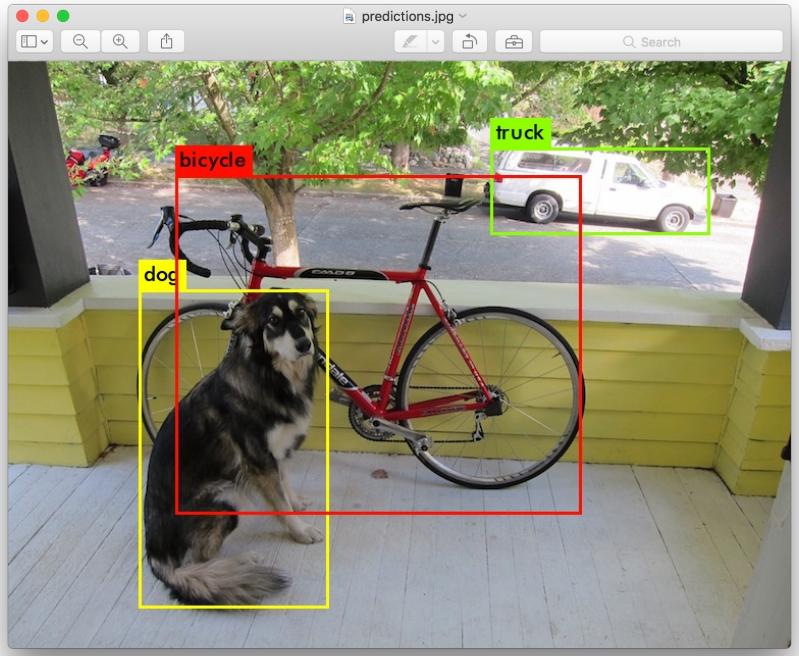


Figure source: https://pjreddie.com/darknet/yolo/

Style Transfer with CNN

Figure source: Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "Image style transfer using convolutional neural networks." CVPR. 2016.

Advanced CNNs

- [ResNet] Deep Residual Learning for Image Recognition
  - https://arxiv.org/abs/1512.03385
- [DenseNet] Densely Connected Convolutional Networks
  - https://arxiv.org/abs/1608.06993
- https://slazebni.cs.illinois.edu/spring17/lec04_advanced_cnn.pdf

Additional Links

- **PyTorch CNN example**:

  https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html
- **Batch Normalization:**
  - https://wandb.ai/wandb_fc/GroupNorm/reports/Group-Normalization-in-Pytorch-With-Examples---VmlldzoxMzU0MzMy
  - https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739/

Thank you!

Instructor: 林英嘉

✉ yjlin@cgu.edu.tw

TA: 劉美辰

✉ m1461014@cgu.edu.tw