經典模型

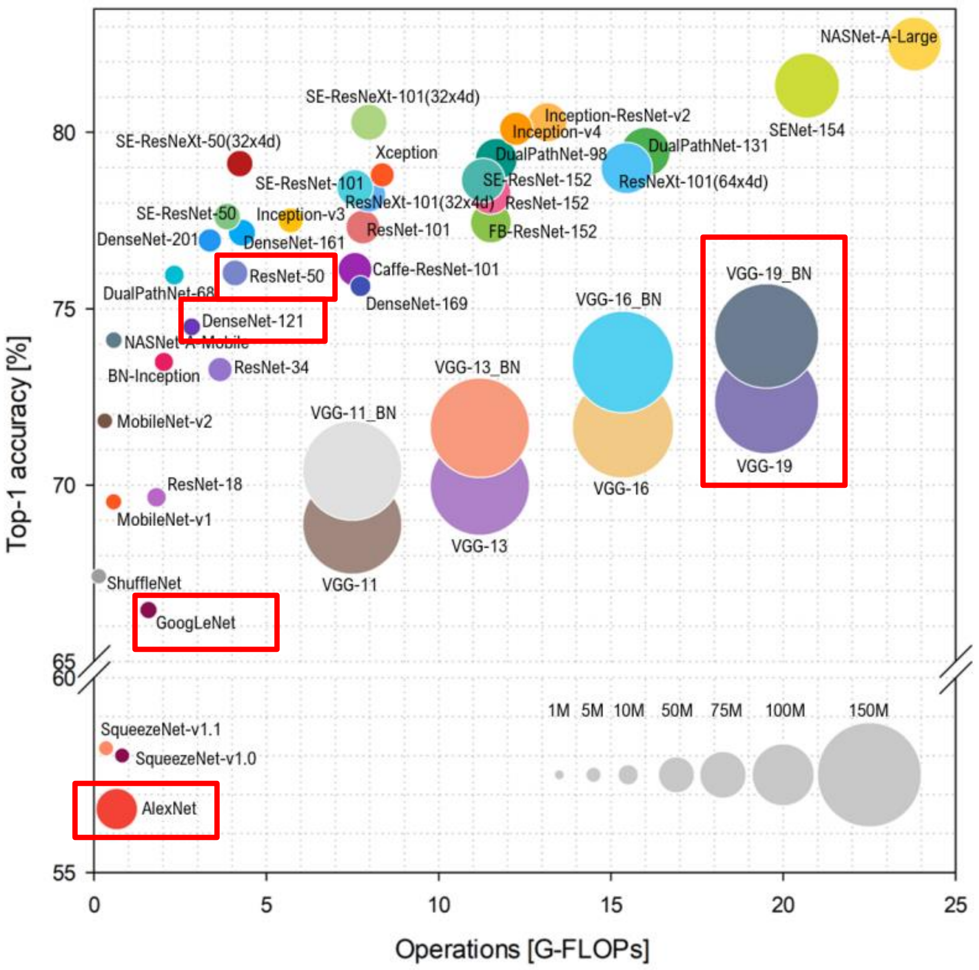
2025/07/01

PRESENTED BY AI Foundation

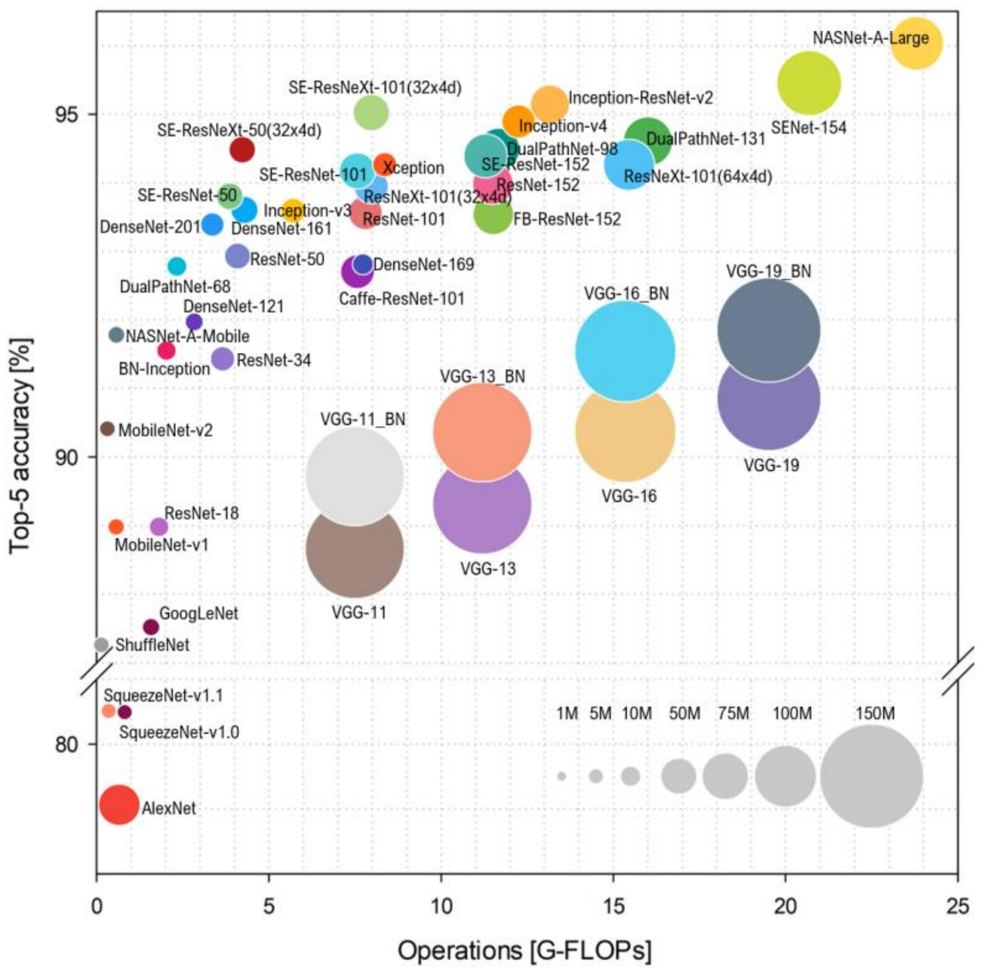
CREATED FOR

INNOLUX

2019年前的CNN模型大比較



Top-1 acc. vs. #Operations with #parameters



Top-5 acc. vs. #Operations with #parameters

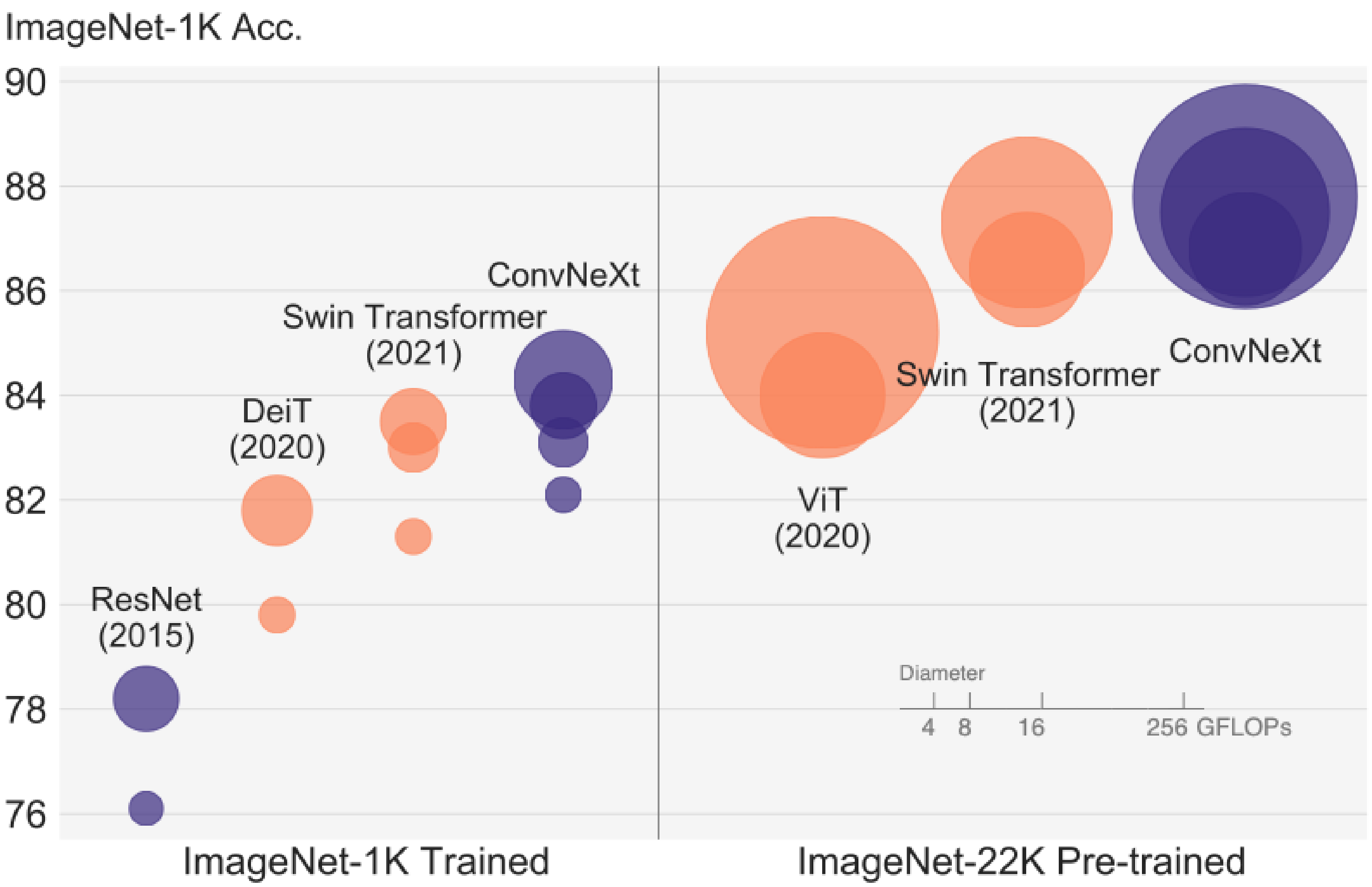
資料來源: 1810.00736.pdf (arxiv.org)

後CNN時代 2019年後

一個好的CNN模型需要有的條件

- 模型精簡
- 高精準度
- 精簡且高精準

計算速度 – 快
參數量 – 少
精準度 – 強



資料來源: CNN Model ImageNet Comparison

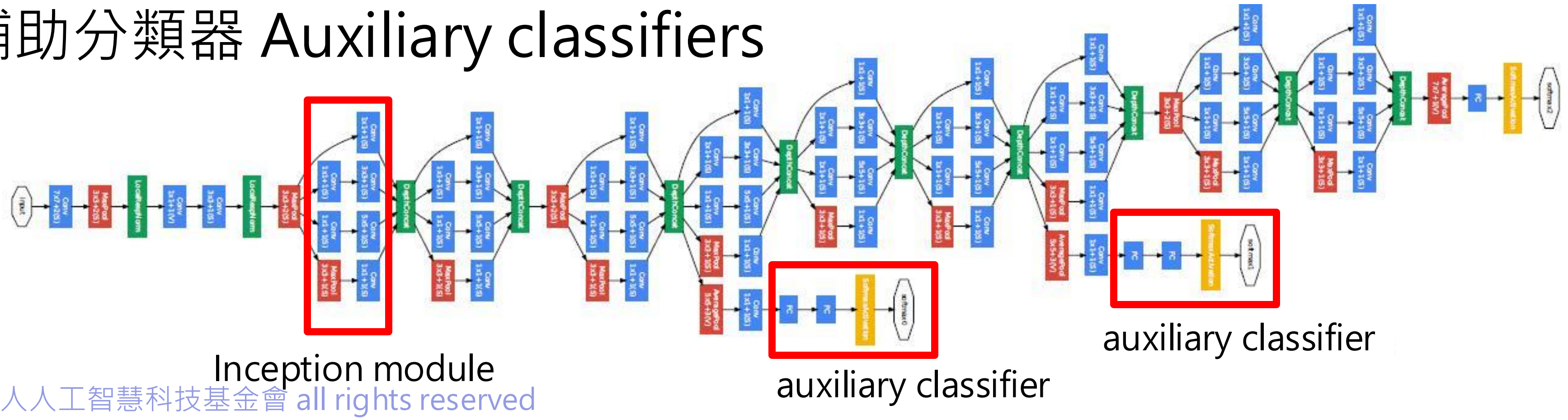
經典模型

目錄

GoogleNet

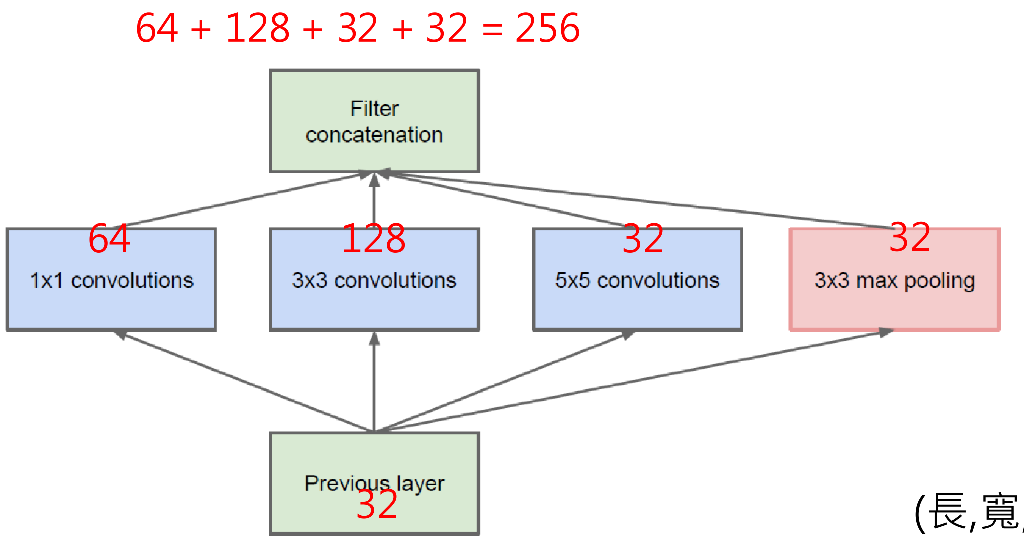
GoogleNet (Inception V1)

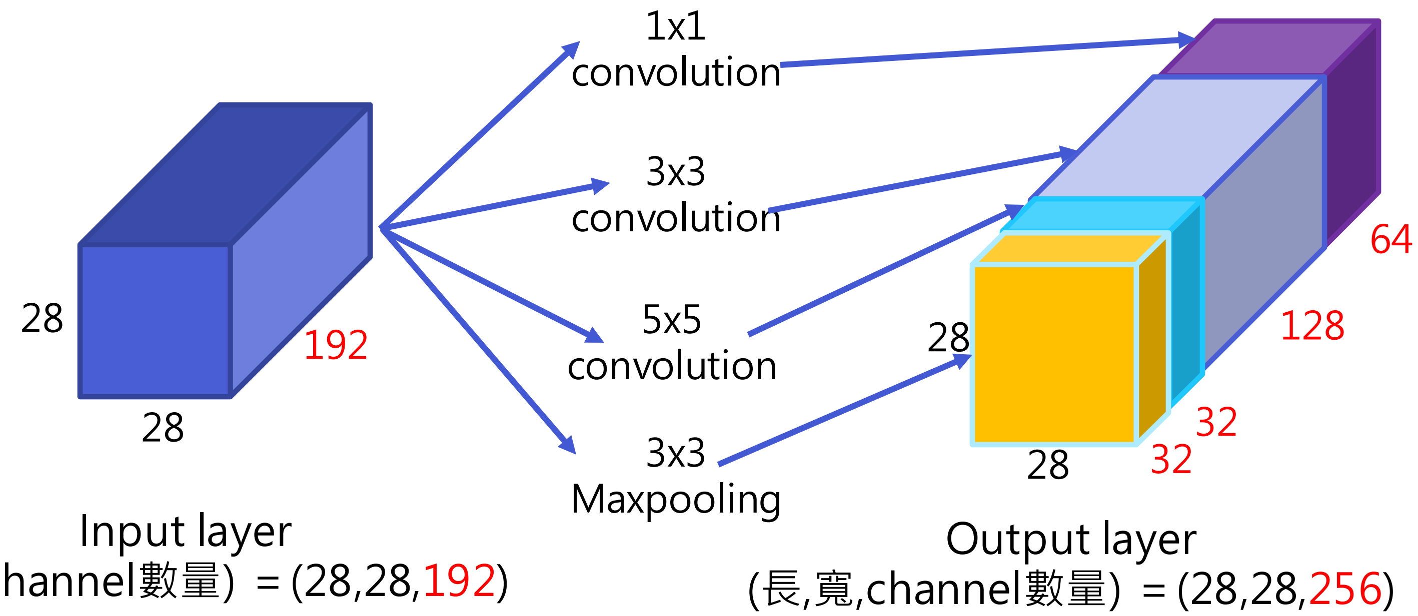
- 主要架構
  - 22 層
  - 準確率比 VGGNet-16 高
  - 參數量從 AlexNet 的 6000萬 縮減至 400萬
  - Inception module
  - 輔助分類器 Auxiliary classifiers



GoogleNet: Inception module

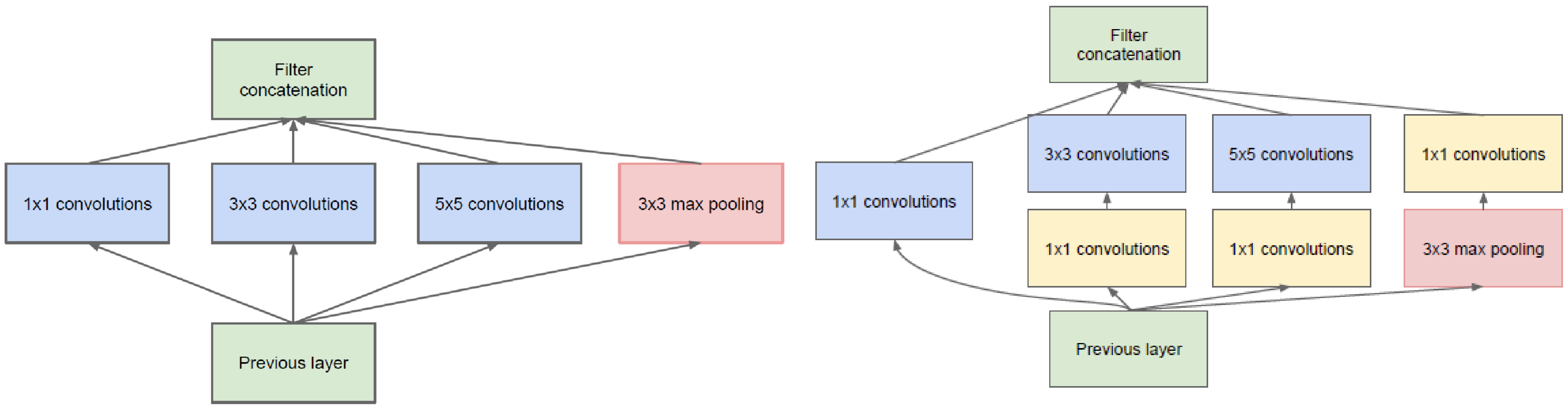
- Inception Module: 同時看到不同尺度的特徵
- 使用不同的 filter 大小尋找不同尺度的特徵
  - Filter 大小 : 1x1, 3x3, 5x5 filter 和一個 3x3 池化
- 合併所有不同尺度的特徵圖





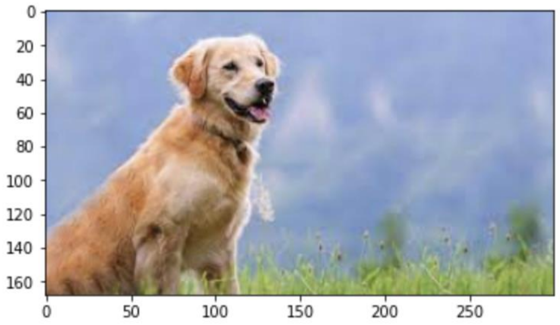
GoogleNet: Inception module

- 利用運算成本較低的 1x1 卷積層減少特徵圖的數量

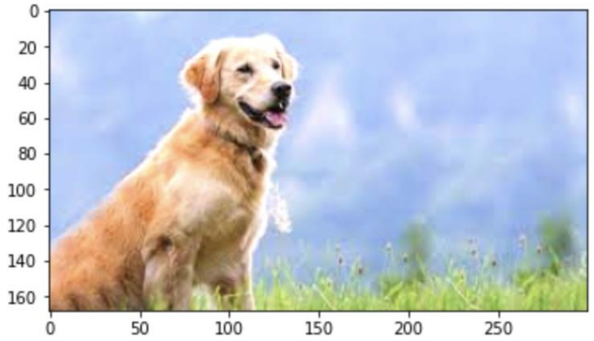


1 x 1 Convolution

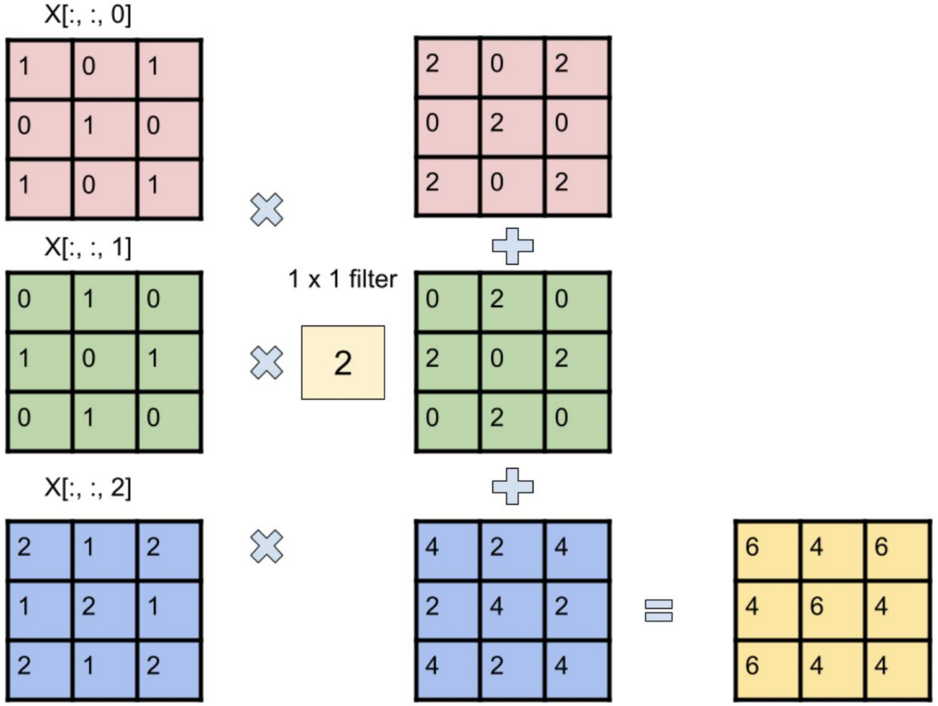
- 1x1 卷積層可以做到對 channel 維度的降維或升維
- 等效於對每張特徵圖都乘上一個數值



Raw image

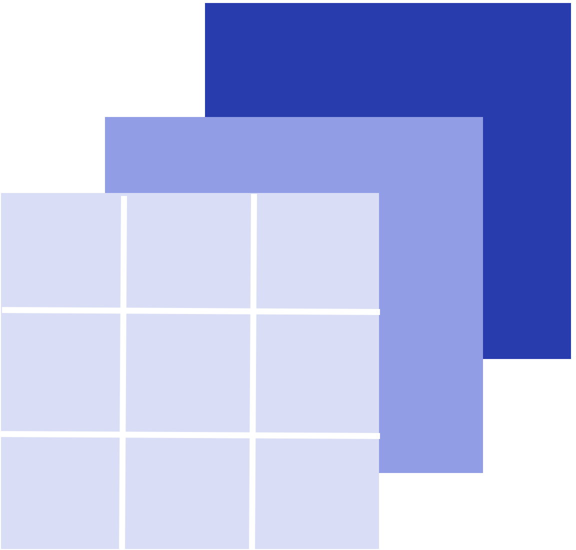


Raw image * 1.2

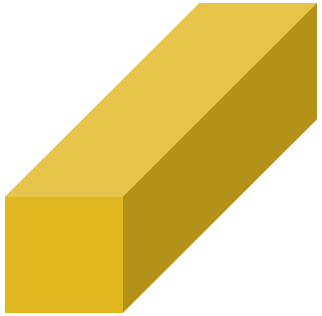


1 x 1 Convolution

nn.Conv2d(in_channels=in_channels, out_channels=filter_num, kernel_size=1)



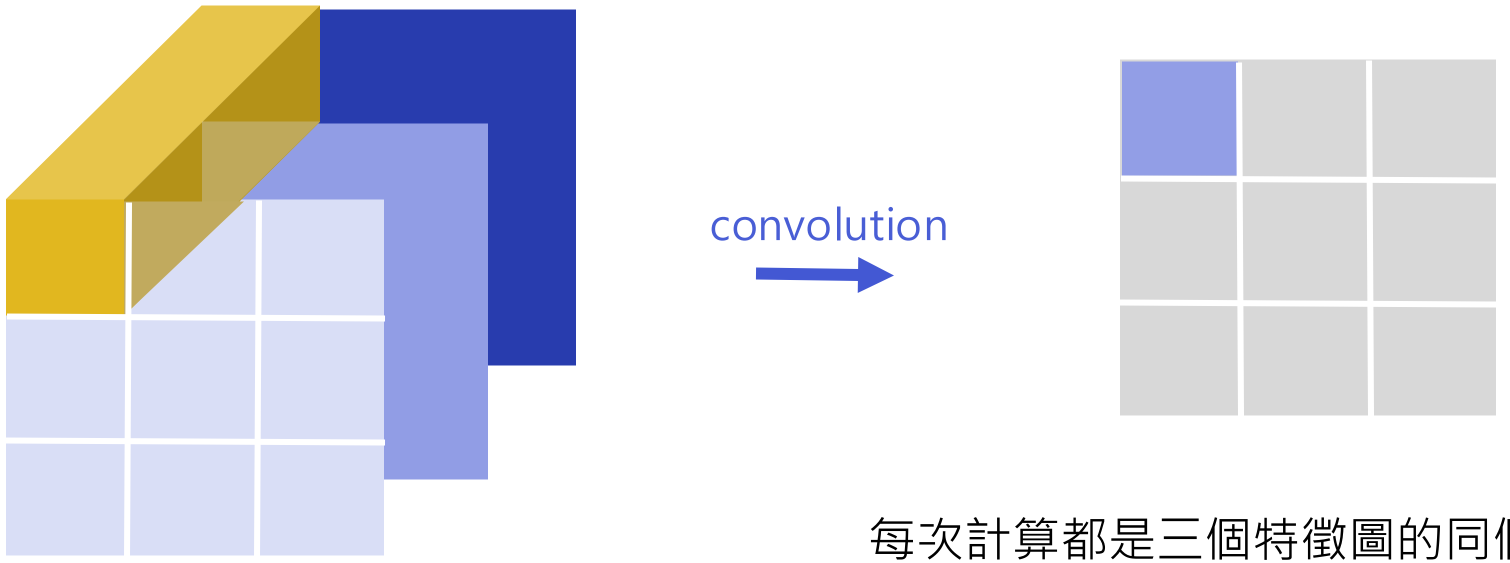
Feature maps
3 x 3 x 3



Filter
1 x 1 x 3

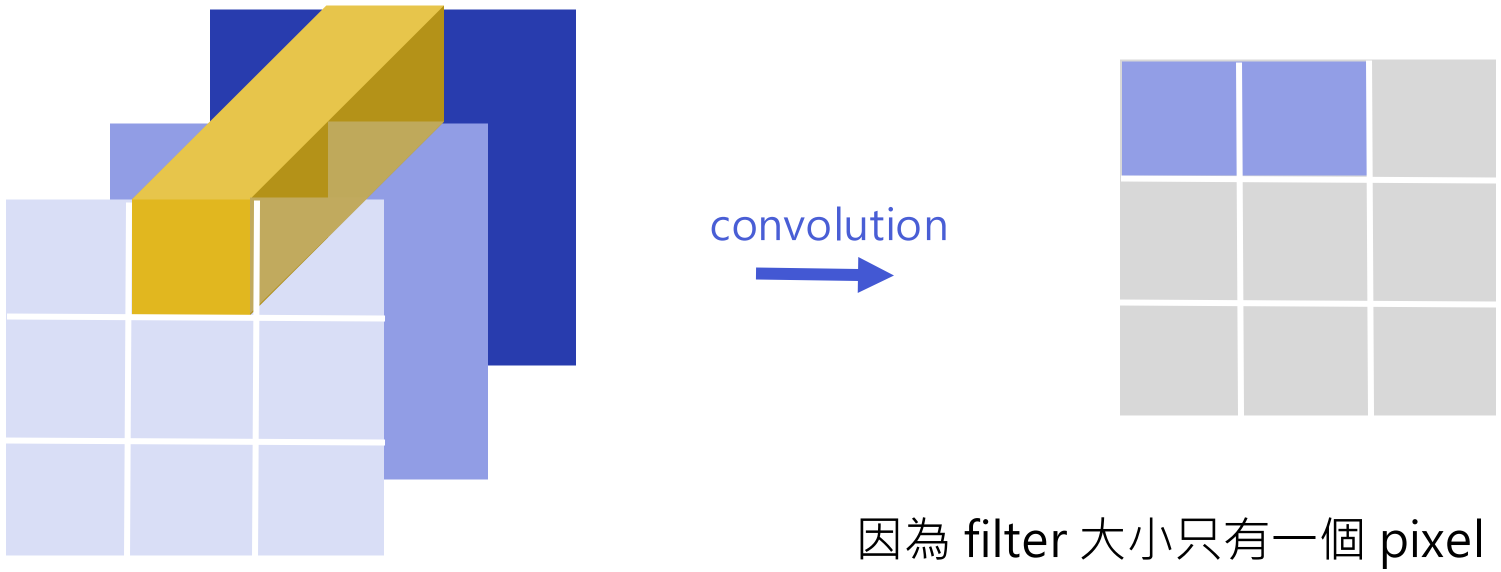
如果 input 特徵圖有三張，那麼每個 1 x 1 filter 裡面就會有三個數值。

1 x 1 Convolution



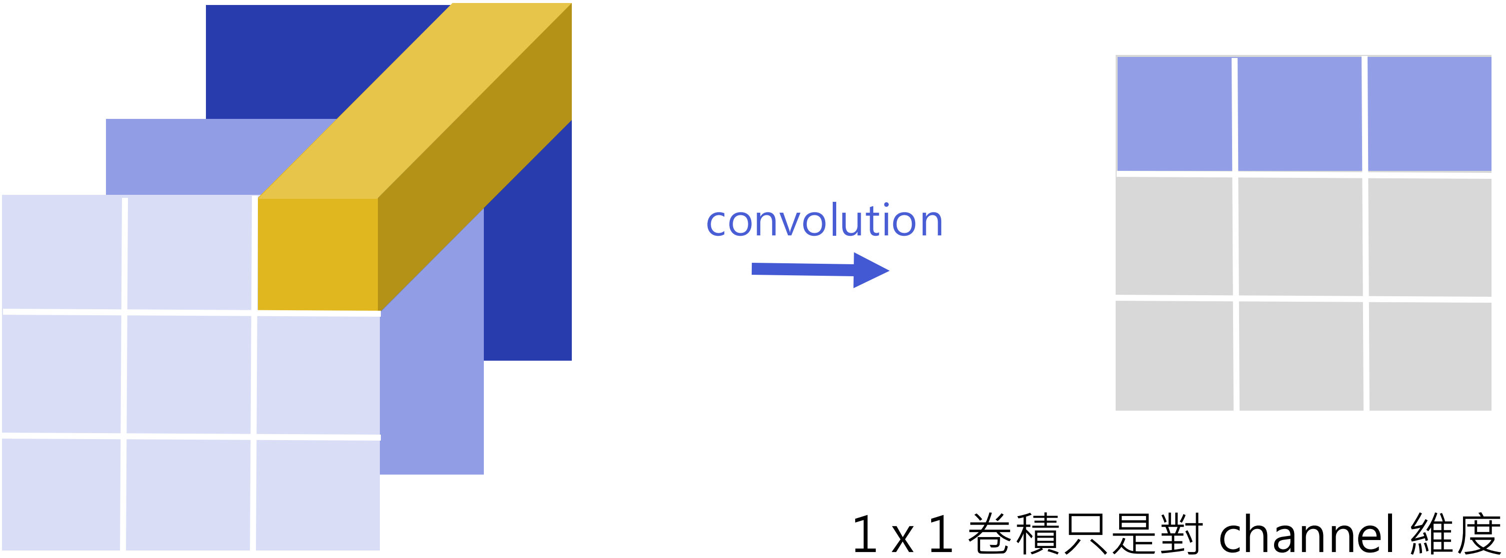
每次計算都是三個特徵圖的同個 pixel，各自乘上一個權重並加總。

1 x 1 Convolution



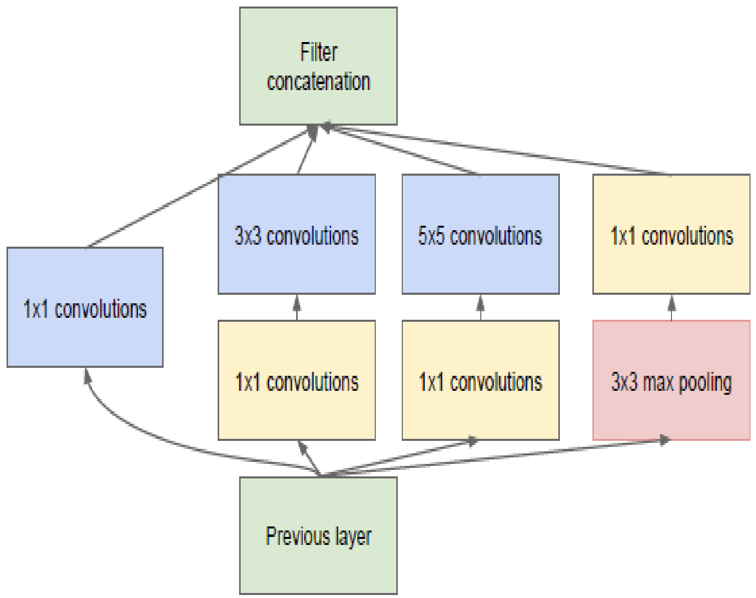
因為 filter 大小只有一個 pixel，因此同一張特徵圖的每個 pixel 都會乘上同一個數值。

1 x 1 Convolution



1 x 1 卷積只是對 channel 維度做加權總合的計算，並不會改變特徵圖中圖像的分布，因此可以當作是純粹升降維的工具。

Inception module



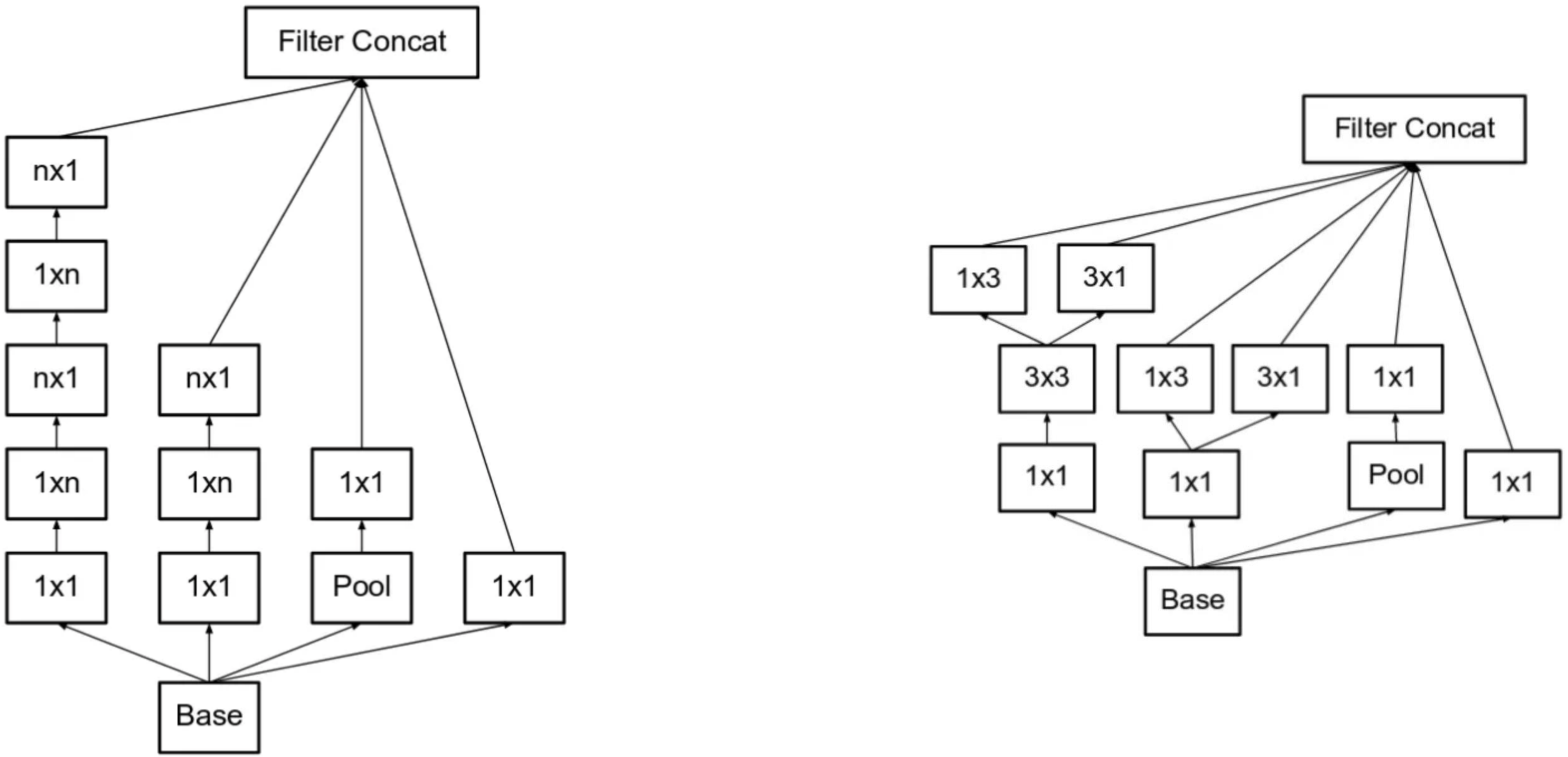

```python
class InceptionBlock(nn.Module):
    def __init__(self, in_channels, Filter_List):
        super().__init__()
        self.ConvA = nn.Sequential(nn.Conv2d(in_channels=in_channels, out_channels=Filter_List[0], kernel_size=1),
                                   nn.ReLU(inplace=True))

        self.ConvB = nn.Sequential(nn.Conv2d(in_channels=in_channels, out_channels=Filter_List[1], kernel_size=1),
                                   nn.ReLU(inplace=True),
                                   nn.Conv2d(in_channels=Filter_List[1], out_channels=Filter_List[2], kernel_size=3, padding=1),
                                   nn.ReLU(inplace=True))

        self.ConvC = nn.Sequential(nn.Conv2d(in_channels=in_channels, out_channels=Filter_List[3], kernel_size=1),
                                   nn.ReLU(inplace=True),
                                   nn.Conv2d(in_channels=Filter_List[3], out_channels=Filter_List[4], kernel_size=5, padding=2),
                                   nn.ReLU(inplace=True))

        self.ConvD = nn.Sequential(nn.MaxPool2d(kernel_size=3, stride=1, padding=1),
                                   nn.Conv2d(in_channels=in_channels, out_channels=Filter_List[5], kernel_size=1),
                                   nn.ReLU(inplace=True))
    def forward(self, x):
        out1 = self.ConvA(x)
        out2 = self.ConvB(x)
        out3 = self.ConvC(x)
        out4 = self.ConvD(x)
        out = torch.cat([out1, out2, out3, out4], dim=1)
        return out
```

Inception other module



ResNet

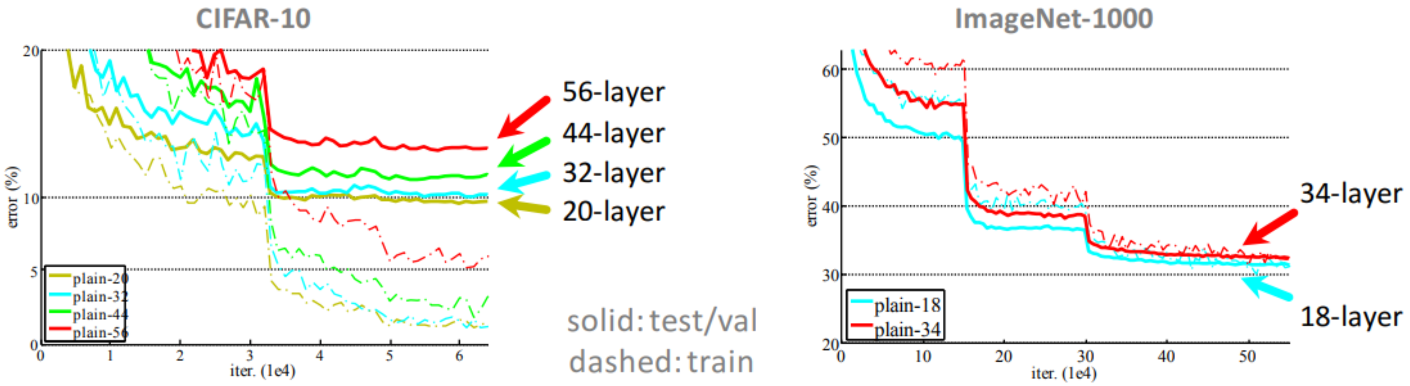
ResNet

[He et al., CVPR' 16]

Best Paper Award!

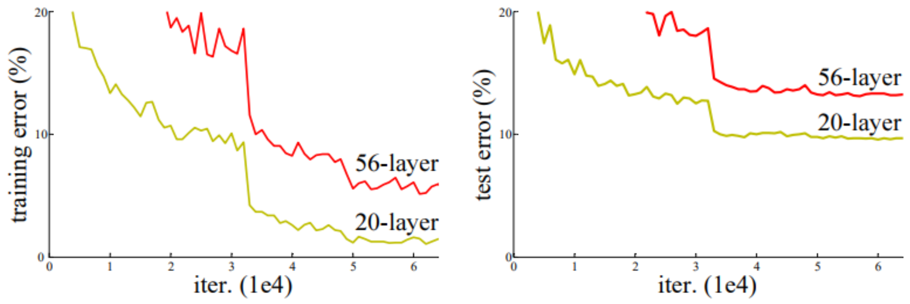
- 網路越深越好
  - 更大的視野域 (receptive field)
  - 更多的非線性
  - 更佳的擬合能力

真的嗎?
是不是CNN網路越深越好?
會不會遇到甚麼問題?



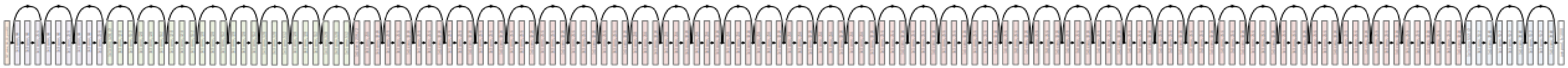
ResNet

- 過擬合 (Overfitting)？
  - No， train 與 test 的結果都與圖中狀況一致 。
  - 已經在多種資料集上驗證，都有圖中的問題。
- 梯度消失/爆炸?
  - 在每一層的網路都有使用 normalize 來緩解。
  - 激活函數使用 ReLU
- 此狀況稱為 ” 神經網路退化 ”



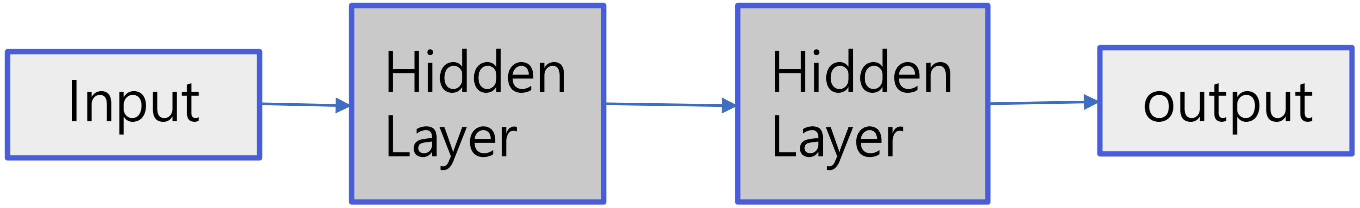
ResNet

- 主要架構
  - 常見層數為 50, 101, 及 152 層
  - 網路架構非常深, 甚至超過 1,000 層
  - 由堆疊多層 residual modules 所組成

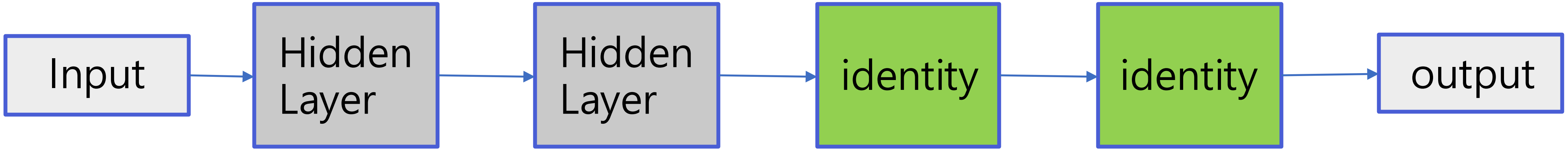


ResNet: Degradation Problem

Network #1



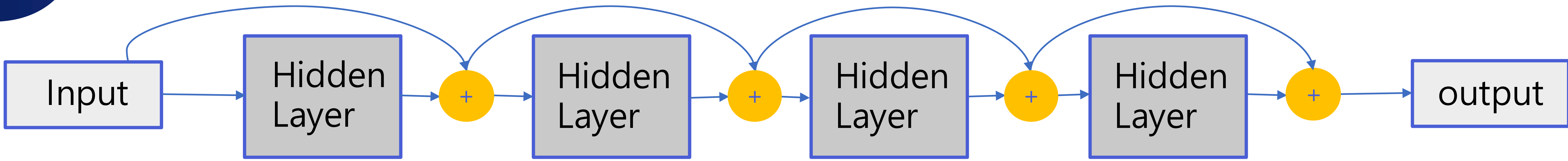
Network #2

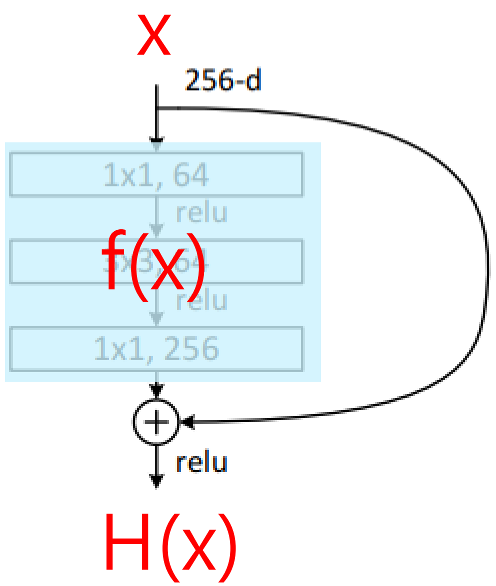


Identity : f(x) = x

https://medium.com/@realmichaelye/intuition-for-resnet-deep-residual-learning-for-image-recognition-39d24d173e78

ResNet: Residual module





$$H(x) = \boxed{f(x)} + x$$

$$\boxed{f(x)} \rightarrow 0$$

$$f(x) = H(x) - x = 0$$

Residual

ResNeXT

ResNeXT [Xie et al., CVPR' 17]

- 提升CNN模型的表現主要研究的議題為：深度、寬度
  - 如：ResNet（深度）、InceptionNet（寬度）
- 但其實還有一個可以探討的議題，那就是cardinality
  - 深度跟寬度我都要

ResNeXT主要貢獻

- 提出了Cardinality的分組概念。
- 模型結構更加簡單和模組化。
- 大幅降低超參數調整。
  - InceptionNet的可怕超參數調整
- 參數量及層數的減少。
  - 101層的ResNeXT網路與200層的ResNeT有一樣的準確率

Cardinality

- 將高維度的卷積層分組為多個相同的卷積層 (Inception 是各個不同的卷積層)，然後進行卷積運算，最後再將這些卷積層融合。

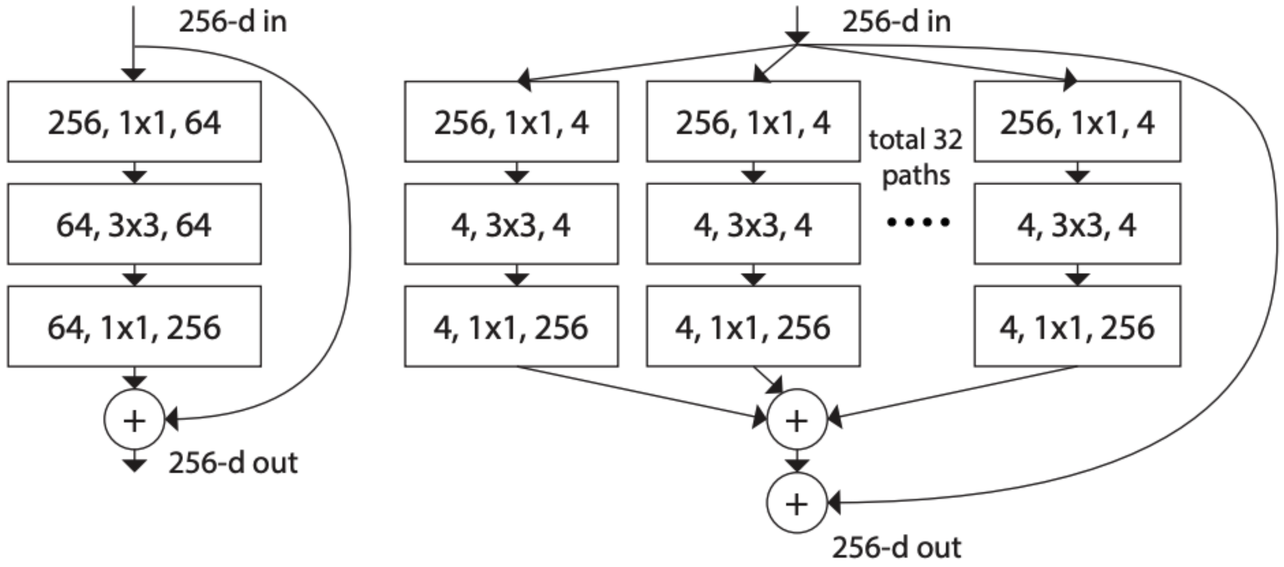
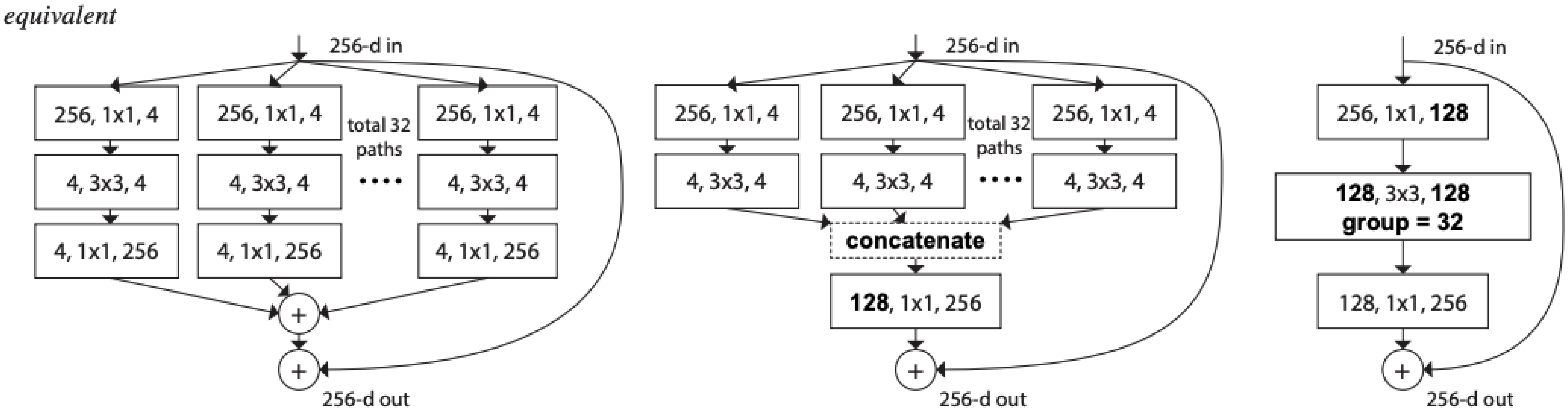


Figure 1. **Left**: A block of ResNet [14]. **Right**: A block of ResNeXt with cardinality = 32, with roughly the same complexity. A layer is shown as (# in channels, filter size, # out channels).

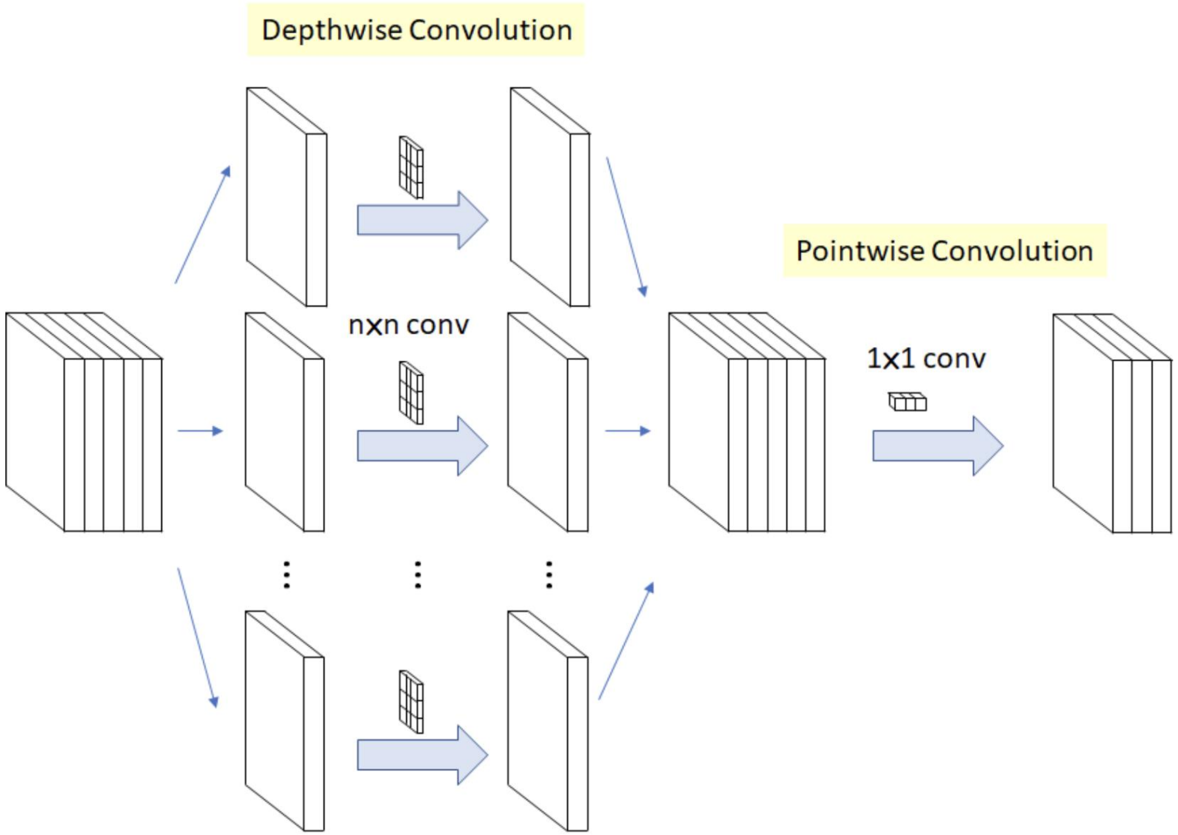
ResNeXT Cardinality



(a) (b) (c)

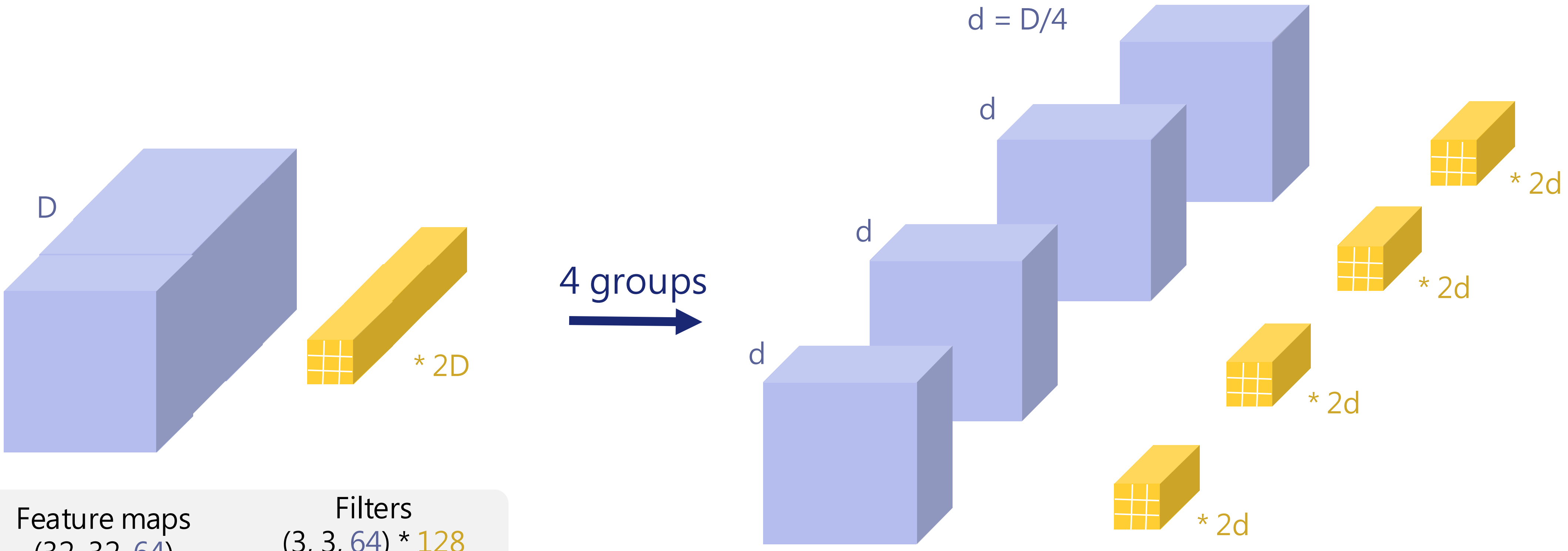
Group Convolution

- 假設原本圖大小為(32,32,64)
- 然後目標經過卷積後會將feature map上升到128張。
- Group convolution則會將64張特徵圖分成n組，假設n=4
- (32,32,64) -> (4,32,32,16)
- 然後每一組的(32,32,16)都會卷積出32張的特徵圖，所以卷積過後的每一組都是(32,32,32)
- 接下來concate這4組的結果，變成(32,32,128)即完成。



Group Convolution

nn.Conv2d(in_channels=64, out_channels=128, kernel_size=3, groups=4)



Feature maps
(32, 32, 64)

Filters
(3, 3, 64) * 128

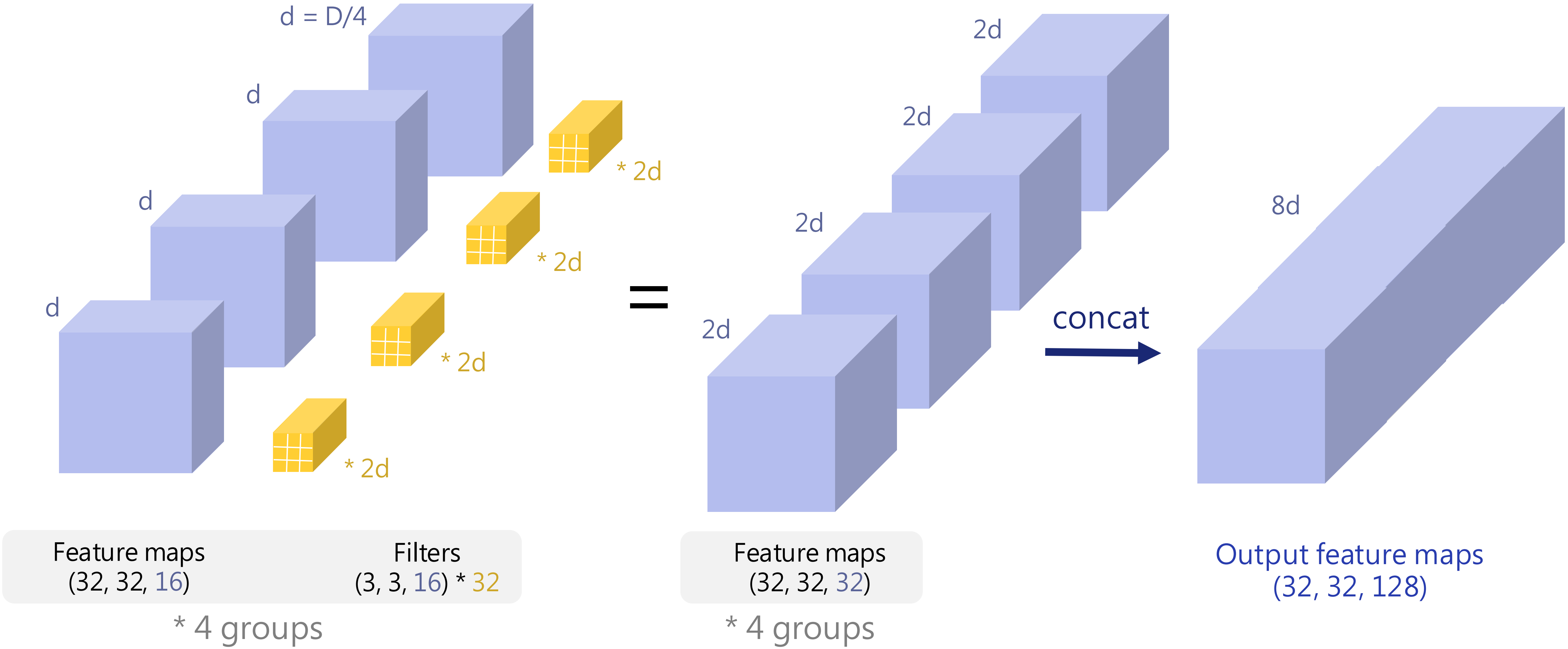
Output feature maps
(32, 32, 128)

Feature maps
(32, 32, 16)

Filters
(3, 3, 16) * 32

* 4 groups

Group Convolution



實作練習