Lab 2: Deep Residual Learning

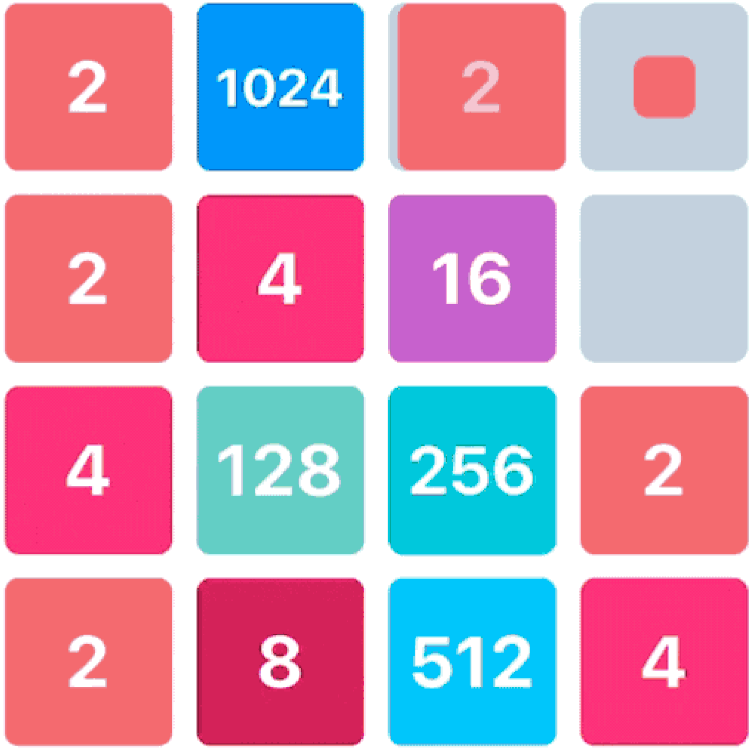
Department of Computer Science, NCTU

TA Yu-Chuan Chuang (莊祐銓)

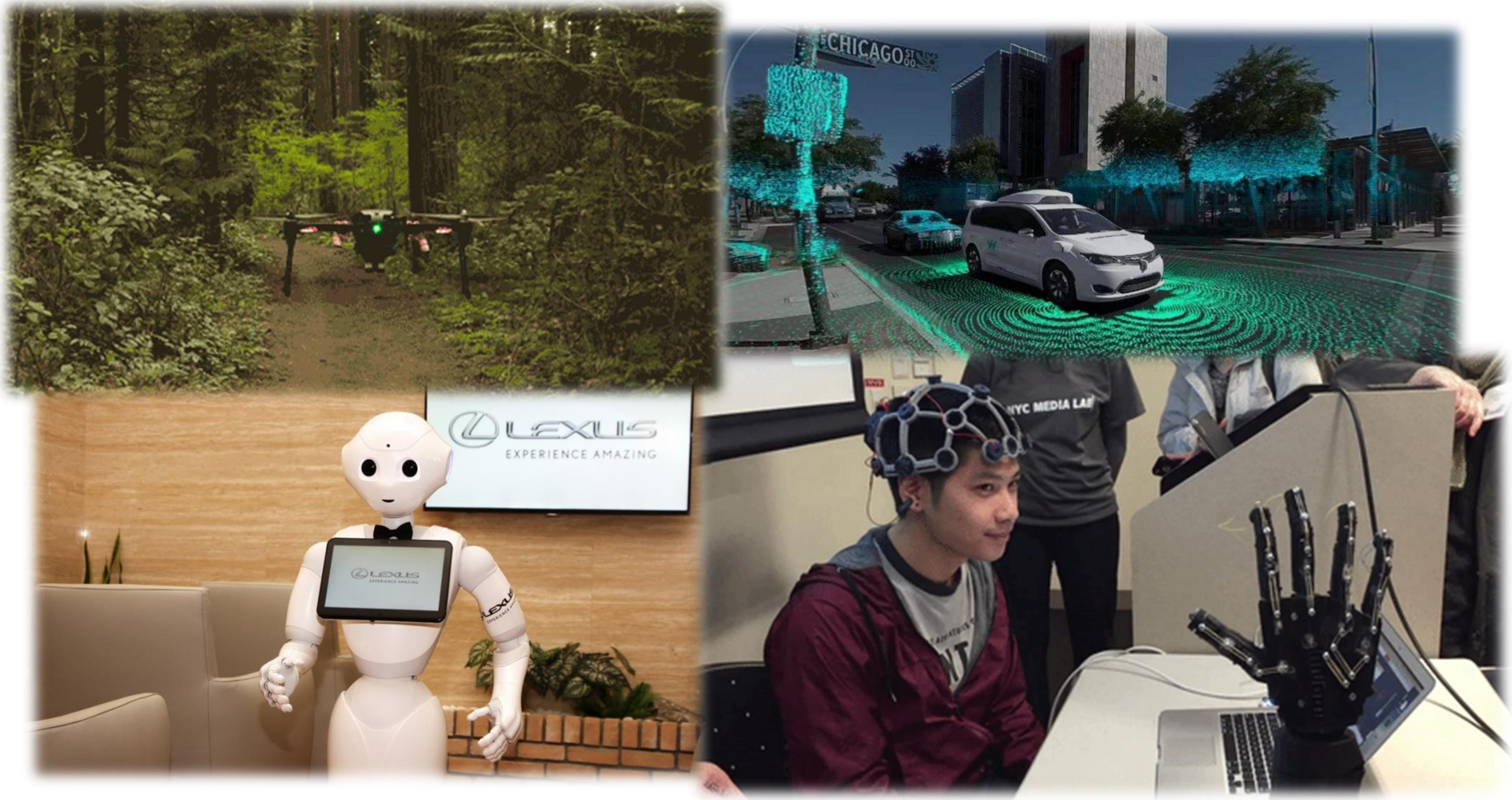
Source: Hung-yi Lee, “Machine Learning course”

Paper: He, Kaiming, et al. "Deep residual learning for image recognition." CVPR, 2016.

Applications - Playing Go & Games



Applications - Autonomous Driving & Robotics

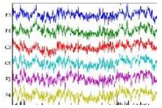


How it(AI) works?

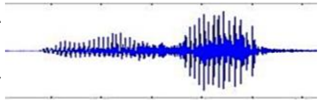
深度學習≈找尋一個函數(根據資料)

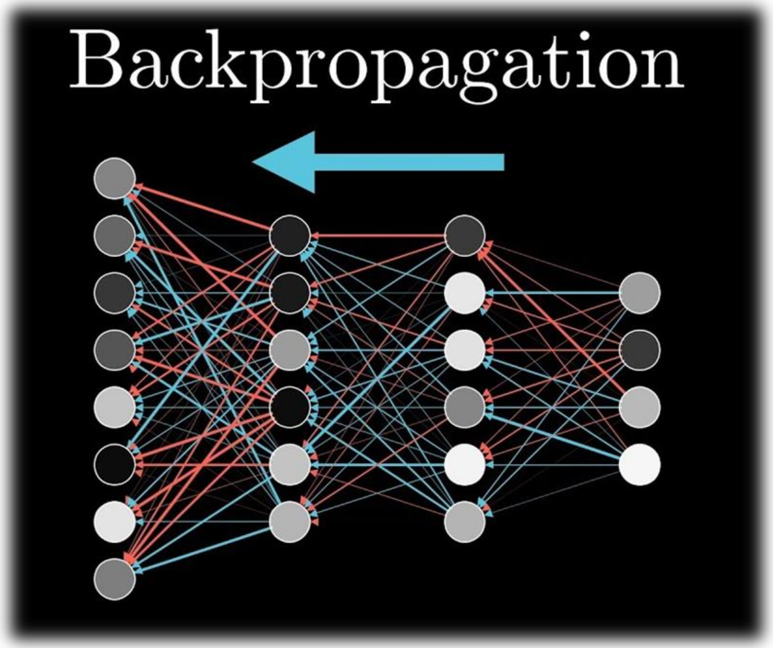
Mail filtering: ***f***() = 垃圾郵件?

Image classification: ***f***() = 貓or狗?

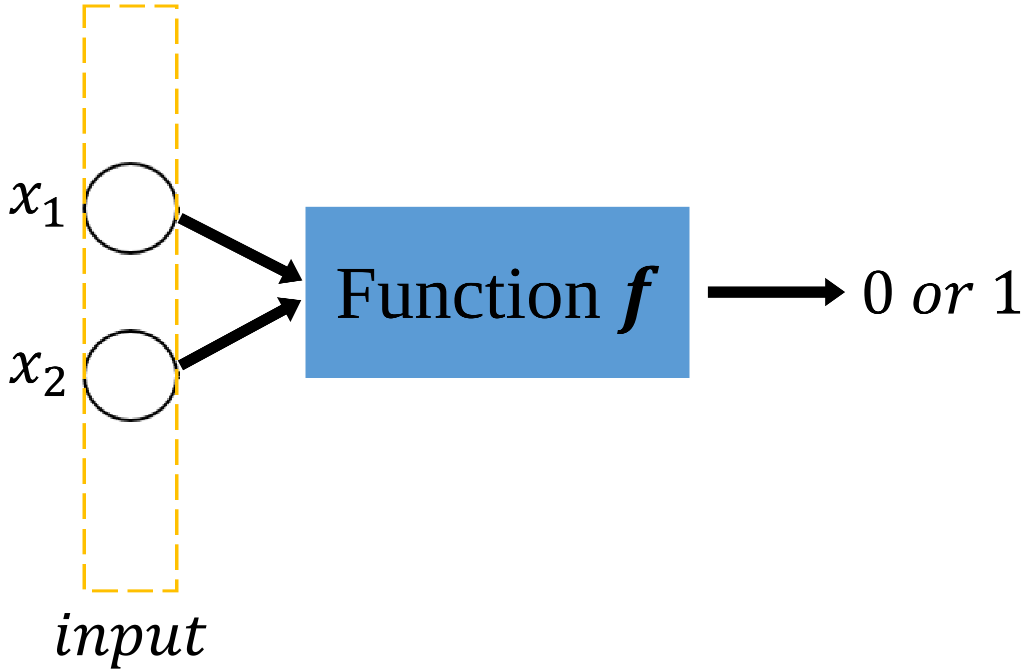
EEG classification : ***f***() = 動左手?動右手?

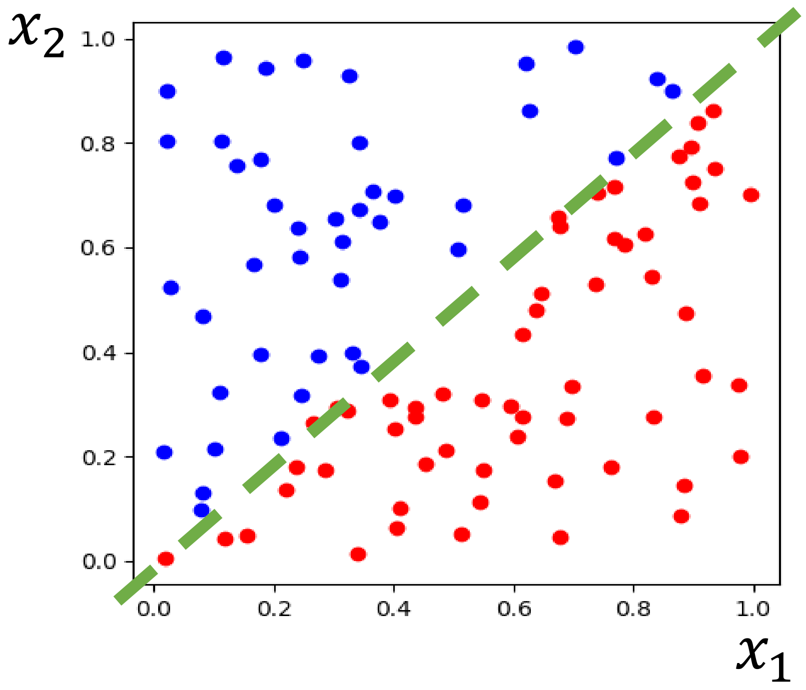
Playing go: ***f***() = 下在天元?

Speech recognition: ***f***() = "Wow"



Binary Classification(二元分類)



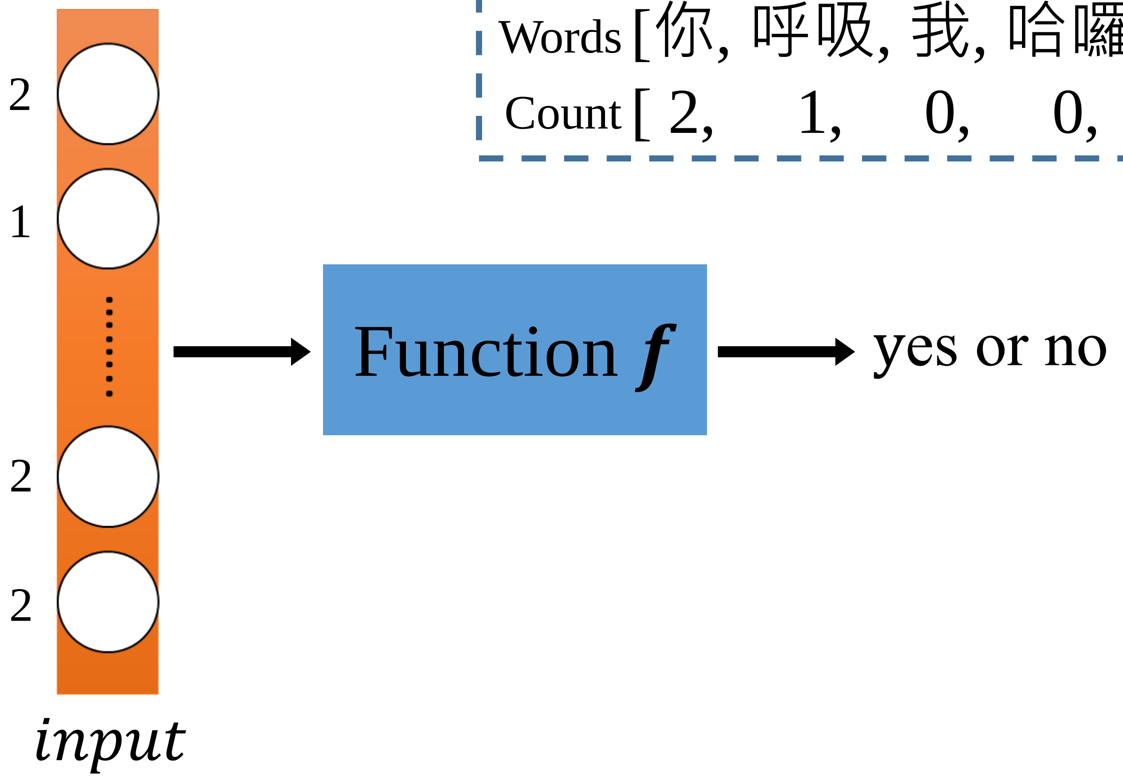


Mail Filtering(郵件過濾)

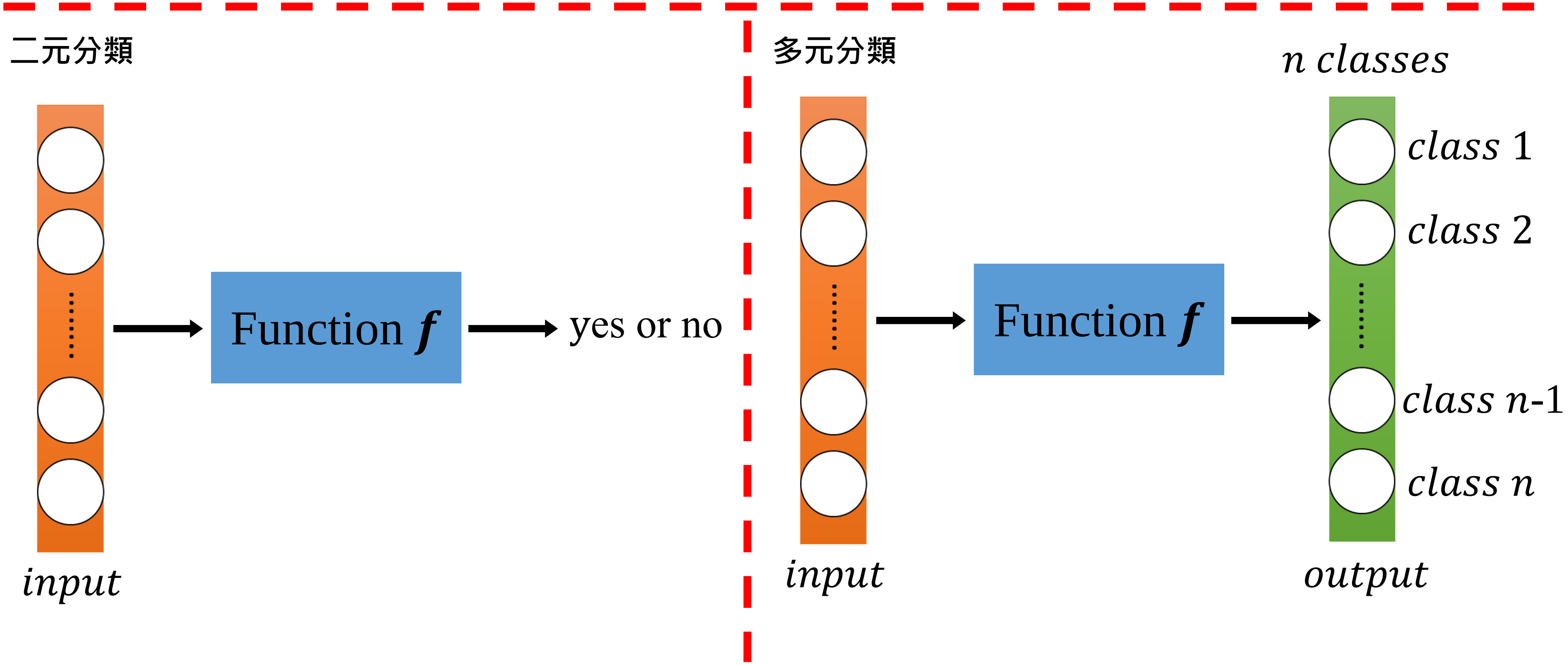
Mail filtering: $\boldsymbol{f}$( ) = yes or no?

你知道在非洲過了60秒，就等於過了一分鐘
你知道呼吸很重要嗎? 吸了第一口，就不能沒有第二口

Words	[你,	呼吸,	我,	哈囉,	沒有,	不能,	.... ,	知道,	口]
Count	[2,	1,	0,	0,	1,	1,	.... ,	2,	2]



Multi-class Classification(多元分類)



MNIST Classification(手寫數字辨識)

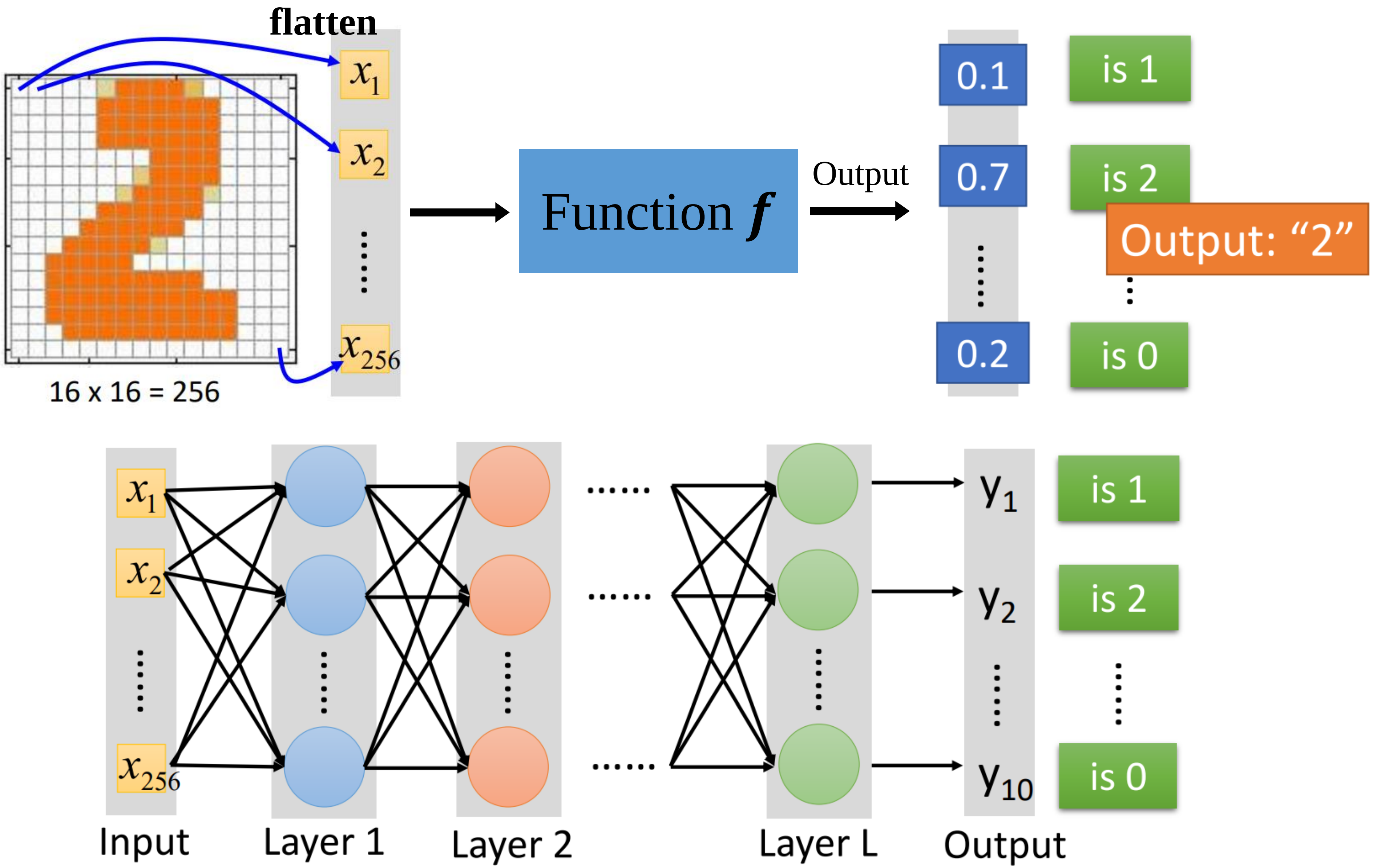
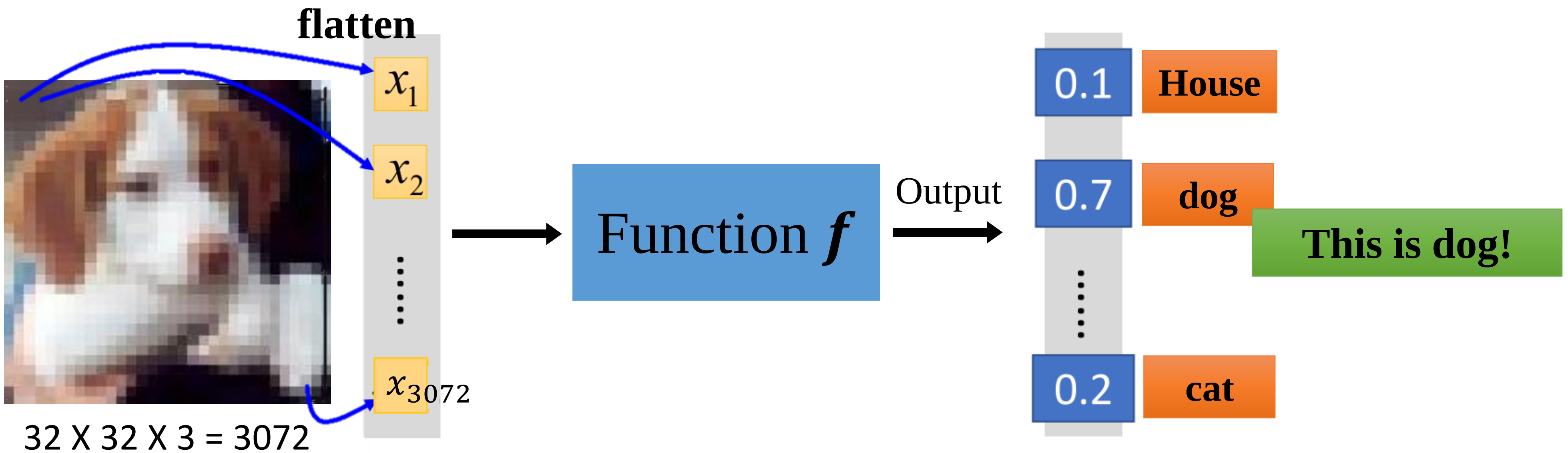
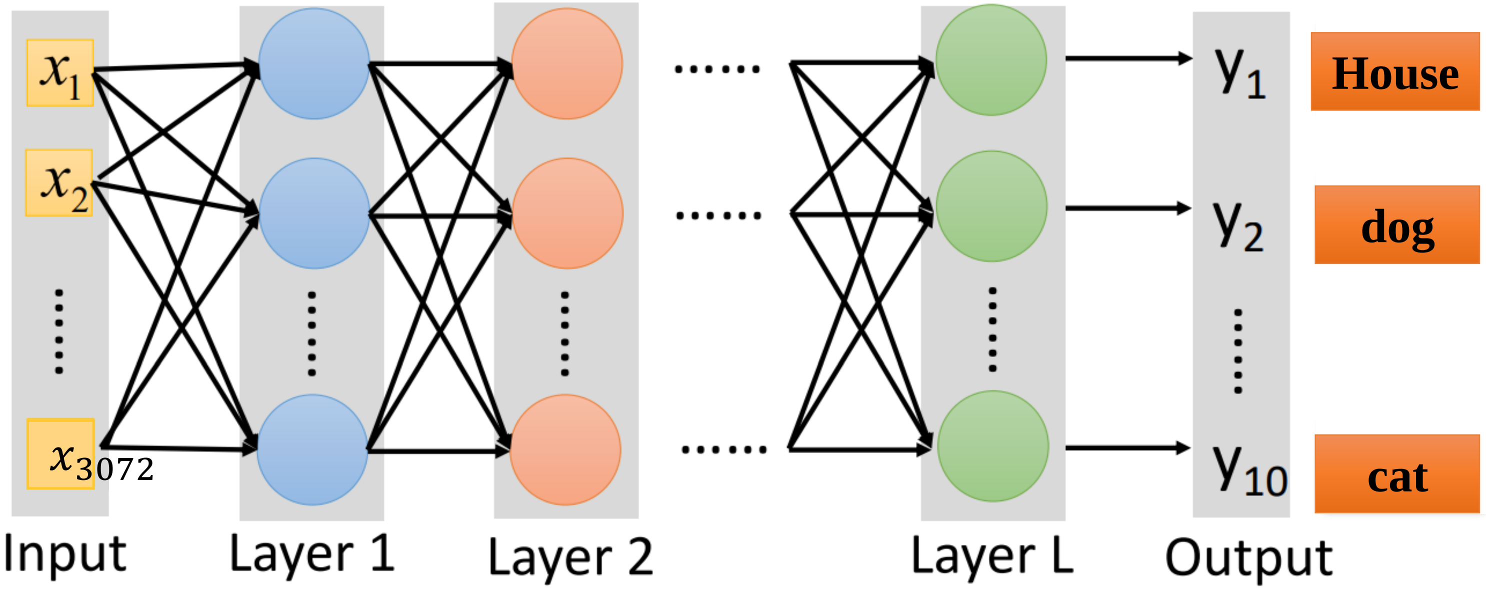


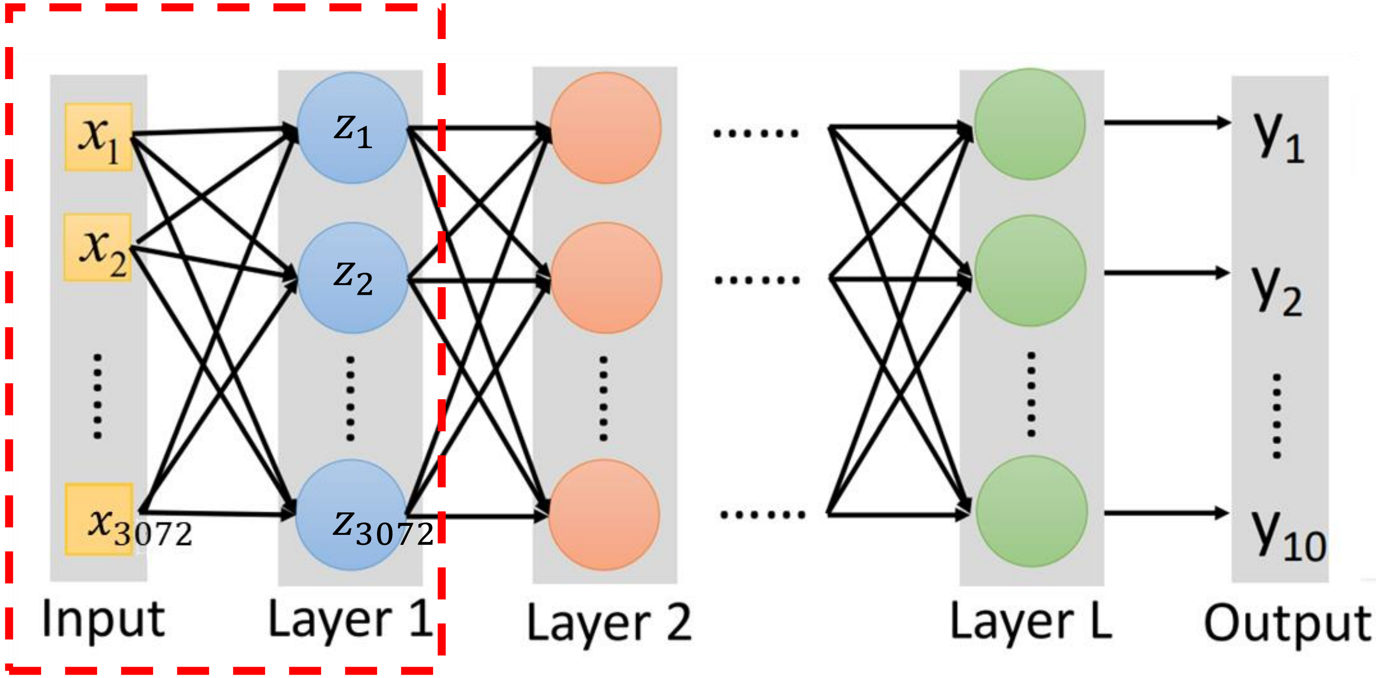
Image Classification(影像辨識)





Fully Connected Layer

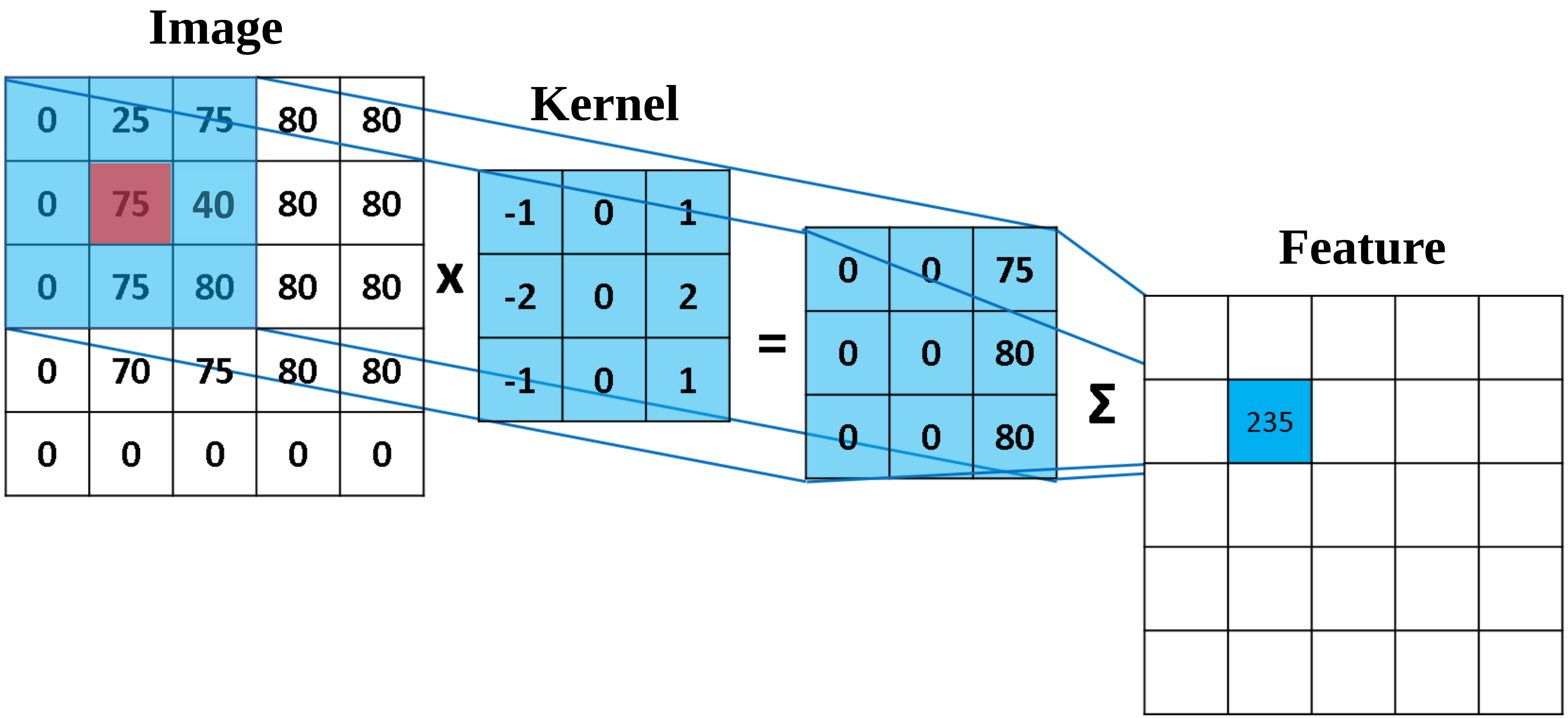
32 X 32 X 3 = 3072



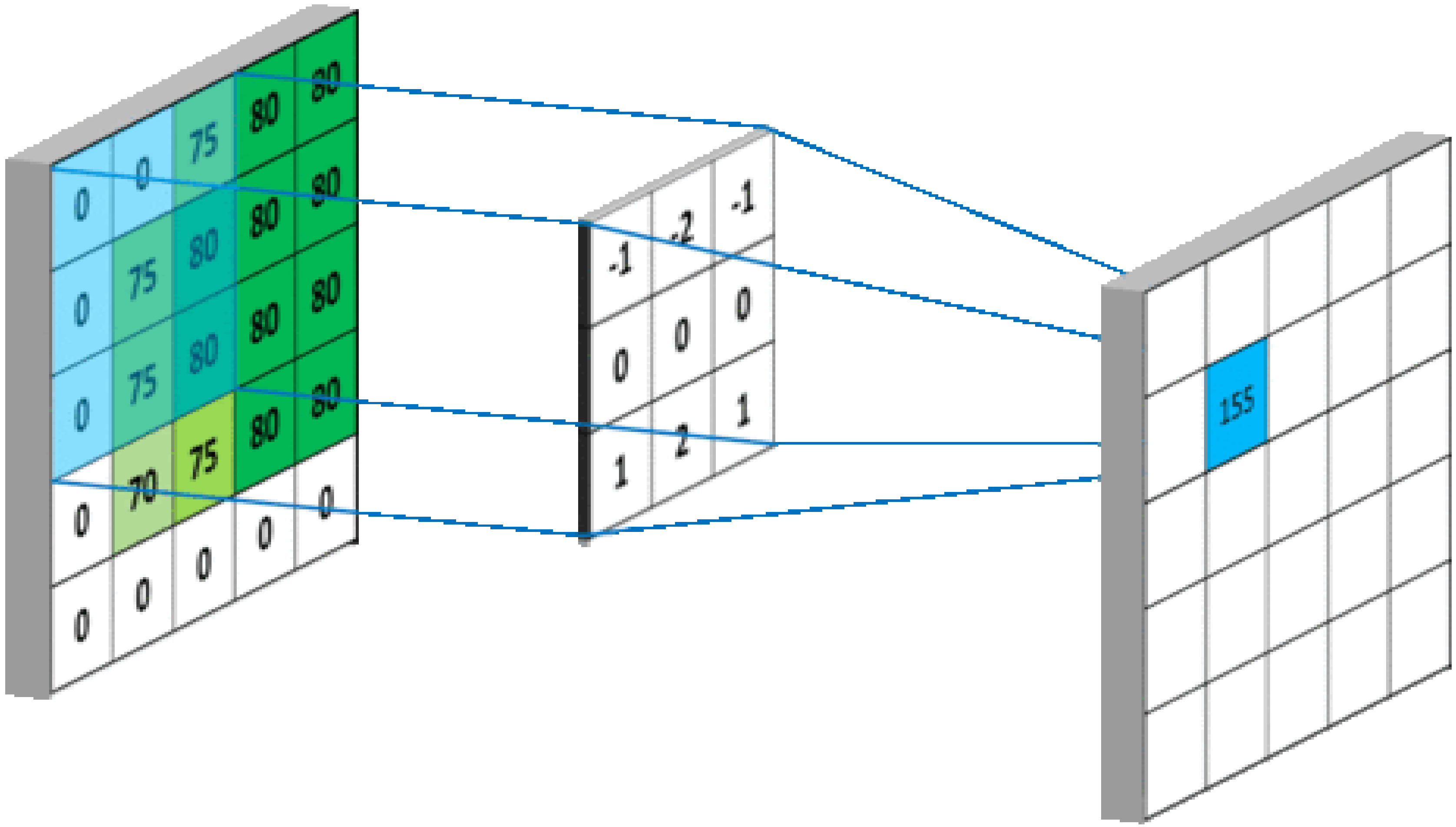
參數量: 3072 X 3072 = 9437184

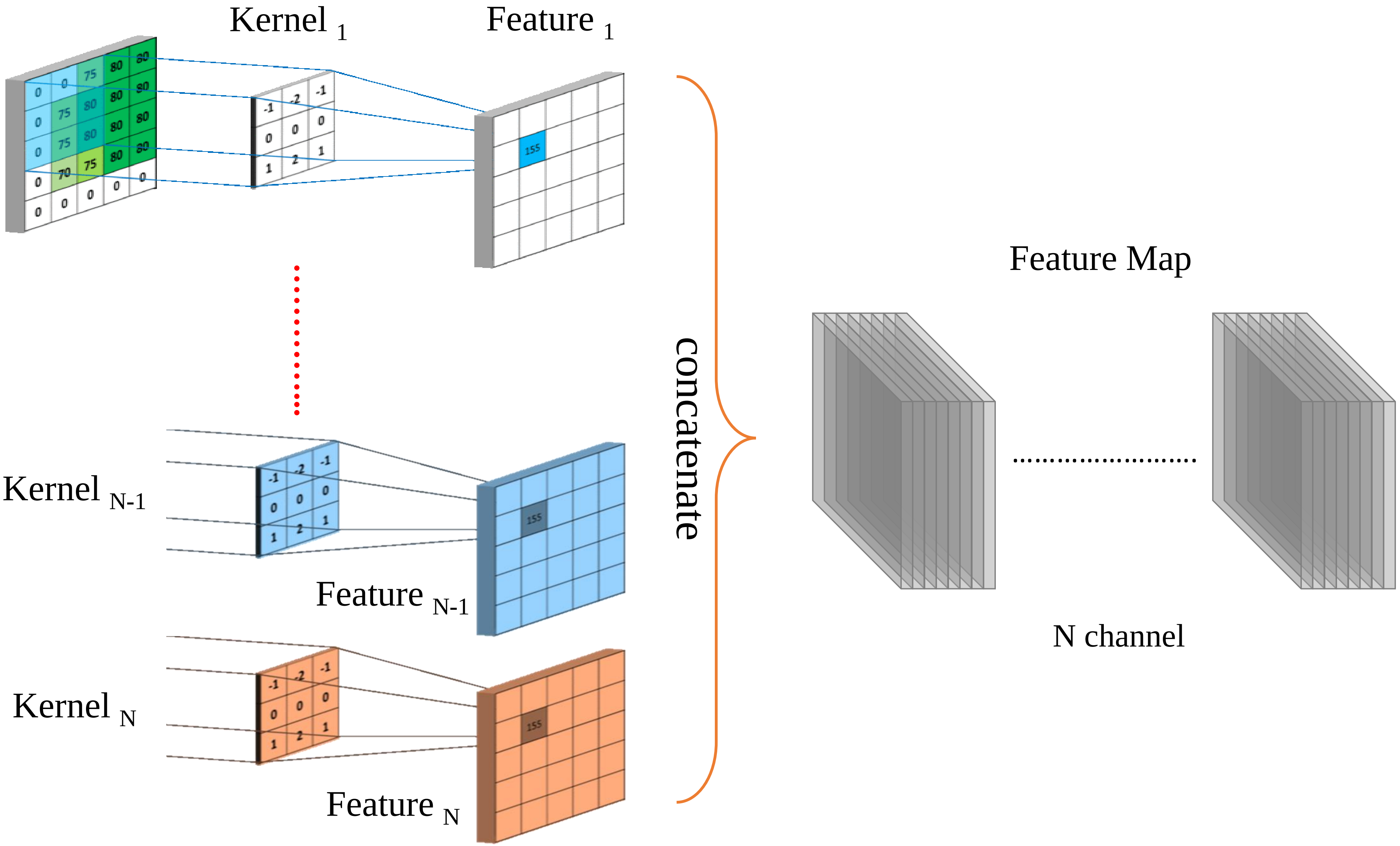
torch.nn.linear(in_features=3072, out_features=3072)

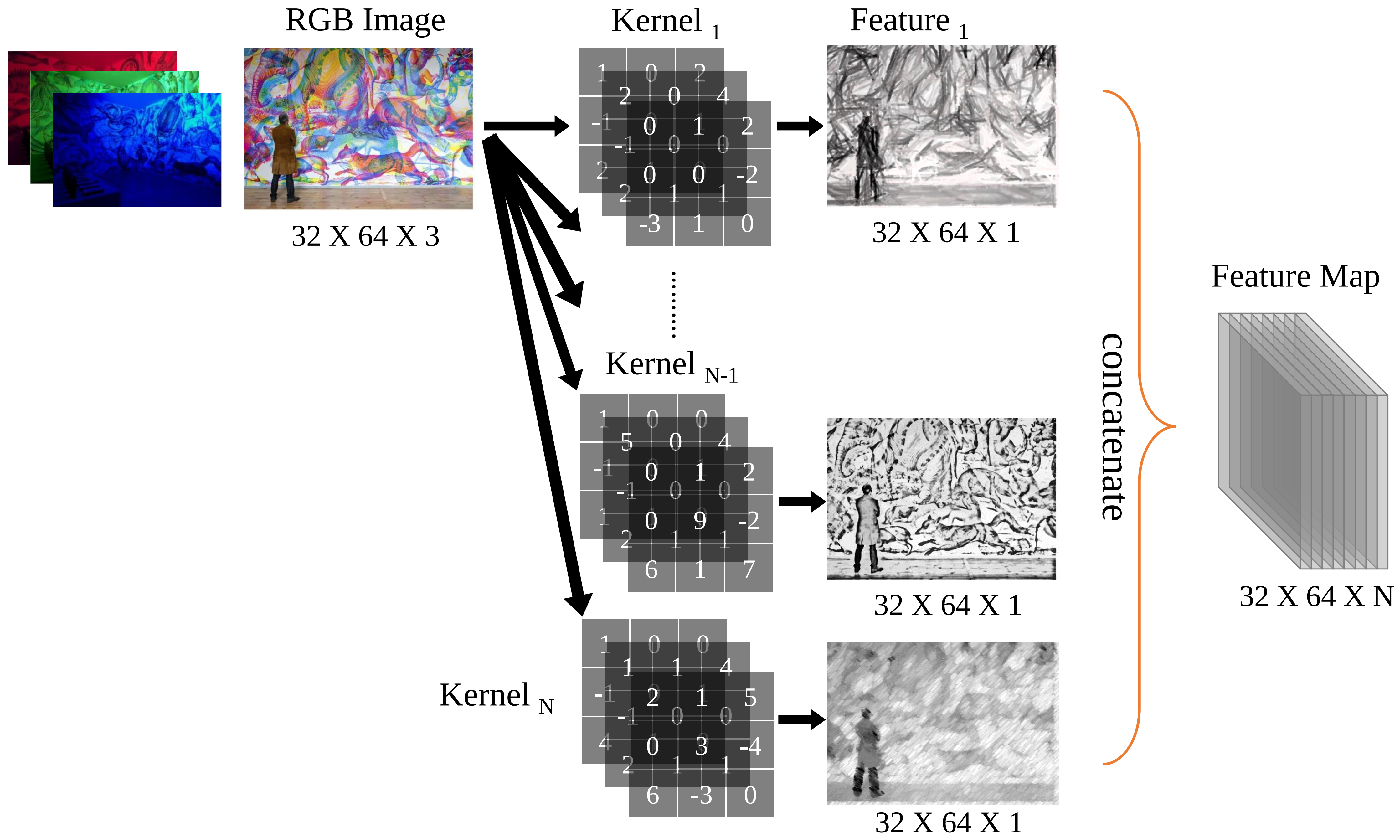
Convolution Layer



Convolution Layer



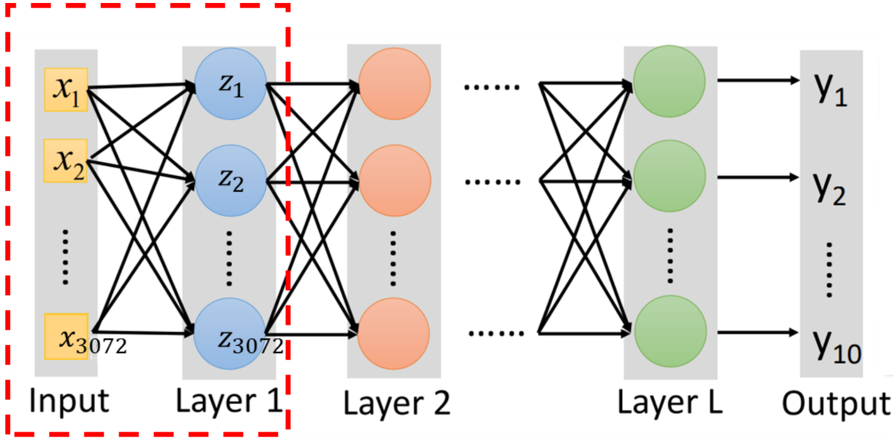

Kernel 1
Feature 1
Kernel N-1
Feature N-1
Kernel N
Feature N
concatenate
Feature Map
N channel



RGB Image
32 X 64 X 3
Kernel 1
Feature 1
32 X 64 X 1
Kernel N-1
32 X 64 X 1
Kernel N
32 X 64 X 1
concatenate
Feature Map
32 X 64 X N

Fully Connected Layer **vs** Convolution Layer

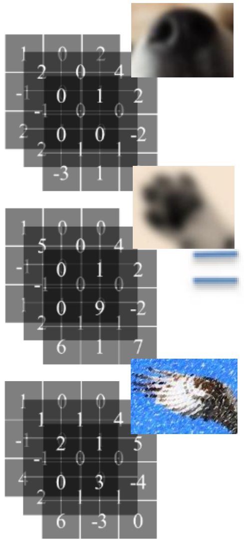
32 X 32 X 3 = 3072



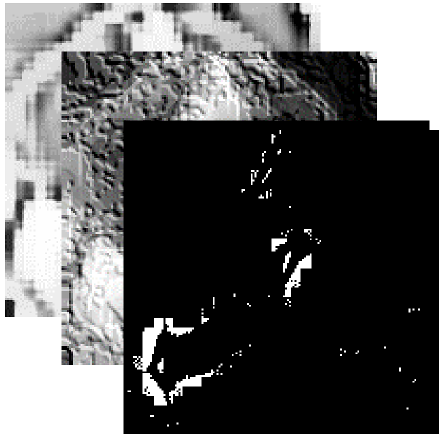
參數量: 3072 X 3072 = 9437184

32 X 32 X 3 = 3072

$\otimes$



=



32 X 32 X 3 = 3072

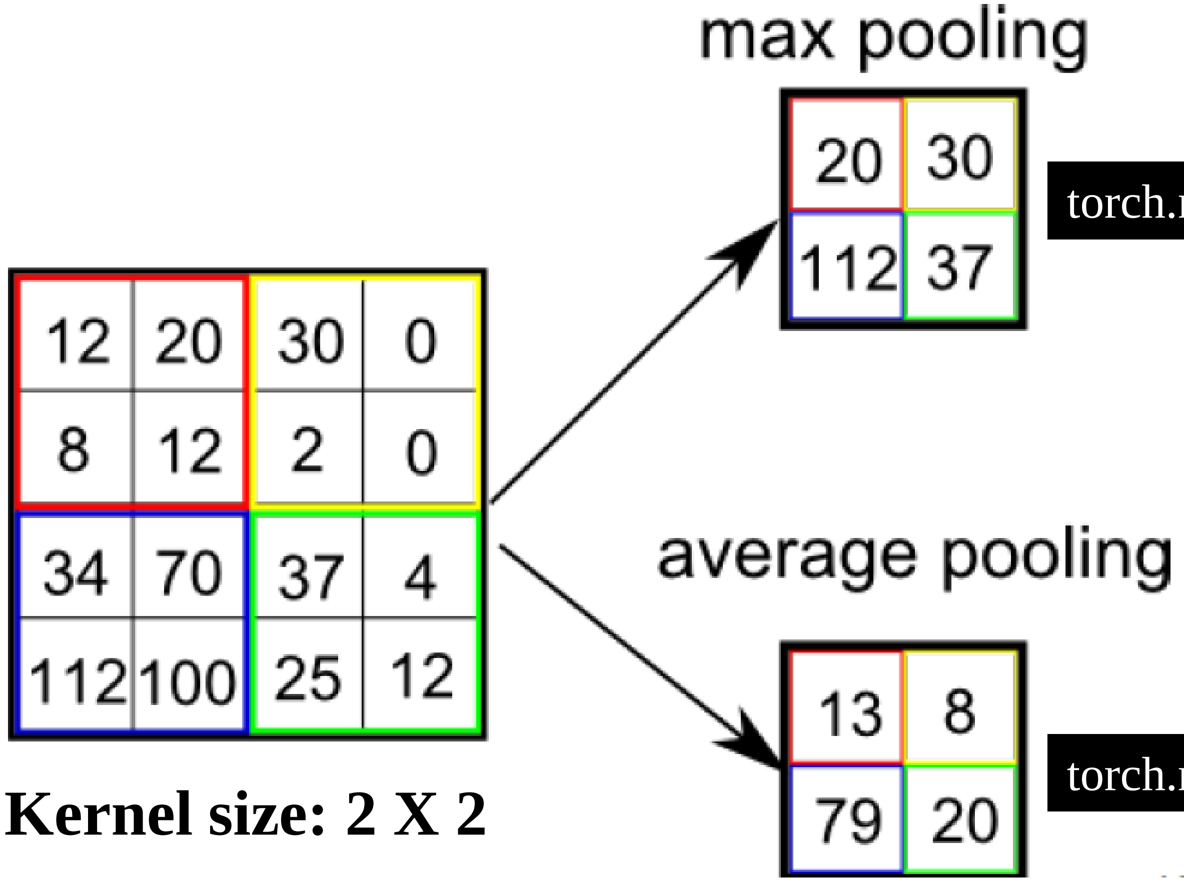
```
torch.nn.Conv2d(in_channels=3, out_channels=3,
                kernel_size=(3,3),
                stride=1,
                padding=1)
```

參數量: 3 X 3 X 3 X 3 = 81

Kernel size X input channel X output channel

Pooling

- A pooling function replaces the output of the net with a summary statistic of the nearby outputs.
- Pooling helps to make the representation become approximately invariant to small translation and rotation.



`torch.nn.MaxPool2d(kernel_size=(2,2))`

`torch.nn.AvgPool2d(kernel_size=(2,2))`

Kernel size: 2 X 2

Example: Feedforward Deep Networks

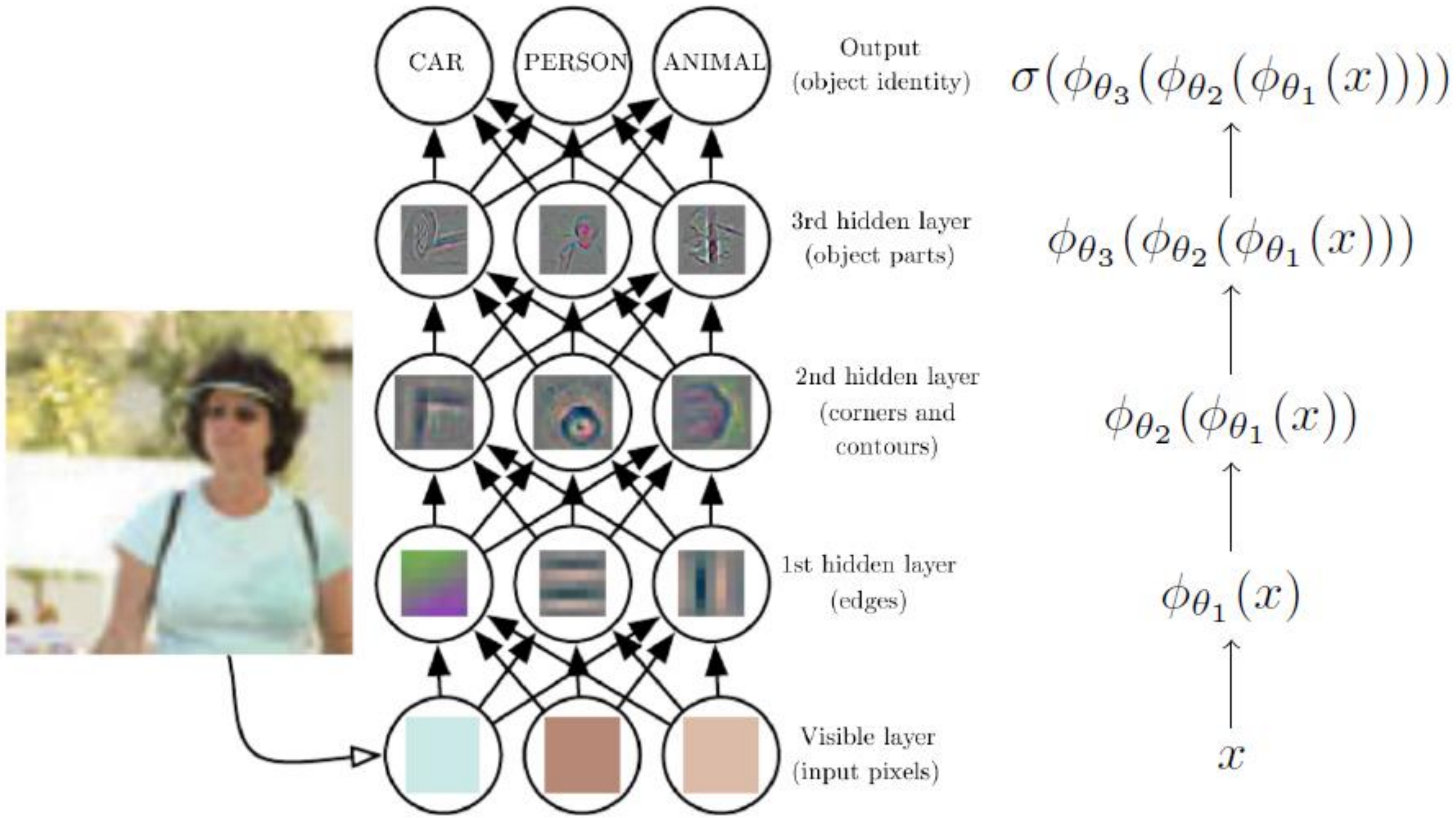
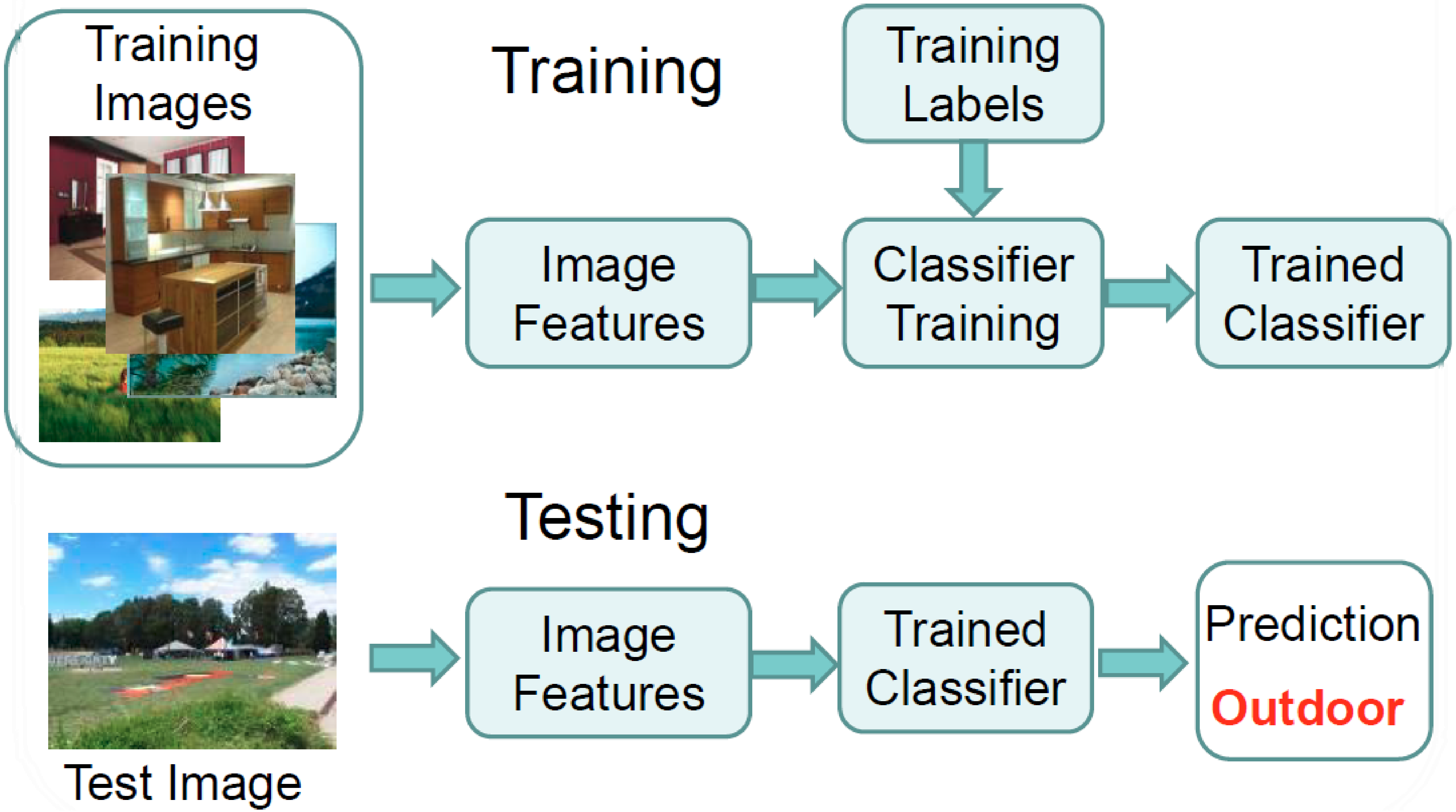
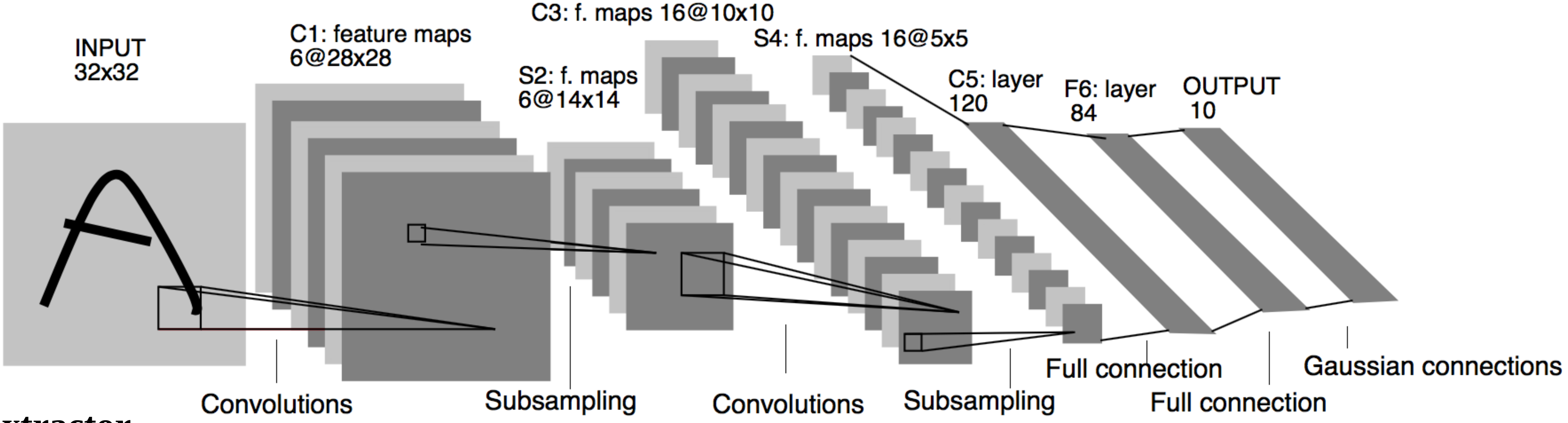


Image Categorization



LeNet(LeCun et al. 1998)



Feature Extractor

Conv2d(1, 6, kernel_size=5, stride=1, padding=0)

ReLU

MaxPool2d(kernel_size=2))

Conv2d(6, 16, kernel_size=5, stride=1, padding=0)

ReLU

MaxPool2d(kernel_size=2))

Flatten: 把所有Feature全部攤平，變成一個vector

Flatten

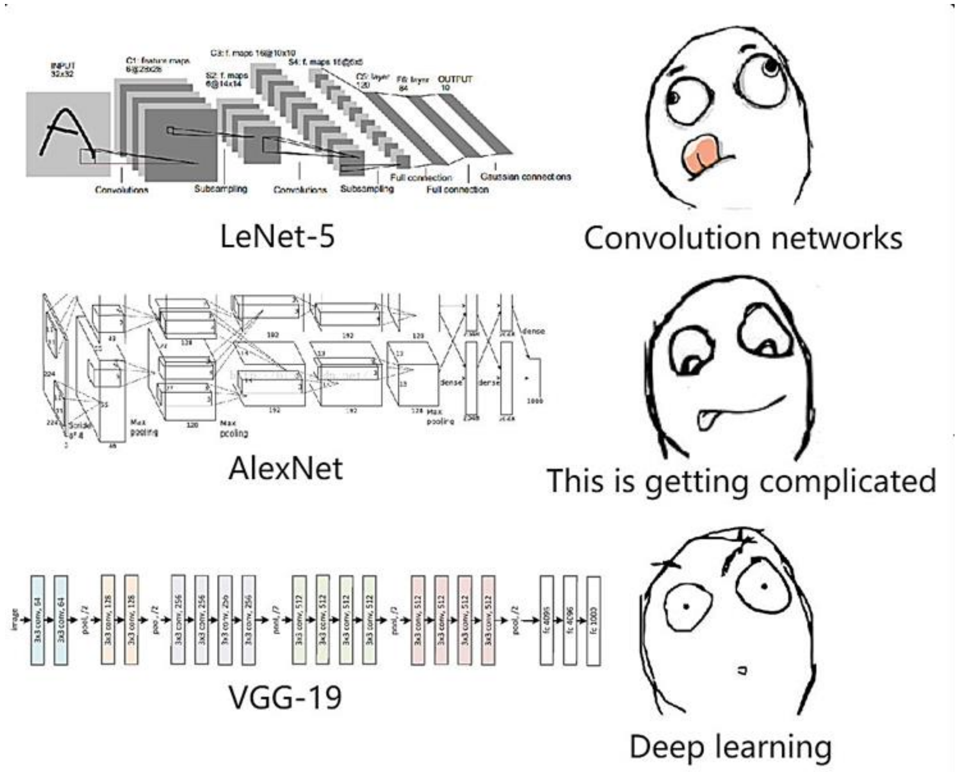
Classifier

linear(5*5*16, 120) → linear(120, 84) → linear(84, 10) → y_1, y_2, ⋮, y_{10}

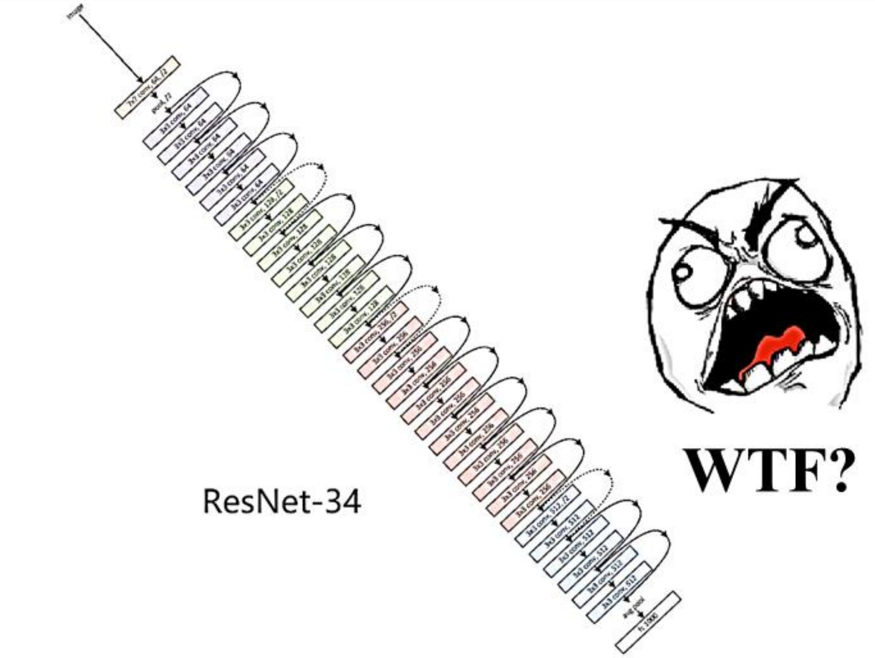
Output

ResNet

- Won 1st place in the ILSVRC 2015 classification competition
- ResNet makes it possible to train up to hundreds or even thousands of layers and still achieves compelling performance.

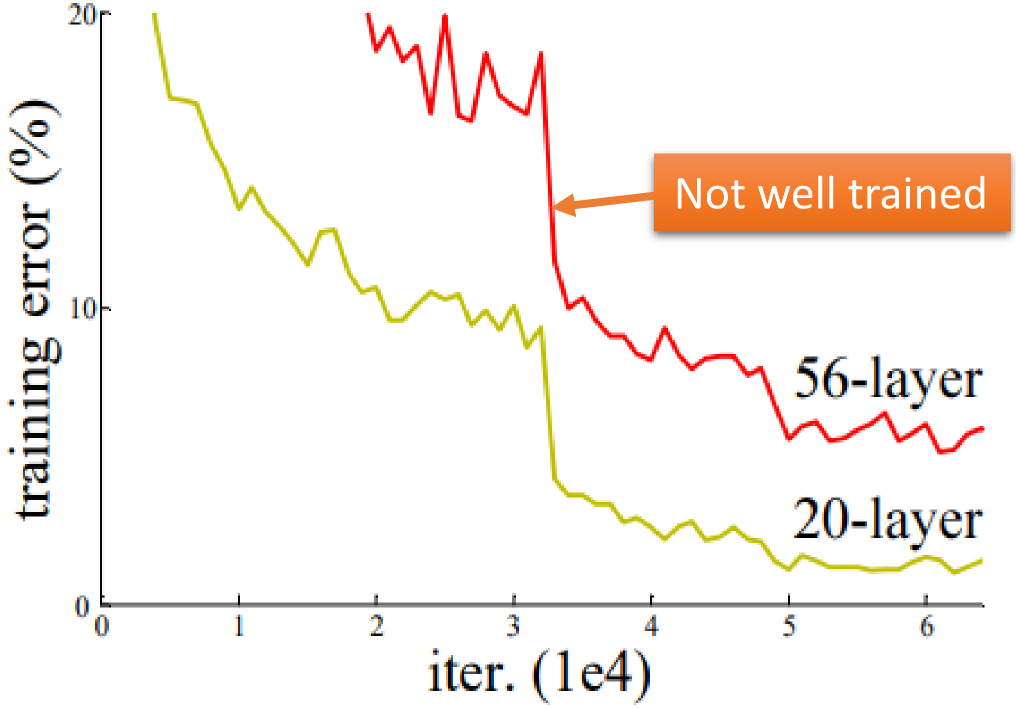


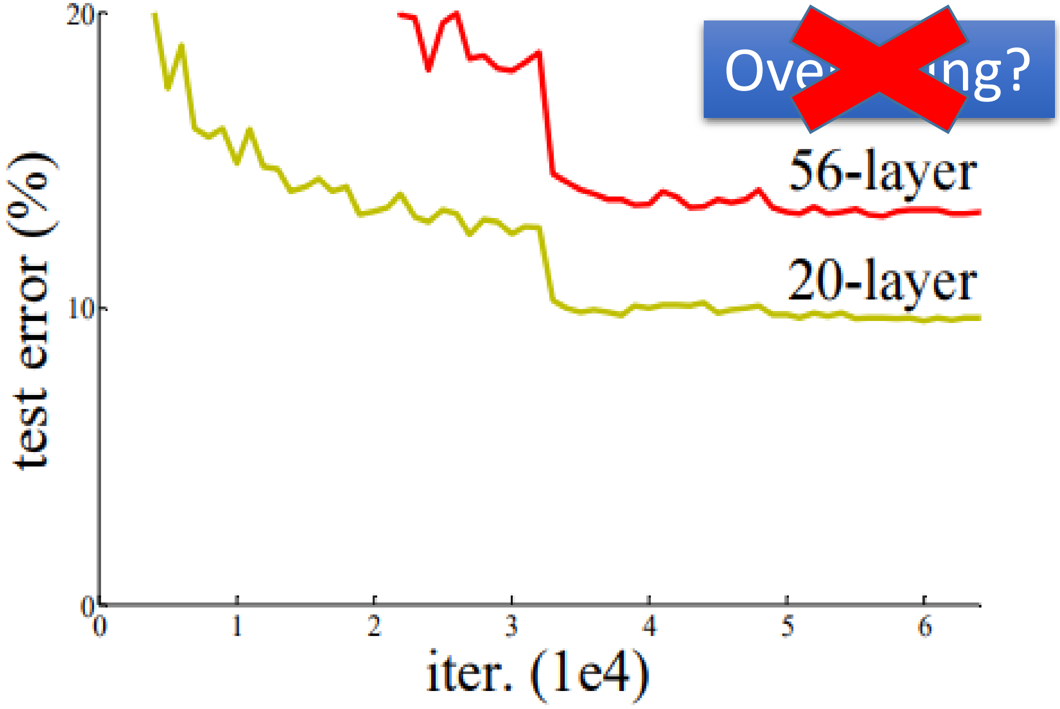
網路參數越多，網路能力就越強，效果就越好



Deeper is Better?

- Increasing network depth does not work by simply stacking layers together
- Deep networks are hard to train because of the **gradient vanishing problem**





Gradient Vanishing

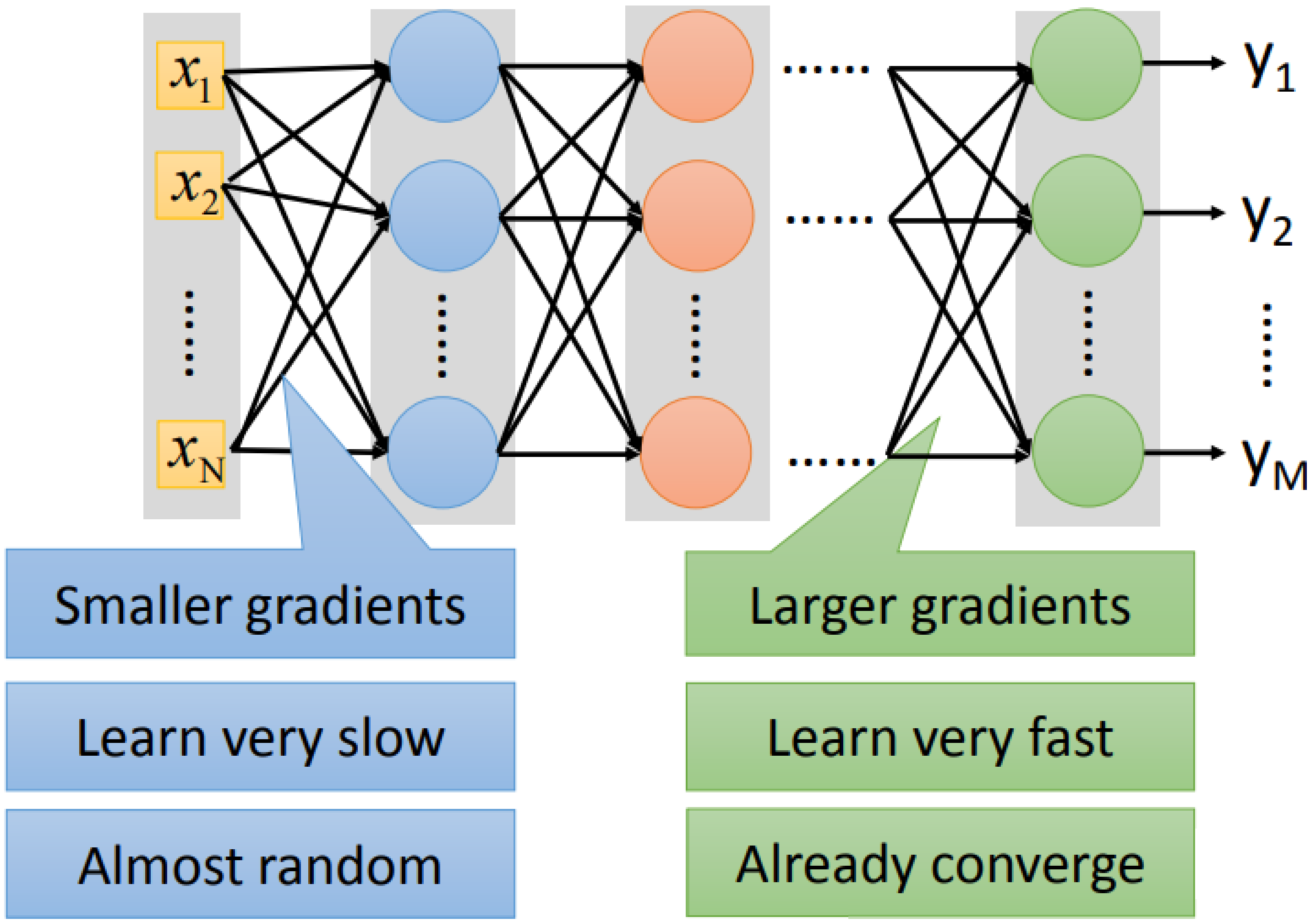
$$x \rightarrow w_1 \rightarrow w_2 \rightarrow w_3 \rightarrow w_4 \rightarrow Loss$$
$$y_1 \quad y_2 \quad y_3 \quad y_4$$

$$\frac{\partial Loss}{\partial w_1} = \frac{\partial Loss}{\partial y_4}\frac{\partial y_4}{\partial z_4}\frac{\partial z_4}{\partial y_3}\frac{\partial y_3}{\partial z_3}\frac{\partial z_3}{\partial y_2}\frac{\partial y_2}{\partial z_2}\frac{\partial z_2}{\partial y_1}\frac{\partial y_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$

$$= \frac{\partial Loss}{\partial y_4}\sigma'(z_4)w_4\sigma'(z_3)w_3\sigma'(z_2)w_2\sigma'(z_1)x_1$$

If $w_i < 1$, it will produce gradient vanishing.

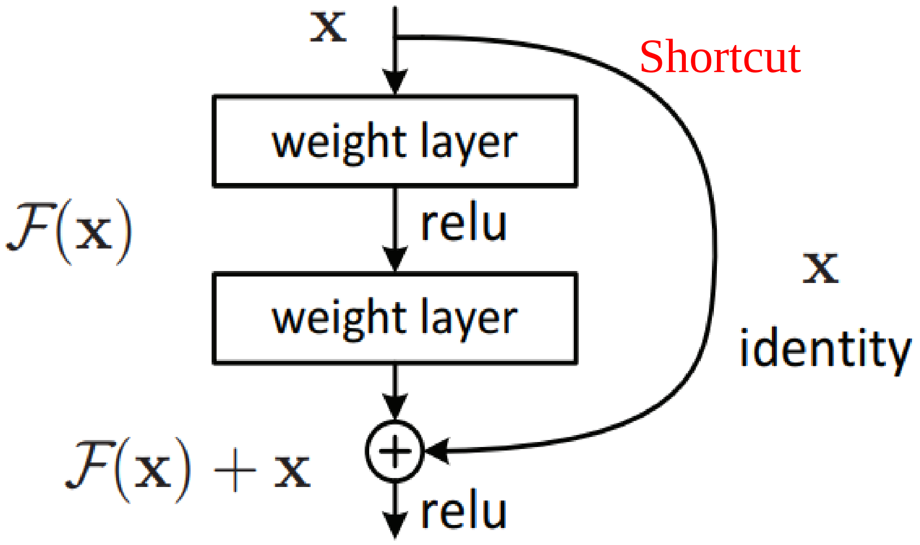
Gradient Vanishing



How to Avoid Gradient Vanishing?

- This identity mapping does not have any parameters and is just there to add the output from the previous layer to the layer ahead

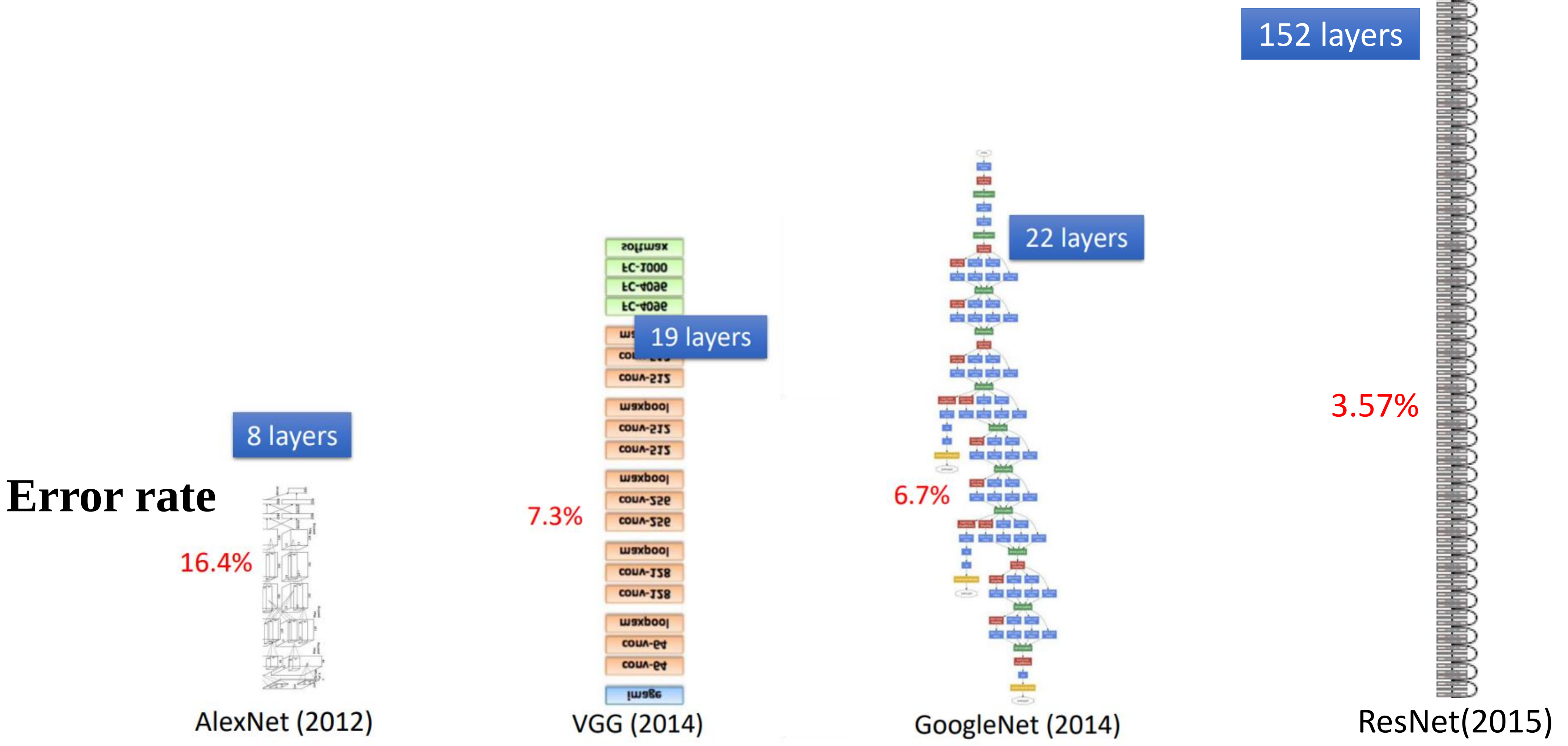
Residual Block



$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i),$$

$$\frac{\partial \mathcal{E}}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L} \frac{\partial \mathbf{x}_L}{\partial \mathbf{x}_l} = \frac{\partial \mathcal{E}}{\partial \mathbf{x}_L} \left(1 + \frac{\partial}{\partial \mathbf{x}_l} \sum_{i=l}^{L-1} \mathcal{F}(\mathbf{x}_i, \mathcal{W}_i) \right).$$

ImageNet Challenge(Image Classification)



ResNet Architecture

減少記憶體消耗與運算量

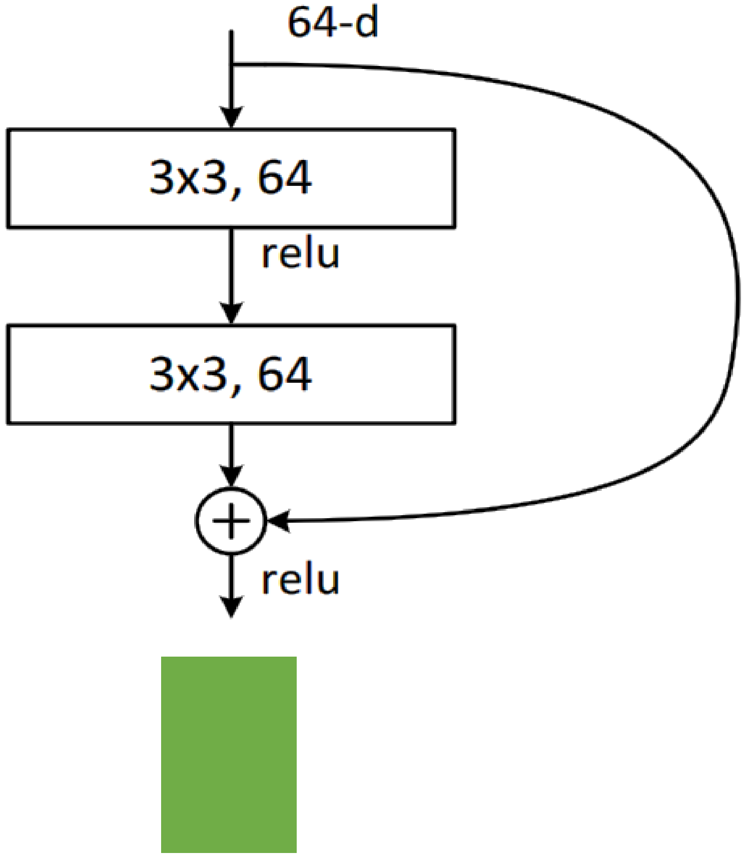
- ResNe18(Basic block), ResNet50(Bottleneck block)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

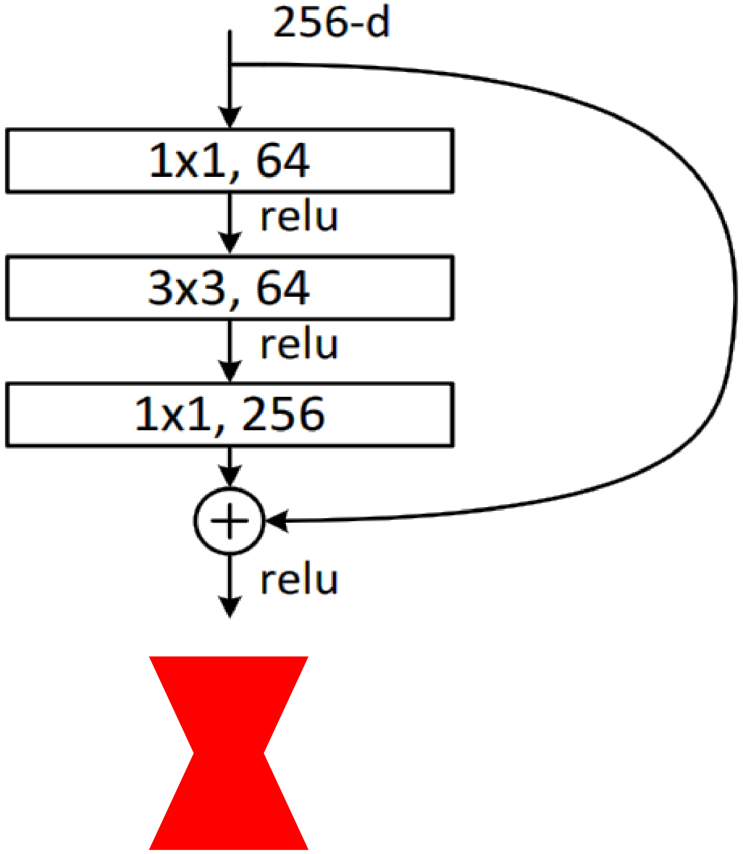
ResNet Block

- ResNe18(Basic block), ResNet50(Bottleneck block)

Basic block



Bottleneck block



Deeper is Better?

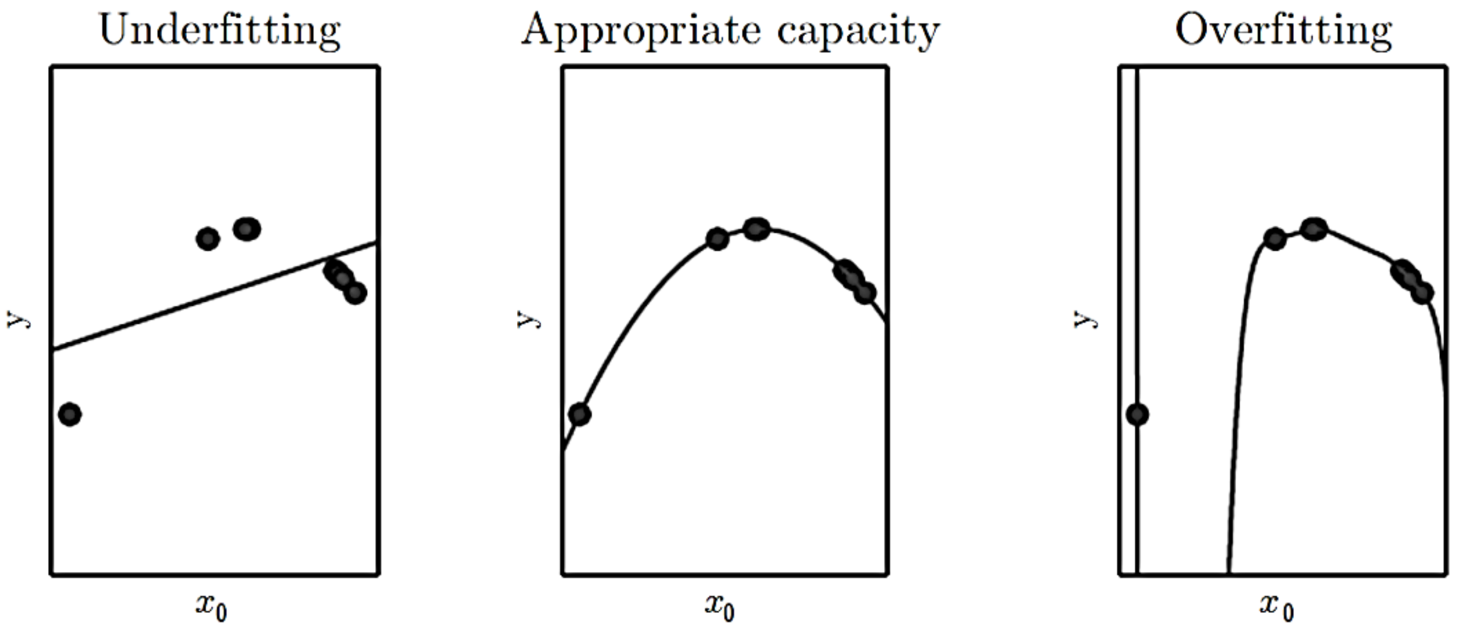
model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

	# layers	# params	
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93

- 網路太淺(參數少)，能力不足，所以效果相對較差
- 網路越深(參數多)，能力越強，可處理更複雜的問題

Model Capacity

- Models with insufficient capacity are unable to solve complex tasks
- Models with high capacity can solve complex tasks, but may be overfitting



$\hat{y} = b + wx$

$\hat{y} = b + w_1 x + w_2 x^2$

$\hat{y} = b + \sum_{i=1}^{9} w_i x^i$

Trade off
(Find optimal capacity)

Trial and error + Intuition