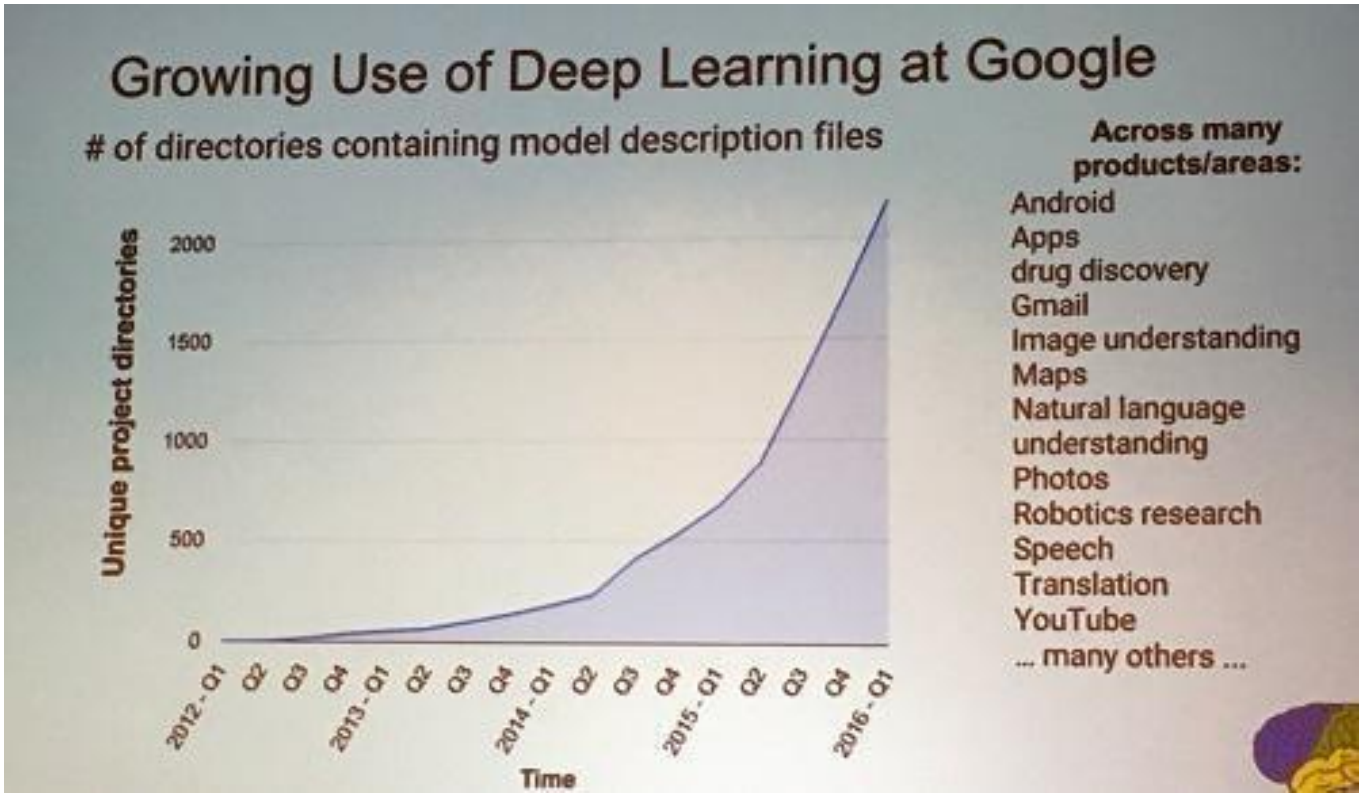
深度學習教程

台大李弘毅教授 http://speech.ee.ntu.edu.tw/~tlkagk/courses.html

深度學習得到大量關注

- 我相信你以前見過很多振奮人心的結果



在穀歌深度學習的趨勢。 資源: SIGMOD/Jeff Dean

這次演講側重基本技術

大綱

Lecture I:深度學習的介紹

Lecture II:訓練神經網路的技巧

Lecture III:神經網路的變體

Lecture IV:下一個浪潮

Lecture I:
深度學習的介紹

Outline of Lecture I

深度學習的介紹

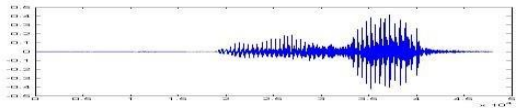
讓我們從一般的機器學習開始.

為什麼用深層網路?

深度學習簡單實戰

機器學習
≈ 尋找一個函數

- 語音辨識

$f($  $)=$“How are you”

- 圖片識別

$f($  $)=$“Cat”

- 下圍棋

$f($  $)=$“5-5” (下一步棋)

- 對話系統

$f($ “Hi” $)=$ “Hello”

(你說的內容) (系統的回答)

框架

圖片識別:

$f($  $)=$ “cat”

一系列函數

模型

$f_1, f_2 \cdots$

$f_1($  $)=$ “cat”

$f_2($  $)=$ “money”

$f_1($  $)=$ “dog”

$f_2($  $)=$ “snake”

框架

圖像識別:

$f($  $)=$ “cat”

一系列函數

模型

$f_1, f_2 \cdots$

$f_1($  $)=$ “cat”

$f_2($  $)=$ “money”

Better!

$f_1($  $)=$ “dog”

$f_2($  $)=$ “snake”

模型和資料的擬合

訓練數據

監督學習

函數輸入:   

函數輸出: “monkey” “cat” “dog”

框架

圖像識別:

$f($  $)=$ “cat”

訓練

測試

一系列函數

模型

$f_1, f_2 \cdots$

Step 1

模型和資料的擬合

Step 2

選擇最優函數

f^*

Step 3

訓練數據

“monkey”

“cat”

“dog”

“cat”

Using f^*

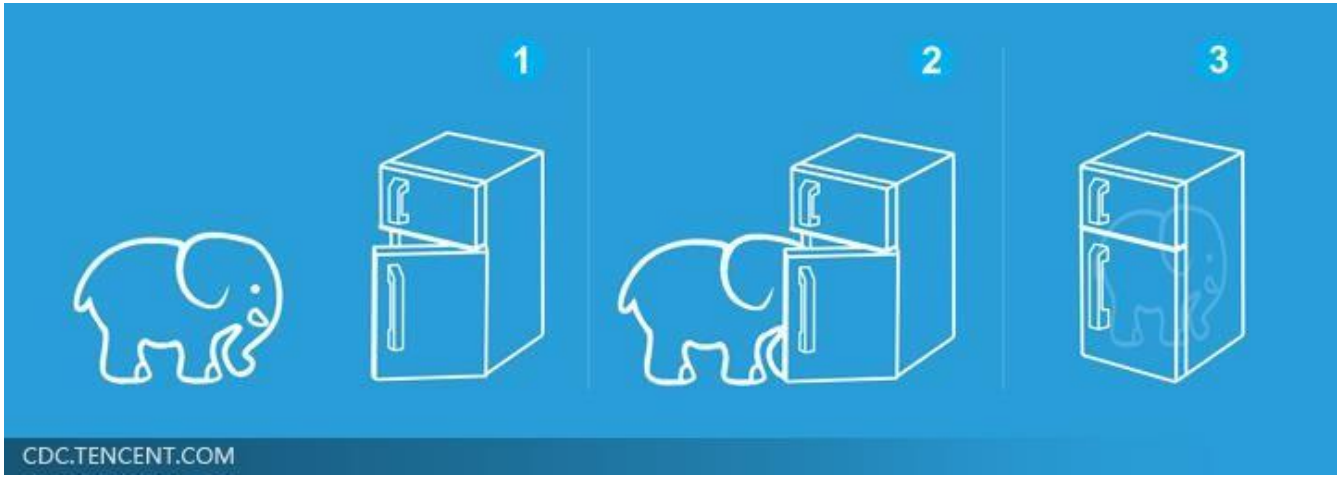
深度學習的三個步驟

步驟 1:
定義一組函數

步驟 2:
模型和資料的擬合

步驟 3:
選擇最優函數

深度學習是如此簡單.....



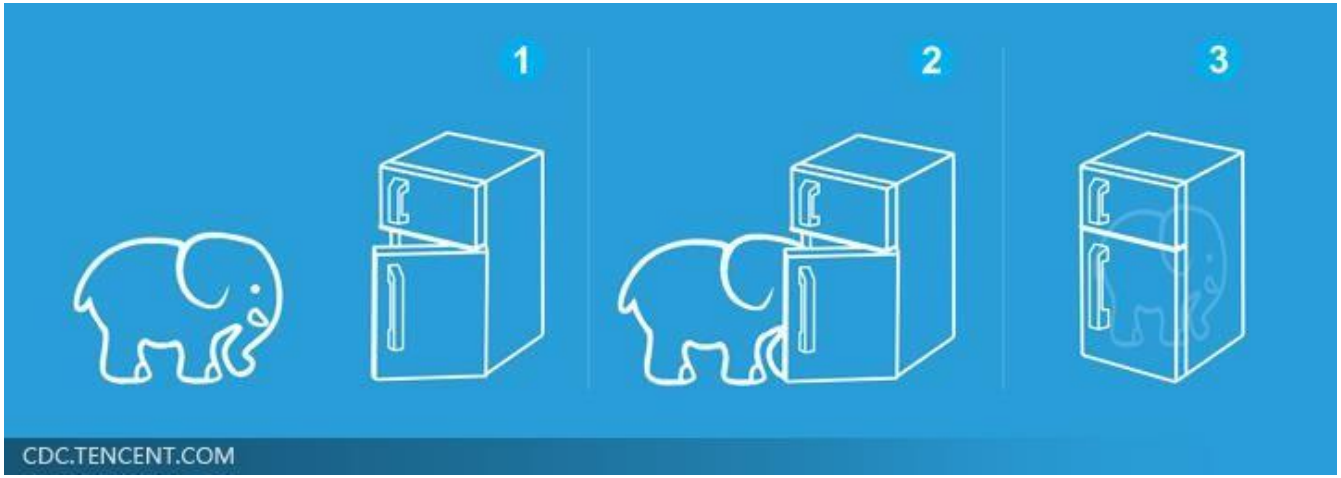
深度學習的三個步驟

步驟1:
神經網絡

步驟 2:
模型和數據的擬合

步驟 3:
選擇最優函數

深度學習是如此簡單.....



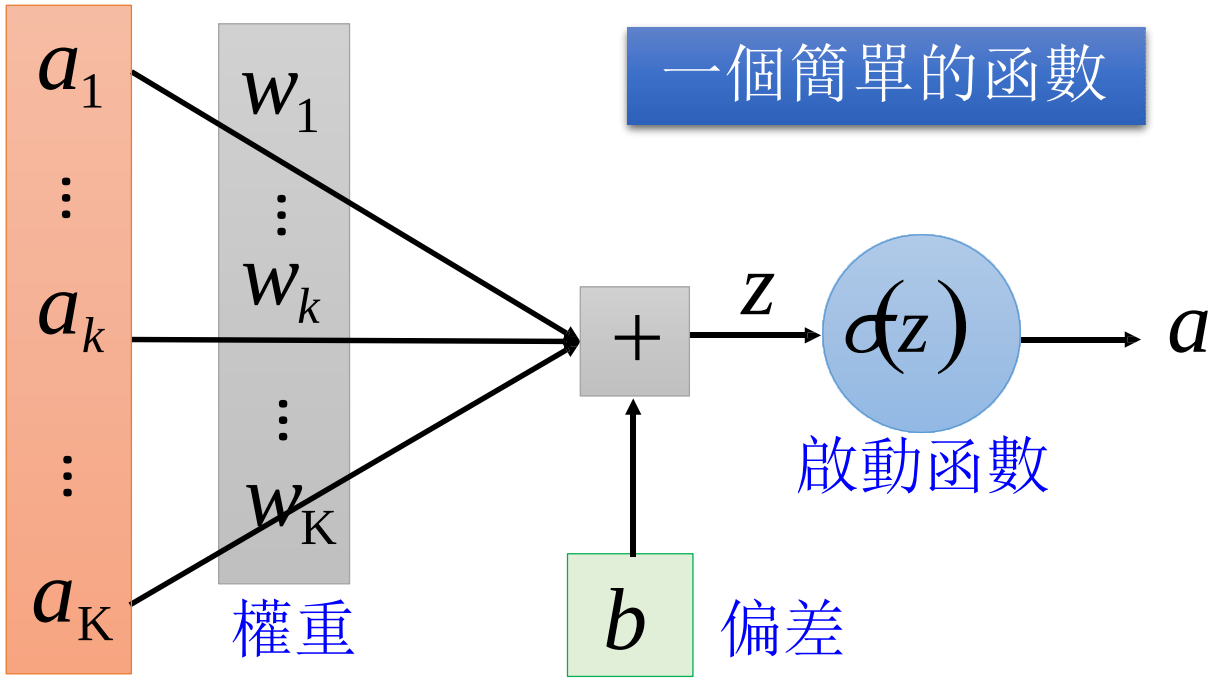
人類的大腦

Dendrites
Stimulus
Nucleus
Presynaptic cell
Axon hillock
Cell body
Axon
Synapse
Synaptic terminals
Postsynaptic cell
Neurotransmitter

神經網路

神經元

$$z = a_1 w_1 + \cdots + a_k w_k + \cdots + a_K w_K + b$$

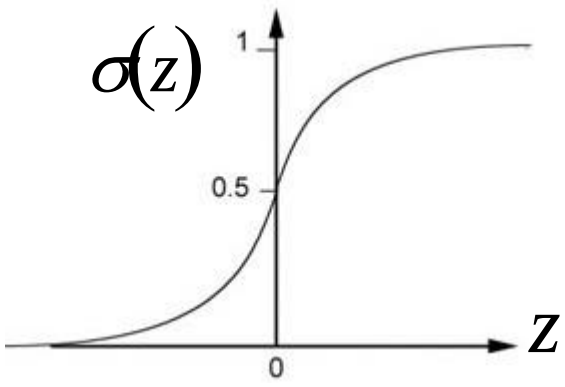


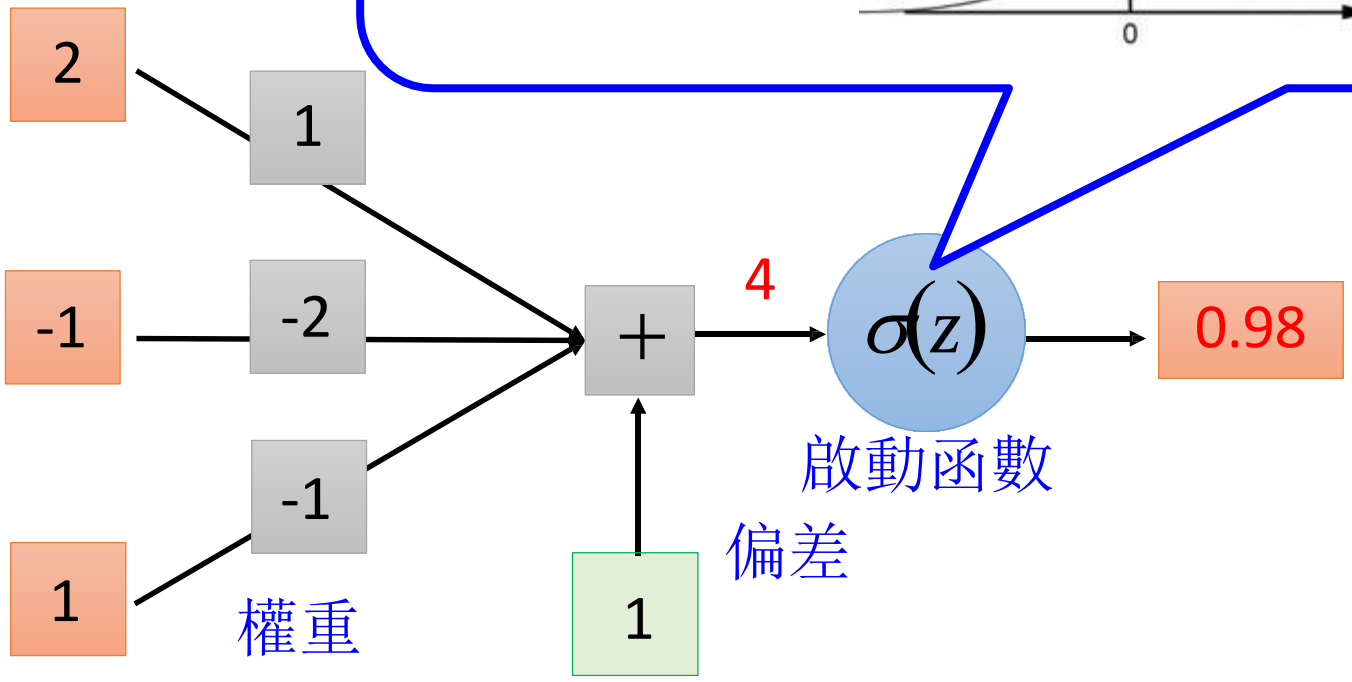
神經網路

神經元

Sigmoid 函數

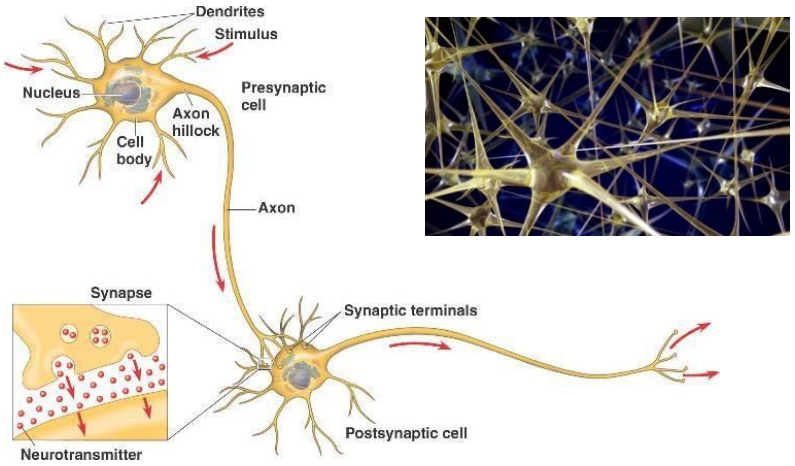
$$\sigma(z) = \frac{1}{1+e^{-z}}$$

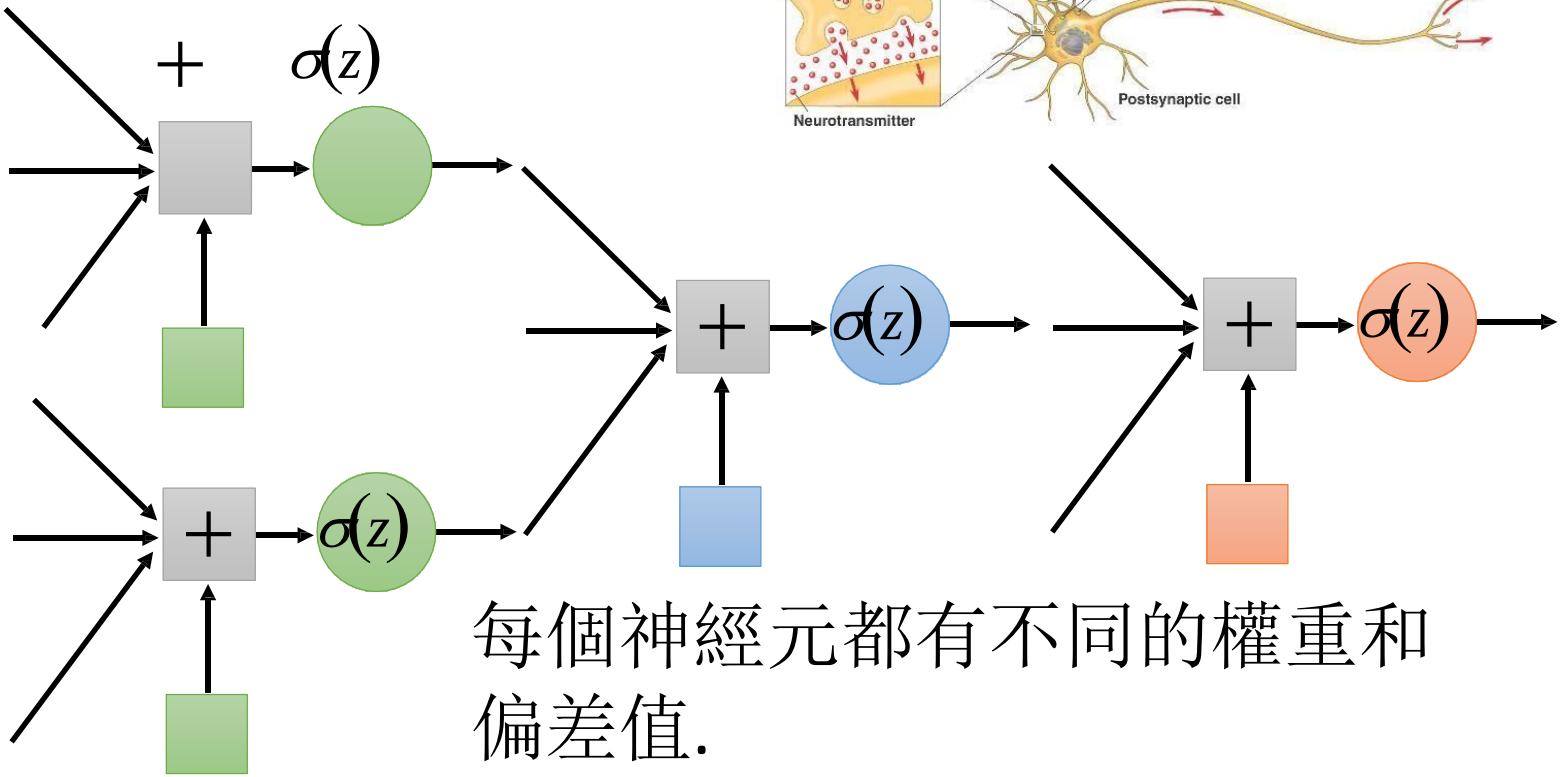




神經網路

不同的連接導致不同的網路結構

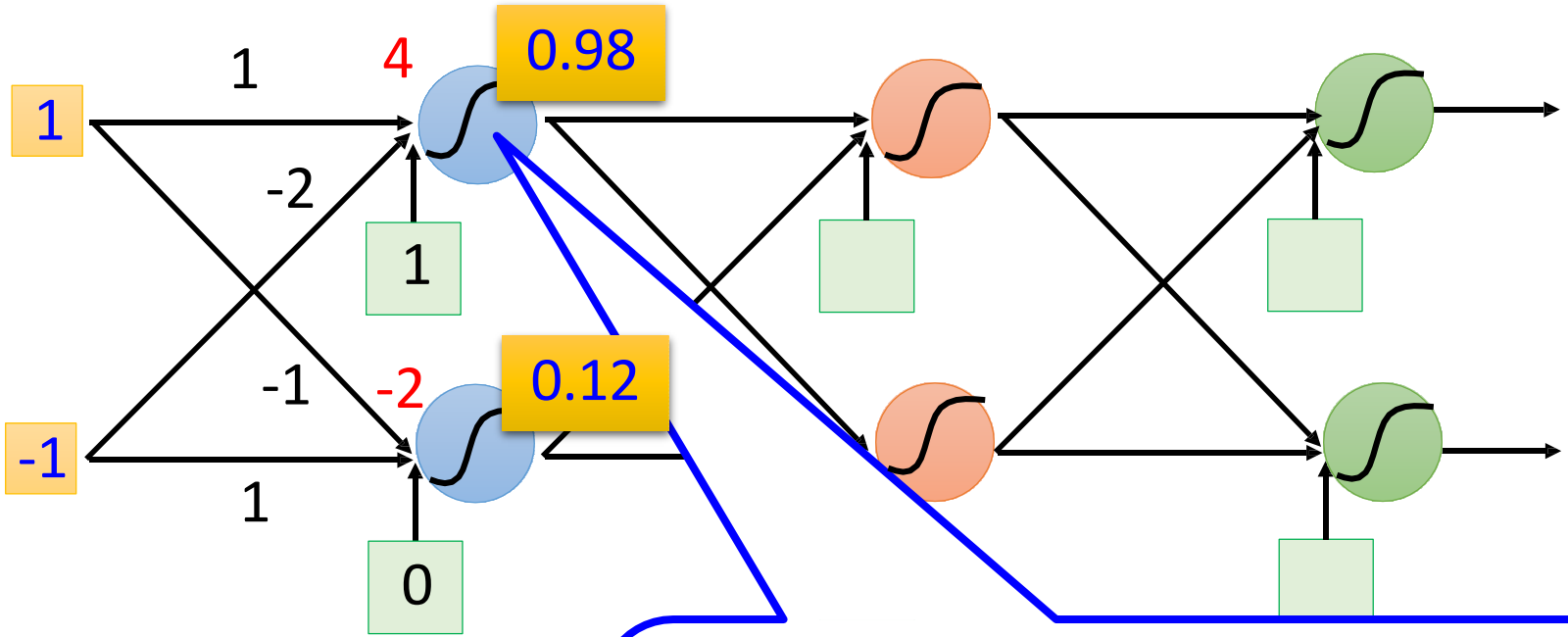




每個神經元都有不同的權重和偏差值.

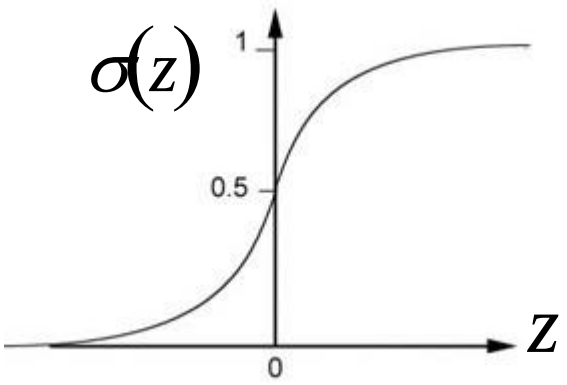
權重和偏差都是神經網路的參數 θ

完全連接前饋網路

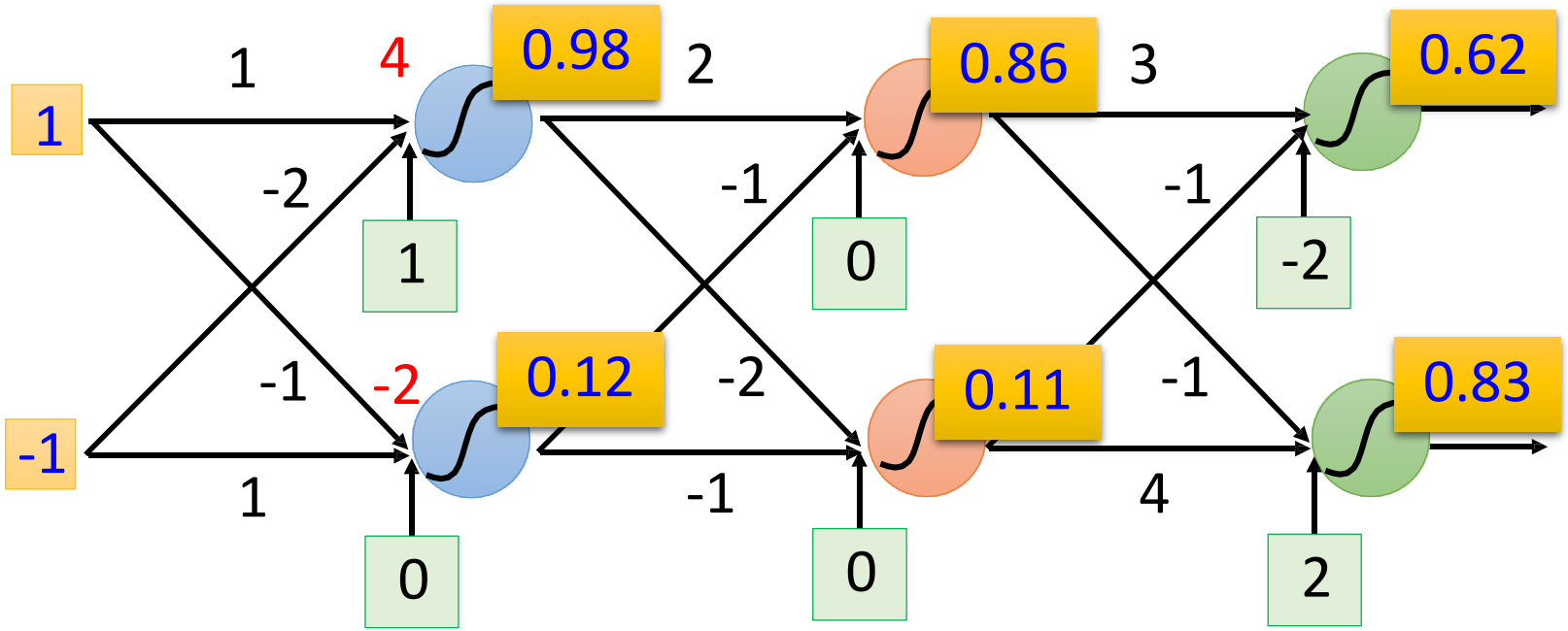


Sigmoid 函數

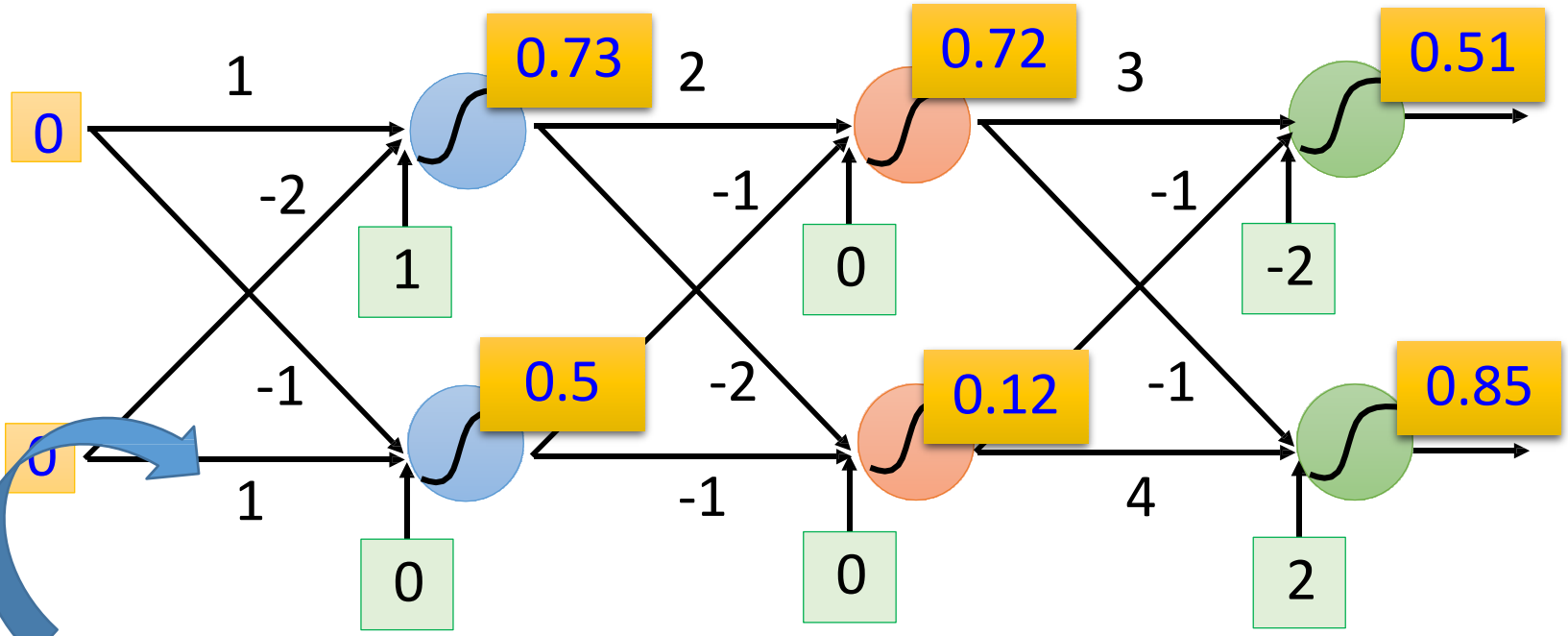
$$\sigma(z)=\frac{1}{1+e^{-z}}$$



完全連接前饋網路



完全連接前饋網路



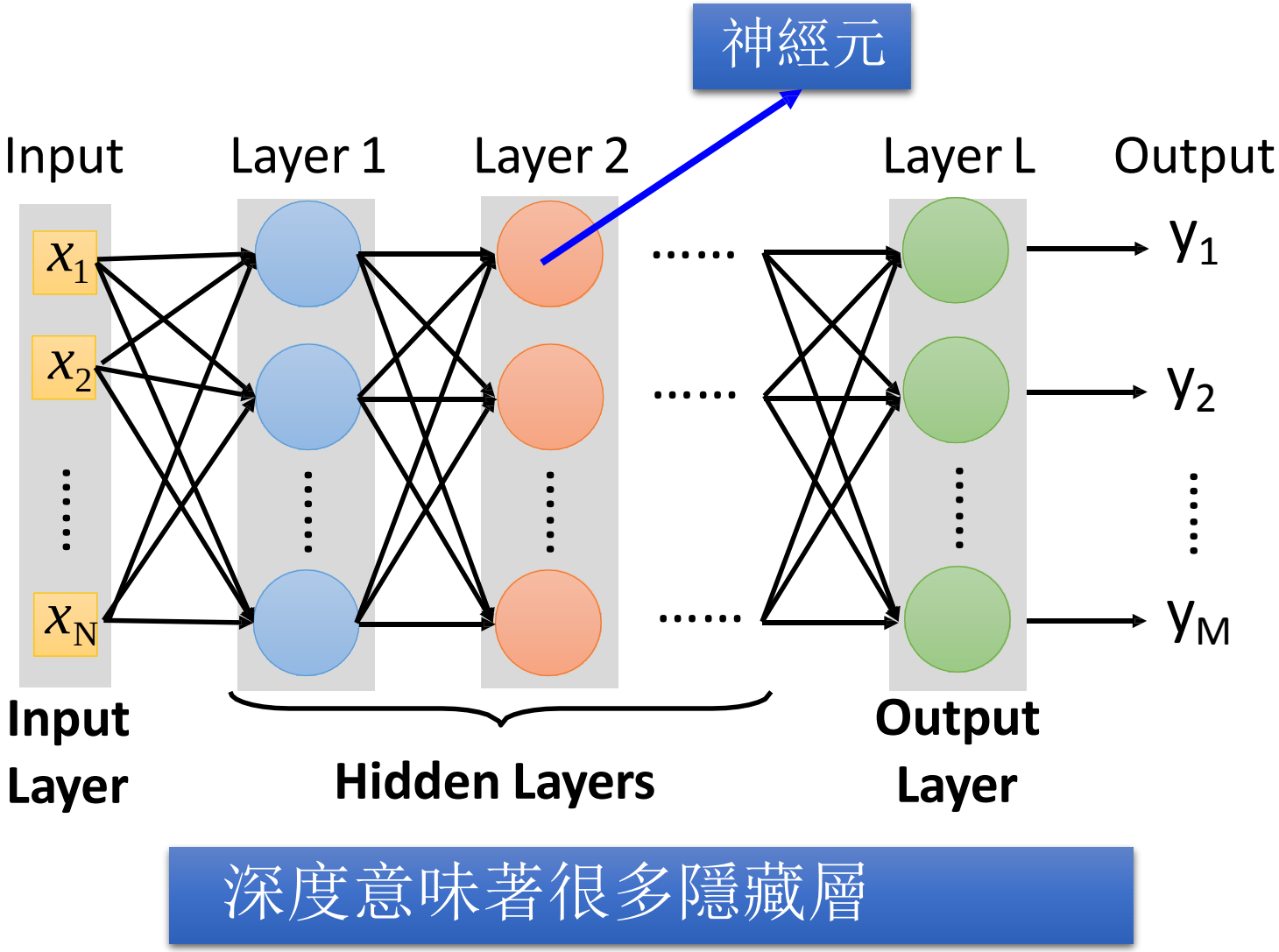
這是一個函數.
輸入向量, 輸出向量

$$f\left(\begin{bmatrix}1\\-1\end{bmatrix}\right)=\begin{bmatrix}0.62\\0.83\end{bmatrix} \quad f\left(\begin{bmatrix}0\\0\end{bmatrix}\right)=\begin{bmatrix}0.51\\0.85\end{bmatrix}$$

給定參數 θ, 定義一個函數

給定網路結構，定義一個函數集

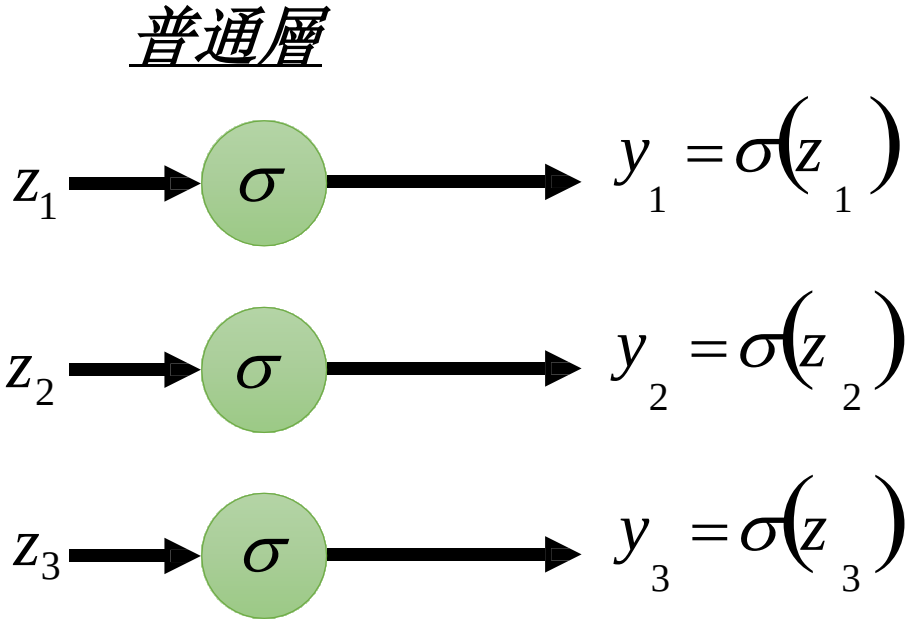
完全連接前饋網路



深度意味著很多隱藏層

輸出層(Option)

- Softmax 層作為輸出層



一般來說，網路的輸出可以是任何值.

可能不容易解釋

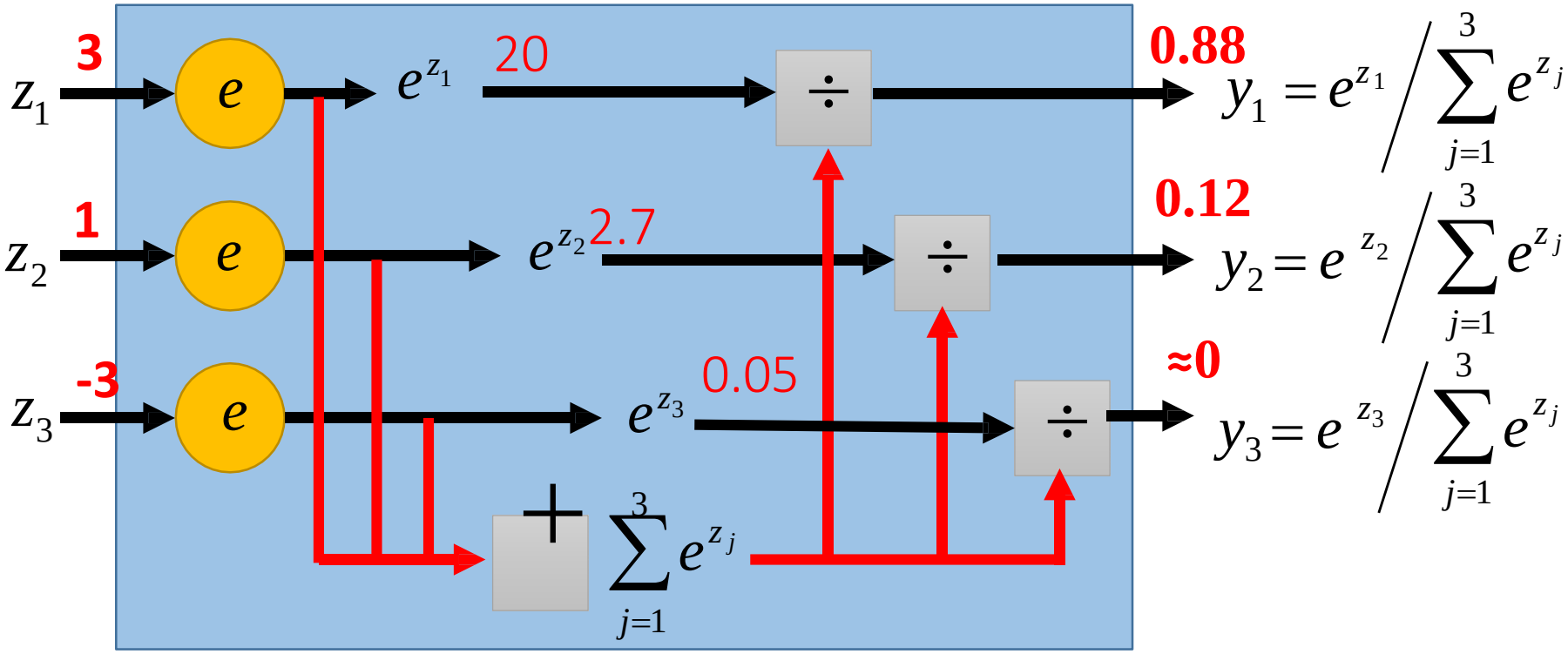
輸出層(Option)

- Softmax 層作為一個輸出層

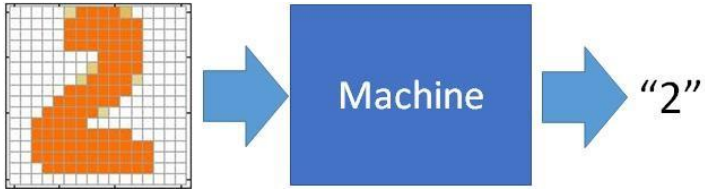
Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$

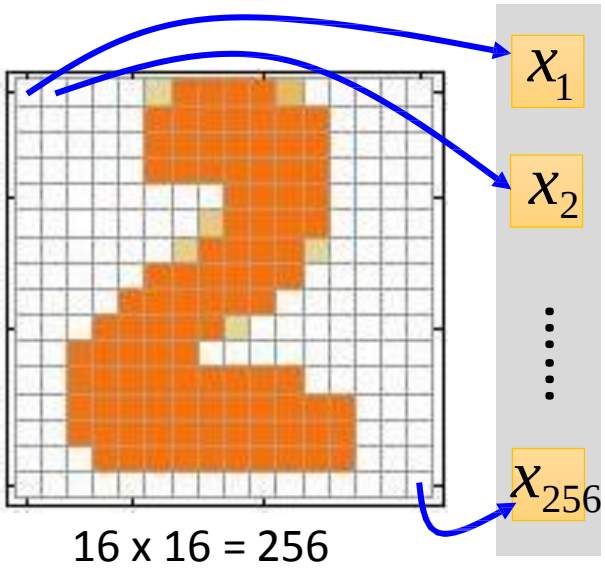
Softmax Layer



示例



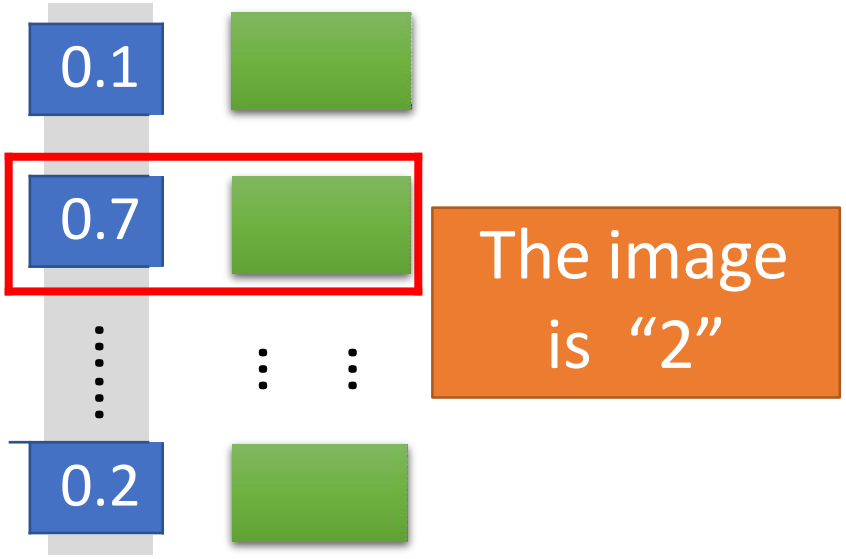
輸入



Ink → 1

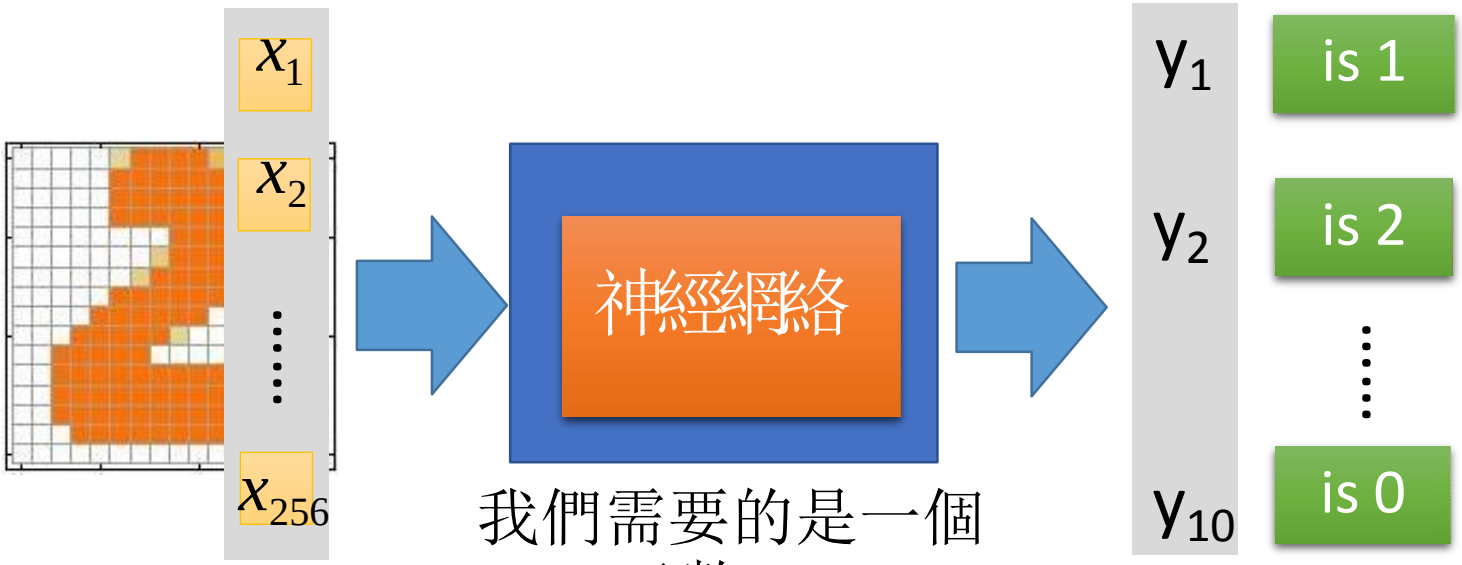
No ink → 0

輸出

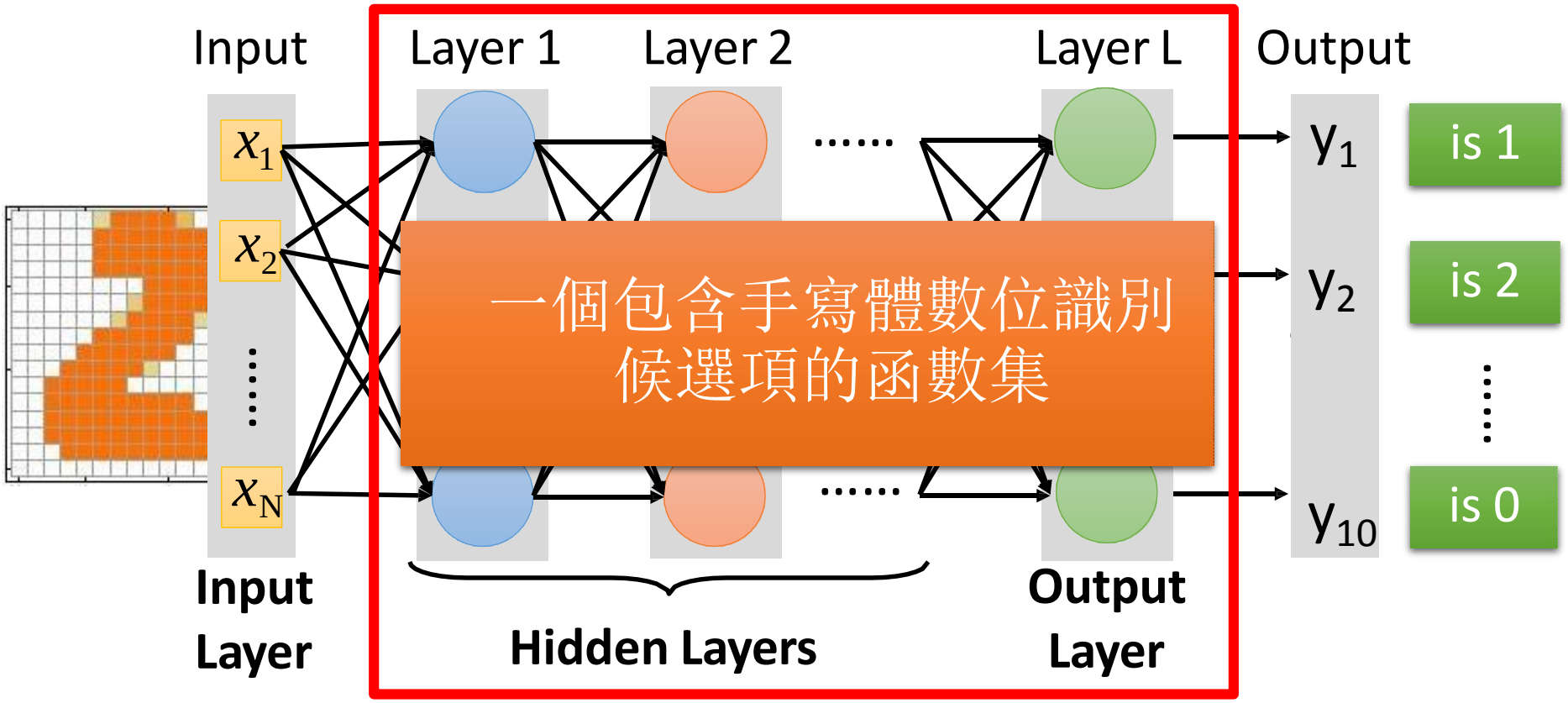


示例

- 書寫體數字識別

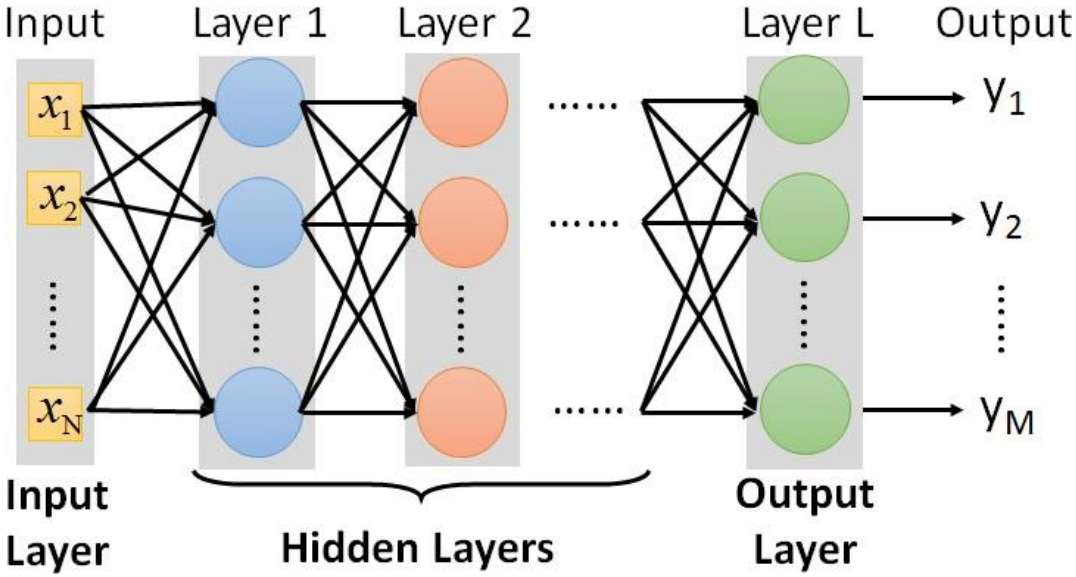

x_1
x_2
x_{256}
神經網絡
我們需要的是一個函數 ……
y_1
y_2
y_{10}
is 1
is 2
is 0
Input:
256-dim vector
output:
10-dim vector

示例



你需要決定網路結構，讓你的函數有好的功能

FAQ



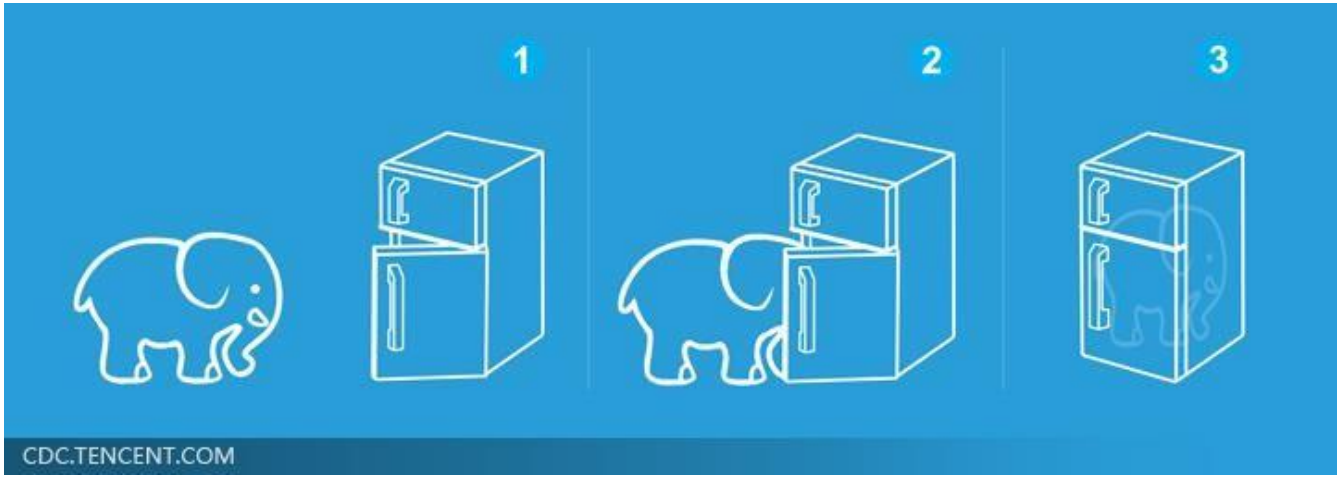
- Q: 多少層?每層有多少個神經元?

試驗和錯誤 + 直覺

- Q: 網路結構是否可以自動確定?

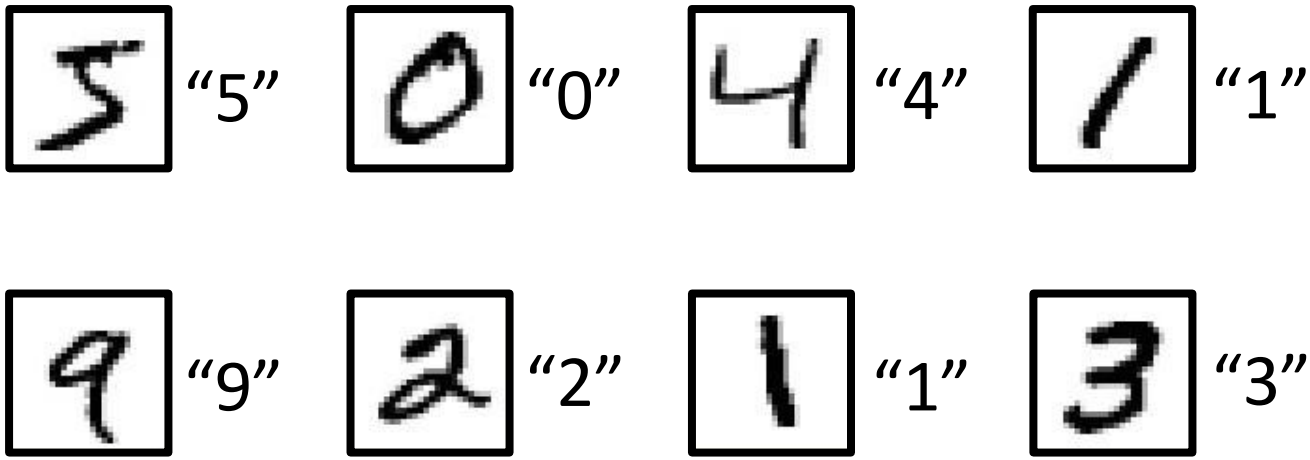
Three Steps for Deep Learning

深度學習是如此簡單 ……



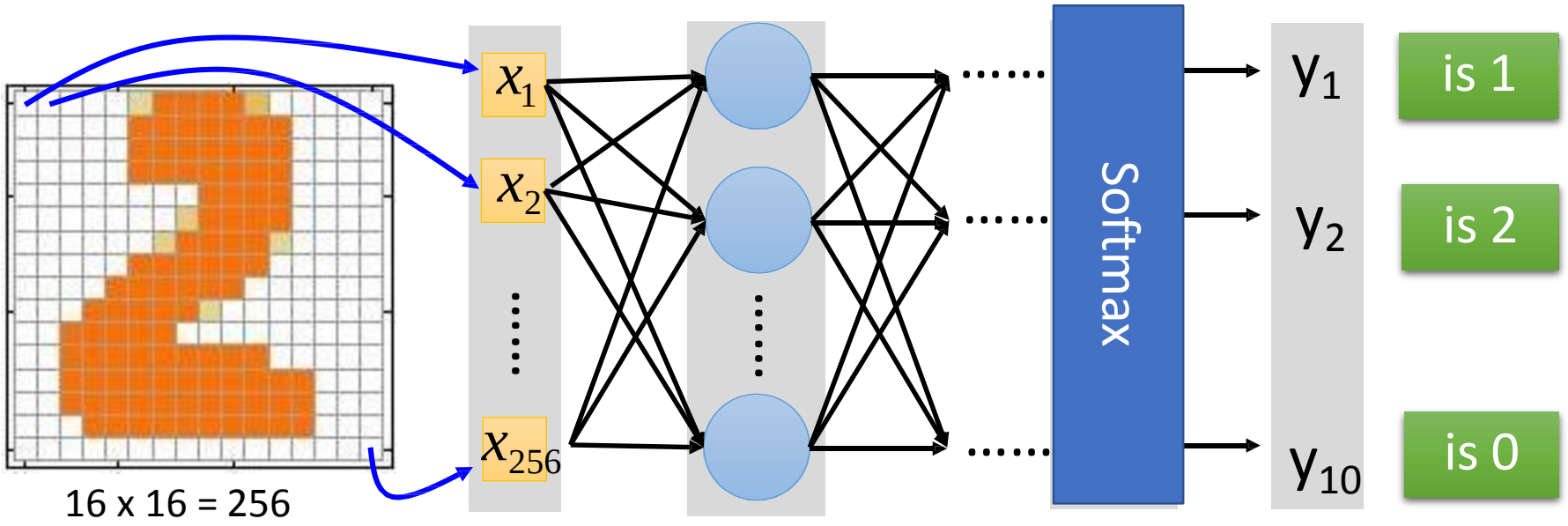
訓練數據

- 準備訓練數據:圖片和他們的標籤



學習目標是在訓練資料上定義的

學習目標



16 x 16 = 256

Ink → 1

No ink → 0

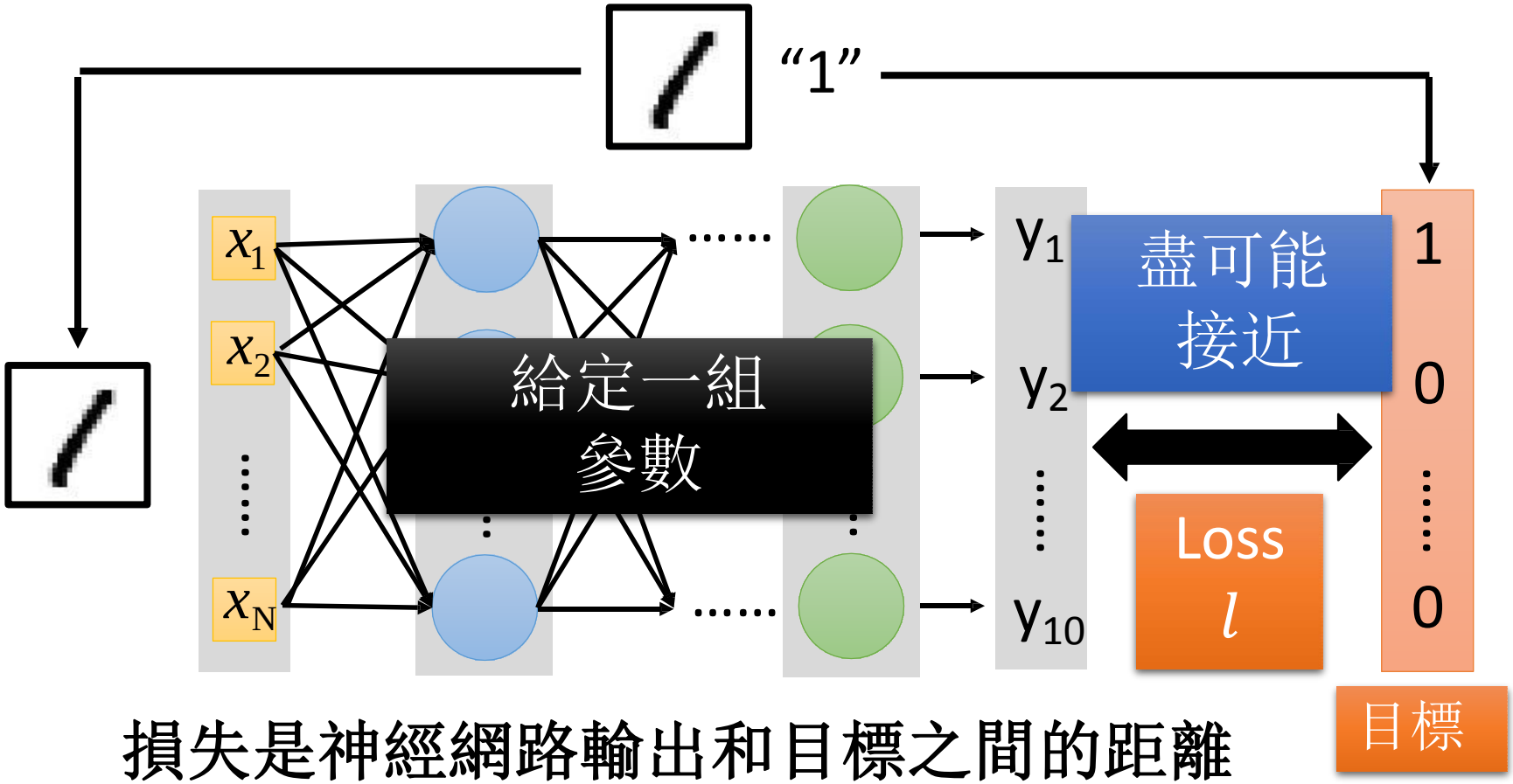
學習目標是 ……

Input:  → y_1 has the maximum value

Input:  → y_2 has the maximum value

損失

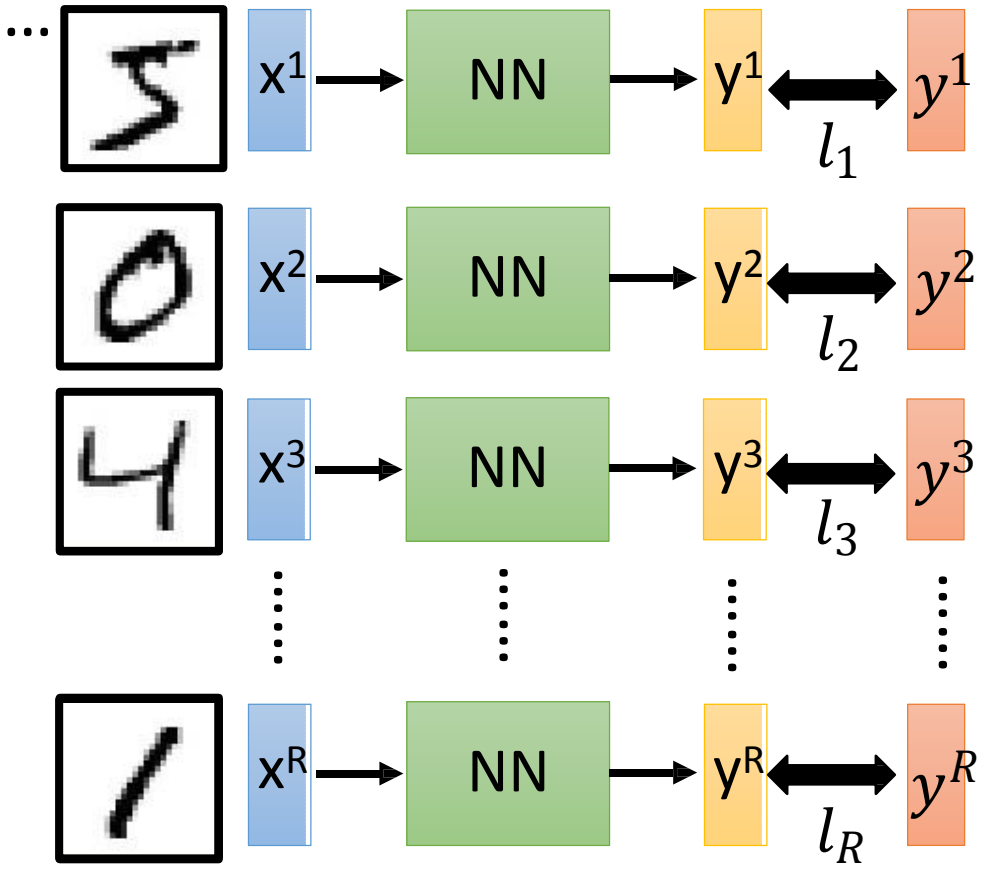
一個好的函數盡可能使損失最低



損失是神經網路輸出和目標之間的距離

全部損失

對所有的訓練資料



全部損失:

$$L = \sum_{r=1}^{R} l_r$$

盡可能的小

找到一個函數使損失函數 L 最小

找到參數 θ 使損失函數 L 最小

Three Steps for Deep Learning

Step 1:

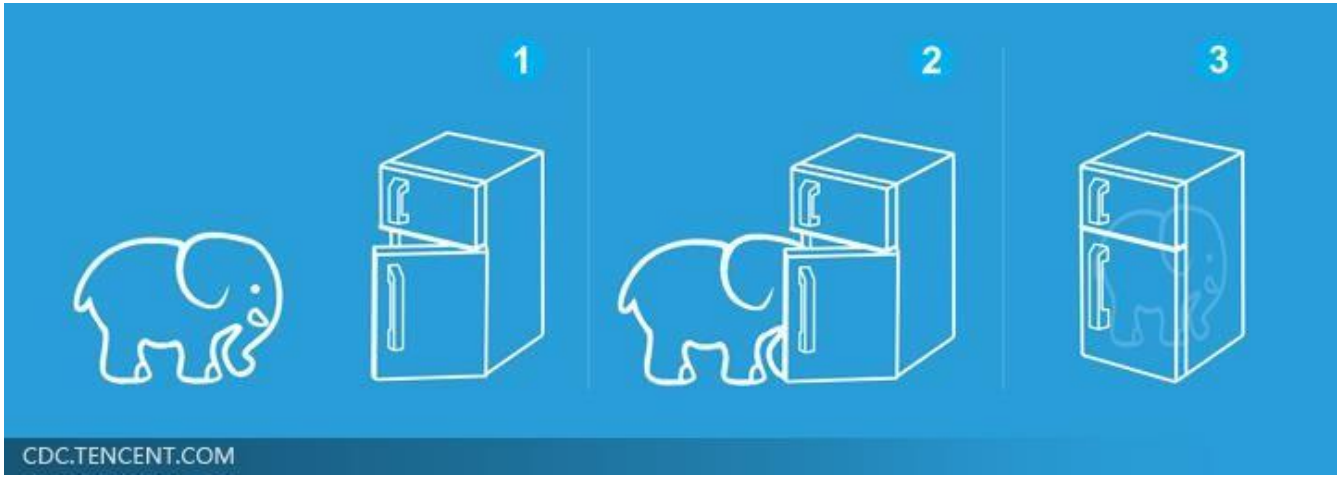
神經網路

Step 2:

模型和資料的擬合

Step 3: 選擇最優函數

深度學習如此簡單……



怎樣選擇最好的函數

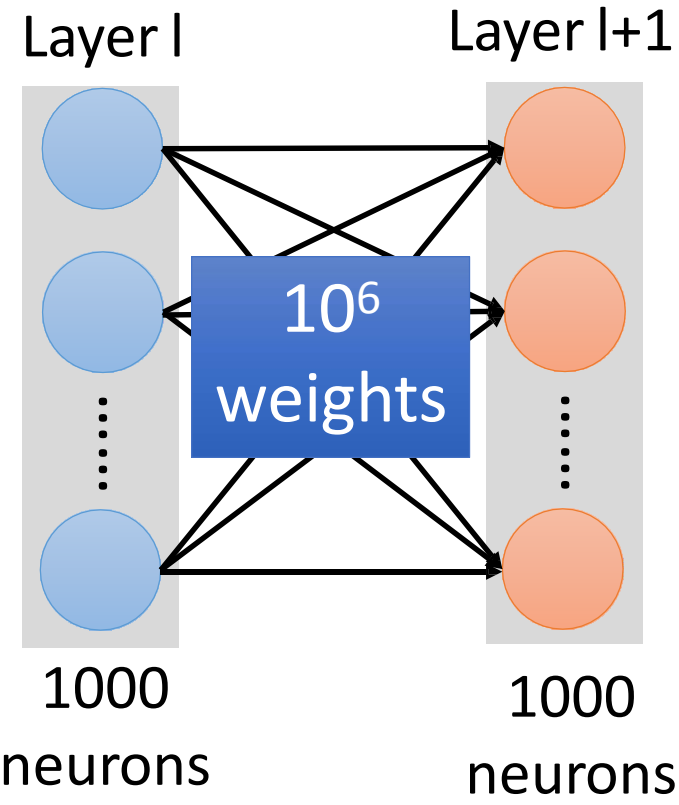
找到使總損失最小的參數 $\boldsymbol{\theta}$

列舉所有可能的值

Network parameters $\theta = \{w_1, w_2, w_3, \cdots, b_1, b_2, b_3, \cdots\}$

Millions of parameters

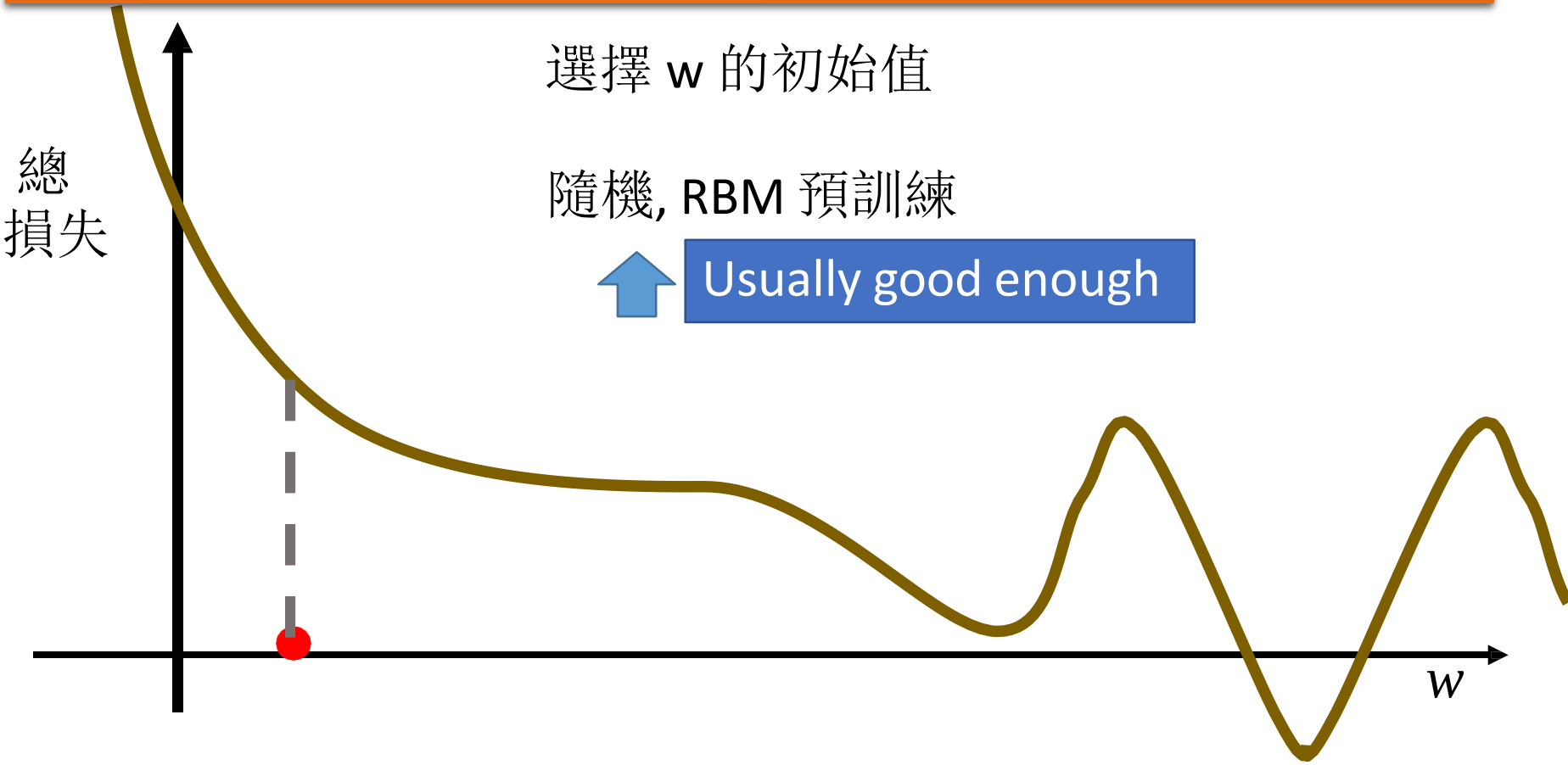
E.g. 語音辨識: 8 layers and 1000 neurons each layer



梯度下降

參數 $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

找到使總損失最小的參數$\boldsymbol{\theta}$

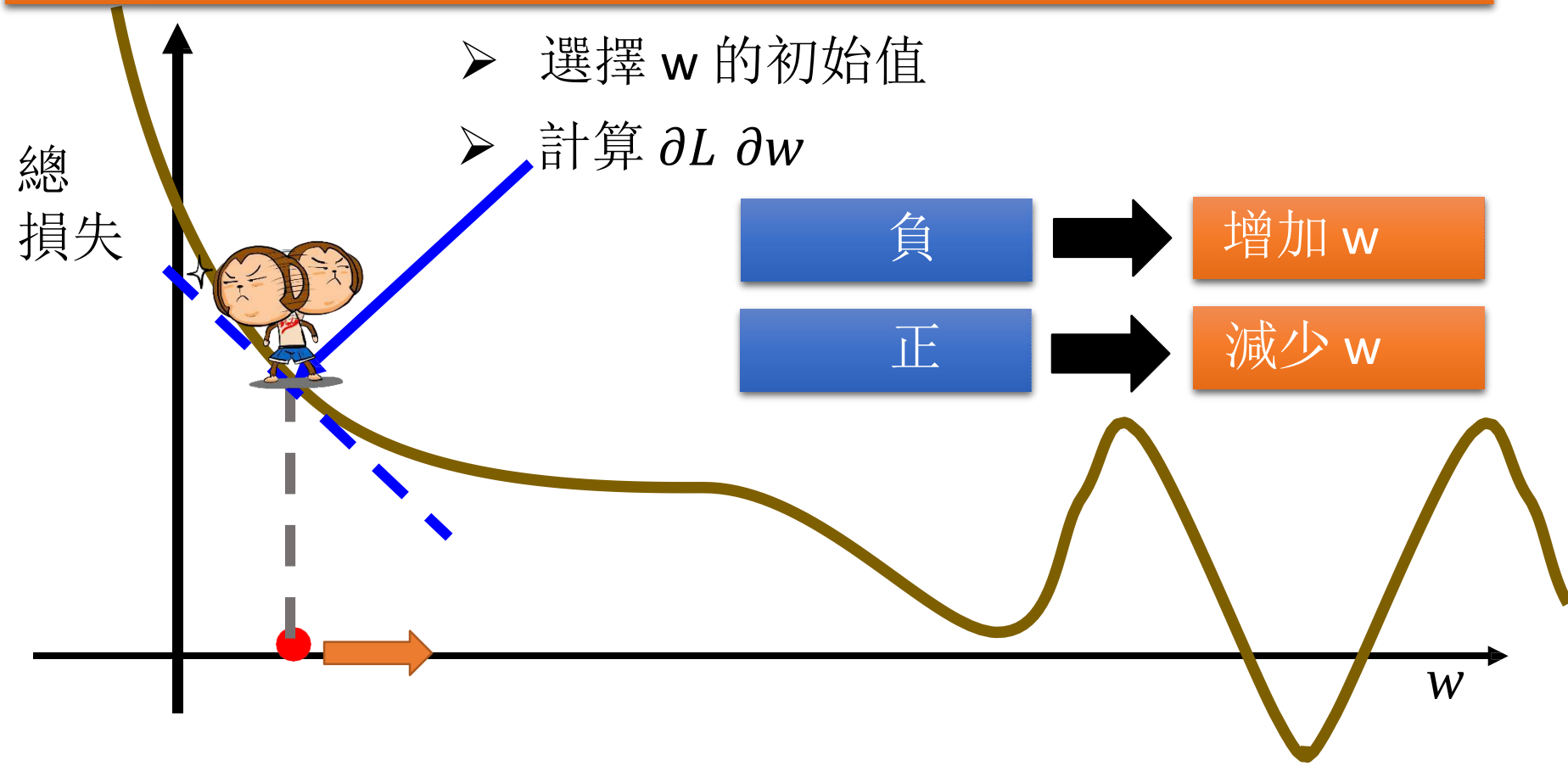


梯度下降

參數θ =
$\{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

找到使總損失最小的參數$\boldsymbol{\theta}^*$

- 選擇 w 的初始值
- 計算 $\partial L \ \partial w$



梯度下降

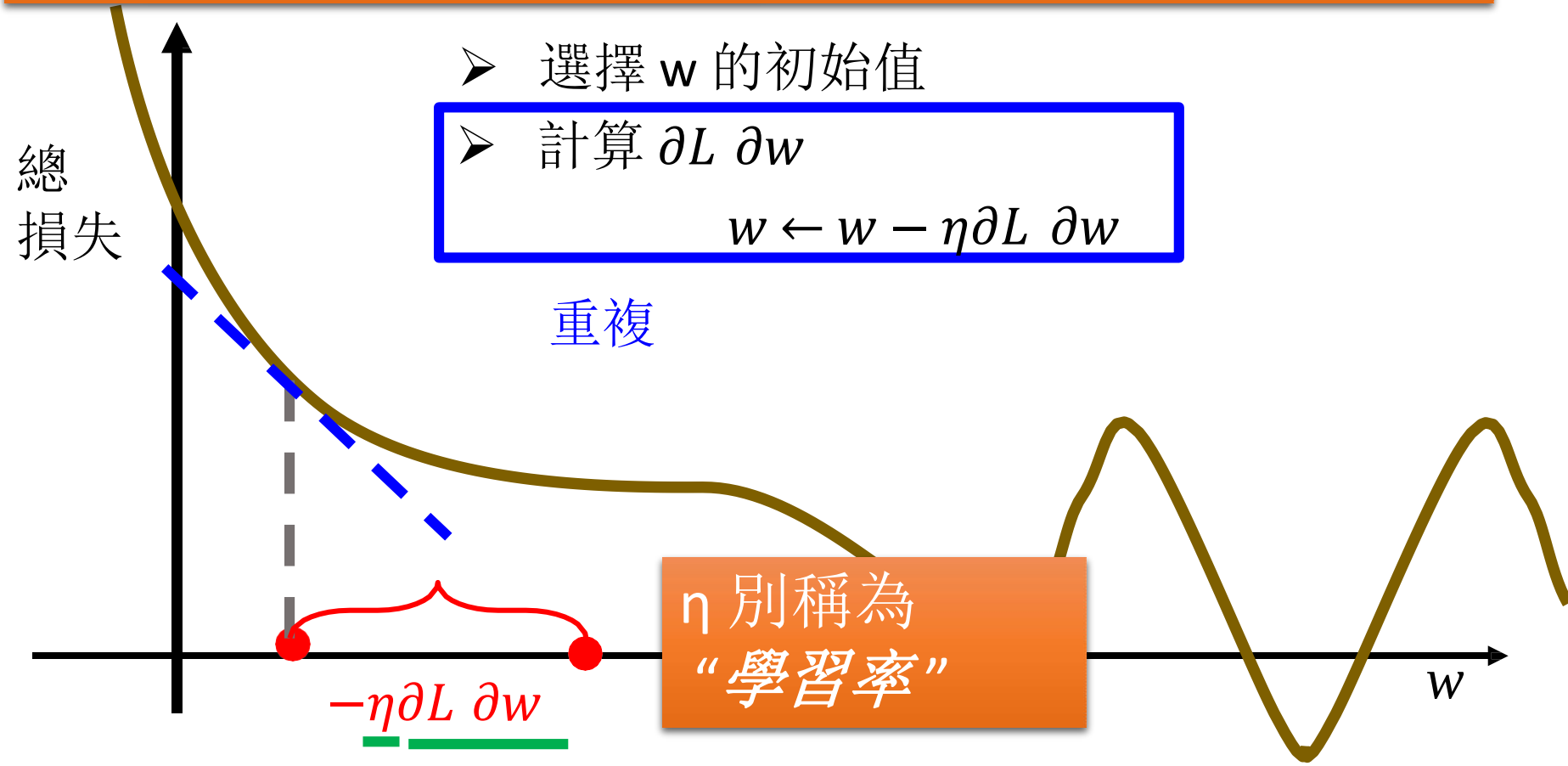
參數 $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

找到使總損失最小的參數$\boldsymbol{\theta}^*$

- 選擇 w 的初始值
- 計算 $\partial L \ \partial w$

$$w \leftarrow w - \eta \partial L \ \partial w$$

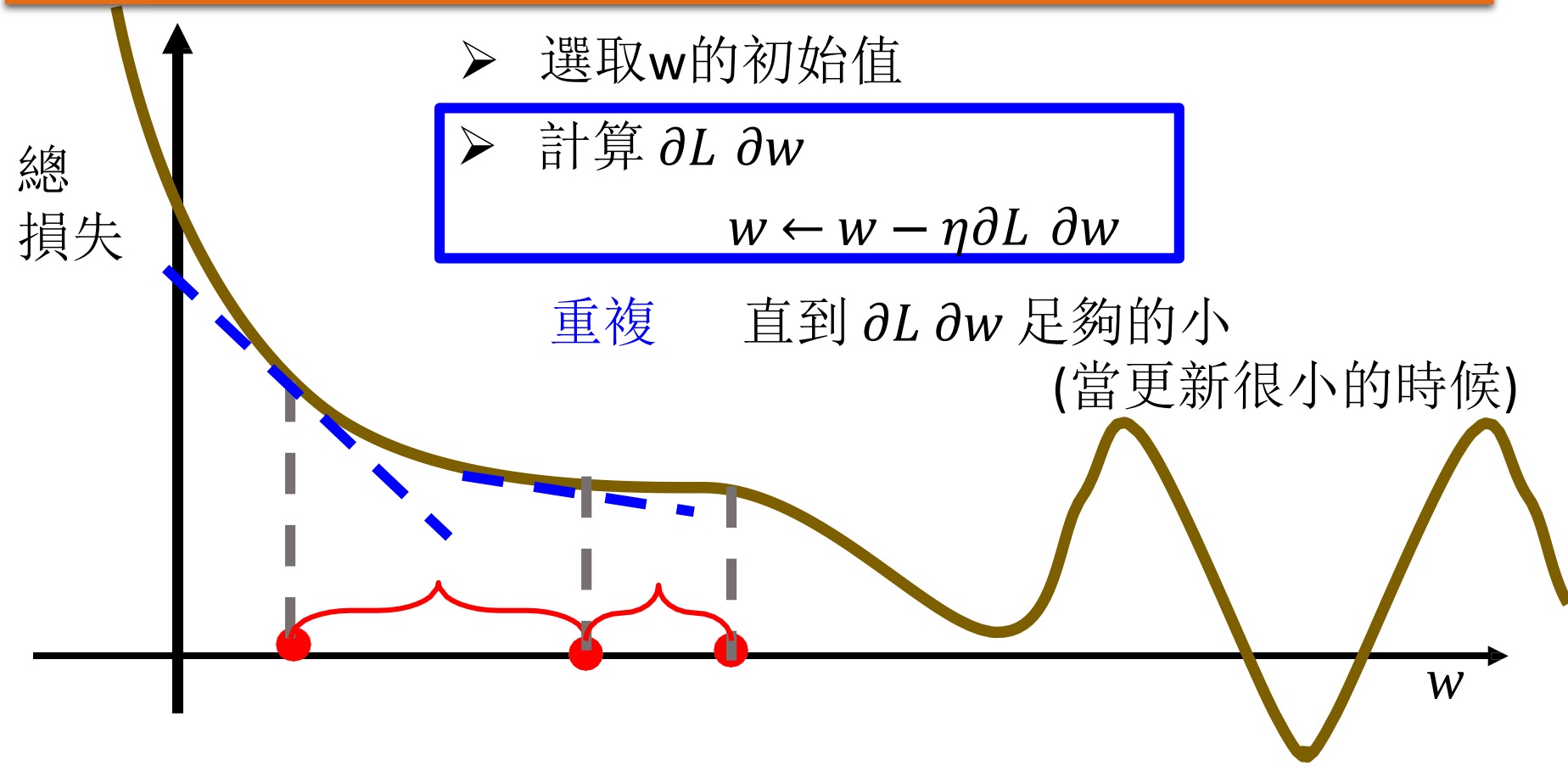
重複



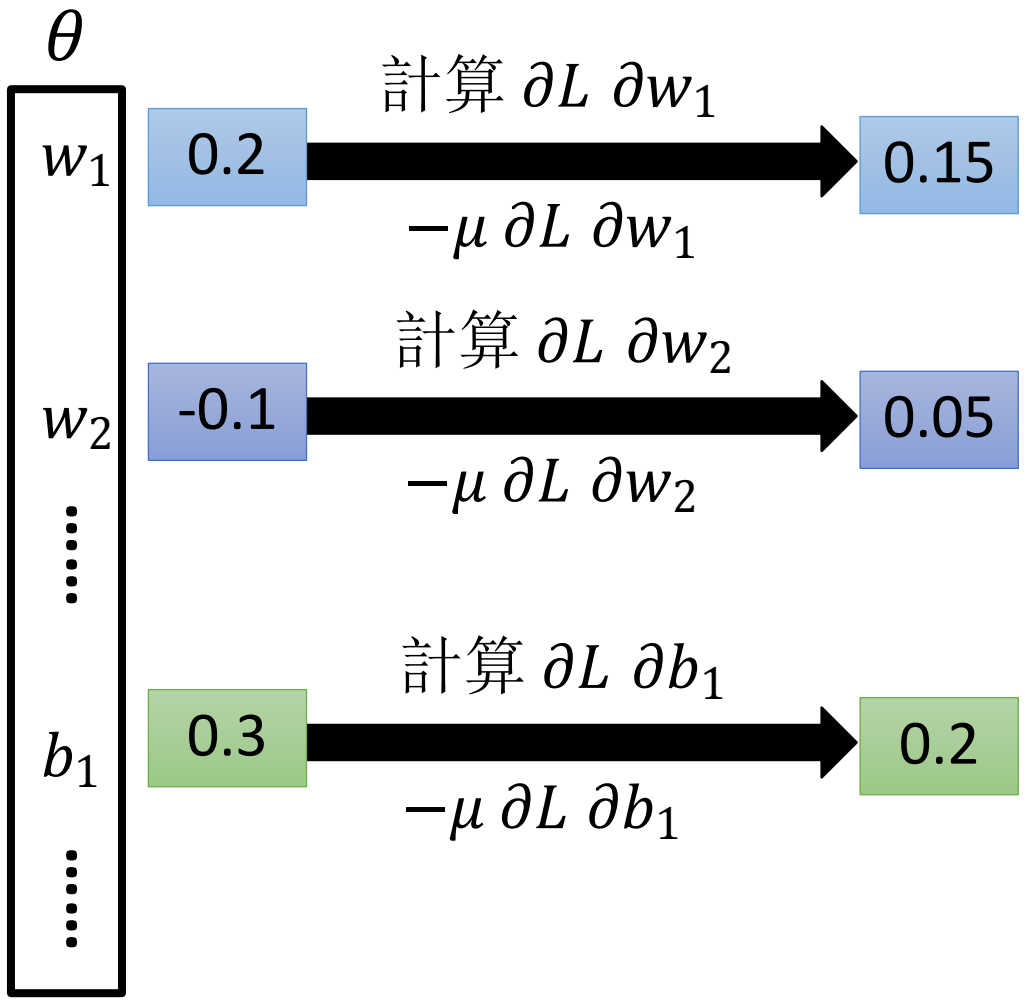
梯度下降

參數θ =
$\{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

找到使總損失最小的參數$\boldsymbol{\theta}^*$



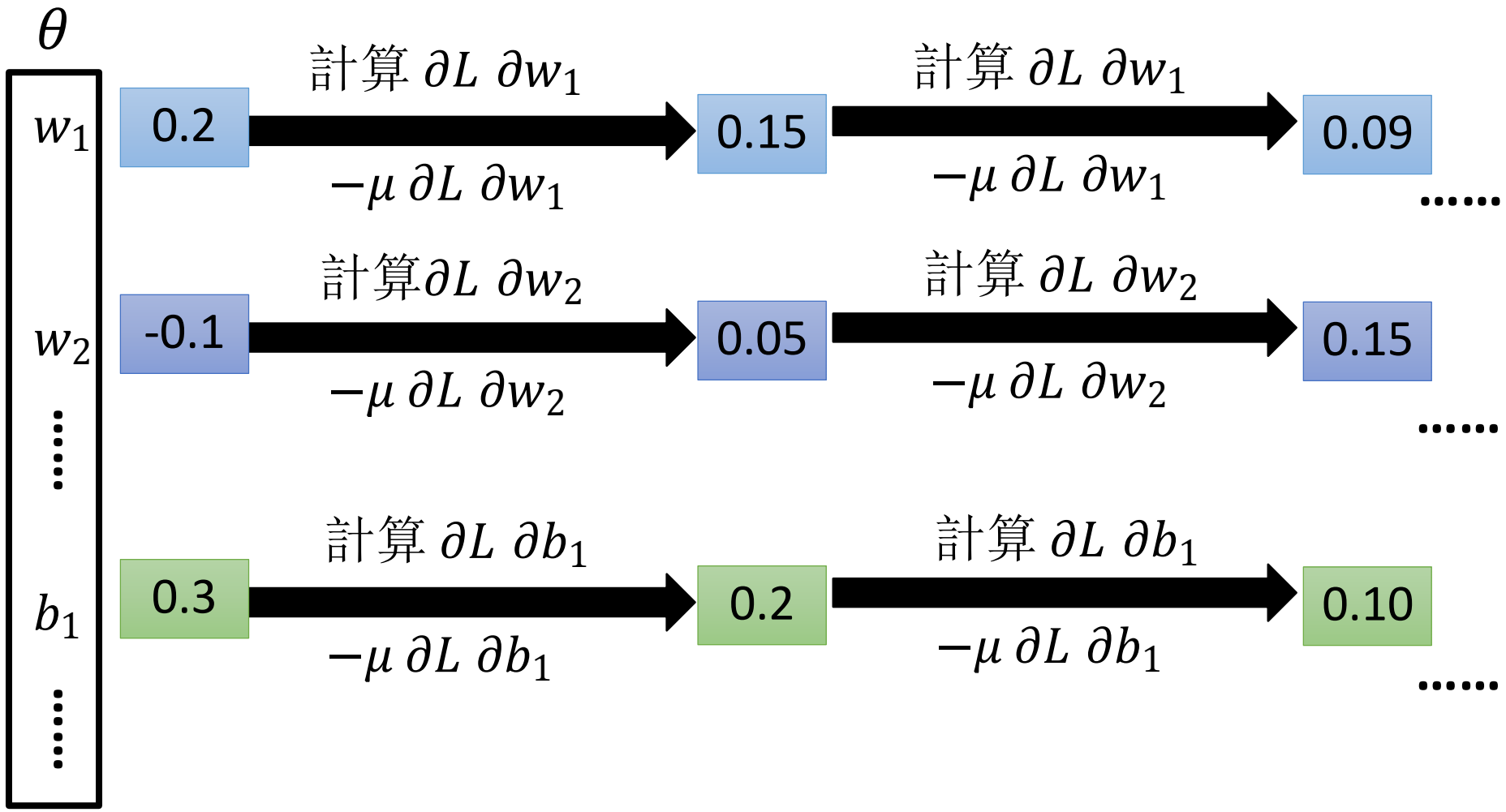
梯度下降



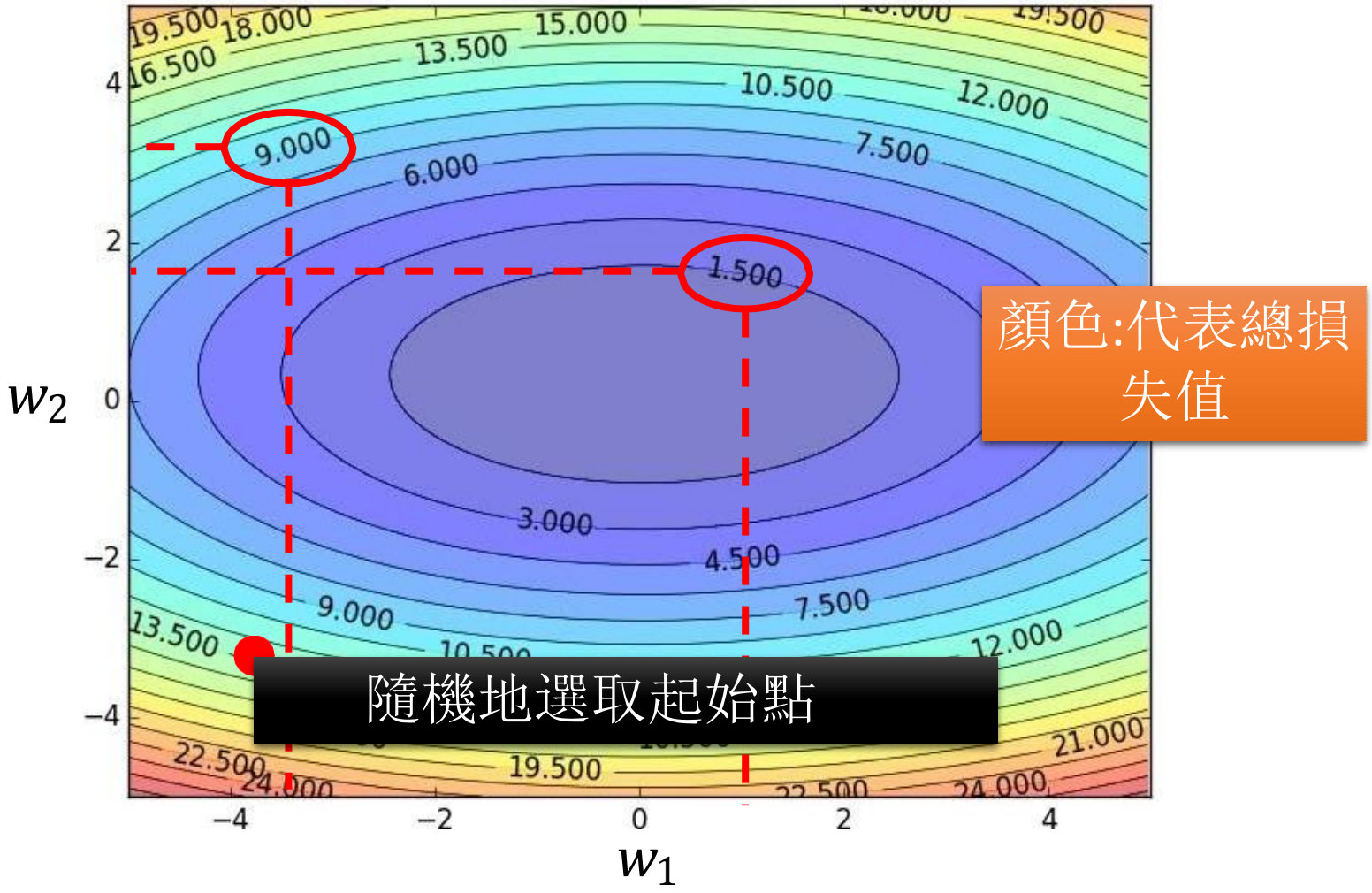
$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial w_1} \\ \frac{\partial L}{\partial w_2} \\ \vdots \\ \frac{\partial L}{\partial b_1} \\ \vdots \end{bmatrix}$$

梯度

梯度下降

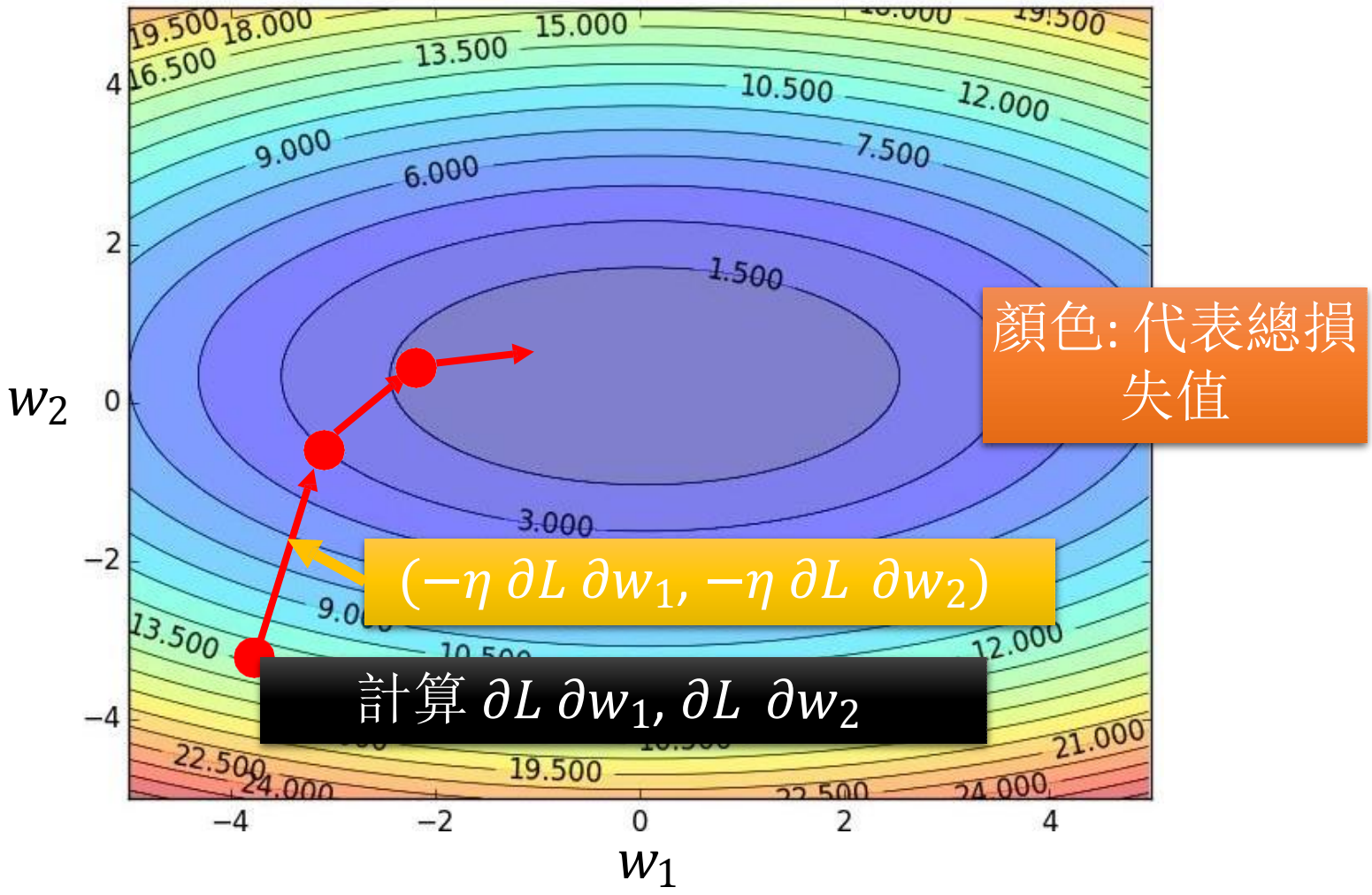

θ
w_1
w_2
b_1
0.2
-0.1
0.3
計算 ∂L ∂w_1
$-\mu\,\partial L$ ∂w_1
0.15
計算 ∂L ∂w_1
$-\mu\,\partial L$ ∂w_1
0.09
計算∂L ∂w_2
$-\mu\,\partial L$ ∂w_2
0.05
計算 ∂L ∂w_2
$-\mu\,\partial L$ ∂w_2
0.15
計算 ∂L ∂b_1
$-\mu\,\partial L$ ∂b_1
0.2
計算 ∂L ∂b_1
$-\mu\,\partial L$ ∂b_1
0.10

梯度下降


19.500
18.000
16.500
15.000
13.500
10.500
12.000
7.500
9.000
6.000
1.500
3.000
4.500
9.000
7.500
13.500
12.000
19.500
21.000
22.500
24.000
4
2
0
−2
−4
−4
−2
0
2
4
w_2
w_1
顏色:代表總損失值
隨機地選取起始點

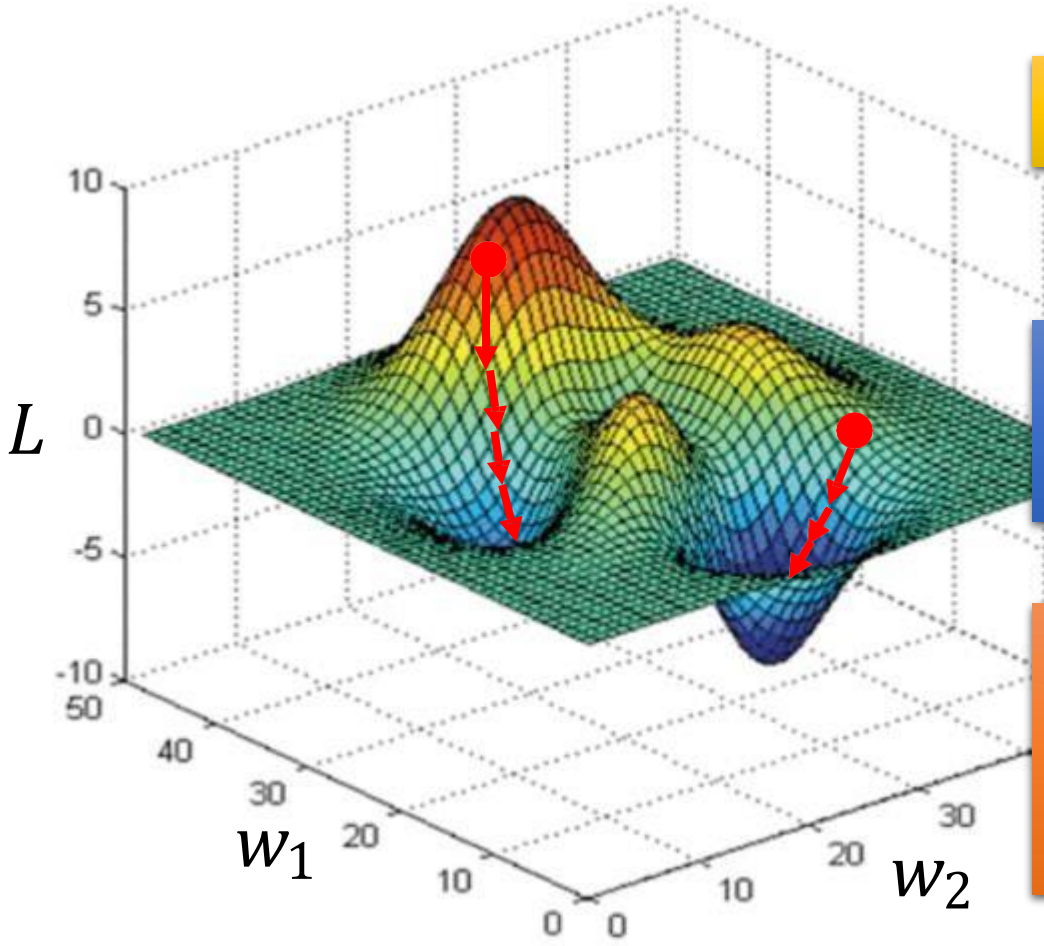
梯度下降

可以收斂到最小值.....



梯度下降法- 難點

- 梯度下降法不能保證全域最小值



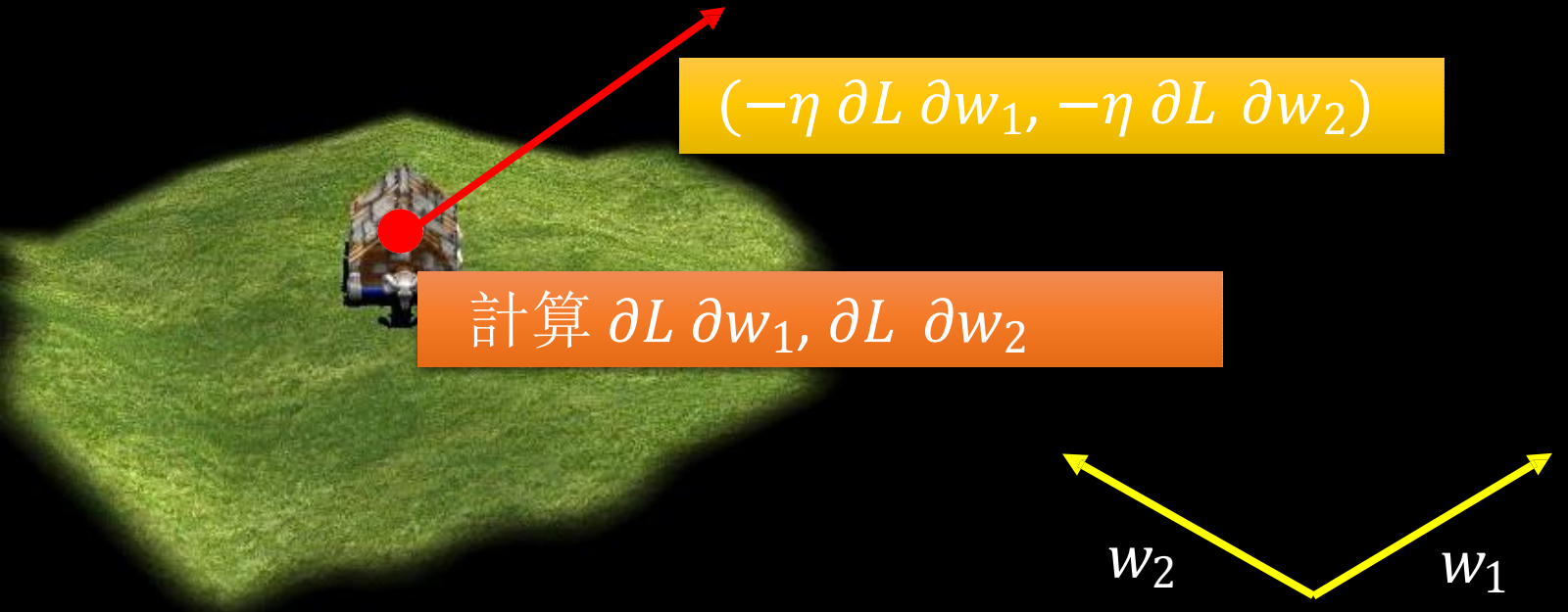
不同的初始點

到達不同的最小值，所以有不同的結果

有一些技巧可以說明你避免局部最小值，但不能保證.

當你在玩帝國時代...

你不可能看到整個地圖.

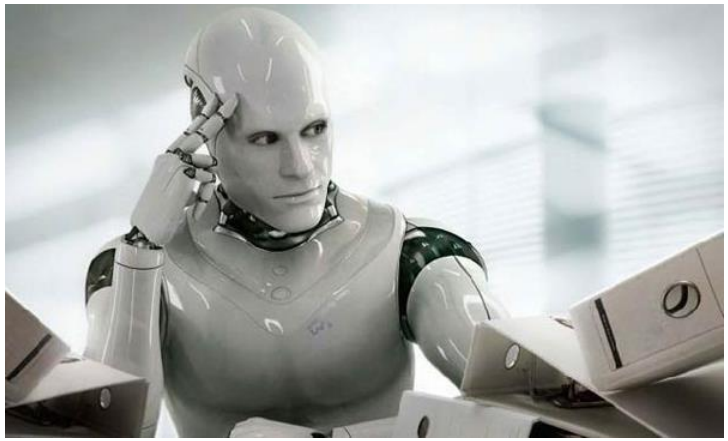


梯度下降

這就是機器在在深度學習中的“學習”

甚至 alpha go 也使用這種方法.

人們的想像.....



事實上 .....

我希望你不要太失望

反向傳播

- 反向傳播: 一種有效的計算方法 $\partial L \ \partial w$

libdnn
台大周伯威
同學開發

不用擔心 $\partial L \ \partial w$, 有專門的工具去解決

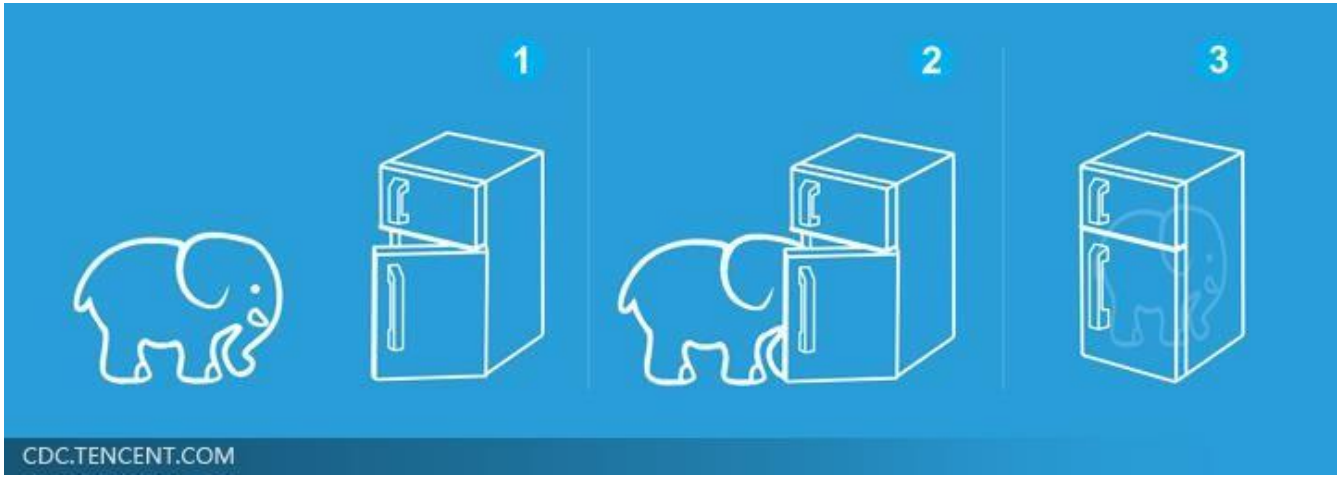
結束語

Step 1: 定義一組函數

Step 2: 模型和資料的擬合

Step 3: 選擇最優函數

深度學習是如此簡單 ……



大綱

深度學習的介紹

為什麼用深度網路?

深度學習簡單實戰

Deeper is Better?

Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1

不是更多的參數意味著更好的性能

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

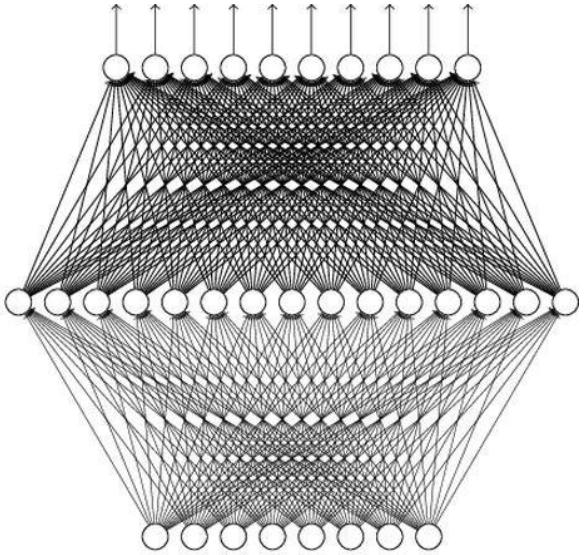
普遍性定理

任何連續函數 f

$$f : R^N \rightarrow R^M$$

可以通過一個隱藏層的網路實現

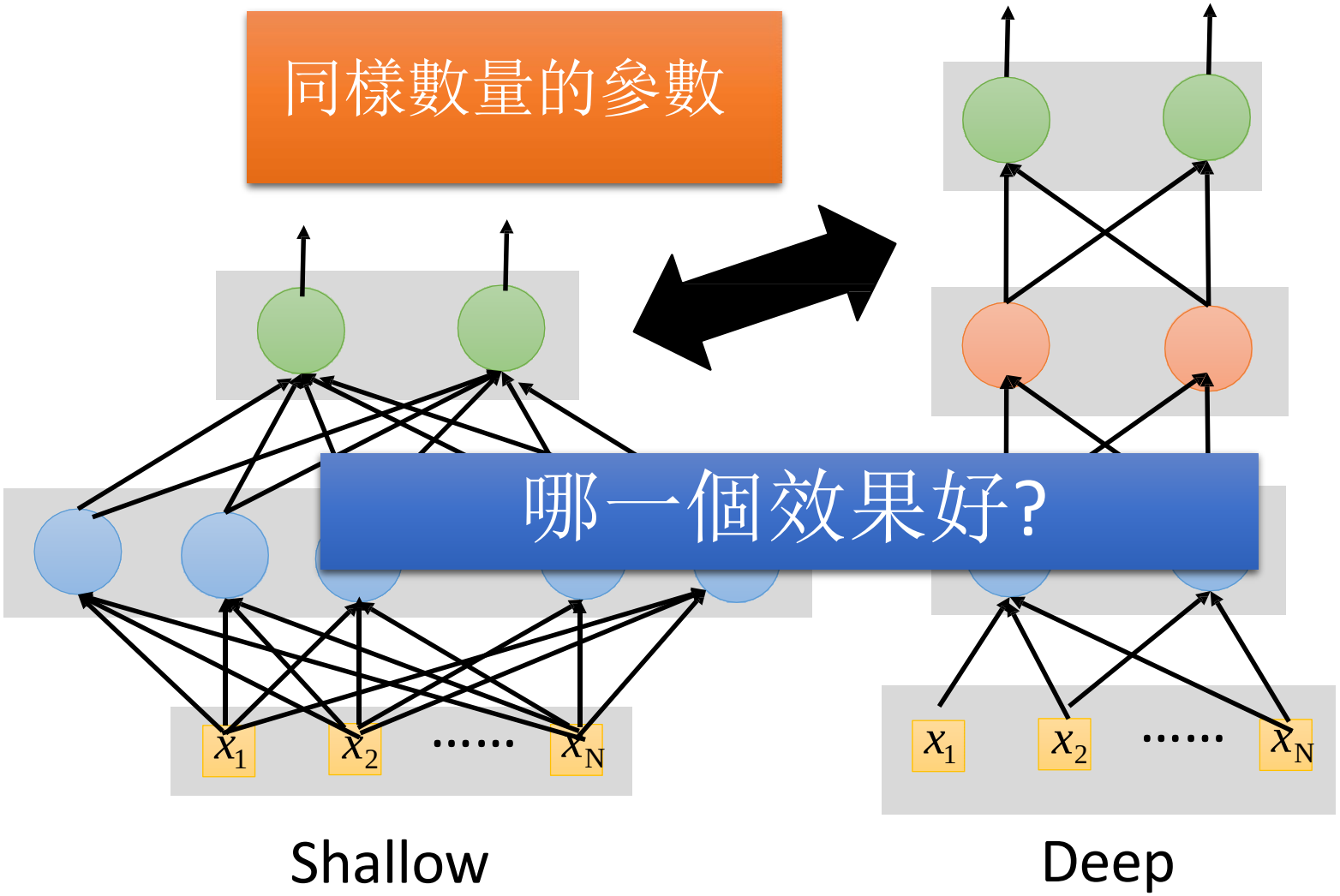
(給予足夠多的隱藏的神經元)



Reference for the reason:
http://neuralnetworksanddeplearning.com/chap4.html

為什麼是“深”神經網路而不是“胖”神經網路

Fat + Short v.s. Thin + Tall



Fat + Short v.s. Thin + Tall

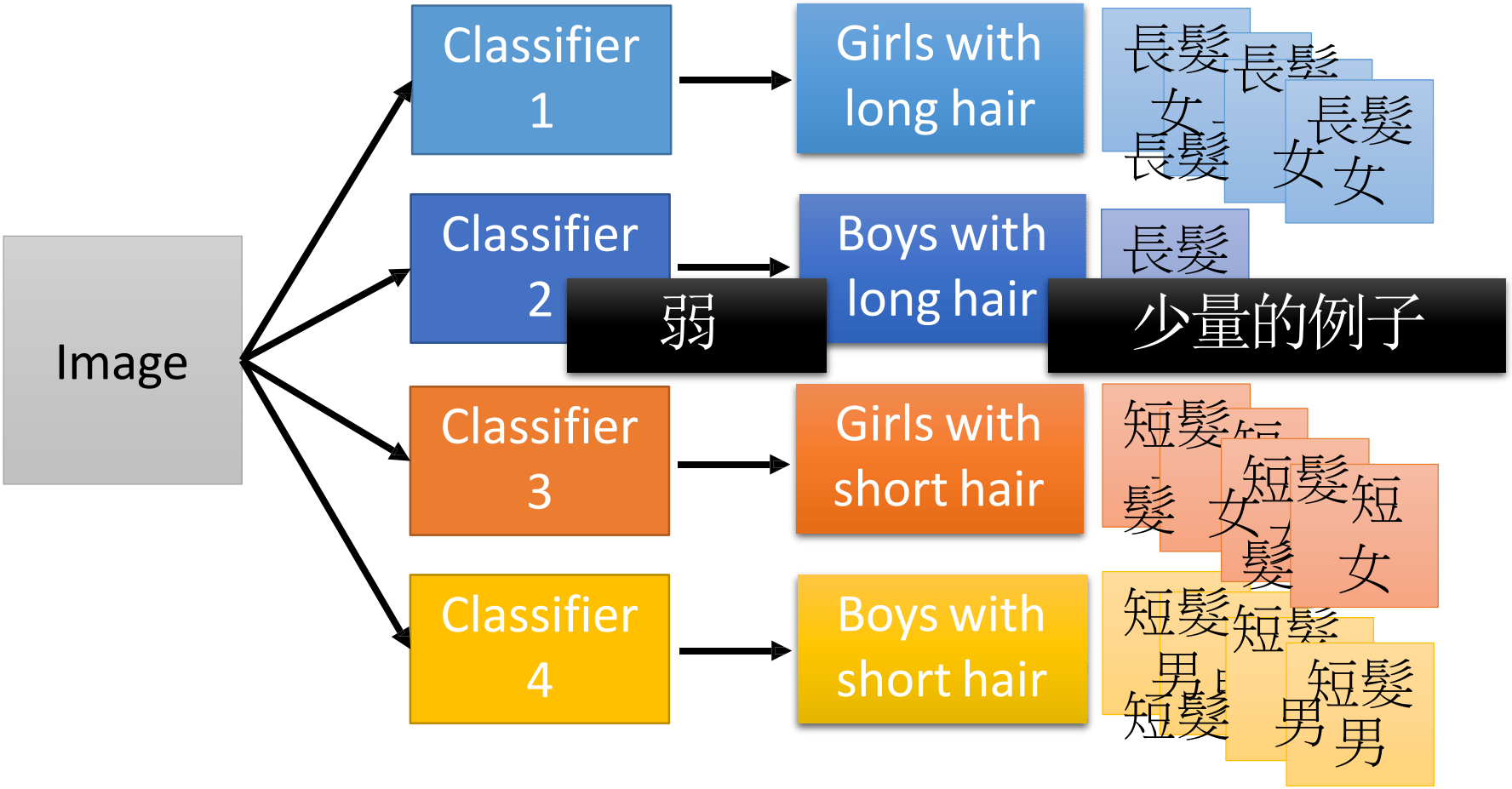
Layer X Size	Word Error Rate (%)	Layer X Size	Word Error Rate (%)
1 X 2k	24.2		
2 X 2k	20.4		
3 X 2k	18.4		
4 X 2k	17.8		
5 X 2k	17.2	1 X 3772	22.5
7 X 2k	17.1	1 X 4634	22.6
		1 X 16k	22.1

Why?

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

模組化

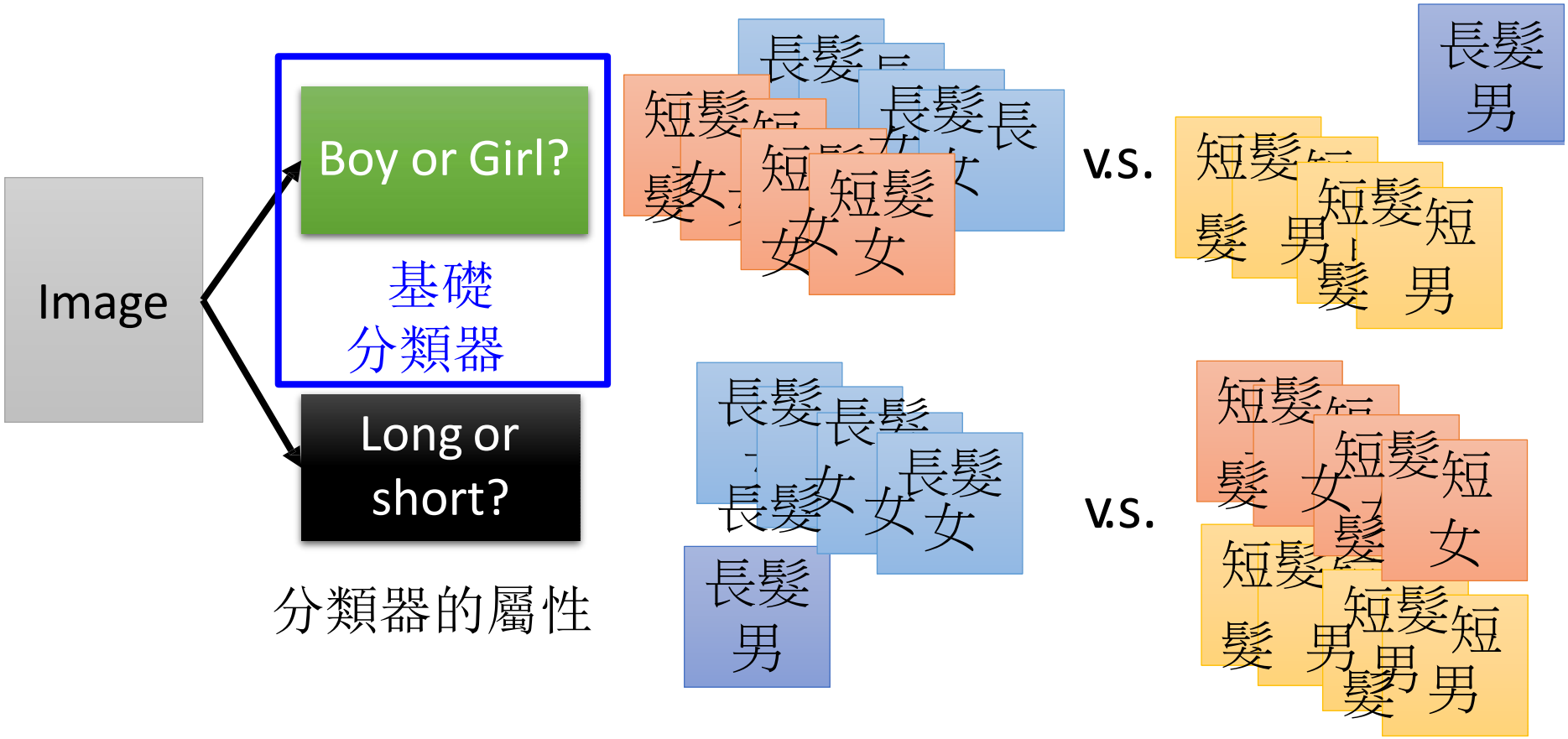
- Deep → Modularization



模組化

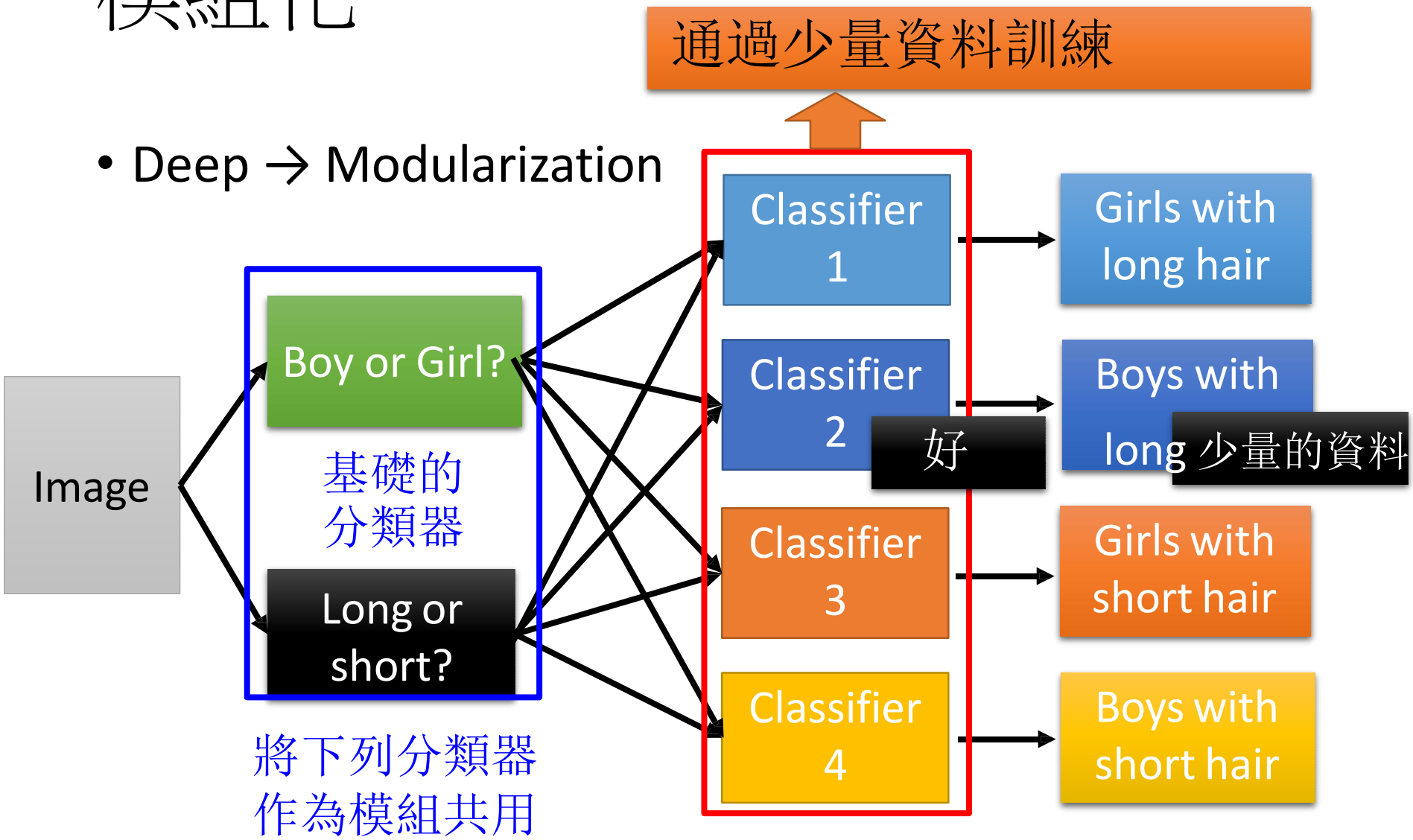
每個基礎分類器都可以有足夠的訓練實例

- Deep → Modularization



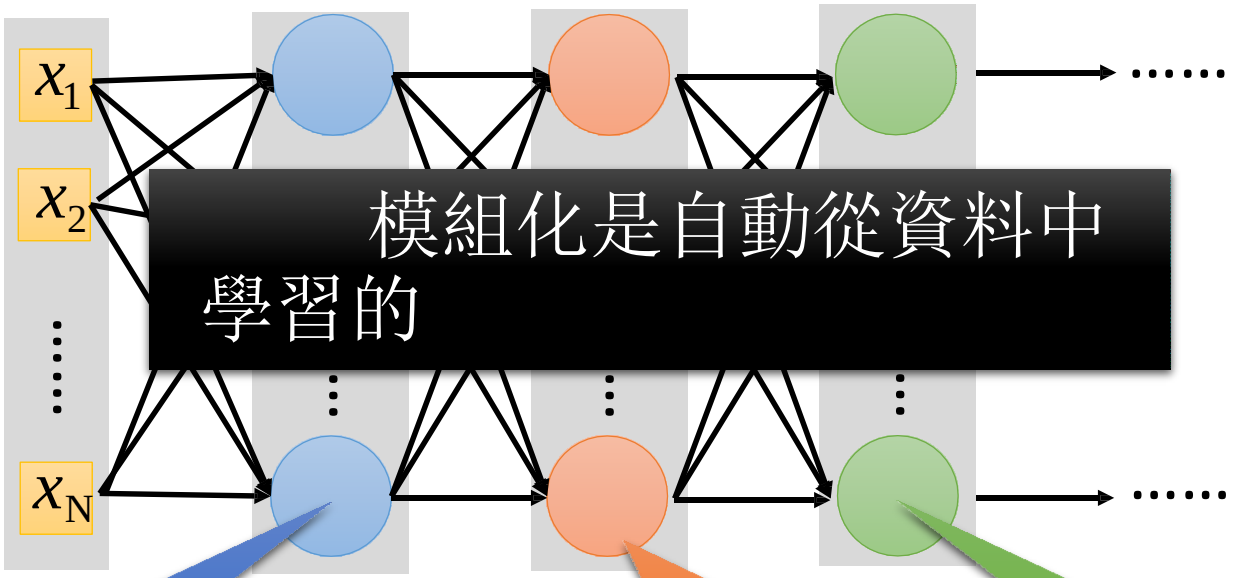
模組化

- Deep → Modularization



模組化

- Deep → Modularization → 更少的訓練數據?



最基礎的分類器

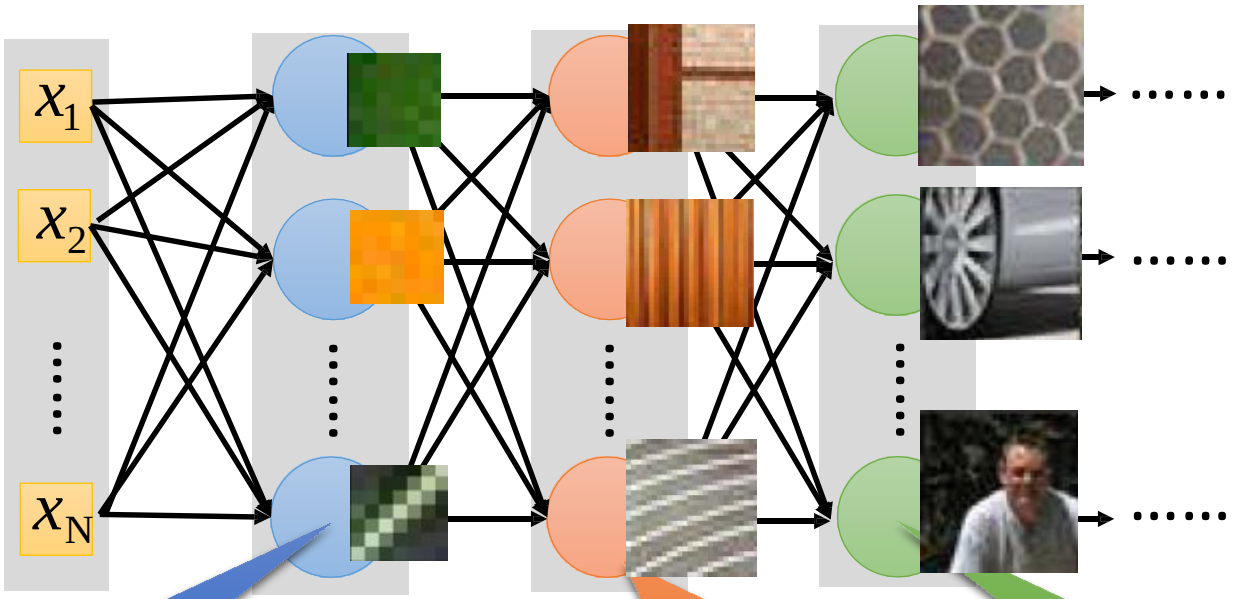
使用第一層作為模組構建分類器

使用第二層作為模組 .....

模組化

Reference: Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014* (pp. 818-833)

- Deep → Modularization



最基礎的分類器

使用第一層作為模組構建分類器

使用第二層作為模組 .....

大綱

深度學習的介紹

為什麼用深度網路?

深度學習簡單實戰

Keras

If you want to learn theano:

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/Theano%20DNN.ecm.mp4/index.html

http://speech.ee.ntu.edu.tw/~tlkagk/courses/MLDS_2015_2/Lecture/RNN%20training%20(v6).ecm.mp4/index.html

or theano

非常靈活

需要一些努力去學習

TensorFlow 或 Theano 的介面

keras

易於學習和使用

(仍然有一些靈活性)

Keras

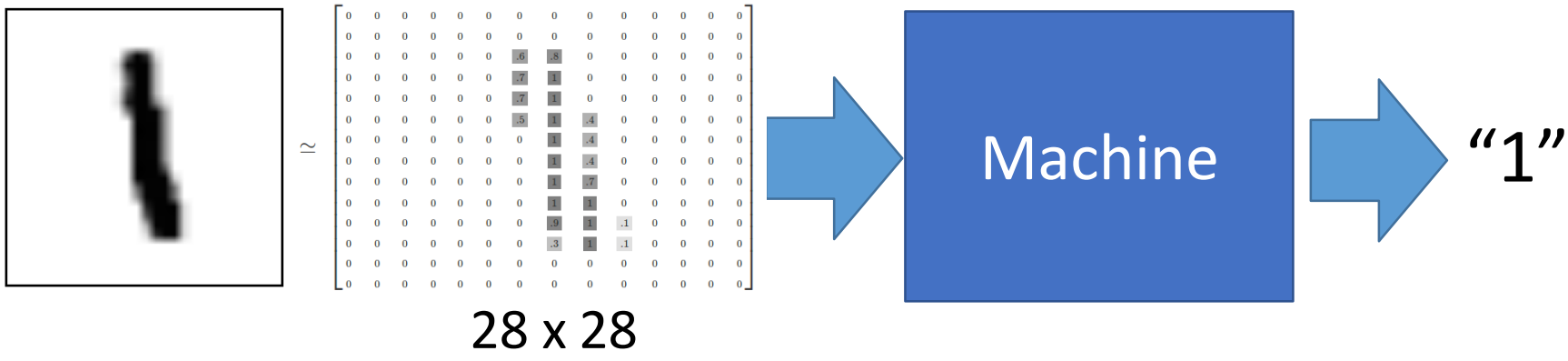
- François Chollet 是 Keras 的創建者.
  - 他目前在穀歌擔任深度學習工程師和研究員
- Keras means *horn* in Greek
- 文檔: http://keras.io/
- 示例: https://github.com/fchollet/keras/tree/master/exa mples

使用 Keras 心


Deep Learning研究生
朋友覺得我在
我媽覺得我在
大眾覺得我在
指導教授覺得我在
我以為我在
事實上我在

示例應用程式

- 手寫體數位識別



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

“Hello world” for deep learning

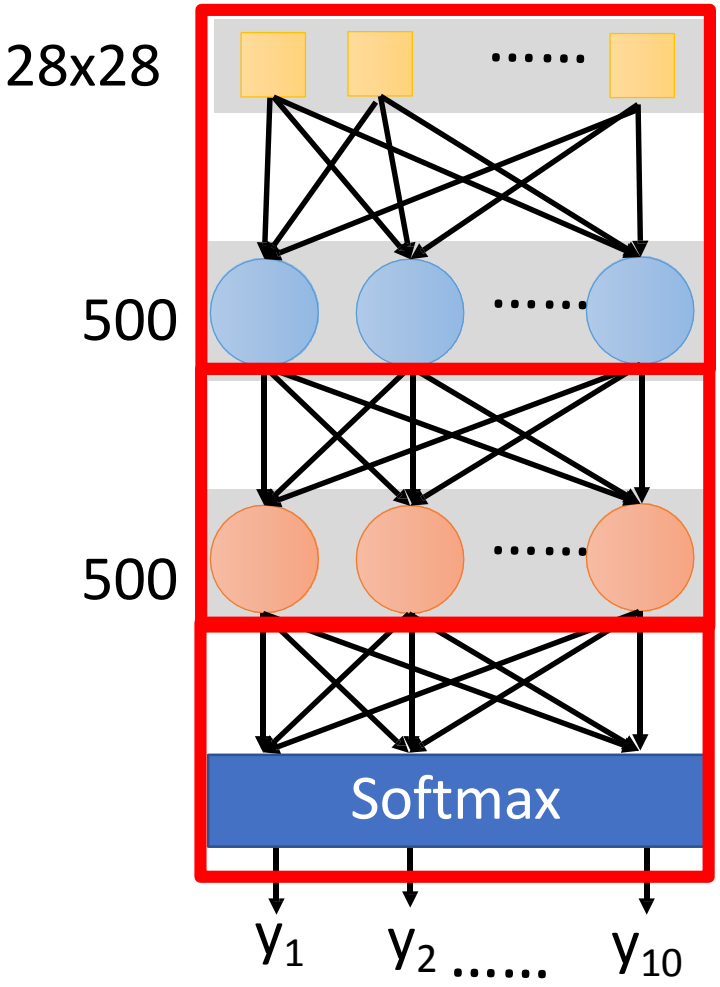
Keras provides data sets loading function: http://keras.io/datasets/

Keras

Step 1: define a set of function

Step 2: goodness of function

Step 3: pick the best function



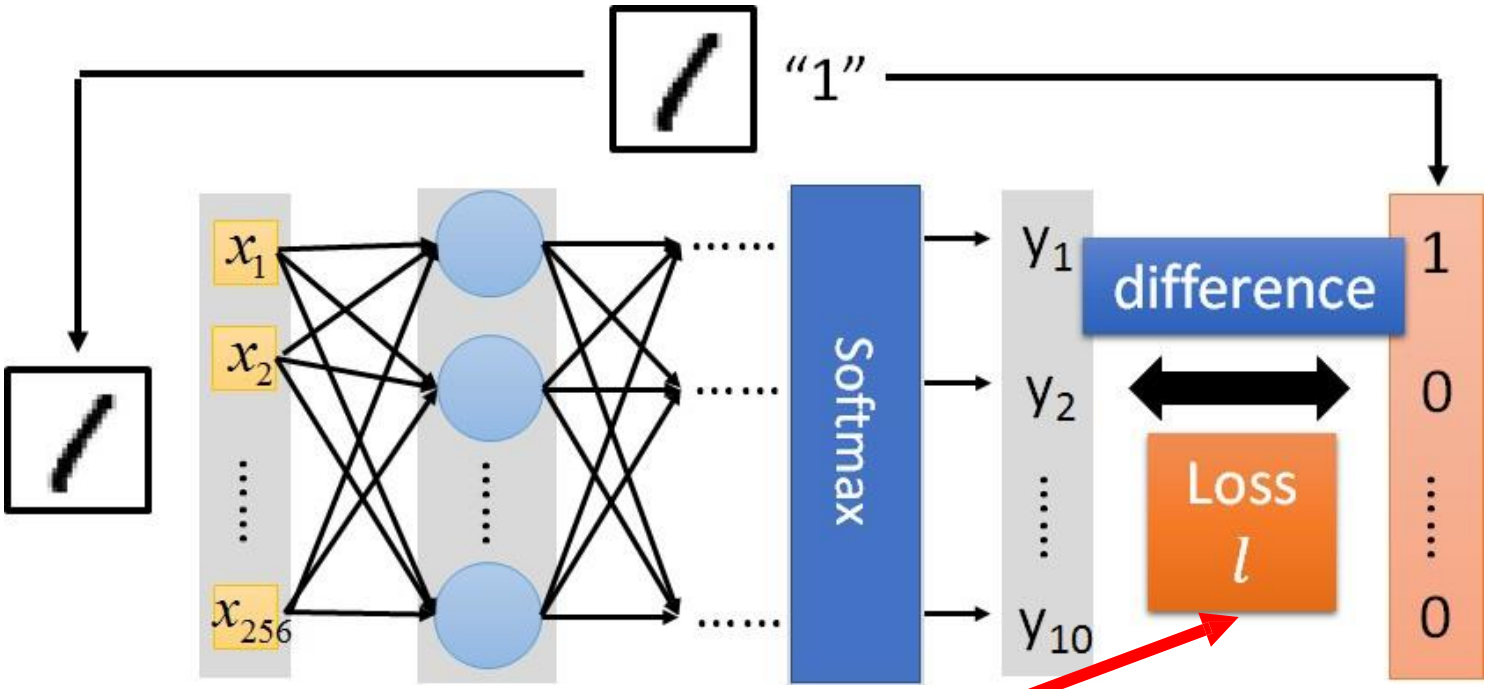

```
model = Sequential()
```

```
model.add( Dense( input_dim=28*28,
                  output_dim=500 ))
model.add( Activation('sigmoid') )
```

```
model.add( Dense( output_dim=500 ) )
model.add( Activation('sigmoid') )
```

```
model.add( Dense(output_dim=10 ) )
model.add( Activation('softmax') )
```

Keras




```
model.compile(loss='mse',
              optimizer=SGD(lr=0.1),
              metrics=['accuracy'])
```

Keras

Step 3.1: 配置

```
model.compile(loss='mse',
              optimizer=SGD(lr=0.1),
              metrics=['accuracy'])
```

$$w \leftarrow w - \eta \partial L \; \partial w$$

0.1

Step 3.2: 找到最優的參數

```
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```

Training data (Images)

Labels (digits)

Next lecture

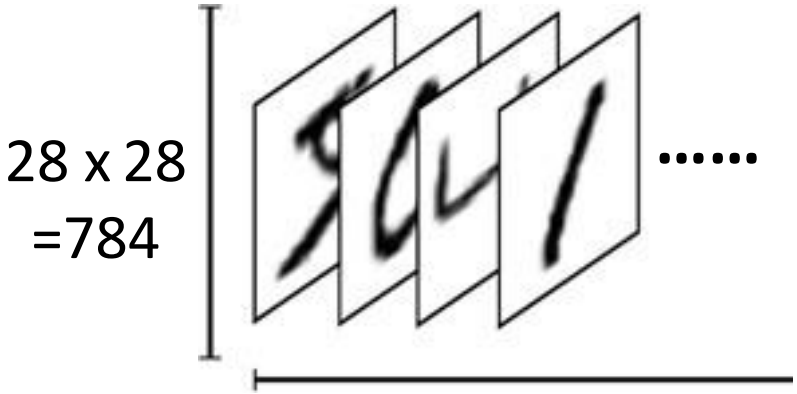
Keras

Step 3.2: 找到最優的參數

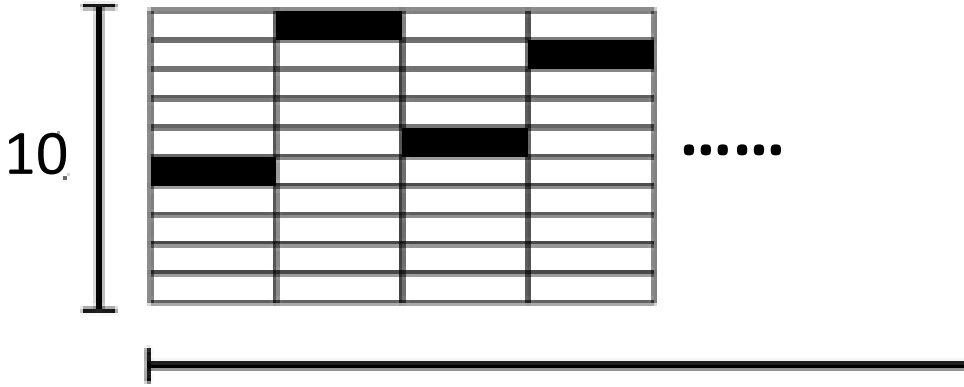
```
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```

numpy array

numpy array



Number of training examples



Number of training examples

https://www.tensorflow.org/versions/r0.8/tutorials/mnist/beginners/index.html

Keras

保存和載入模型

http://keras.io/getting-started/faq/#how-can-i-save-a-keras-model

如何使用神經網路 (測試):

case 1:

```
score = model.evaluate(x_test,y_test)
print('Total loss on Testing Set:', score[0])
print('Accuracy of Testing Set:', score[1])
```

case 2:

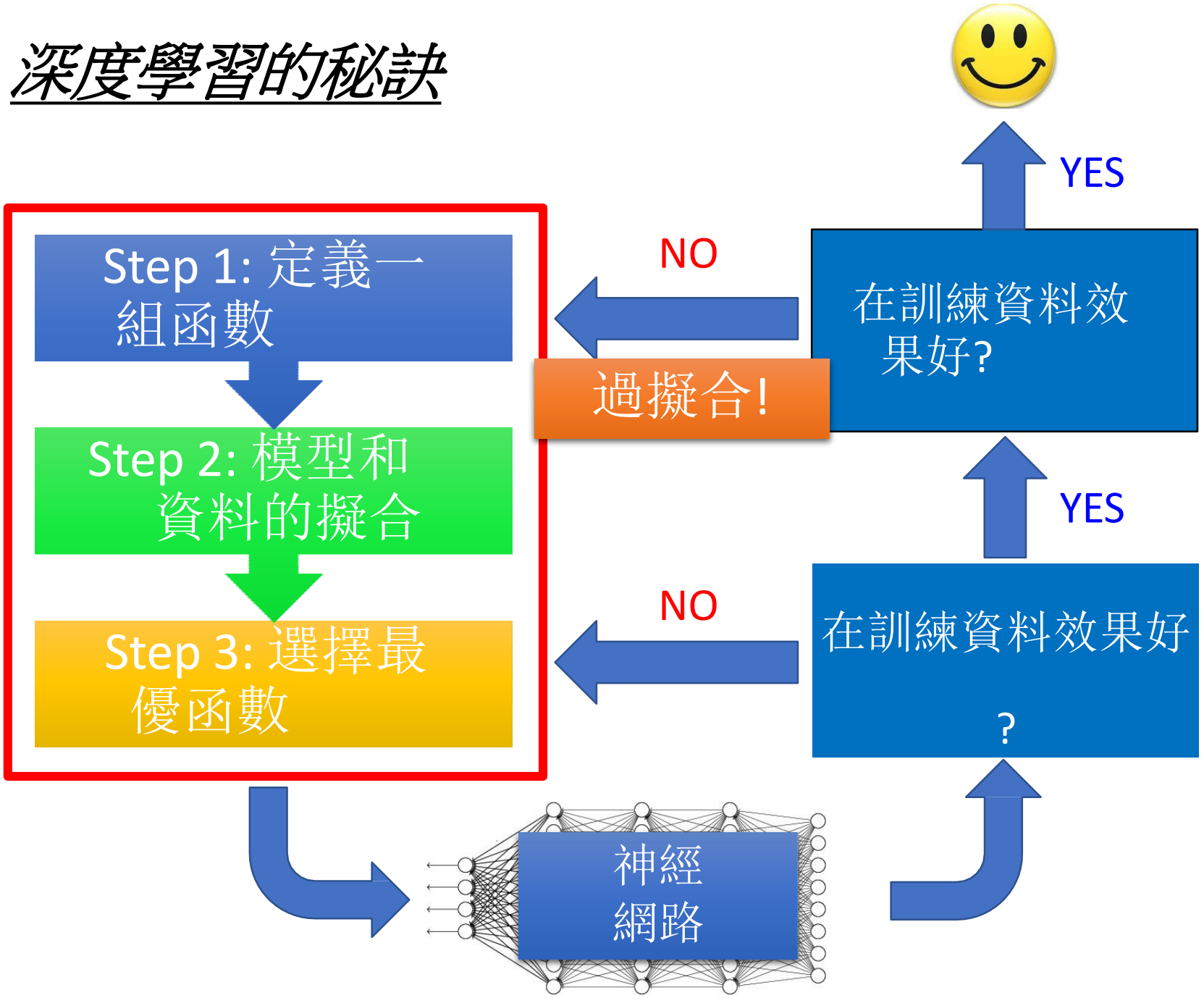
```
result = model.predict(x_test)
```

Keras

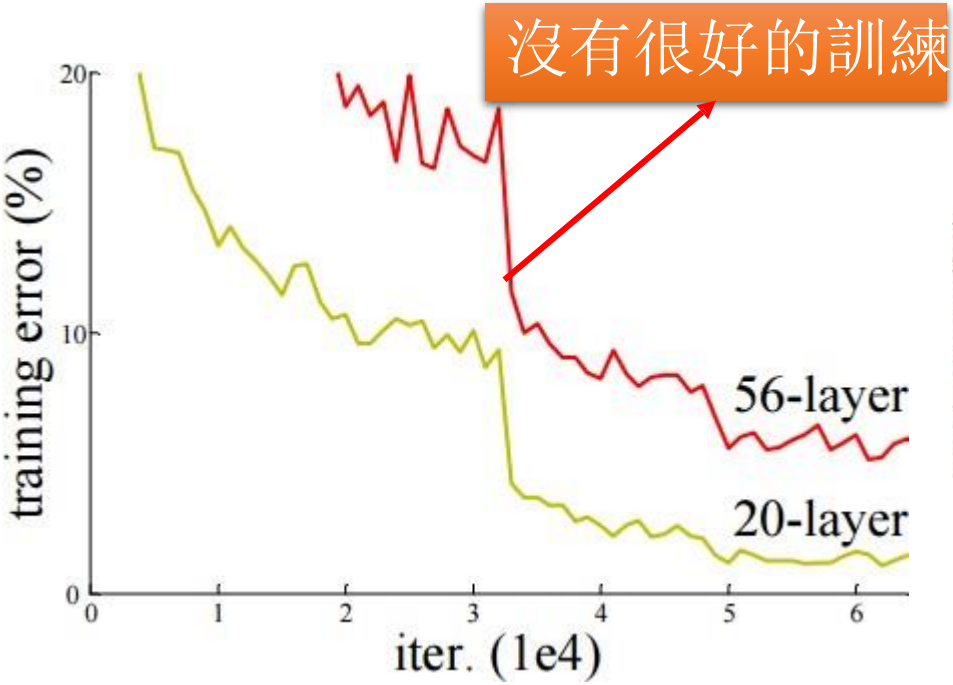
- 使用 GPU 加速
  - Way 1
    - THEANO_FLAGS=device=gpu0 python YourCode.py
  - Way 2 (in your code)
    - import os
    - os.environ["THEANO_FLAGS"] = "device=gpu0"

Lecture II:
訓練 DNN 的技巧

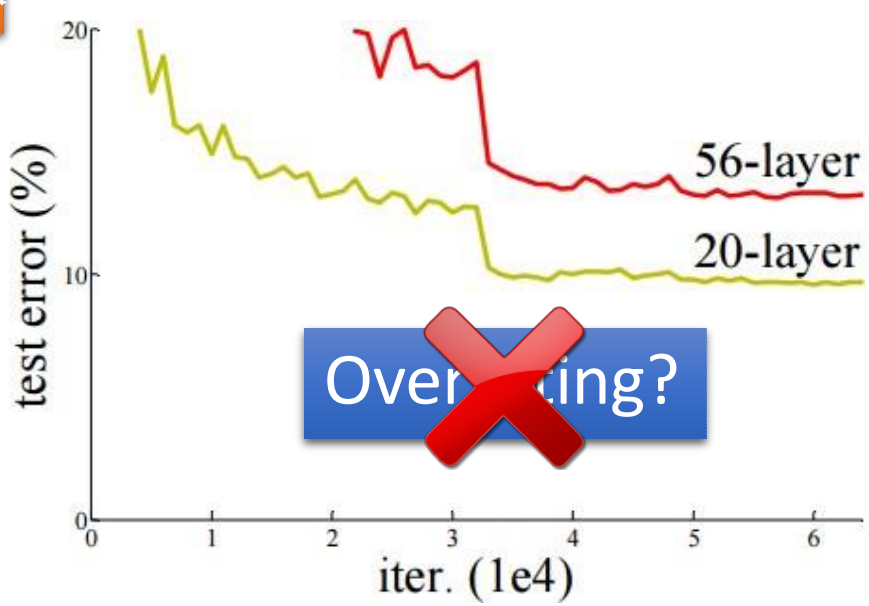
深度學習的秘訣


Step 1: 定義一組函數
Step 2: 模型和資料的擬合
Step 3: 選擇最優函數
神經網路
在訓練資料效果好?
NO
過擬合!
在訓練資料效果好?
NO
YES
YES

不要總是怪過擬合



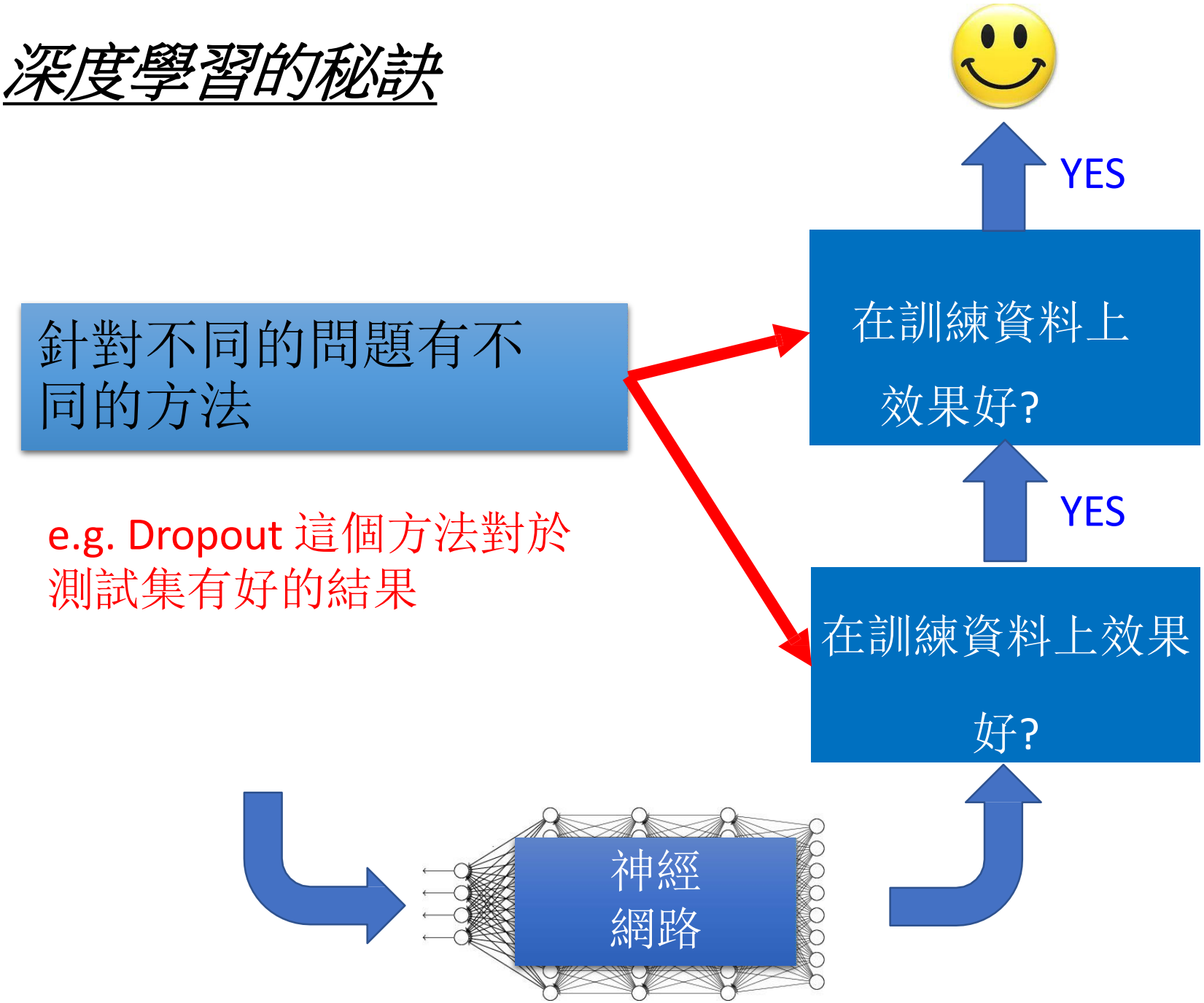
訓練數據



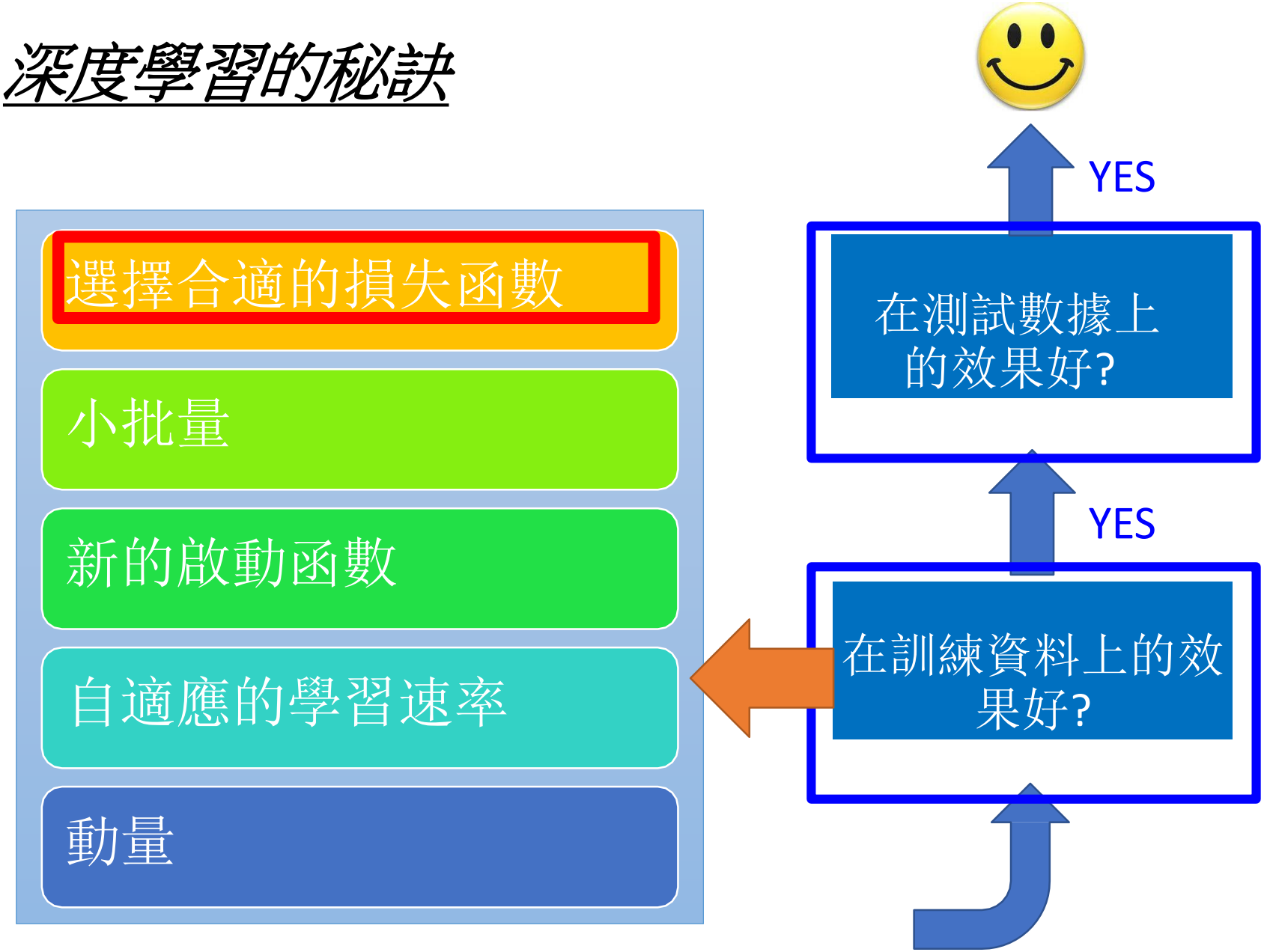
測試資料

Deep Residual Learning for Image Recognition
http://arxiv.org/abs/1512.03385

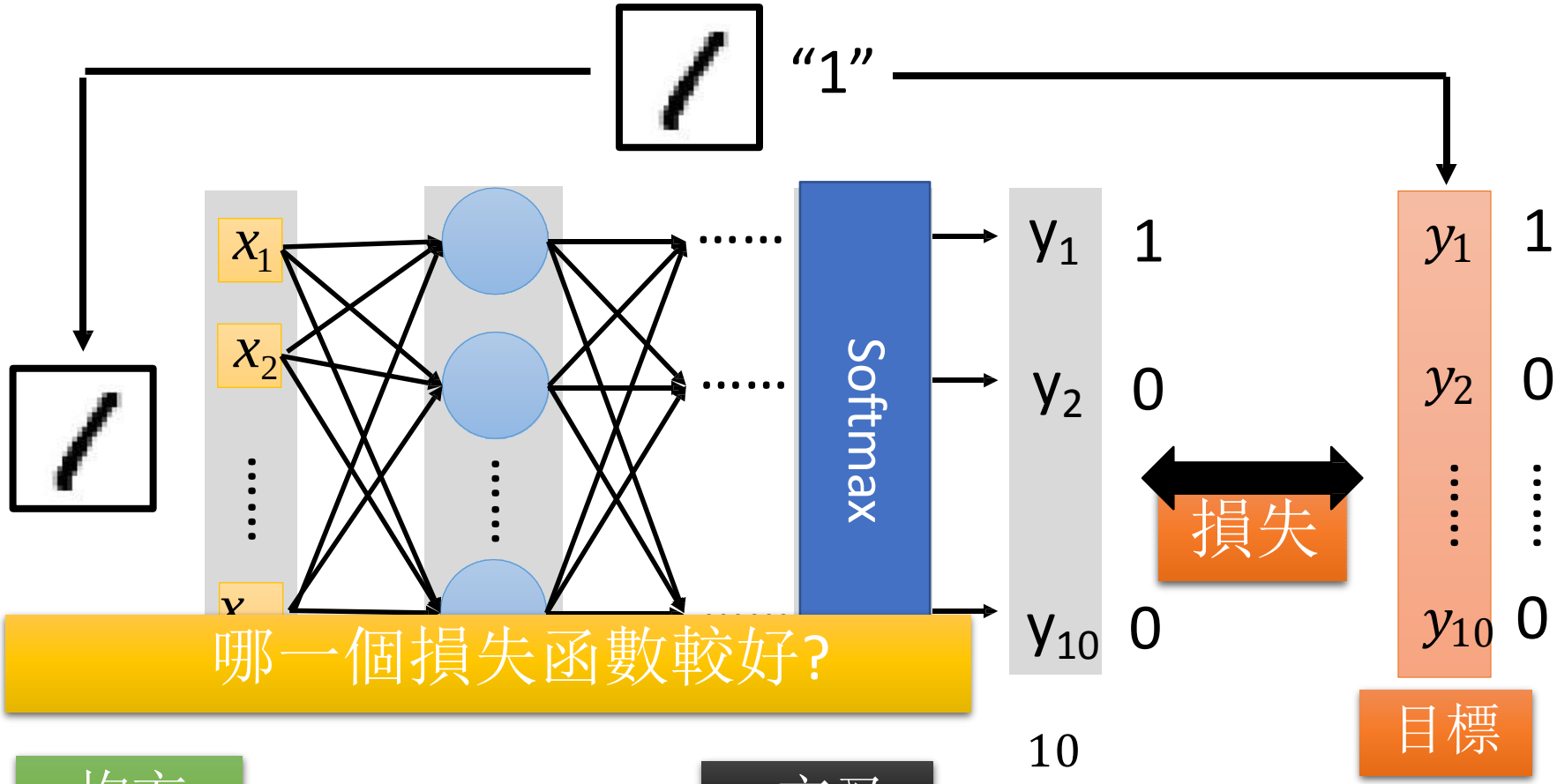
深度學習的秘訣


針對不同的問題有不同的方法
e.g. Dropout 這個方法對於測試集有好的結果
在訓練資料上效果好?
YES
在訓練資料上效果好?
YES
神經網路

深度學習的秘訣


選擇合適的損失函數
小批量
新的啟動函數
自適應的學習速率
動量
在訓練資料上的效果好?
YES
在測試數據上的效果好?
YES

選擇合適的損失函數


"1"
x_1
x_2
x_N
Softmax
y_1 1
y_2 0
y_{10} 0
損失
y_1 1
y_2 0
y_{10} 0
目標
哪一個損失函數較好?
均方誤差
$\sum_0$
交叉熵
$-\sum_{i=1}^{10} y_i ln y_i$
=0

Let’s try it

平方誤差

```
model.compile(loss='mse',
              optimizer=SGD(lr=0.1),
              metrics=['accuracy'])
```

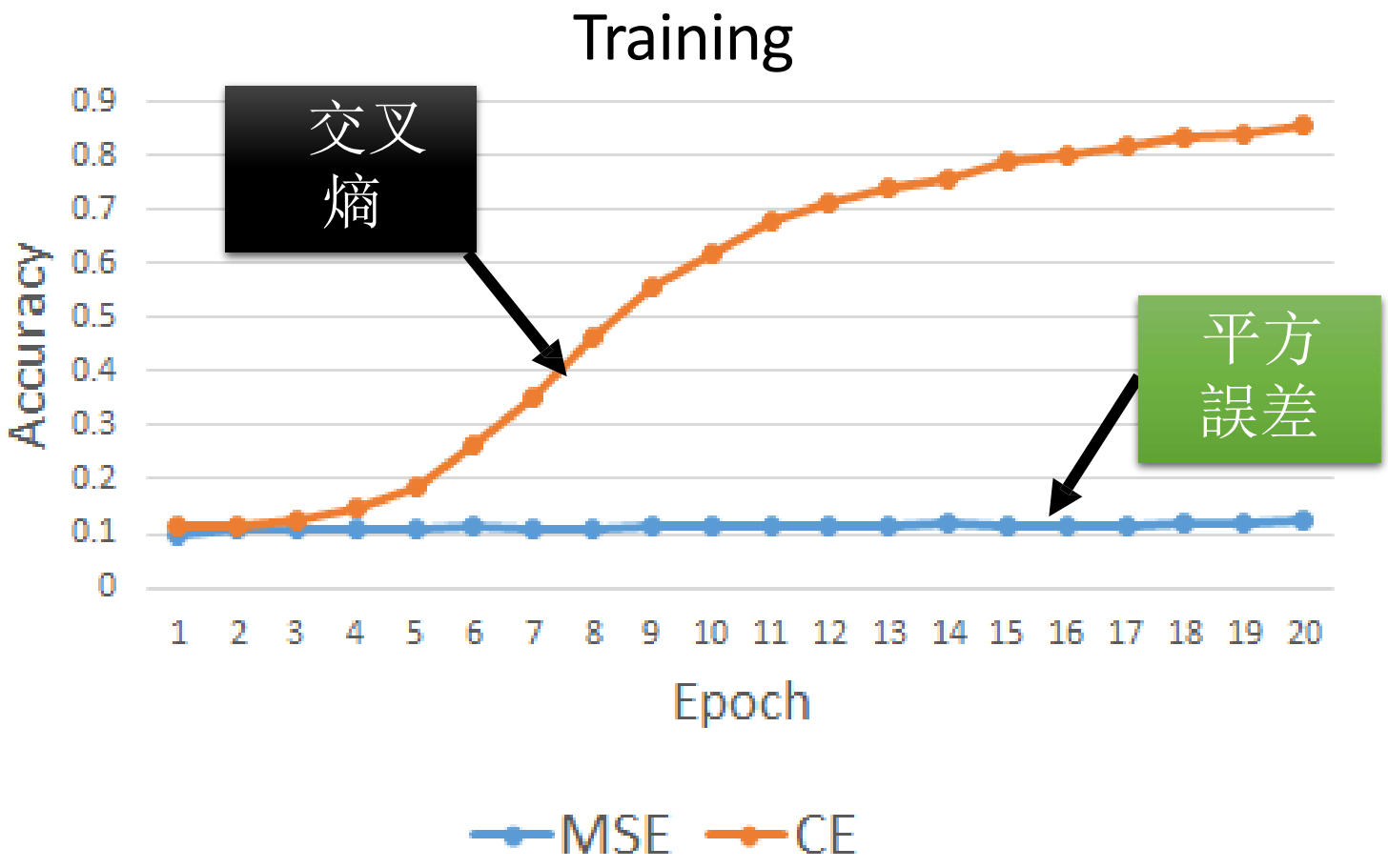
交叉熵

```
model.compile(loss='categorical_crossentropy',
              optimizer=SGD(lr=0.1),
              metrics=['accuracy'])
```

讓我們試一試

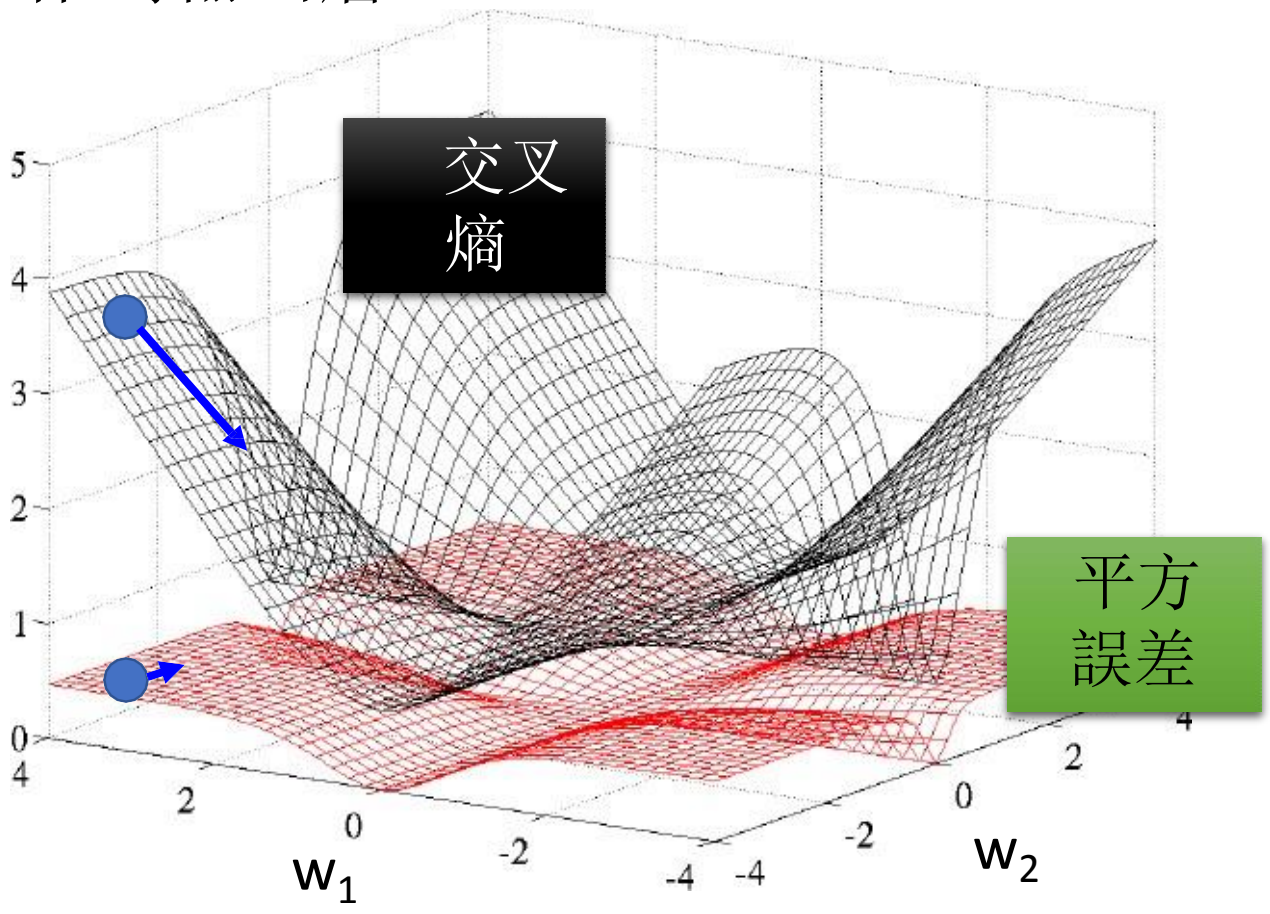
測試:

	準確率
平方誤差	0.11
交叉熵	0.84



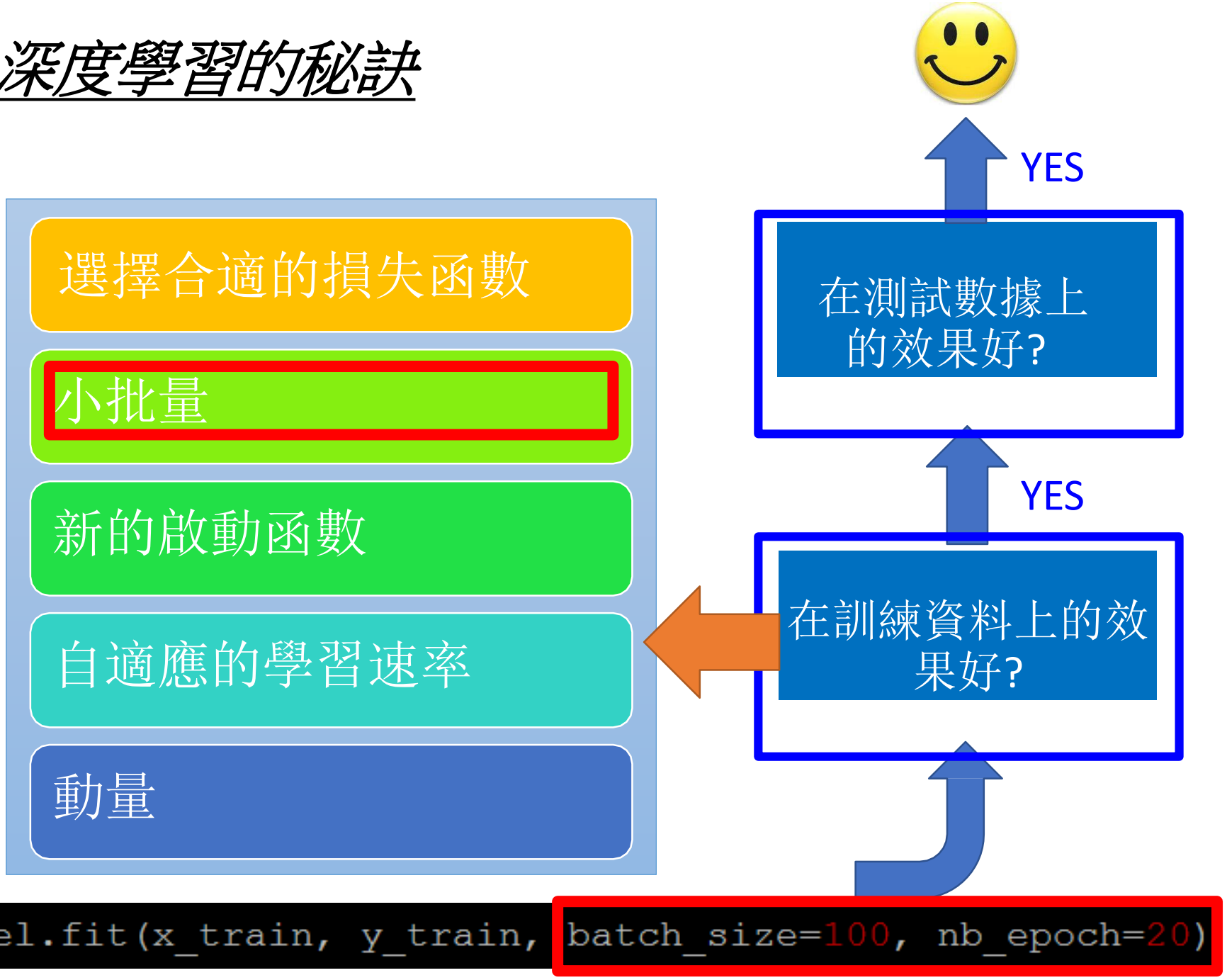
選擇合適的損失函數

當使用softmax 作為輸出層，
選擇交叉熵



http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf

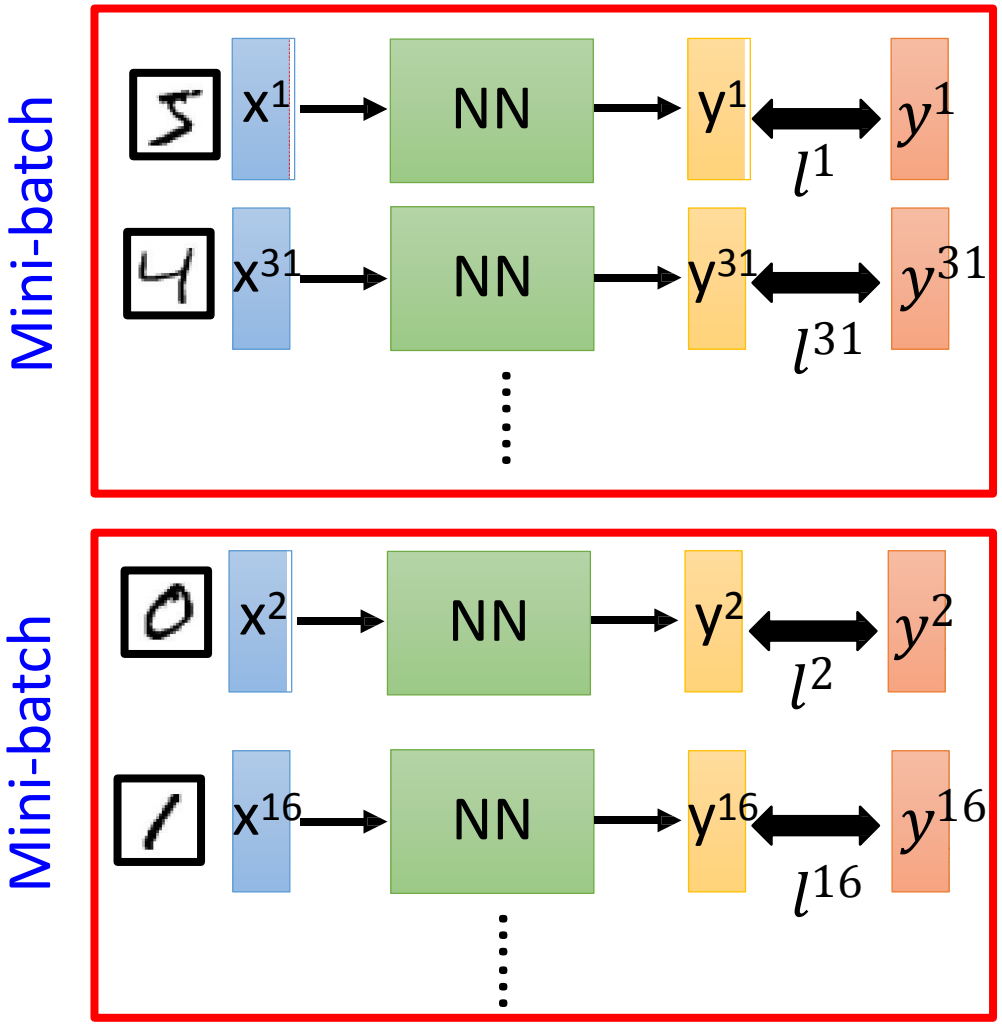
深度學習的秘訣


選擇合適的損失函數
小批量
新的啟動函數
自適應的學習速率
動量
在訓練資料上的效果好?
YES
在測試數據上的效果好?
YES
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)

我們並沒有真正講損失降到最低!

Mini-batch

- 隨機初始化參數



- Pick the 1st batch

  $L' = l^1 + l^{31} + \cdots$

  更新一次參數

- Pick the 2nd batch

  $L'' = l^2 + l^{16} + \cdots$

  更新一次參數

  ⋮

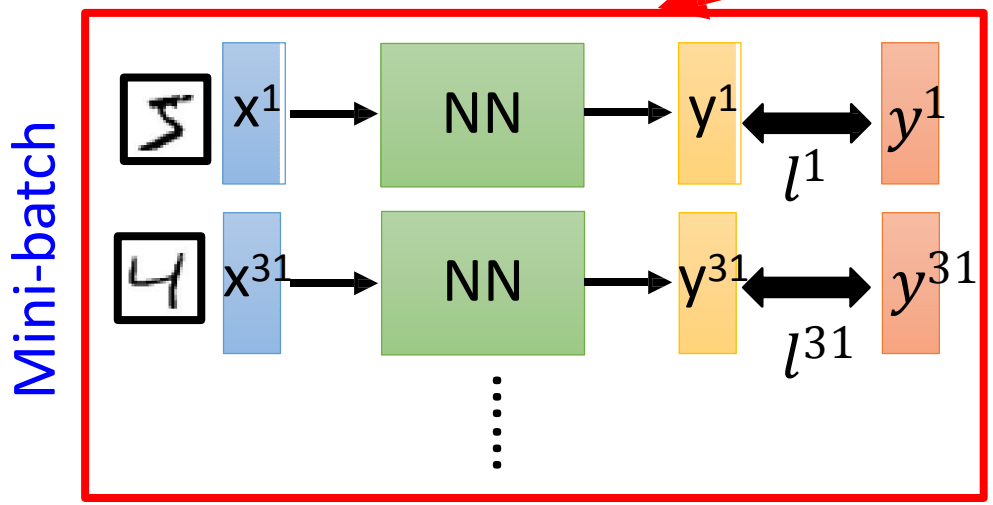
- 直到所有的小批量都被選中

one epoch

重複上述過程

Mini-batch

```
model.fit(x_train, y_train, batch_size=100, nb_epoch=20)
```



100個數據在 mini-batch 中

- Pick the 1st batch

  $L' = l^1 + l^{31} + \cdots$

  更新一次參數

- Pick the 2nd batch

  $L'' = l^2 + l^{16} + \cdots$

  更新一次參數

  ⋮

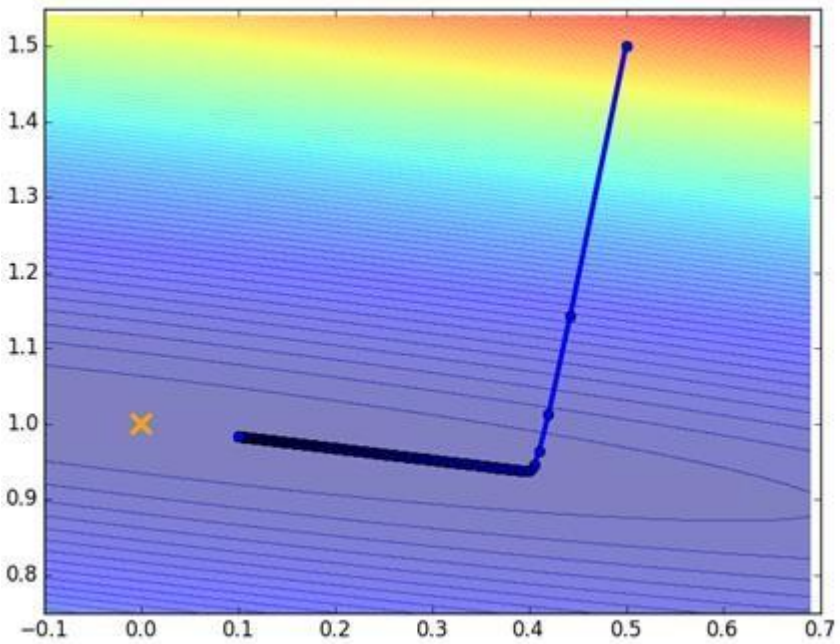
- 直到所有的小批量都被選中

one epoch

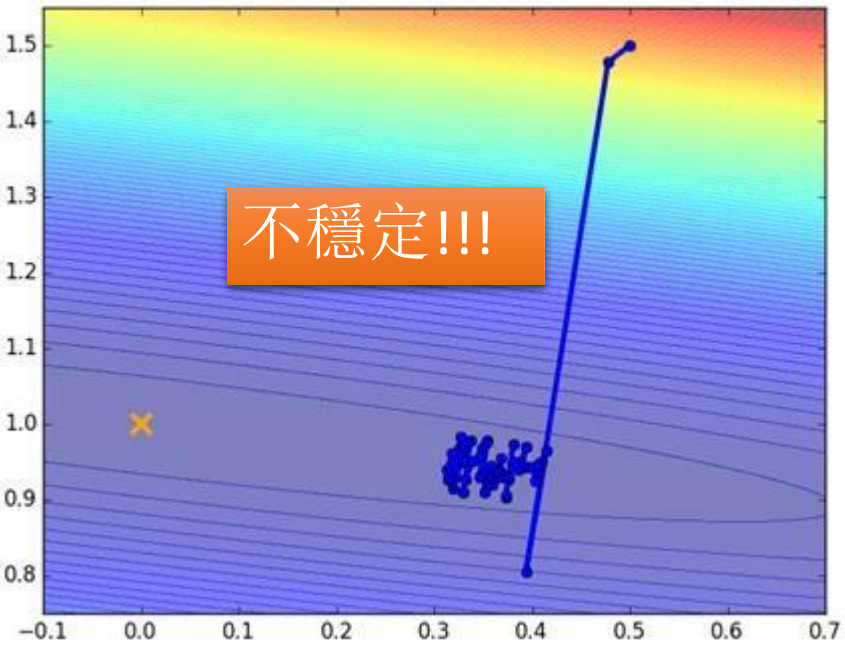
重複20次

Mini-batch

Original Gradient Descent



With Mini-batch



顏色代表全部損失

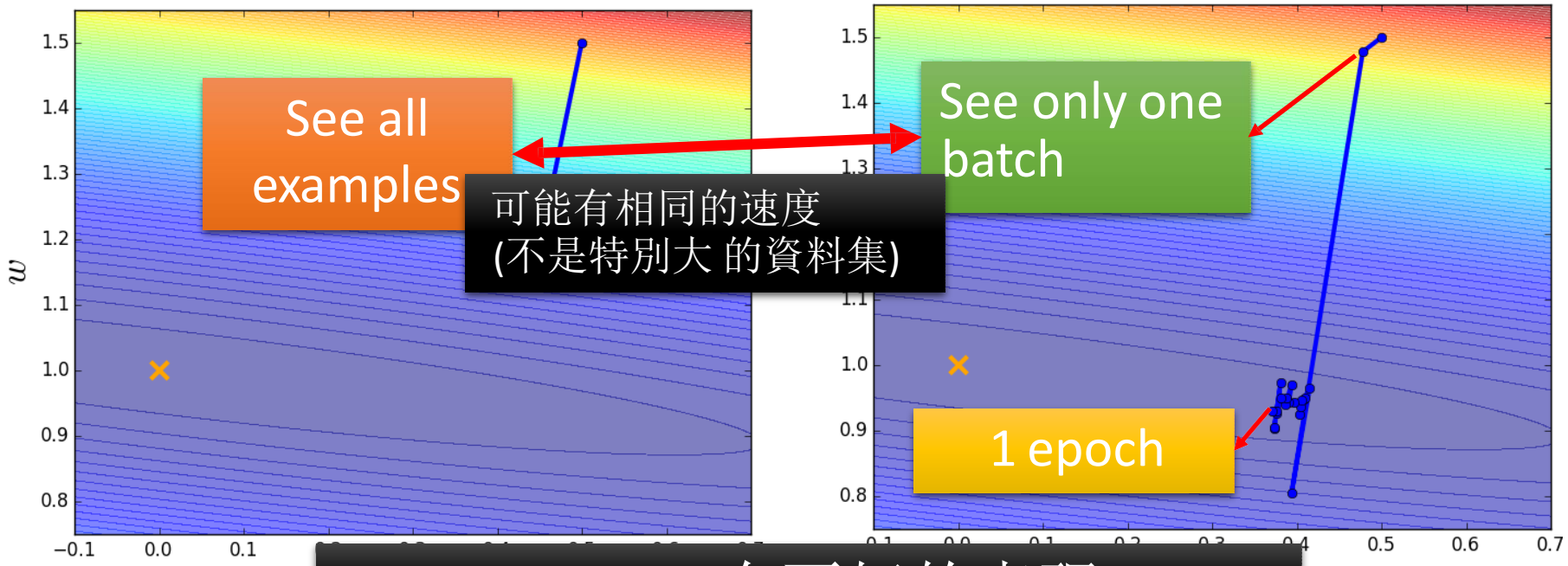
Mini-batch 更快

Original Gradient Descent

Update after seeing all examples

With Mini-batch

If there are 20 batches, update 20 times in one epoch.

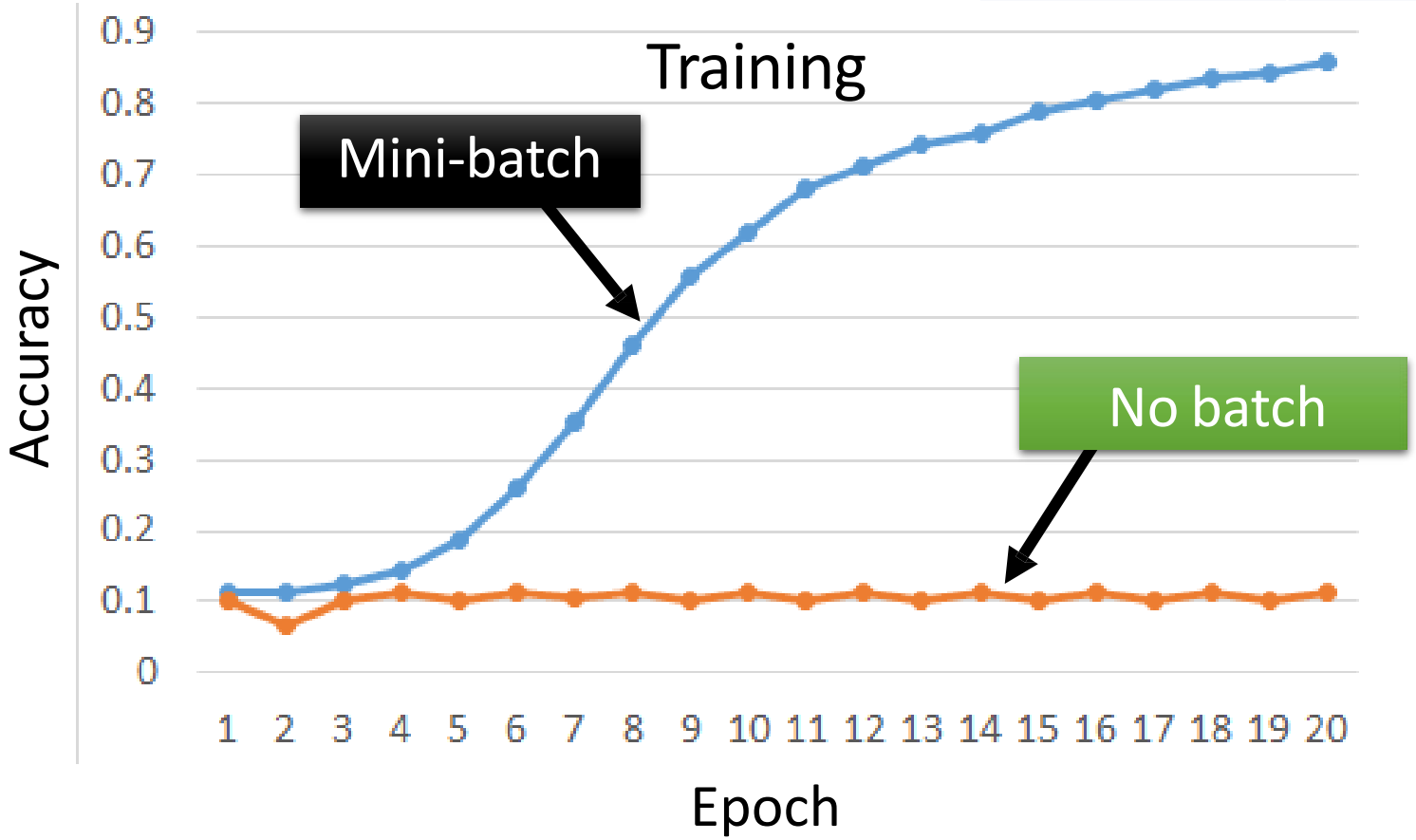


Mini-batch 有更好的表現!

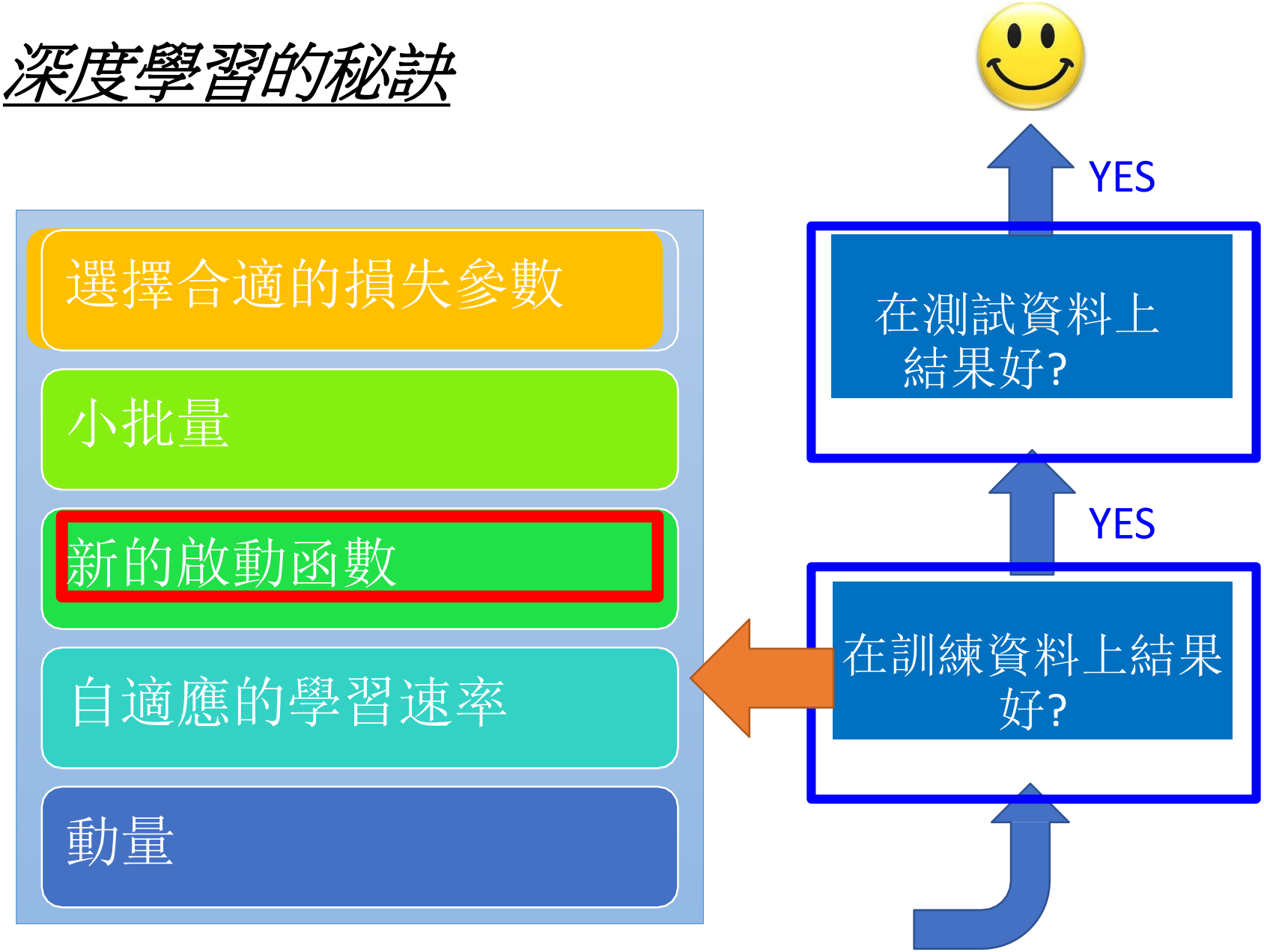
Mini-batch 更好!

Testing:

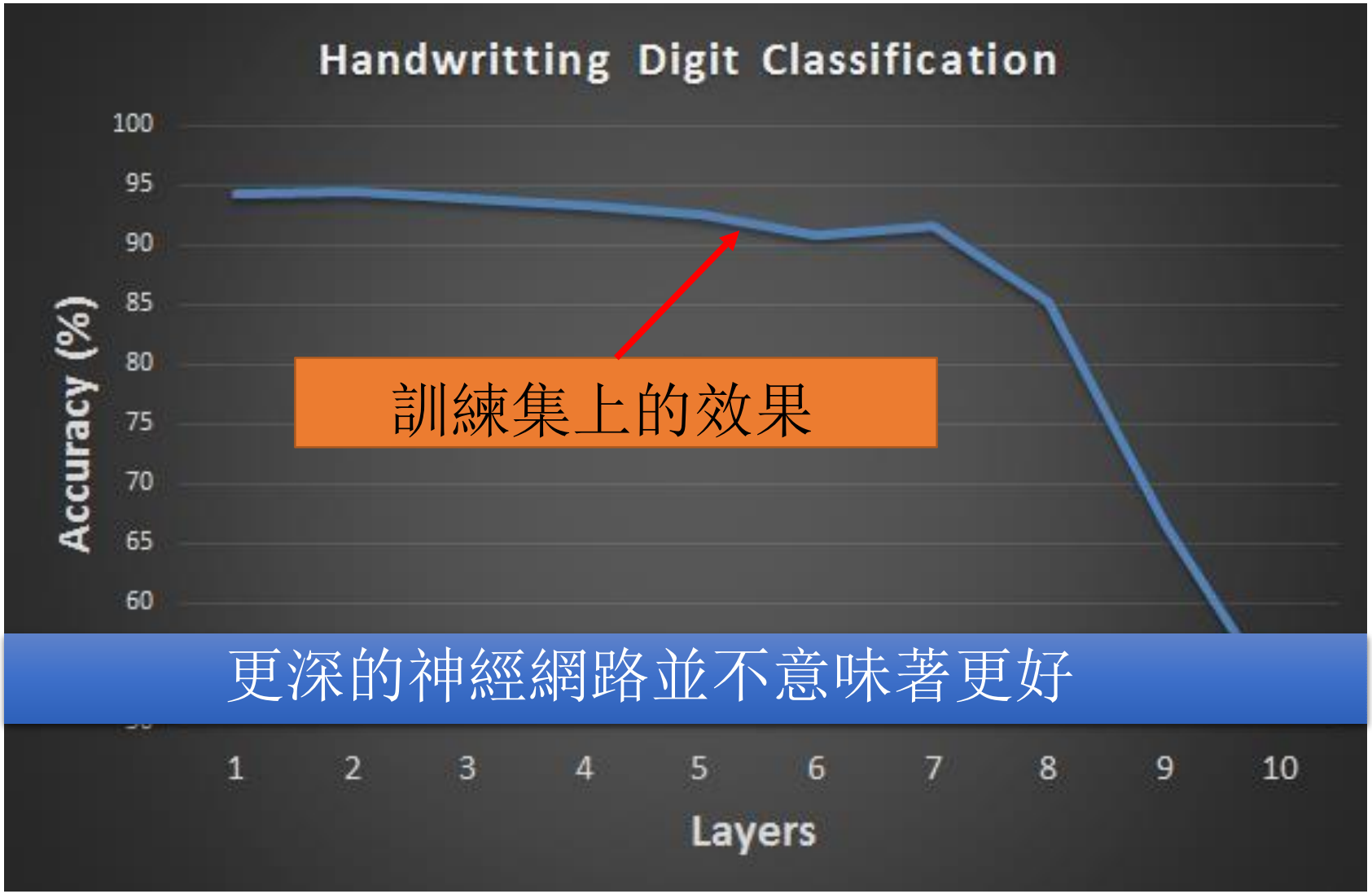
	Accuracy
Mini-batch	0.84
No batch	0.12



深度學習的秘訣


選擇合適的損失參數
小批量
新的啟動函數
自適應的學習速率
動量
在訓練資料上結果好?
YES
在測試資料上結果好?
YES

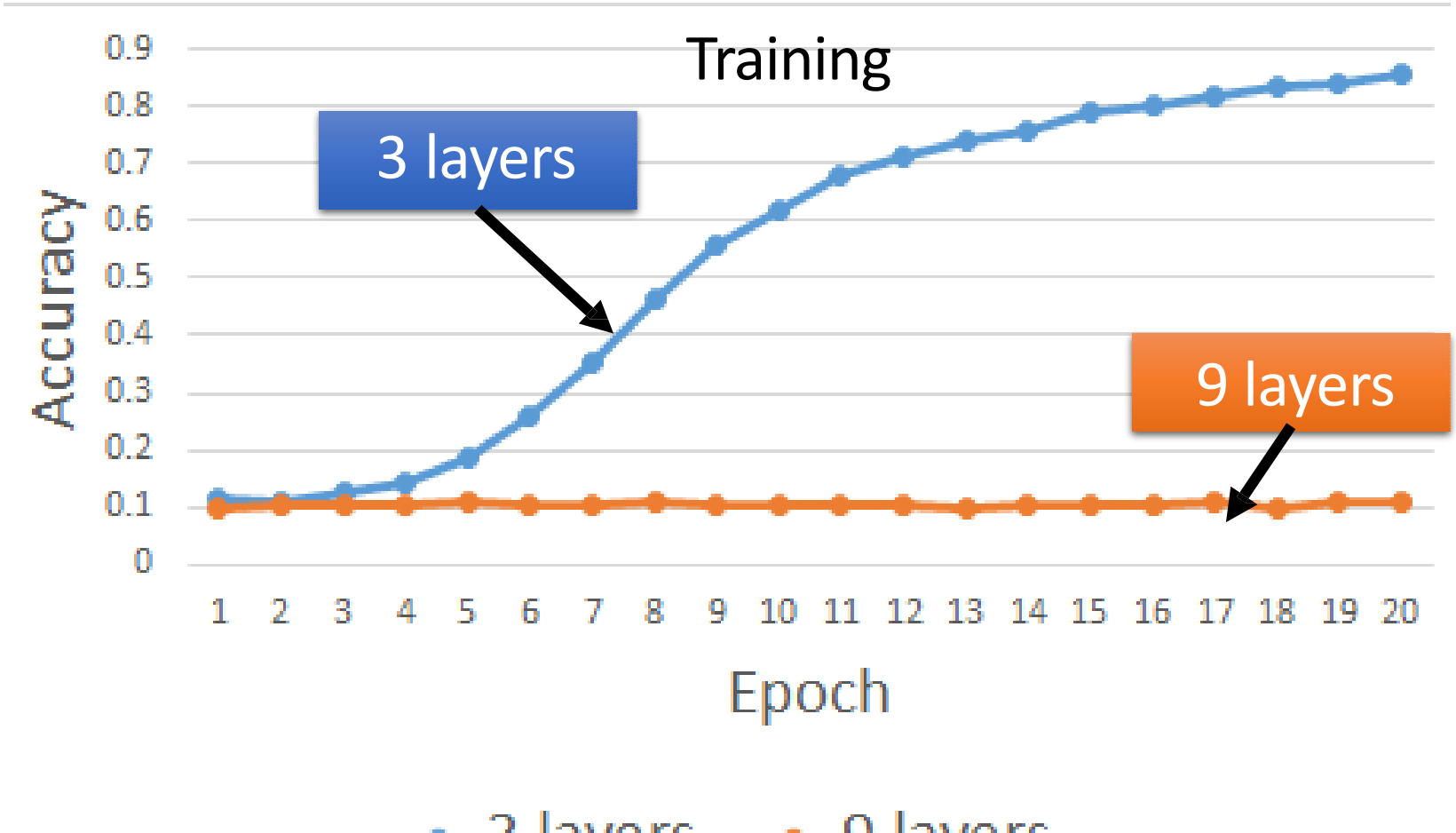
難以獲得深度神經網路的力量...



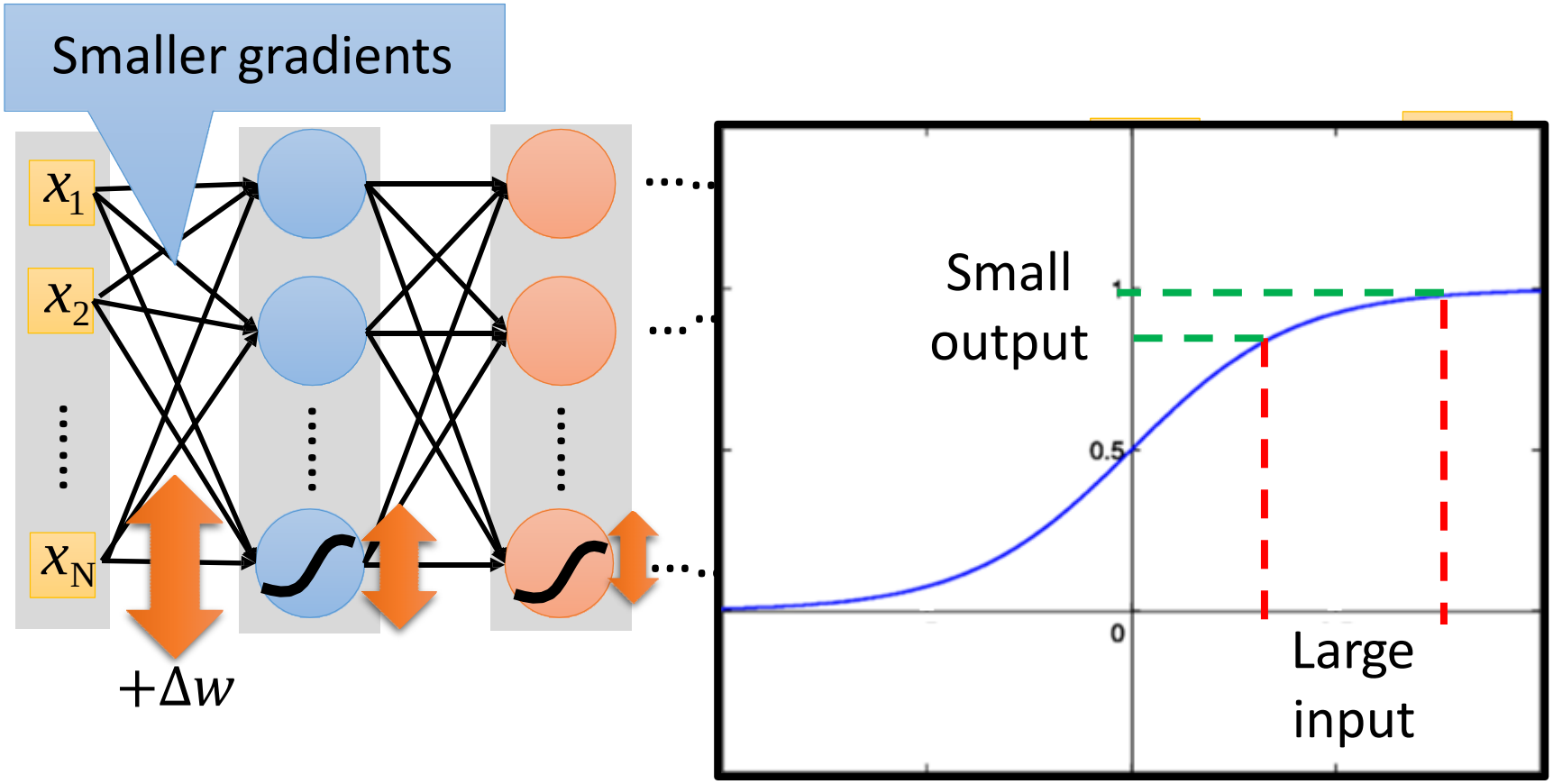
讓我們試一試

Testing:

	Accuracy
3 layers	0.84
9 layers	0.11



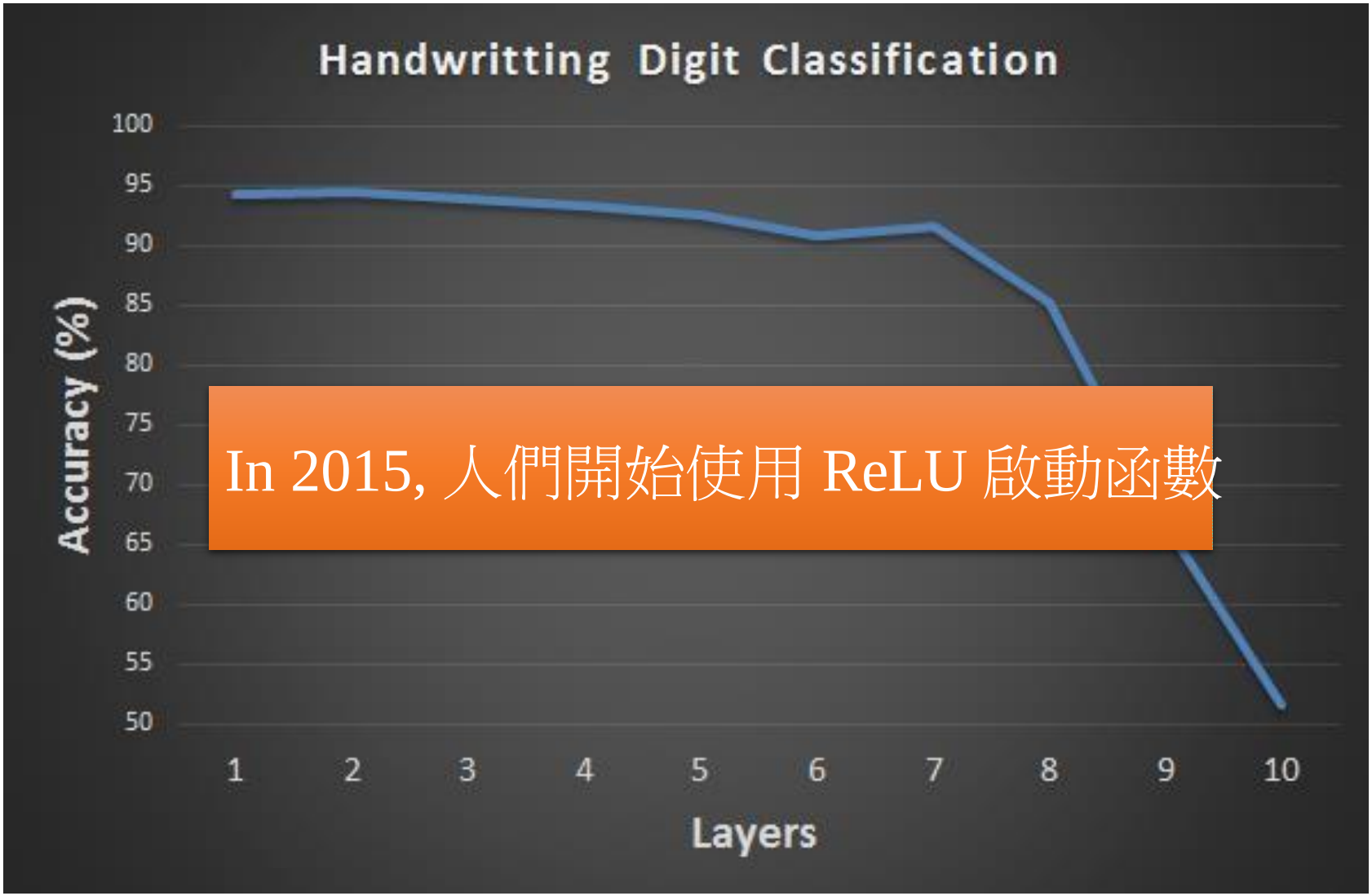
梯度消失問題



直觀的計算導數的方法 ...

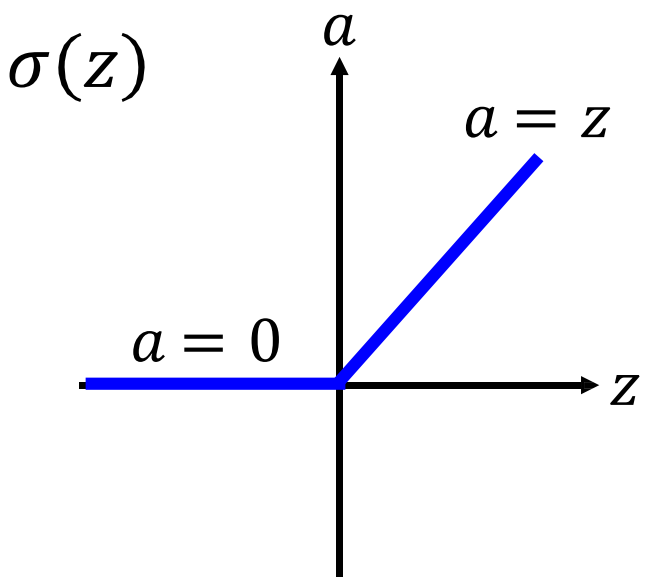
$$\frac{\partial l}{\partial w} =? \frac{\Delta l}{\Delta w}$$

難以獲得深度神經網路的力量...



ReLU

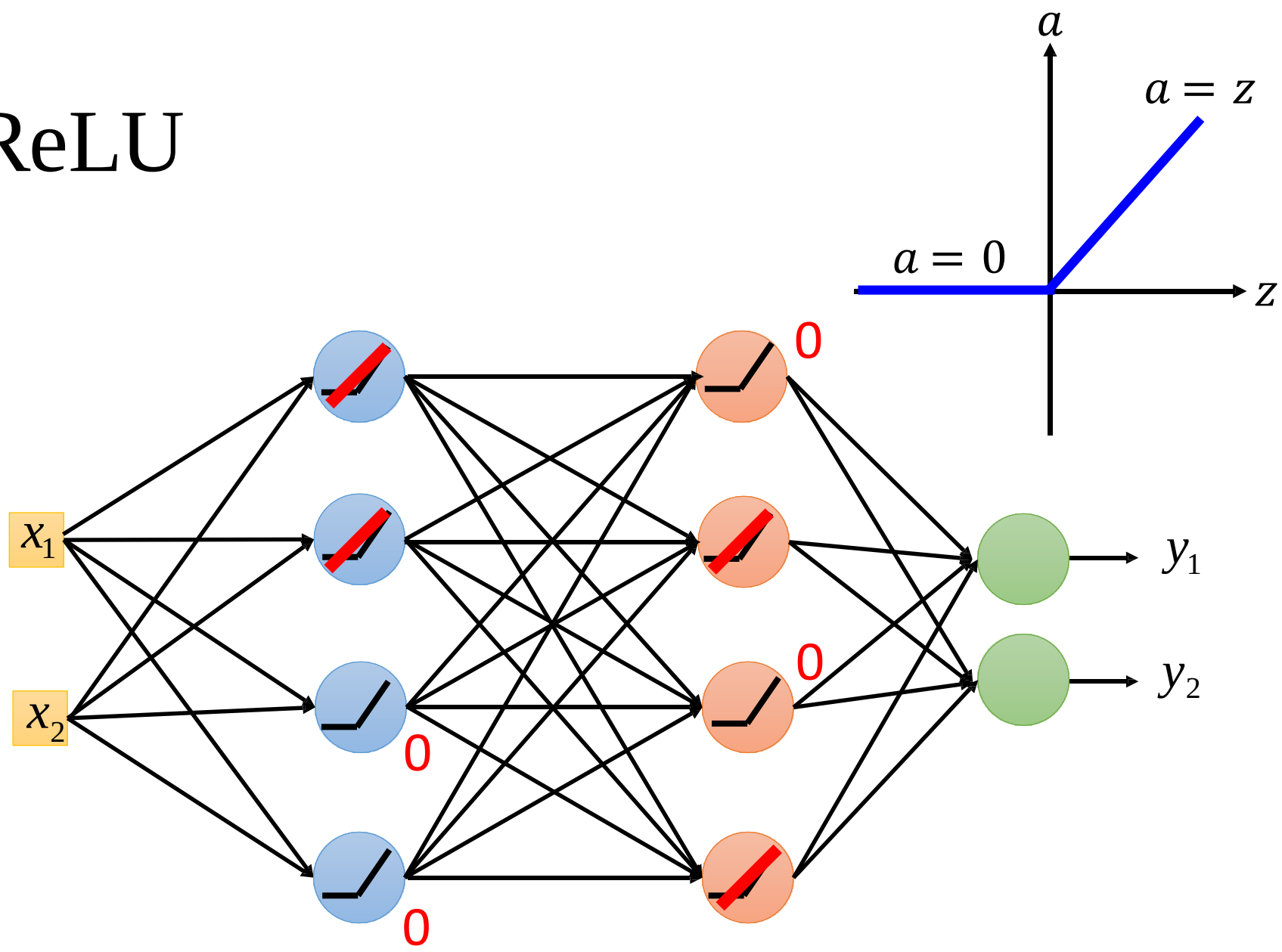
- Rectified Linear Unit (ReLU)



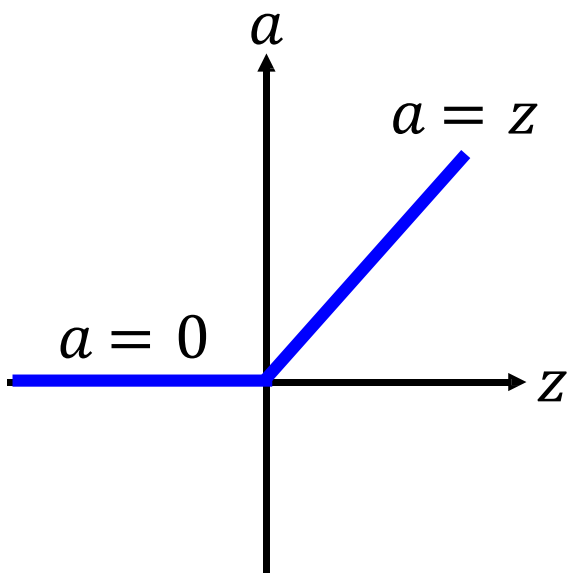
Reason:

1. 計算速度快
2. 生物上的原因
3. 解決梯度下降問題

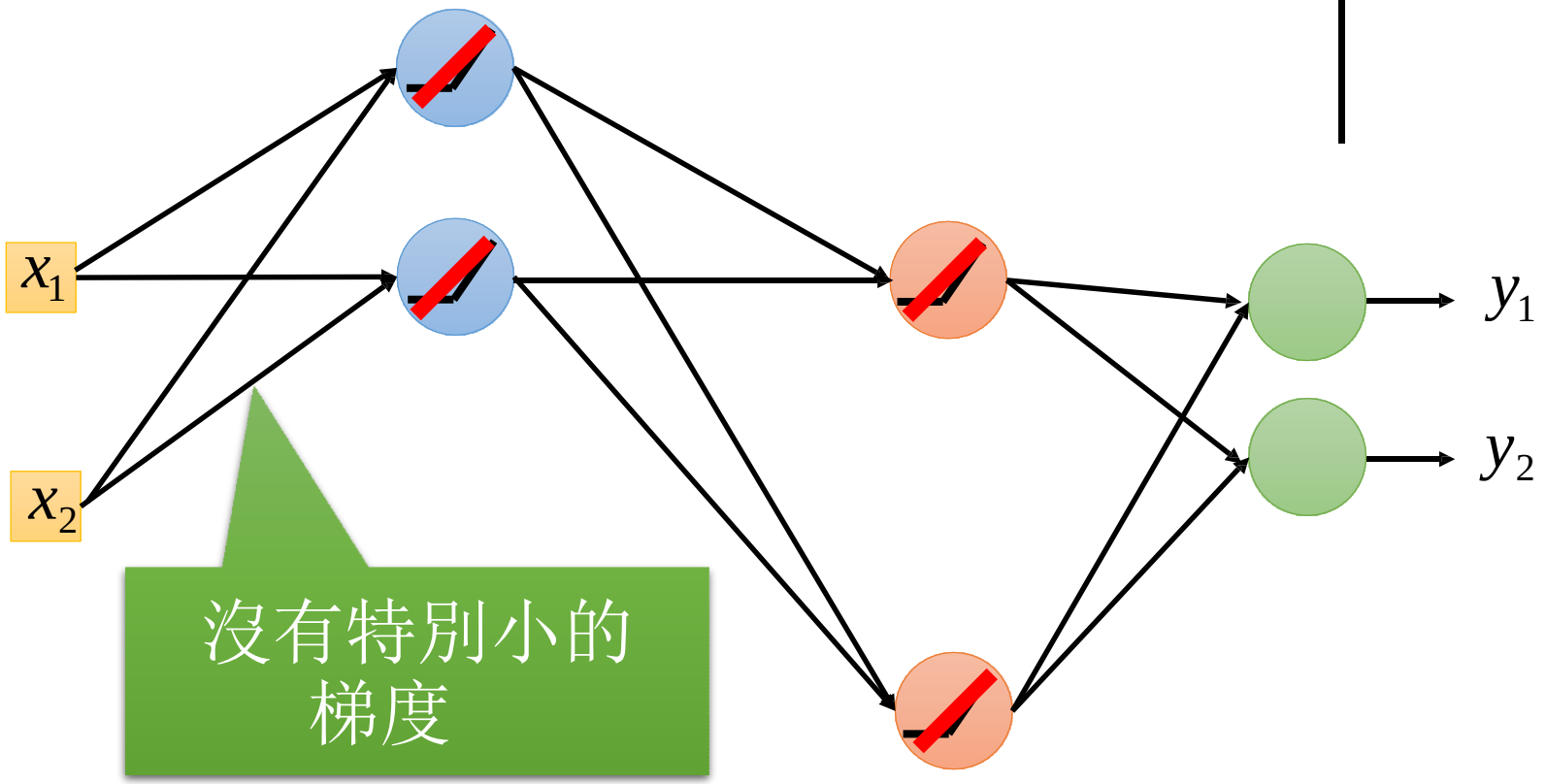
ReLU



ReLU



一個更瘦長的線性網路



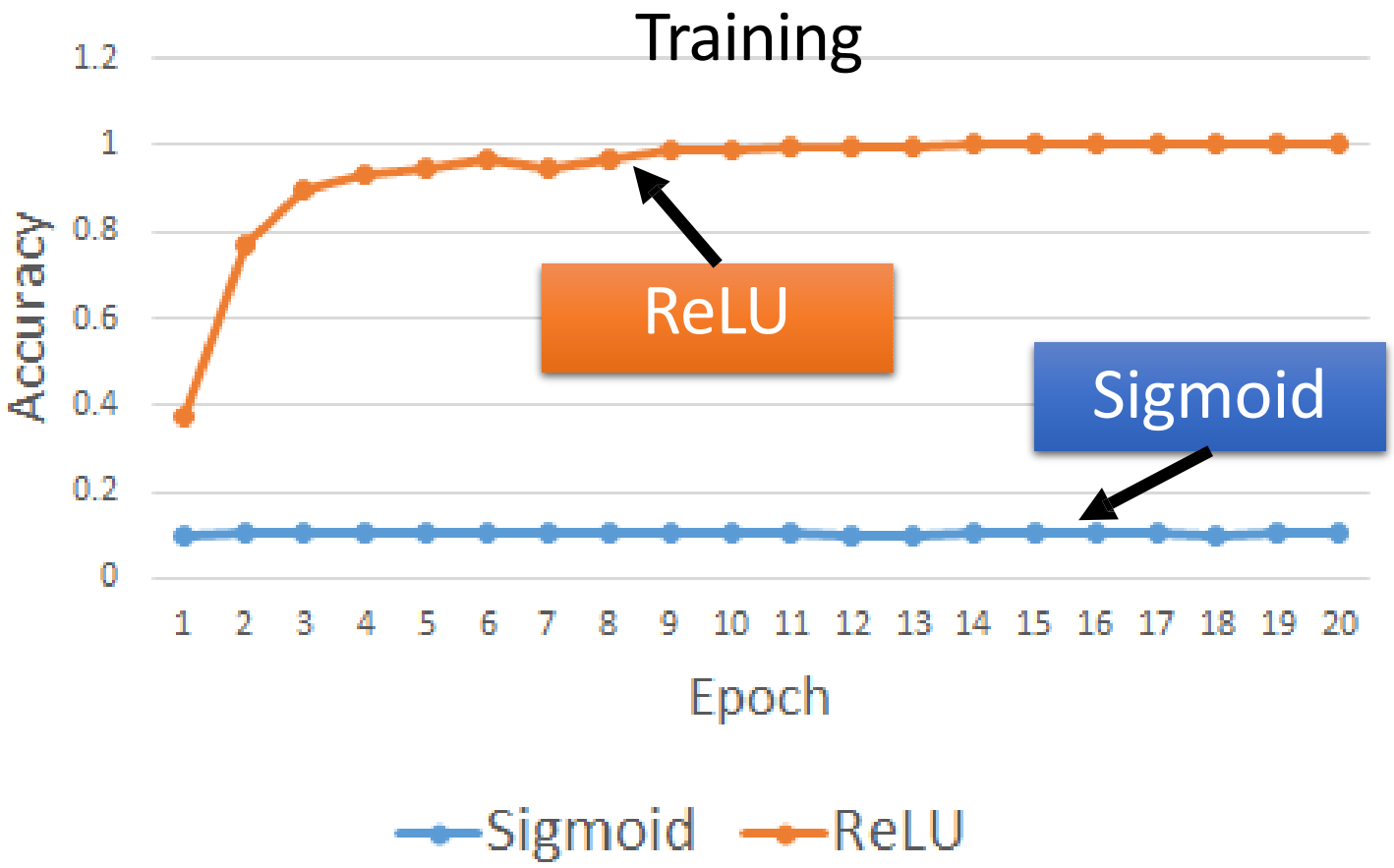
沒有特別小的梯度

Let's try it

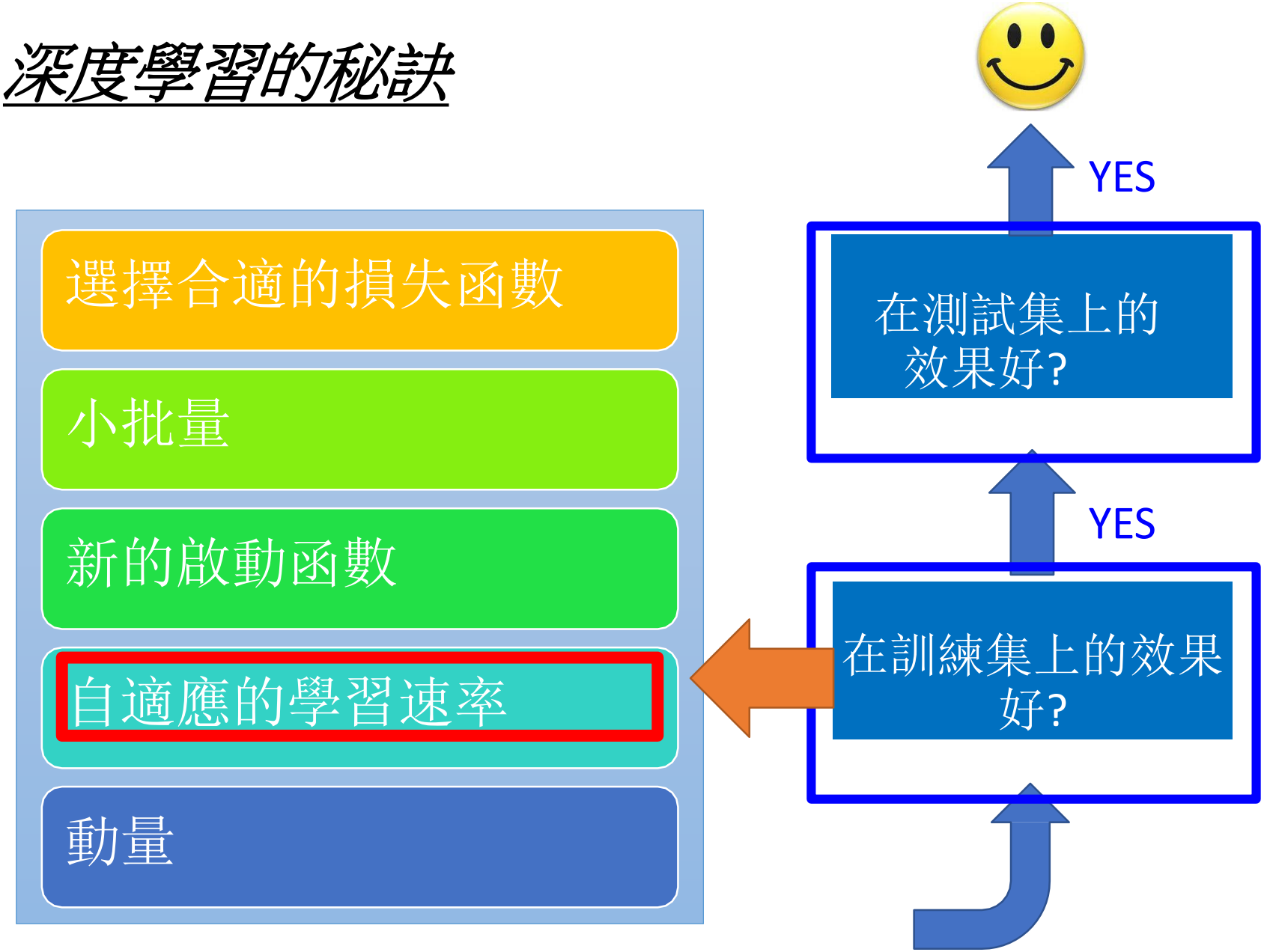
- 9 layers

Testing:

9 layers	Accuracy
Sigmoid	0.11
ReLU	0.96

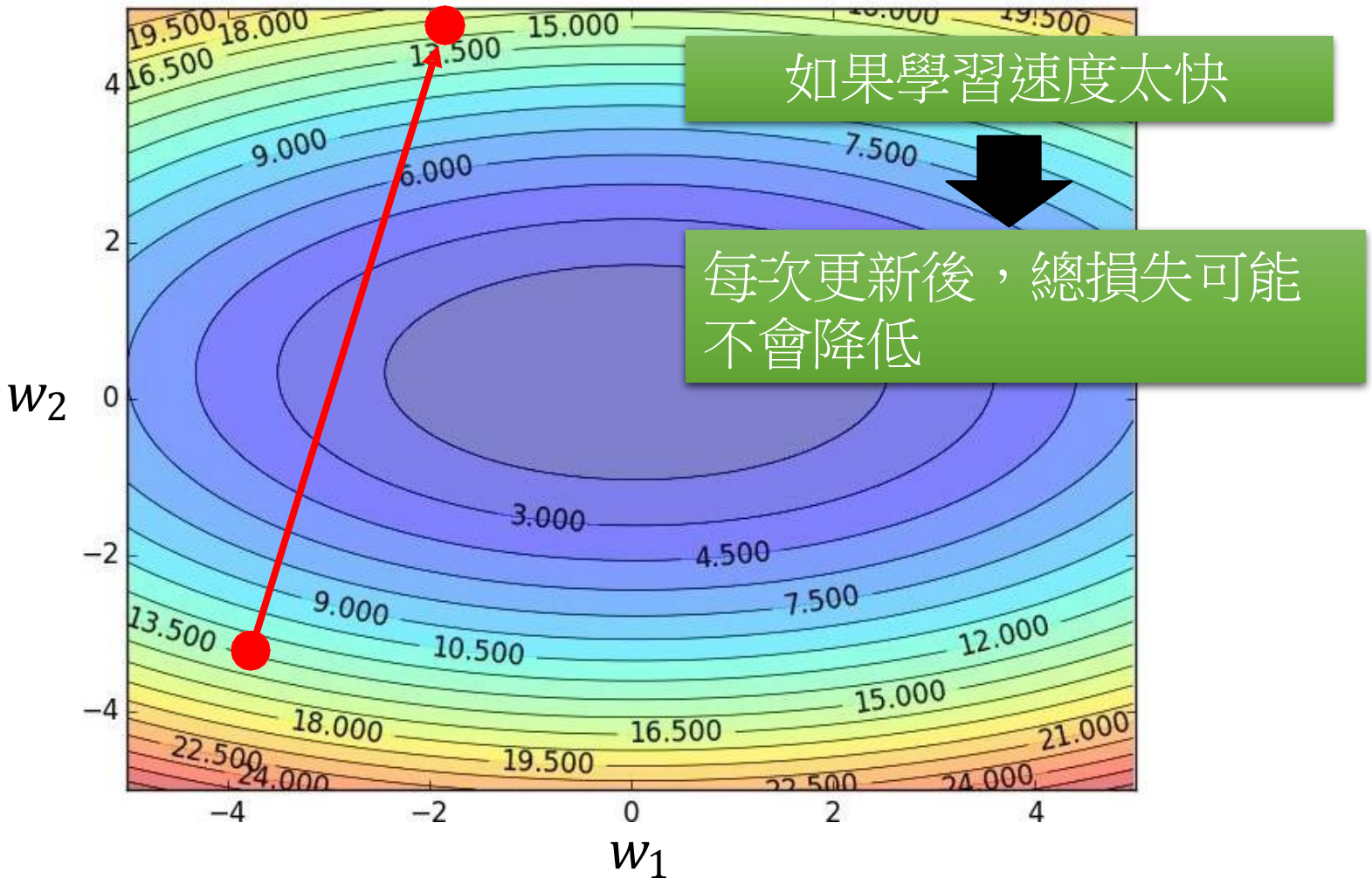


深度學習的秘訣


選擇合適的損失函數
小批量
新的啟動函數
自適應的學習速率
動量
在訓練集上的效果好?
YES
在測試集上的效果好?
YES

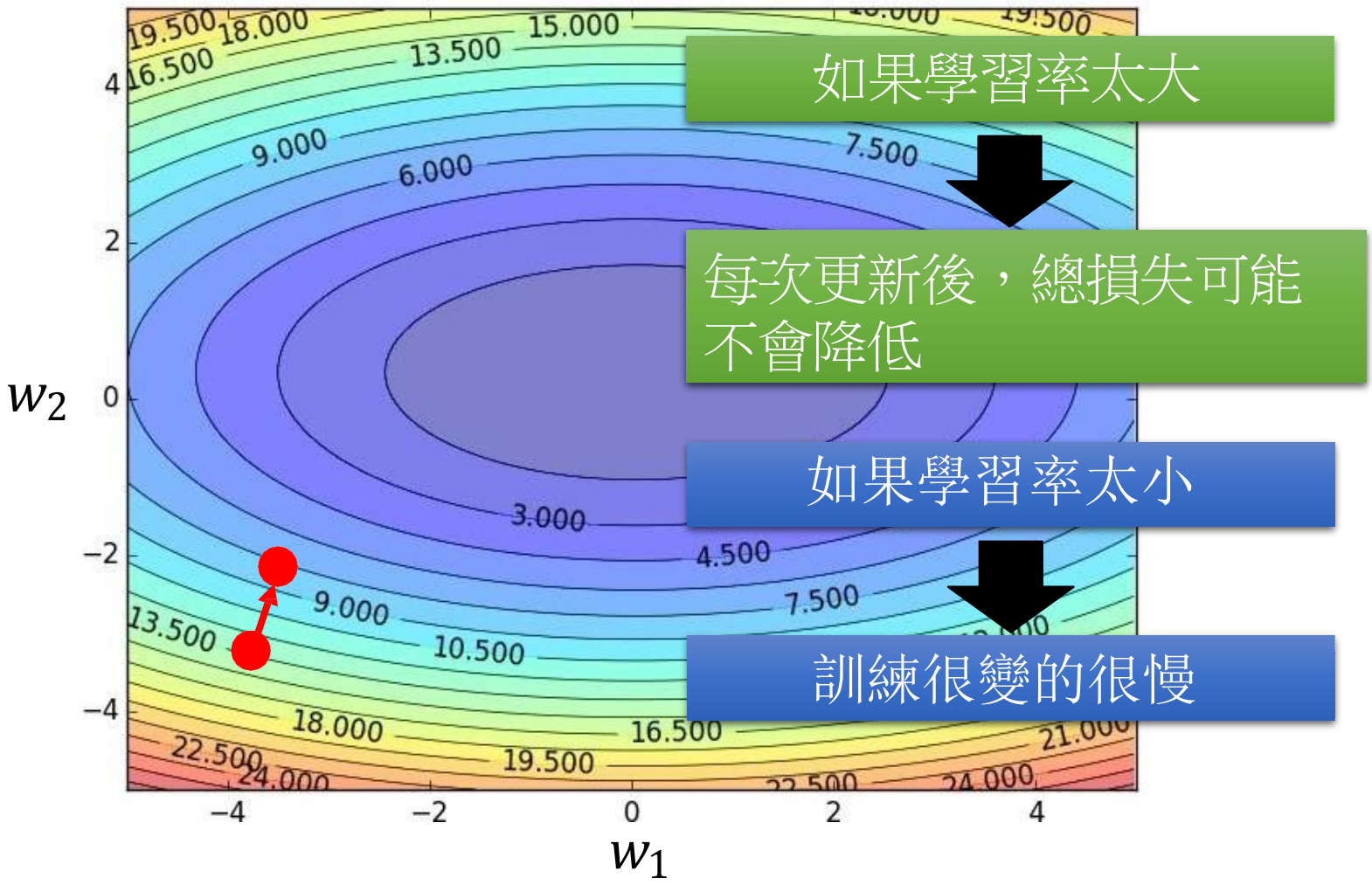
學習率

設置學習率 η



學習率

設置學習率 η



如果學習率太大

每次更新後，總損失可能不會降低

如果學習率太小

訓練很變的很慢

Adagrad

Original: $w \leftarrow w - \eta \partial L / \partial w$

Adagrad: $w \leftarrow w - \square_w \partial L / \partial w$

參數依賴學習率

$\square_w = \dfrac{\eta}{\sqrt{\sum_{i=0}^{t} (g^i)^2}}$

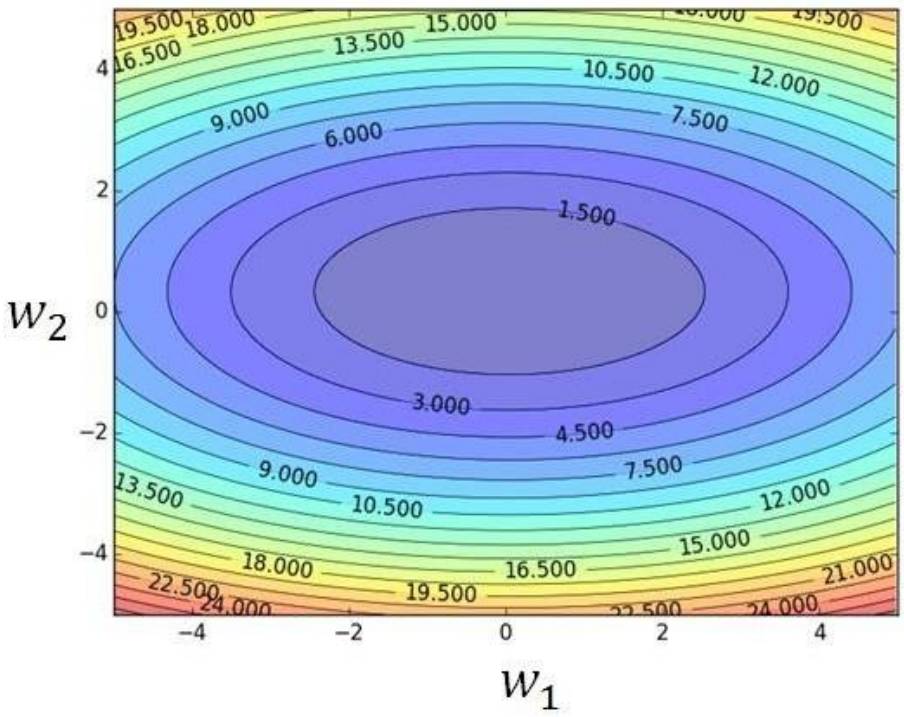
常量

g^i is $\partial L / \partial w$ obtained at the i-th update

前一個導數的平方的總和

較大的導數

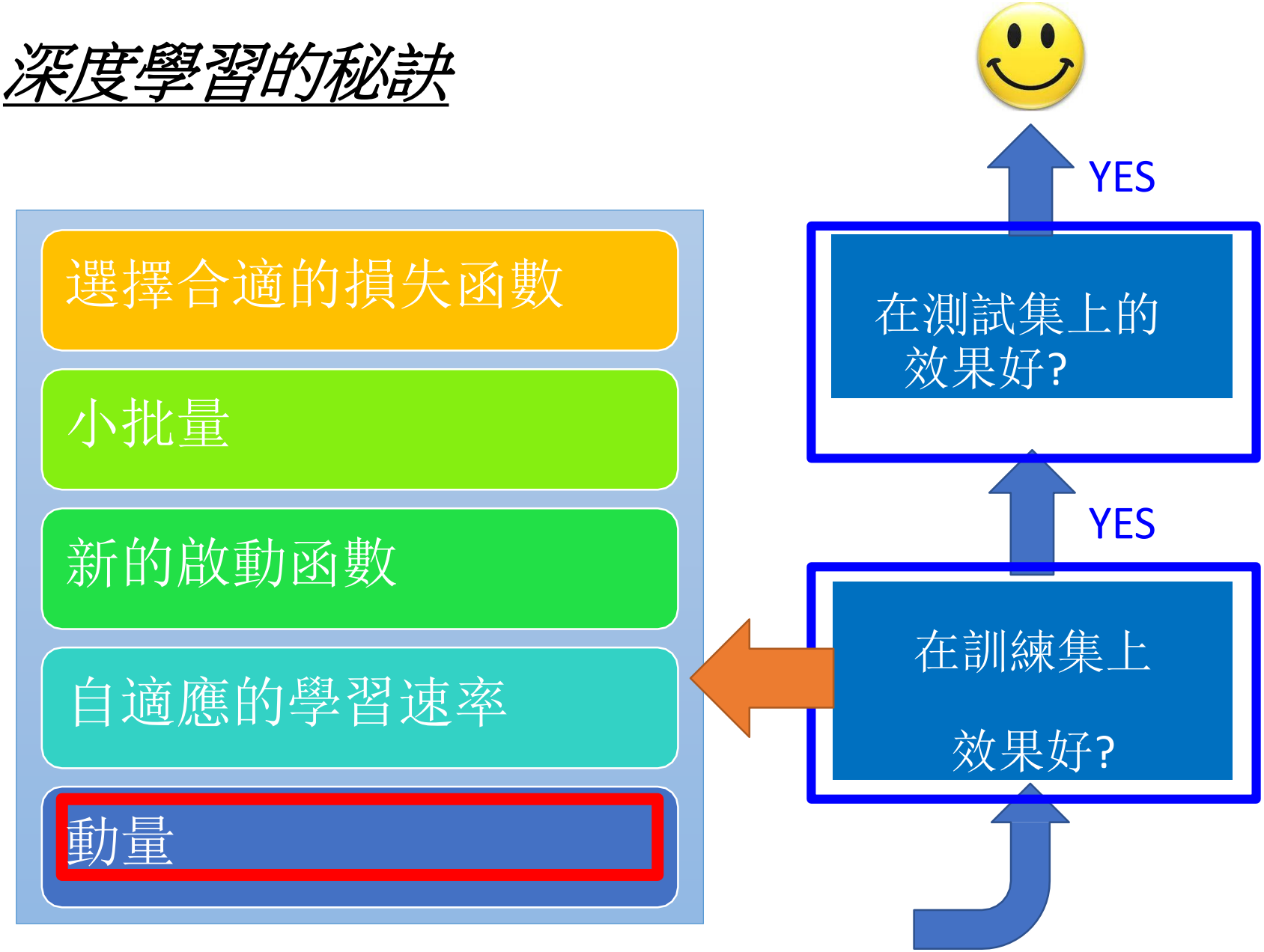
較小的學習速率



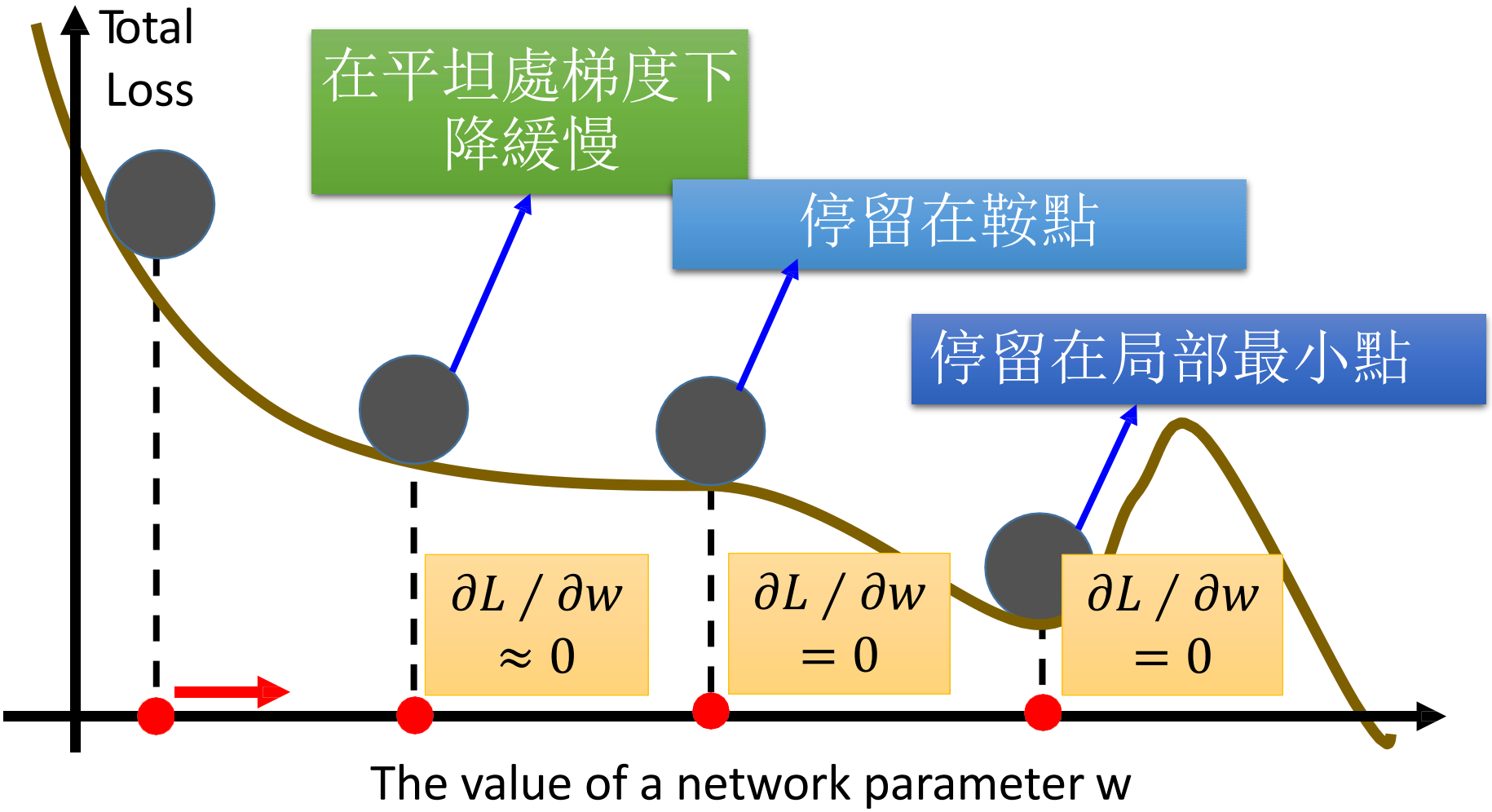
較小的導數

較大的學習速率

深度學習的秘訣

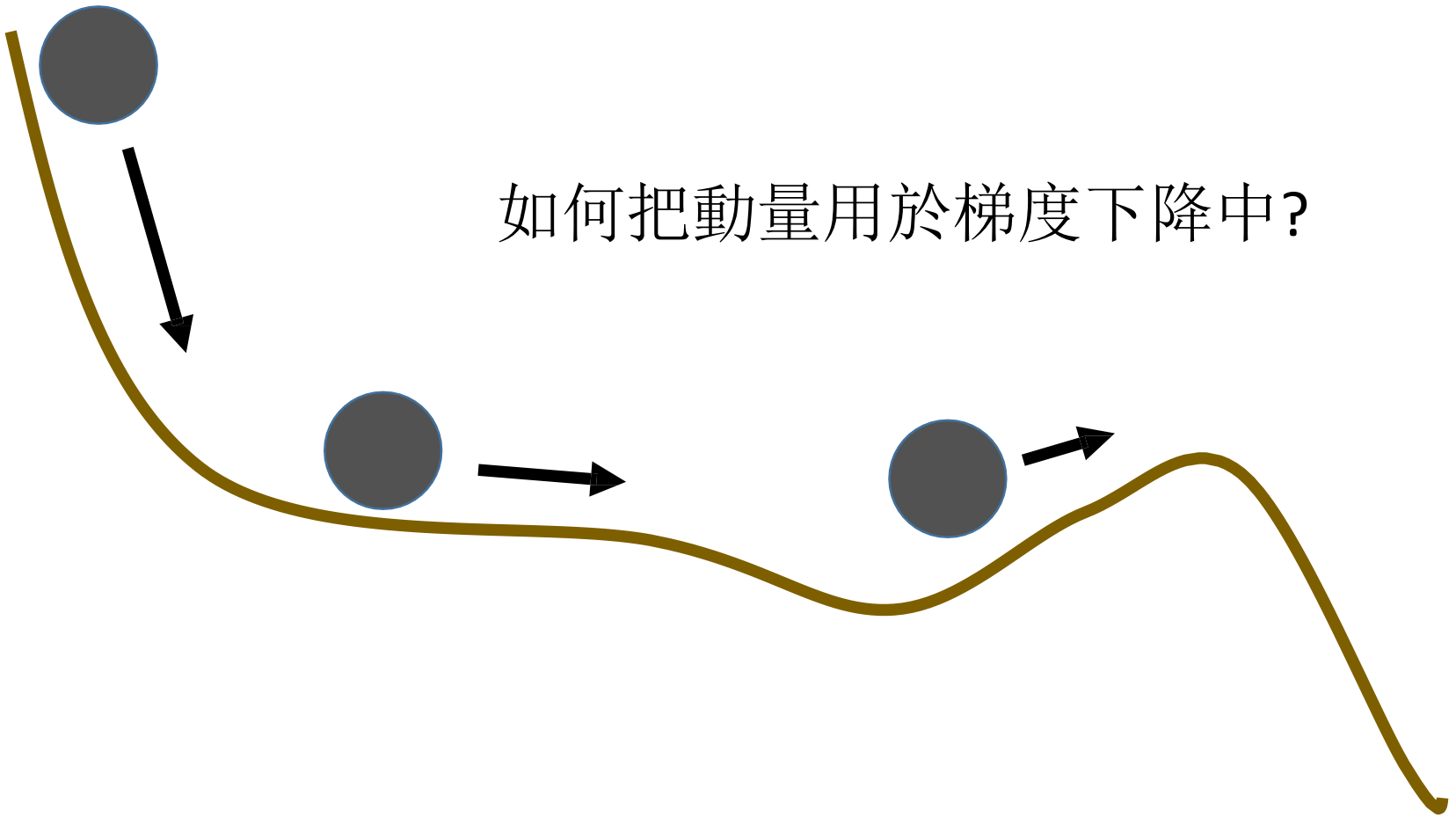

選擇合適的損失函數
小批量
新的啟動函數
自適應的學習速率
動量
在訓練集上
效果好?
YES
在測試集上的
效果好?
YES

很難找到最佳的網路參數



在真實的世界裡 ……

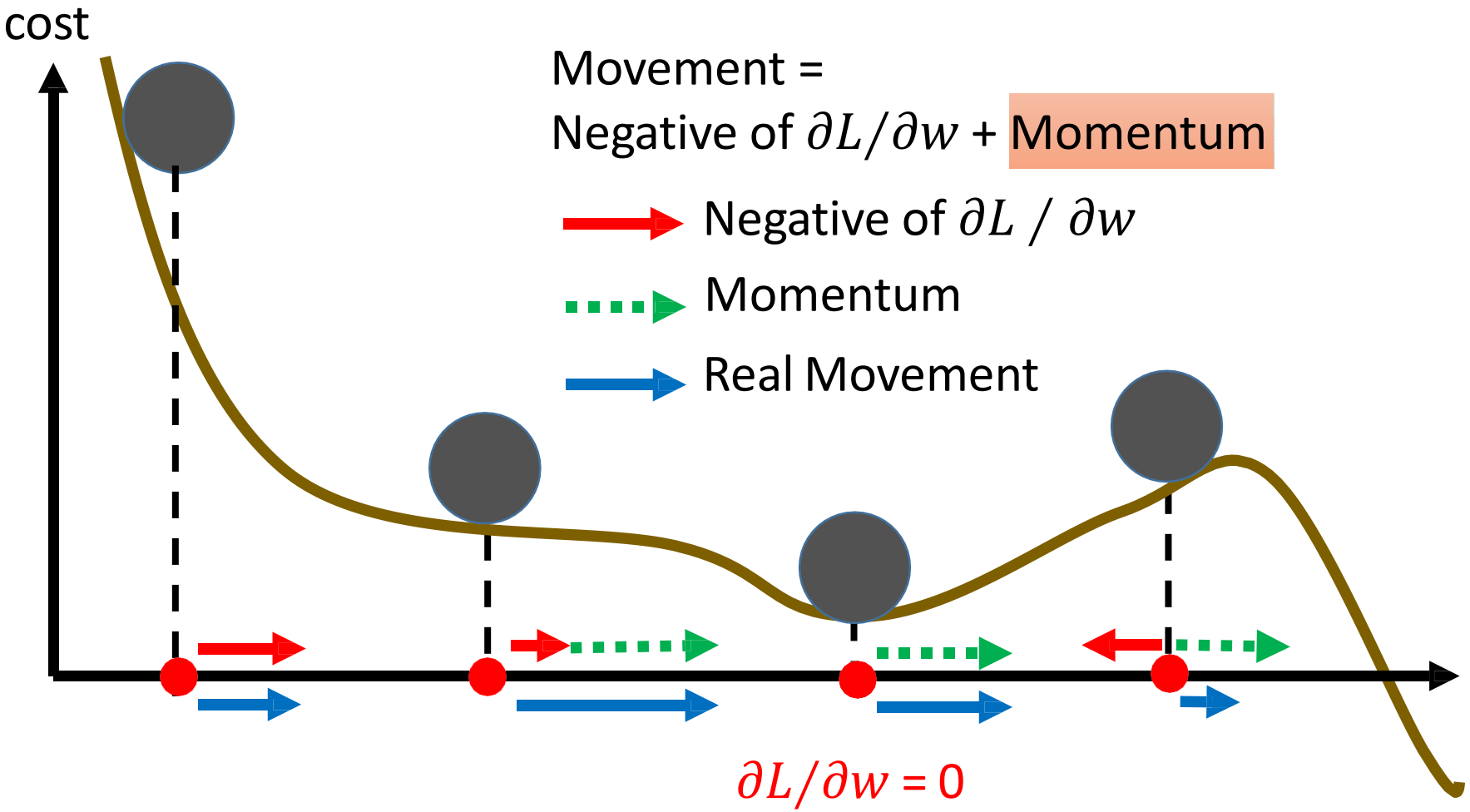
- 動量



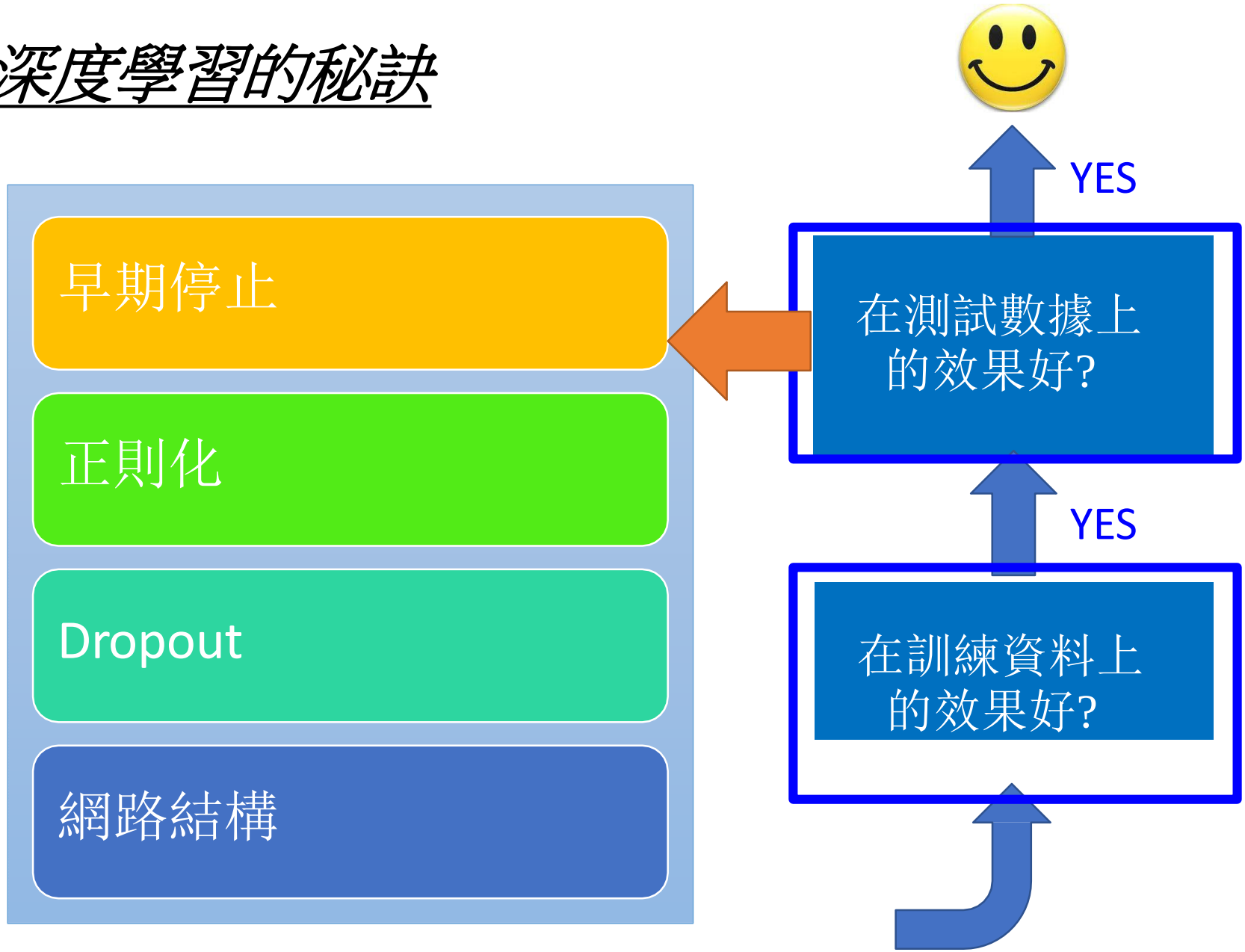
動量

仍然不能保證達到全域最小，但給了我們一些希望

……



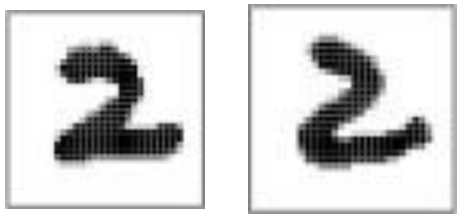
深度學習的秘訣



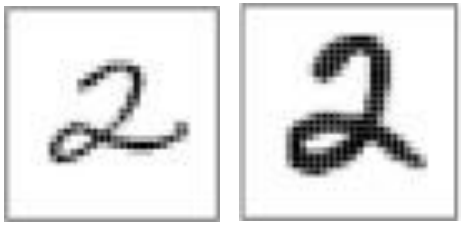
為什麼會過擬合?

- 訓練數據和測試資料是不同的

訓練數據:



測試資料:



學習目標是由訓練數據定義的

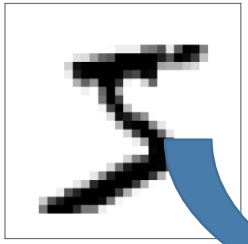
達到學習目標的參數不一定對測試數據有好的結果

過擬合的靈丹妙藥

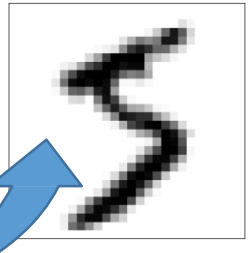
- 有更多的訓練資料
- **創造**更多的數據(?)

手寫體識別:

原始的訓練資料:



創造的訓練數據:



旋轉 15。

為什麼會過擬合?

為了進行實驗，我們在測試資料中加入了一些雜訊

```
       -1.36230370e-01,   1.03749340e-01,   1.15432226e-01,
        2.58670464e-01,   1.48774333e+00,   1.92885328e+00,
        1.70038673e+00,   2.46242981e+00,   1.21244572e+00,
       -9.28660713e-01,   3.63209761e-01,  -1.81327713e+00,
       -1.97910760e-01,   4.32874592e-01,  -5.40565788e-01,
        2.95630655e-01,   2.07984424e+00,  -1.84243292e+00,
       -5.11166017e-01,  -5.80935128e-01,   1.06273647e+00,
        1.80551097e-02,   2.27983997e-02,  -1.67979148e+00,
        8.12423001e-01,  -6.25888706e-01,  -1.25027082e+00,
        6.15135458e-01,  -1.21394611e-01,  -1.28089527e+00,
        3.24609806e-01,   6.70569391e-01,   1.49161323e-01,
        8.01573609e-01,   6.43116741e-01,  -9.37629233e-02,
        1.74677366e+00,   6.80996008e-01,  -7.03717611e-01,
        1.02079749e-01,   1.19505614e+00,  -2.77959386e-01,
       -5.21652916e-02,   3.53683601e-01,  -4.08310762e-01,
       -1.81042967e+00,  -9.03308062e-01,   1.05404509e+00,
       -9.80876877e-01,   3.52078891e-01,   6.65981840e-01,
        1.06550150e+00,  -2.28433613e-01,   3.64483904e-01,
       -1.51484666e+00,  -7.52612872e-02,  -2.97058082e-01,
       -7.27414382e-01,  -2.45875340e-01,  -1.27948942e-01,
       -3.69310620e-01,  -2.62300428e+00,   2.11585073e+00,
        6.85561585e-01,  -1.57443985e-01,   1.38128777e+00,
        6.84265587e-02,   3.12536292e-01,   4.54253185e-01,
       -7.88471875e-01,  -6.58403343e-02,  -1.41847985e+00,
       -1.39753340e-01,  -5.55354856e-01,  -5.01917779e-01,
        6.93118522e-01,  -2.45360497e-01,  -1.26943186e+00,
       -2.62323855e-01])

n [3]: x test[0]
```

為什麼會過擬合?

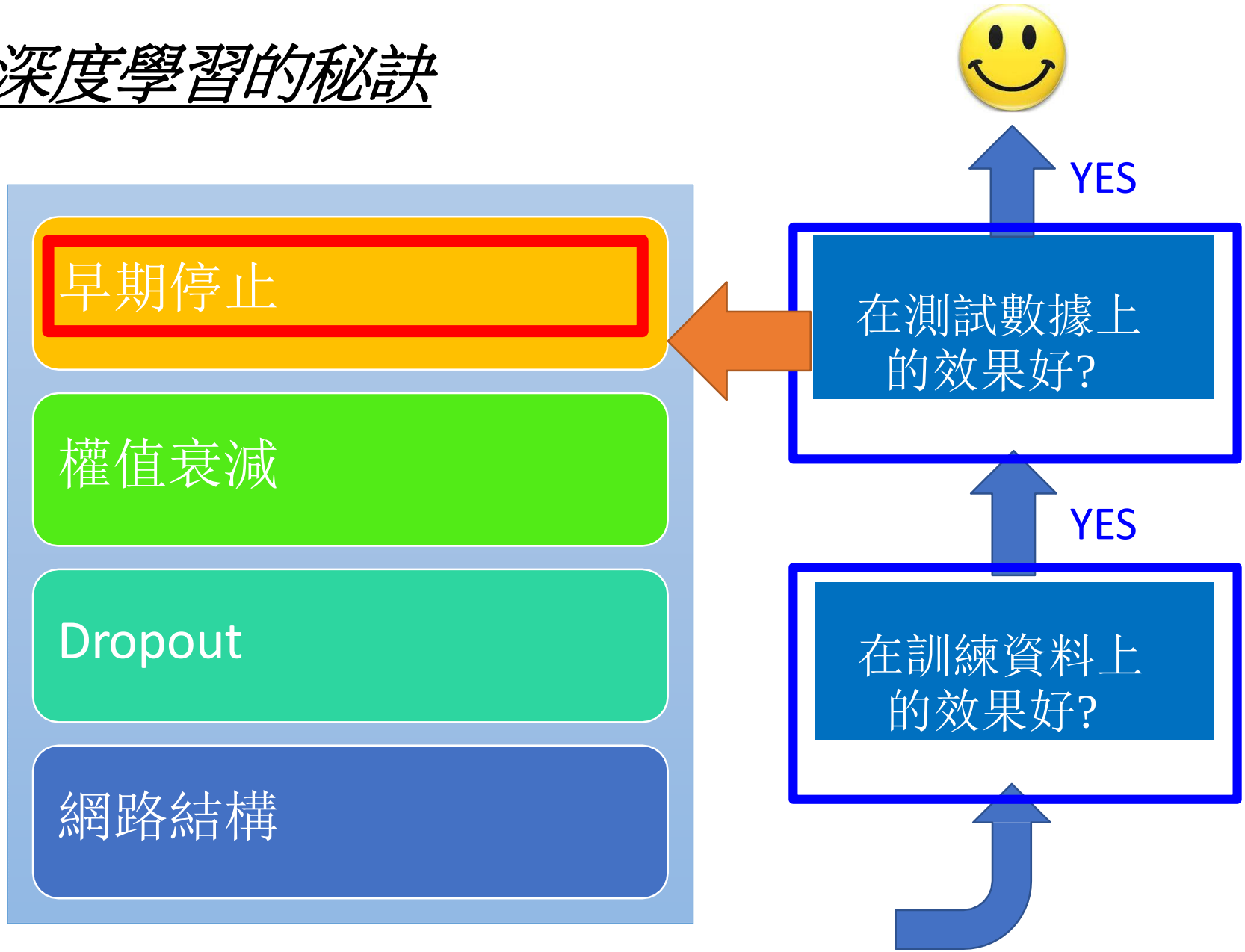
為了進行實驗，我們在測試資料中加入了一些雜訊

測試:

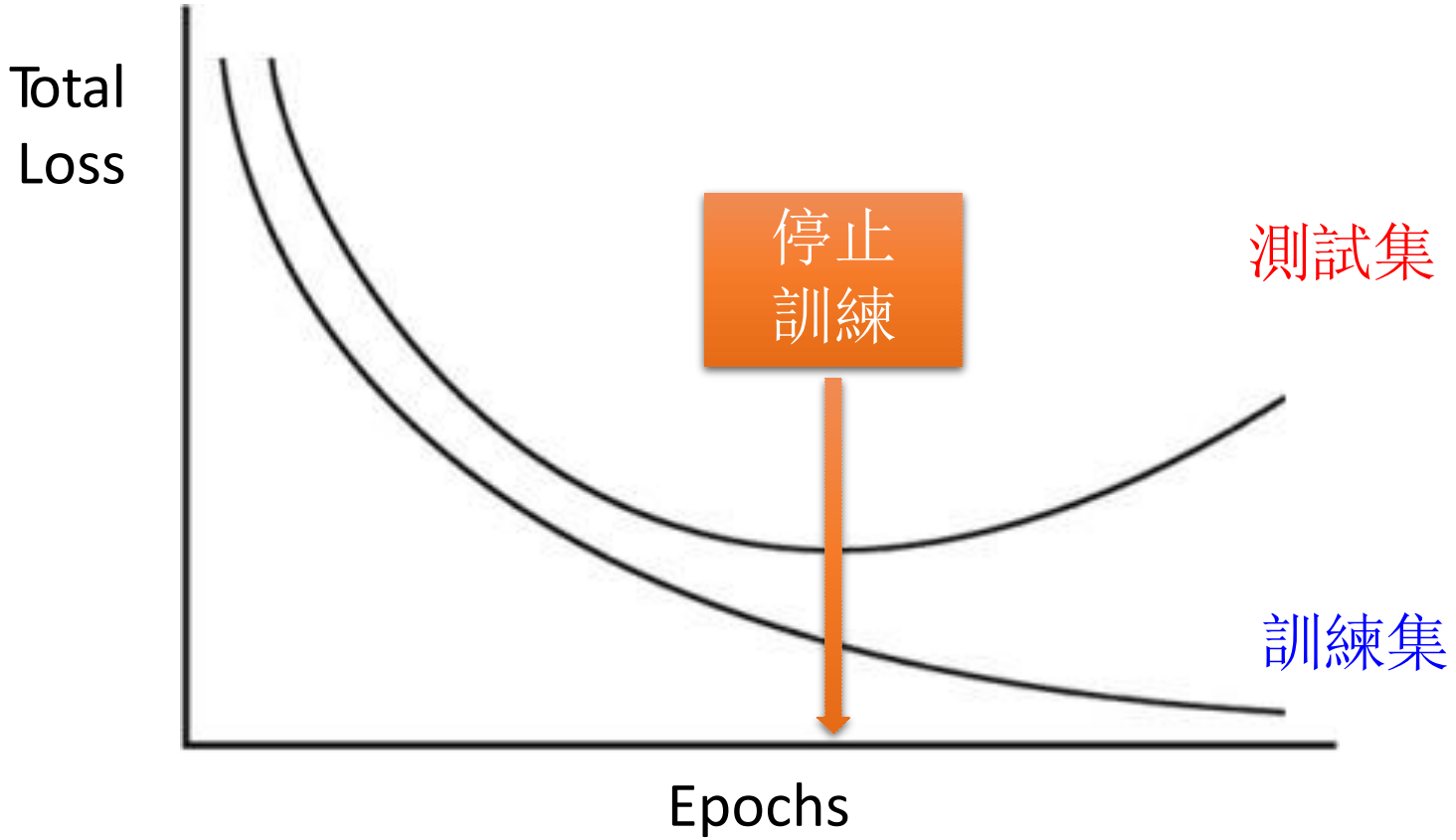
	準確率
Clean	0.97
Noisy	0.50

訓練不受影響

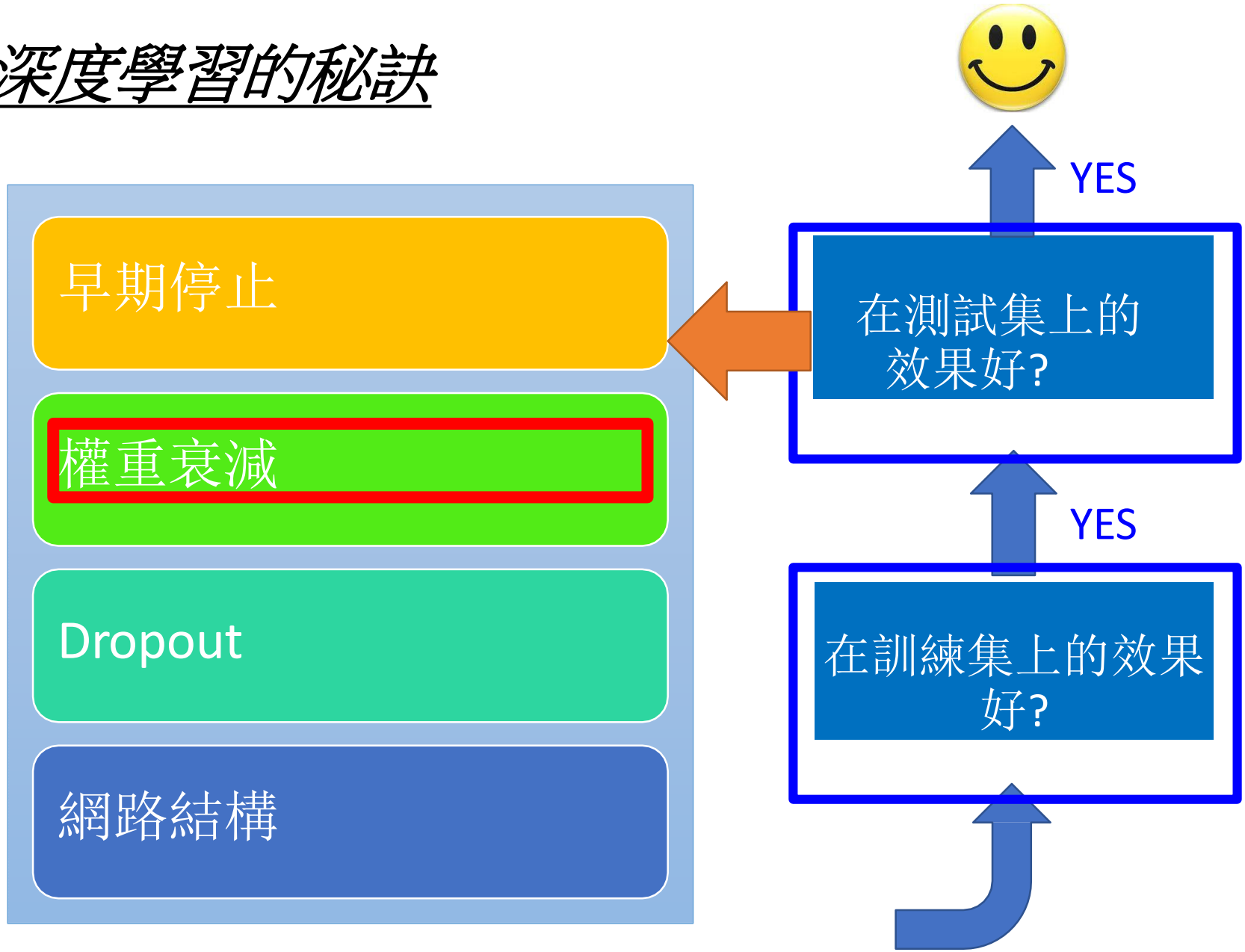
深度學習的秘訣



早期停止

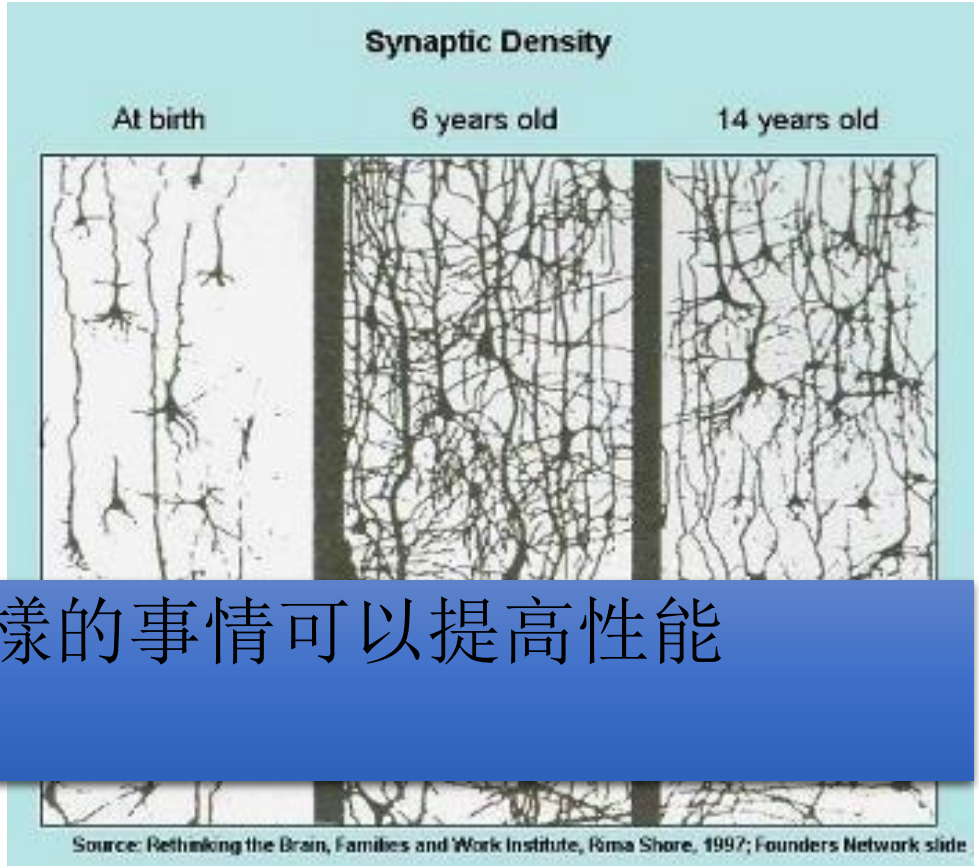


深度學習的秘訣



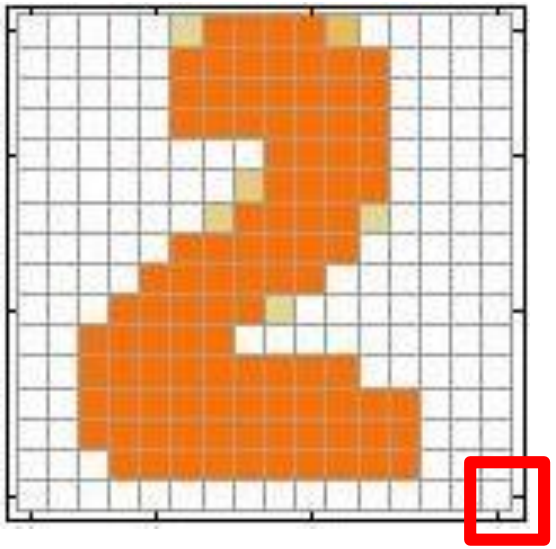
權重衰減

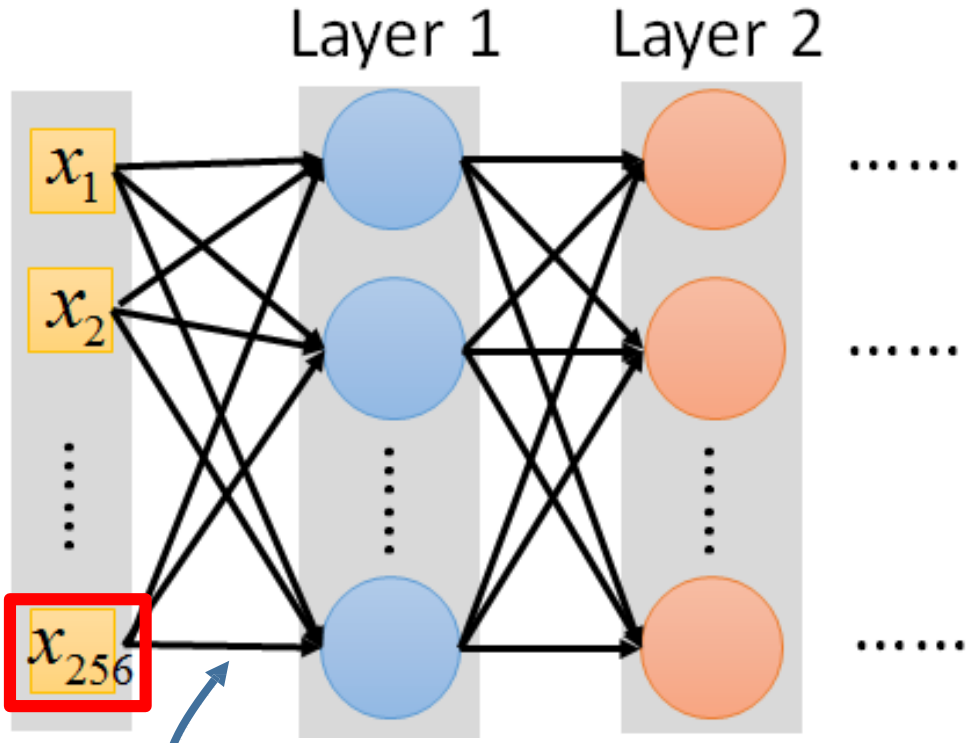
- 我們的大腦刪除掉了神經元之間無用的聯系



對機器的大腦做同樣的事情可以提高性能

權重衰減





權重衰減是正則化方式的一種

無用的

權重衰減到0 (萎縮了)

權重衰減

- 實現

原始: $w \leftarrow w - \eta \frac{\partial L}{\partial w}$

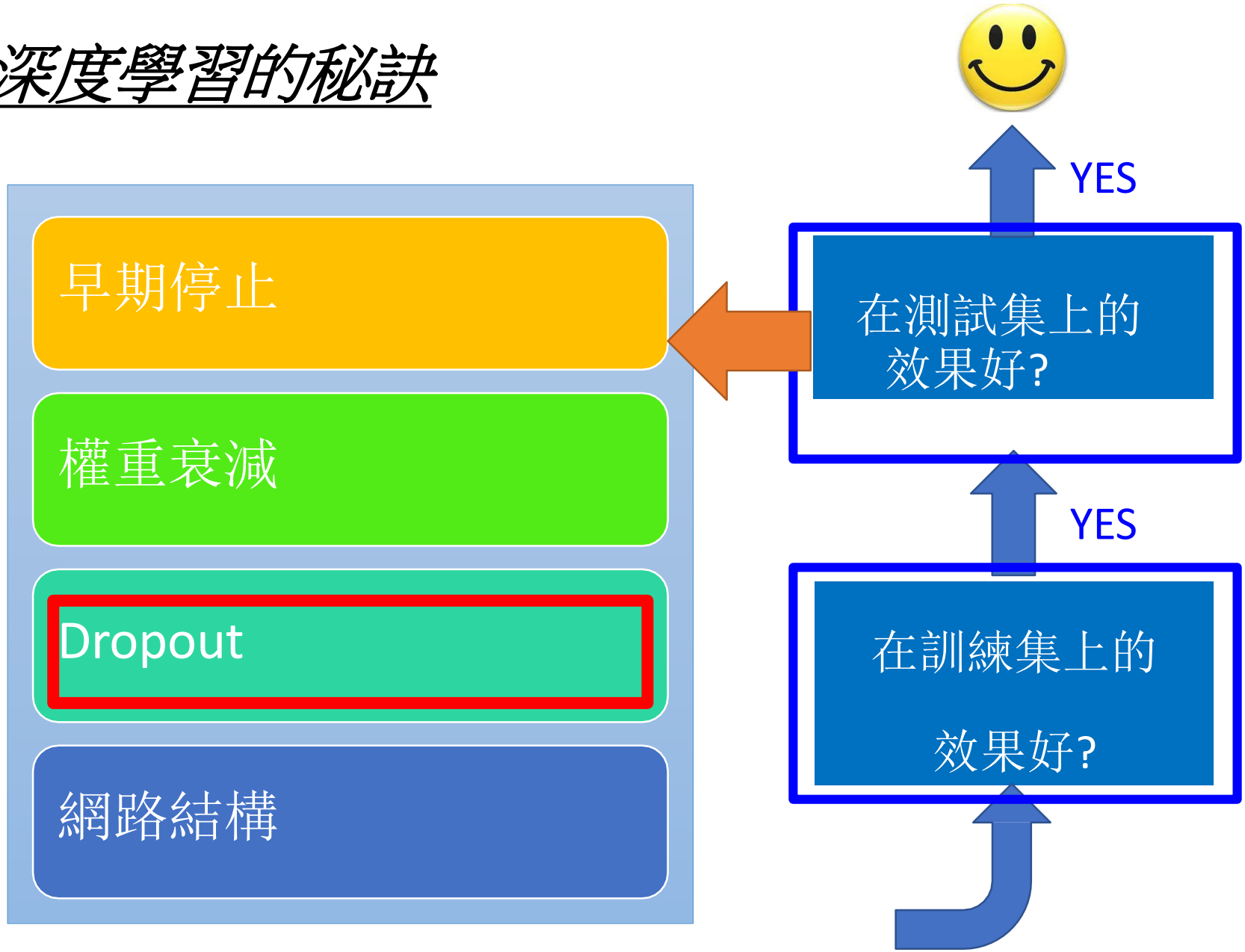
$\lambda = 0.01$

權重衰減: $w \leftarrow 0.99 w - \eta \frac{\partial L}{\partial w}$

越來越小

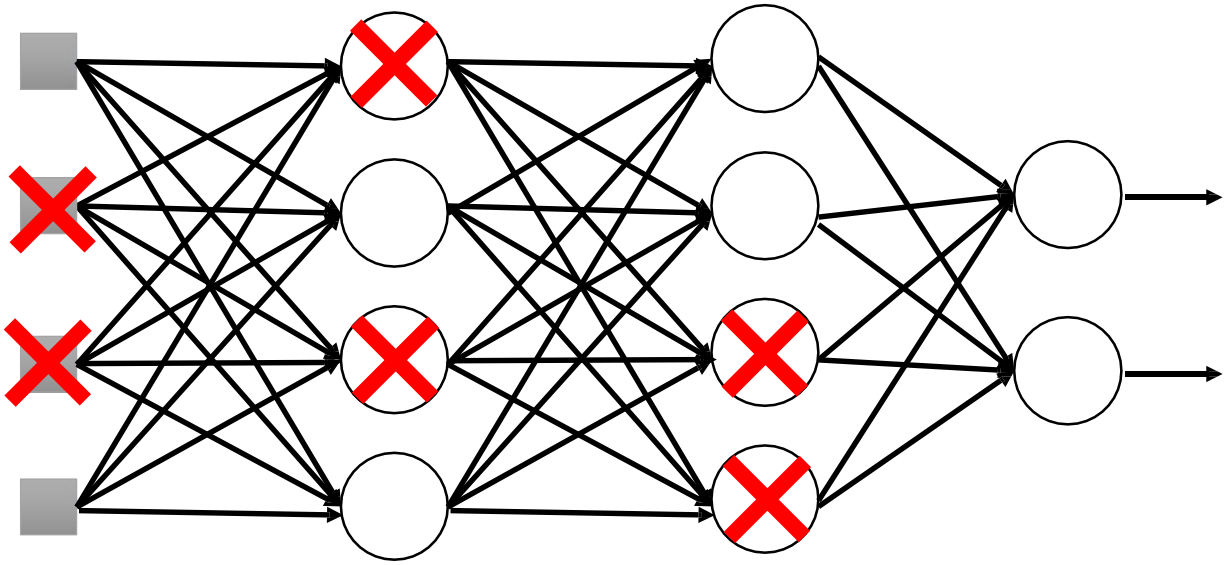
Keras: http://keras.io/regularizers/

深度學習的秘訣



Dropout

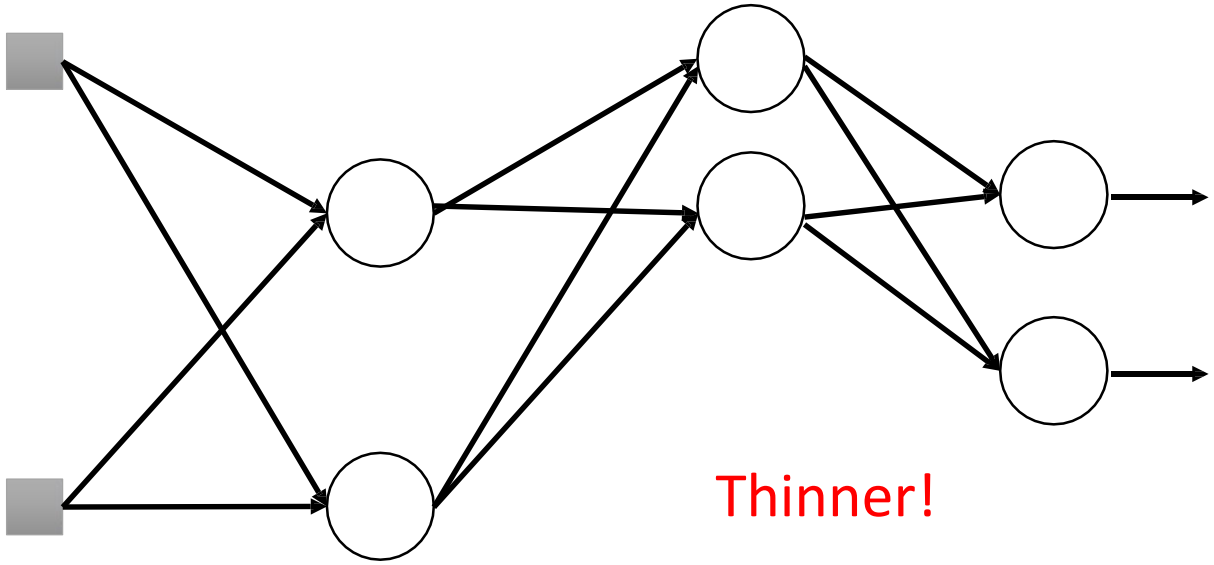
訓練:



- 每次更新參數之前
  - 每個神經元 都有p%的概率被丟棄

Dropout

訓練:



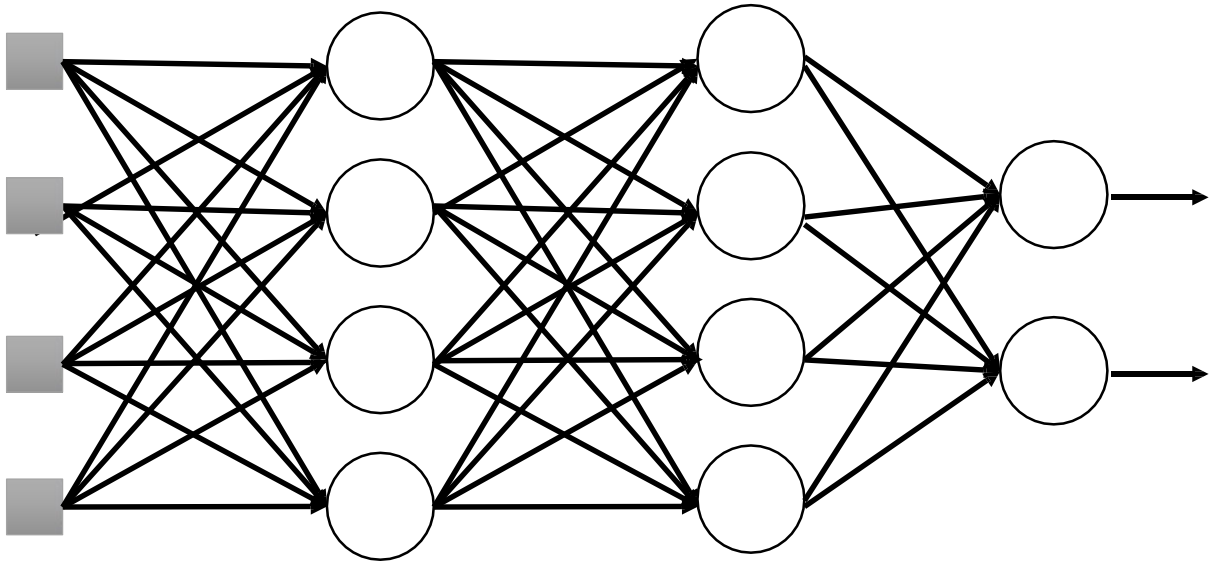
- **每個神經元 都有p%的概率被丟棄**
  - 每個神經元 都有p%的概率被丟棄

    網路結構發生了變化
  - 使用新的網路去訓練

對於每個小批量資料，我們重新採樣來dropout神經元

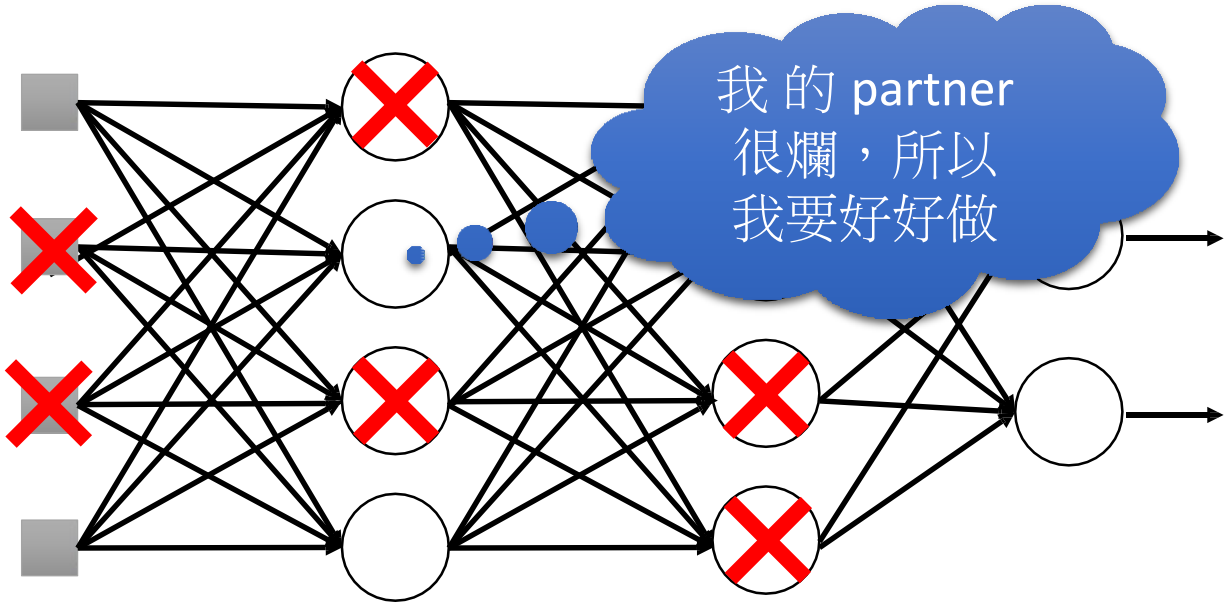
Dropout

測試:



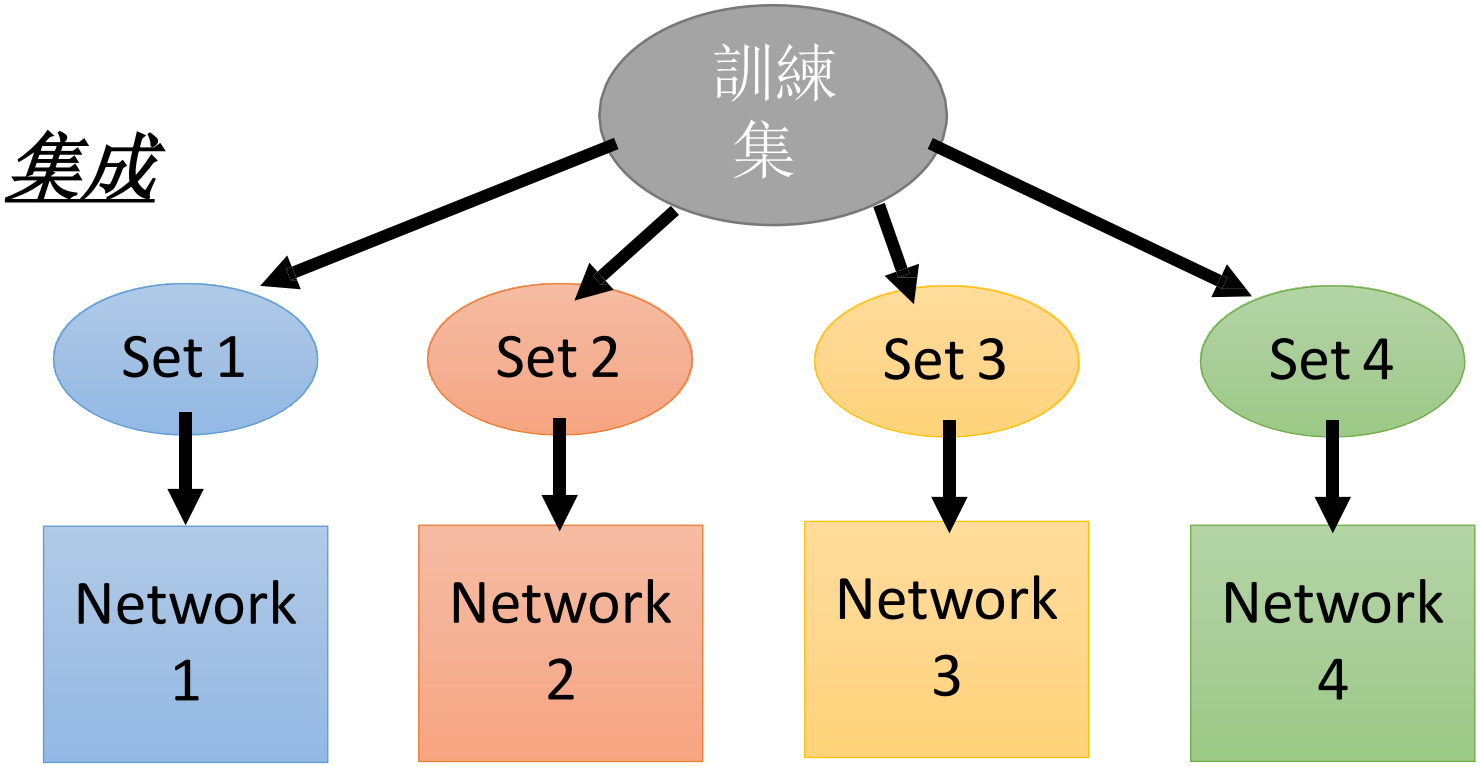
- 沒有 **dropout**
  - 如果dropout rate 在訓練集上是 p%, 所有的權重值乘以 (1-p)%
  - 假設 dropout rate 是 50%.
    If a weight $w = 1$ by training, set $w = 0.5$ for testing.

Dropout – 直觀的理由



- 當團隊合作時，如果大家都期待合作夥伴做好工作，最後什麼都不會做
- 但是，如果你知道你的搭檔會別拋棄，你會做的更好
- 當測試時, 實際上沒有人被拋棄, 所有最終能取得不錯的結果

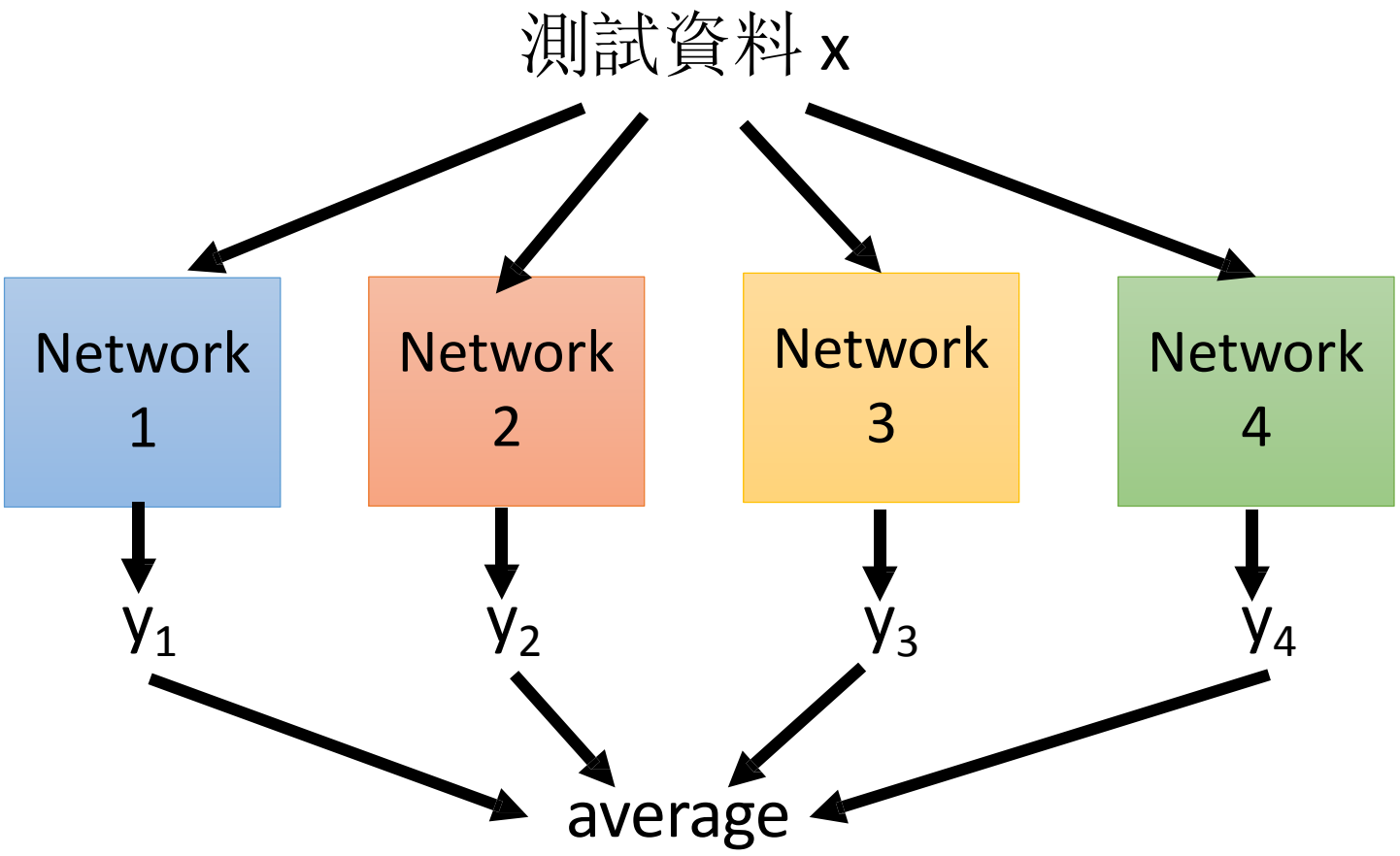
Dropout 是集成方法的一種



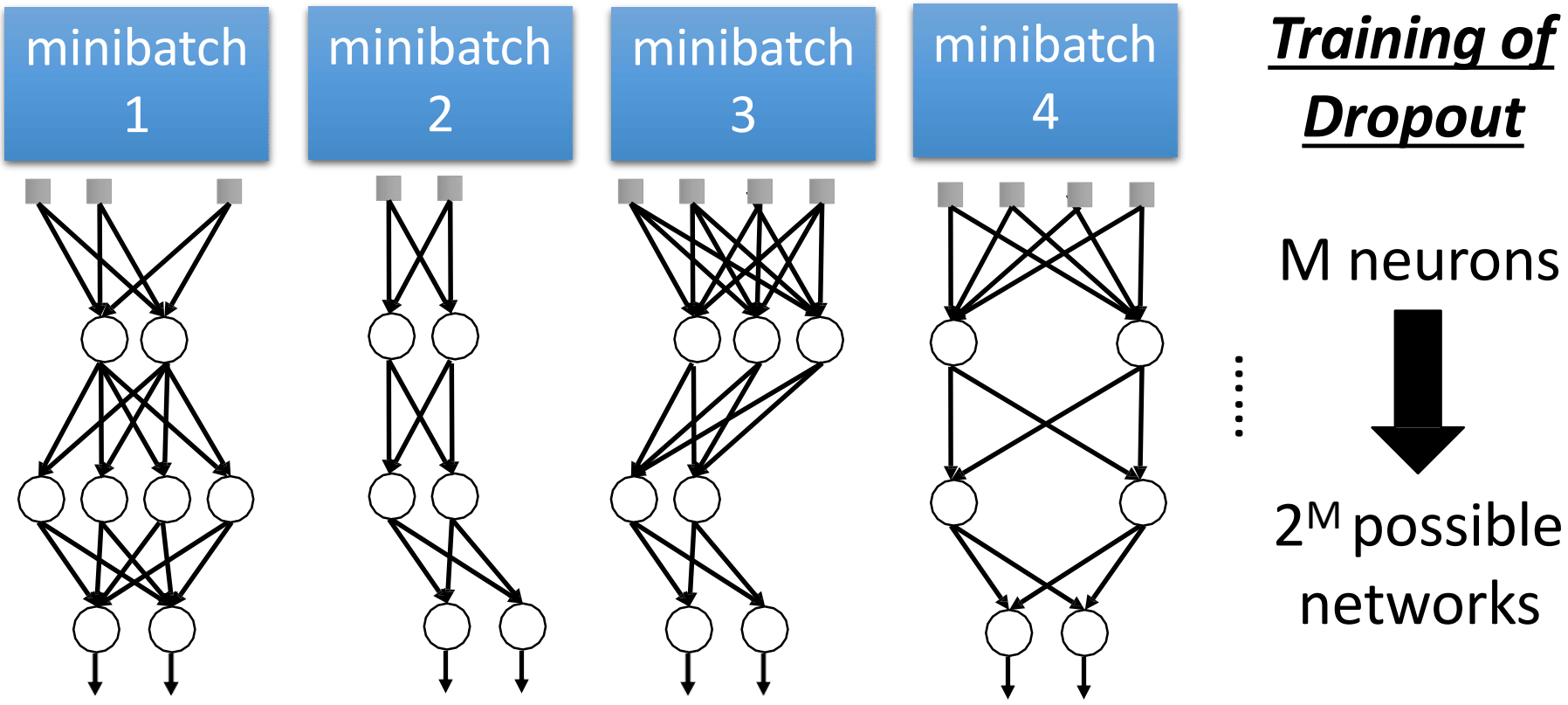
訓練一些具有不同結構的網路

Dropout 是集成方法的一種

集成



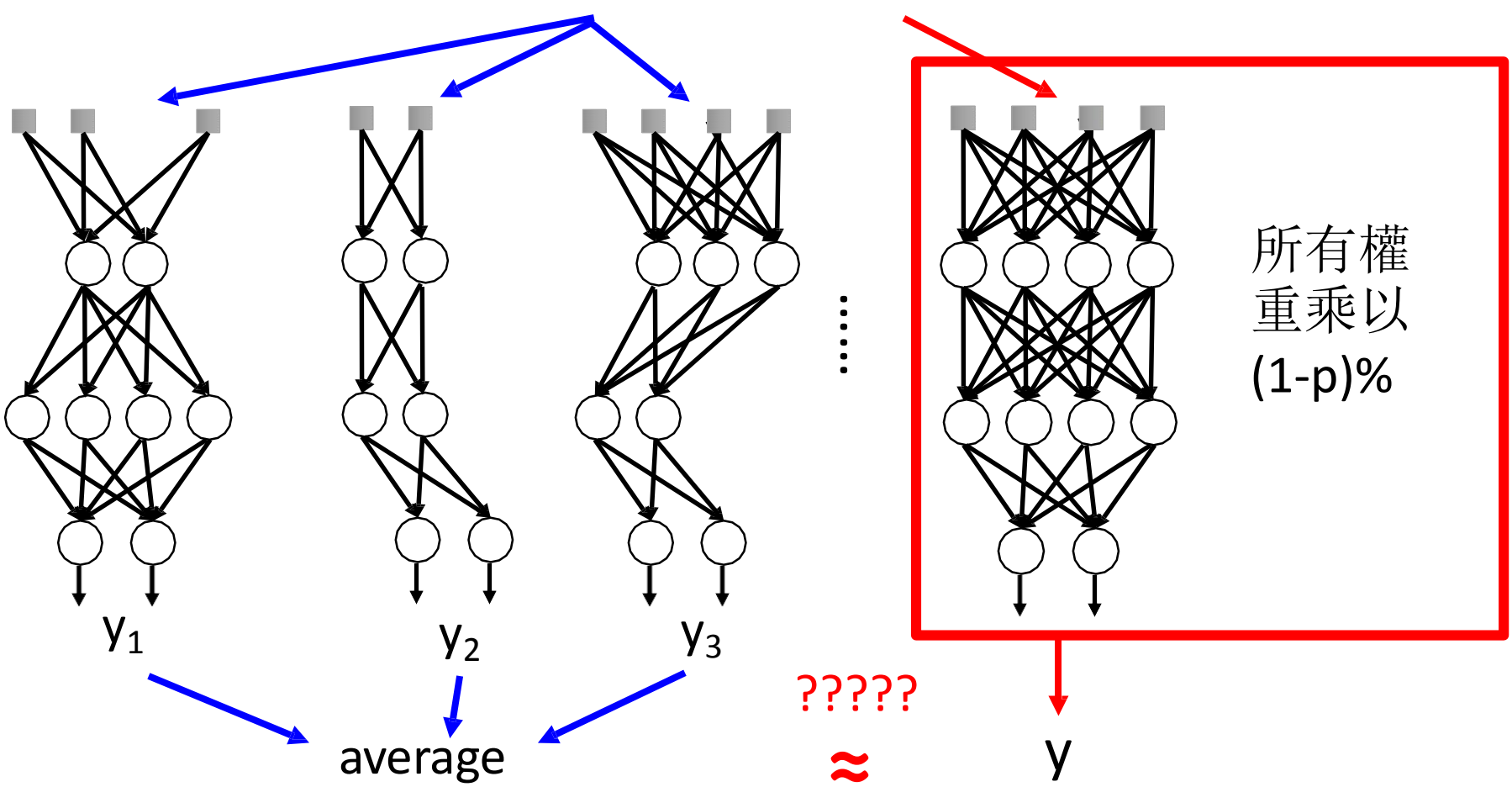
Dropout 是集成方法的一種



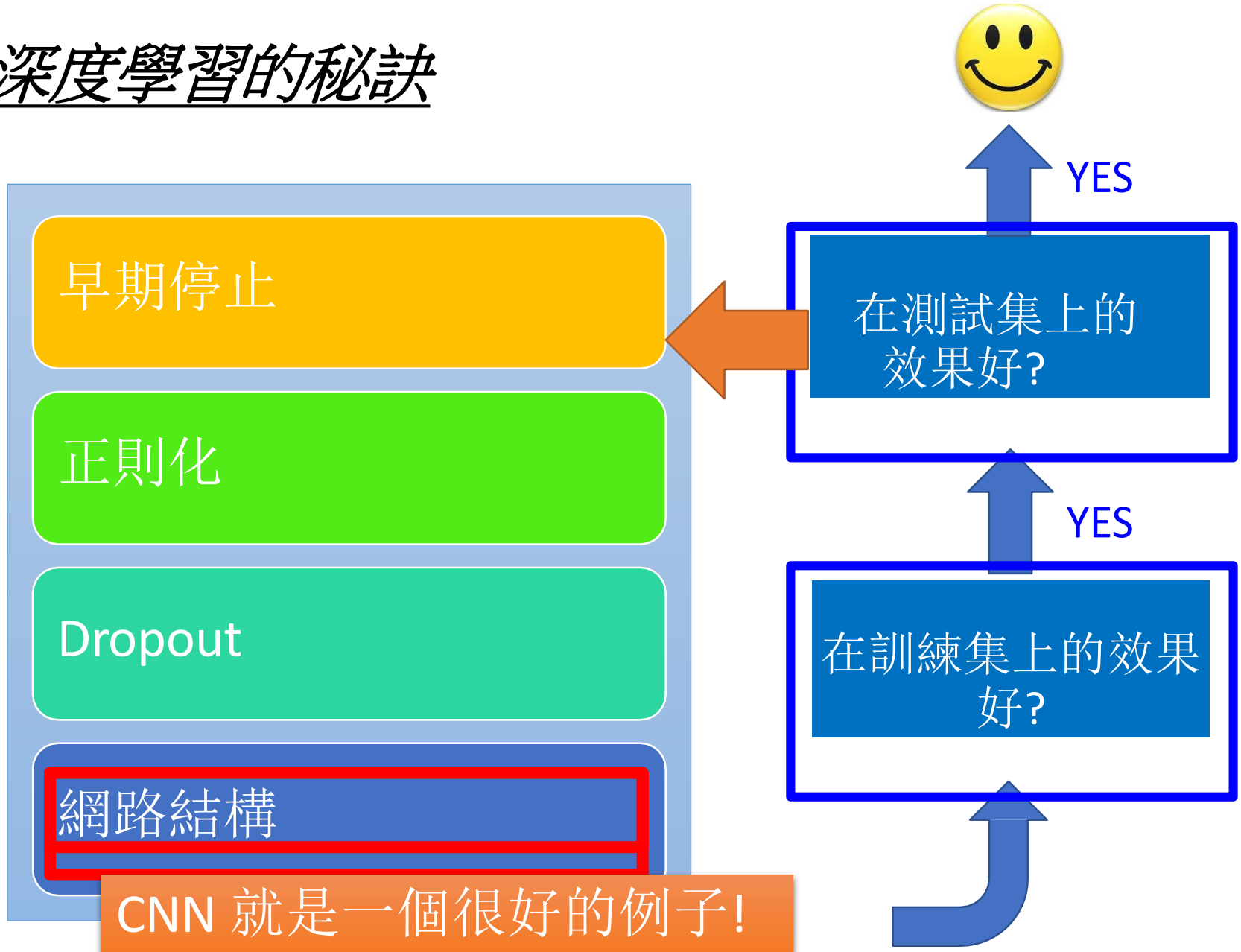
- 使用一個小批量資料來訓練一個網路
- 網路中的一些參數是可以共用的

Dropout 是集成方法的一種

Testing of Dropout



深度學習的秘訣


早期停止
正則化
Dropout
網路結構
CNN 就是一個很好的例子!
(下一節)
在測試集上的效果好?
YES
在訓練集上的效果好?
YES

Lecture III:
神經網路的變體

神經網路的變體

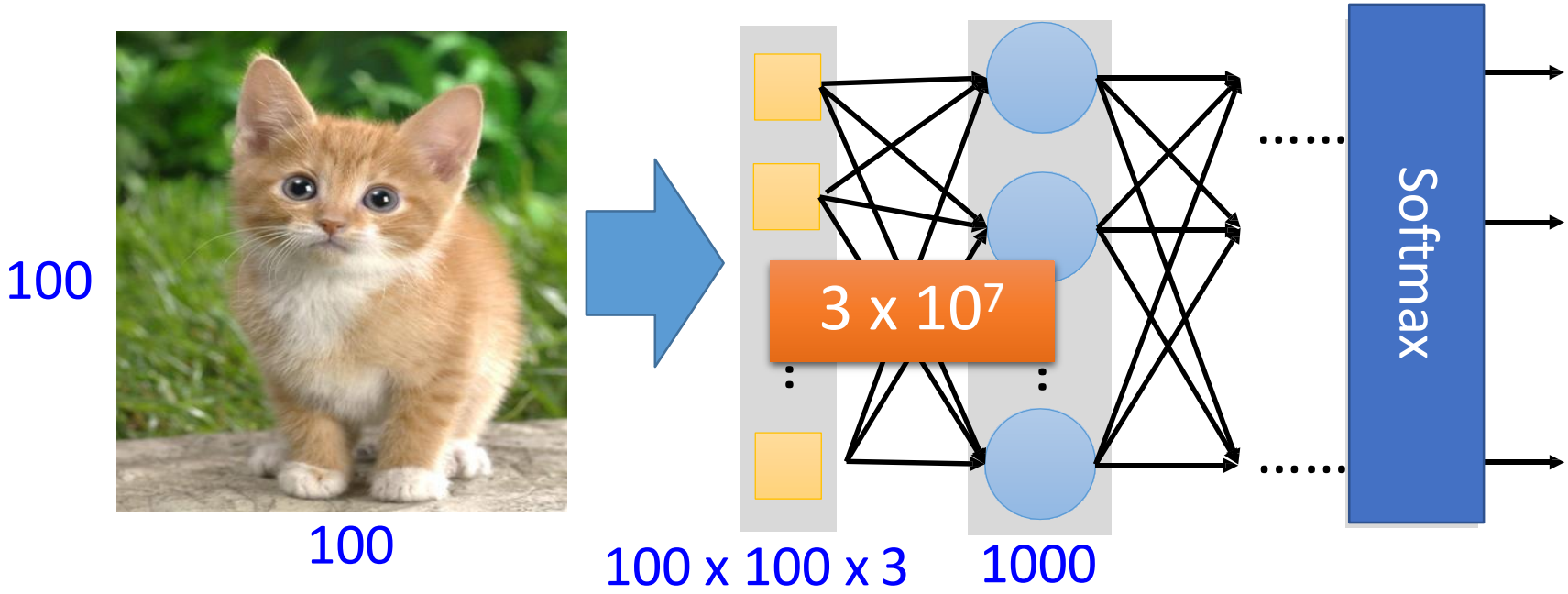
卷積神經網路(CNN)

廣泛用於
影像處理

迴圈神經網路(RNN)

為什麼CNN 用於圖像識別?

- 當處理圖像時，全連接網路的第一層將會非常大.....



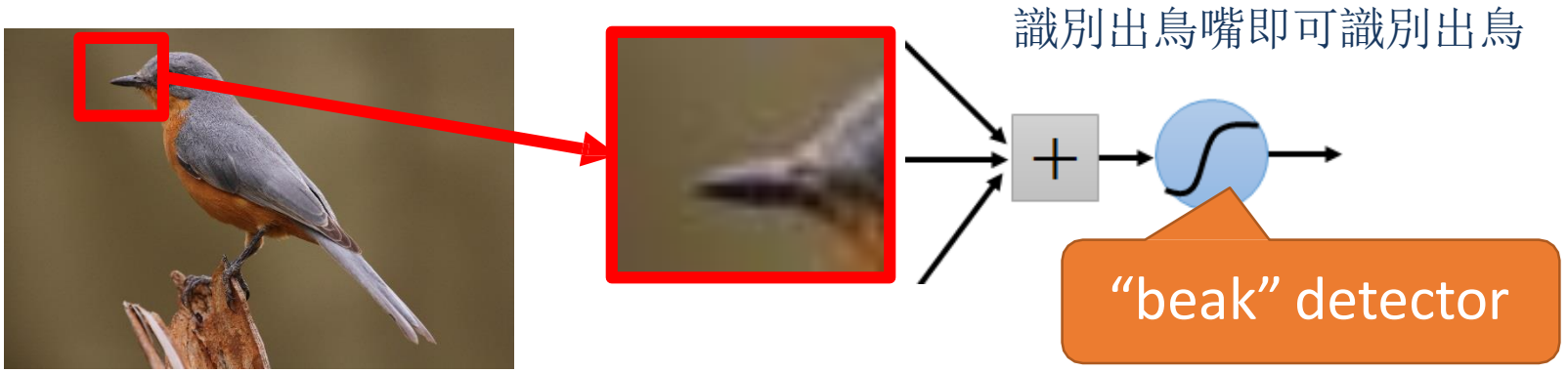
可以通過考慮圖像識別的性質來簡化全連接網路嗎?

為什麼CNN 用於圖像識別?

- 一些模式比整張圖片小得多

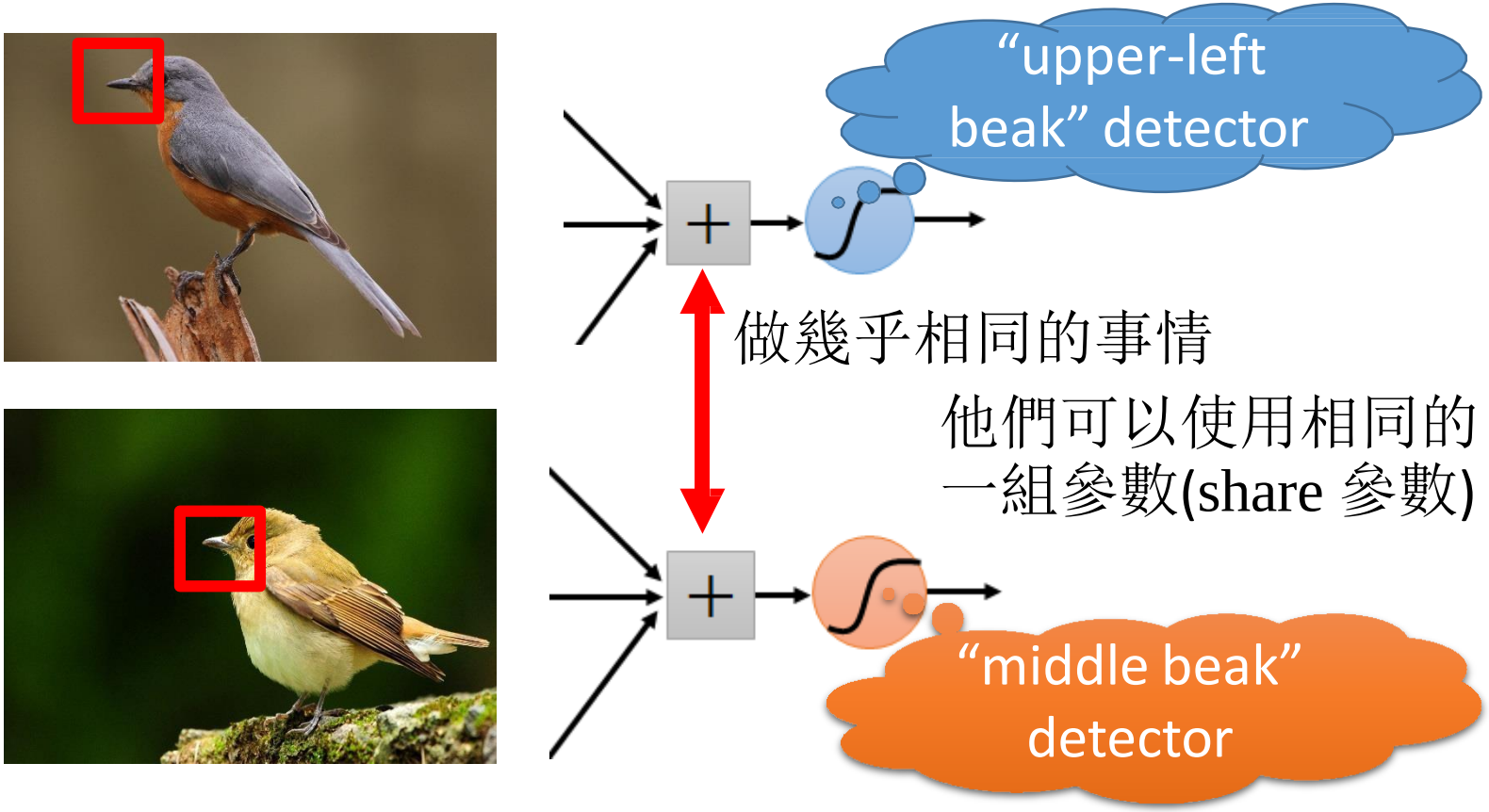
一個神經元不需要看到整個圖像去發現模式

通過較少的參數連接到小區域



為什麼CNN 用於圖像識別?

- 同樣的模式可能出現在圖像的不同區域



為什麼CNN 用於圖像識別?

- 對圖像進行二次採樣不會改變圖像中的物體

二次採樣

我們可以對圖像進行二次採樣以使圖像變小

更少的參數來處理圖像

The whole CNN

Convolution

Max Pooling

Convolution

Max Pooling

可以重複多次

Flatten

Fully Connected Feedforward network

cat dog ……

The whole CNN

屬性 1

- 一些模式比整張圖片小得多

屬性 2

- The同樣的模式可能出現在圖像的不同區域

屬性 3

- 對圖像的二次採樣不會改變圖像中的物體

Convolution

Max Pooling

Convolution

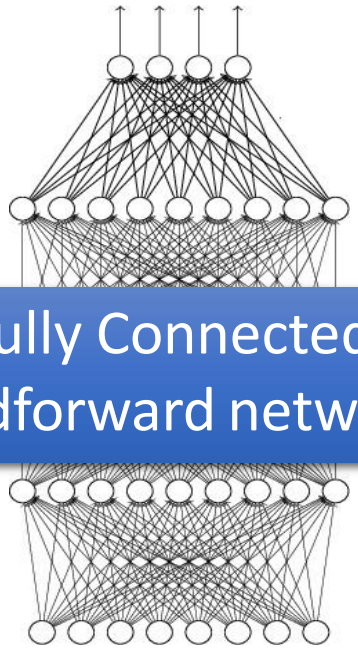
Max Pooling

可以重複多次

Flatten

The whole CNN

cat dog ……



Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

可以重複多次

Flatten

CNN – 卷積

這些都是要學習的參數

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

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

矩陣

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

矩陣

Property 1

每個 filter 探測一個小的模式(3 x 3).

CNN – 卷積

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

步長=1

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

3 -1

6 x 6 image

CNN – Convolution

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If 步長=2

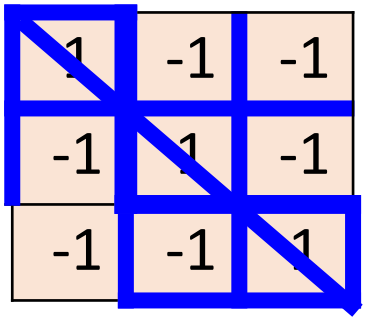
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

3 -3

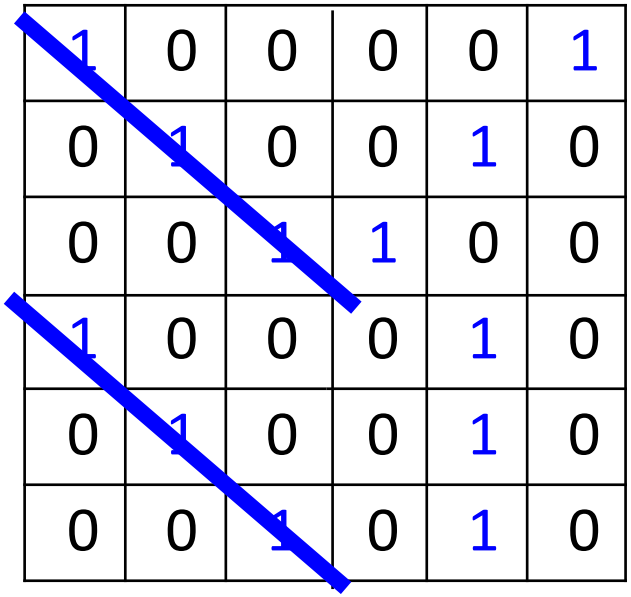
We set stride=1 below

CNN – Convolution

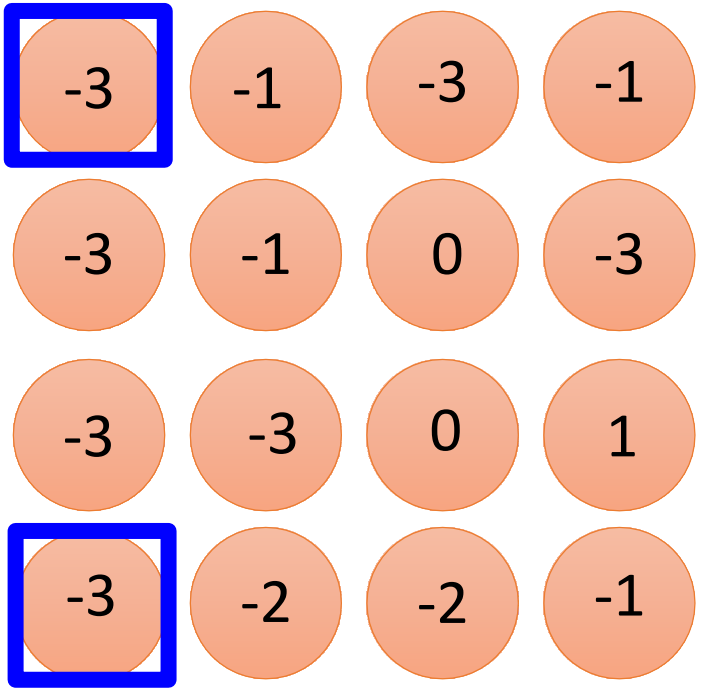


Filter 1

步長=1



6 x 6 image



Property 2

CNN – Convolution

-1	1	-1
-1	1	-1
-1	1	-1

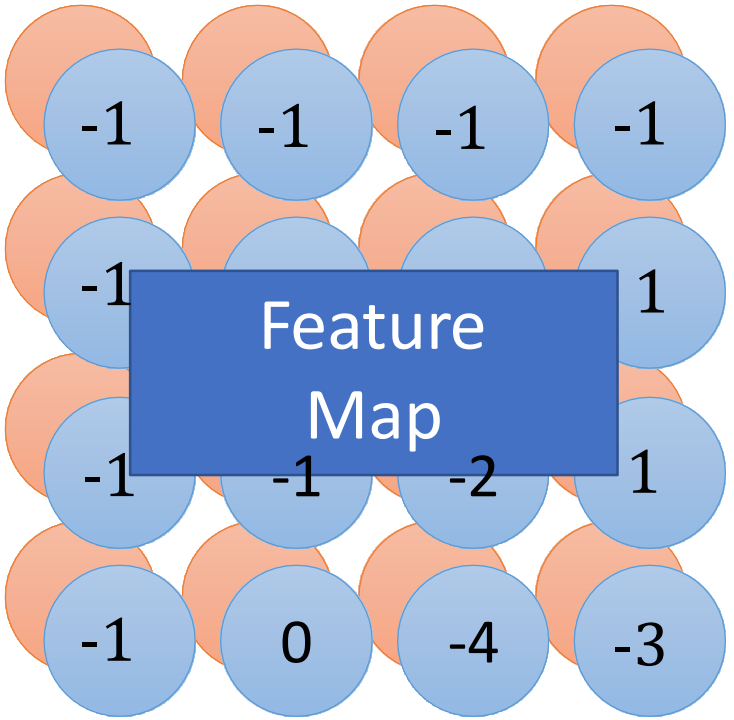
Filter 2

步長=1

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

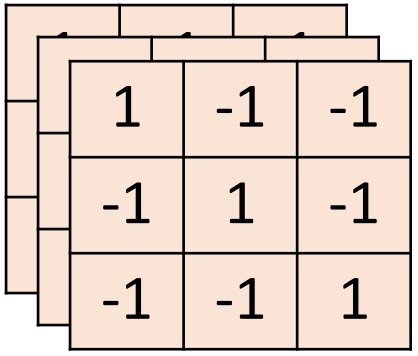
6 x 6 image

對於每一個 filter 做同樣的事

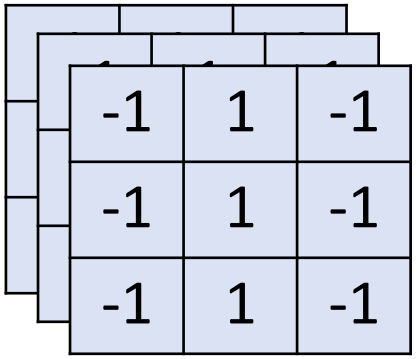


4 x 4 image

CNN – 彩色圖像

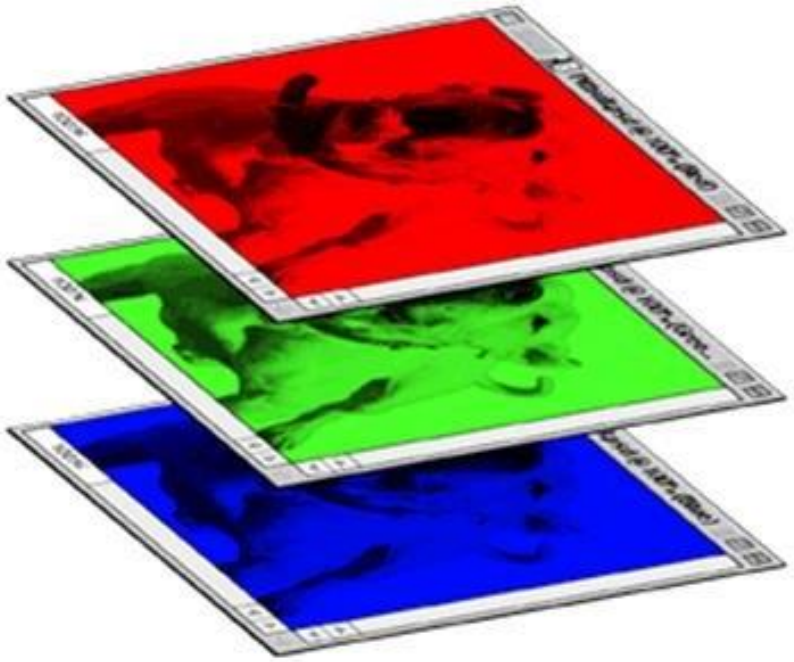


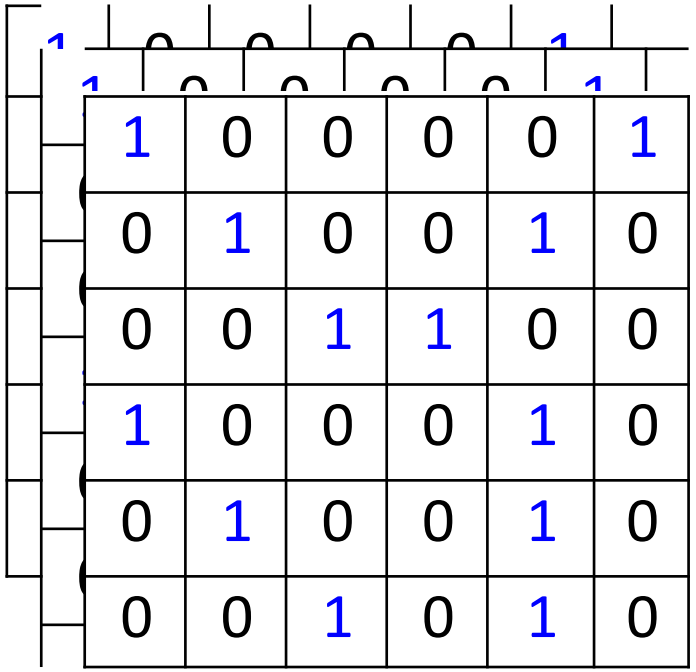
Filter 1



Filter 2

彩色圖像





The whole CNN

Convolution

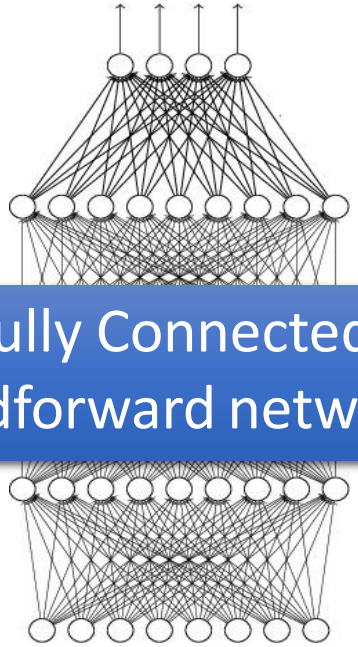
Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten



Fully Connected Feedforward network

cat dog ……

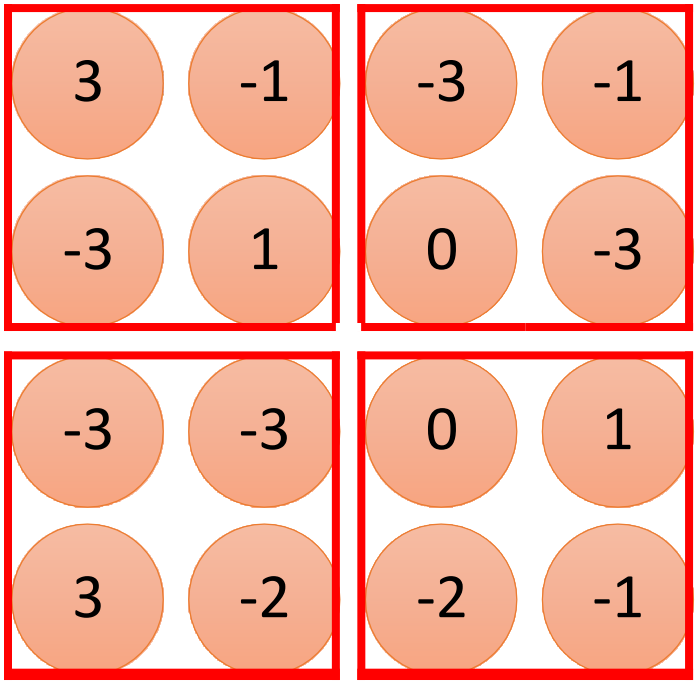
CNN – Max Pooling

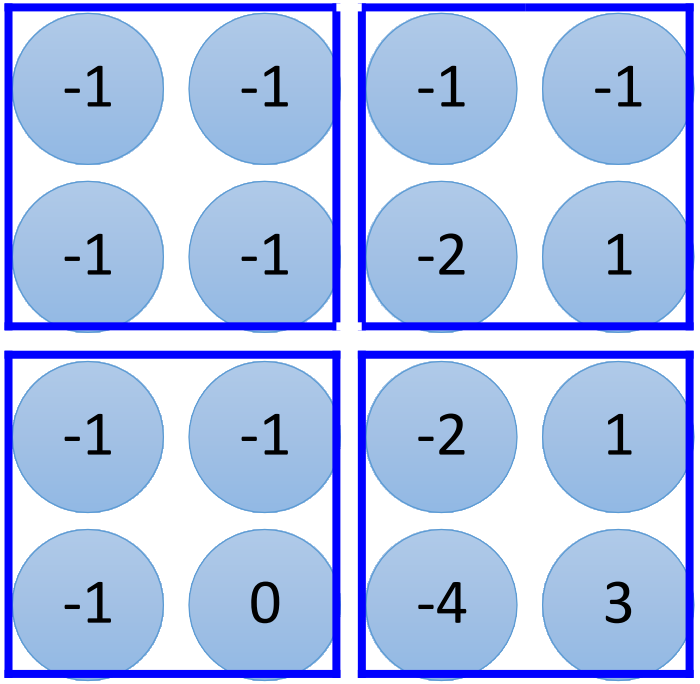
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

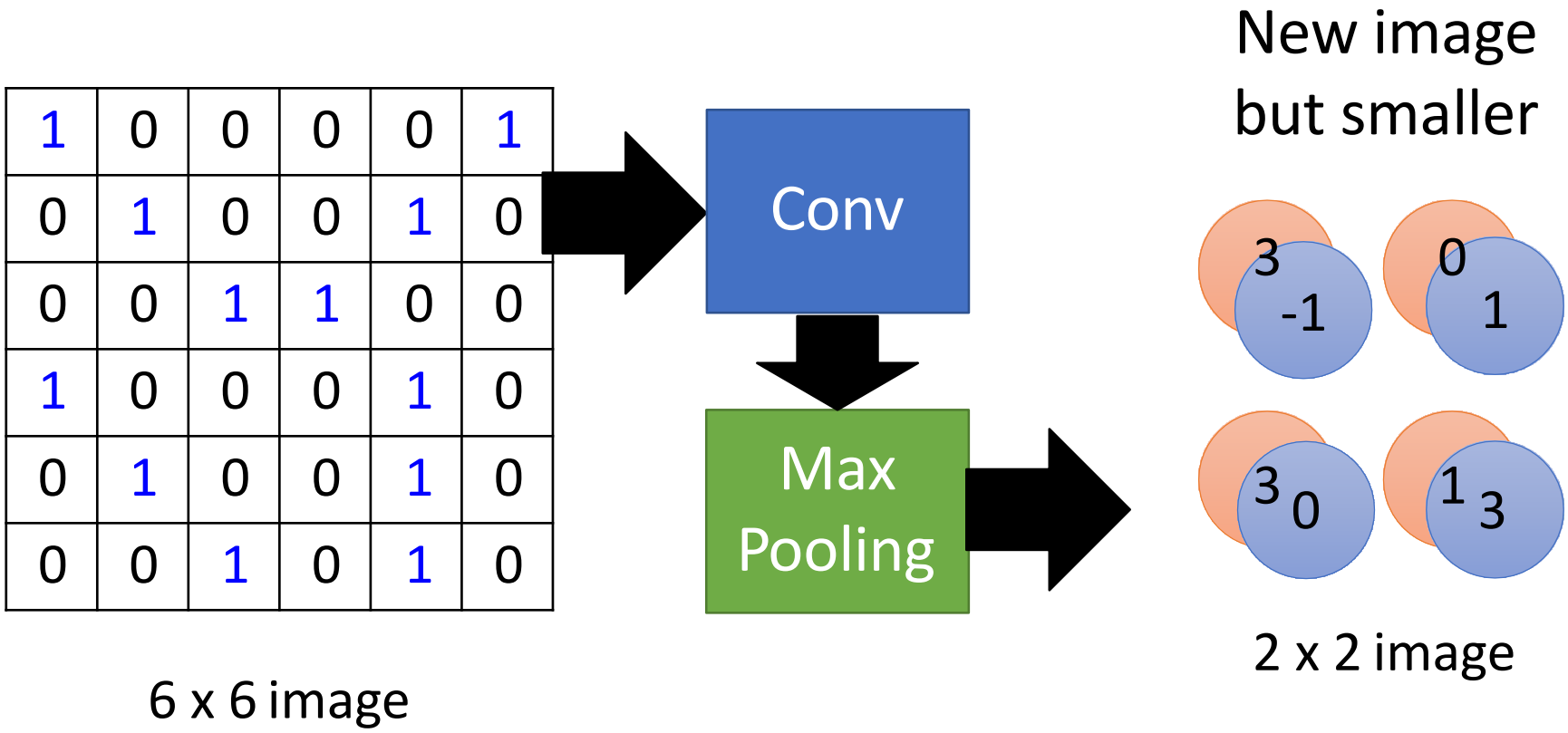
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

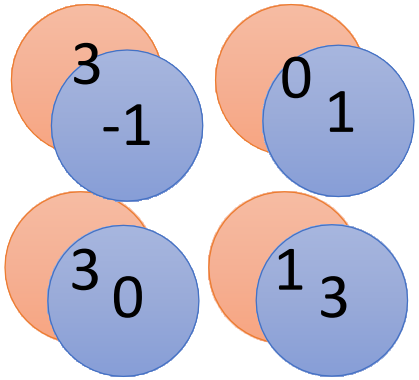




CNN – Max Pooling



The whole CNN



新的圖像

Convolution

Max Pooling

Convolution

Max Pooling

重複多次

比原始圖像更小

The whole CNN

cat dog ……

Convolution

Max Pooling

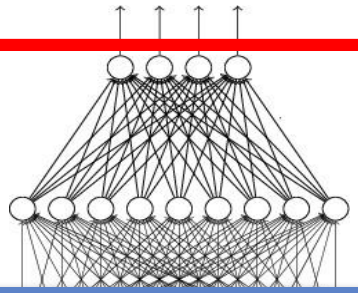
A new image

Convolution

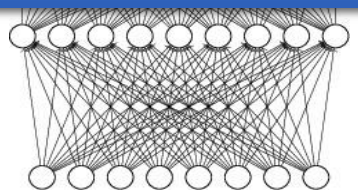
Max Pooling

A new image

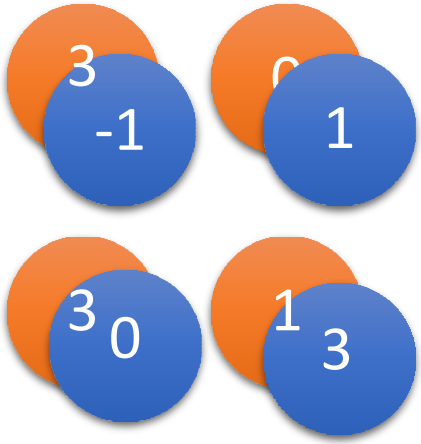
Flatten



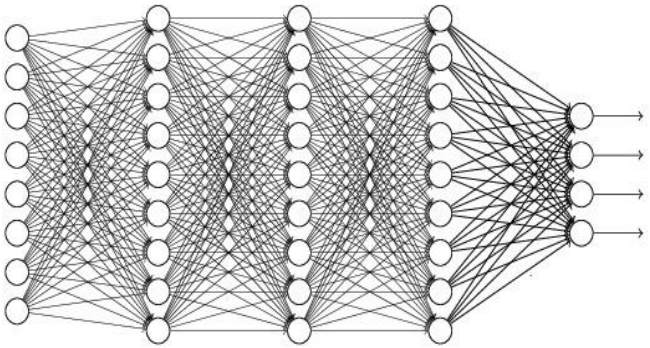
Fully Connected Feedforward network



Flatten

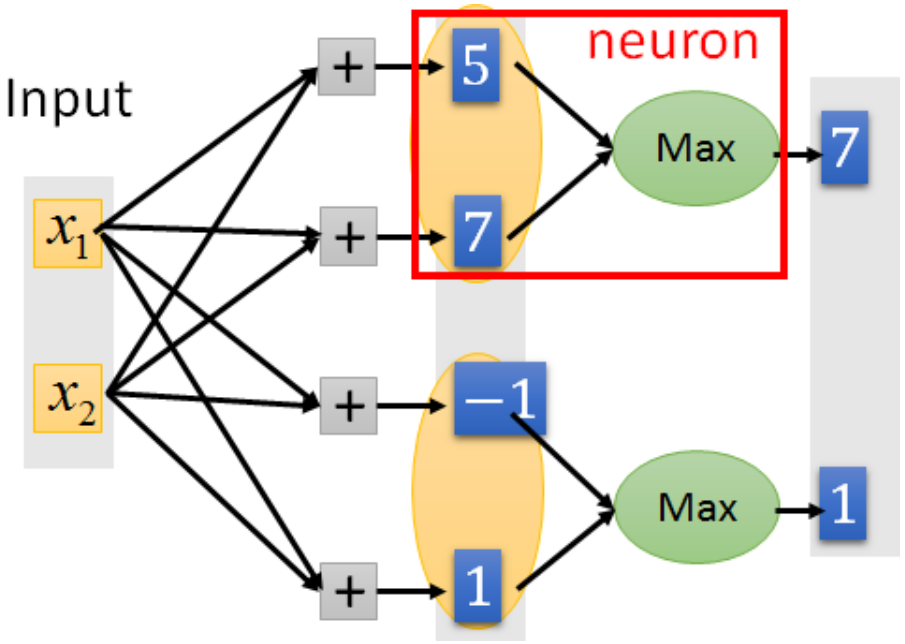


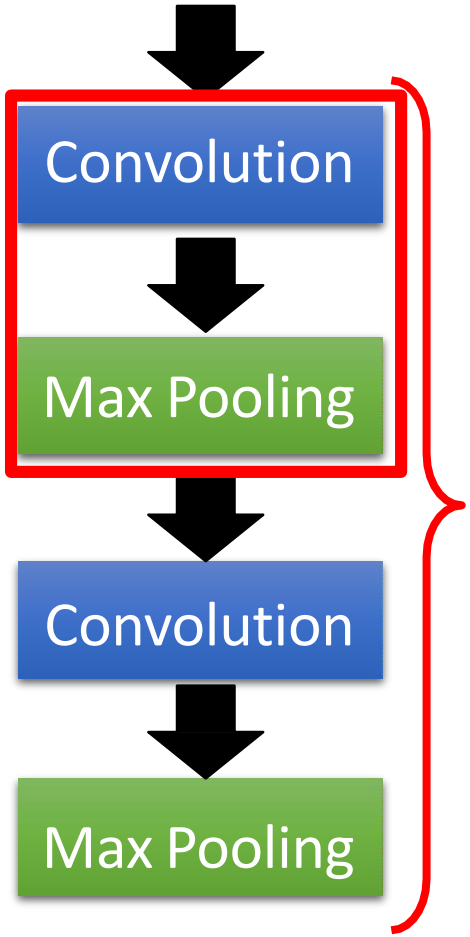
Flatten

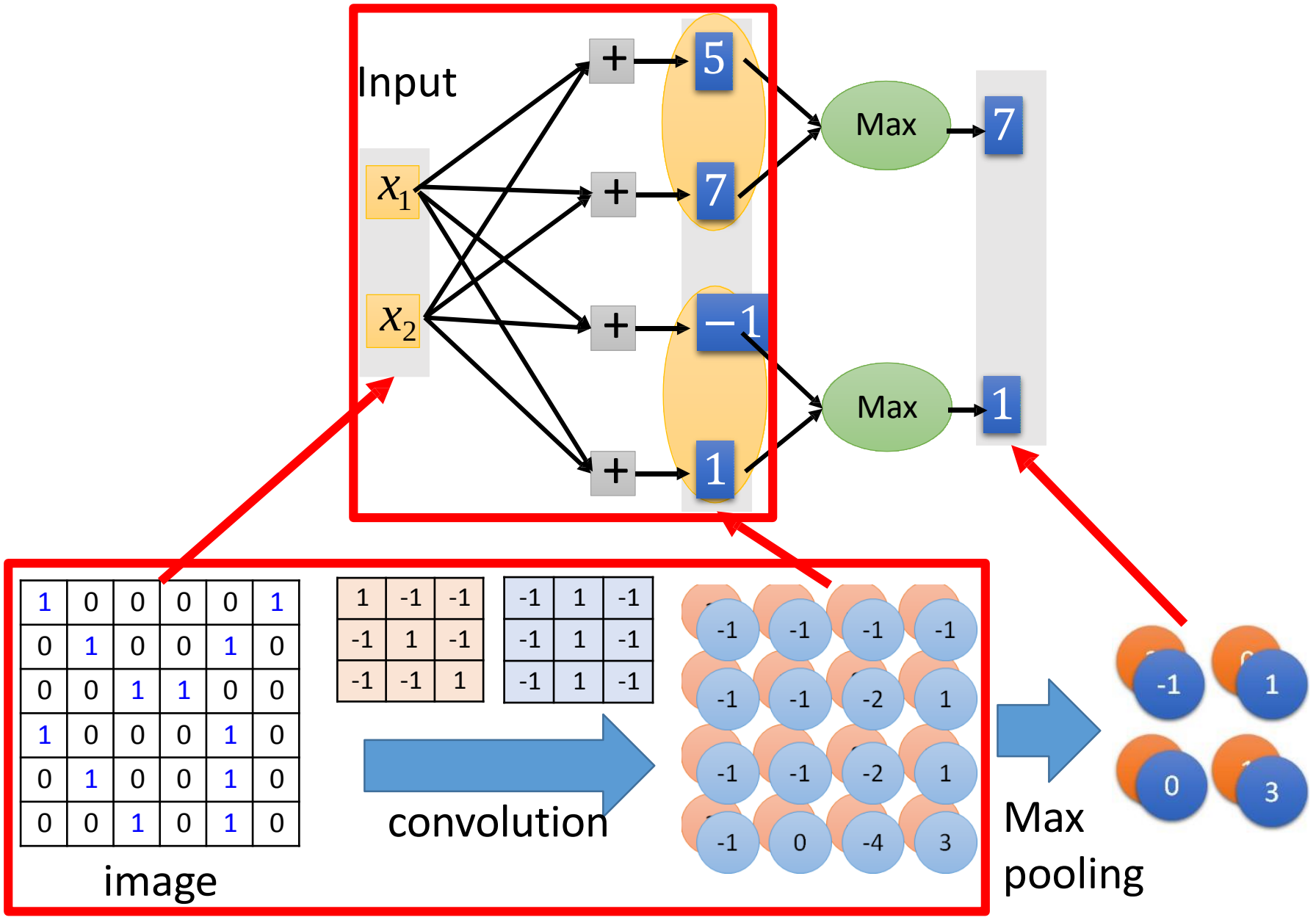


Fully Connected
Feedforward network

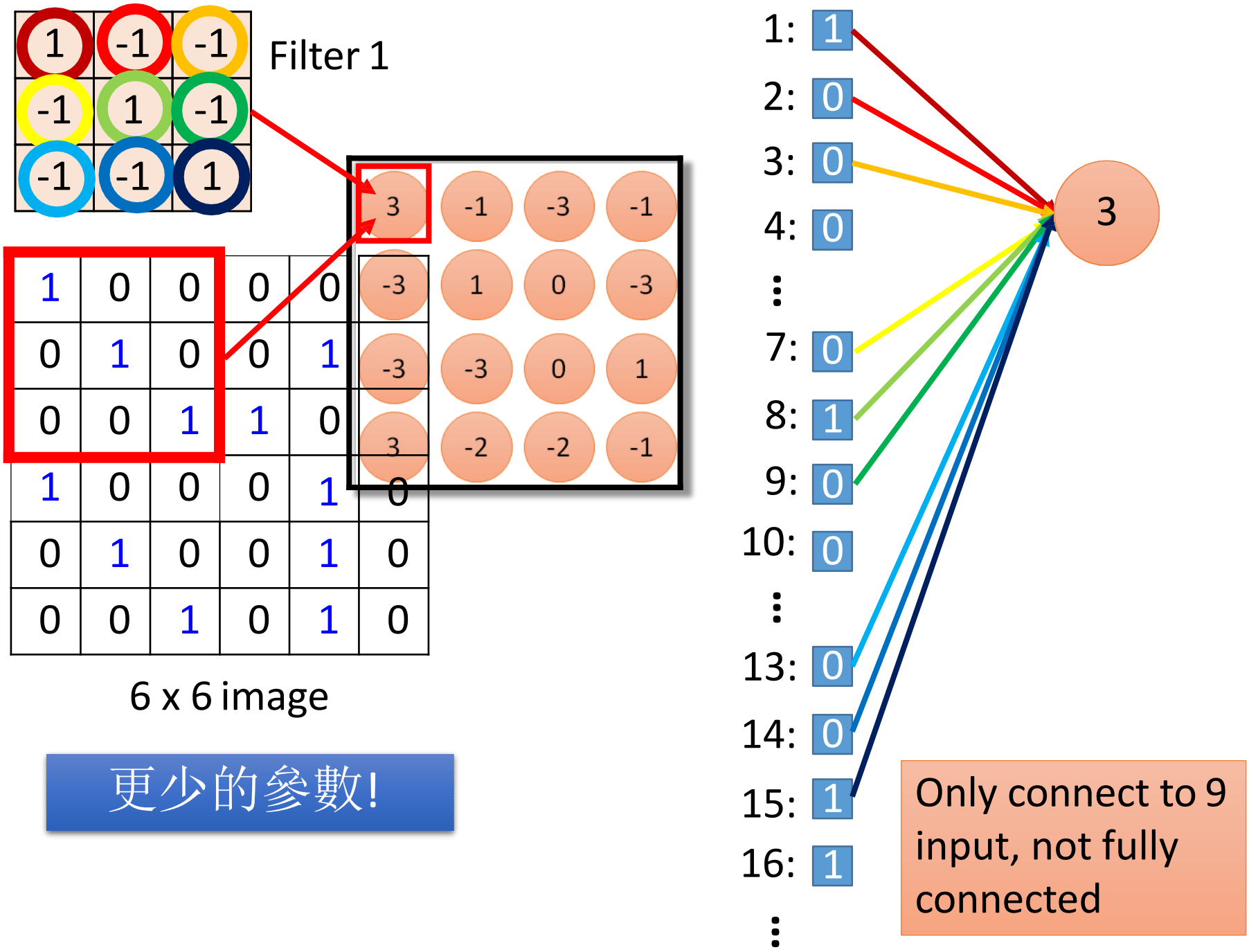
The whole CNN

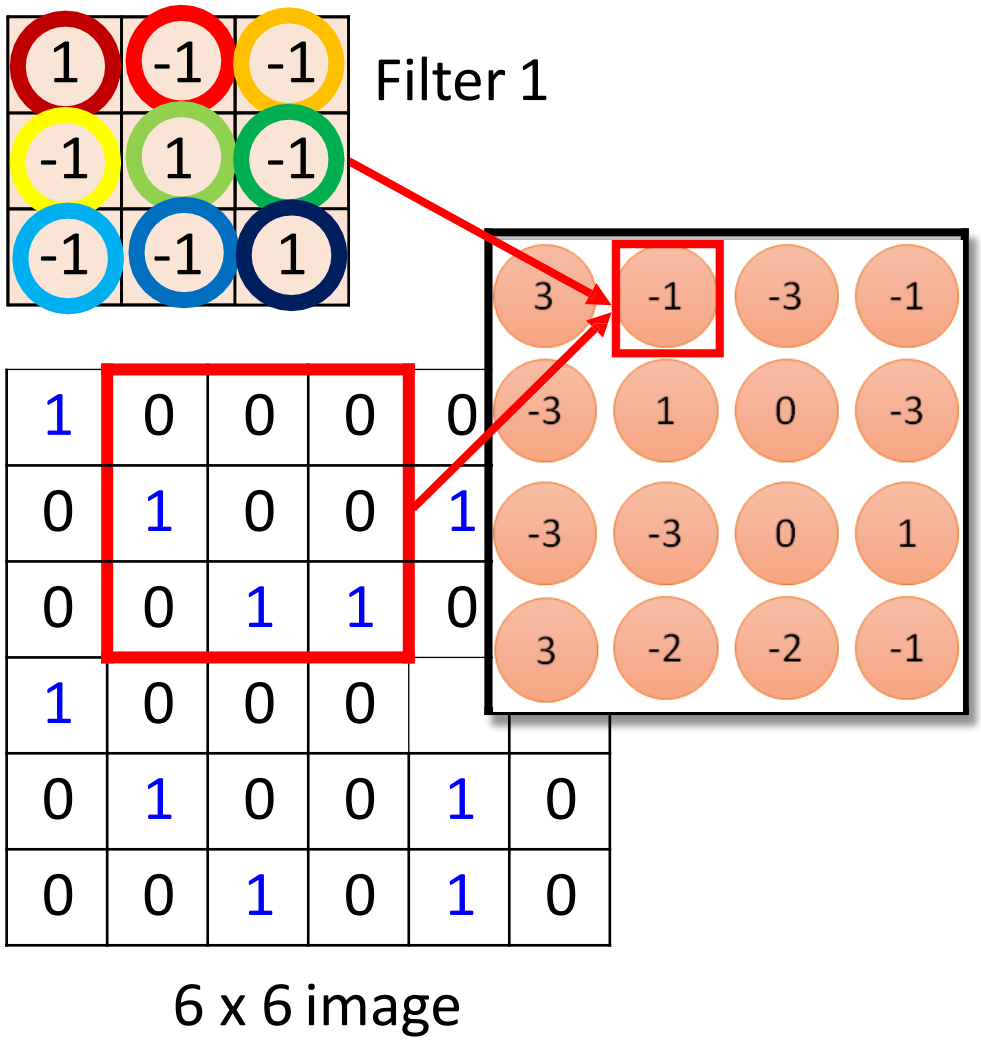




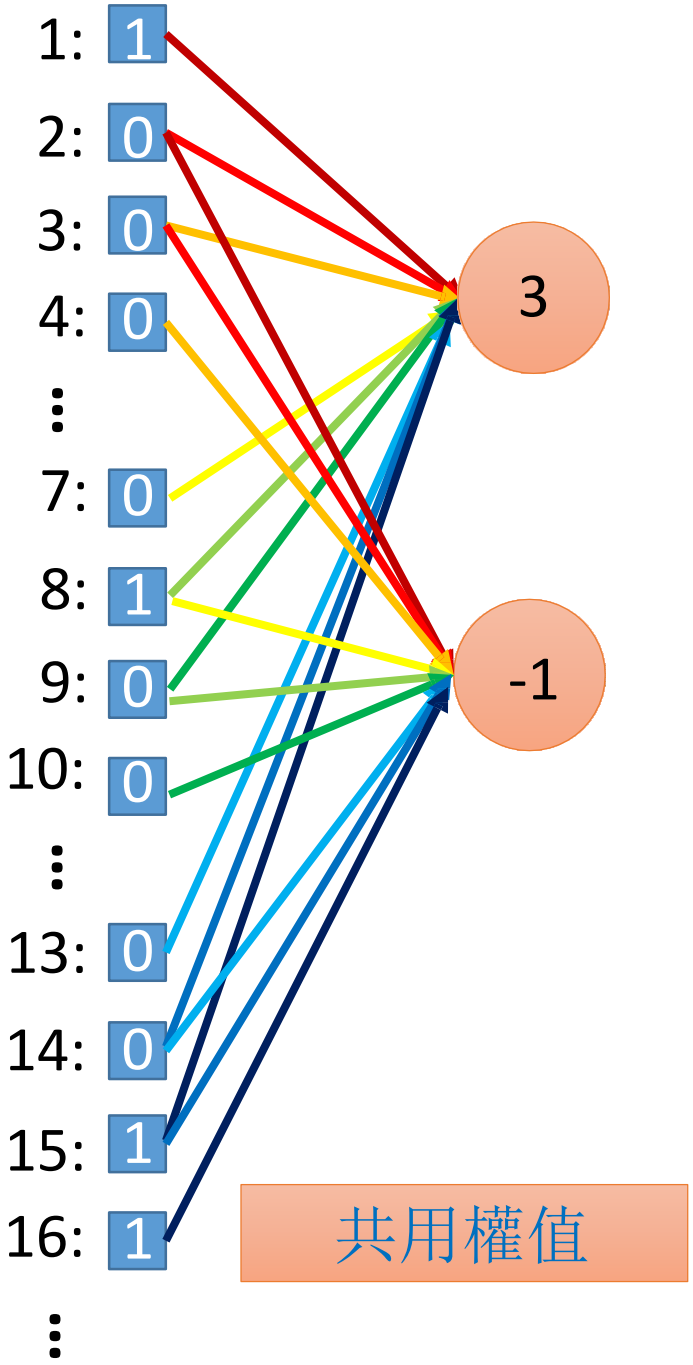


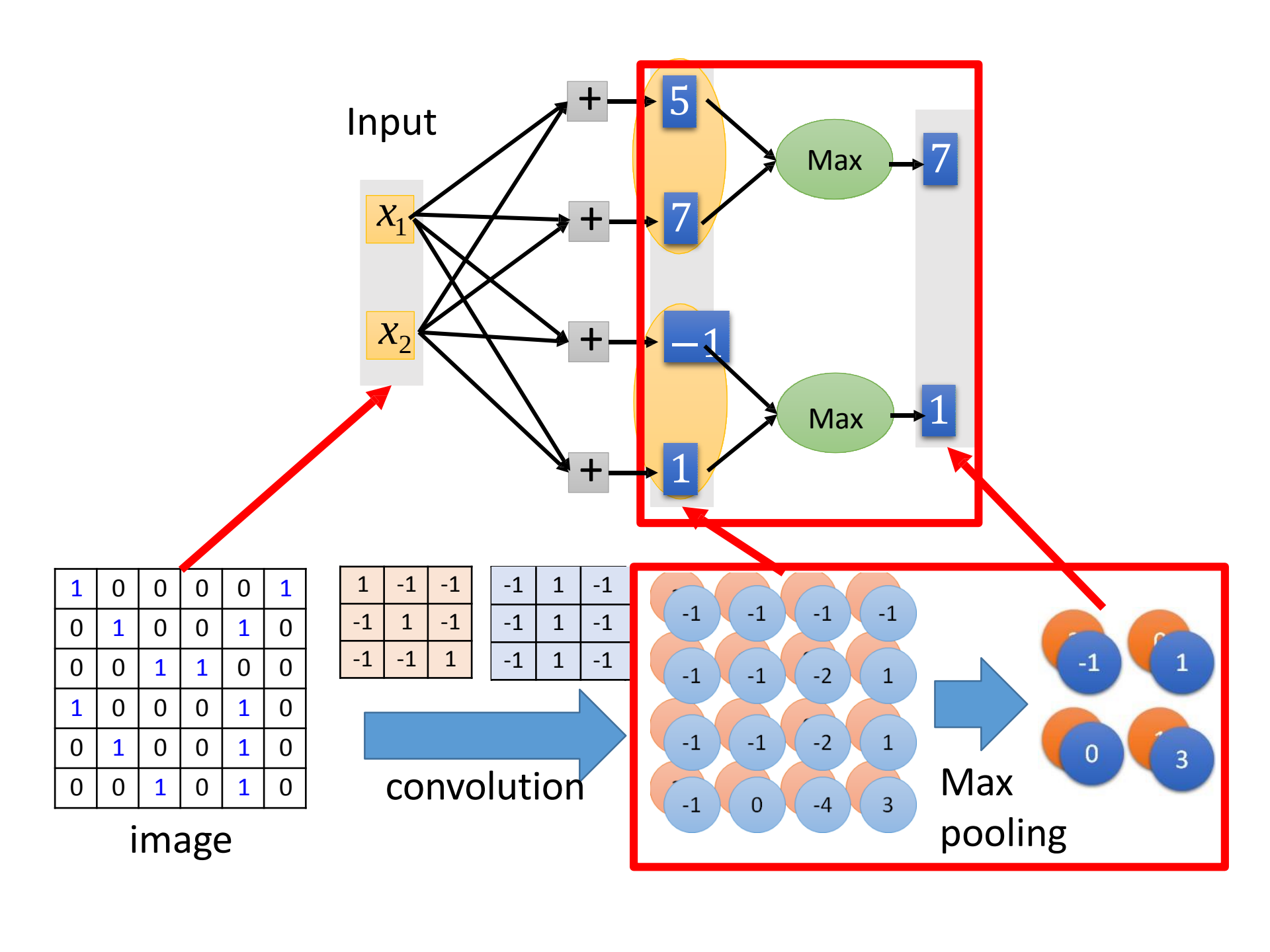
(忽略了卷積後的非線性啟動函數)

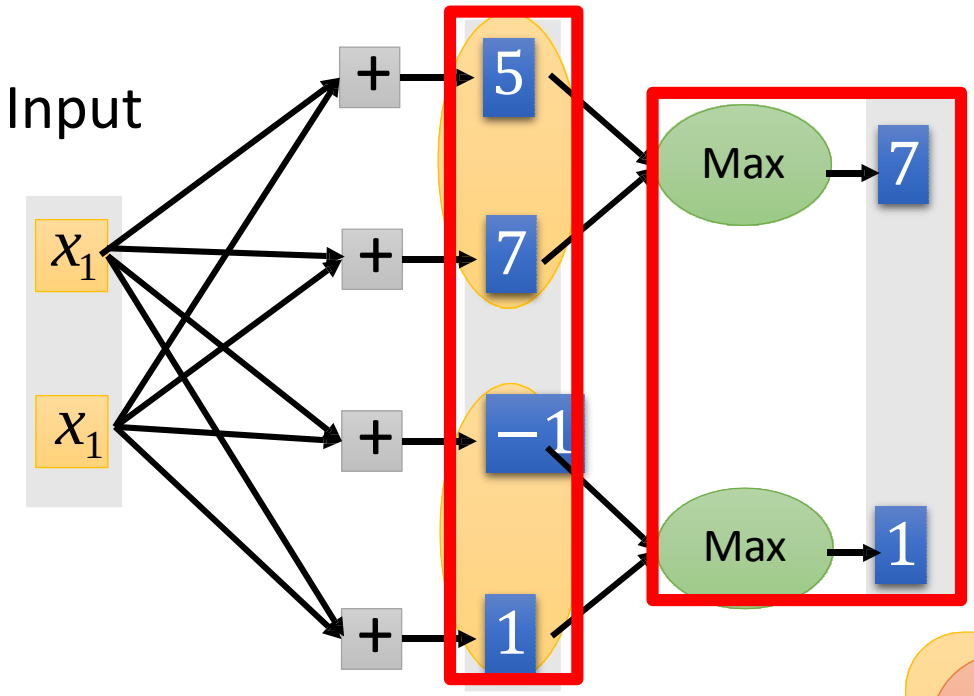

1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0 0
0 1 0 0 1 0
0 0 1 1 0 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image
1:
2:
3:
4:
7:
8:
9:
10:
13:
14:
15:
16:
3
更少的參數!
Only connect to 9 input, not fully connected

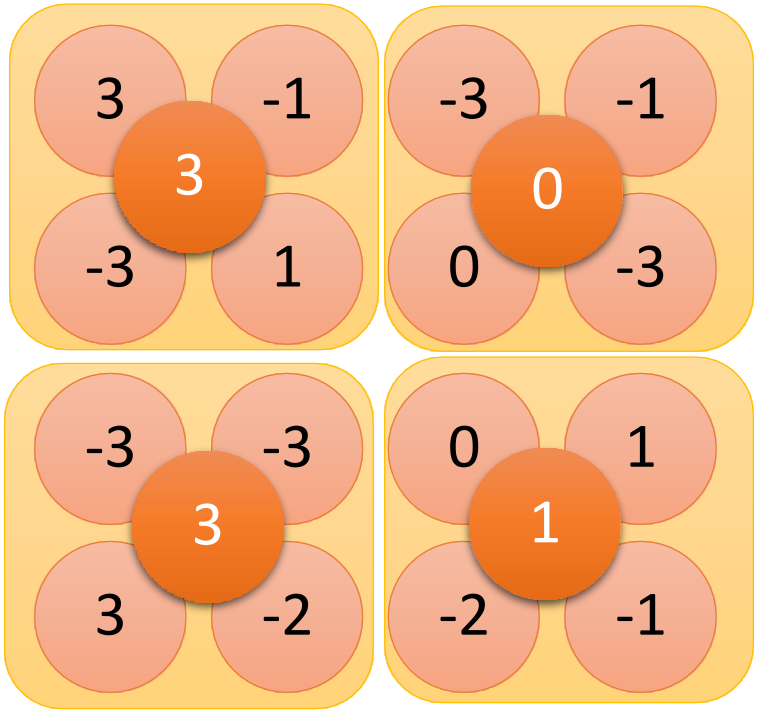

1 -1 -1
-1 1 -1
-1 -1 1
Filter 1
3 -1 -3 -1
-3 1 0 -3
-3 -3 0 1
3 -2 -2 -1
1 0 0 0 0
0 1 0 0 1
0 0 1 1 0
1 0 0 0
0 1 0 0 1 0
0 0 1 0 1 0
6 x 6 image

更少的參數!


1:
2:
3:
4:
7:
8:
9:
10:
13:
14:
15:
16:
1 0 0 0 0 1 0 0 1 0 0 1
3
-1
共用權值


Input
x_1
x_2
+
+
+
+
5
7
−1
1
Max
Max
7
1
1 0 0 0 0 1
0 1 0 0 1 0
0 0 1 1 0 0
1 0 0 0 1 0
0 1 0 0 1 0
0 0 1 0 1 0
image
1 -1 -1
-1 1 -1
-1 -1 1
-1 1 -1
-1 1 -1
-1 1 -1
convolution
-1 -1 -1 -1
-1 -1 -2 1
-1 -1 -2 1
-1 0 -4 3
Max
pooling
-1 1
0 3


Input
x_1
x_1
+
+
+
+
5
7
−1
1
Max
Max
7
1


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

神經網路的變體

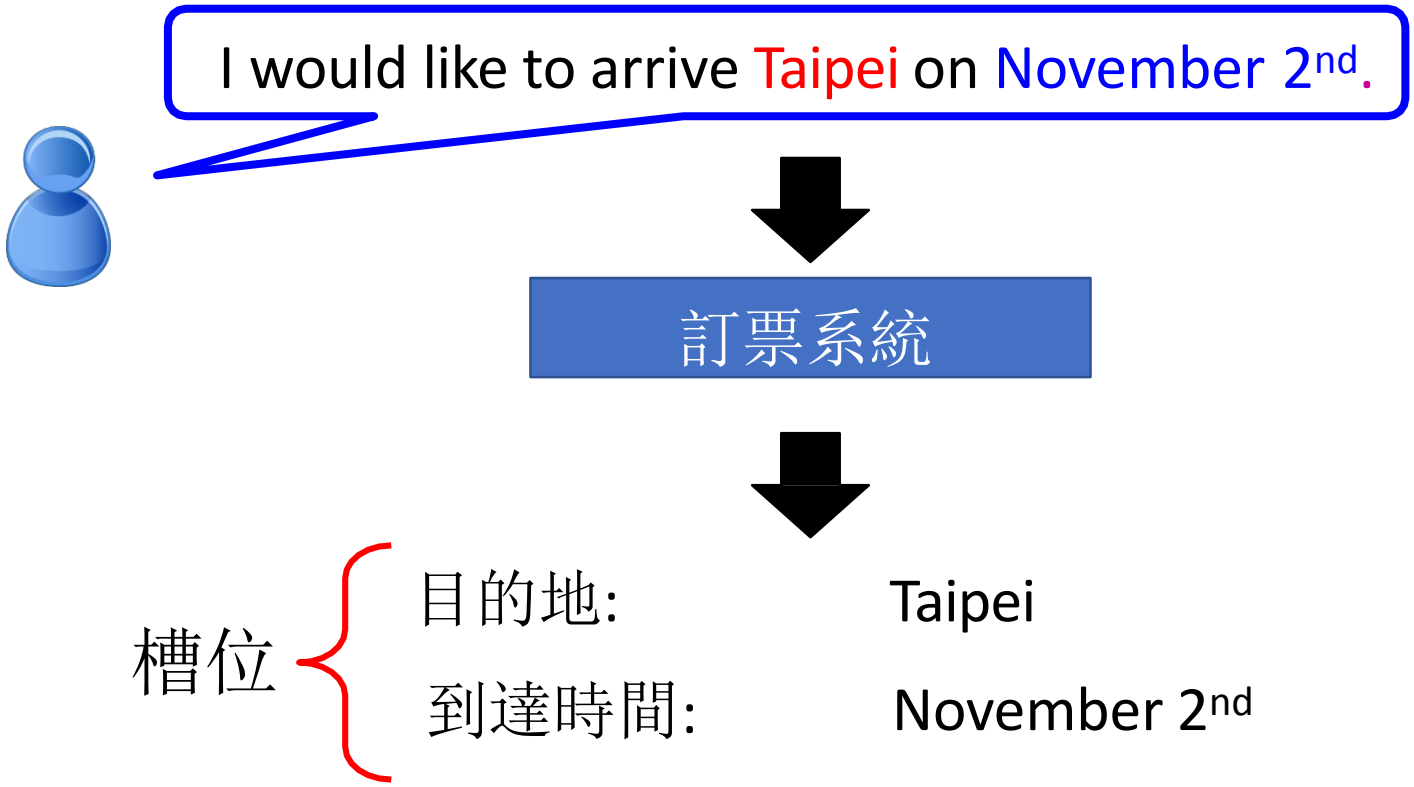
卷積神經網路 (CNN)

迴圈神經網路 (RNN)

具有記憶的神經網路

示例應用程式

- 插槽填充

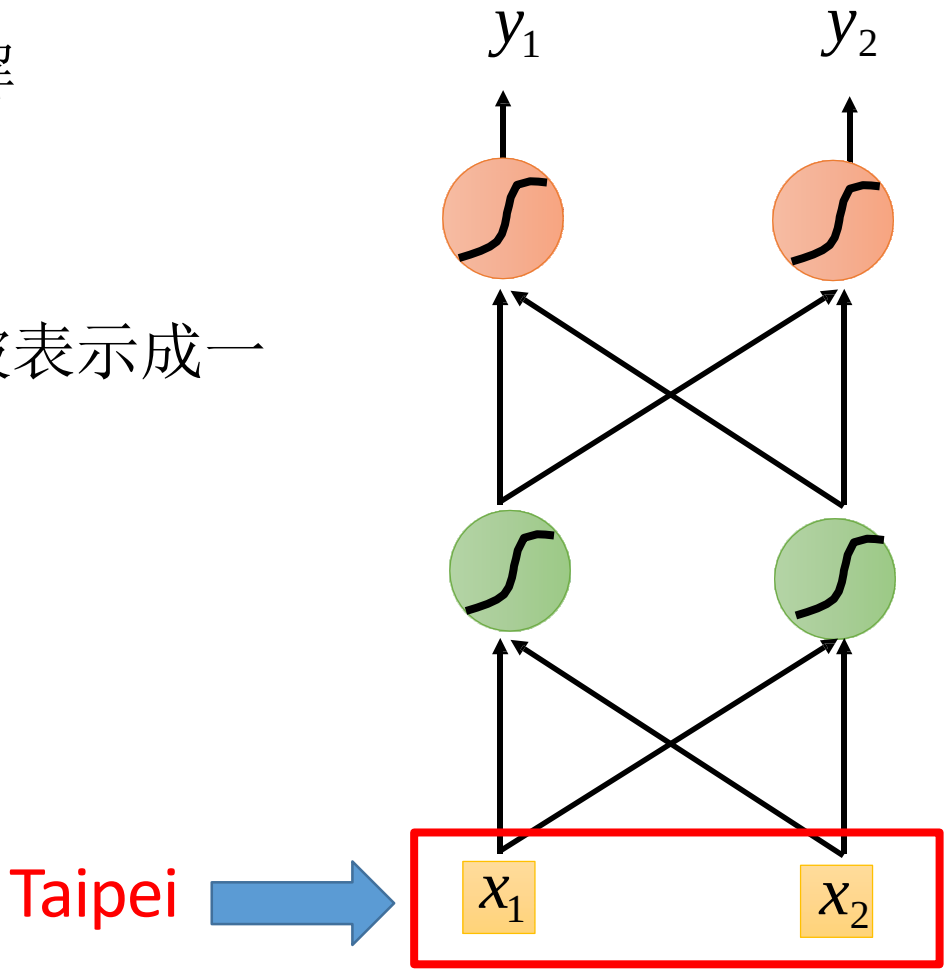

I would like to arrive Taipei on November 2nd.
訂票系統
槽位
目的地:
Taipei
到達時間:
November 2nd

示例應用程式

通過前饋神經網路解決插槽填充問題?

輸入: a word

(每個單詞都被表示成一個向量)



1-of-N encoding

如何將每個單詞表示成一個詞向量?

1-of-N Encoding 詞典 = {apple, bag, cat, dog, elephant}

向量是詞典大小

每個維度對應於詞典中的一個單詞

這個詞的維數是1，其他的是0

apple = [1 0 0 0 0]

bag = [0 1 0 0 0]

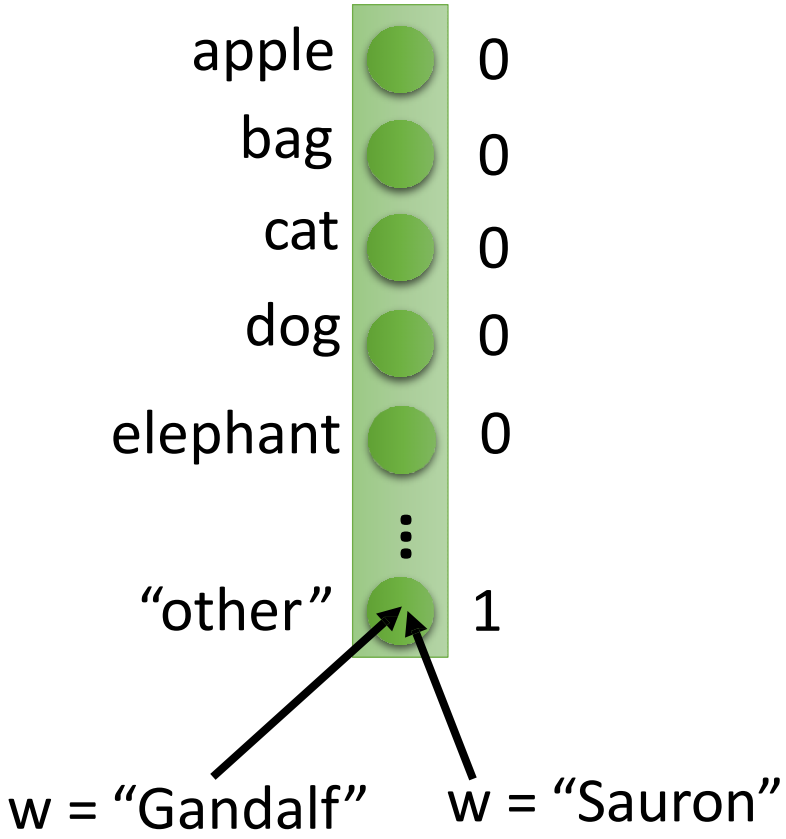
cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

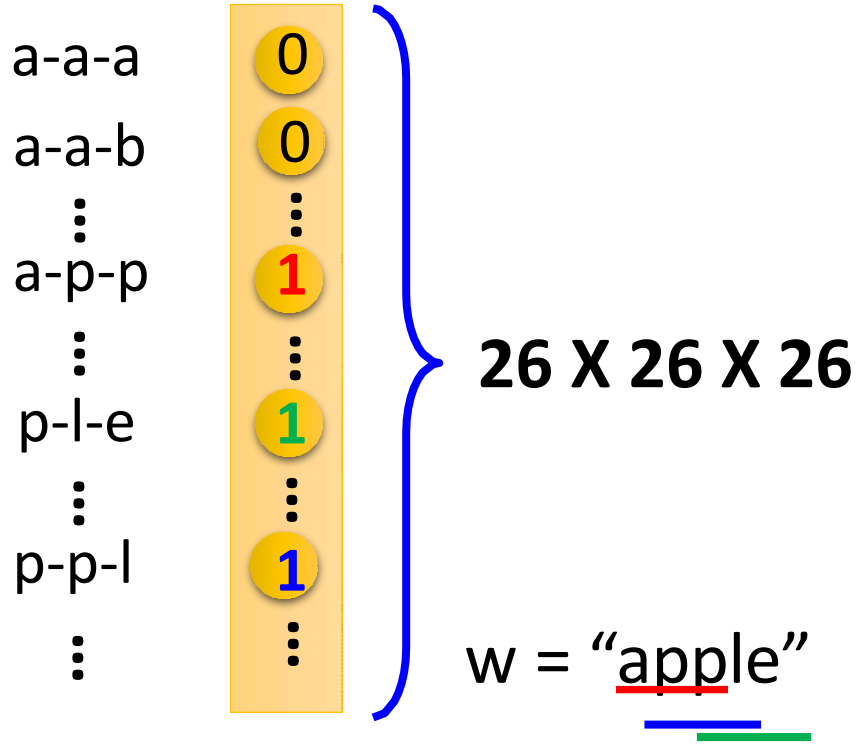
elephant = [0 0 0 0 1]

Beyond 1-of-N encoding

Dimension for "Other"



Word hashing



示例應用程式

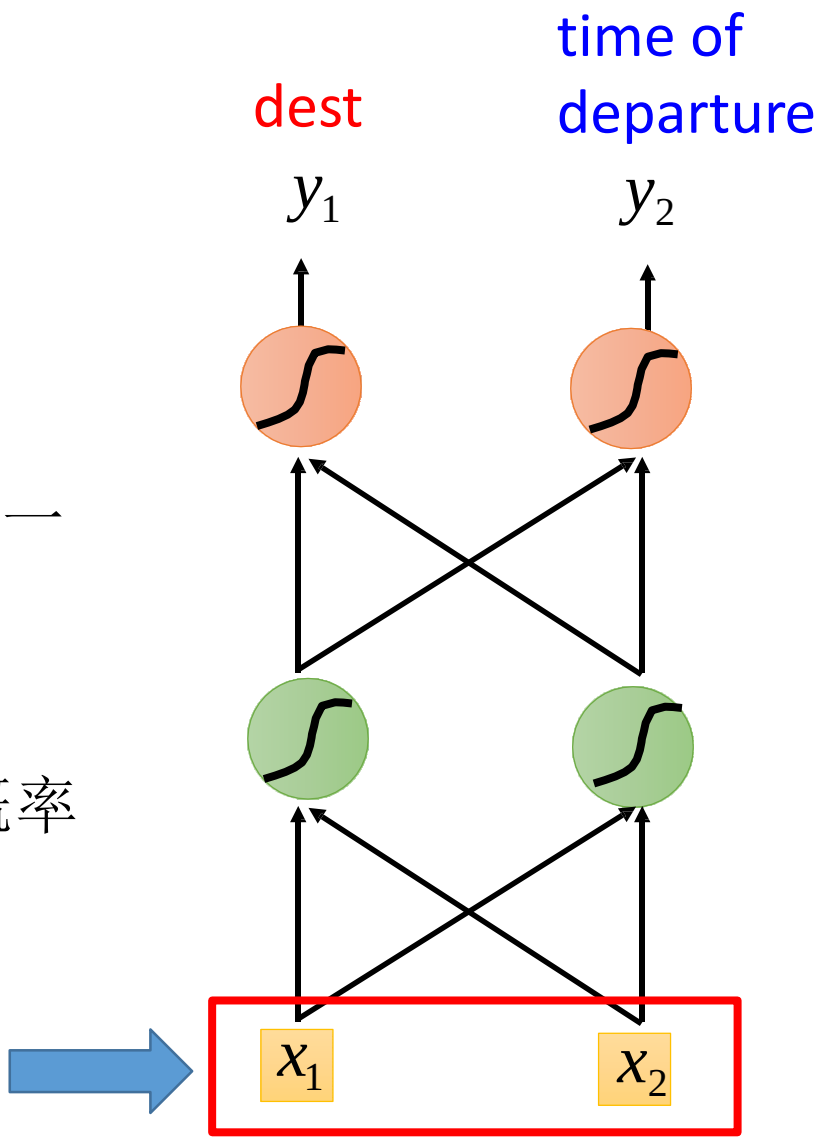
通過前饋神經網路解決插槽填充問題?

輸入: a word

(每個單詞都被表示成一個向量)

輸出:

輸入單詞屬於插槽的概率分佈



示例應用程式

arrive Taipei on November 2nd

other dest other time time

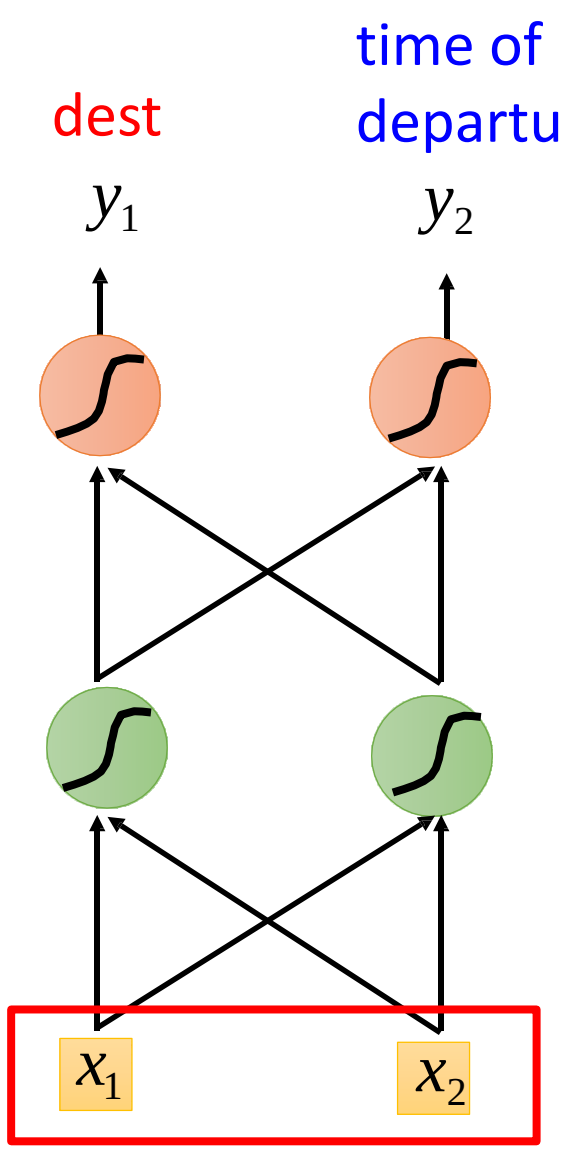
Problem?

leave Taipei on November 2nd

place of departure

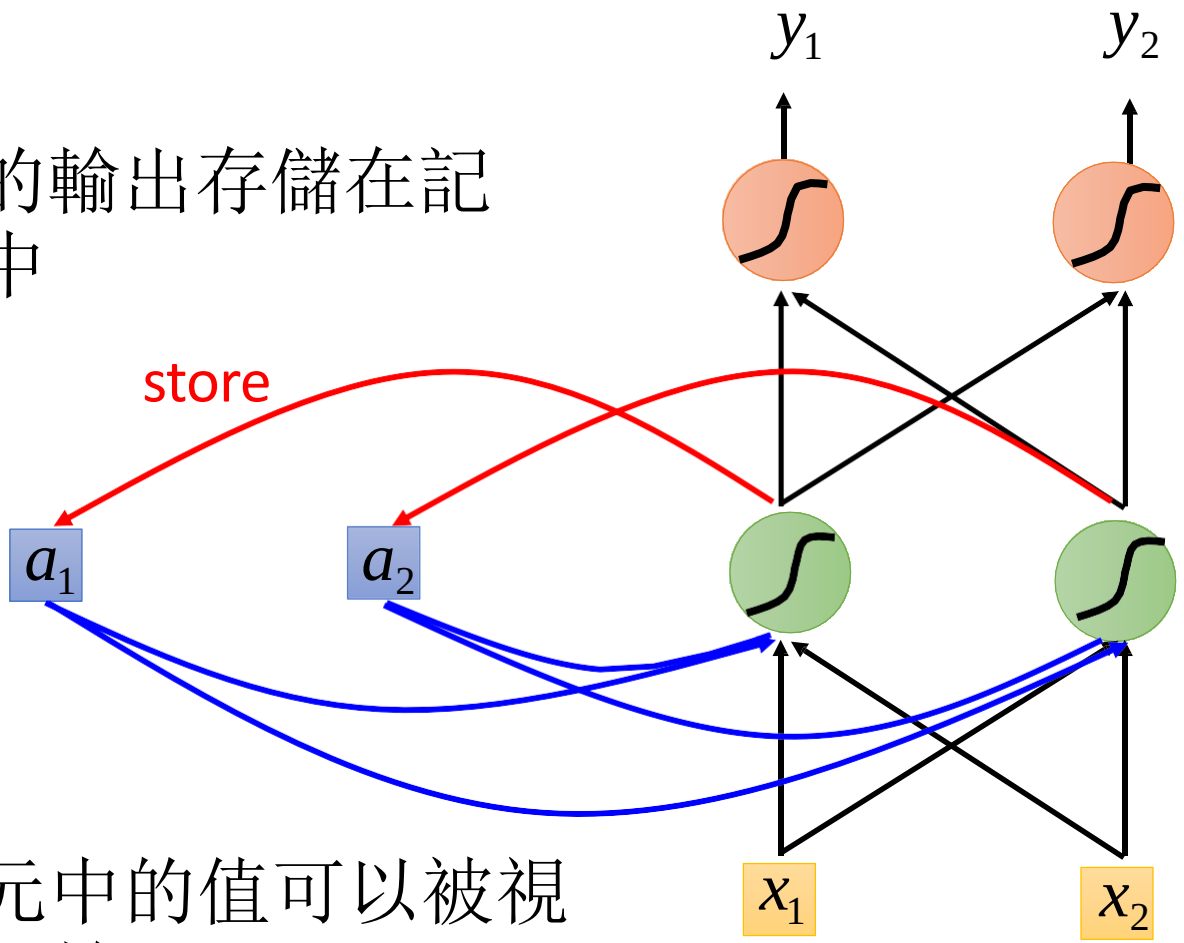
神經網路需“記憶”！

Taipei



迴圈神經網路 (RNN)

隱藏層的輸出存儲在記憶單元中



記憶單元中的值可以被視為另一個輸入

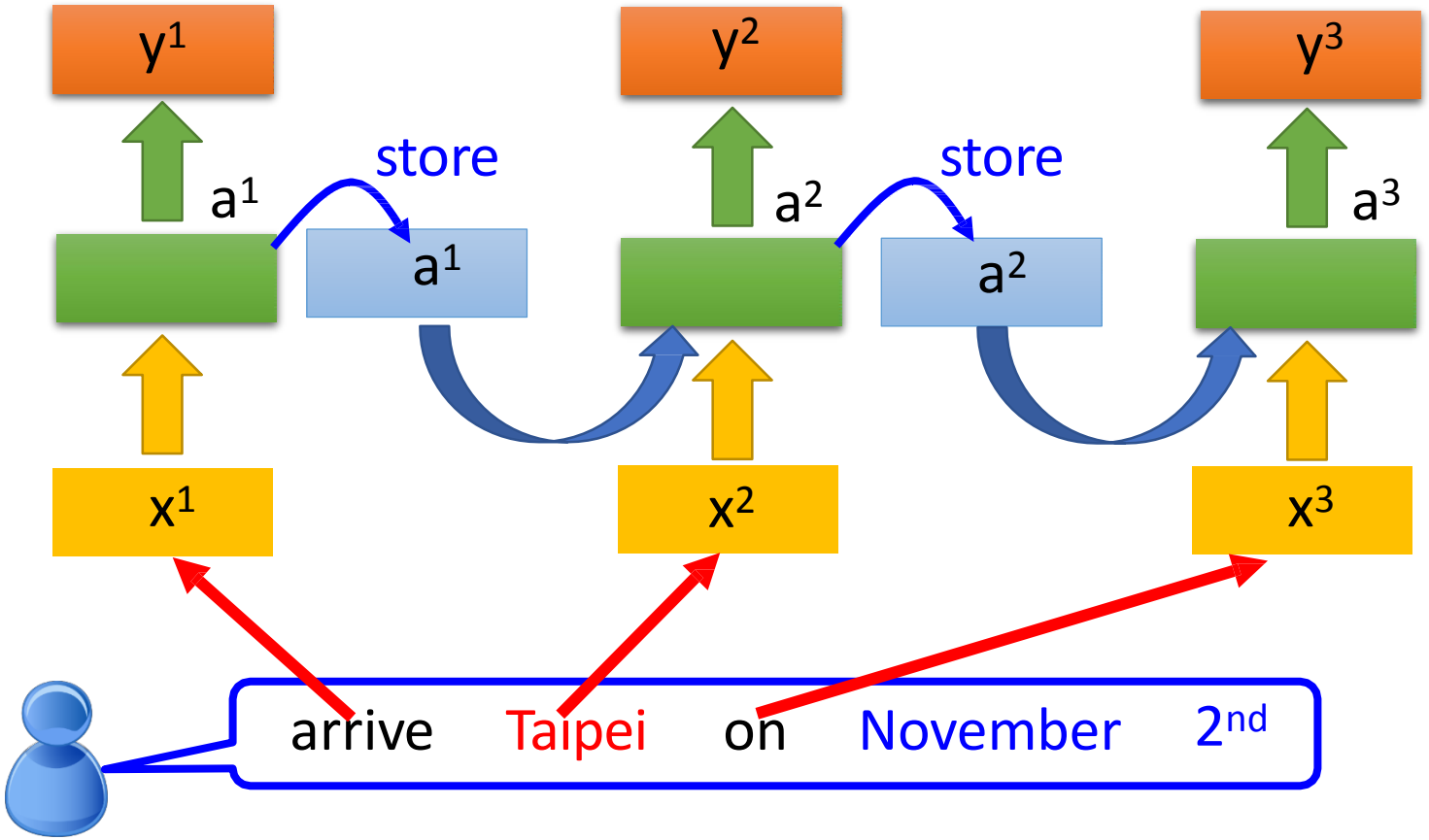
RNN

同樣的網路多次使用

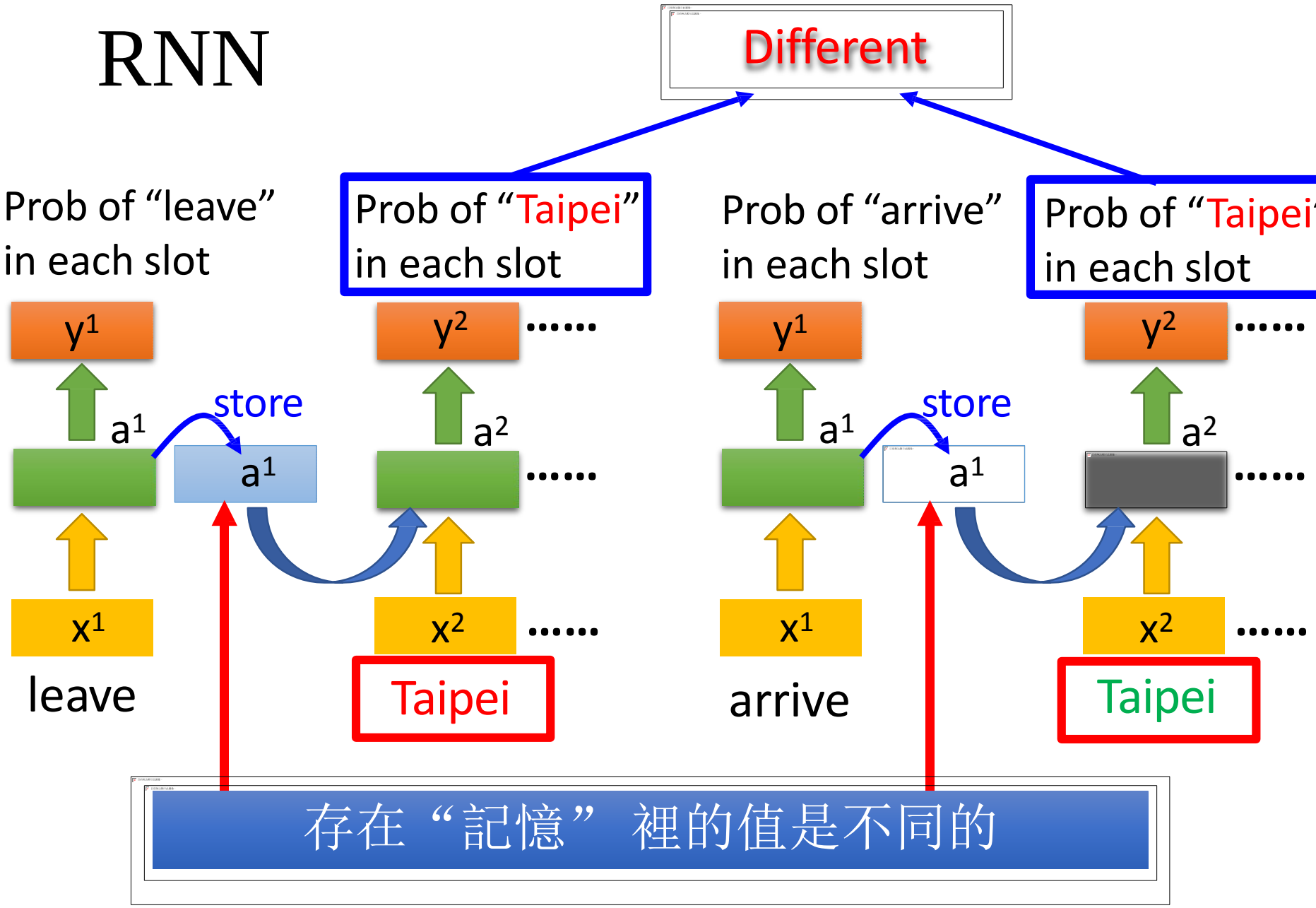
Probability of "arrive" in each slot

Probability of "Taipei" in each slot

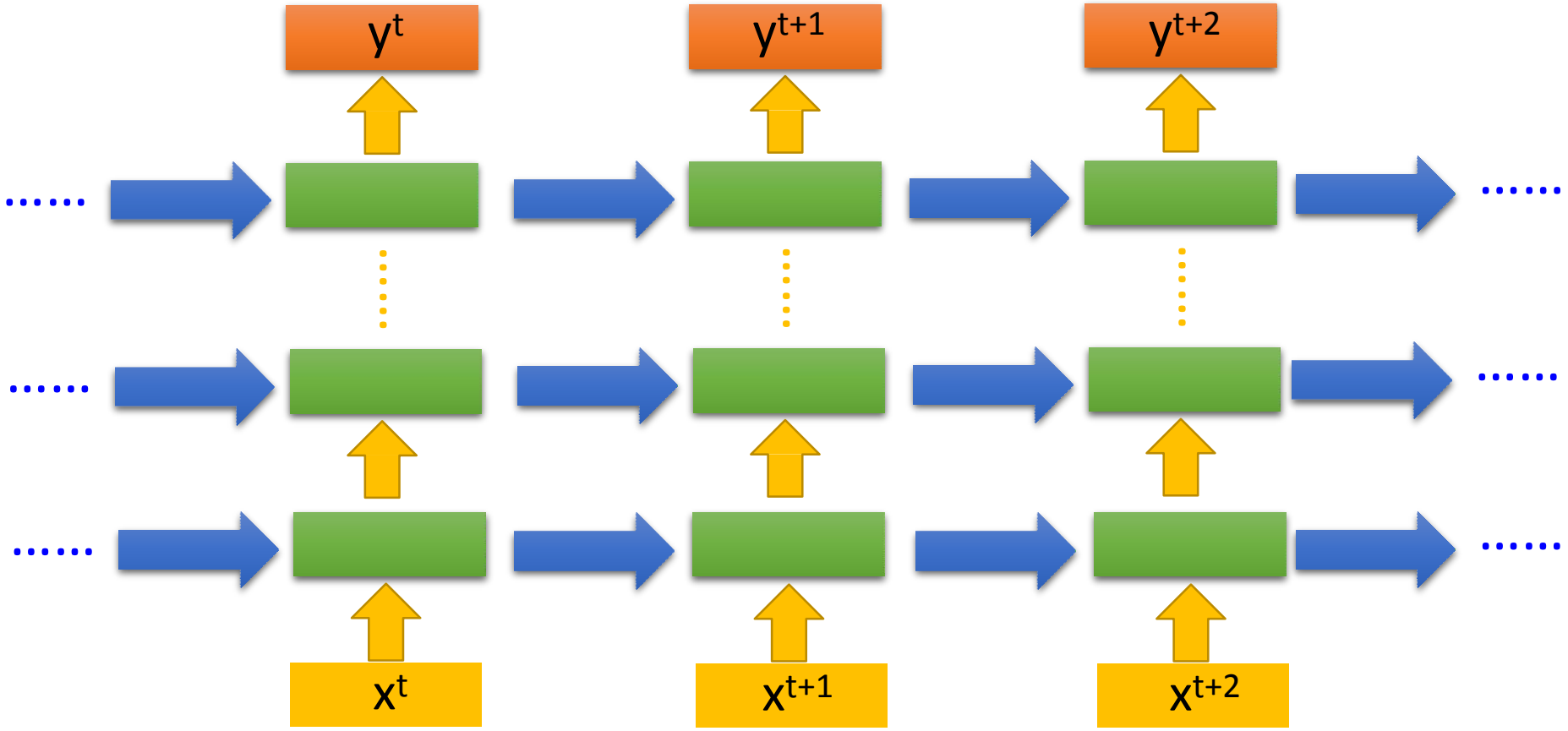
Probability of "on" in each slot



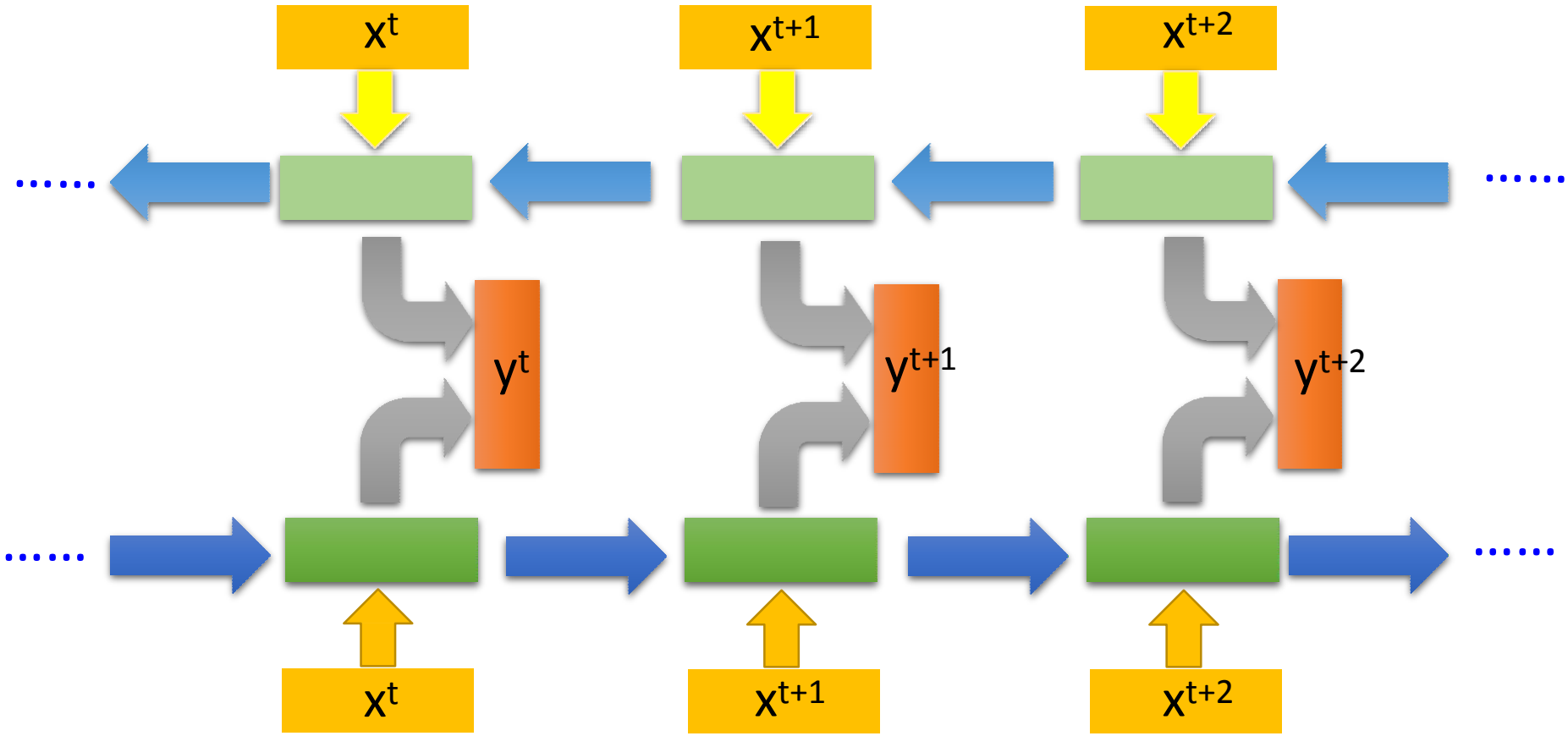
RNN


Different
Prob of "leave" in each slot
Prob of "Taipei" in each slot
Prob of "arrive" in each slot
Prob of "Taipei" in each slot
y1
y2
a1
a2
store
x1
x2
leave
Taipei
arrive
Taipei
存在"記憶"裡的值是不同的

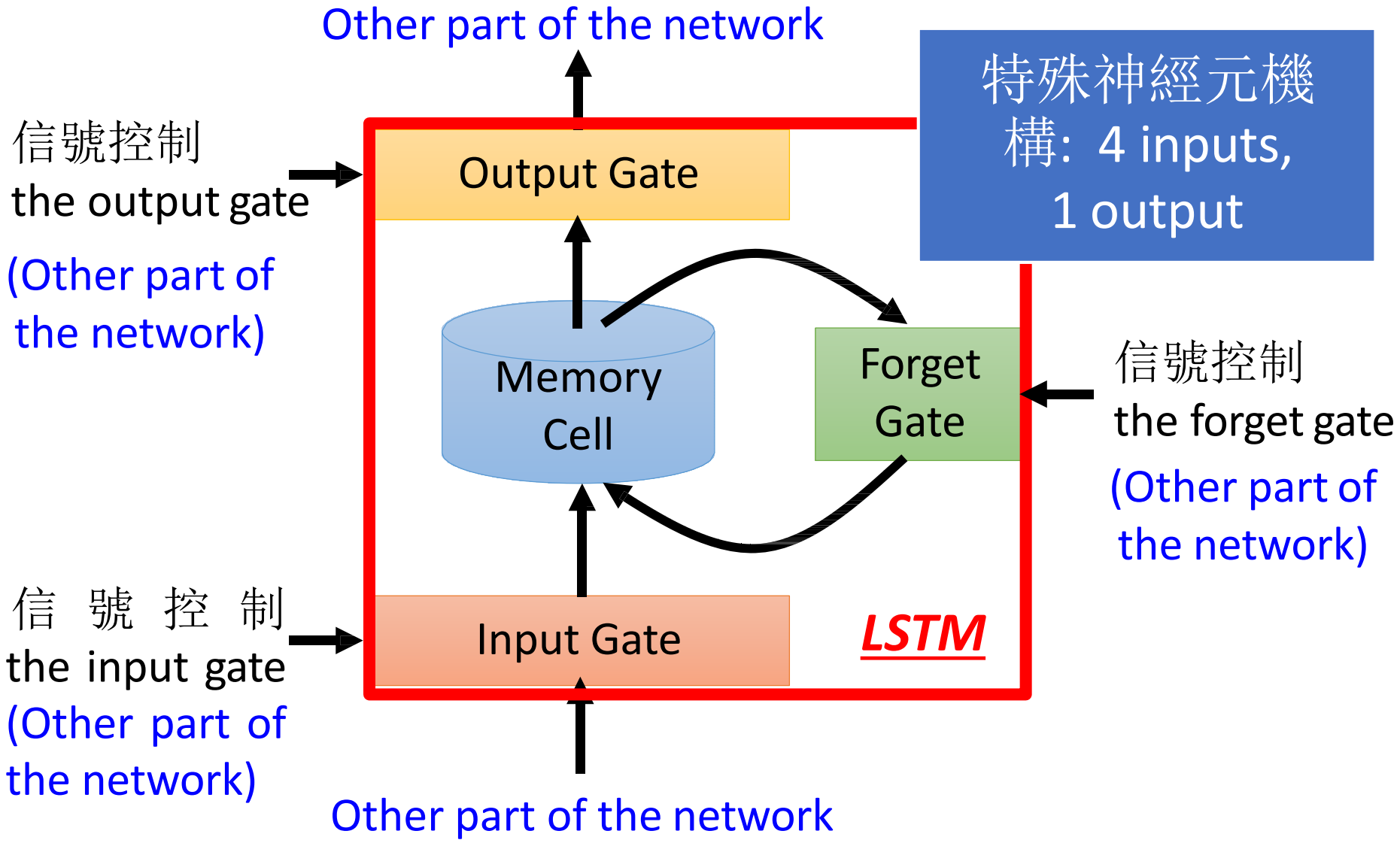
當然，它可以是深層網路 …

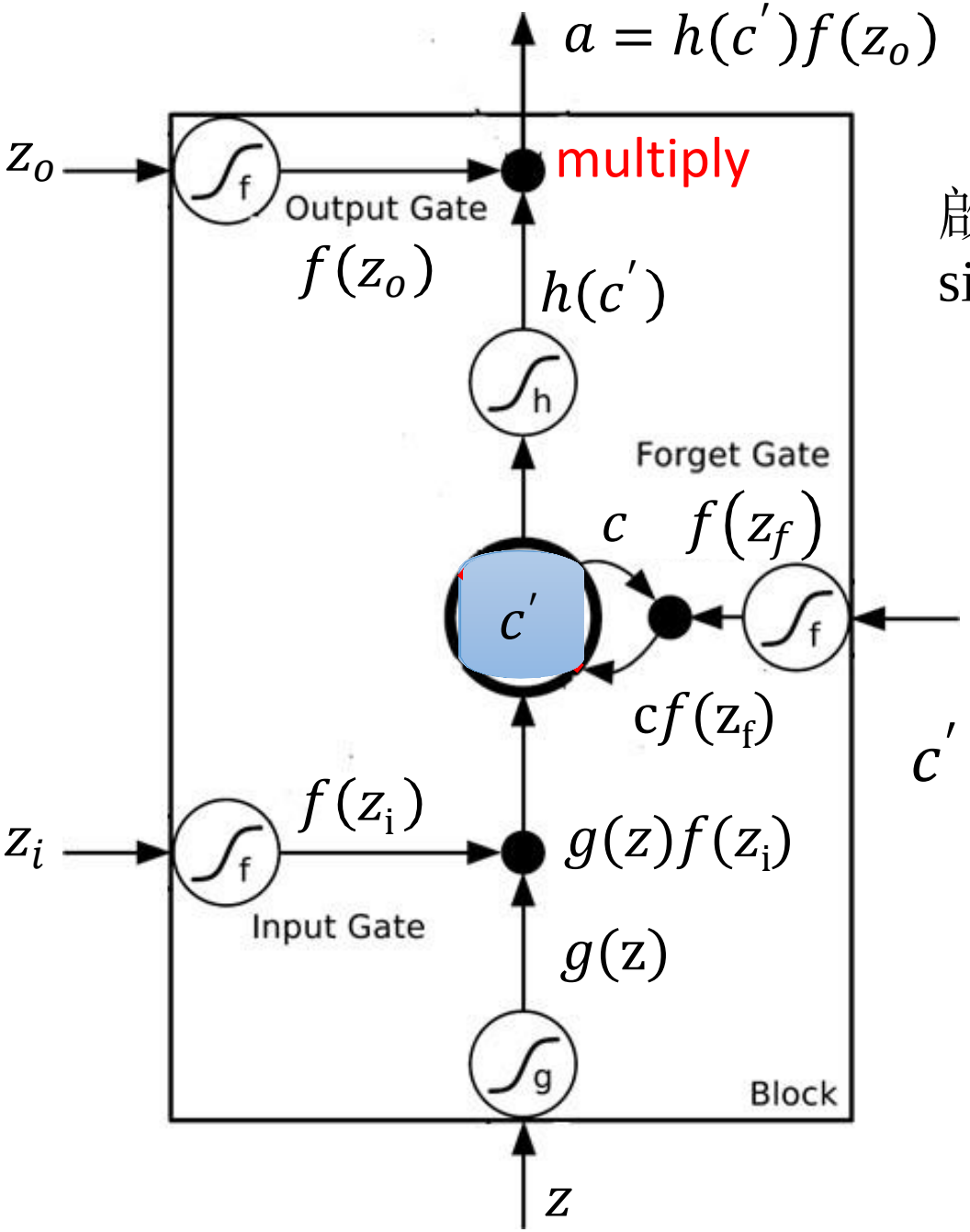


雙向 RNN



Long Short-term Memory (LSTM)





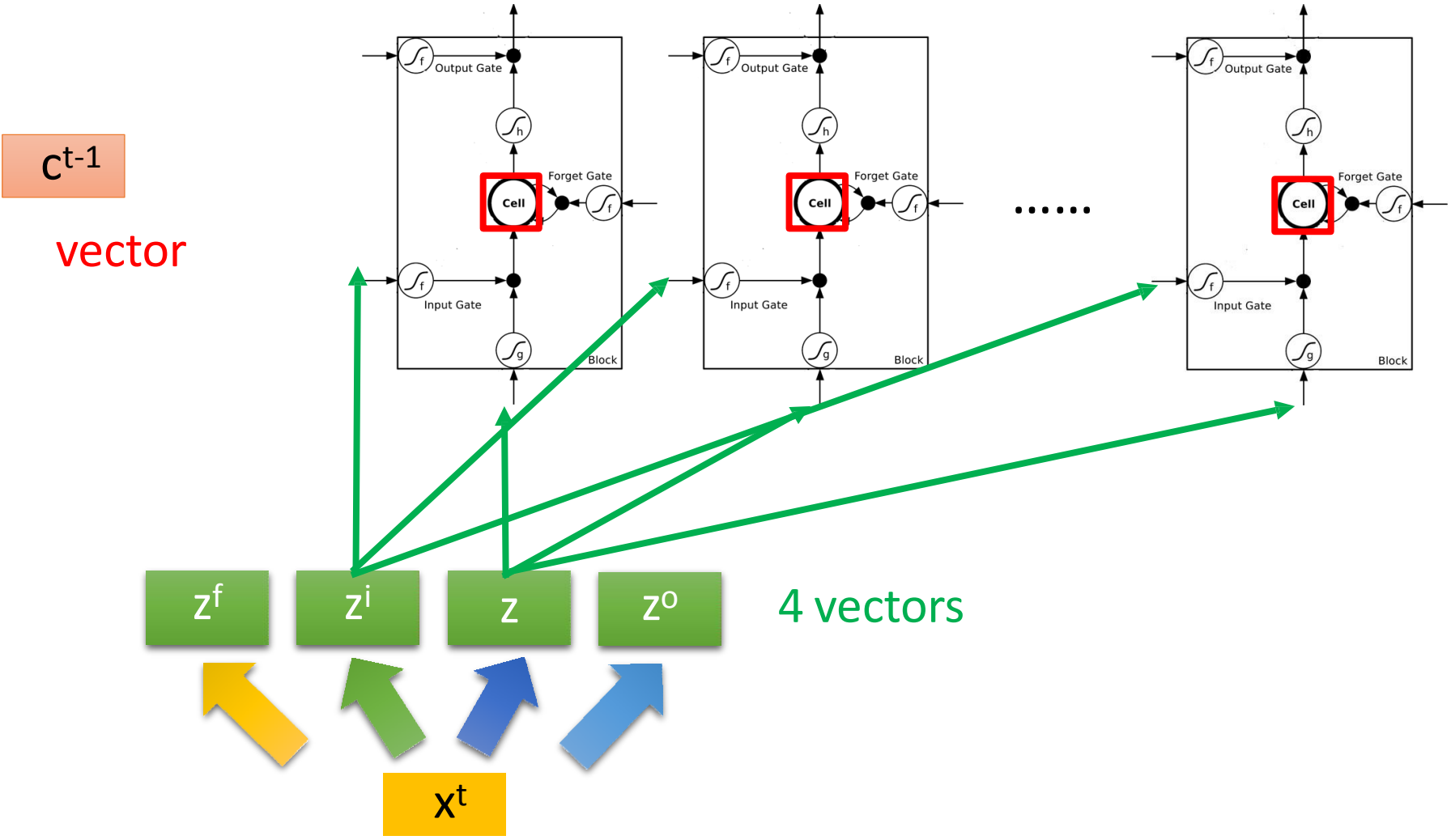
啟動函數通常選用
sigmoid function

輸出介於 0 到 1 之間

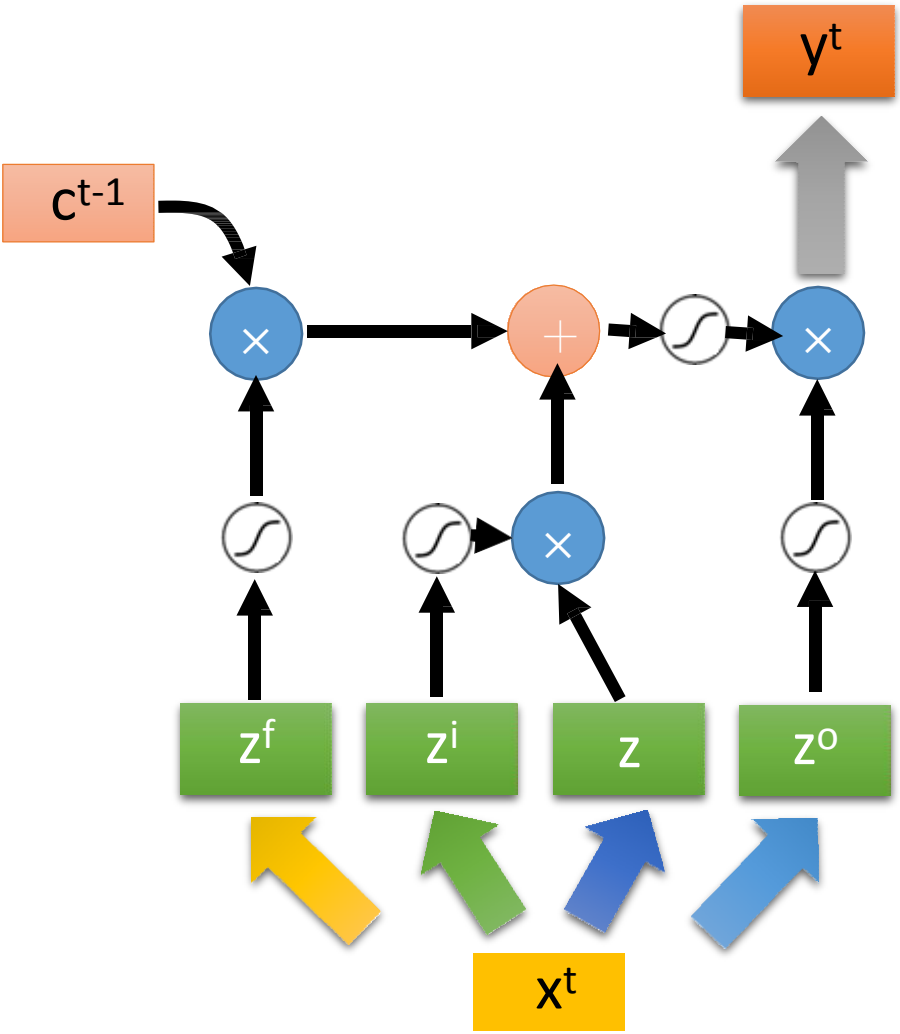
表示了門的打開和關閉

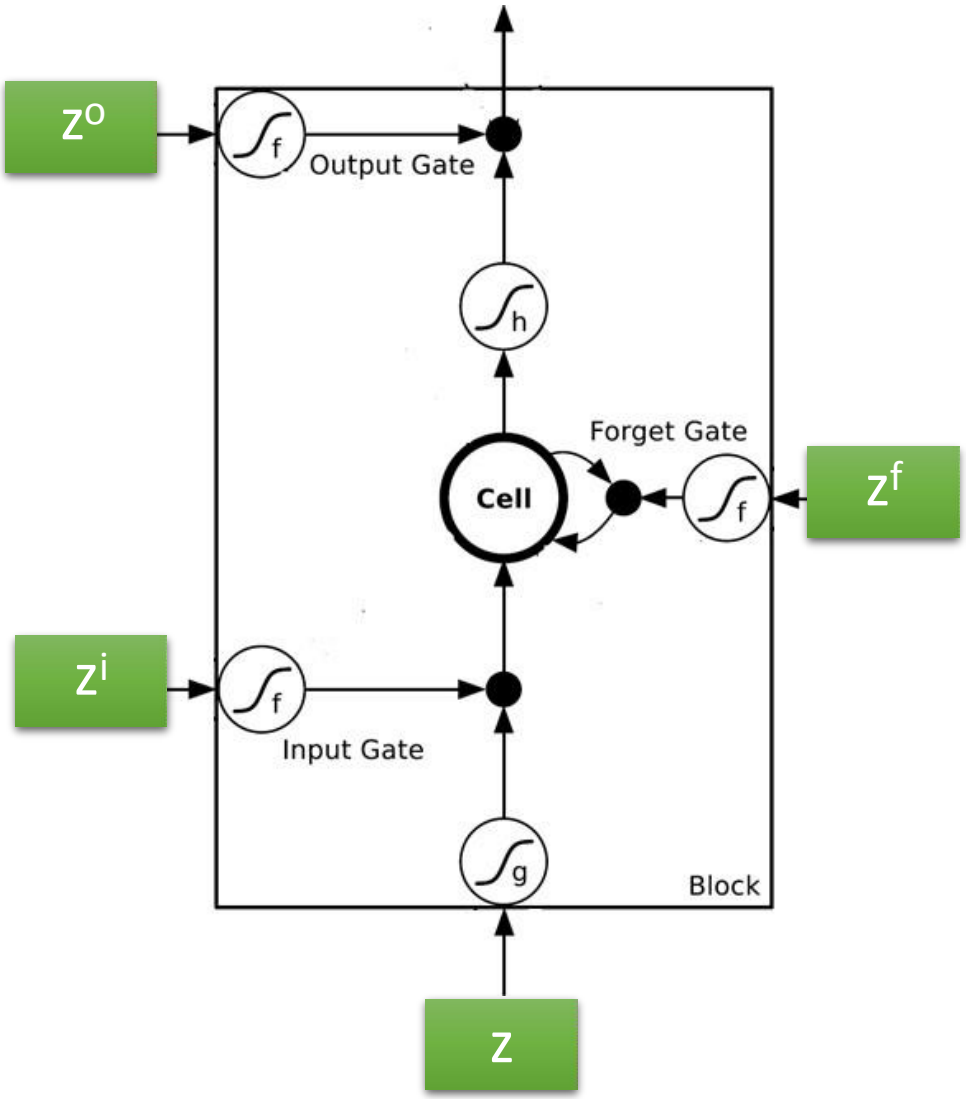
$$c' = g(z)f(z_i) + cf(z_f)$$

LSTM

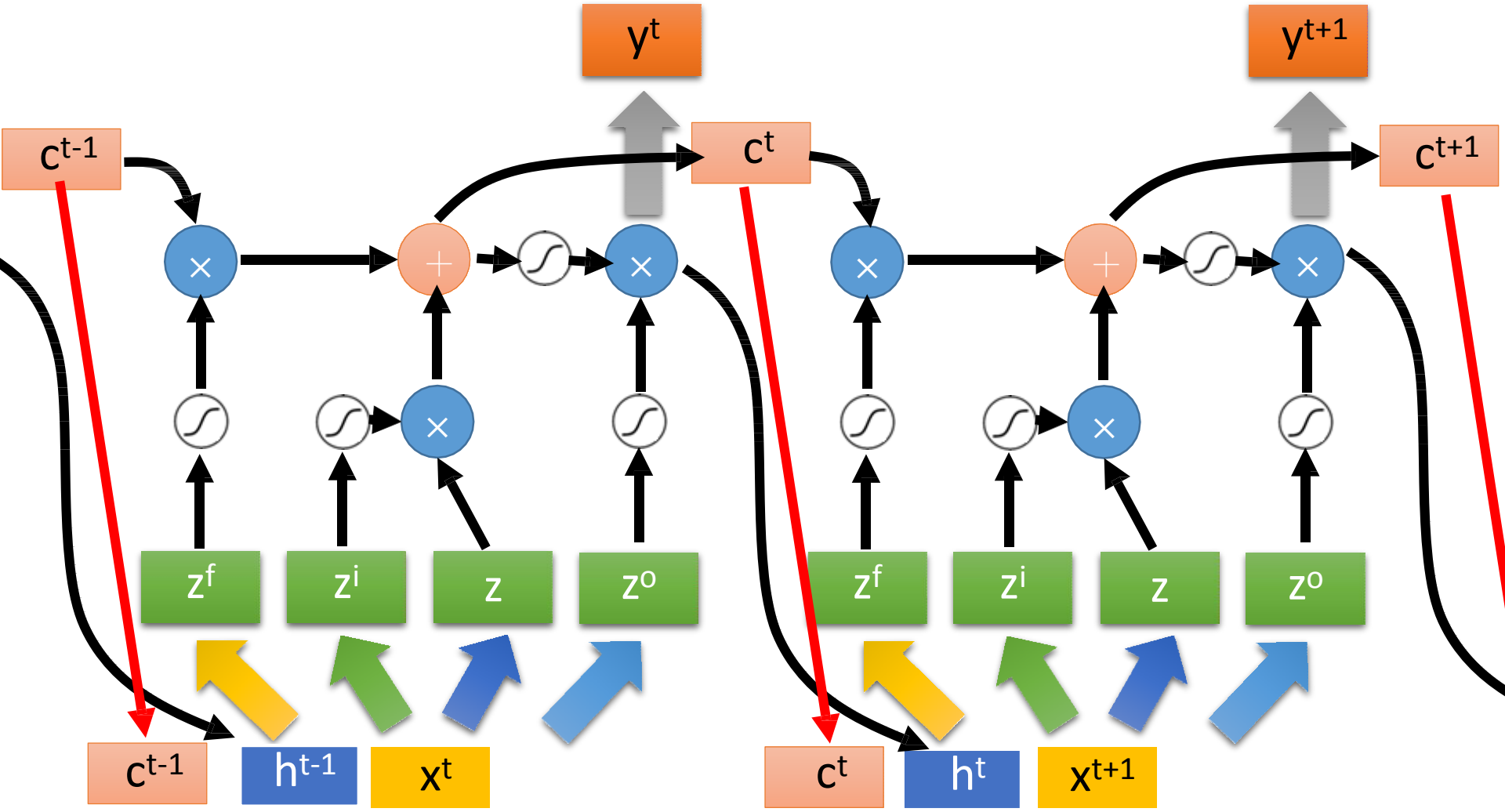

c^{t-1}
vector
Output Gate
Forget Gate
Cell
Input Gate
Block
.....
z^f
z^i
z
z^o
4 vectors
x^t

LSTM

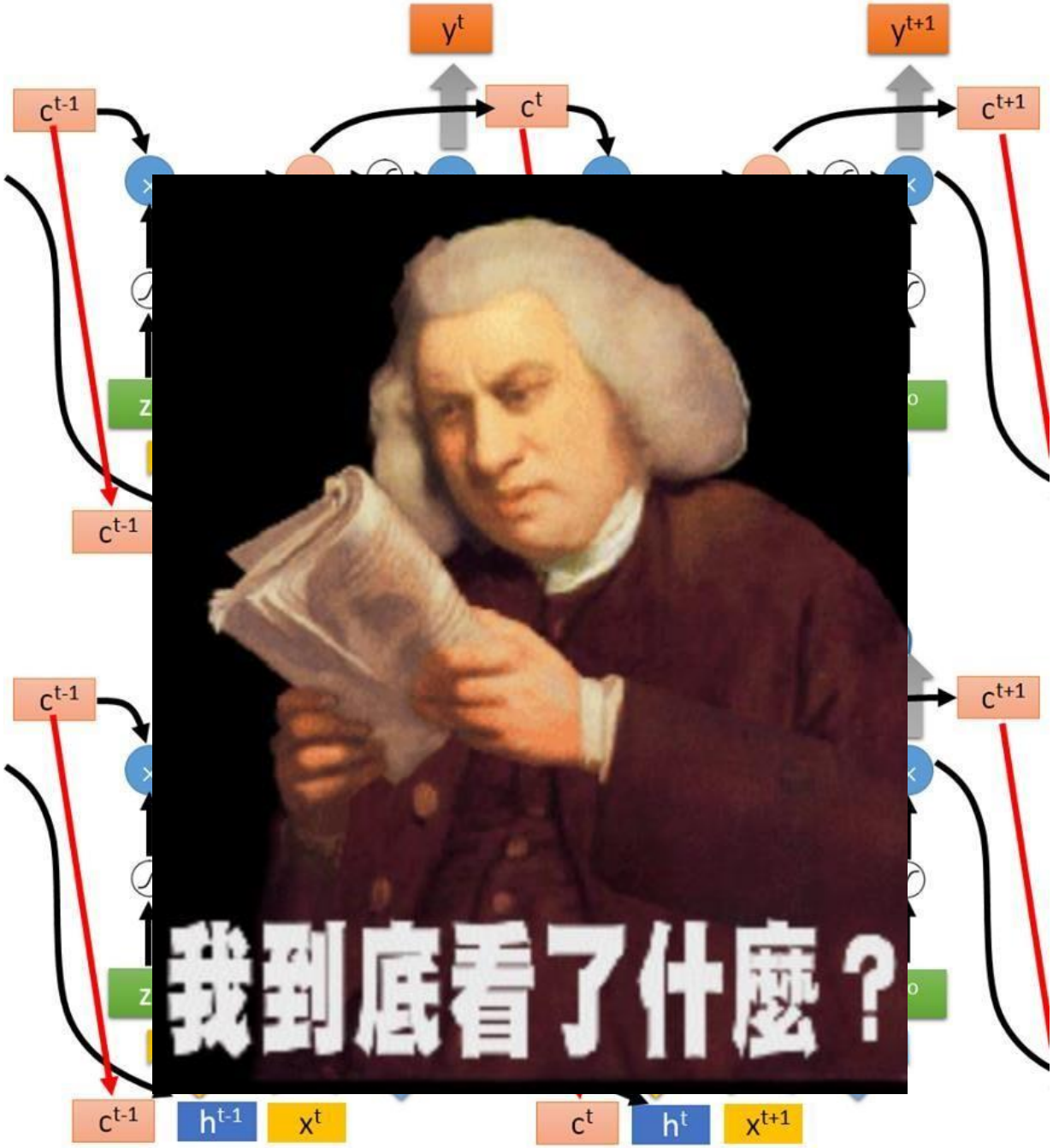

y^t
c^{t-1}
z^f
z^i
z
z^o
x^t


z^o
Output Gate
Forget Gate
Cell
z^f
z^i
Input Gate
Block
z

LSTM

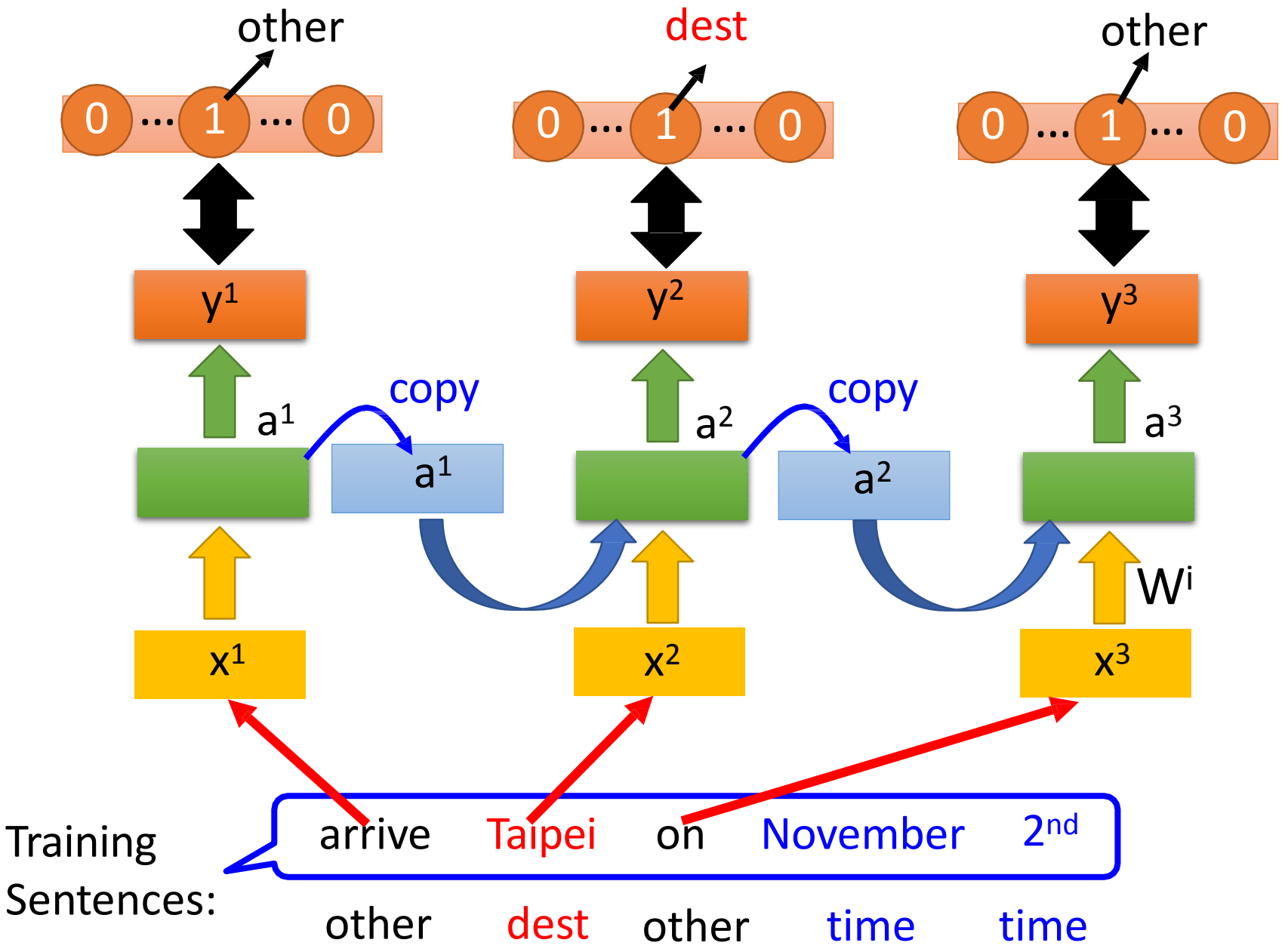


Multiple-layer LSTM



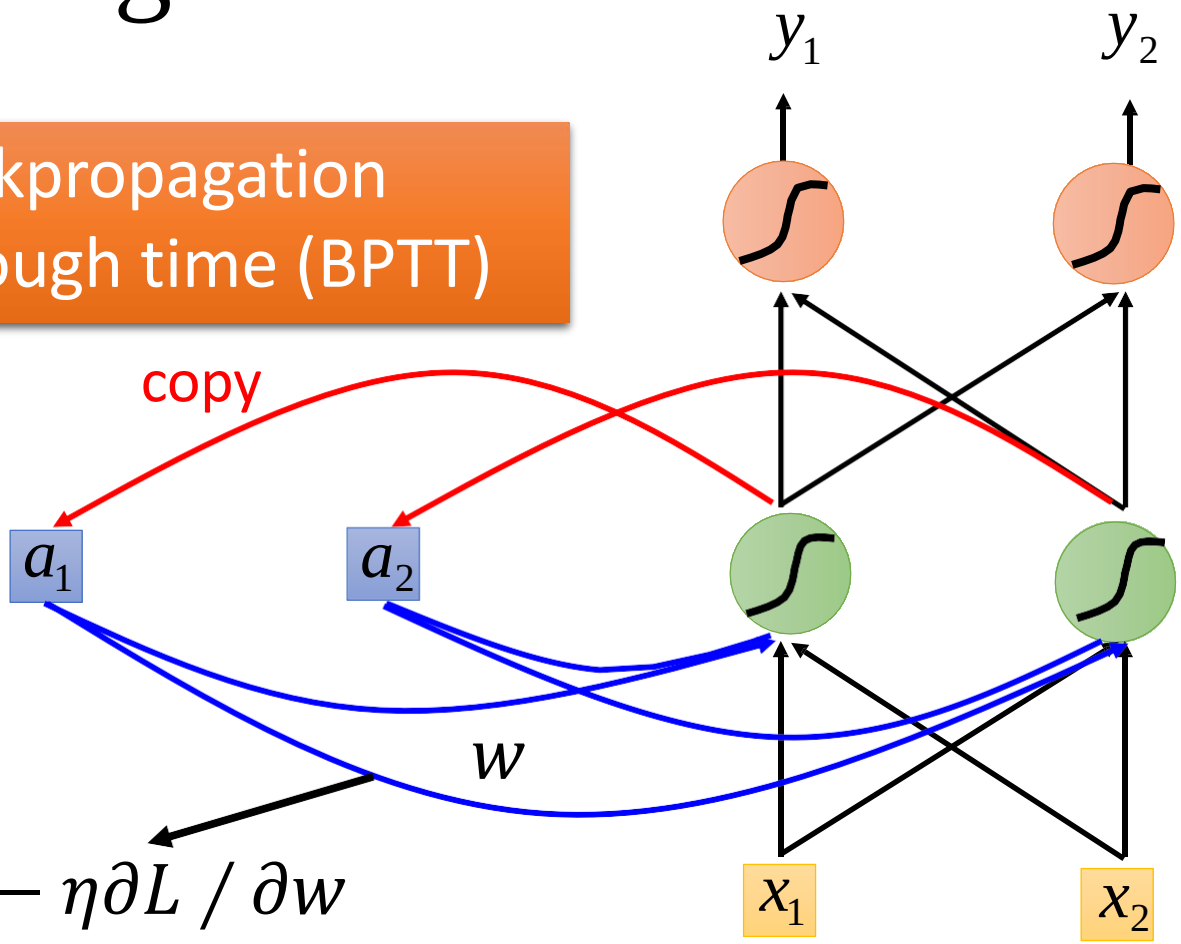
https://img.komicolle.org/2015-09-20/src/14426967627131.gif

Learning Target



Learning

Backpropagation through time (BPTT)

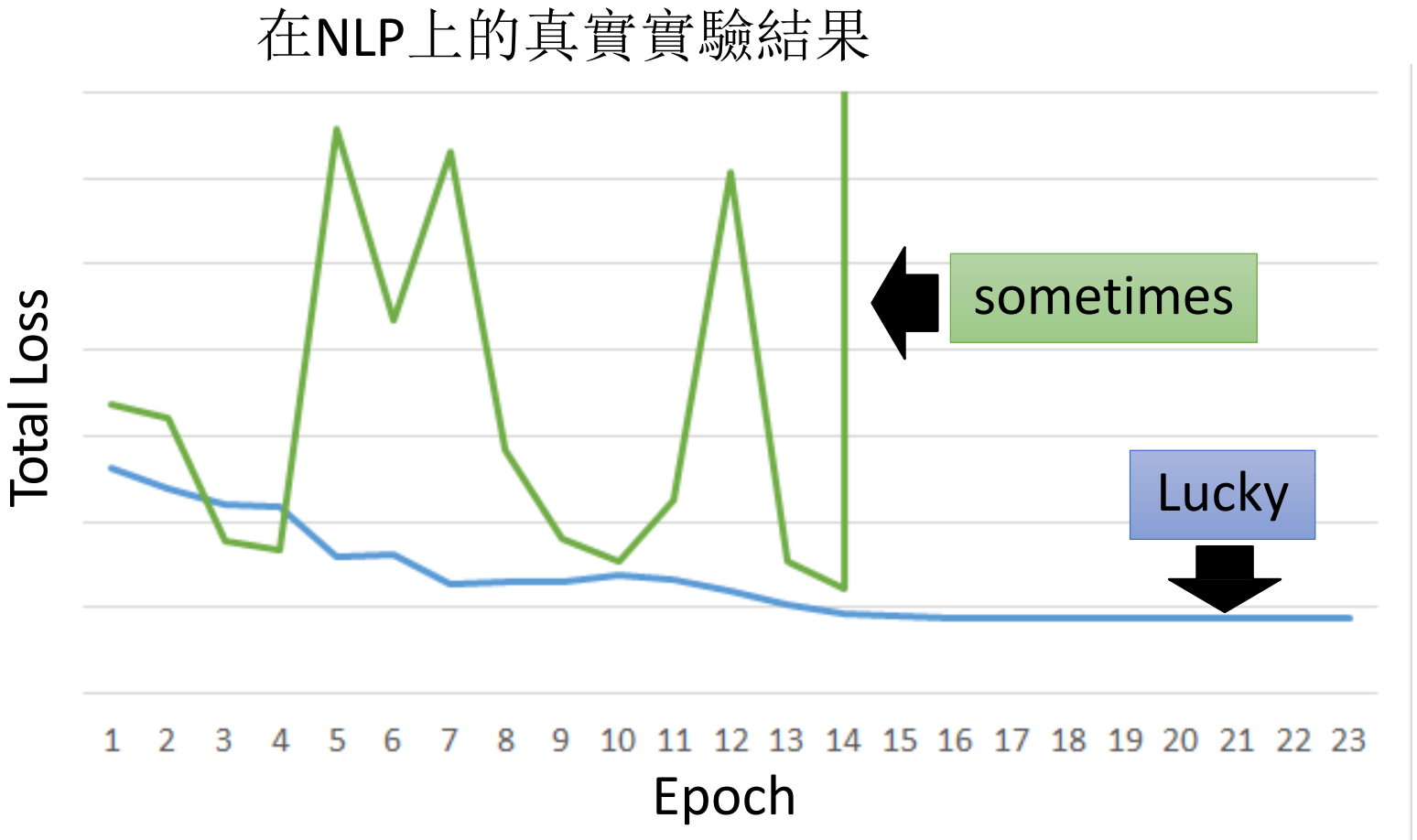


$w \leftarrow w - \eta \partial L / \partial w$

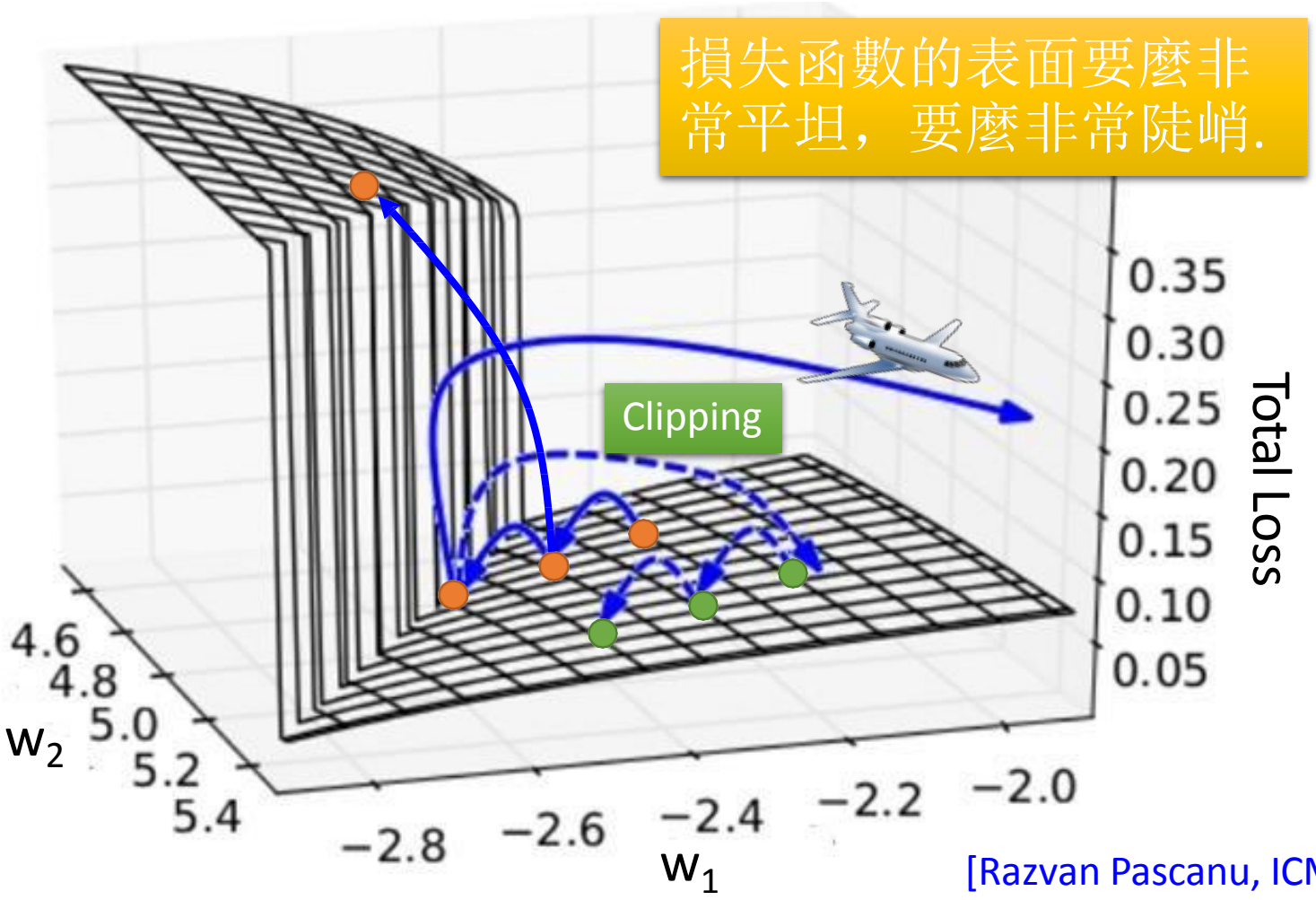
RNN的學習在實踐中是非常困難的

不幸的是 ……

- 基於RNN的網路並不總是容易學習



誤差表面粗糙



[Razvan Pascanu, ICML'13]

Why?

$w = 1 \Rightarrow y^{1000} = 1$

$w = 1.01 \Rightarrow y^{1000} \approx 20000$

Large $\partial L / \partial w$ → Small Learning rate?

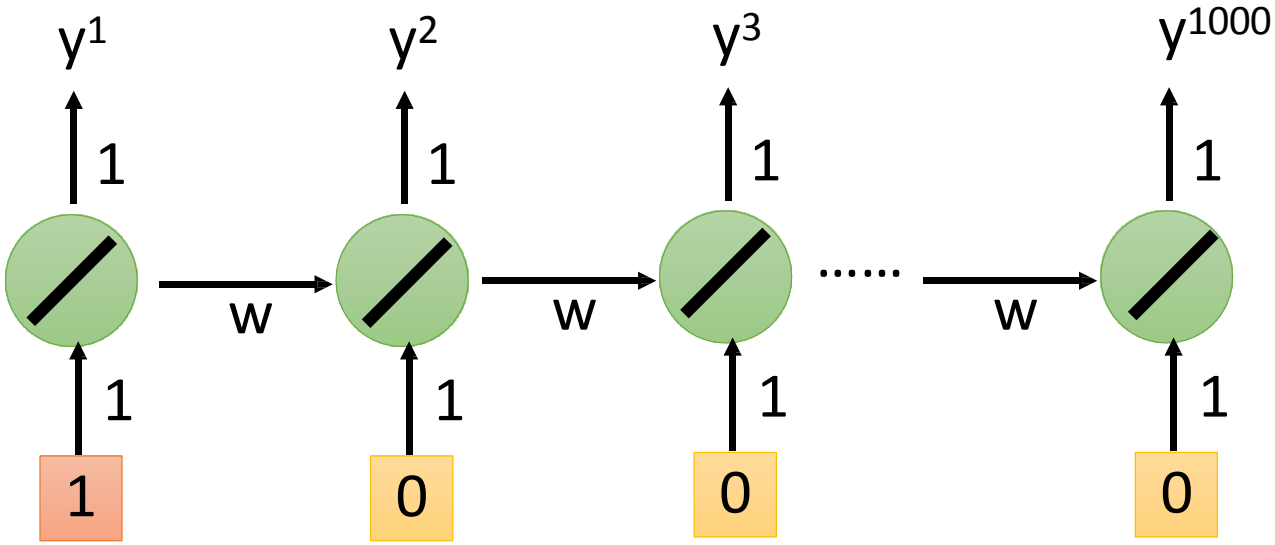
$w = 0.99 \Rightarrow y^{1000} \approx 0$

$w = 0.01 \Rightarrow y^{1000} \approx 0$

small $\partial L / \partial w$ → Large Learning rate?

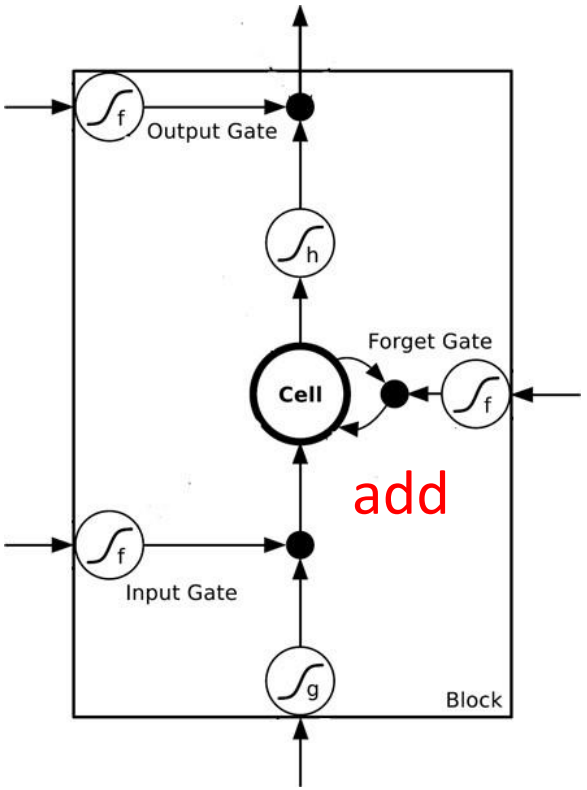
$=w^{999}$

Toy Example



有用的技巧

- Long Short-term Memory (LSTM)
  - 可以解決梯度消失 (不能解決梯度爆炸)
    - Memory and input are ***added***
    - 除非 forget gate 被關閉，否則影響不會消失



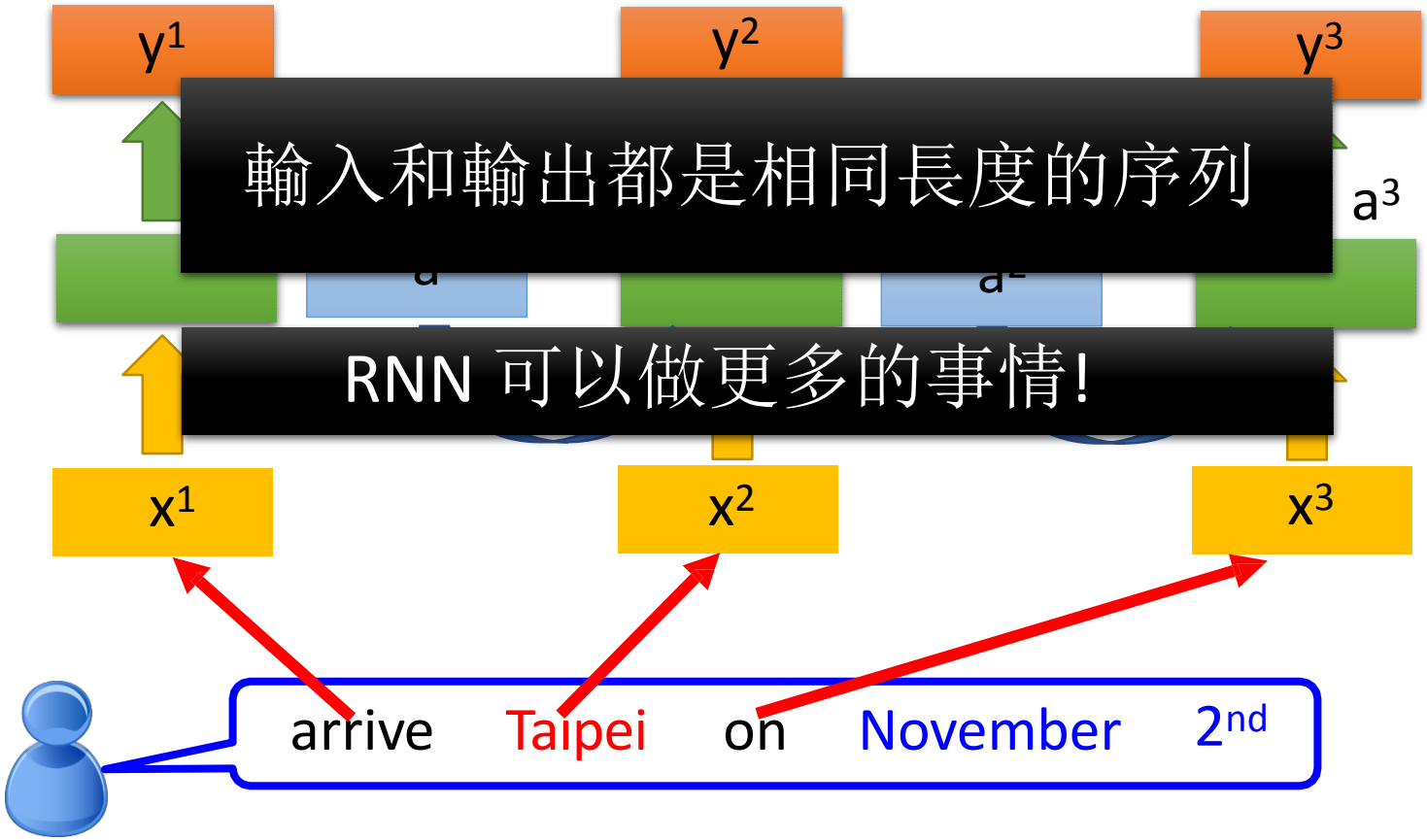
不會發生梯度消失
(如果forget gate是開啟的)

更多的應用程式

Probability of "arrive" in each slot

Probability of "Taipei" in each slot

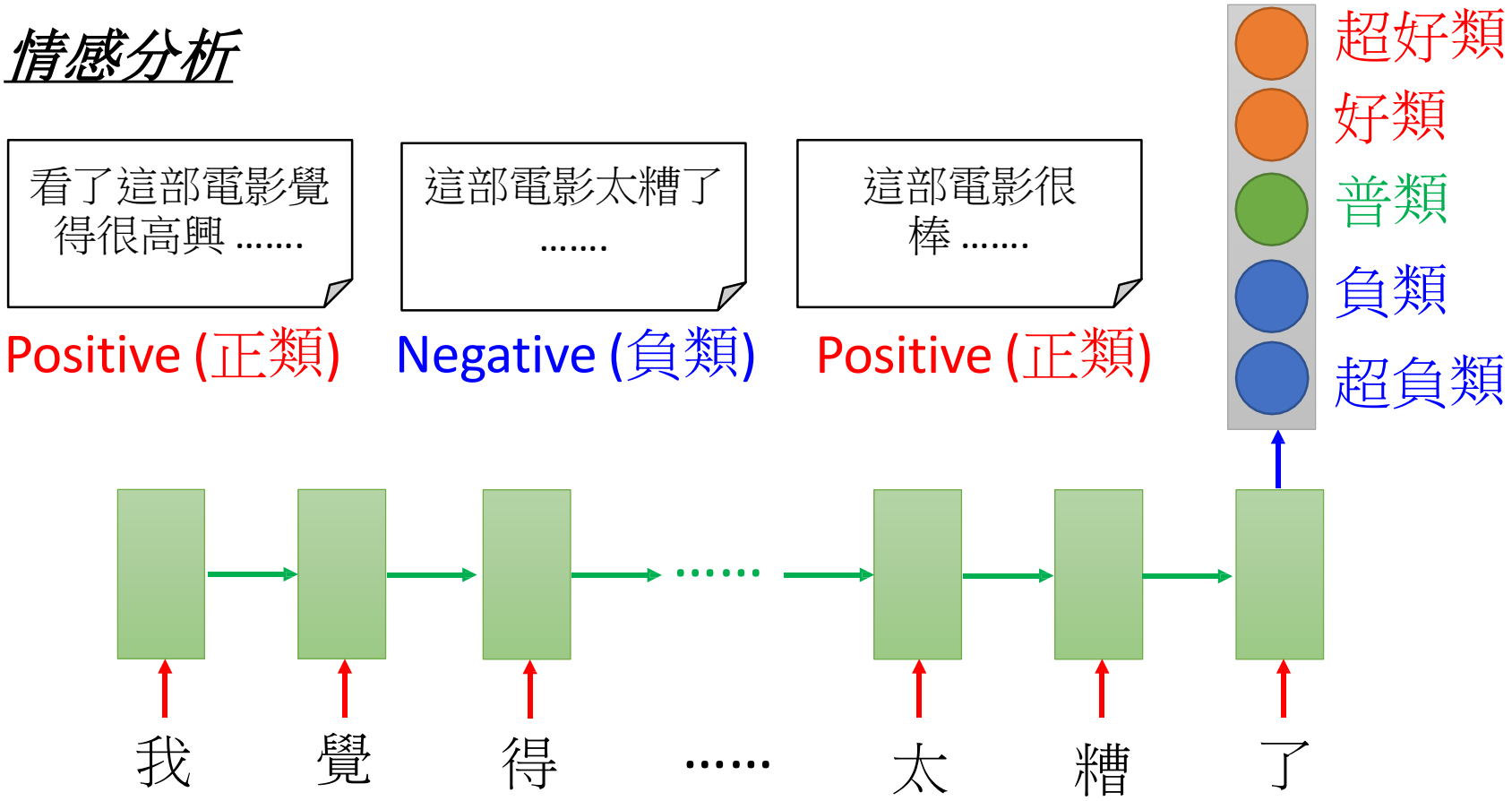
Probability of "on" in each slot



Many to one

- 輸入是一個向量序列，但輸出只是一個向量

情感分析



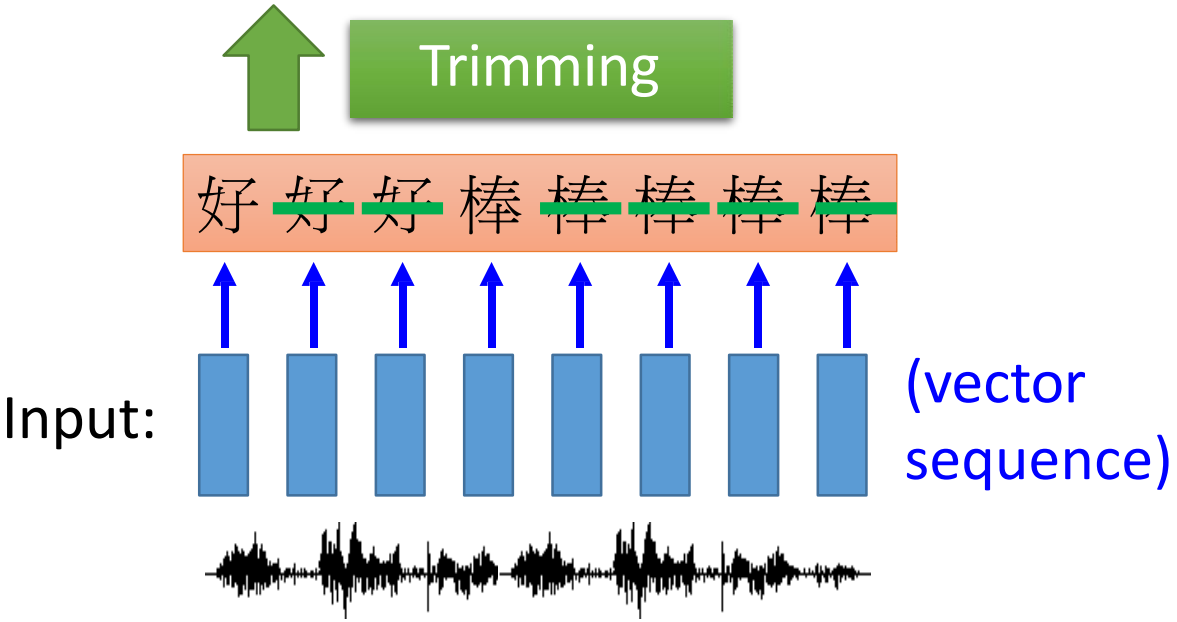
Many to Many (輸出較短)

- 輸入和輸出都是序列, ***但輸出比較短***
  - E.g. 語音辨識

Output: “好棒” (character sequence)

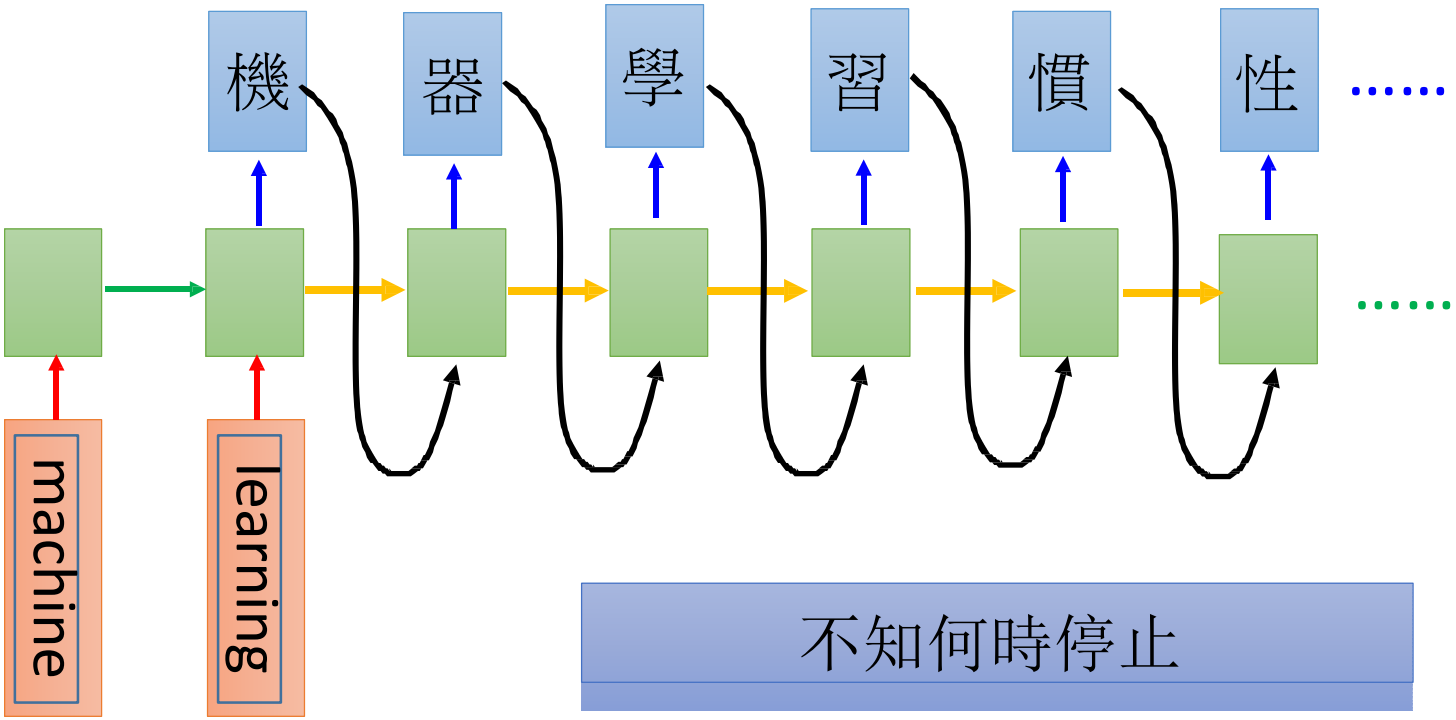
Problem?

Why can’t it be “好棒棒”



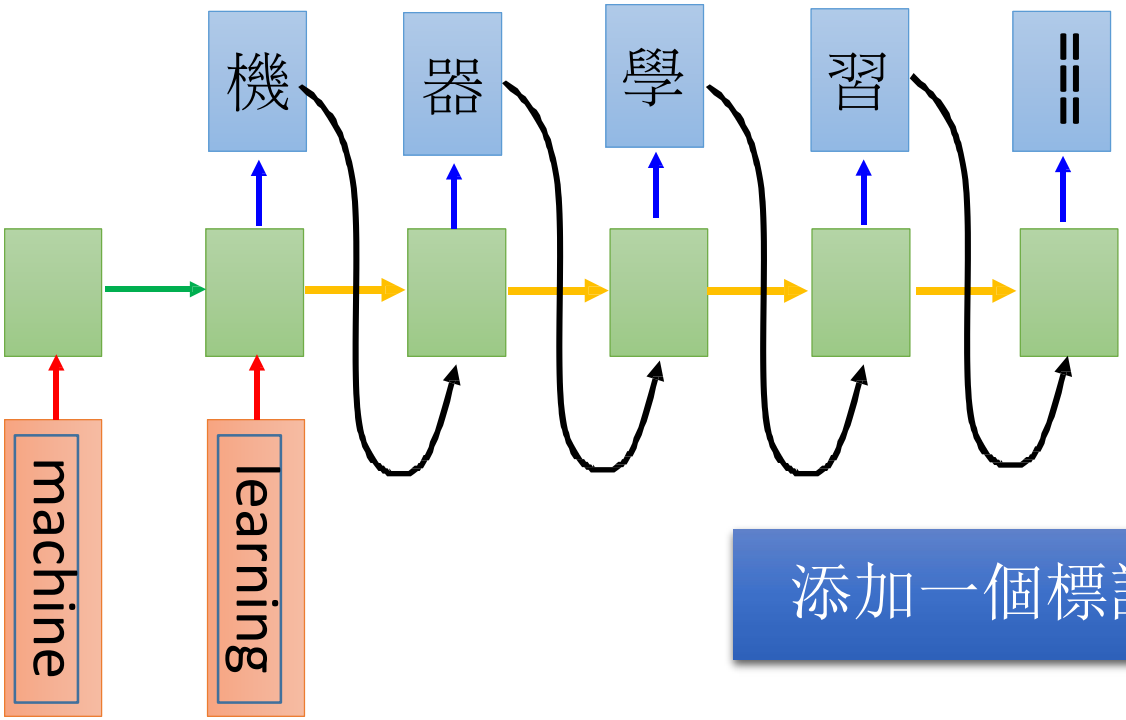
Many to Many (No Limitation)

- 輸入和輸出都是不同長度的序列 → ***Sequence to sequence learning***
  - E.g. ***機器翻譯*** (machine learning→機器學習)



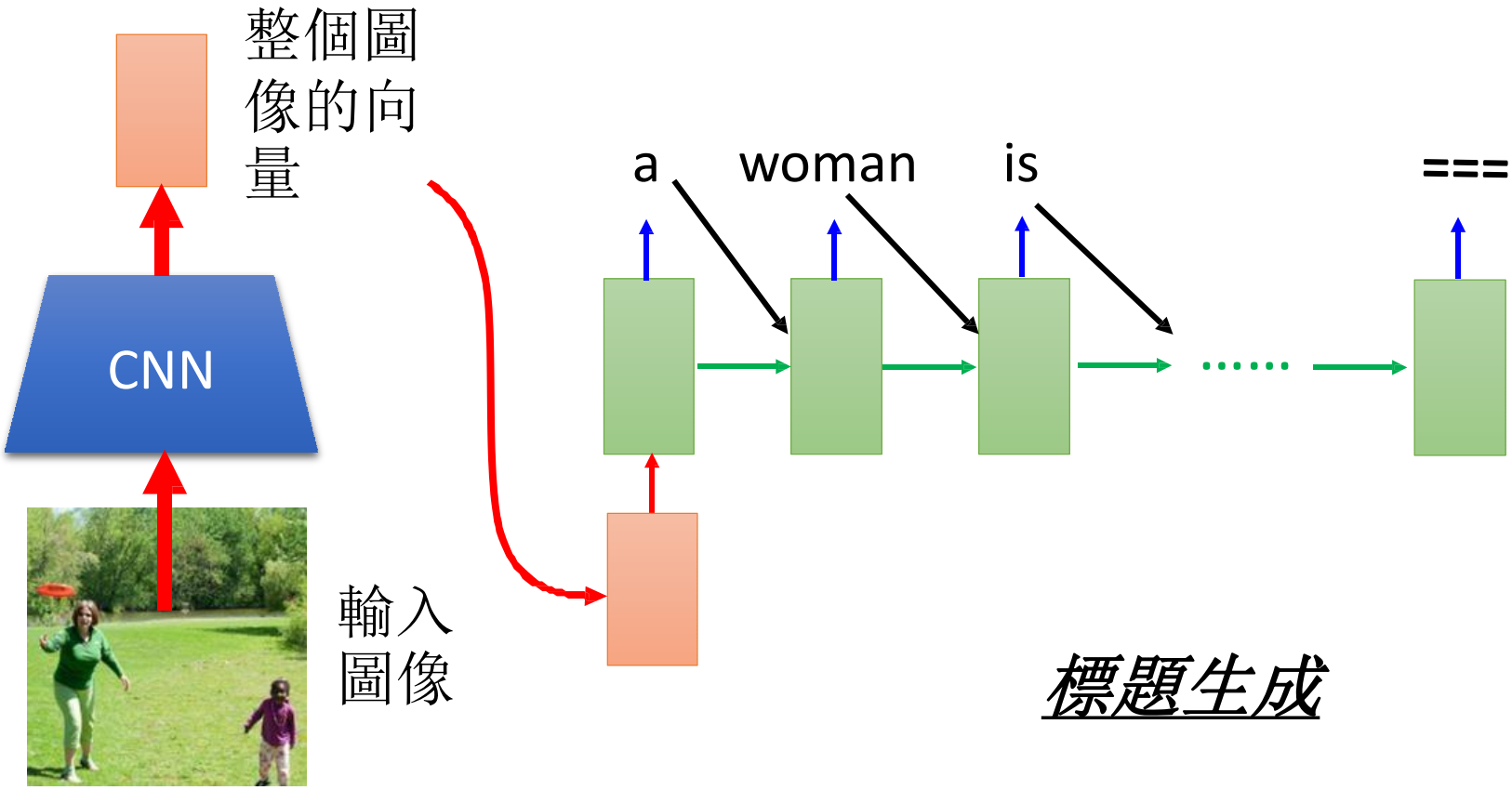
Many to Many (沒有限制)

- 輸入和輸出都是不同長度的序列 → ***Sequence to sequence learning***
  - E.g. ***機器翻譯*** (machine learning→機器學習)

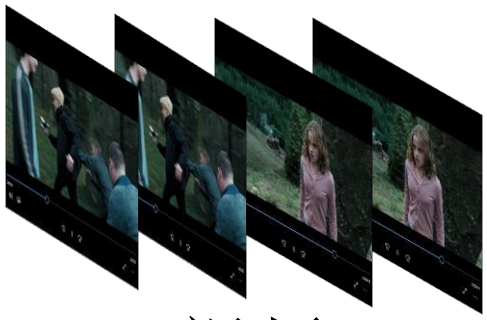


One to Many

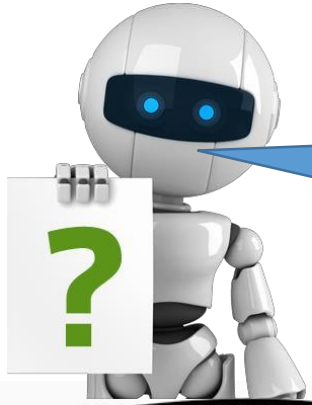
- 輸入圖像，但輸出一系列單詞



應用:
視頻字幕生成

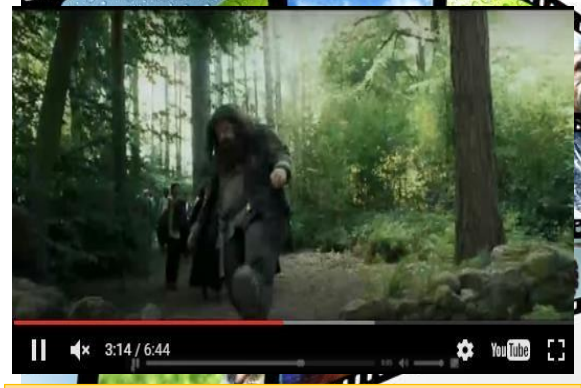


視頻



一個女孩子正在奔跑

一群人被一棵樹撞到



一群人走在森林裡

結束語

卷積神經網路
(CNN)

迴圈神經網路 (RNN)

Lecture IV:
下一個浪潮

大綱

監督學習

- 超深網路
- 注意力模型

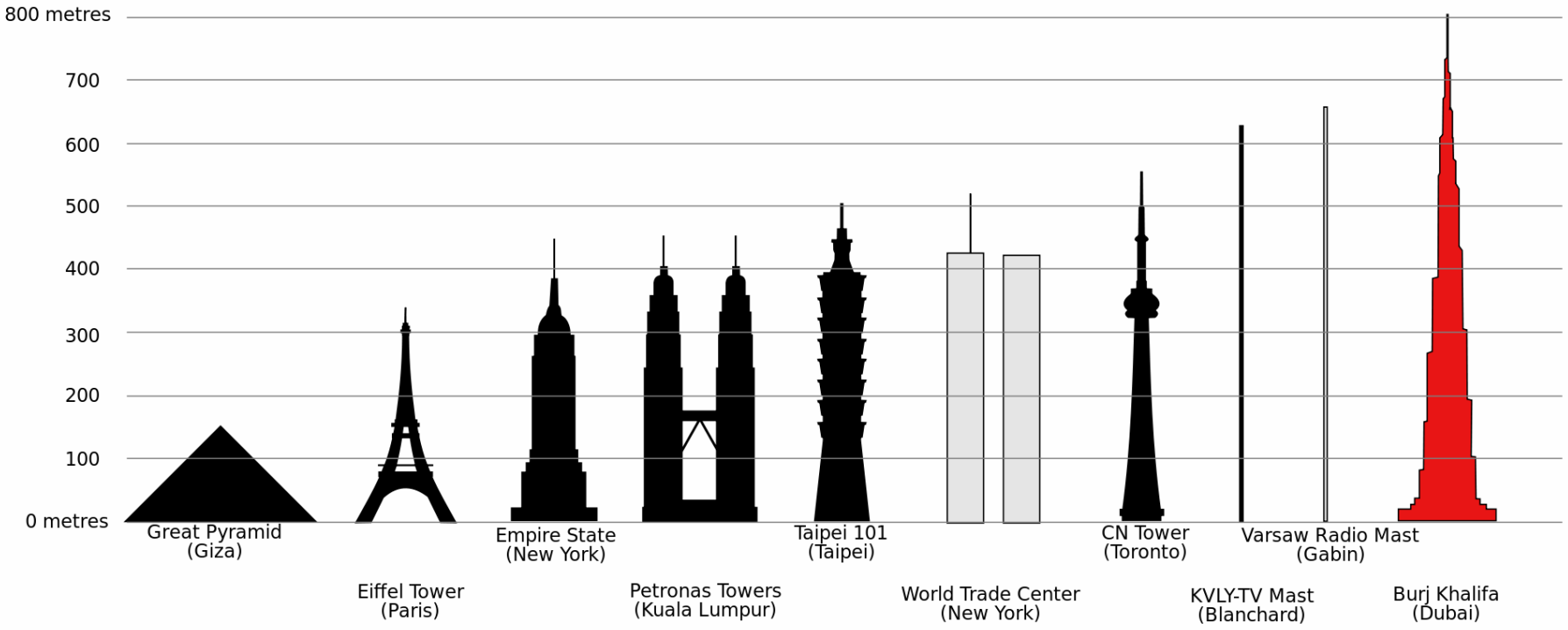
} 新型神經網路結構

強化學習

無監督學習

- 圖像: 認識到世界是什麼樣
- 文本: 理解單詞的含義
- 視頻: 在無監督的情況下學習人類的語言

摩天大樓

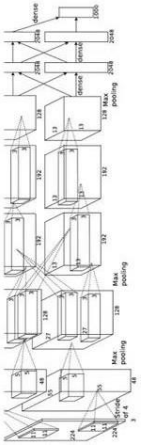


https://zh.wikipedia.org/wiki/%E9%9B%99%E5%B3%B0%E5%A1%94#/media/File:BurjDubaiHeight.svg

超深網路

http://cs231n.stanford.edu/slides/winter1516_lecture8.pdf

8 layers



16.4%

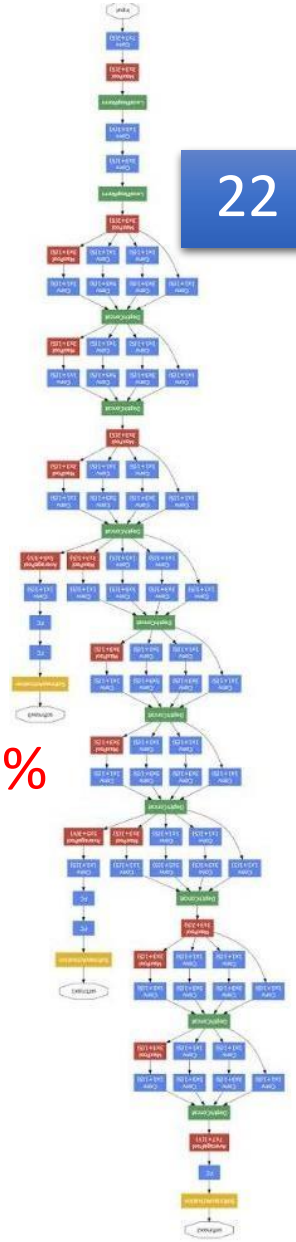
AlexNet (2012)

19 layers

7.3%

VGG (2014)

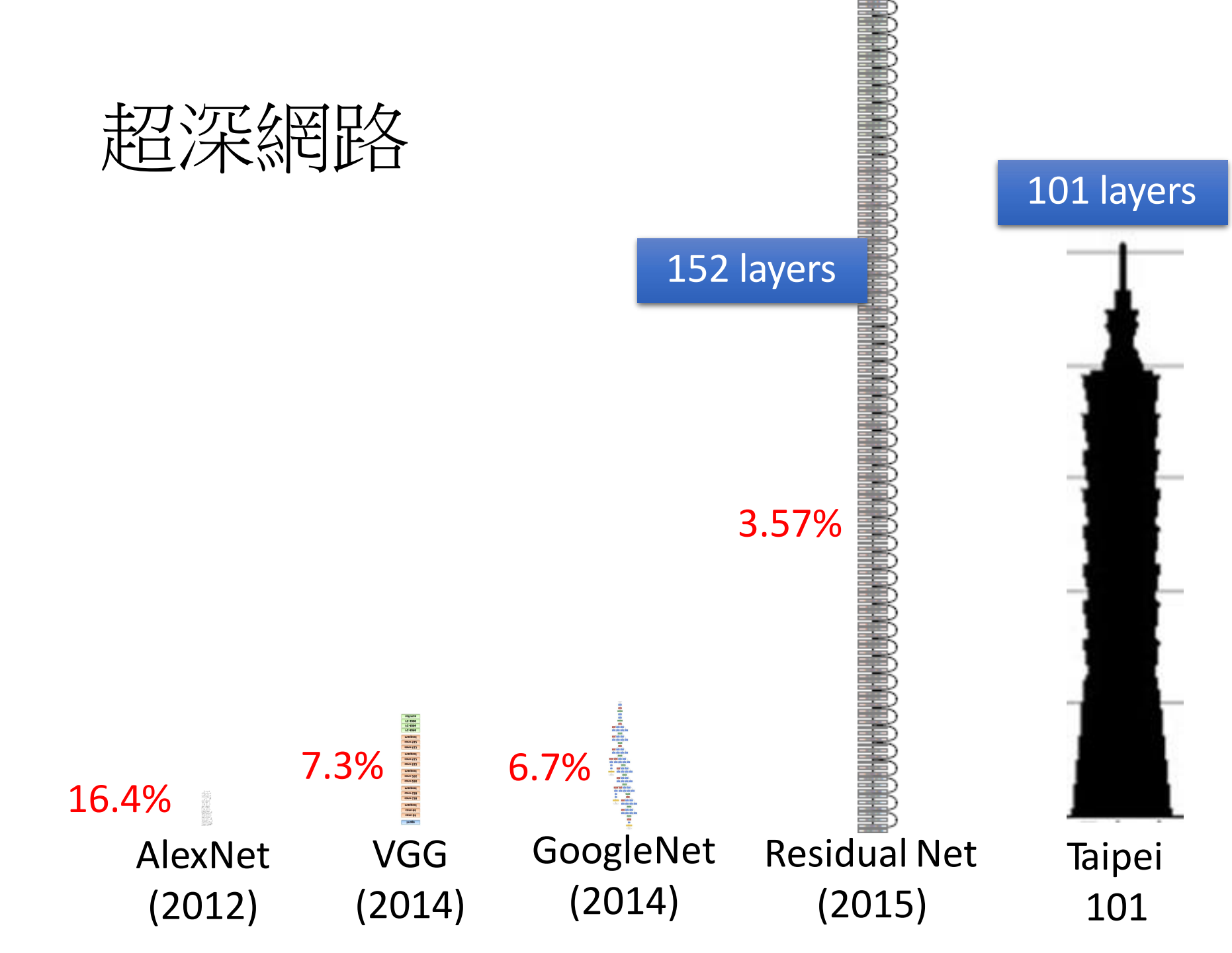
22 layers



6.7%

GoogleNet (2014)

超深網路

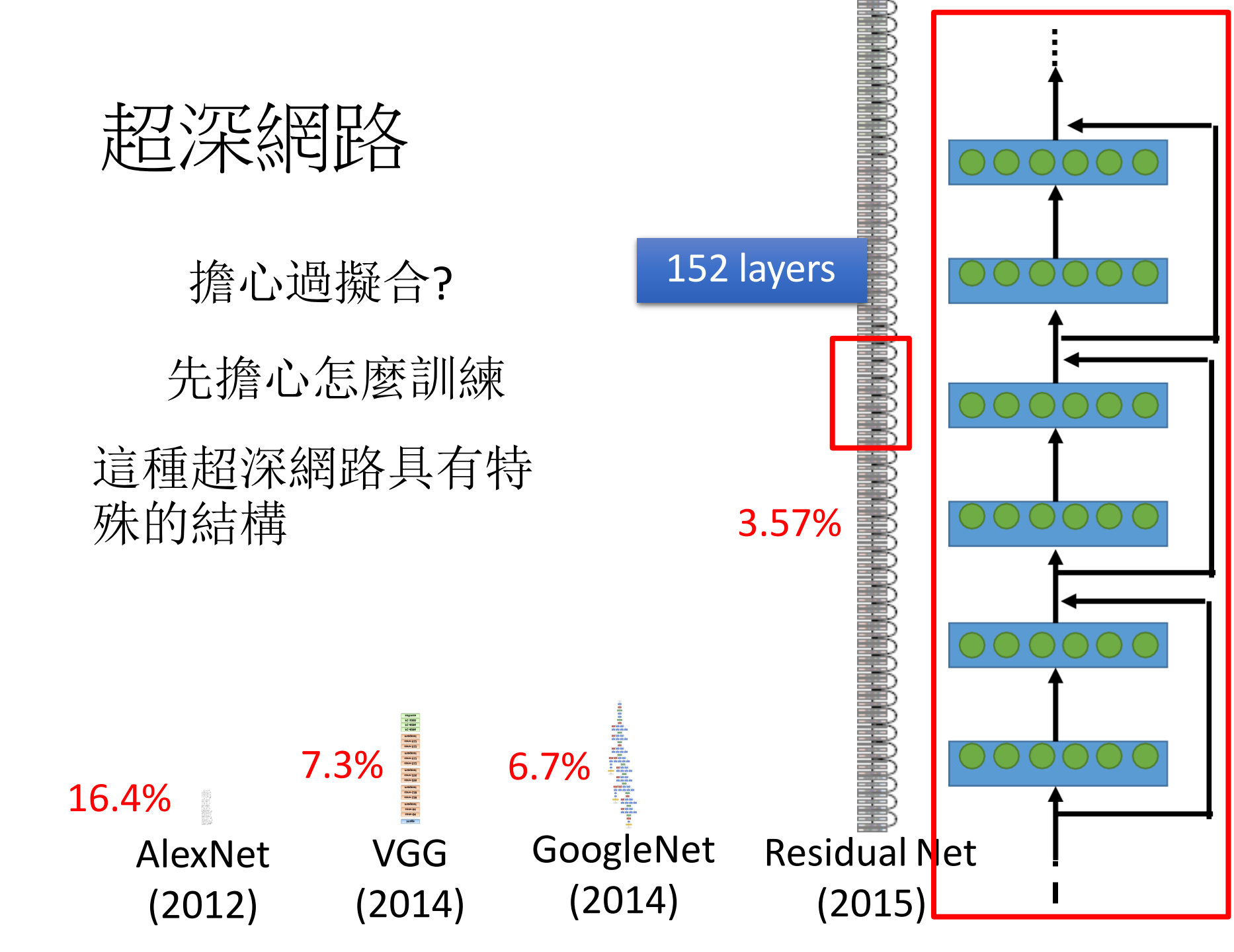

101 layers
152 layers
3.57%
7.3%
6.7%
16.4%
AlexNet
(2012)
VGG
(2014)
GoogleNet
(2014)
Residual Net
(2015)
Taipei
101

超深網路

擔心過擬合?

先擔心怎麼訓練

這種超深網路具有特殊的結構



超深網路

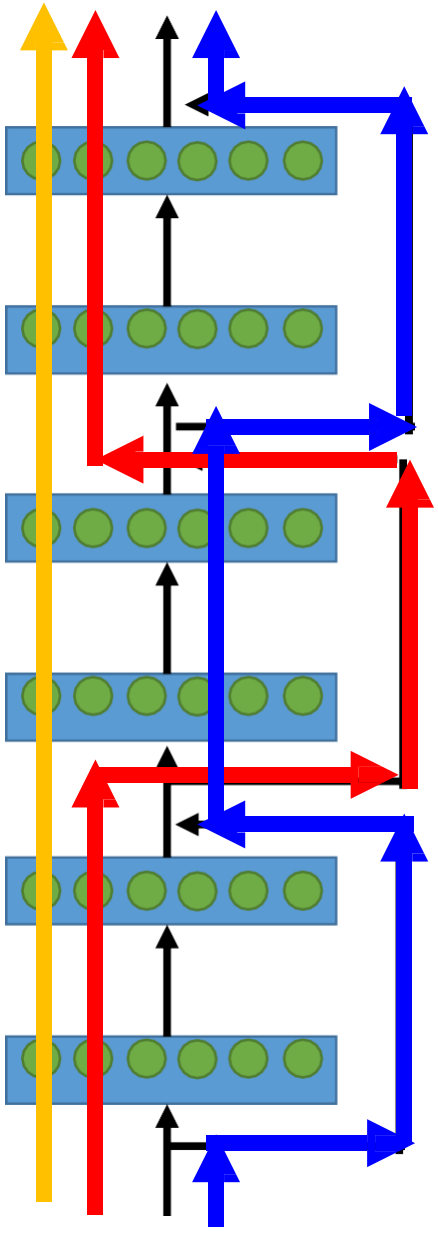
- 超深網路是許多具有不同深度的網路的集成

Ensemble

6 layers

4 layers

2 layers



Residual Networks are Exponential Ensembles of Relatively Shallow Networks
https://arxiv.org/abs/1605.06431

超深網路

- 分形網路結構

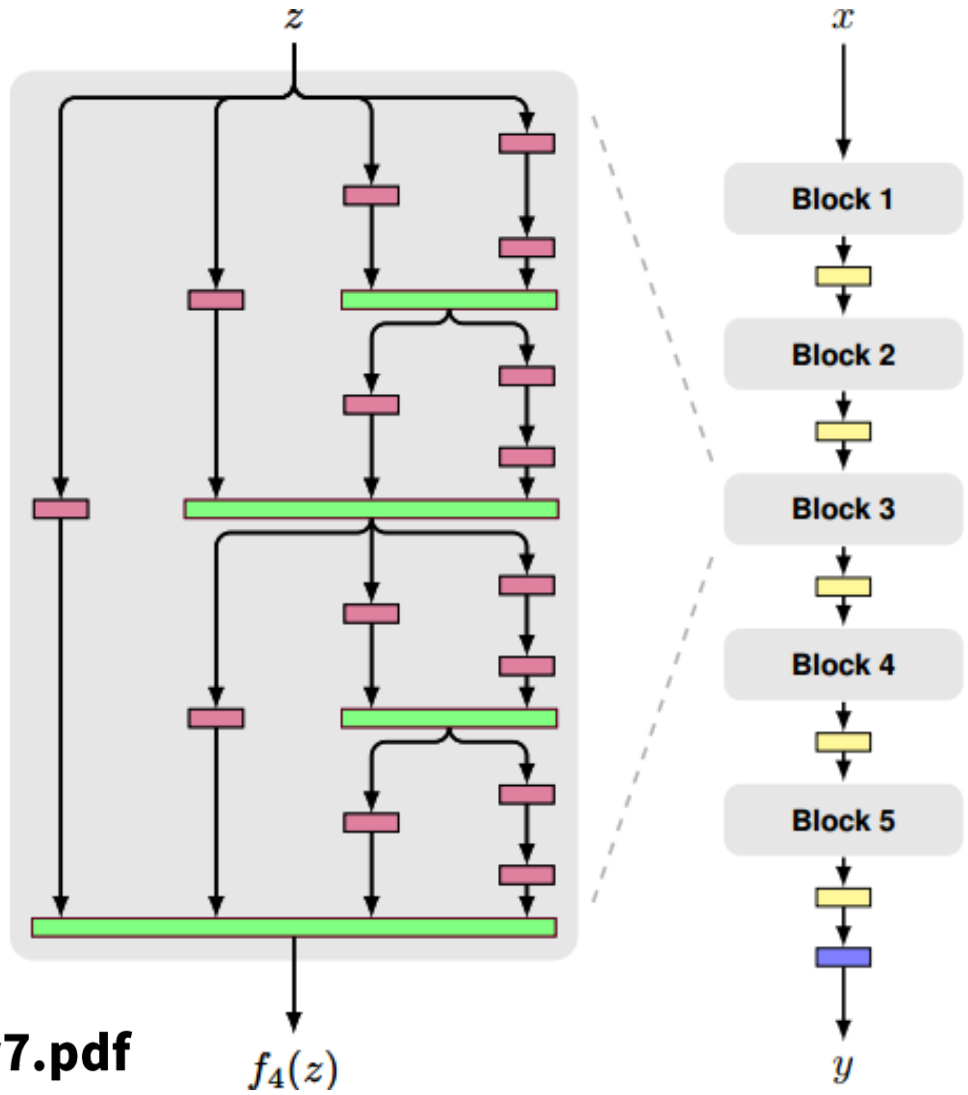
FractalNet: Ultra-Deep Neural Networks without Residuals
https://arxiv.org/abs/1605.07648

Resnet in Resnet

Resnet in Resnet: Generalizing Residual Architectures
https://arxiv.org/abs/1603.08029

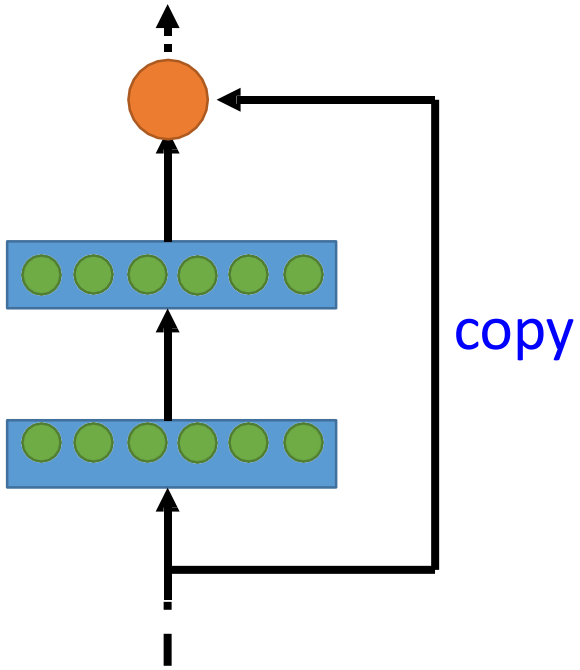
Good Initialization?

All you need is a good init
http://arxiv.org/pdf/1511.06422v7.pdf



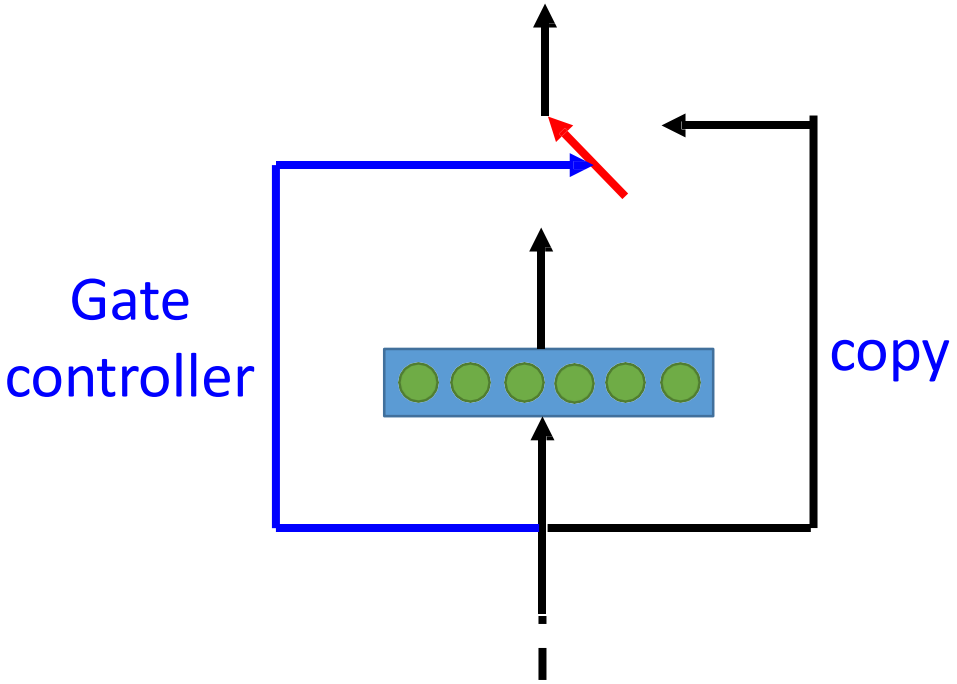
Ultra Deep Network

殘差網路

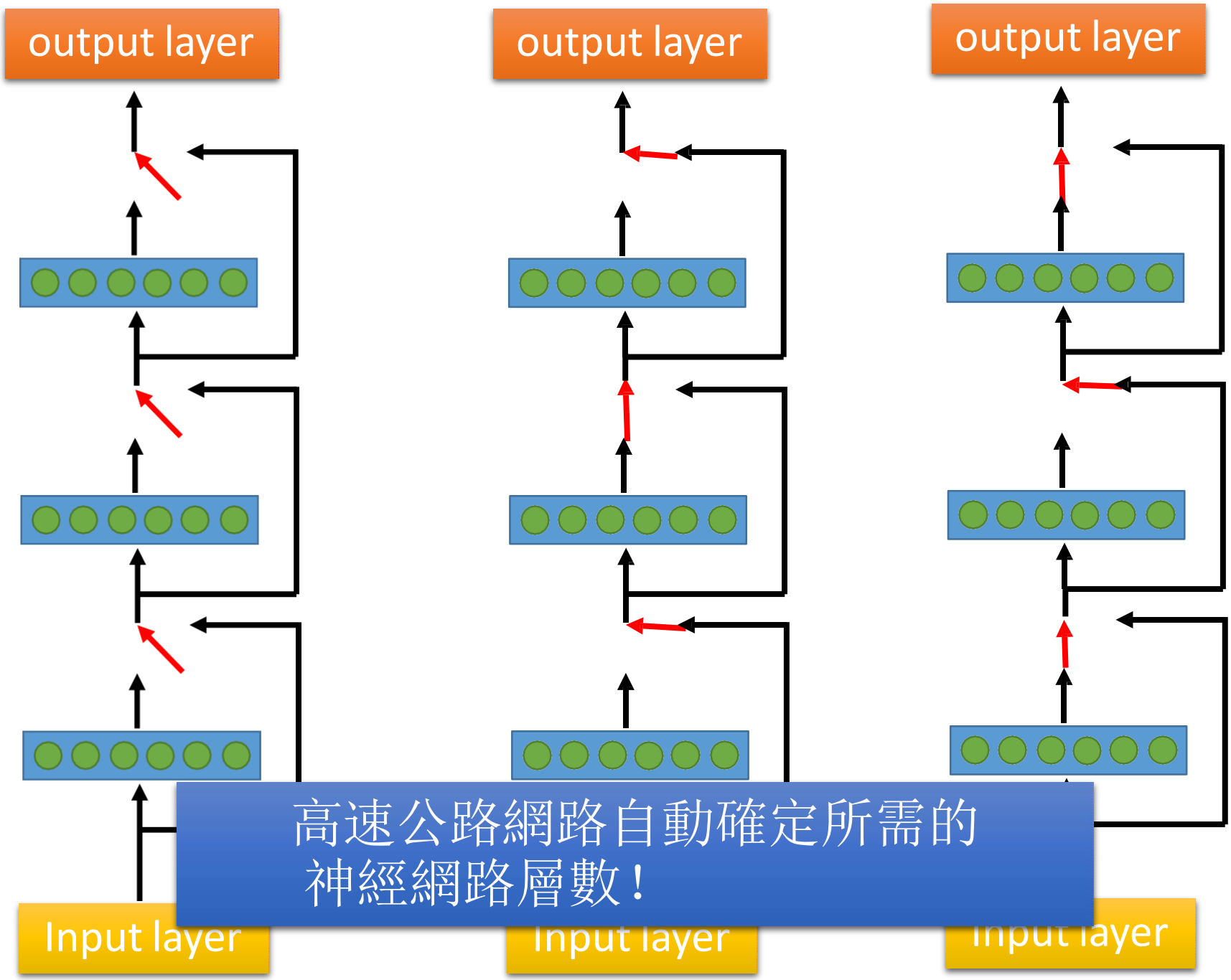


Deep Residual Learning for Image Recognition
http://arxiv.org/abs/1512.03385

高速公路網路



Training Very Deep Networks
https://arxiv.org/pdf/1507.06228v2.pdf


output layer
output layer
output layer
高速公路網路自動確定所需的
神經網路層數！
Input layer
Input layer
Input layer

大綱

監督學習

- 超深網路
- 注意力模型

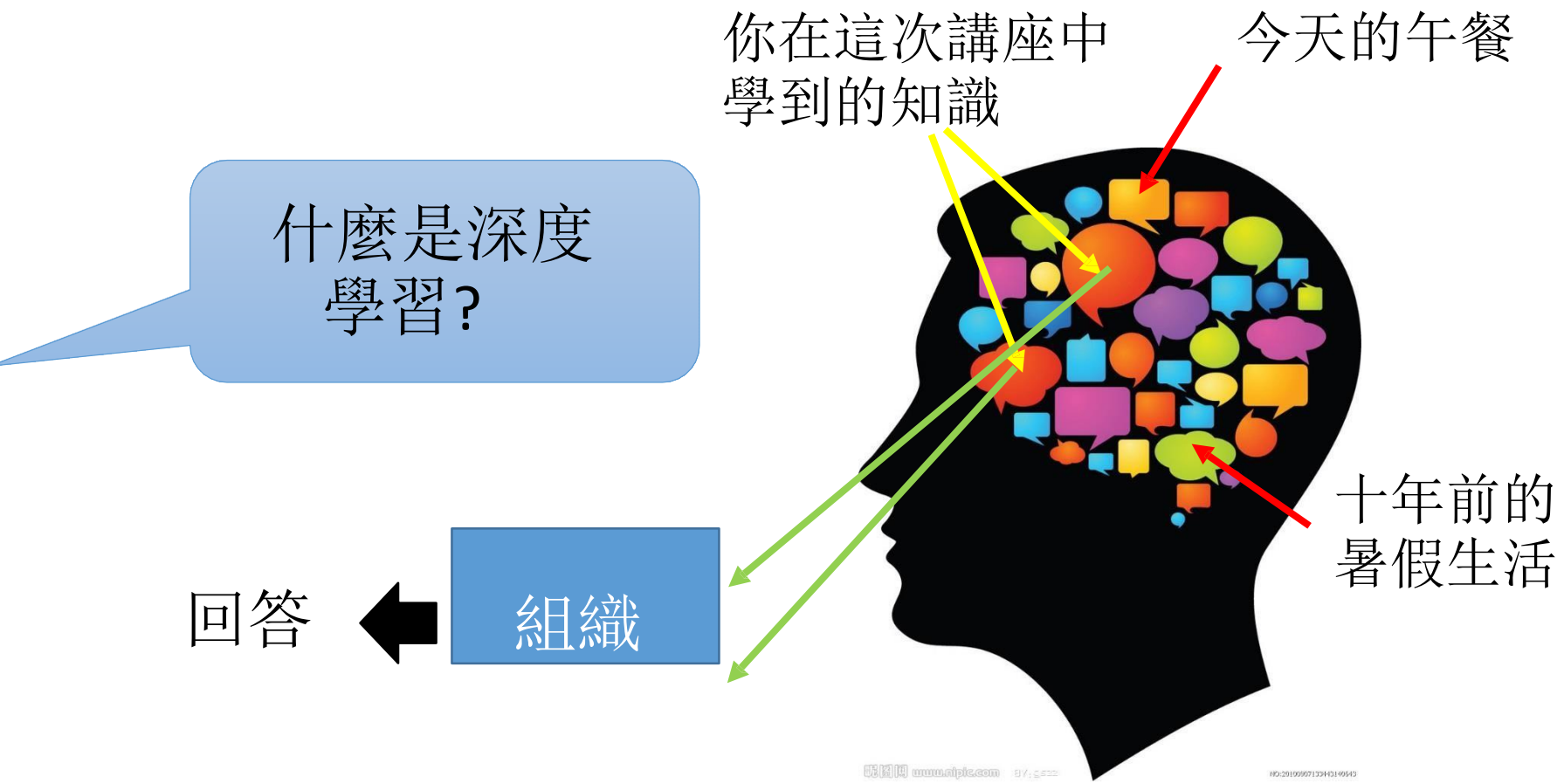
} 新型神經網路結構

強化學習

無監督學習

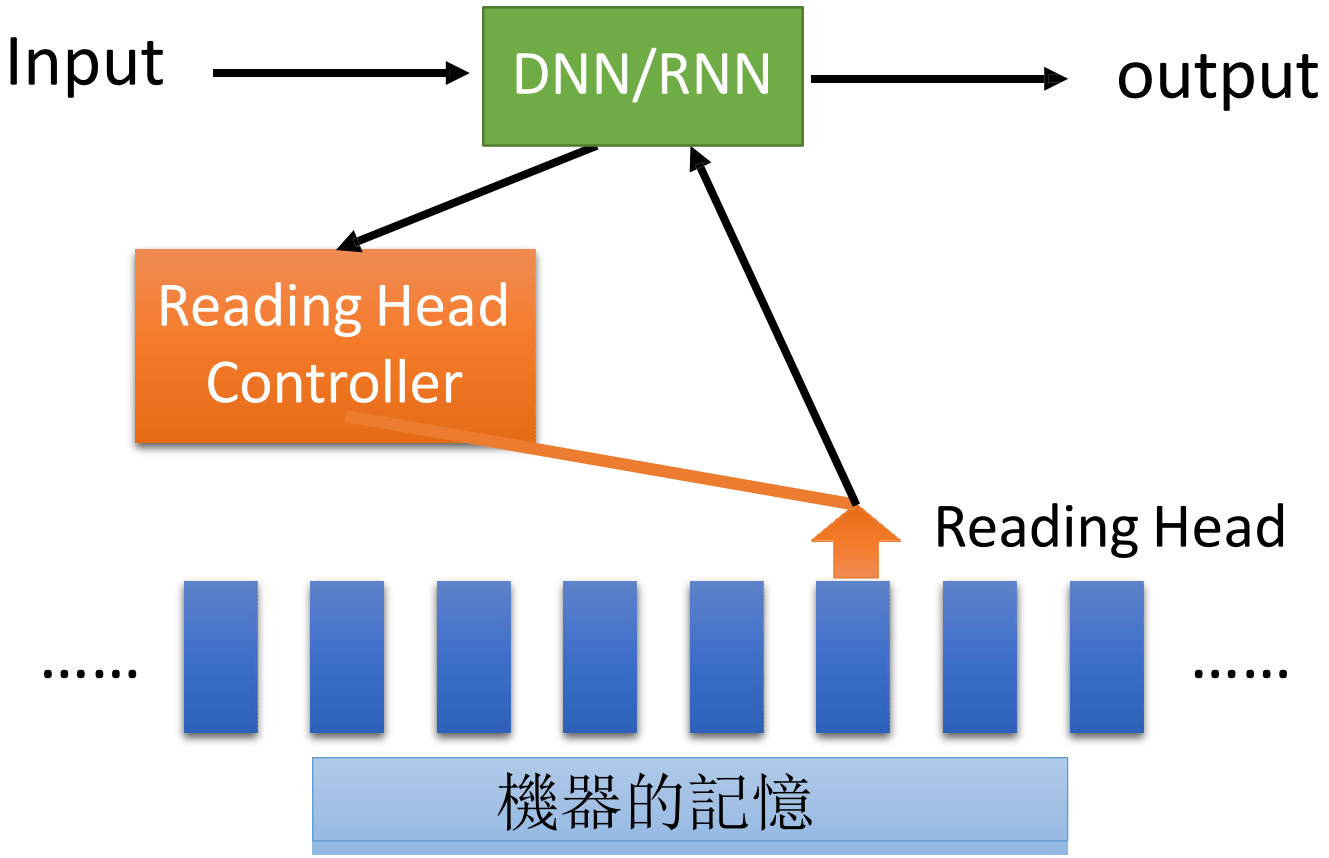
- 圖像: 認識到世界是什麼樣
- 文本: 理解單詞的含義
- 視頻: 在無監督的情況下學習人類的語言

基於注意力的模型

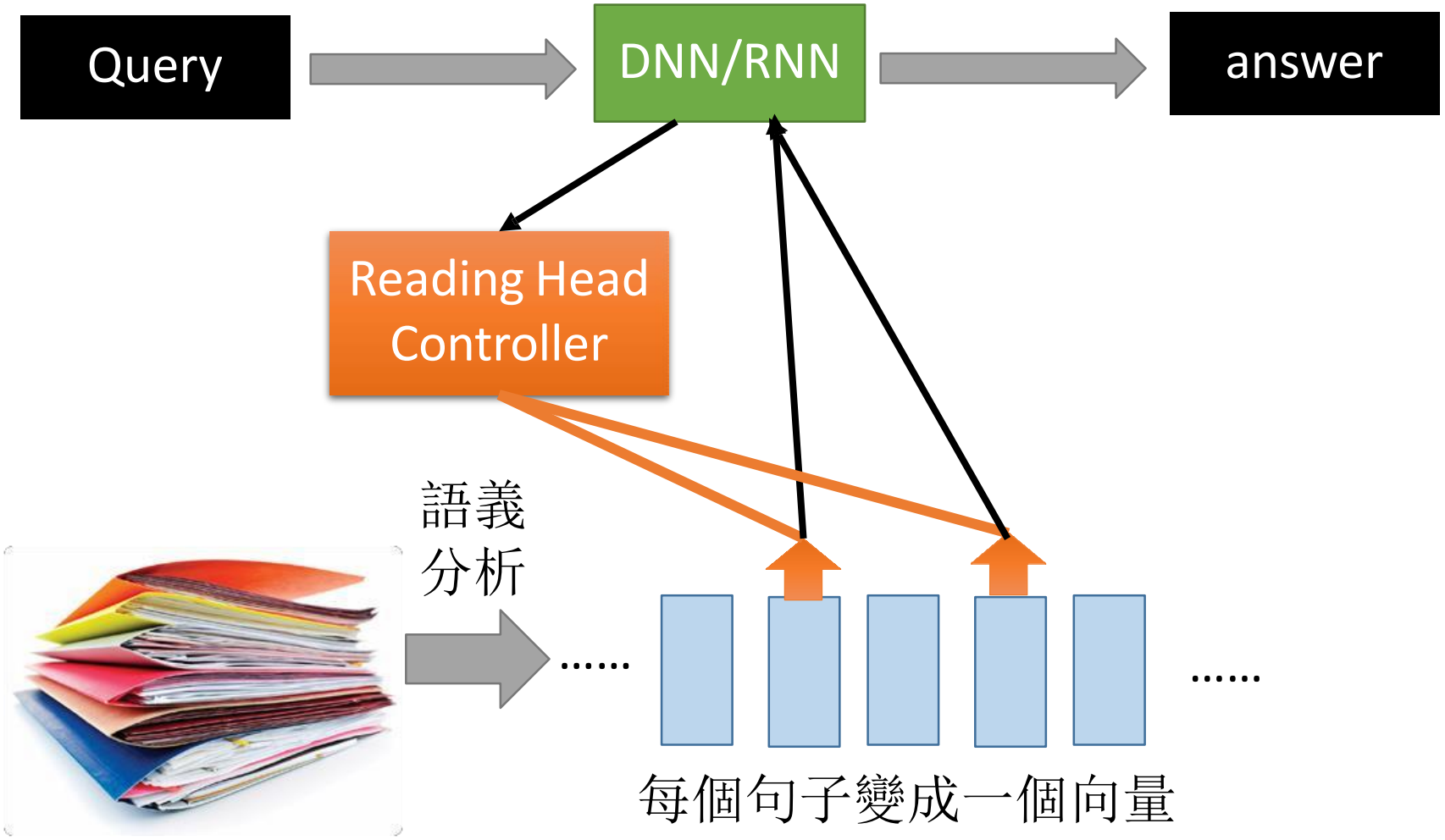


http://henrylo1605.blogspot.tw/2015/05/blog-post_56.html

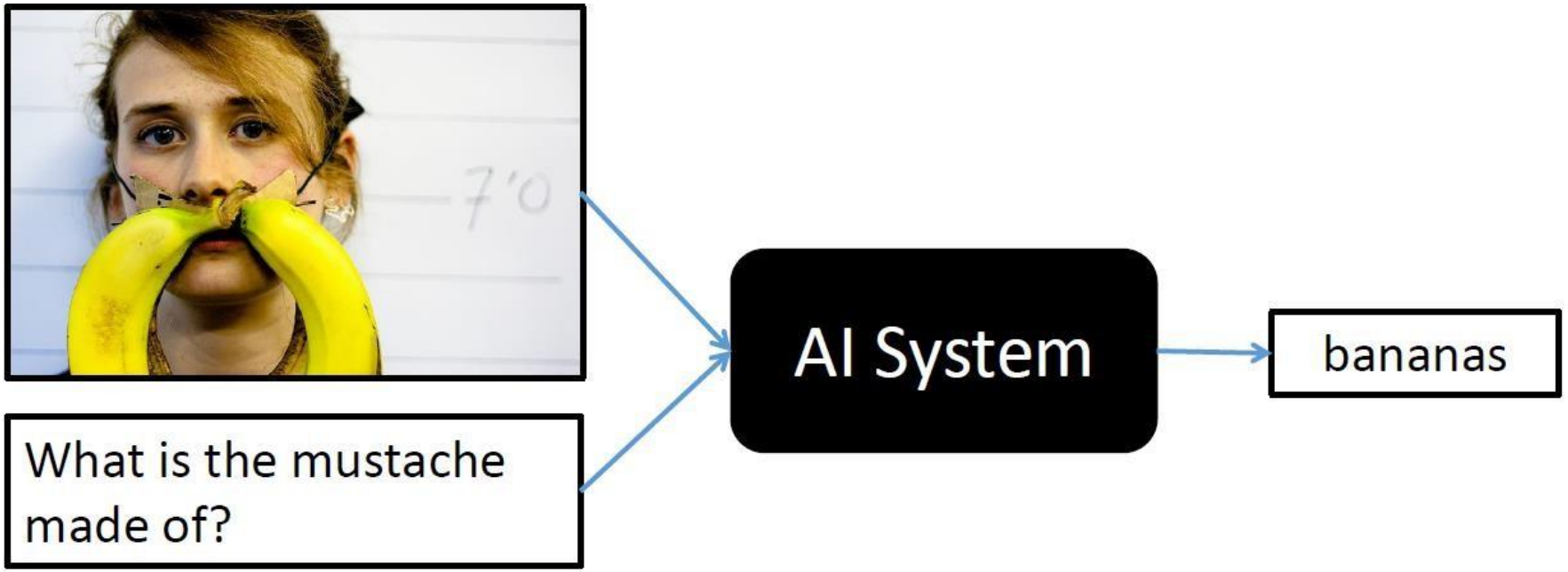
基於注意力的模型


Input
DNN/RNN
output
Reading Head Controller
Reading Head
.....
.....
機器的記憶

閱讀理解

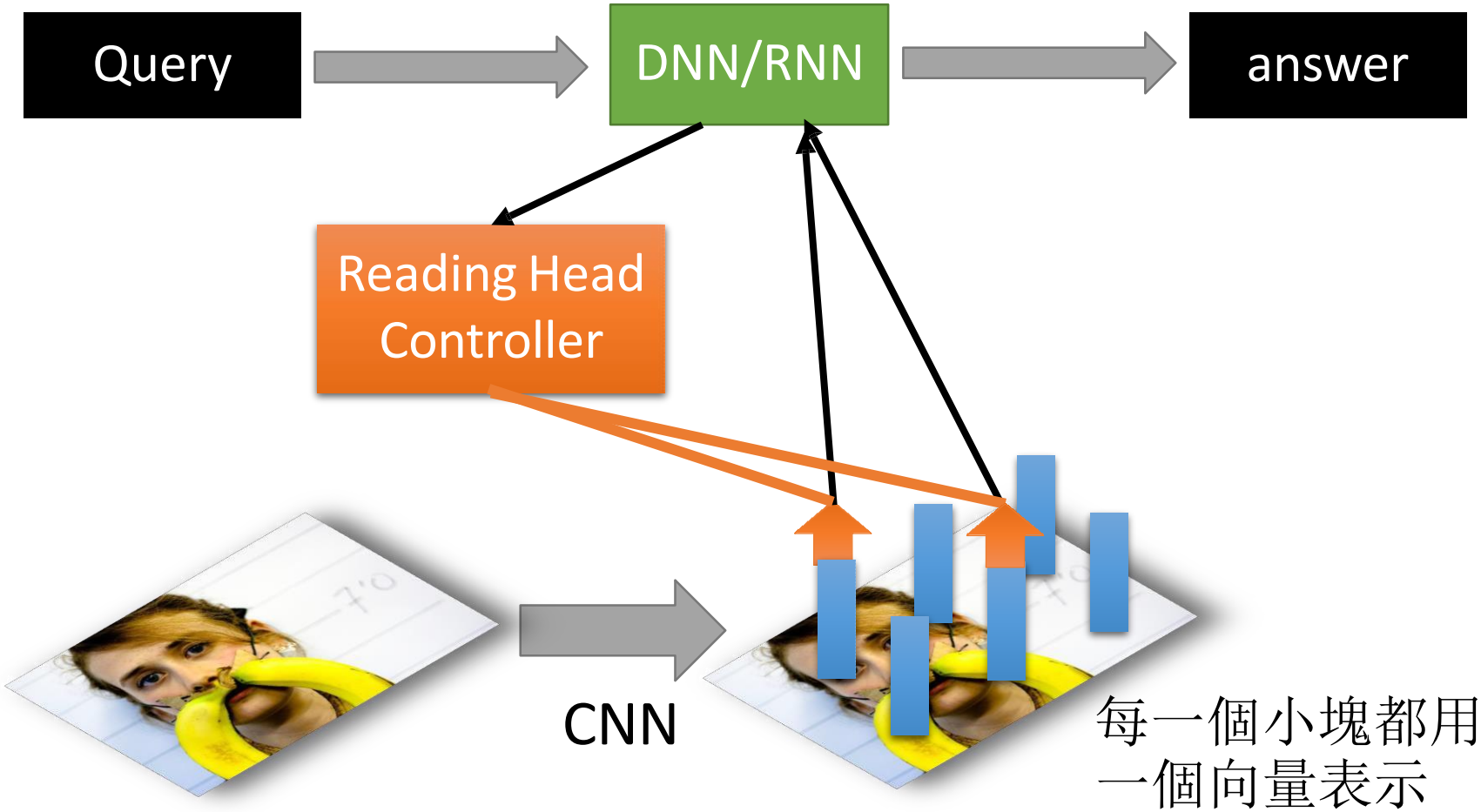

Query
DNN/RNN
answer
Reading Head Controller
語義分析
.....
.....
每個句子變成一個向量

看圖回答



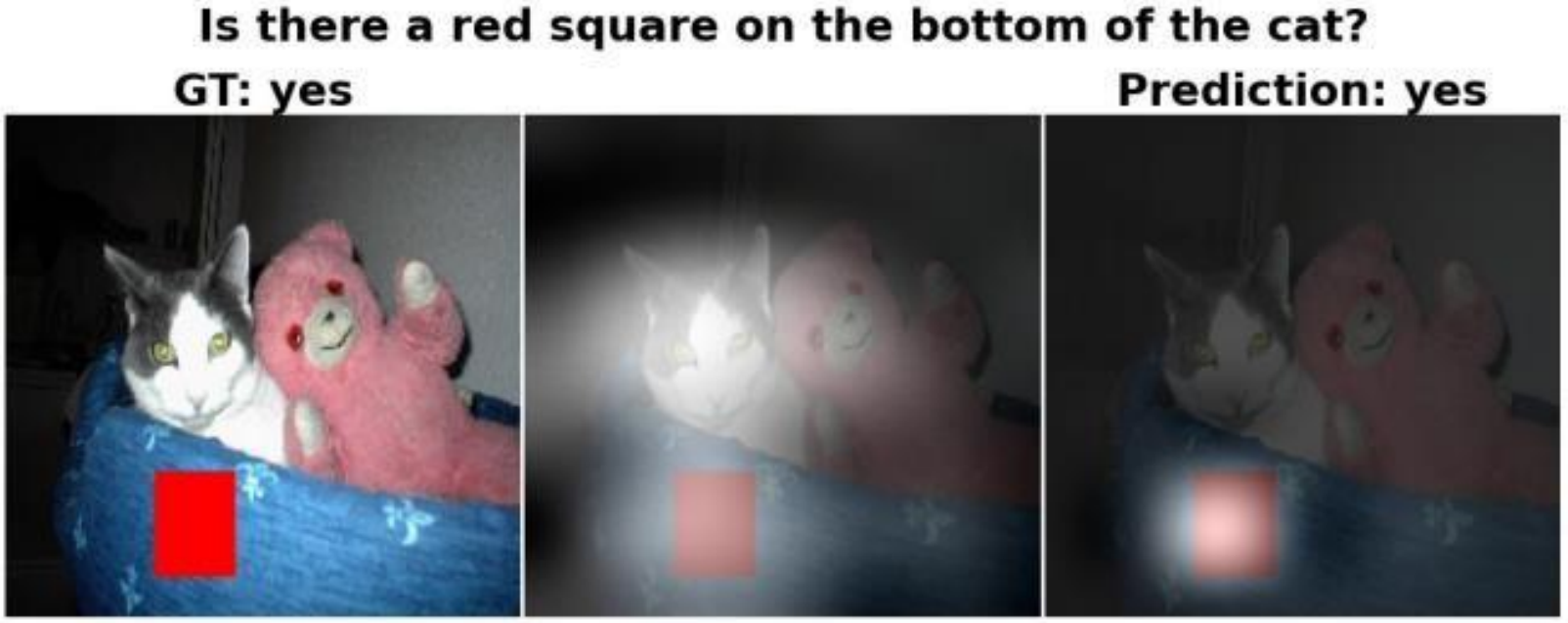
source: http://visualqa.org/

看圖回答


Query
DNN/RNN
answer
Reading Head Controller
CNN
每一個小塊都用一個向量表示

看圖回答

- Huijuan Xu, Kate Saenko. Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering. arXiv Pre-Print, 2015



語音回答

- **托福聽力理解考試**
- Example:

  Audio Story:  (The original story is 5 min long.)

  Question: “ What is a possible origin of Venus’ clouds? ”

  Choices:

  (A) gases released as a result of volcanic activity

  (B) chemical reactions caused by high surface temperatures

  (C) bursts of radio energy from the plane's surface

  (D) strong winds that blow dust into the atmosphere

大綱

監督學習

- 超深網路
- 注意力模型

} 新型神經網路結構

強化學習

無監督學習

- 圖像: 認識到世界是什麼樣
- 文本: 理解單詞的含義
- 視頻: 在無監督的情況下學習人類的語言

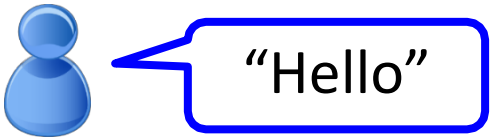
強化學習的情景

強化學習的情景

監督學習 v.s. 強化學習

- 監督學習

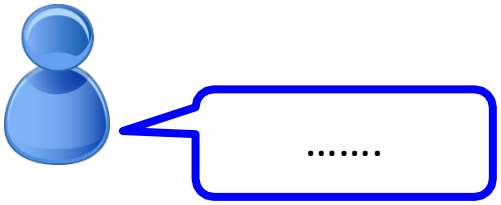
跟老師學習

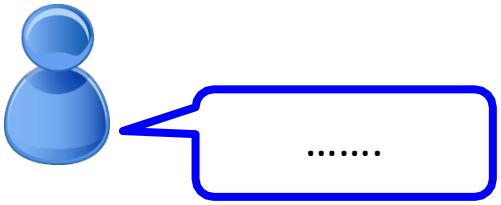


Say "Hi"

Say "Good bye"

- 強化學習

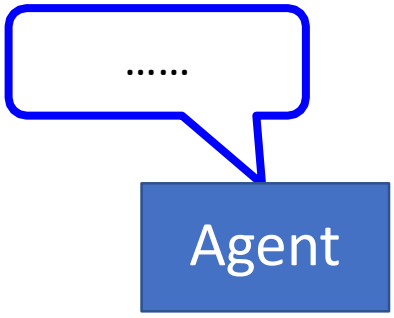




.....

Bad

從批評中學習



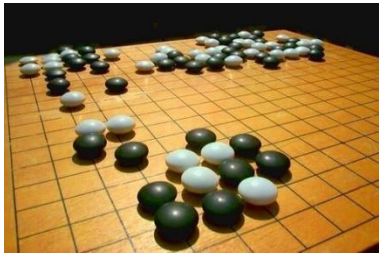
強化學習的情景


智能體(Agent) 學會採取行動
最大化預期回報
觀察
動作
AlphaGo
Reward
下一次移動
If win, reward = 1
If loss, reward = -1
Otherwise, reward = 0
環境

監督學習 v.s. 強化學習

- 監督學習:

Next move:
“5-5”



Next move:
“3-3”

- 強化學習:

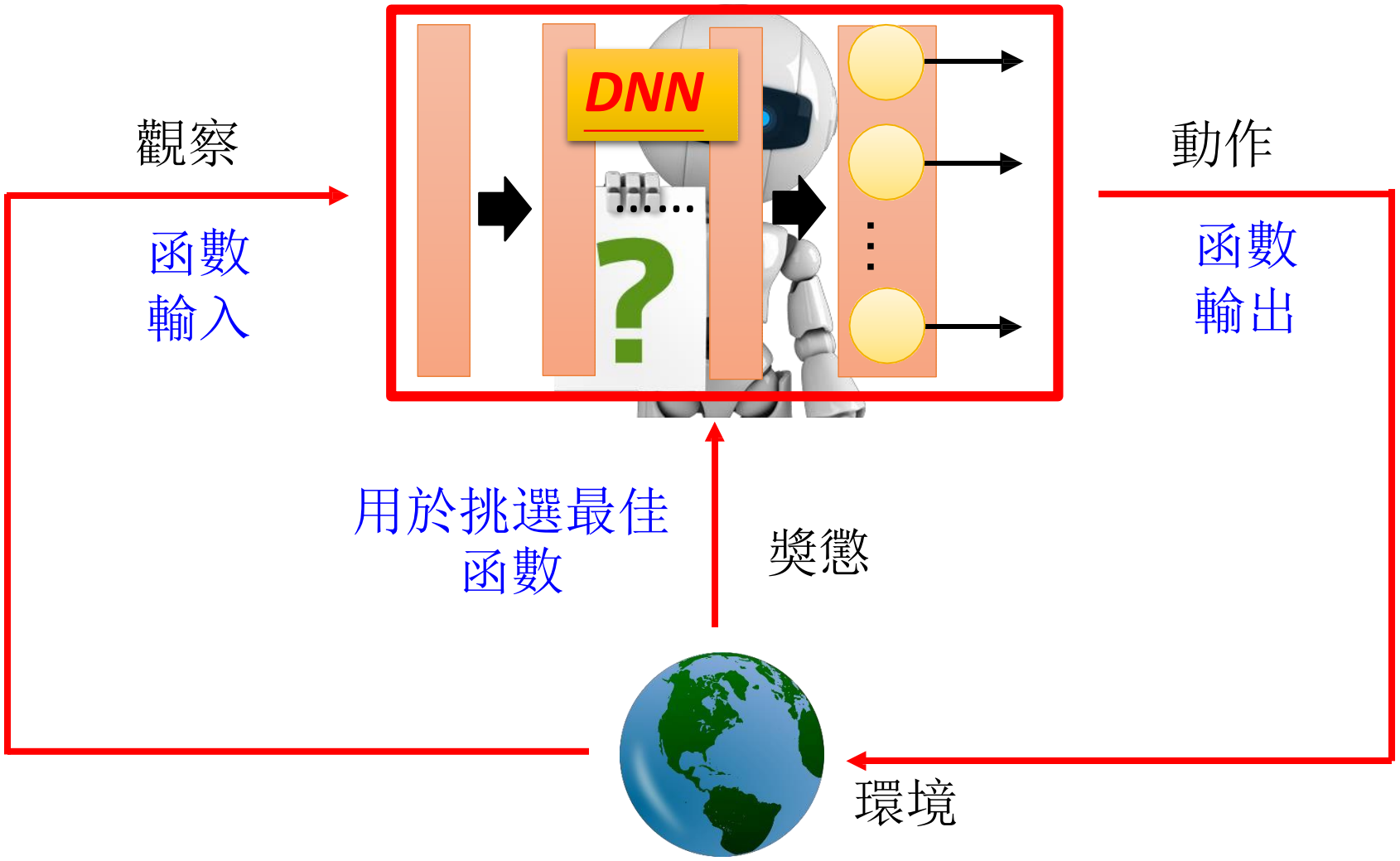
First move → …… many moves …… → Win!

Alpha Go 是監督學習+ 強化學習

強化學習的難題

- 為了獲得更長期的回報，最好是犧牲眼前的回報
  - E.g. Playing Go
- 智慧體的操作會影響它接收到的後續資料
  - E.g. Exploration

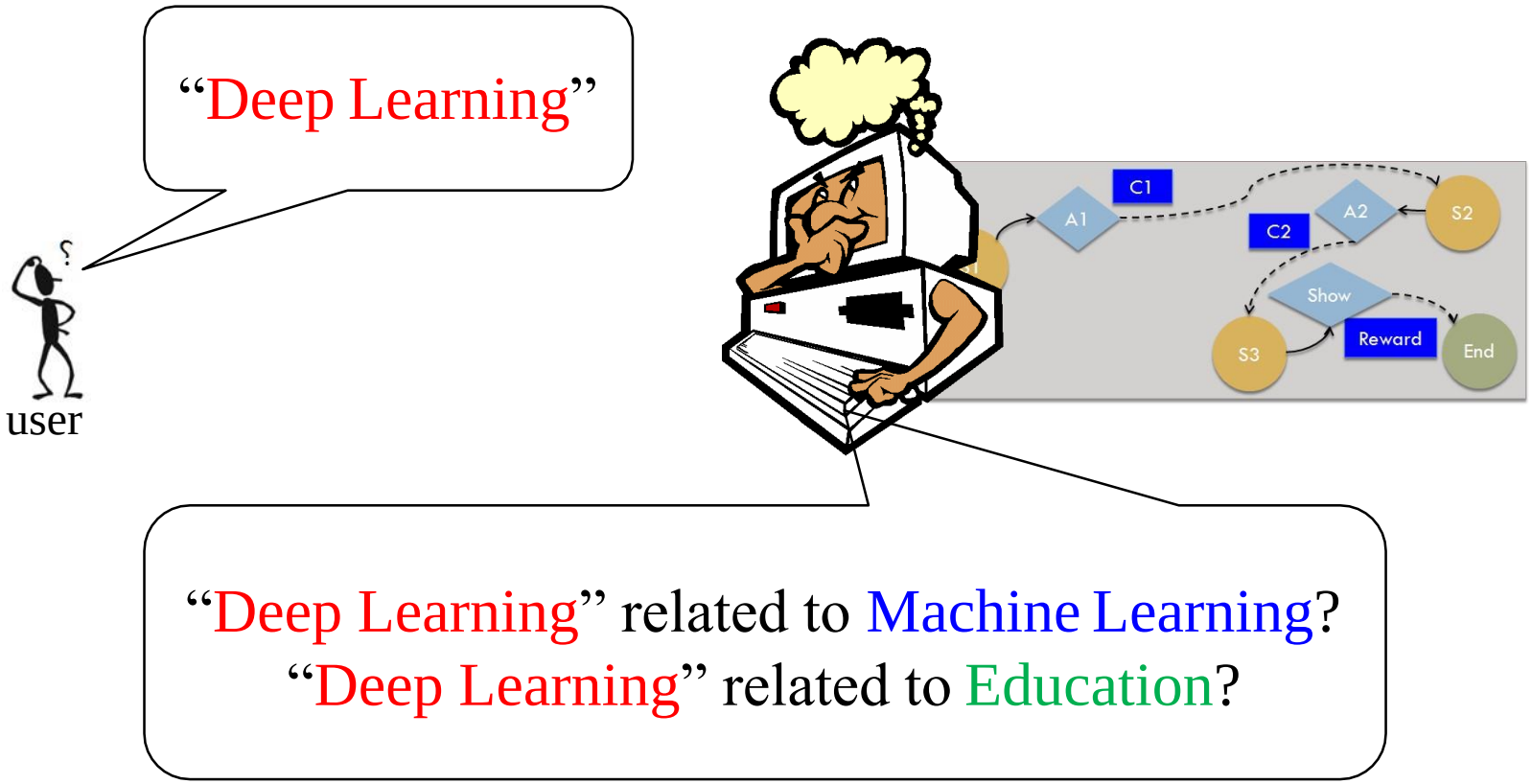
深度強化學習



應用: 互動式檢索

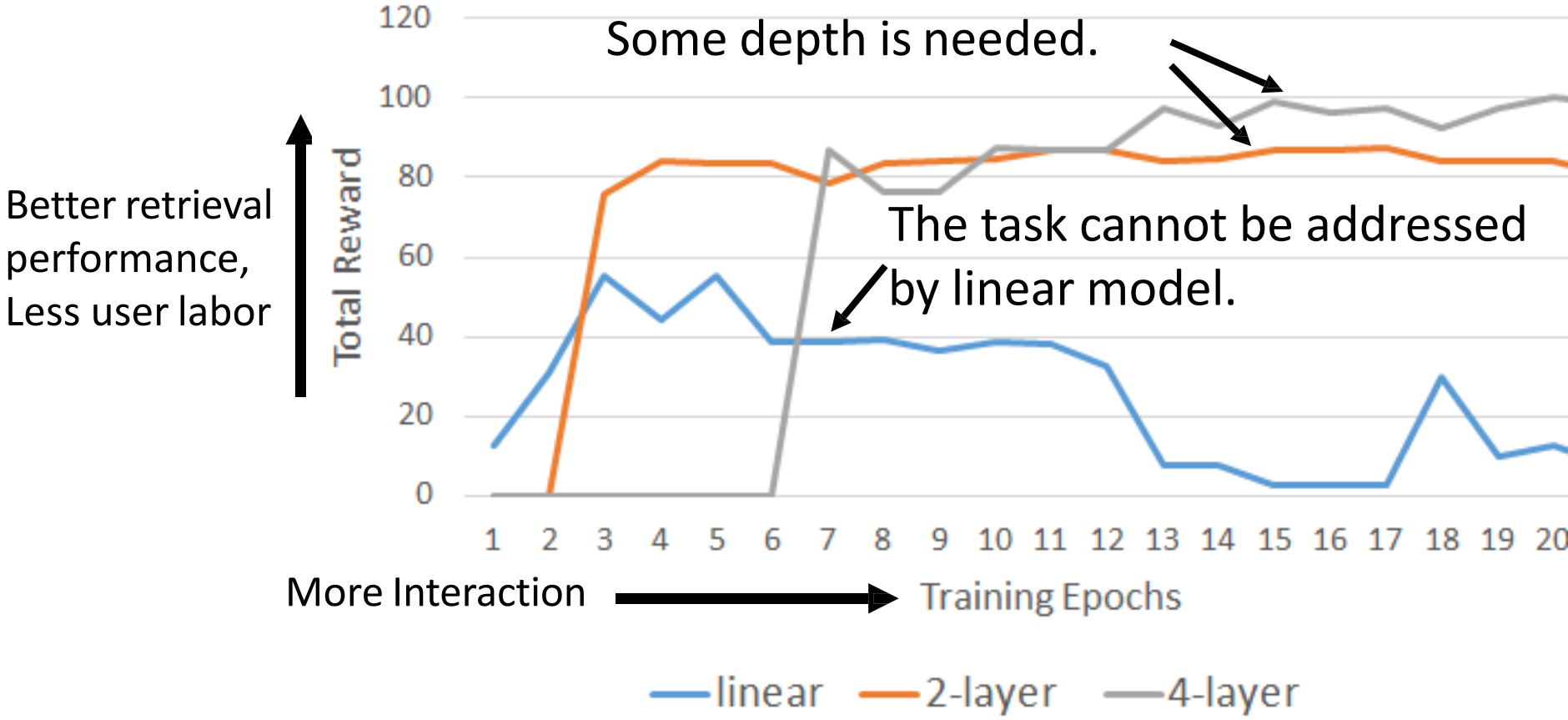
- 互動式檢索是非常有幫助的

[Wu & Lee, INTERSPEECH 16]



深度強化學習

- 不同的網路深度



更多的應用

- Alpha Go, 玩電子遊戲, 對話
  - 直升機飛行 https://www.youtube.com/watch?v=0JL04JJjocc
- Driving
  - https://www.youtube.com/watch?v=0xo1Ldx3L5Q
- Google Cuts Its Giant Electricity Bill With DeepMind-Powered AI
  - http://www.bloomberg.com/news/articles/2016-07-19/google-cuts-its-giant-electricity-bill-with-deepmind-powered-ai

大綱

Supervised Learning

- Ultra Deep Network
- Attention Model

} New network structure

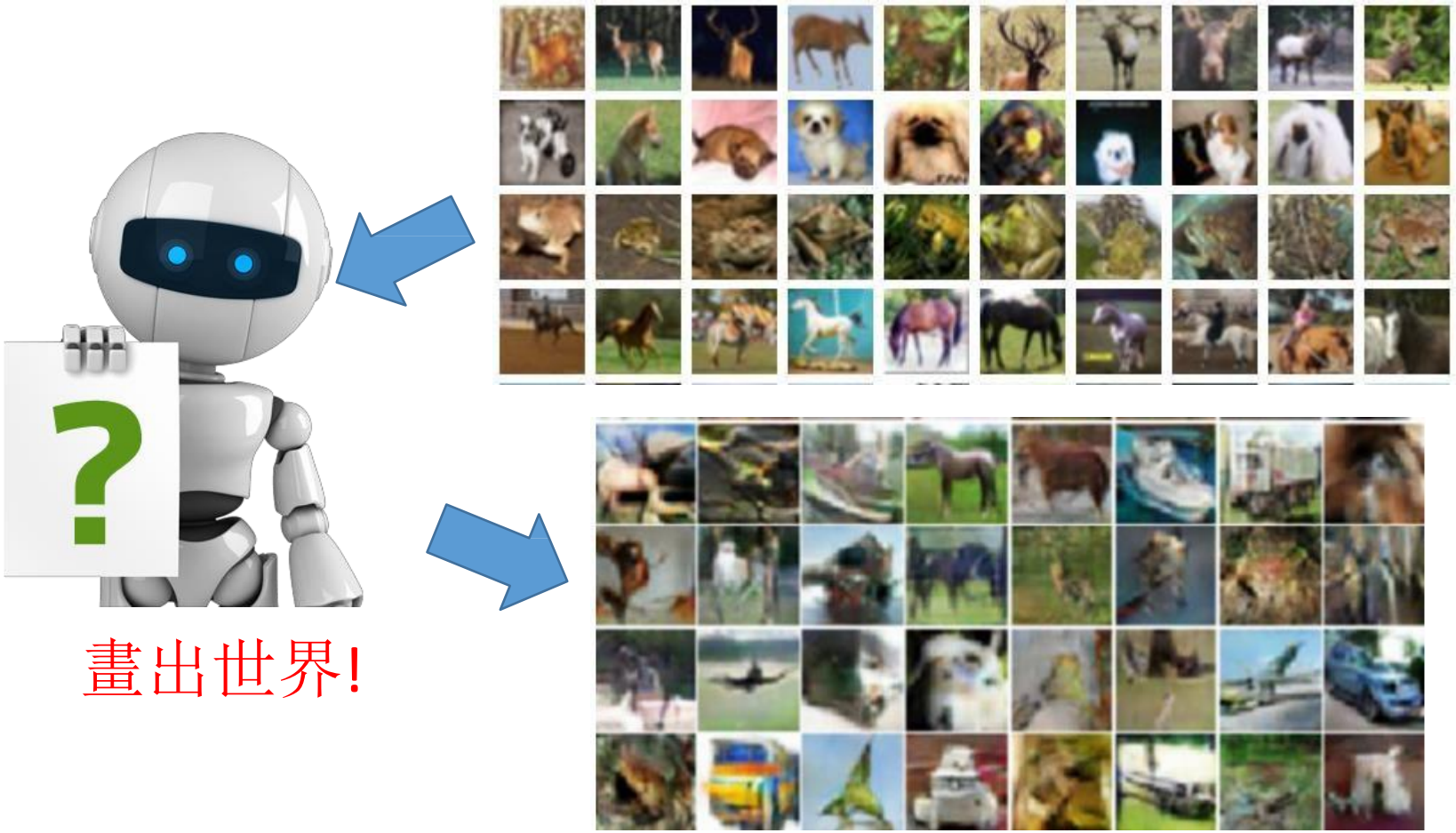
Reinforcement Learning

Unsupervised Learning

- 圖像: 認識到世界是什麼樣
- Text: Understanding the Meaning of Words
- Audio: Learning human language without supervision

機器知道世界是什麼樣子嗎?

Ref: https://openai.com/blog/generative-models/



Deep Dream

- 給定一張照片, 機器添加它所看到的 .....

http://deepdreamgenerator.com/

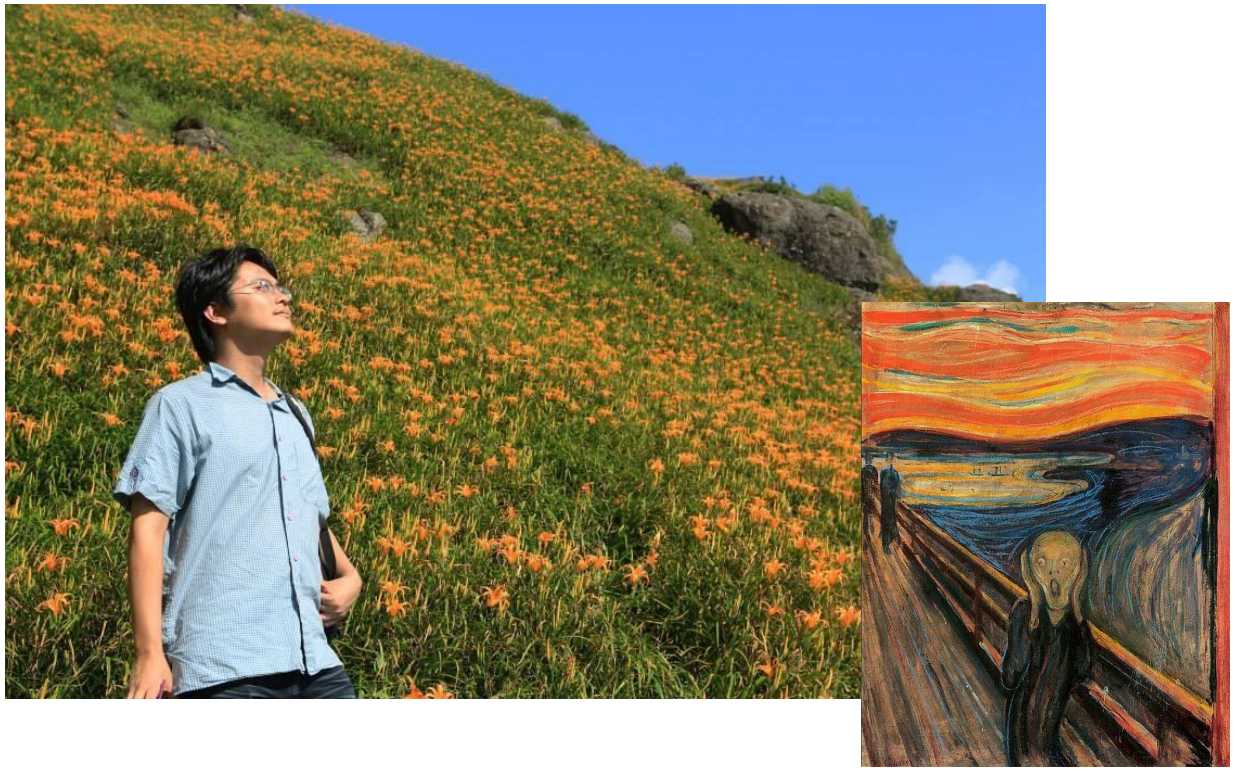
Deep Dream

- 給定一張照片, 機器添加它所看到的 .....

http://deepdreamgenerator.com/

Deep Style

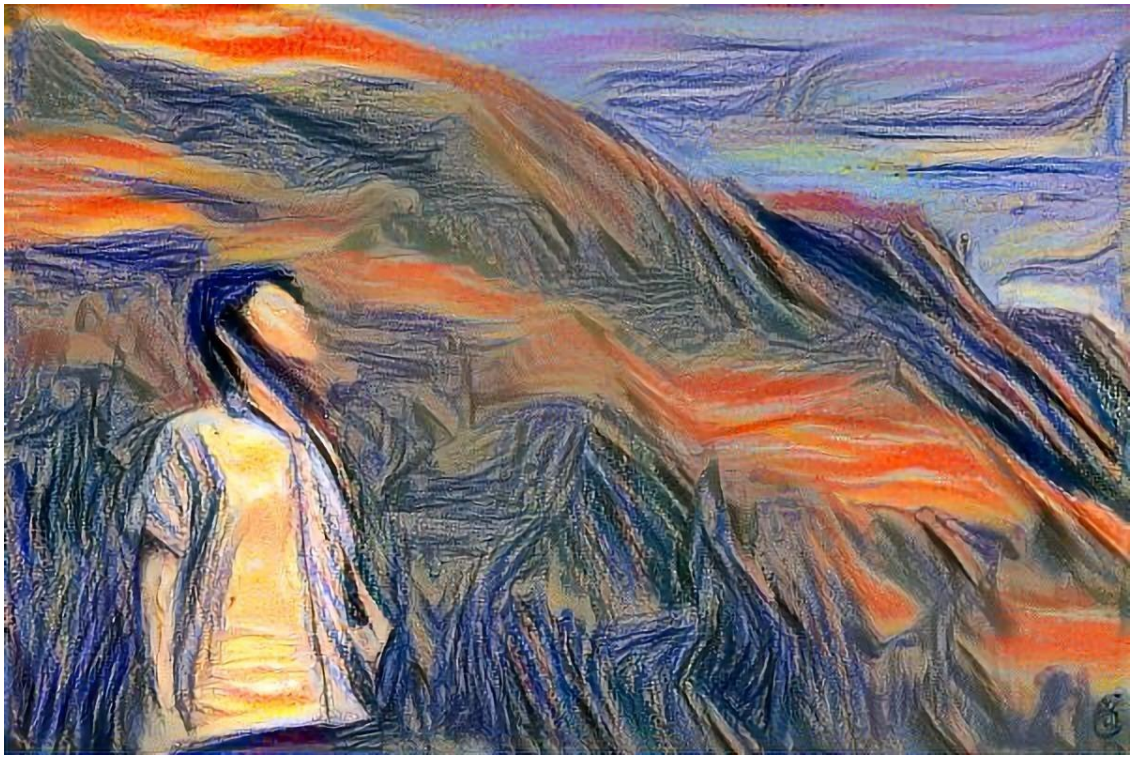
- 給定一張照片，讓它的風格像名畫一樣



https://dreamscopeapp.com/

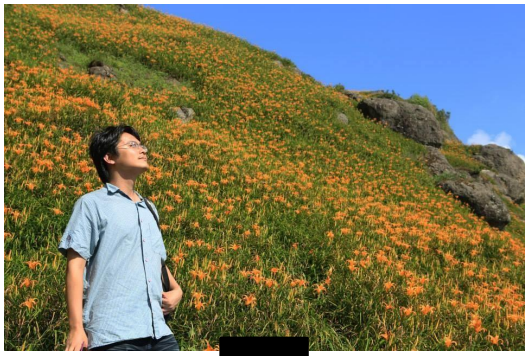
Deep Style

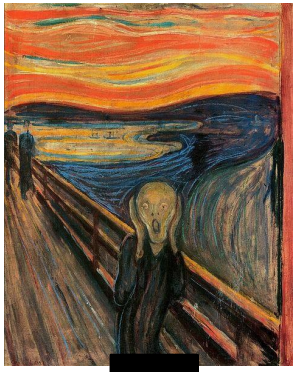
- 給你一張照片，讓它的風格像名畫一樣



https://dreamscopeapp.com/

Deep Style





CNN

CNN

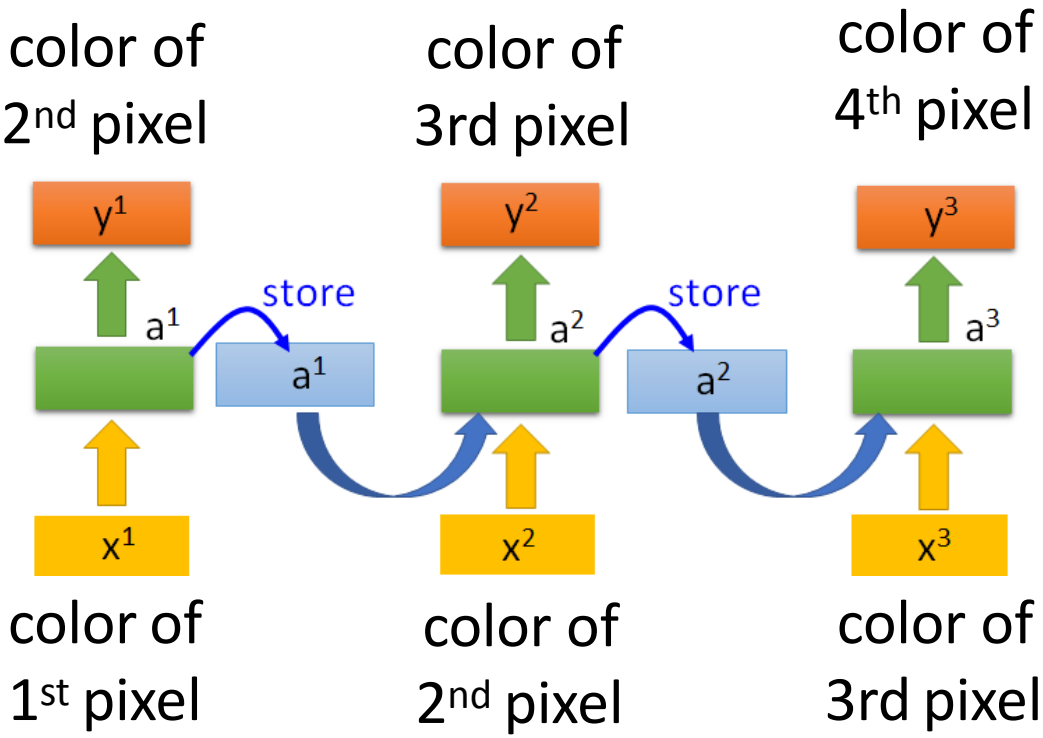
content

style

CNN

?

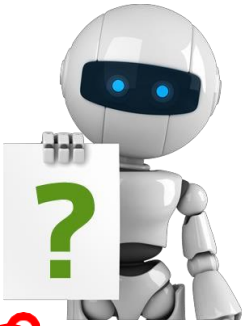
通過RNN生成圖片

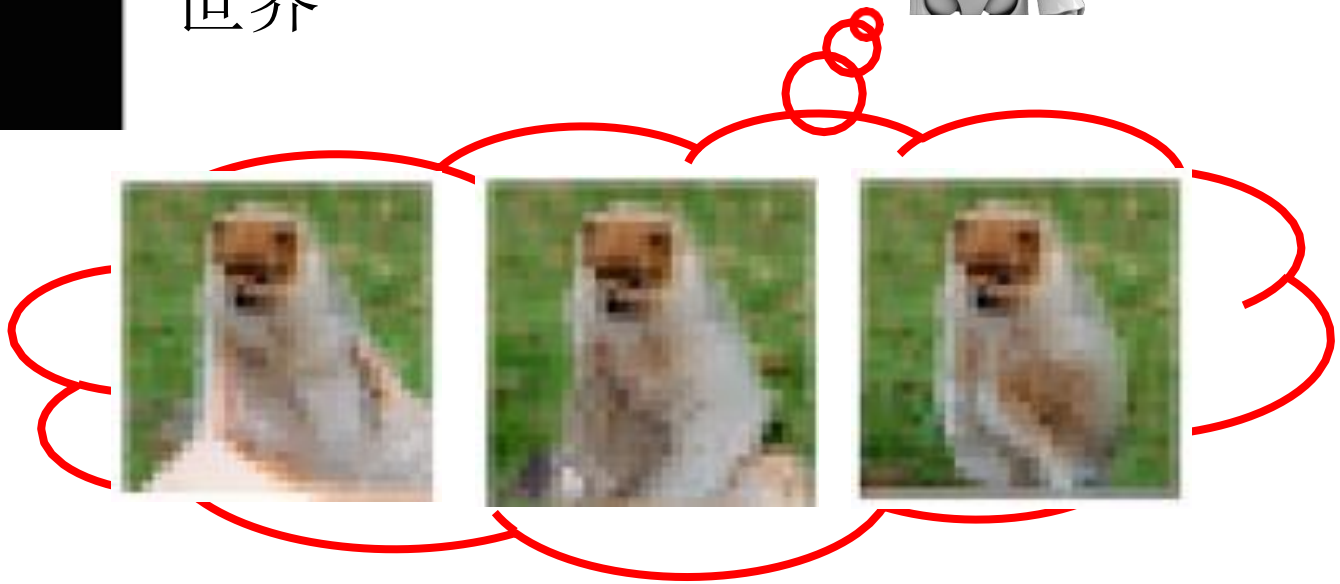

color of 2nd pixel
color of 3rd pixel
color of 4th pixel
y1
y2
y3
store
store
a1
a2
a3
a1
a2
x1
x2
x3
color of 1st pixel
color of 2nd pixel
color of 3rd pixel

通過RNN生成圖片

- **圖元遞迴神經網路**
  - https://arxiv.org/abs/1601.06759

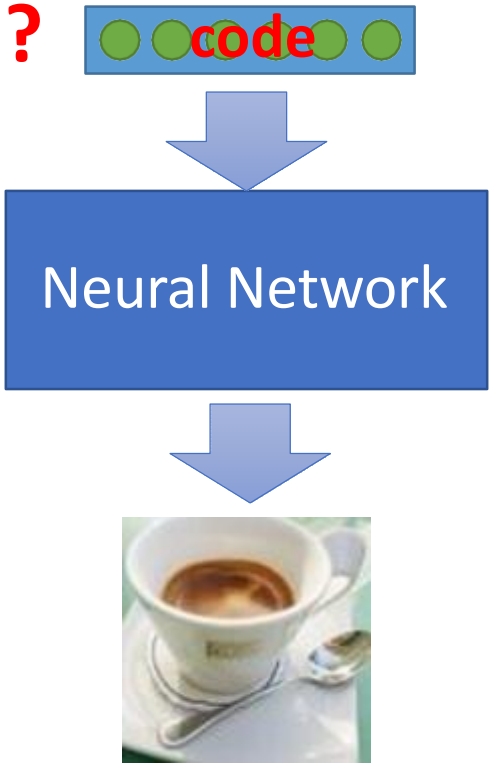
真實
世界



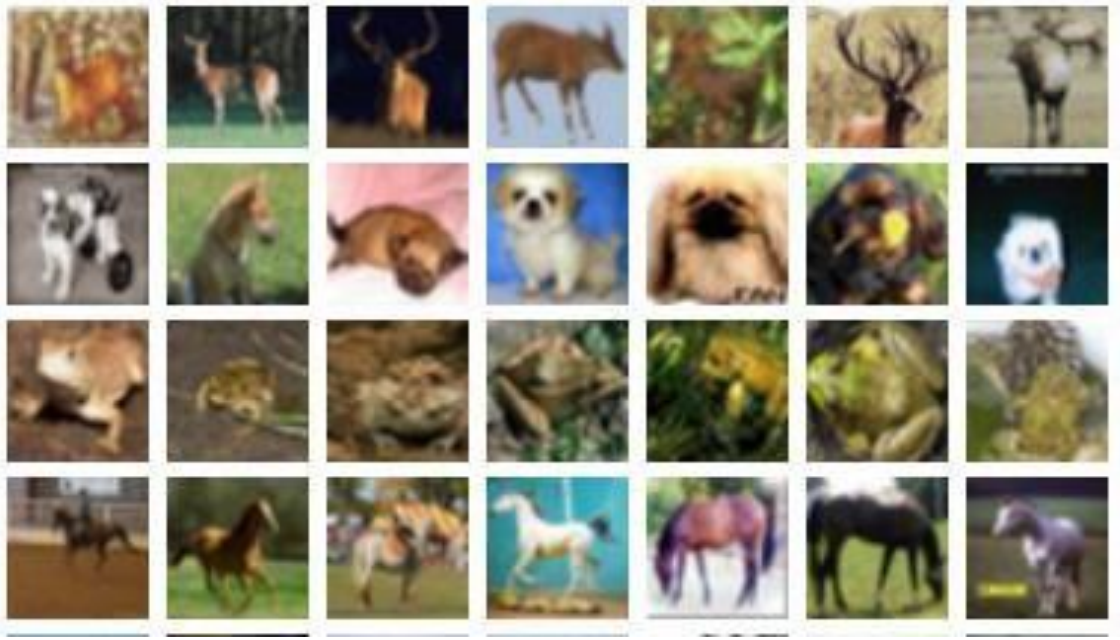


生成圖片

- 訓練解碼器生成圖像是無監督的

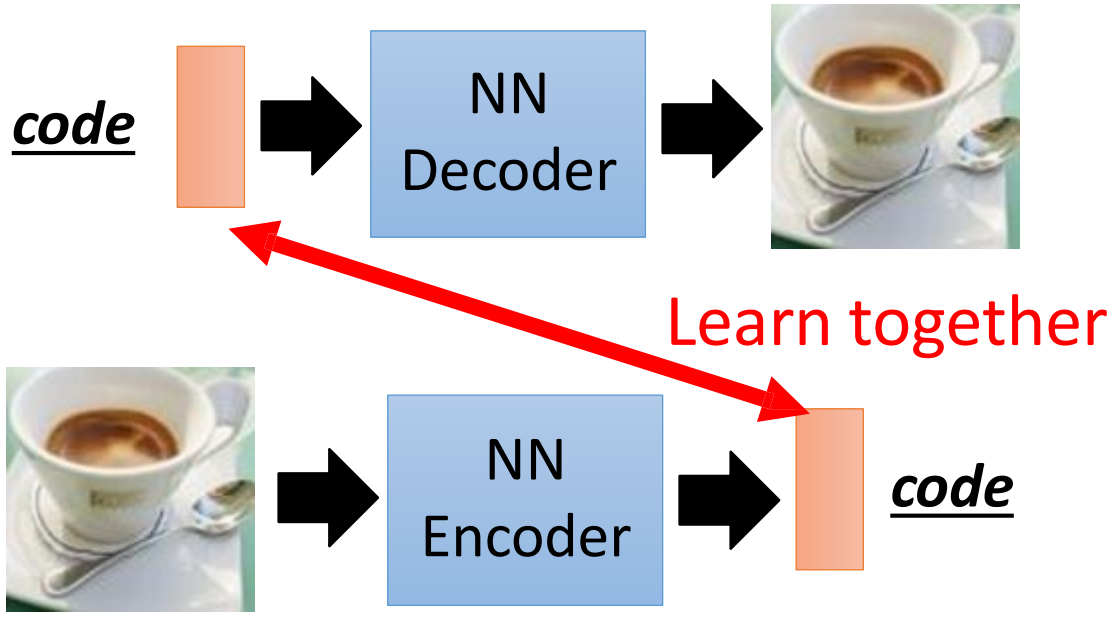


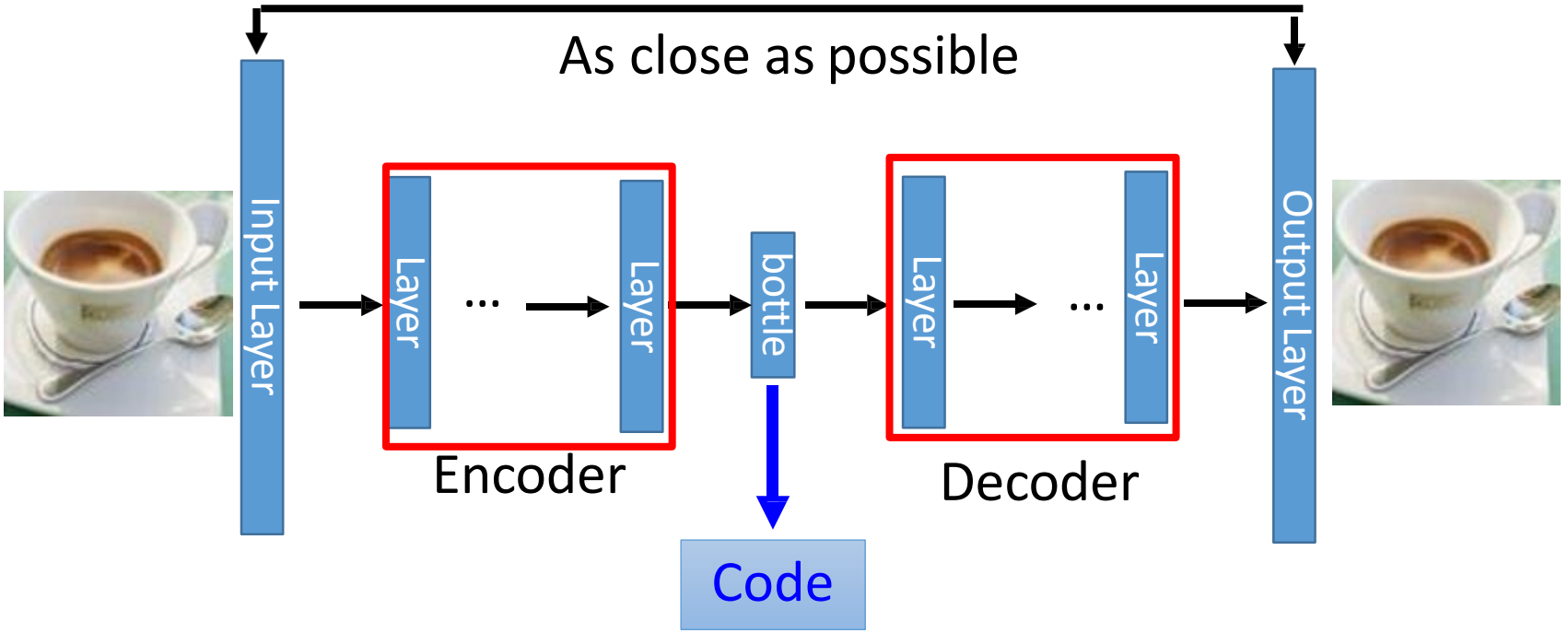
訓練資料是大量的圖片



Auto-encoder

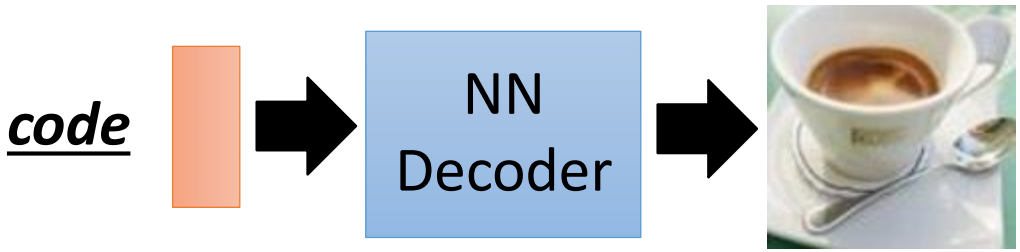
不是最先進的方法



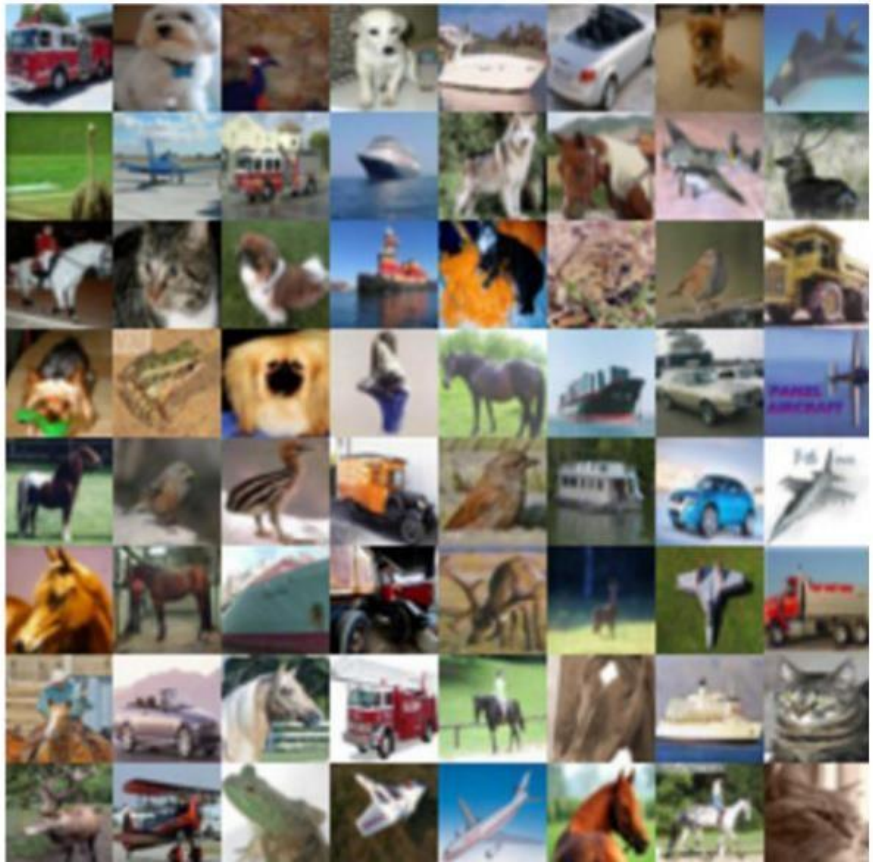


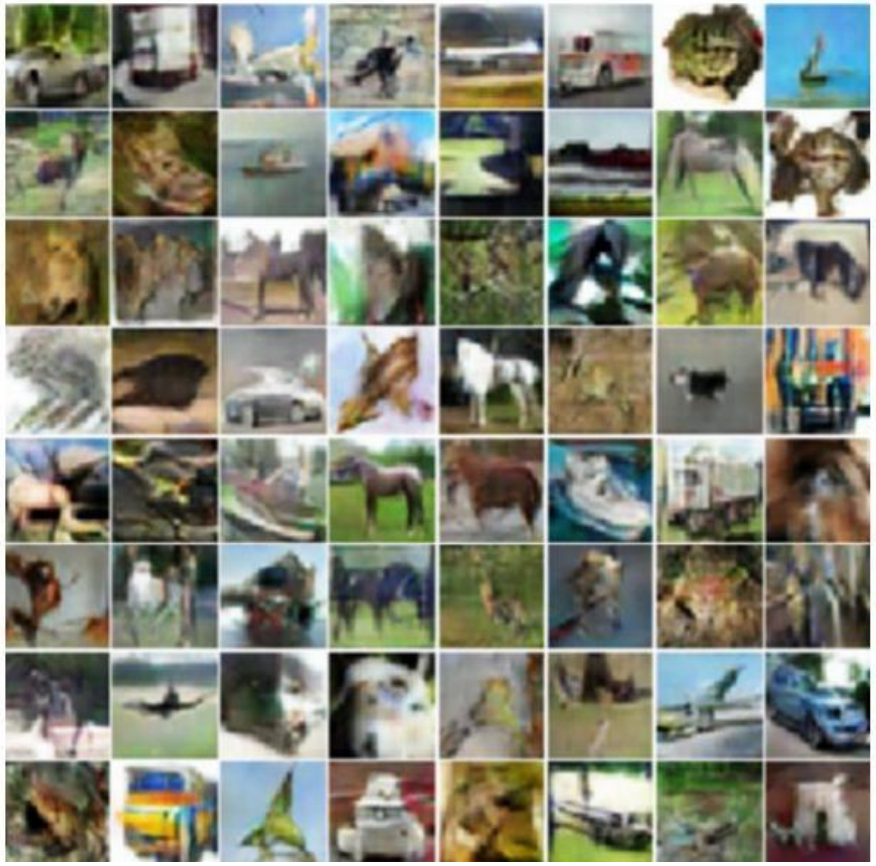
生成圖像

- 訓練解碼器生成圖像是無監督的
- 變分自動編碼器 (VAE)
  - Ref: **Auto-Encoding Variational Bayes, https://arxiv.org/abs/1312.6114**
- 生成對抗網路 (GAN)
  - Ref: **Generative Adversarial Networks, http://arxiv.org/abs/1406.2661**



哪一個機器生成的?





Ref: https://openai.com/blog/generative-models/

畫漫畫!!!

https://github.com/mattya/chainer-DCGAN

大綱

監督學習

- 超深網路
- 注意力模型

} 新型神經網路結構

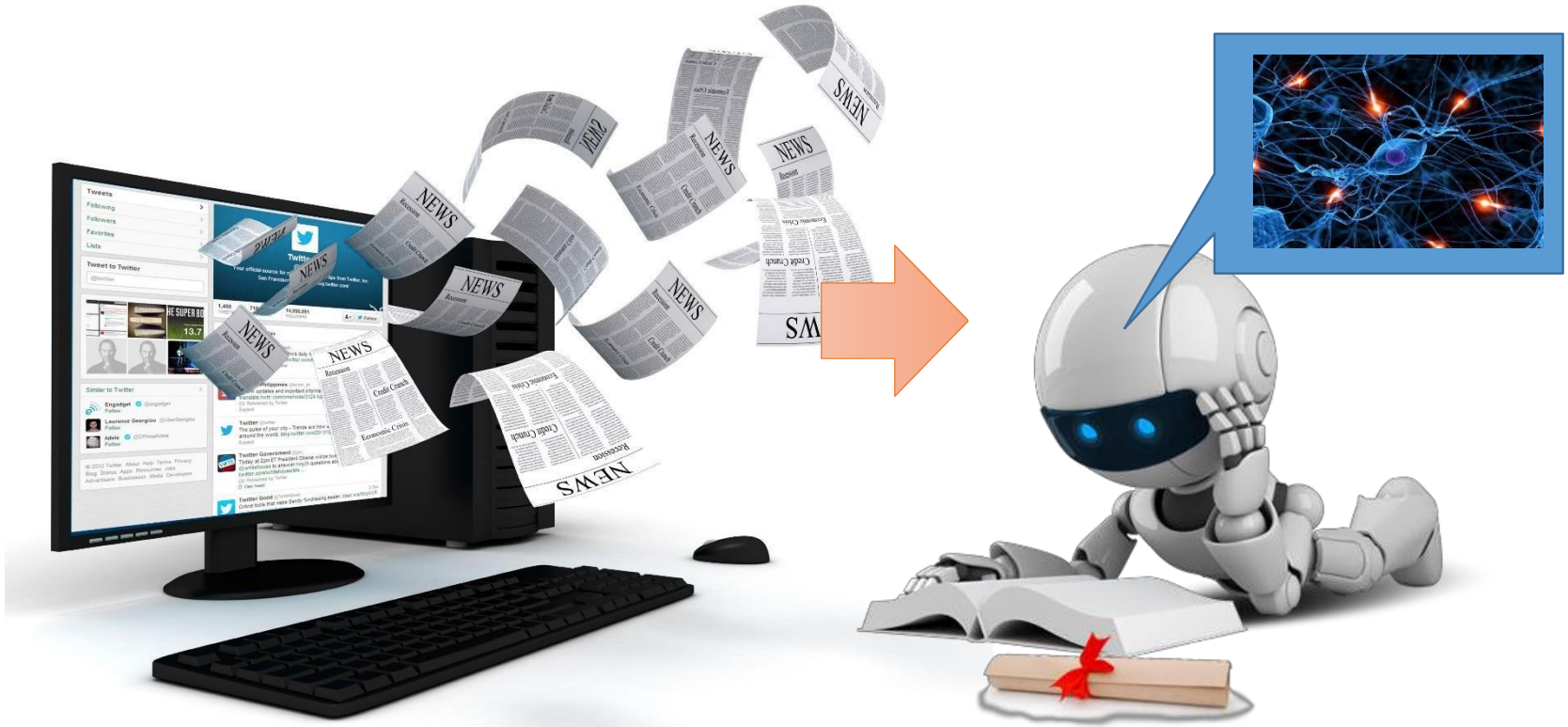
強化學習

無監督學習

- 圖像: 認識到世界是什麼樣
- 文本: 理解單詞的含義
- 視頻: 在無監督的情況下學習人類的語言

機器閱讀

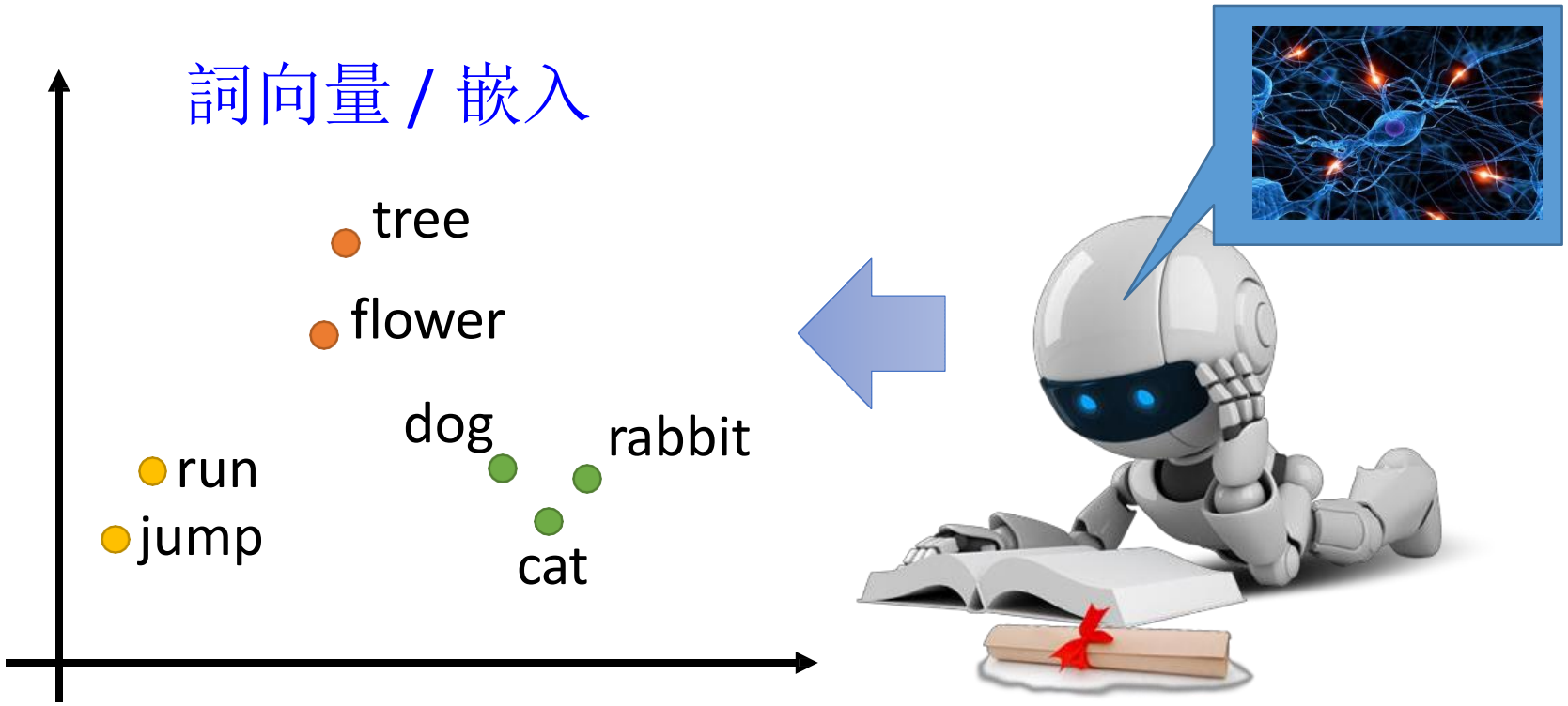
- 在無監督的情況下，機器通過閱讀大量檔來學習單詞的含義



http://top-breaking-news.com/

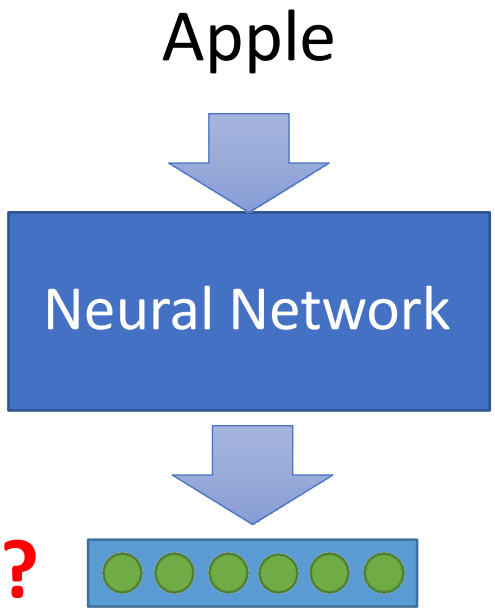
機器閱讀

- 在無監督的情況下，機器通過閱讀大量檔來學習單詞的含義



機器閱讀

- 生成詞向量/Embedding是無監督的



訓練數據是很多文本

https://garavato.files.wordpress.com/2011/11/stacksdocuments.jpg?w=490

機器閱讀

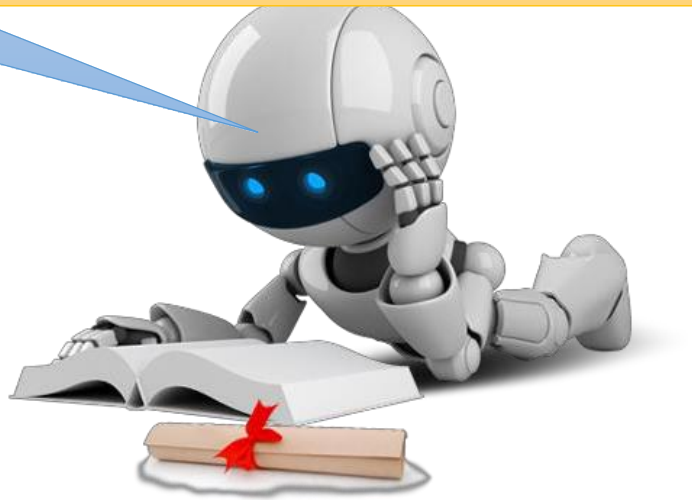
- 在無監督的情況下，機器通過閱讀大量檔來學習單詞的含義
- 一個詞可以通過它的上下文來理解

蔡英文、馬英九是非常相似的

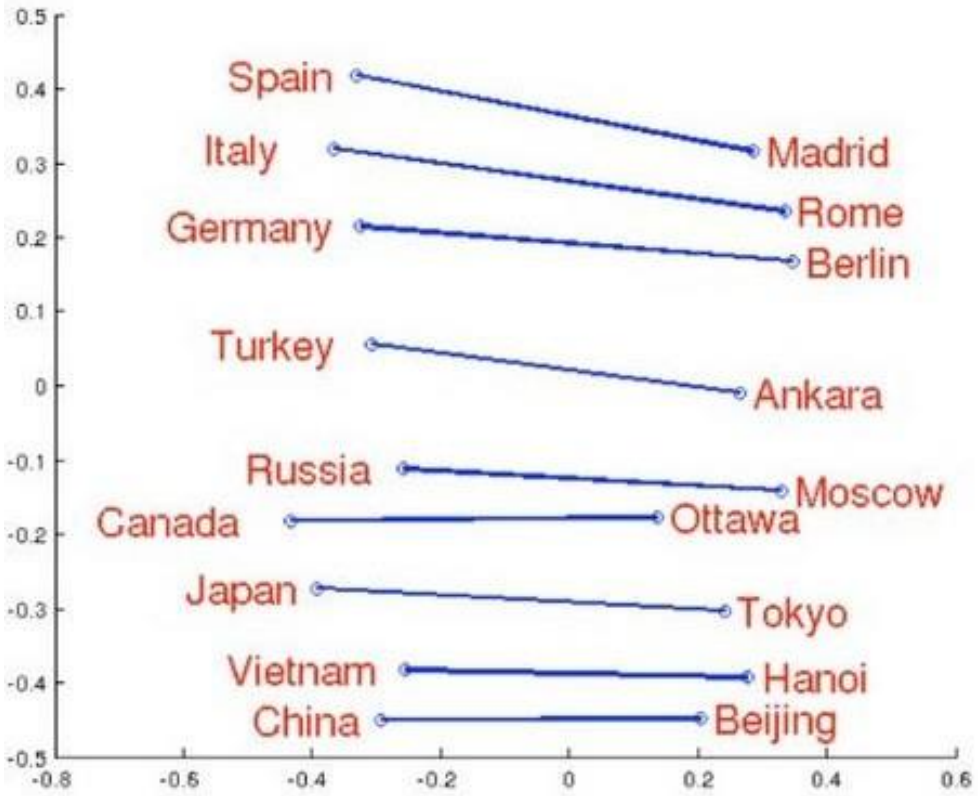
You shall know a word by the company it keeps

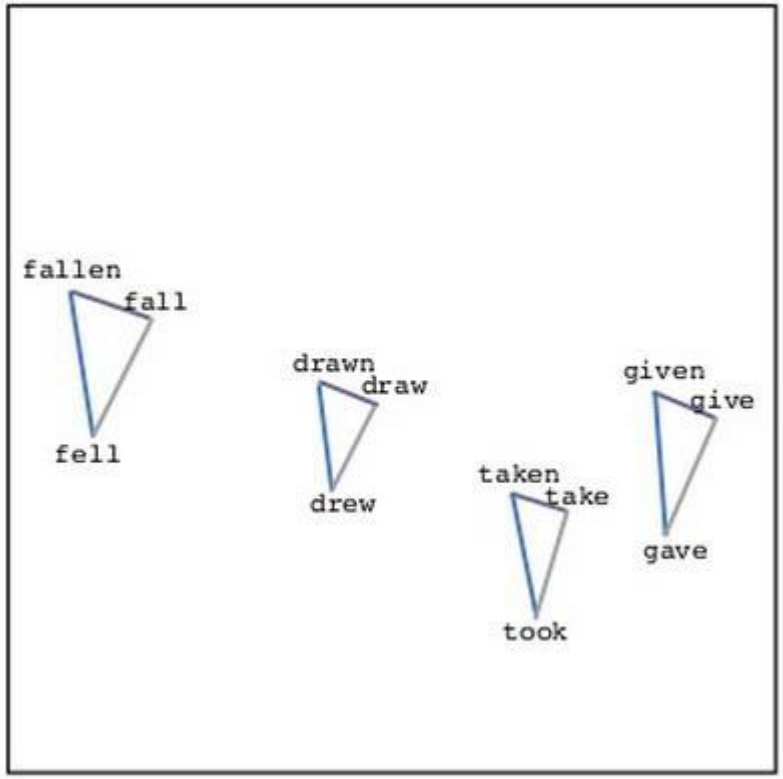
馬英九 520宣誓就職

蔡英文 520宣誓就職



詞向量





Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

詞向量

$$V(Germany) \approx V(Berlin) - V(Rome) + V(Italy)$$

- 特徵

$$V(hotter) - V(hot) \approx V(bigger) - V(big)$$
$$V(Rome) - V(Italy) \approx V(Berlin) - V(Germany)$$
$$V(king) - V(queen) \approx V(uncle) - V(aunt)$$

- 解決類比

Rome : Italy = Berlin : ?

Compute $V(Berlin) - V(Rome) + V(Italy)$

Find the word w with the closest V(w)

大綱

監督學習

- 超深網路
- 注意力模型

} 新型神經網路結構

強化學習

無監督學習

- 圖像: 認識到世界是什麼樣
- 文本: 理解單詞的含義
- 音訊: 在無監督的情況下學習人類語言

學習有聲書

機器沒有任何先驗知識

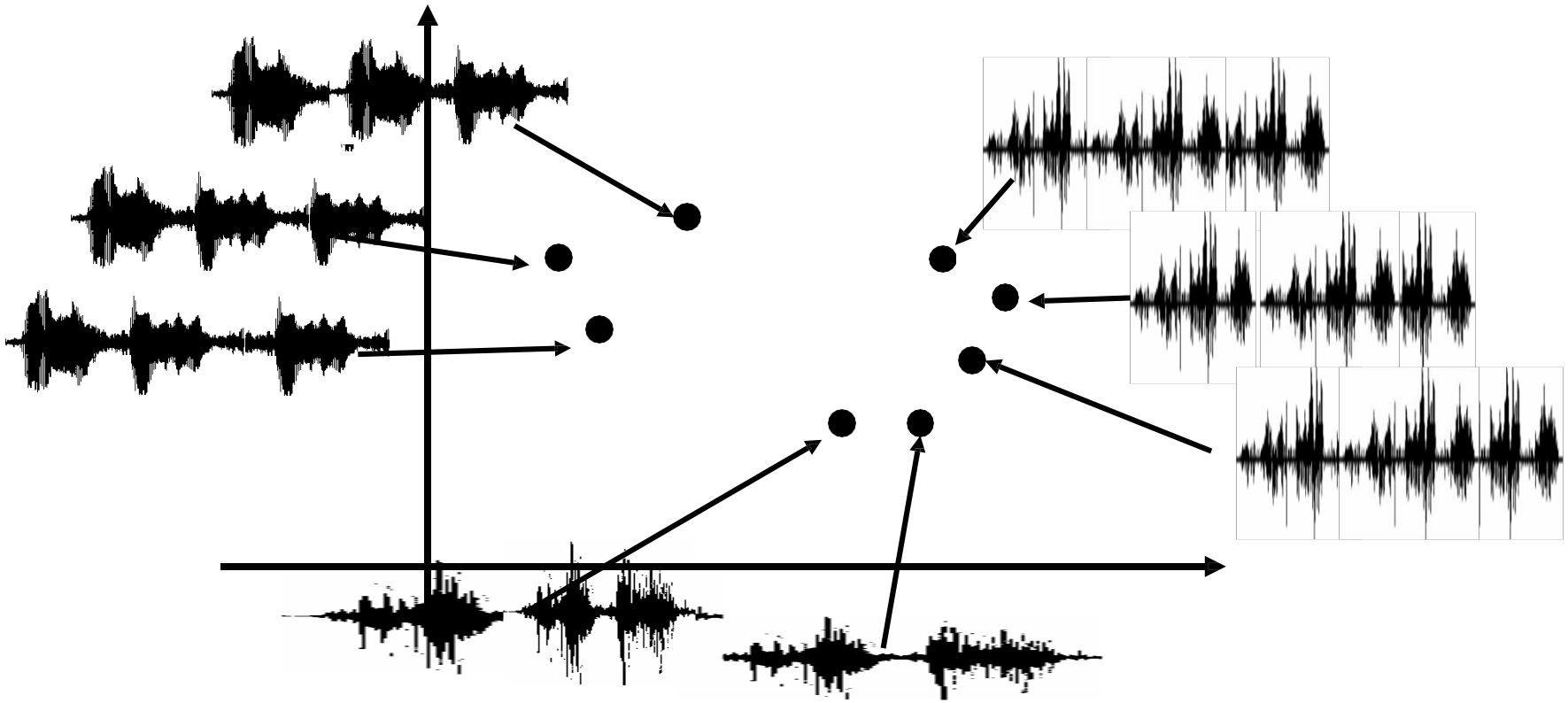
機器聽大量的有聲書

像一個嬰兒

[Chung, Interspeech 16)

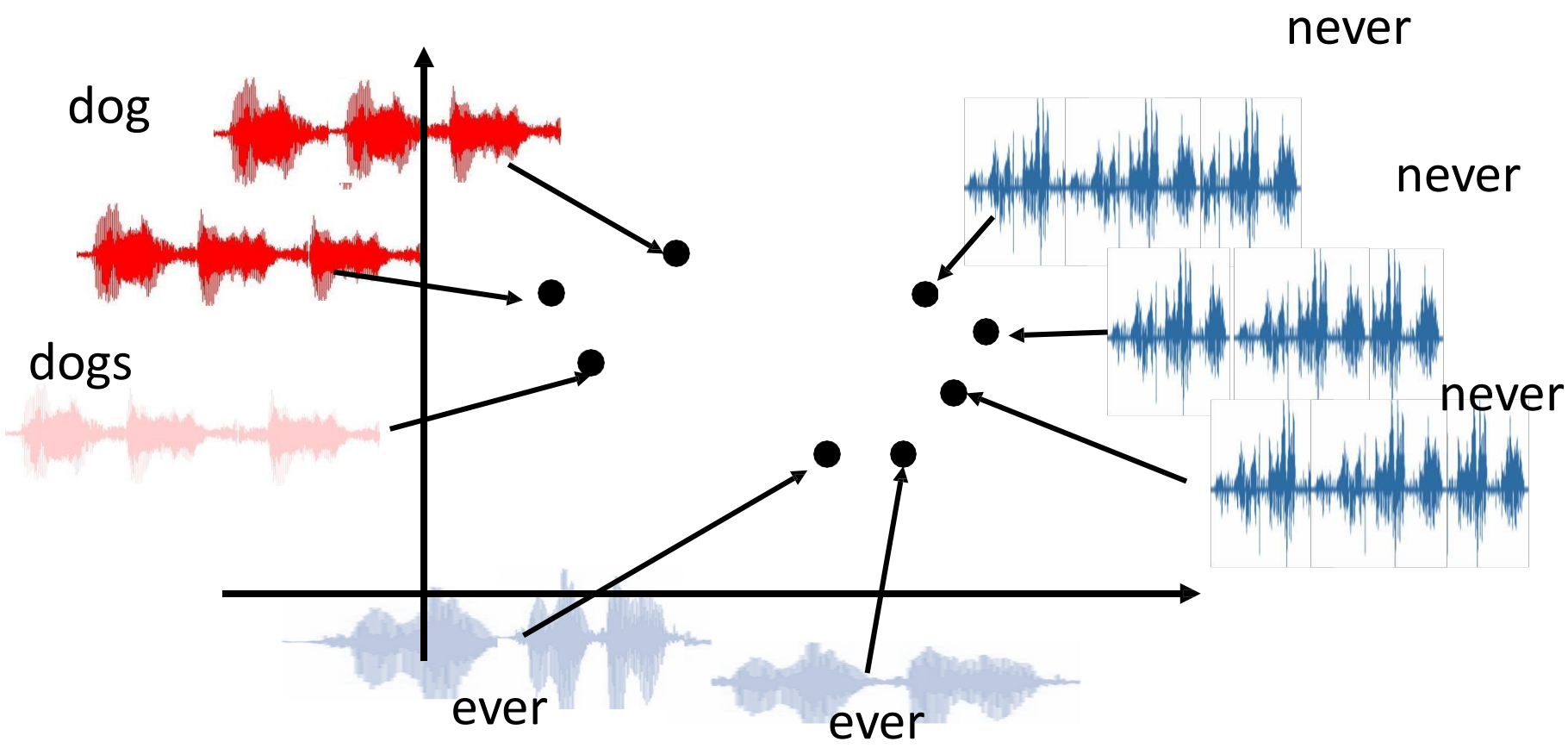
音訊詞向量

- 音訊片段對應於未知單詞

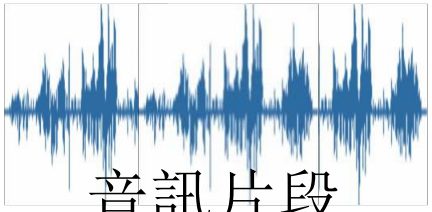


音訊詞向量

- 與發音相近的單詞對應的音段相互靠近



Sequence-to-sequence 自動編碼器



音訊片段

向量

RNN 編碼器

記憶體中的值代表整個音訊段

我們想要的向量

怎樣訓練 RNN 編碼器?

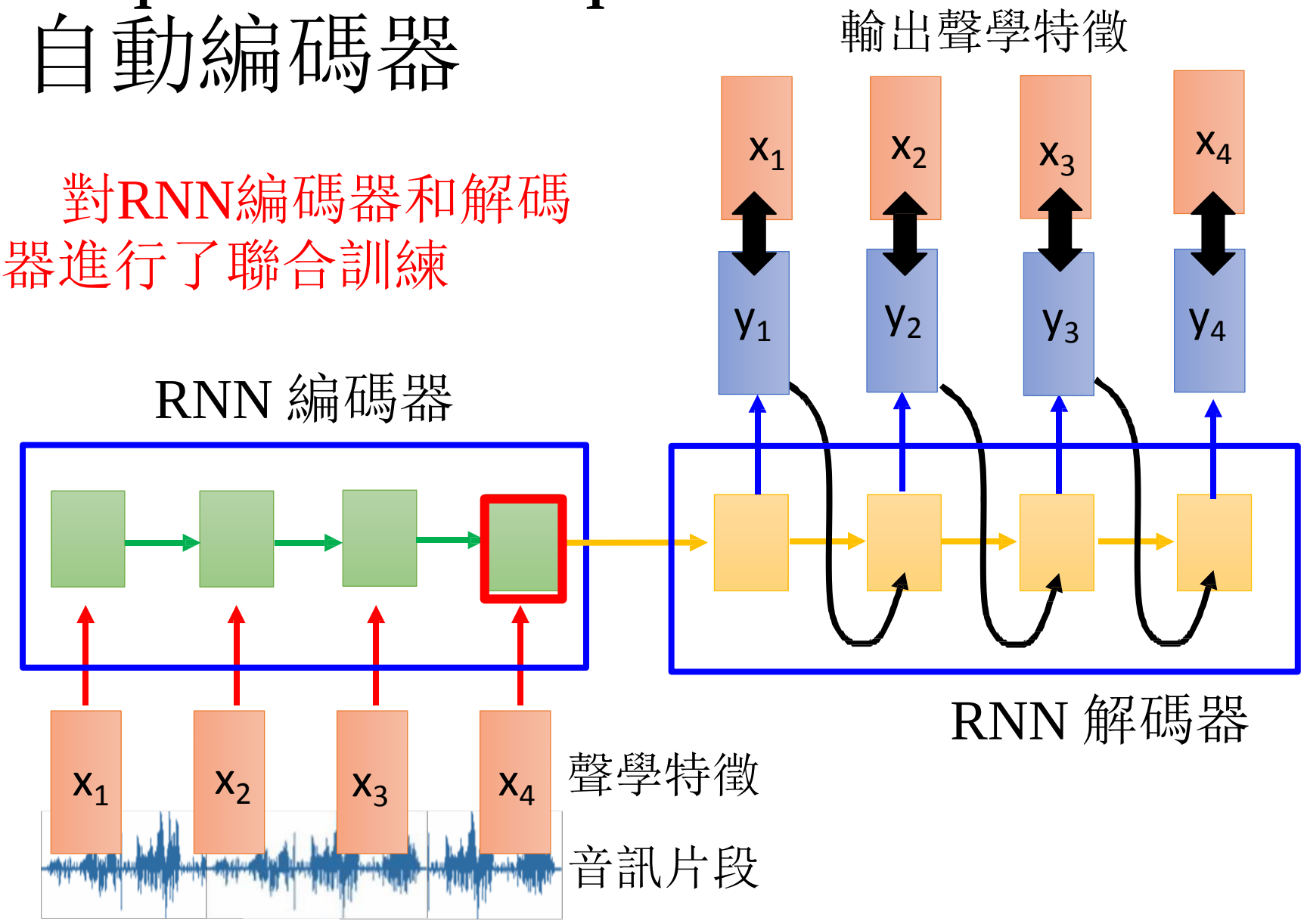
x_1 x_2 x_3 x_4

聲學特徵

音訊片段

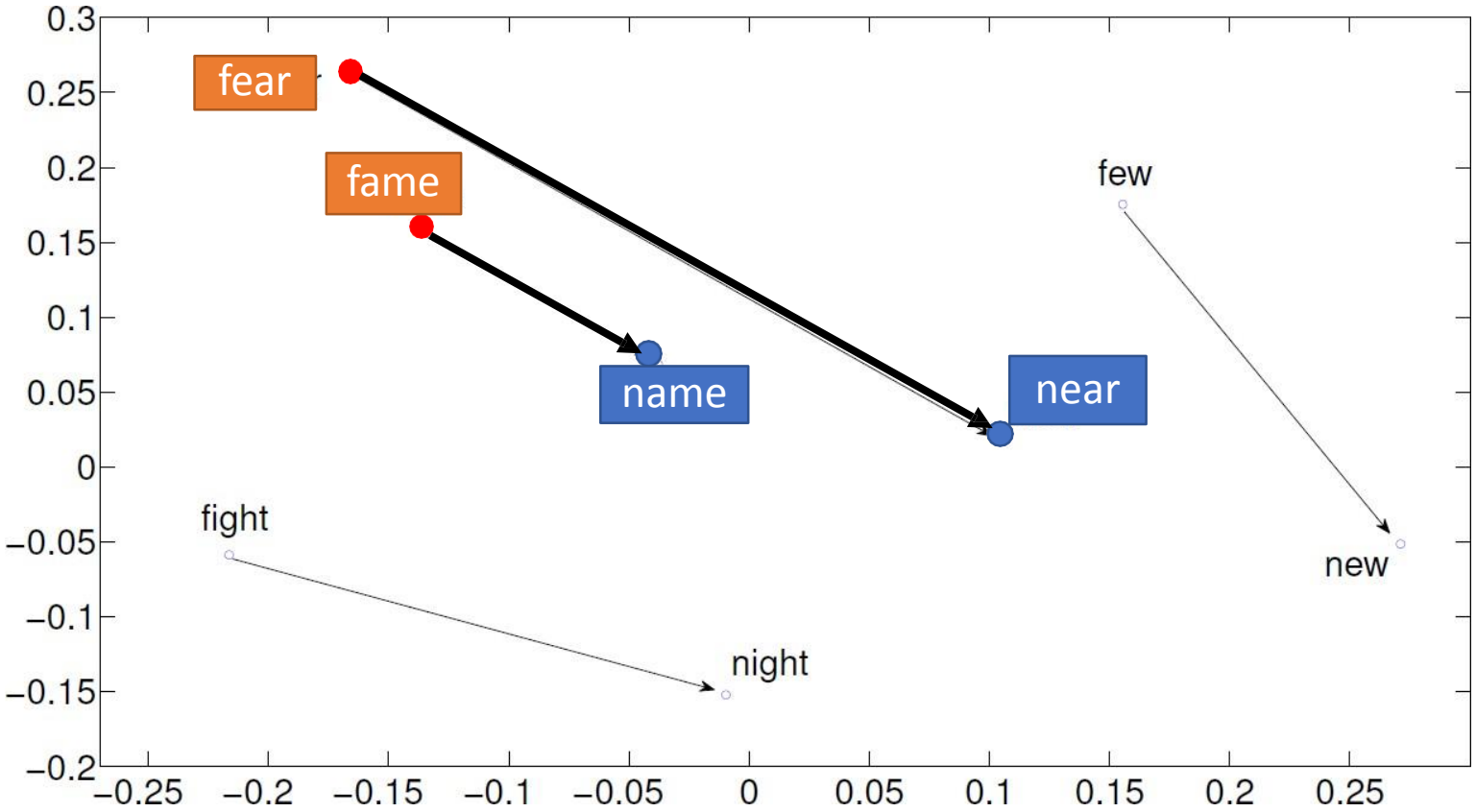
Sequence-to-sequence 自動編碼器

對RNN編碼器和解碼器進行了聯合訓練

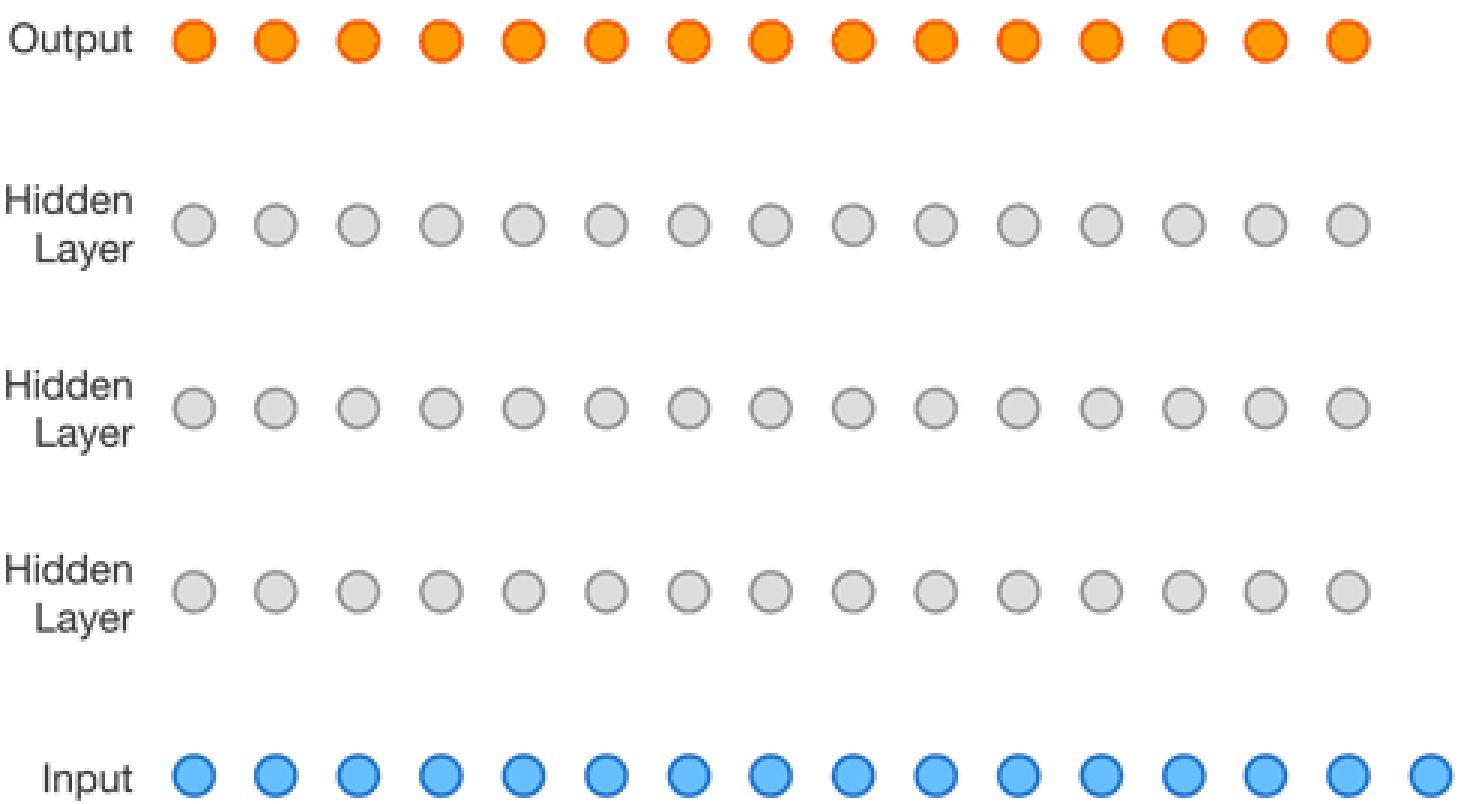


音訊詞向量 - 結果

- 視覺化詞的嵌入向量



WaveNet (DeepMind)



https://deepmind.com/blog/wavenet-generative-model-raw-audio/

結束語

大綱

Lecture I: 深度學習的介紹

Lecture II: 訓練神經網路的技巧

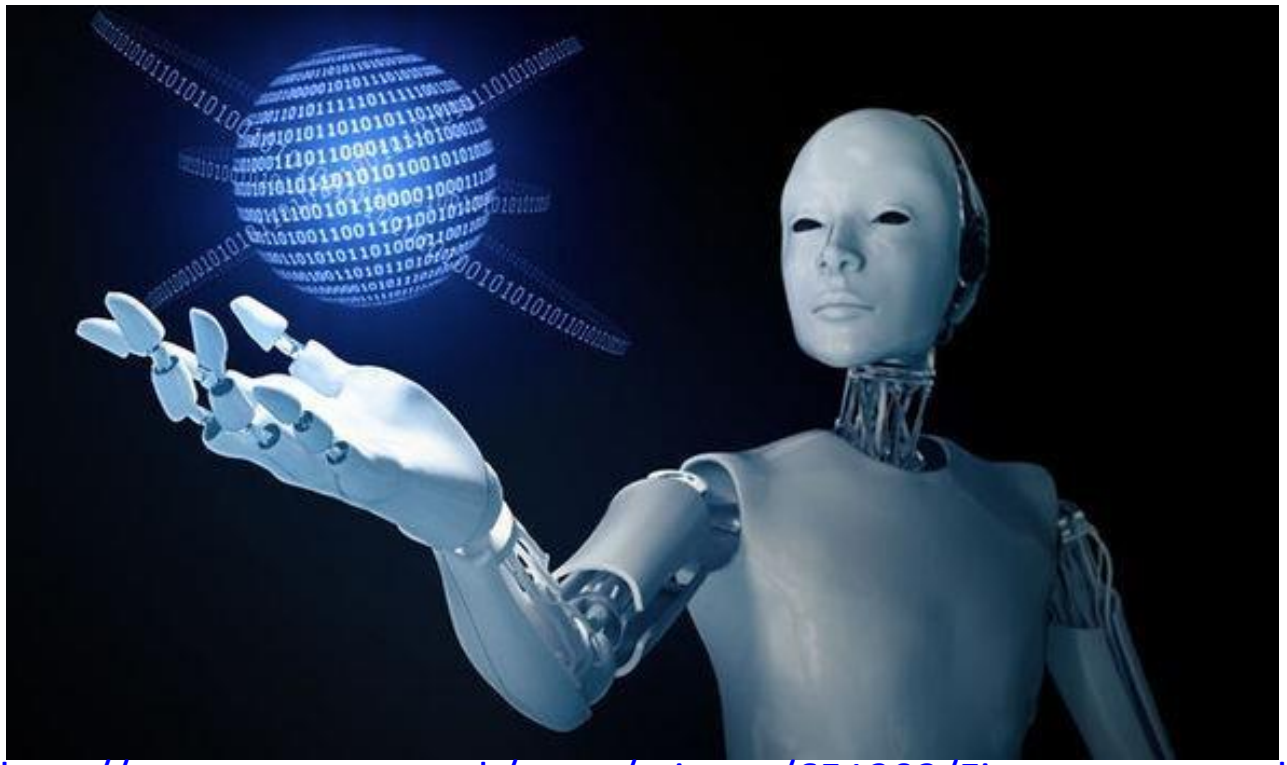
Lecture III: 神經網路的變體

Lecture IV: 下一個浪潮

AI 即將取代多數的工作?

- AI 時代的新工作 → **AI 訓練師**

(機器學習專家、資料科學家)



http://www.express.co.uk/news/science/651202/First-step-towards-The-Terminator-becoming-reality-AI-beats-champ-of-world-s-oldest-game

AI 訓練師

機器不是自己會學嗎？
為什麼需要 AI 訓練師

戰鬥數碼寶貝在打，
為什麼需要數碼寶貝訓練師
？

AI 訓練師

數碼寶貝訓練師

- 數碼寶貝訓練師要挑選適合的數碼寶貝來戰鬥
  - 數碼寶貝有不同的屬性
- 召喚出來的數碼寶貝不一定能操控
  - E.g. 小智的噴火龍
  - 需要足夠的經驗

AI 訓練師

- 在step 1，AI訓練師要挑選合適的模型
  - 不同模型適合處理不同的問題
- 不一定能在 step 3 找出最優的函數
  - E.g. Deep Learning
  - 需要足夠的經驗

AI 訓練師

- 厲害的 AI， AI 訓練師功不可沒
- 讓我們一起朝 AI 訓練師之路邁進