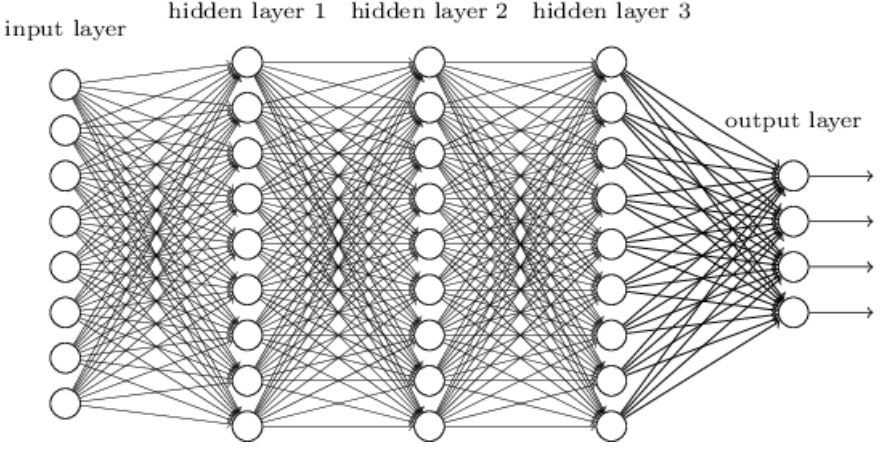
Machine Learning

Lecture 9
Convolutional Neural Network

Chen-Kuo Chiang (江 振 國)
ckchiang@cs.ccu.edu.tw
中正大學 資訊工程學系

Motivation of CNN

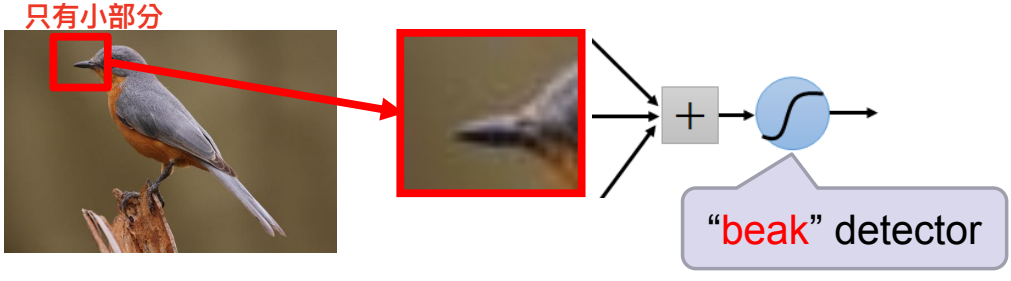
- We know it is good to learn a small model. 偵測鳥嘴位置
- From this fully connected model, do we really need all the edges? 是否要使用全連結
- Can some of these be shared?



Consider Learning ‘Beak Detector’

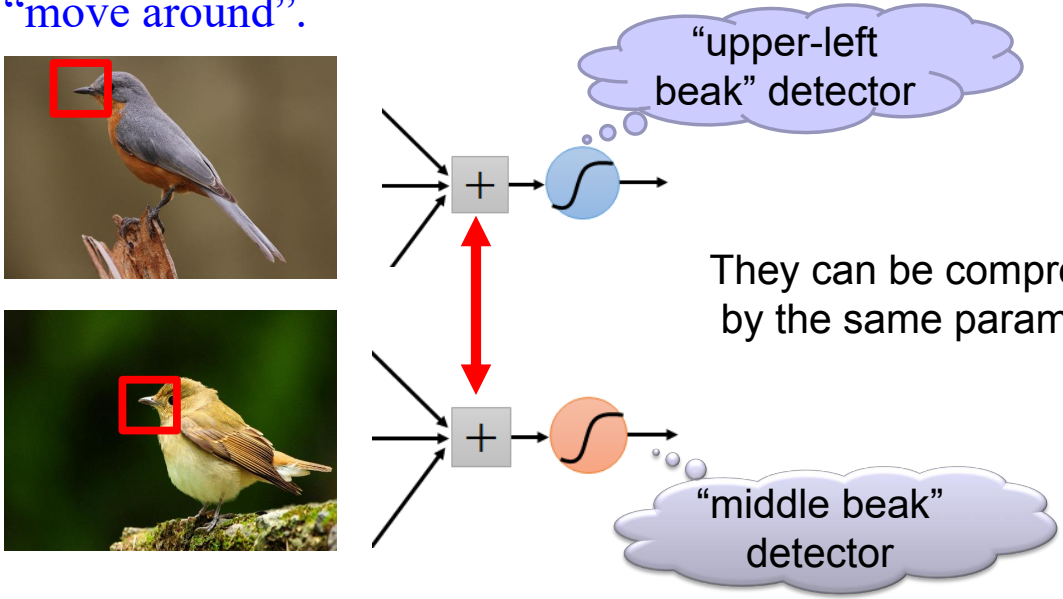
- Beak patterns are much smaller than the whole image.

Can represent a small region with fewer parameters



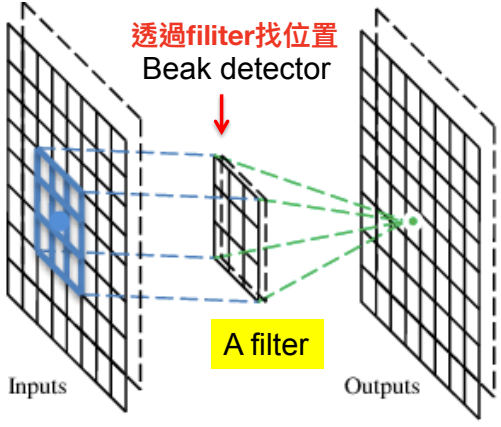
Same pattern appears in different places…

- They can be compressed!
- What about training a lot of such "small" detectors and each detector must "move around".



Convolutional Layer

- A CNN is a neural network with some convolutional layers (and some other layers). 找位置
- A convolutional layer has a number of filters that does convolutional operation.



Convolution Operation

專家設定filter

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

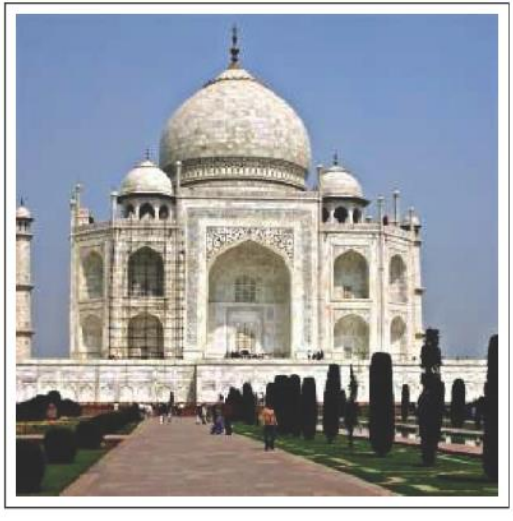
-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

⋮ ⋮

Each filter detects a small pattern (3 x 3).

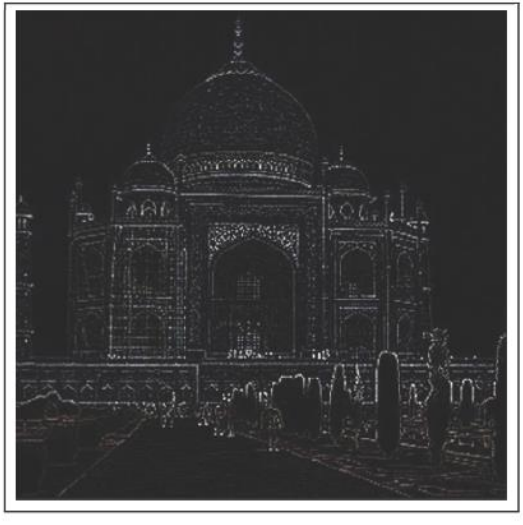
Example of a Kernel



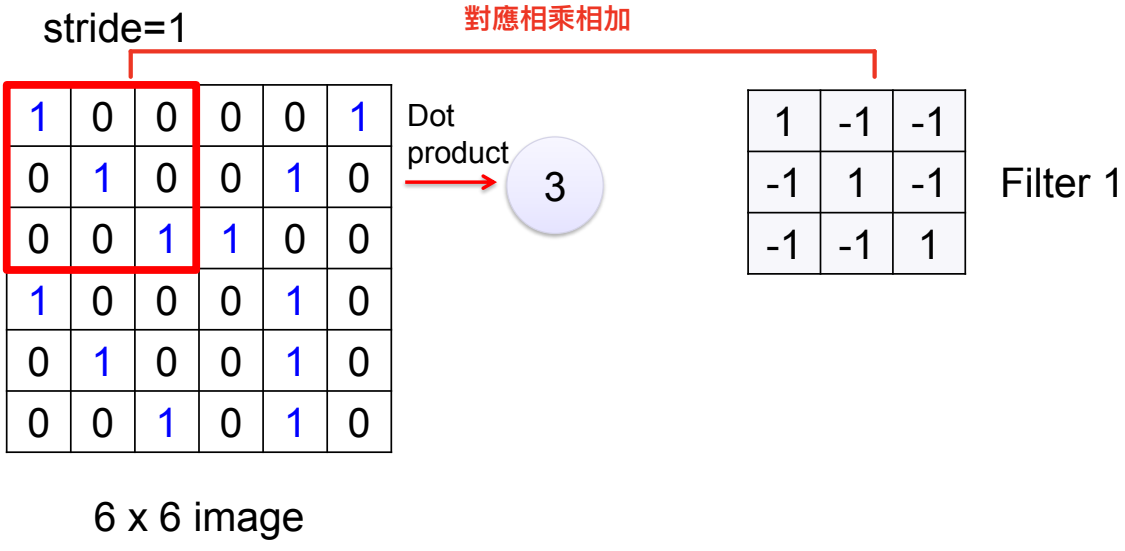
轉成convolution

Edge enhance
Filter

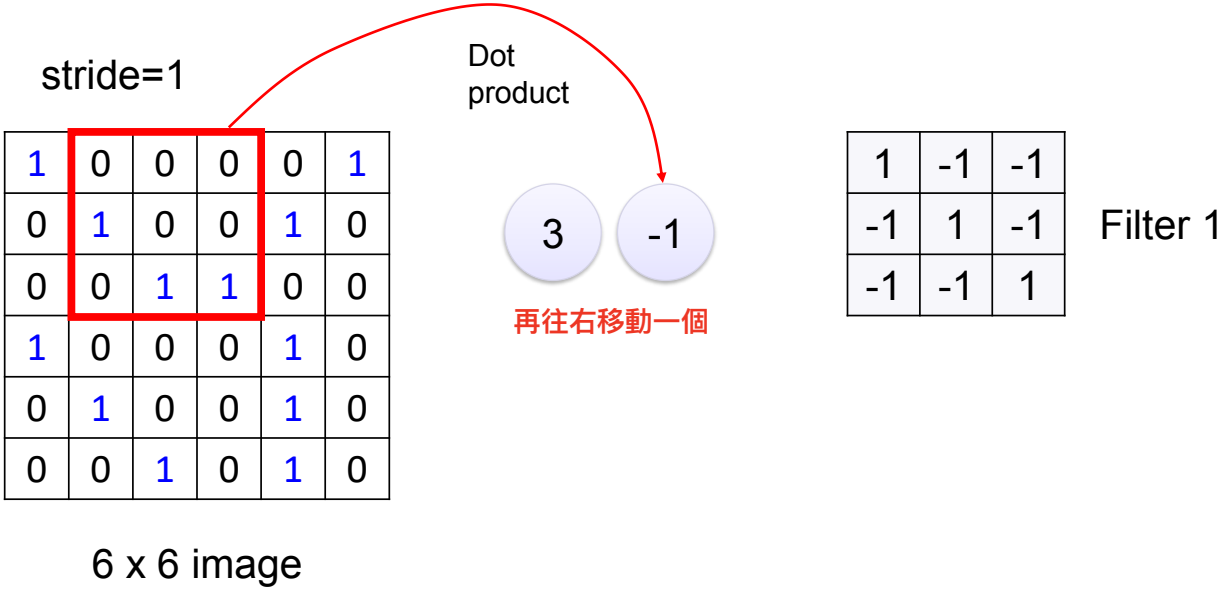
0	0	0
-1	1	0
0	0	0



How to Apply Convolution ?



How to Apply Convolution ?



How to Apply Convolution ?

If stride=2

Dot product

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

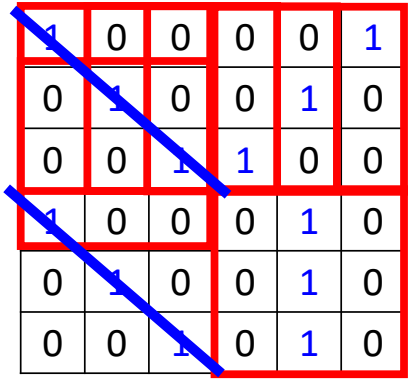
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

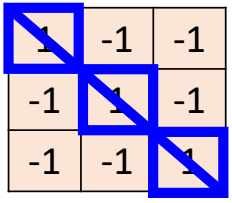
Convolution

過後結果

stride=1

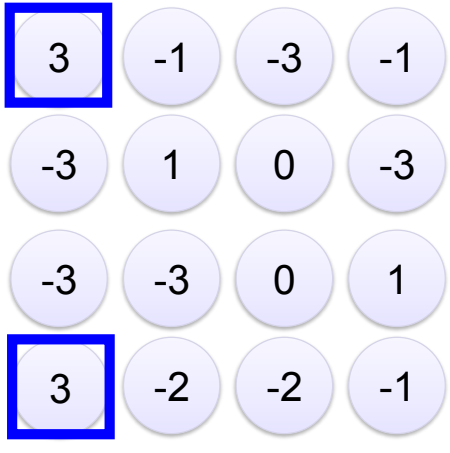


6 x 6 image



Filter 1

找出對
角有3個
1的位置



Convolution

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

Filter 2

stride=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

Input

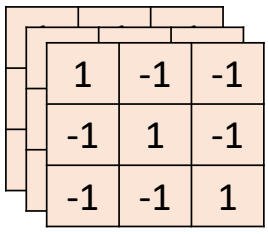
Repeat this for each filter

Two 4 x 4 images Output

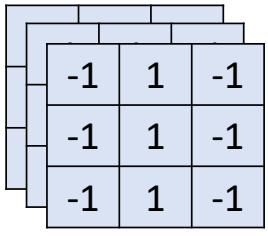
Forming 2 x 4 x 4 matrix

Color image: RGB 3 channels

針對每個channel設計filter

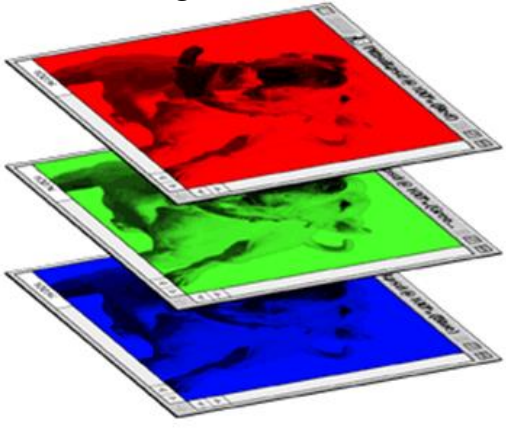


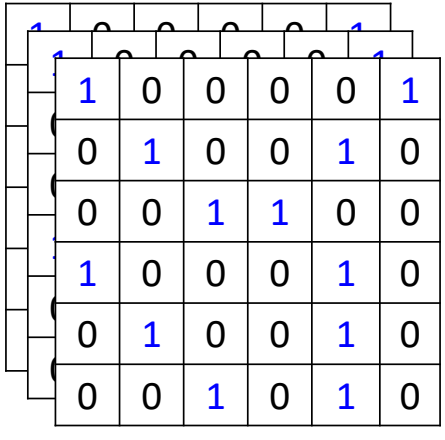
Filter 1



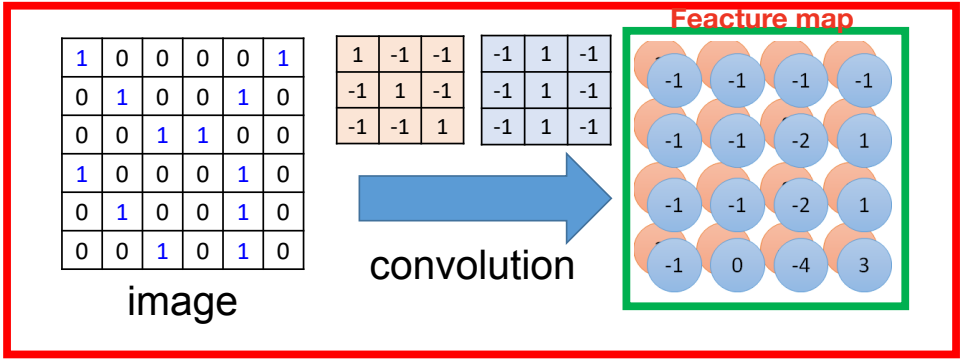
Filter 2

Color image

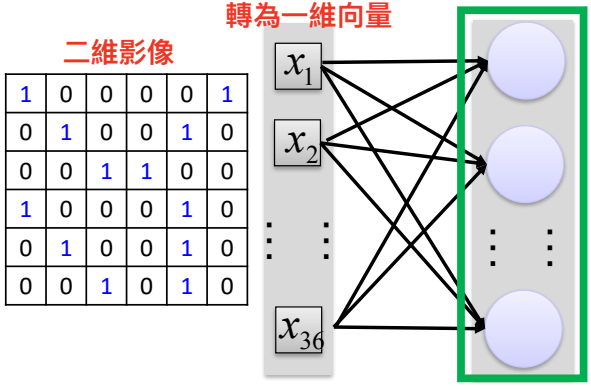


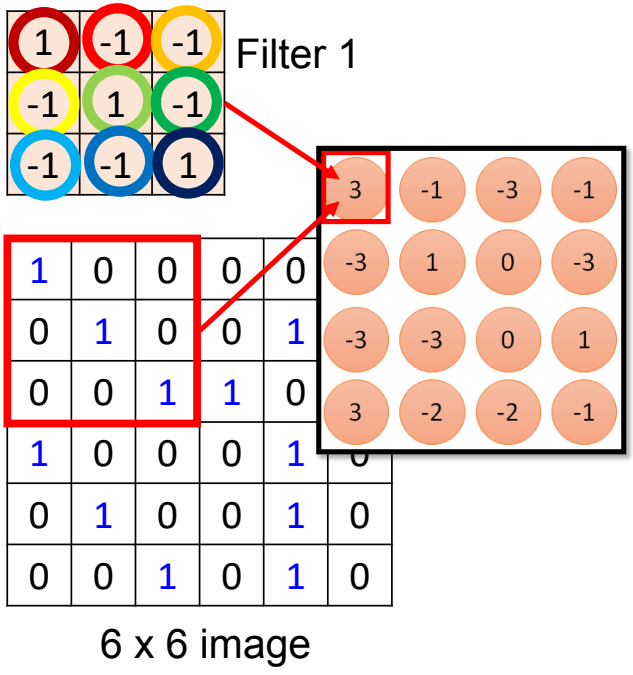


Convolution v.s. Fully Connected

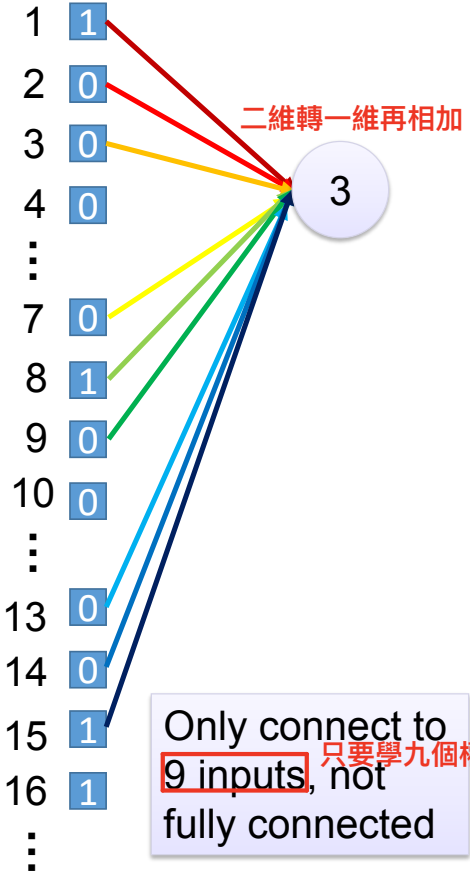


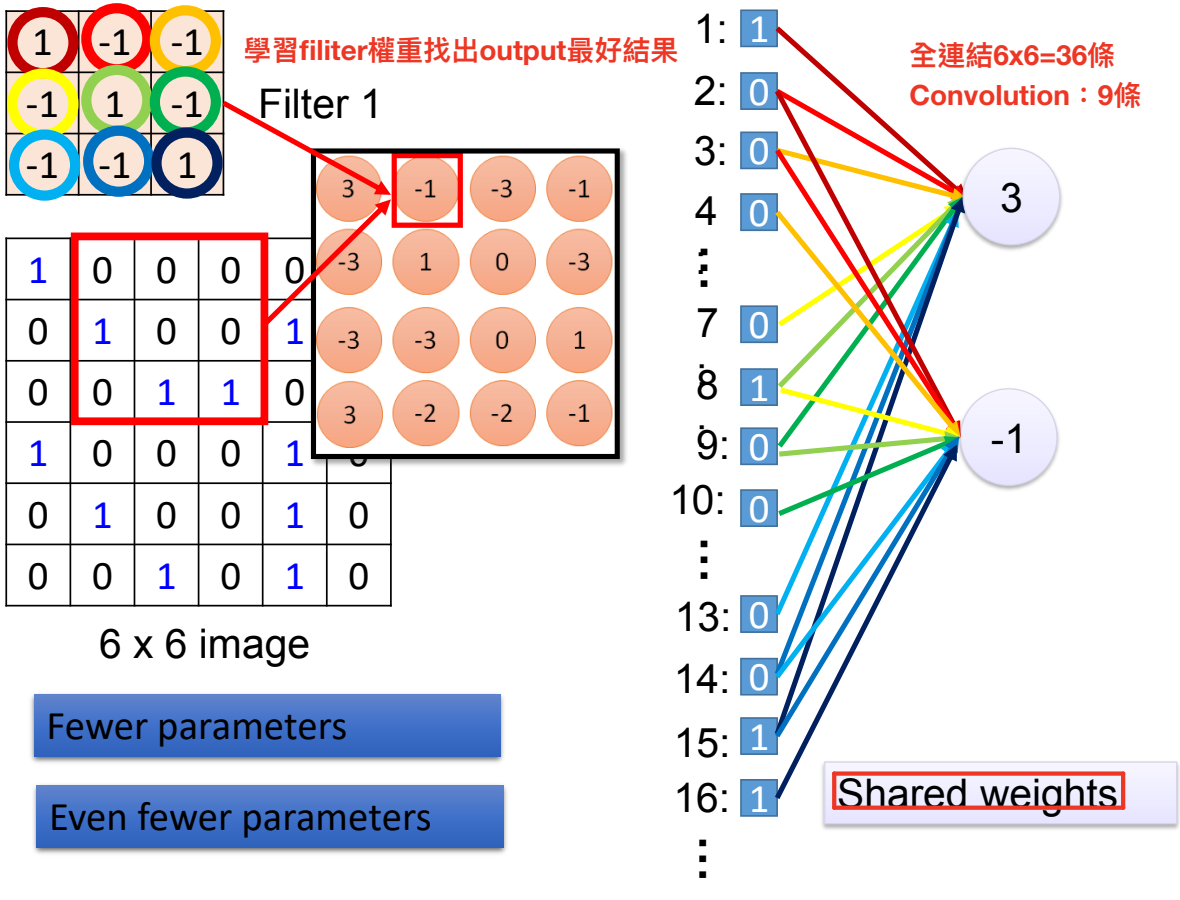
Fully-connected





fewer parameters!




學習filiter權重找出output最好結果
Filter 1
6 x 6 image
Fewer parameters
Even fewer parameters
全連結6x6=36條
Convolution：9條
Shared weights
3
-1

Max Pooling

找出最大response，不看其他local

4x4變2x2降低模型參數量

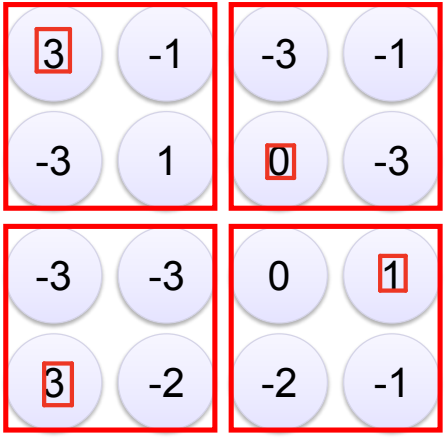
1	-1	-1
-1	1	-1
-1	-1	1

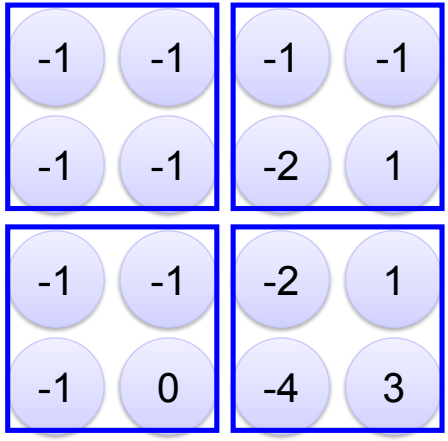
Filter 1

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

Filter 2

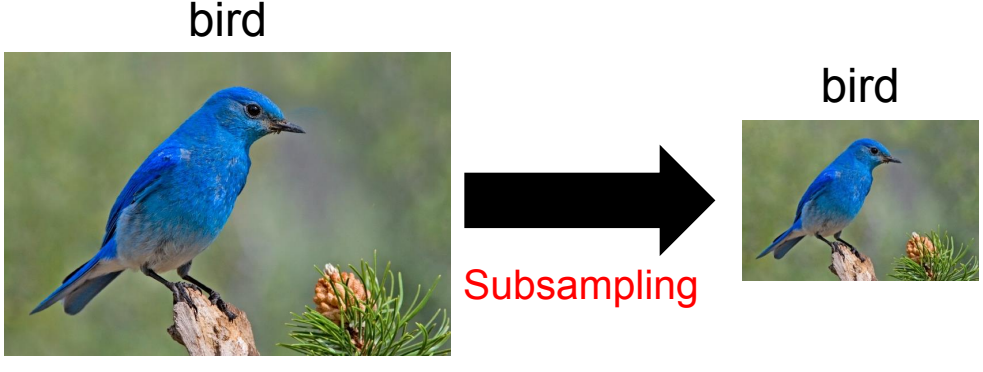
最大找出





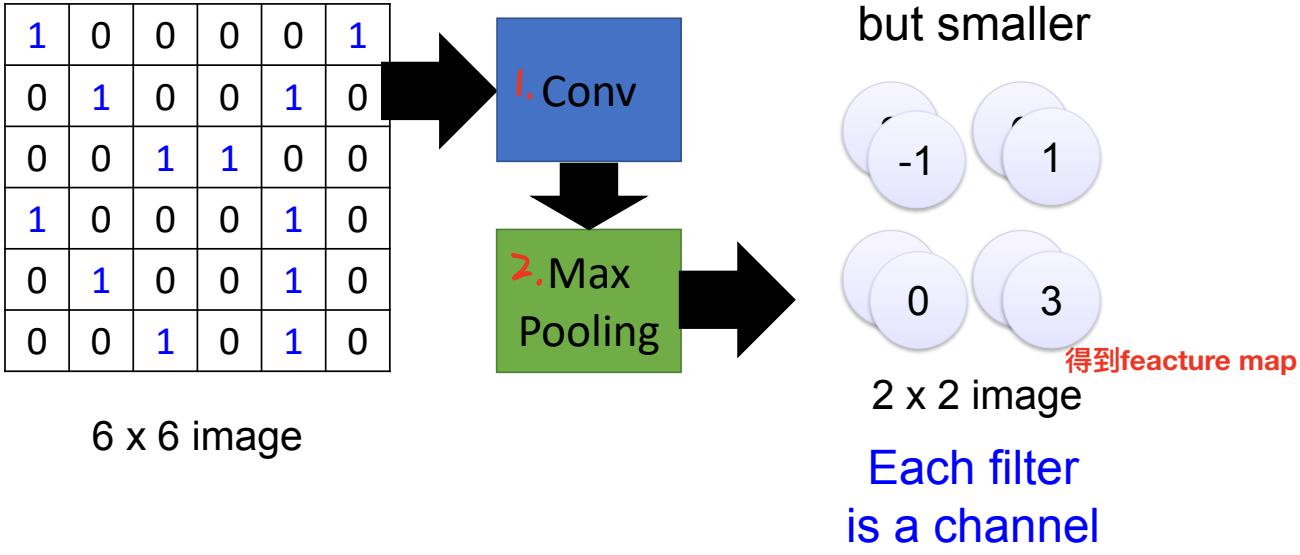
Why Pooling

- Subsampling pixels will not change the object.



- We can subsample the pixels to make image smaller.
- This means fewer parameters to characterize the image.

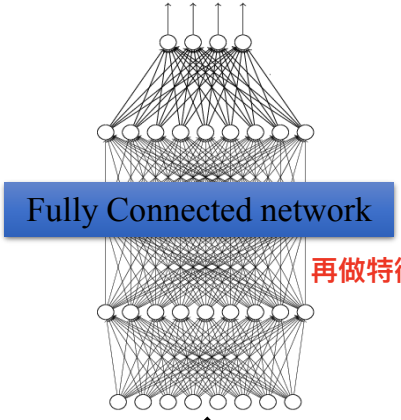
Max Pooling


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
1. Conv
2. Max Pooling
New image but smaller
-1
1
0
3
得到feacture map
6 x 6 image
2 x 2 image
Each filter is a channel

The whole CNN

- Reducing number of connections
- Shared weights on the edges
- Max pooling further reduces the complexity.

cat dog ……



再做特徵全連結

Convolution

Max Pooling

Convolution

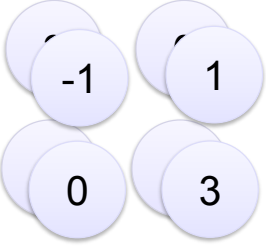
Max Pooling

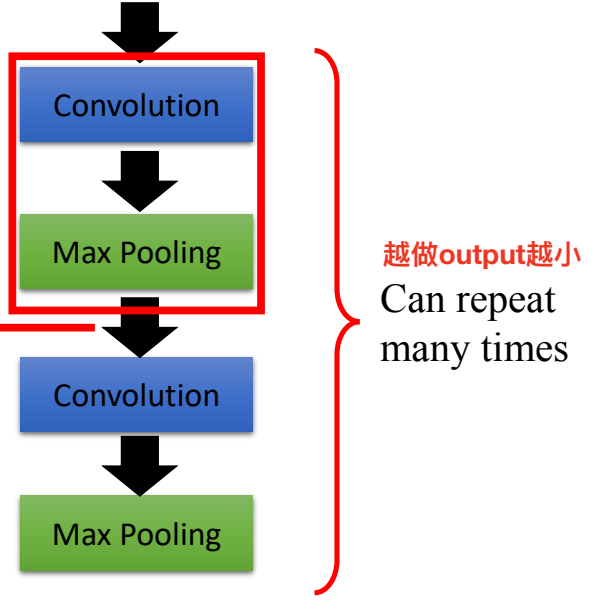
反覆做

Can repeat many times…

Flattened

The whole CNN

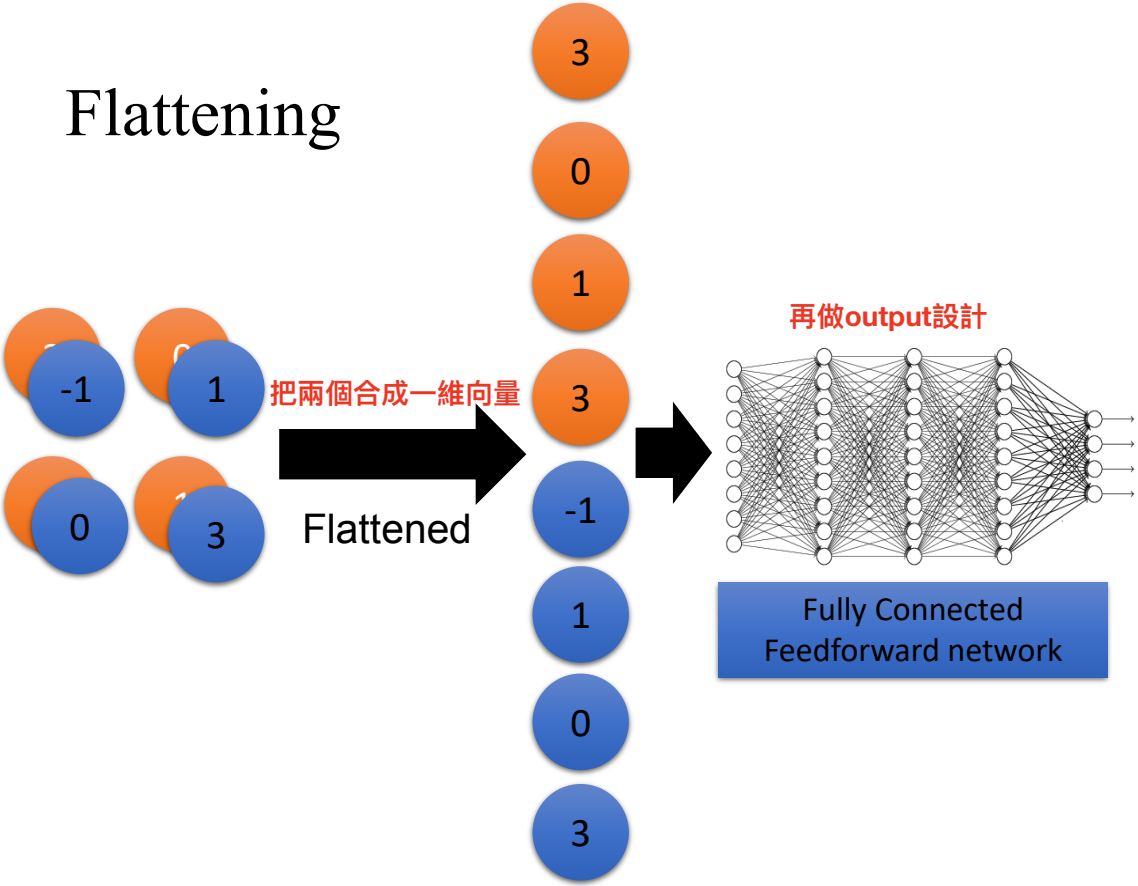




- Smaller than the original image
- The number of channels is the number of filters

一個feacture map

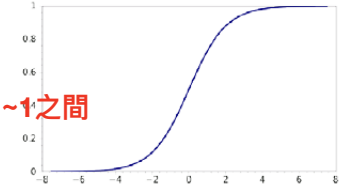
Flattening


-1
1
0
3
把兩個合成一維向量
Flattened
3
0
1
3
-1
1
0
3
再做output設計
Fully Connected
Feedforward network

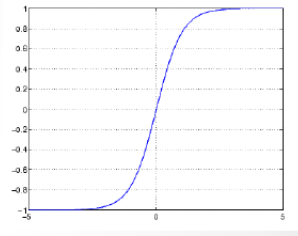
Non-Linear Activation Function

- Sigmoid: $S(t) = \frac{1}{1+e^{-t}}$. 處理機率問題0~1
- Tanh: $\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ -1~1之間
- Rectified Linear Unit (ReLU):

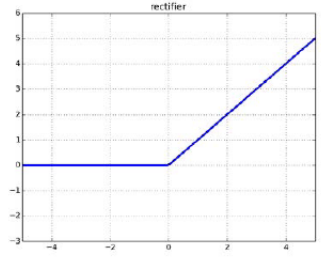
$$f(x) = \max(0, x)$$ -的為零



Sigmoid



Tanh



ReLU

ReLU is the most popular activation function for DNN in 2015, avoiding saturation issues.

Backpropagation in Deep Learning

- Three Important Parameters for Weight Update $\omega_i \leftarrow \omega_i - \eta \frac{\partial E}{\partial w_i} + \alpha\omega_i - \lambda\eta\omega_i$

GD

η- learning rate, 更新要走多長

α - momentum, 更新w的regulation，控制模型複雜度防止overfitting

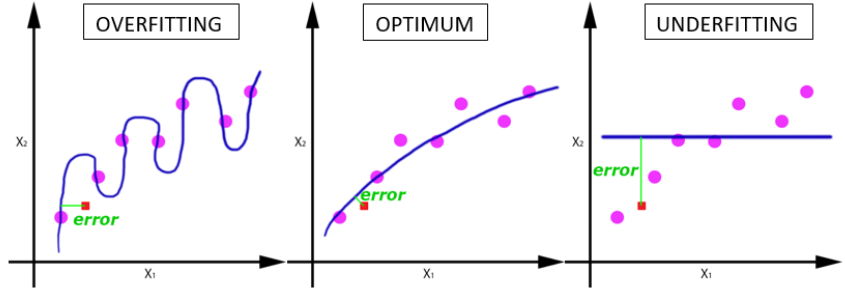
λ - weight decay

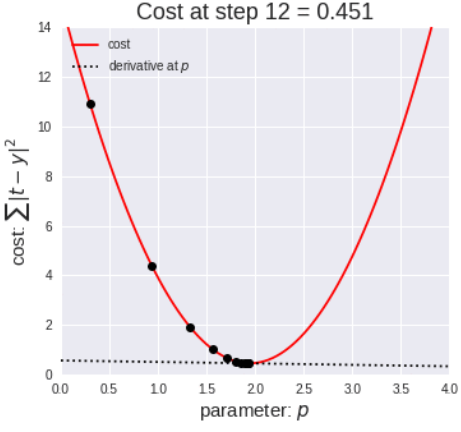
權重衰減 (weight decay) 的主要目的是「抑制更新參數的幅度」，在訓練模型時預測的結果會通過損失函數評估與真實值的差距，再藉由梯度下降更新參數，而權重衰減的方法是在損失函數加上一個懲罰項。

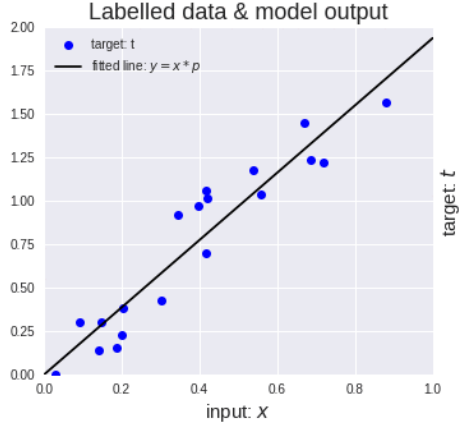
- Momentum (動量)
  - 一般Gradient Descent是朝負梯度方向下降。當考慮Momentum，如果上一次的Momentum(即w_i)與這次負梯度方向相同，這次下降的幅度就會更大。
  - 這可以在梯度平坦區，降低來回震動次數，加速收斂

Epoch vs Batch Size vs Iterations

- Recall: **Gradient Descent** is an iterative optimization algorithm to find the best results.







- The algorithm is iterative- we need to get the results multiple times to get the most optimal result.
- The iterative quality helps a under-fitted graph to make the graph fit optimally to the data.

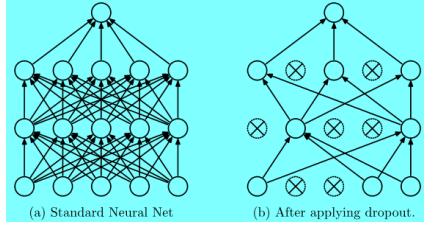
Epoch vs Batch Size vs Iterations

- **Epochs** 整個data set跑一輪
  - One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only ONCE.
  - When using a limited dataset optimize the learning by Gradient Descent, updating the weights with single pass or one epoch is not enough.
- **Batch Size** 一批次的資料，算梯度平均
  - Since you can't pass the entire dataset into the neural net at once to calculate the sum of gradient, you divide dataset into Number of Batches (mini-batches).
- **Iterations** 更新次數
  - The number of batches needed to complete one epoch.

- Example: divide the dataset of 2000 examples into batches of 500. Then, it will take 4 iterations to complete 1 epoch.

Measures to Reduce Overfitting

- These measures help in keeping the number of free parameters in the network low reducing complexity and thus over-fitting
- Dropout Learning Node 50%不被計算 模型太複雜
  - Randomly set the output value of network neurons to 0 with a dropout ratio of 0.5 (50% chance)
- Weight Decay 做regulation
  - Used to keep the magnitude of weights close to zero
- Data Augmentation 訓練資料不夠多，要資料擴增
  - Took random crop of 227x227 from image of 256x256 and randomly mirrored it in each forward-backward training pass
    把圖旋轉增加圖的資料



Error Function – Classification Error v.s Cross Entropy

- Classification Error = Count of error items / Count of all items
- Example: computed 預測結果，targets 標準答案

模型 1

最高

computed			targets			correct?
0.3	0.3	0.4	0	0	1 (democrat)	yes
0.3	0.4	0.3	0	1	0 (republican)	yes
0.1	0.2	0.7	1	0	0 (other)	no

classification error $= 1/3 = 0.33$

模型 2

computed			targets			correct?
0.1	0.2	0.7	0	0	1 (democrat)	yes
0.1	0.7	0.2	0	1	0 (republican)	yes
0.3	0.4	0.3	1	0	0 (other)	no

classification error $= 1/3 = 0.33$

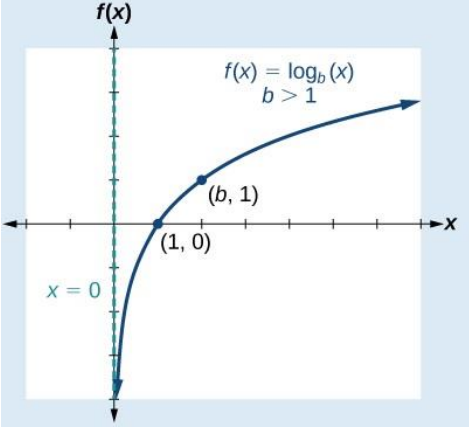
模型一前兩項都是險勝，模型二前兩是大勝；模型一最後一項是大敗，模型二最後一項是惜敗

Classification Error 無法分出兩個模型的差異!!!

Cross Entropy

- In neural network, $s_j = \sum_i w_{ij} x_i$, $a = \sigma(s)$ where $\sigma(.)$ is the activation function by sigmoid function. 其中a為預測結果，y為標準答案

Cross Entropy $C = -\frac{1}{n}\sum_x [y ln a + (1-y)\ln(1-a)]$ 二元分類

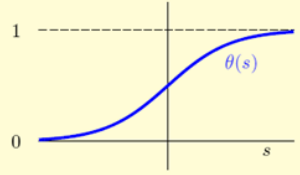


y	*a*	*lna*	*ln*(1-*a*)	*ylna*	(1-*y*)*ln*(1-*a*)	*C*
1	1	0	負很大	0	0	0
1	0	負很大	0	負很大	0	正很大
0	1	0	負很大	0	負很大	正很大
0	0	負很大	0	0	0	0

答對的時候C為0，答錯的時候C很大 = 最小化C可以得到正確答案!

Sigmoid Function

- Sigmoid Function
  - 範圍0到1的平滑函數



$\theta(-\infty) = 0$; $\theta(0) = \frac{1}{2}$; $\theta(\infty) = 1$

$$\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

—smooth, monotonic, sigmoid function of s

Cross Entropy

- Example:

模型 1

computed	targets	correct?
0.3 0.3 0.4	0 0 1 (democrat)	yes
0.3 0.4 0.3	0 1 0 (republican)	yes
0.1 0.2 0.7	1 0 0 (other)	no

classification error $= 1/3 = 0.33$

模型 2

computed	targets	correct?
0.1 0.2 0.7	0 0 1 (democrat)	yes
0.1 0.7 0.2	0 1 0 (republican)	yes
0.3 0.4 0.3	1 0 0 (other)	no

classification error $= 1/3 = 0.33$

第一个模型中第一项的 cross-entropy 是：

-((ln(0.3)*0) + (ln(0.3)*0) + (ln(0.4)*1)) = -ln(0.4)

所以，第一个模型的 ACE (average cross-entropy error) 是

-(ln(0.4) + ln(0.4) + ln(0.1)) / 3 = 1.38

第二个模型的 ACE 是：

-(ln(0.7) + ln(0.7) + ln(0.3)) / 3 = 0.64

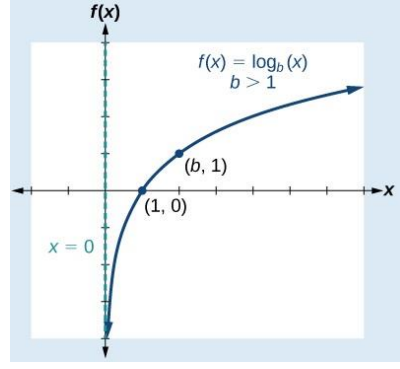
用 Cross Entropy 可以辨别出模型二優於模型一 !!!

Cross Entropy Function in TensorFlow

- Four choices for error functions
  - sigmoid_cross_entropy_with_logits
  - softmax_cross_entropy_with_logits
  - sparse_softmax_cross_entropy_with_logits
  - weighted_cross_entropy_with_logits

Sigmoid_cross_entropy_with_logits

- 使用方式
  - 可以用於binary classification problem
    - 分類的答案只有是或不是兩種
  - 不可用於multi-class classification problem
    - 分類的答案為第一類或第二類…或第十類，答案不重複
  - 不可用於multi-label classification problem
    - 輸入影像可以包含多個類別的答案，例如第二類與第十類



- 不能使用的原因:
  1. 當將y帶入2或10等類別答案時，1-y會出現負值
  2. log函數並非線性函數，a為2或10時，將引入許多誤差。

Cross Entropy $C = -\frac{1}{n}\sum_x [y ln a + (1-y)\ln(1-a)]$ 只有兩類

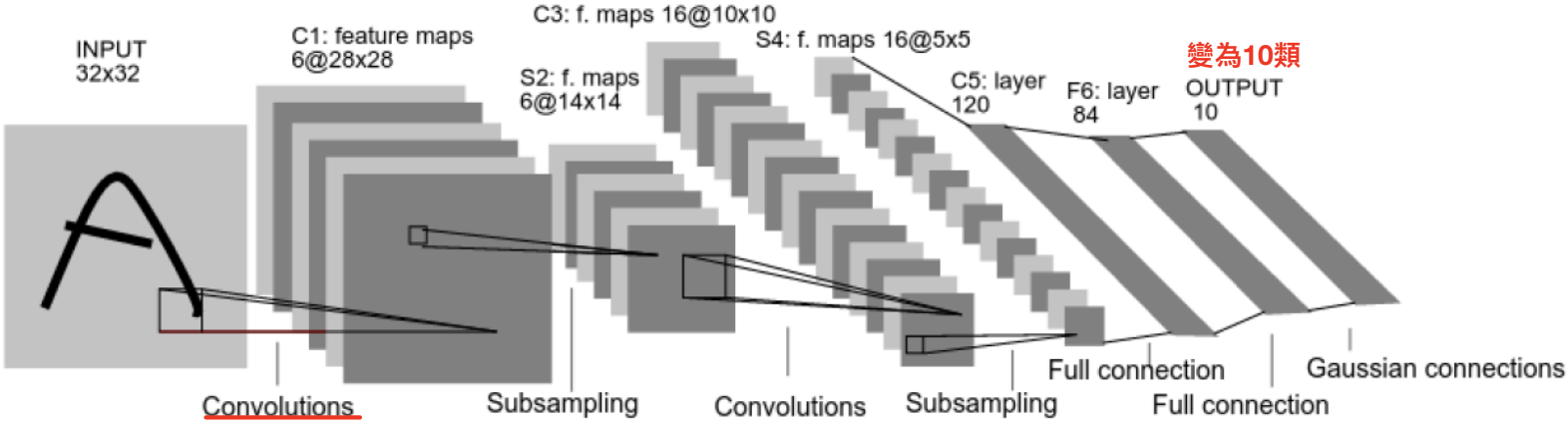
Softmax Function

$$h_\theta(x) = \begin{bmatrix} P(y=1|x;\theta) \\ P(y=2|x;\theta) \\ \vdots \\ P(y=K|x;\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} exp(\theta_j^T x)} \begin{bmatrix} exp(\theta_1^T x) \\ exp(\theta_2^T x) \\ \vdots \\ exp(\theta_K^T x) \end{bmatrix}$$

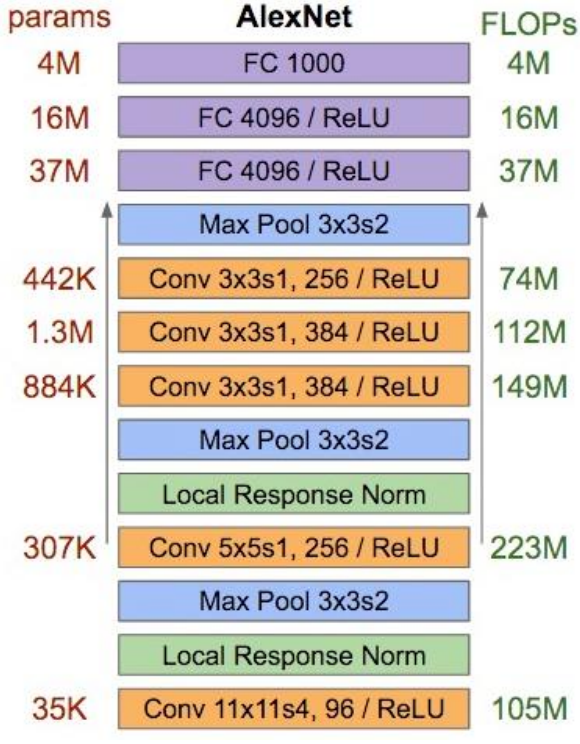
- 特性
  - 輸出每一個類別的機率值
  - 所有類別的機率值總合為1
- Softmax_cross_entropy_with_logits 使用方式
  - 分數值不作sigmoid，直接轉成機率值
  - 可以用於binary classification problem
  - 可以用於multi-class classification problem
  - 不可用於multi-label classification problem
- 原因
  - 對於multi-class問題，每一個類別的機率和為一，帶入Entropy公式結果正確
  - 對於multi-label，類別的機率和不會是一，無法計算Entropy

A simple version of the CNN (LeNet5 Architecture)

- Lenet-5 (Lecun-98), Convolutional Neural Network for digits recognition

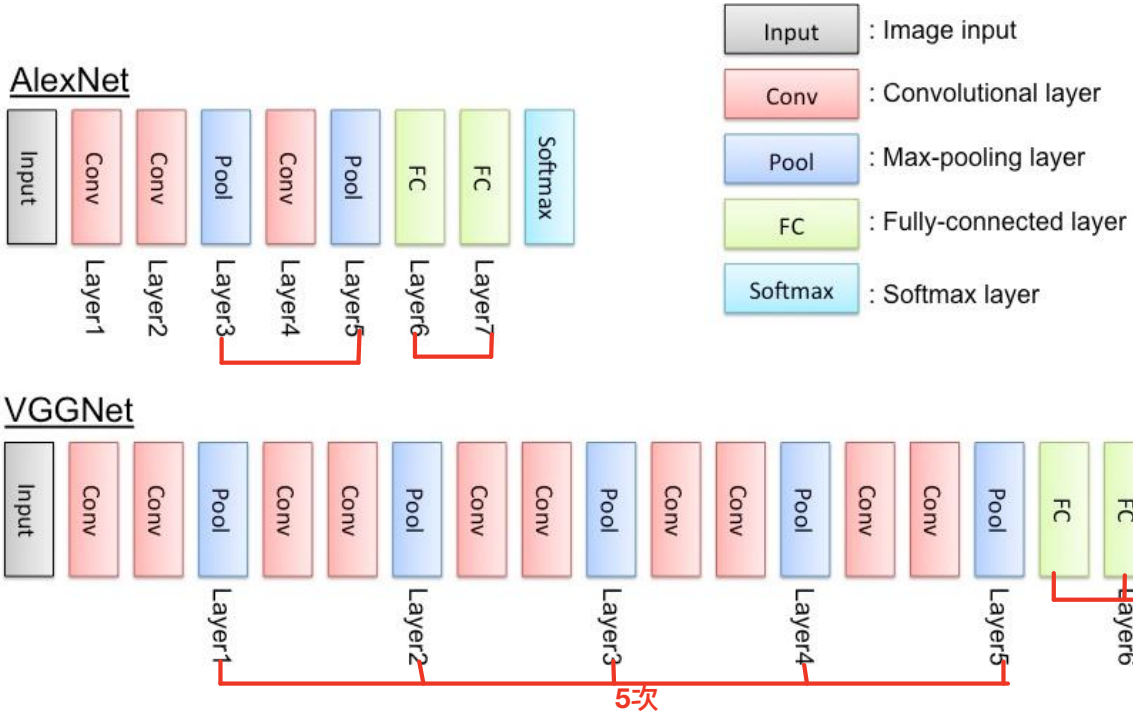


Structure of AlexNet

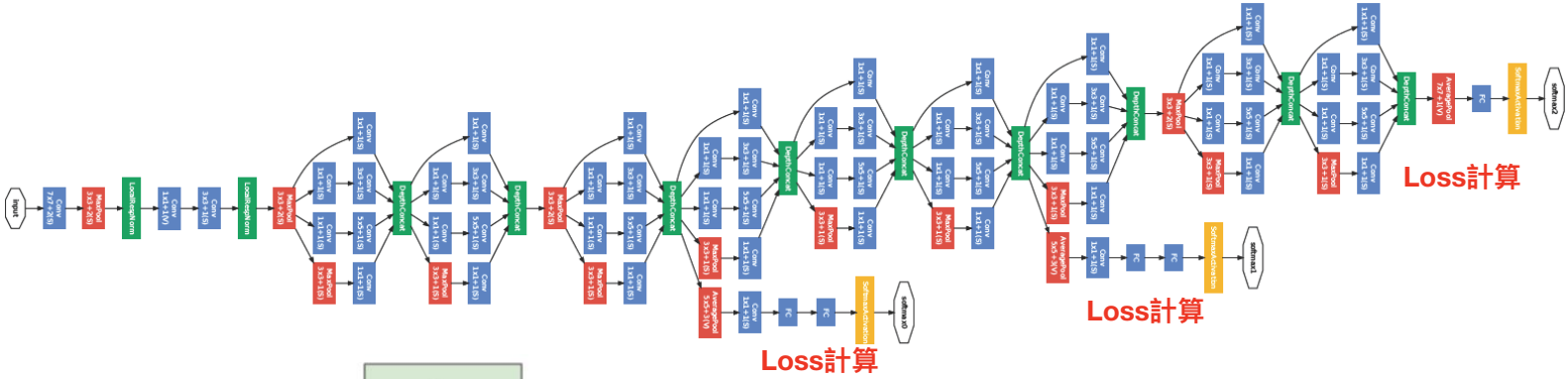


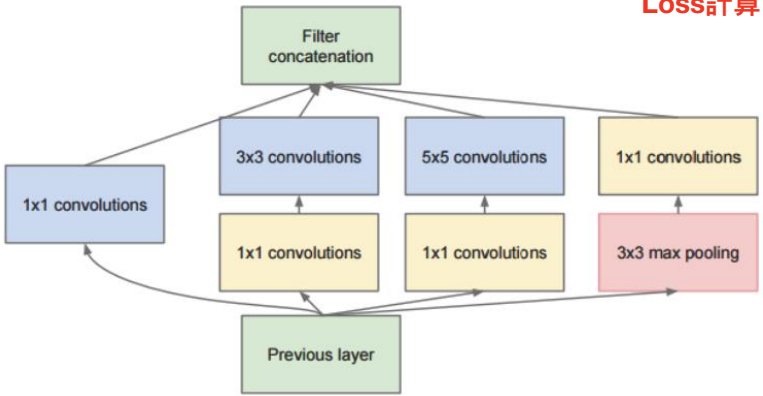
- Note Pooling considered part of the layer
- 96 convolution kernels, then 256, then 384
- Stride of 4 for first convolution kernel, 1 for the rest
- Pooling layers with 3x3 receptive fields and stride of 2 throughout
- Finishes with fully connected (fc) MLP with 2 hidden layers and 1000 output nodes for classes將局部區域做normalize
- Local Response Norm - the outputs of corresponding units of equivalent convolution layer from different kernels are used to normalize using neighbors of points or an accumulative quantity (say mean) over convolution output.

Structure of VGGNet



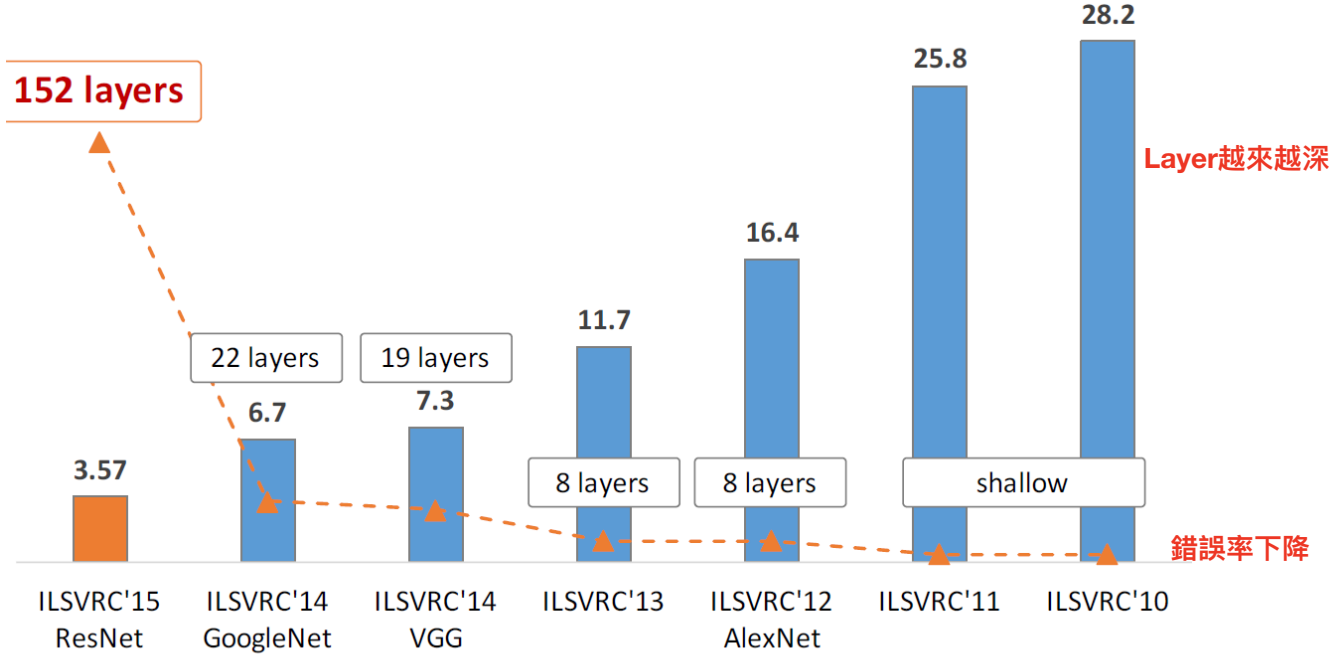
Structure of GoogleNet





The inception module
(convolutions and max pooling)

The Revolution of Depth Increasing



AlphaGo

19 x 19 matrix

Black: 1

white: -1

none: 0

Neural Network

Next move

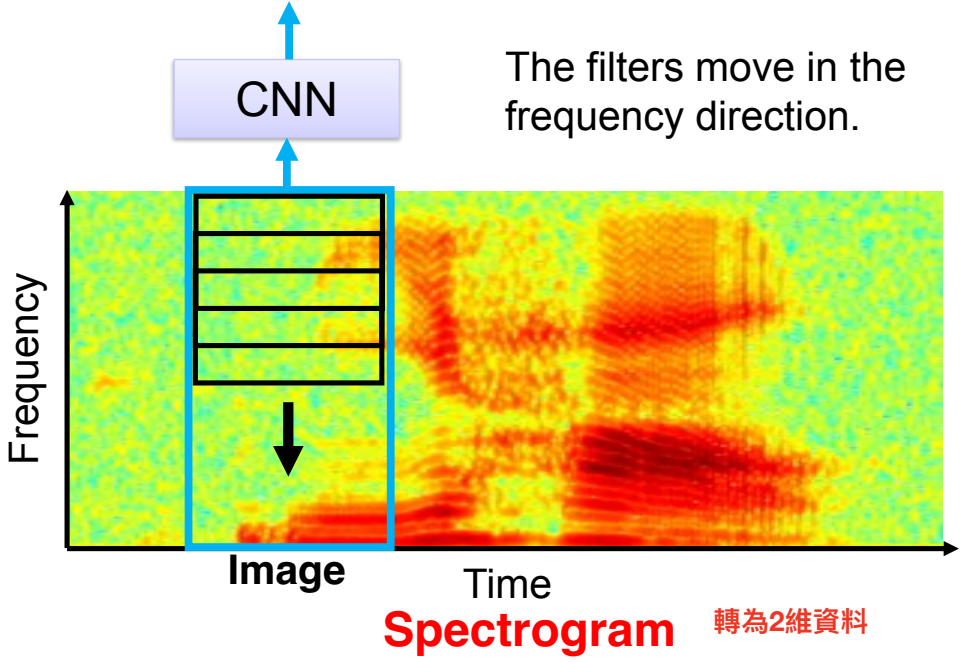
(19 x 19 positions)

機率，下在哪個機率最高

Fully-connected feedforward network can be used

But CNN performs much better

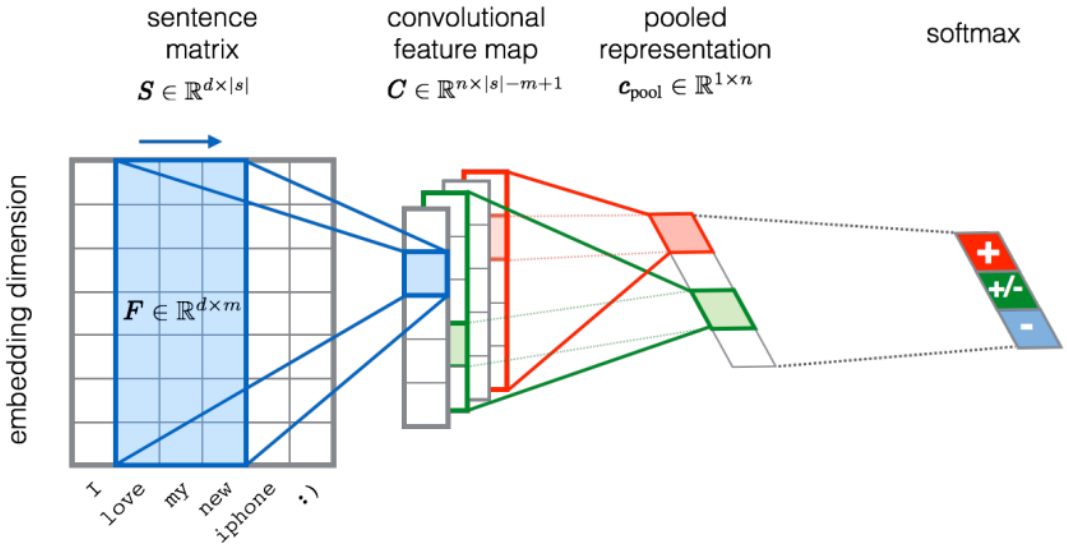
CNN in Speech Recognition



CNN in Text Classification

轉為向量表示

Word embedding

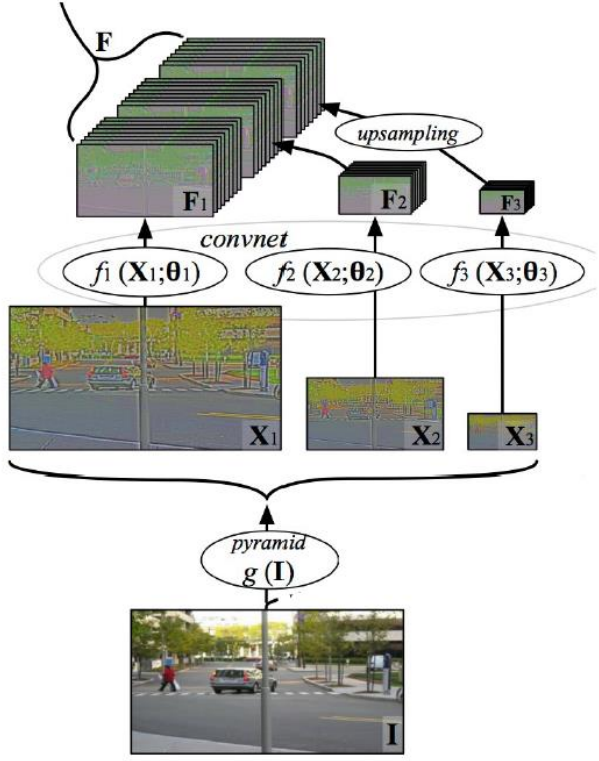


http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.703.6858&rep=rep1&type=pdf

Multi-Scale Deep Learning

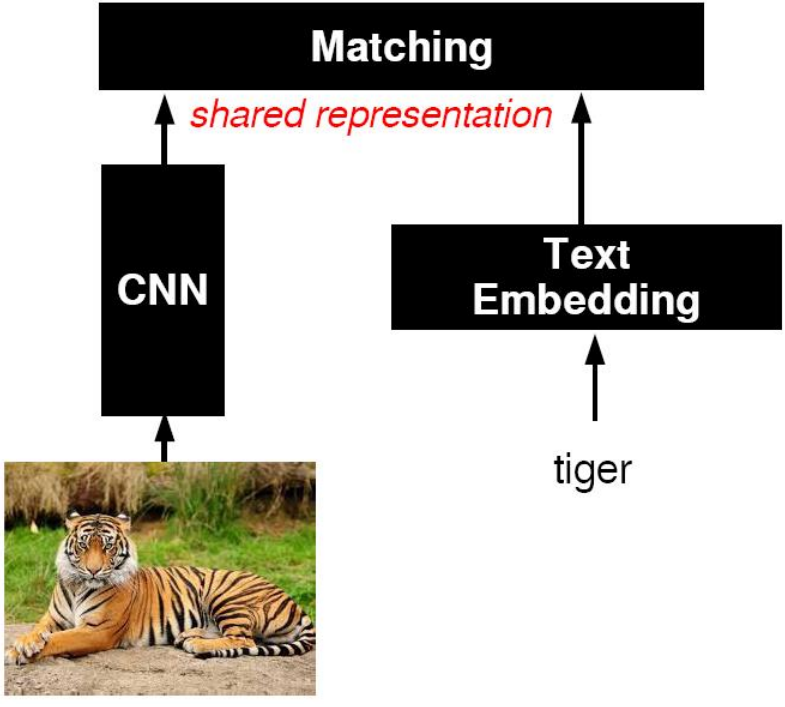
把圖縮放成不同scale

- One deep learning network is for fixed input image resolution.
- Train multiple networks with different input resolution.
- Concatenate features all together within one unified structure.



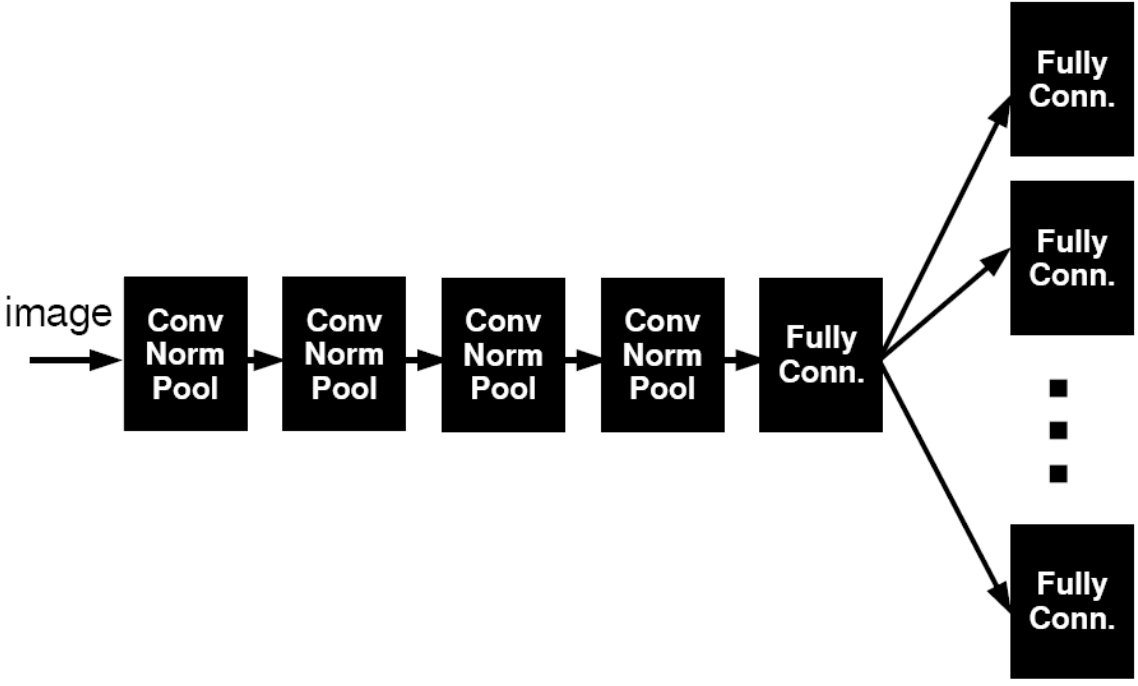
Multi-Model Deep Learning

- Handling image, text, voice or music within one network.
- The combination of features may benefit the classification for specific applications.



Multi-Task Deep Learning

同時知道很多結果



Classification label of actions

Joint locations (x,y) of actions

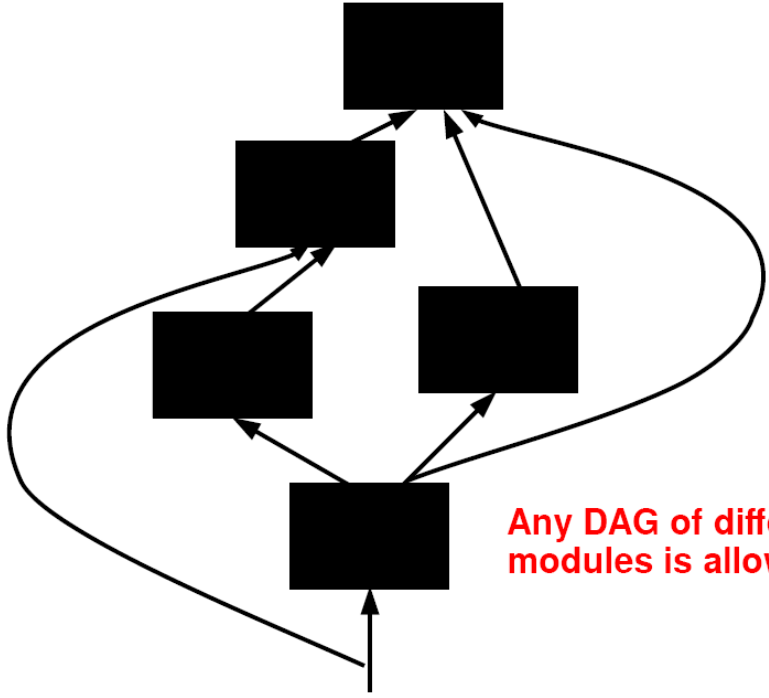
Pose recognition – play or holding

Places label where action happens

Generic DAG Deep Learning Structure

- Deep learning structure can be modified into any directed acyclic graph.

自行決定連結節點



Any DAG of differentialble modules is allowed!

Face Recognition

- Multiple patches
- Training data: 1.2M face images from 18K people

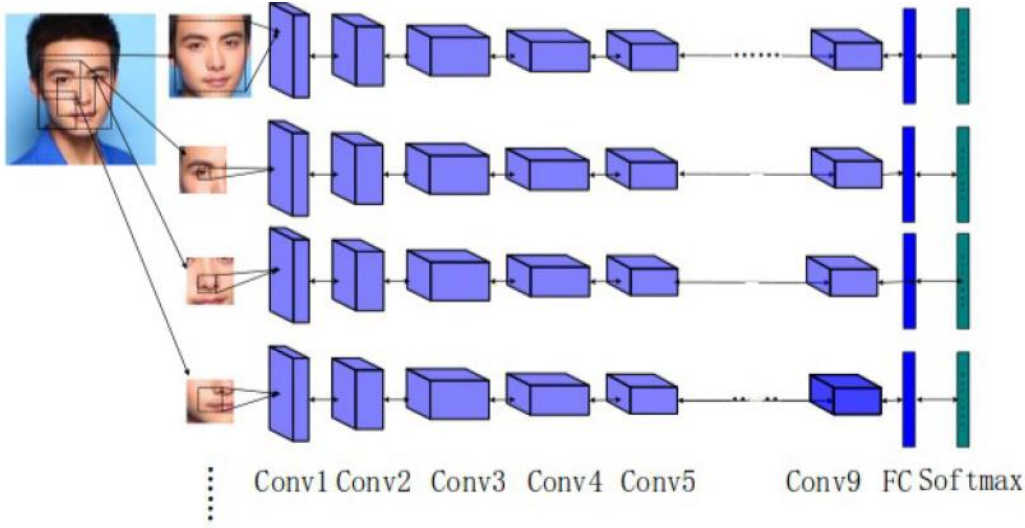
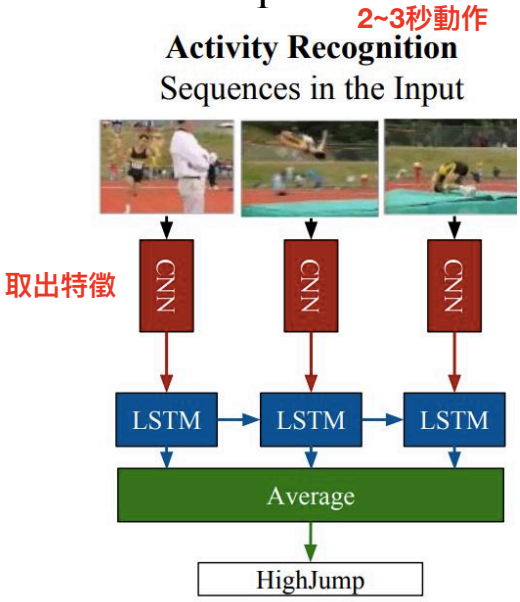
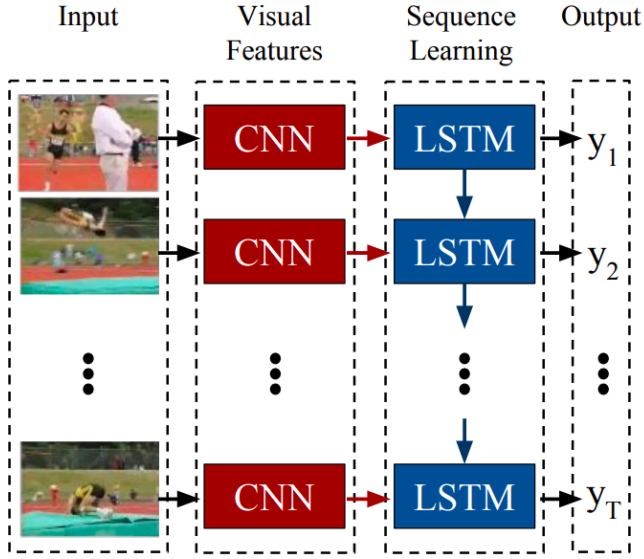


Figure 1. Overview of deep CNN structure on multi-patch.

Action Recognition

- Long-term Recurrent Convolutional Networks for Visual Recognition and Description





Action Recognition: Two Stream Networks

- Two separate networks –
  - one for spatial context (pre-trained): input is a single frame of the video.
  - one for motion context : optical flow stacked across for 10 successive frames.
- The two streams were trained separately and combined using SVM.

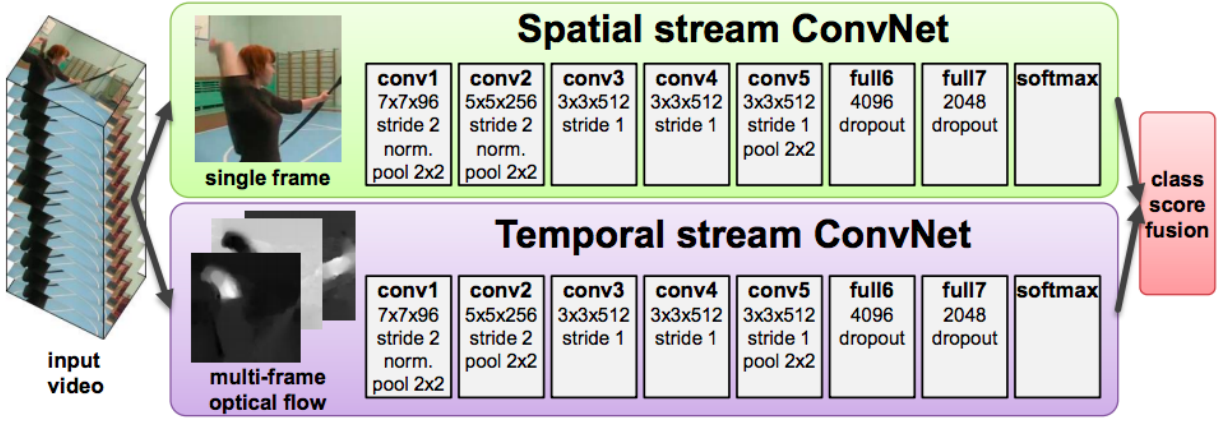
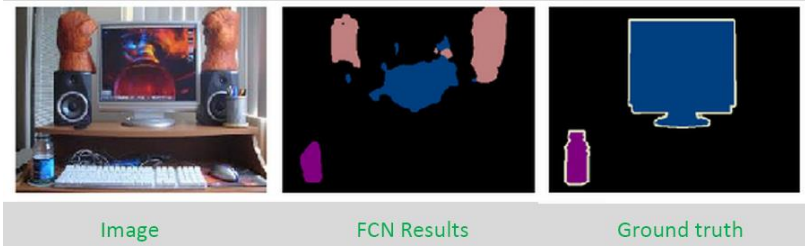
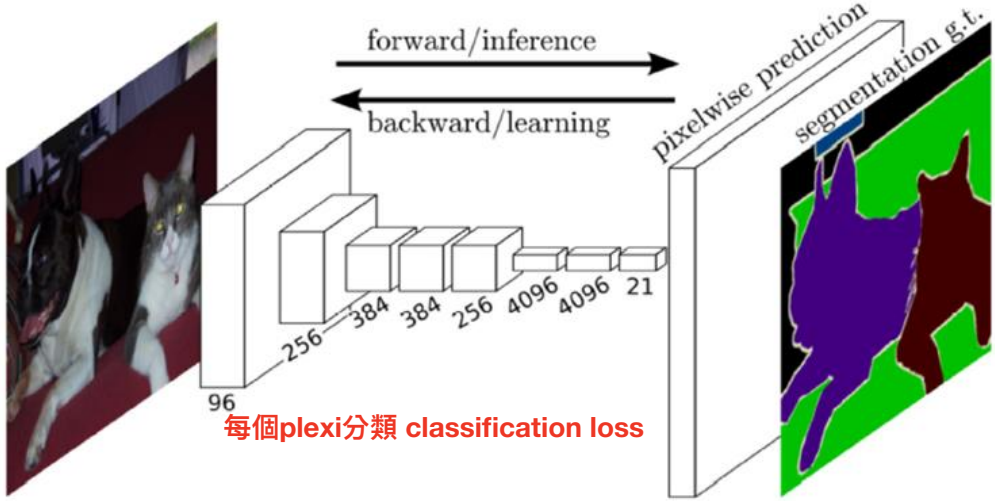


Image Segmentation

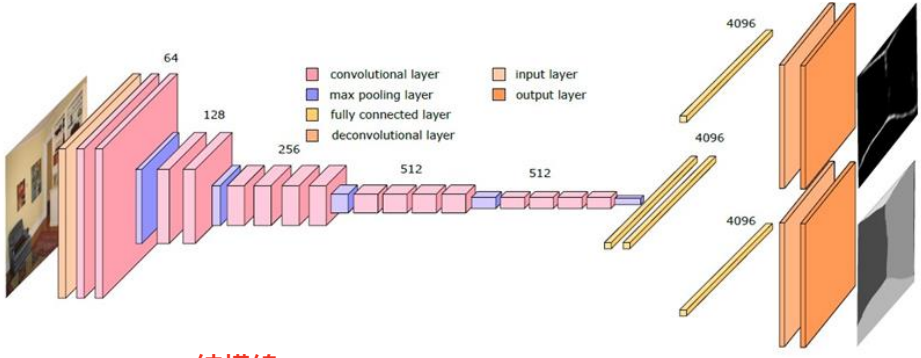
- Fully Convolutional Networks
  將畫面切塊





每個plexi分類 classification loss

Coarse-to-Fine Indoor Layout Estimation



結構線

Input → MFCN → Coarse Layout → Critical Line Detection → Layout Hypotheses and Ranking → Result