Day 17-18

深度學習與電腦視覺

學習馬拉松

cupoy 陪跑專家：楊哲寧

CUPOY

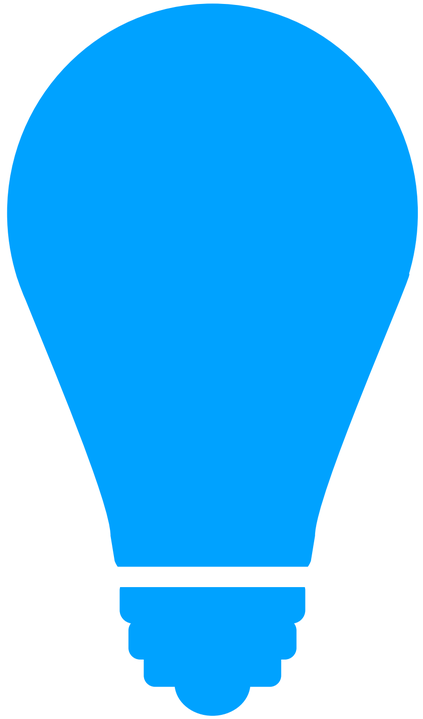
深度學習理論與實作

Classic CNN Backbone
(經典CNN框架)

重要知識點

CUPOY

- 了解Imagenet中CNN框架的演進
- 了解AlexNet、Vgg的優勢



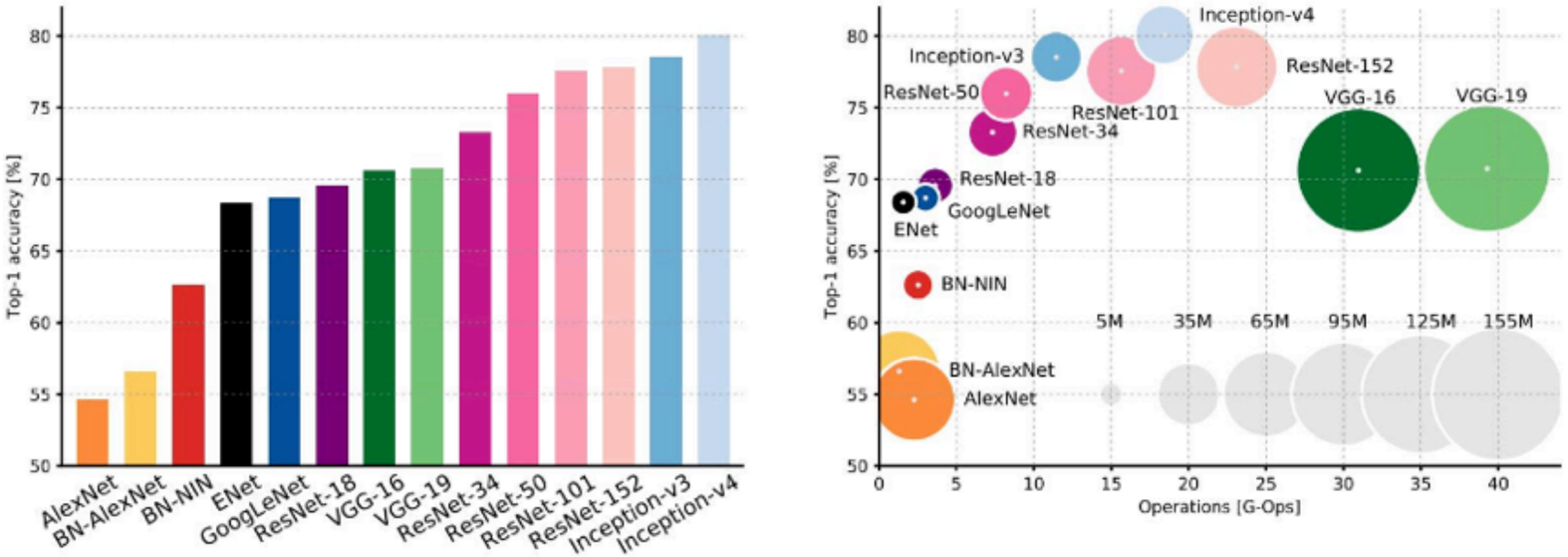
這次內容將分為兩天，
Day017為閱讀ppt內容，Day018實作模型，
時間較為充裕的學員們也可以另外挑戰搭建
Vgg19的模型(實作為搭建Vgg16模型)。

CNN 的演進

CUPOY

下圖為Imagenet中CNN框架的演進(2018前)，其中左圖可以看出各個Model在ImageNet中 **Top-1 Accuracy** 的表現，右圖X軸代表Model的運算量，圓形大小代表Model的參數量，而Y軸同樣為Top-1 Accuracy。

所謂的Top-1 Accuracy是指排名第一的類別與實際結果相符的準確率，
而Top-5 Accuracy是指排名前五的類別包含實際結果的準確率。



An Analysis of Deep Neural Network Models for Practical Applications, 2017.

參考來源： IT Read01

CNN 的演進

CUPOY

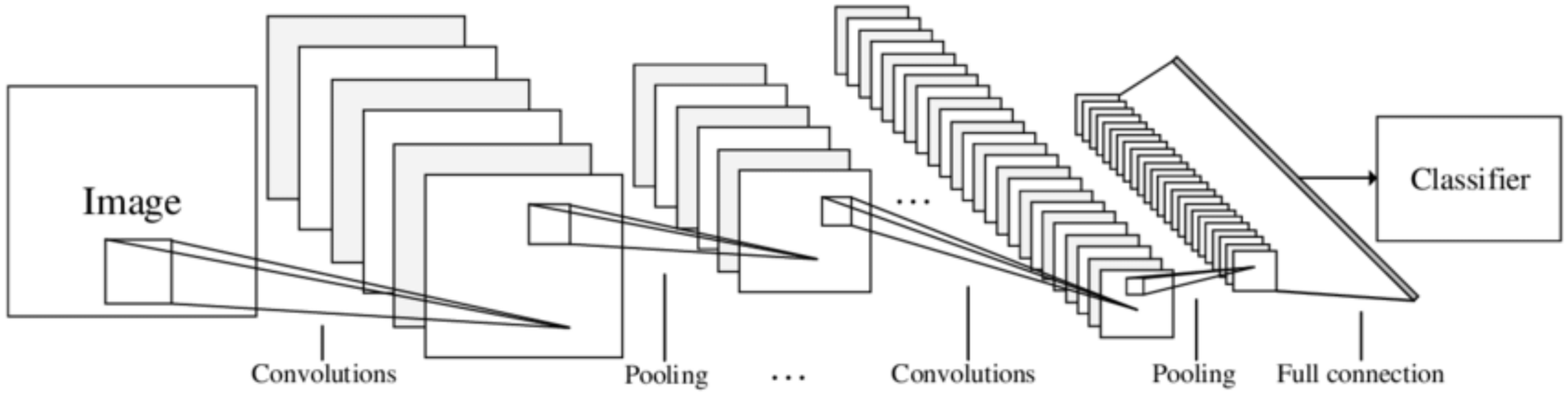
經典架構的演進：大多為ImageNet當年競賽的冠軍或前幾名。

- LeNet：CNN開宗始祖
- AlexNet：CNN正式在CV領域展露頭角
- Vgg：CNN經典框架，應用於諸多模型
- Inception：又稱GoogleNet，有V1-V4版本
- ResNet：奠定深層CNN結構

參考來源： Read01

LeNet

Yann LeCun 於 1998年發表的 『LeNet』為CNN架構的開宗始祖(論文連結)，其正式帶入了卷積的概念。論文中採用**5*5的卷積核**，Strides為 **1**，並運用MaxPooling強化特徵，輸出層採用了Radial Basis Function函式，即徑向歐式距離函式。由於LeNet歷史已經相當悠久，這裡就不會深度介紹，有興趣的學員們可參考論文連結。

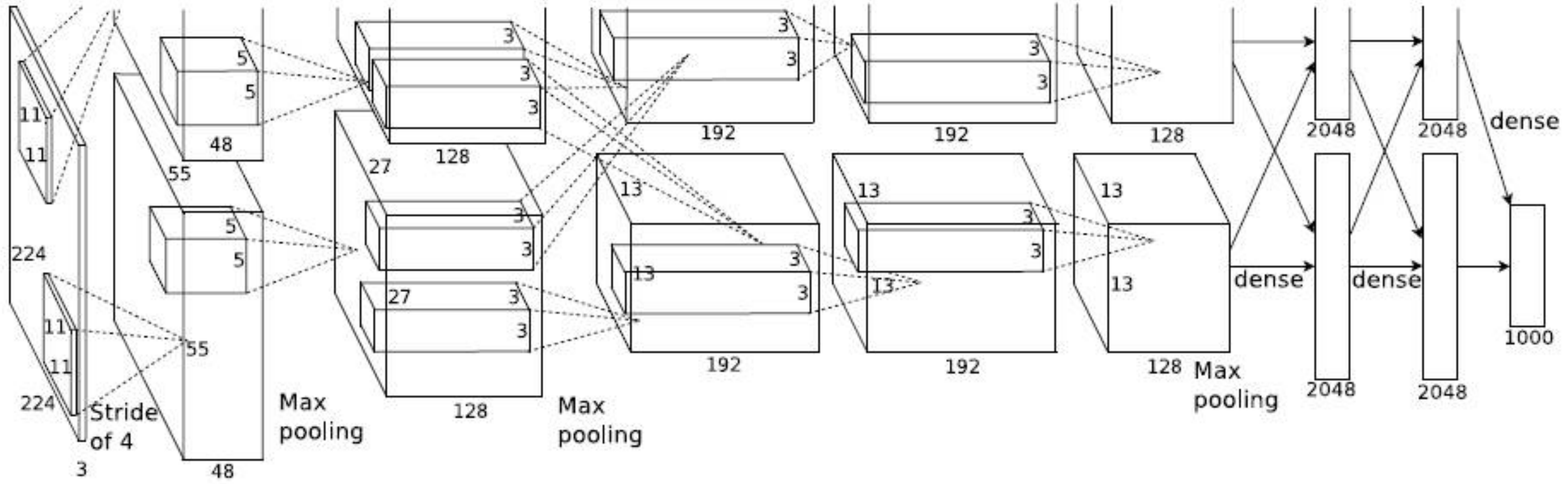


LeNet架構

參考來源： ResearchGate

AlexNet

AlexNet為 2012年ImageNet的冠軍(論文連結)，以作者Alex命名，其出現大幅打破了過去的紀錄，將top-5 error降低至15.3% ，可說是正式將CNN帶入Computer Vision領域的最大功臣。



AlexNet架構

參考來源： Medium

AlexNet

CUPOY

Structure	一共8層，前五層**卷積層**，後三層為**全連接層**
ReLU	提出 ReLU 取代這往的 Sigmoid
Augmentation	透過 Augmentation 增加資料，降低 Overfitting
Dropout	透過 Dropout 降低 Overfitting
LRN	Local Response Normalization

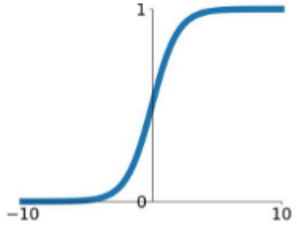
AlexNet-ReLU

CUPOY

ReLU的概念相當簡單，小於0的為0，大於0的值就是f(x)=x

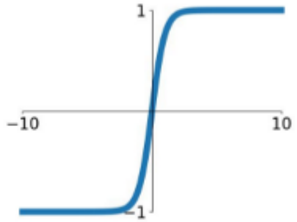
Sigmoid

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



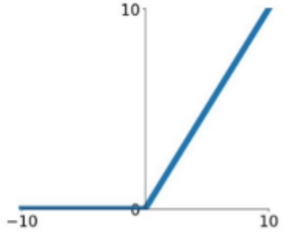
tanh

$$\tanh(x)$$



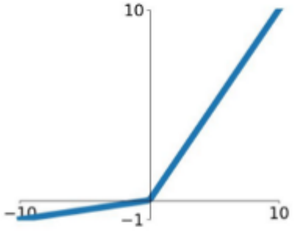
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

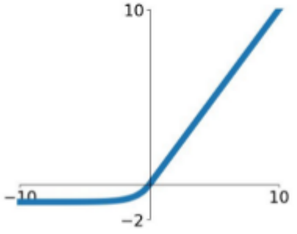


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

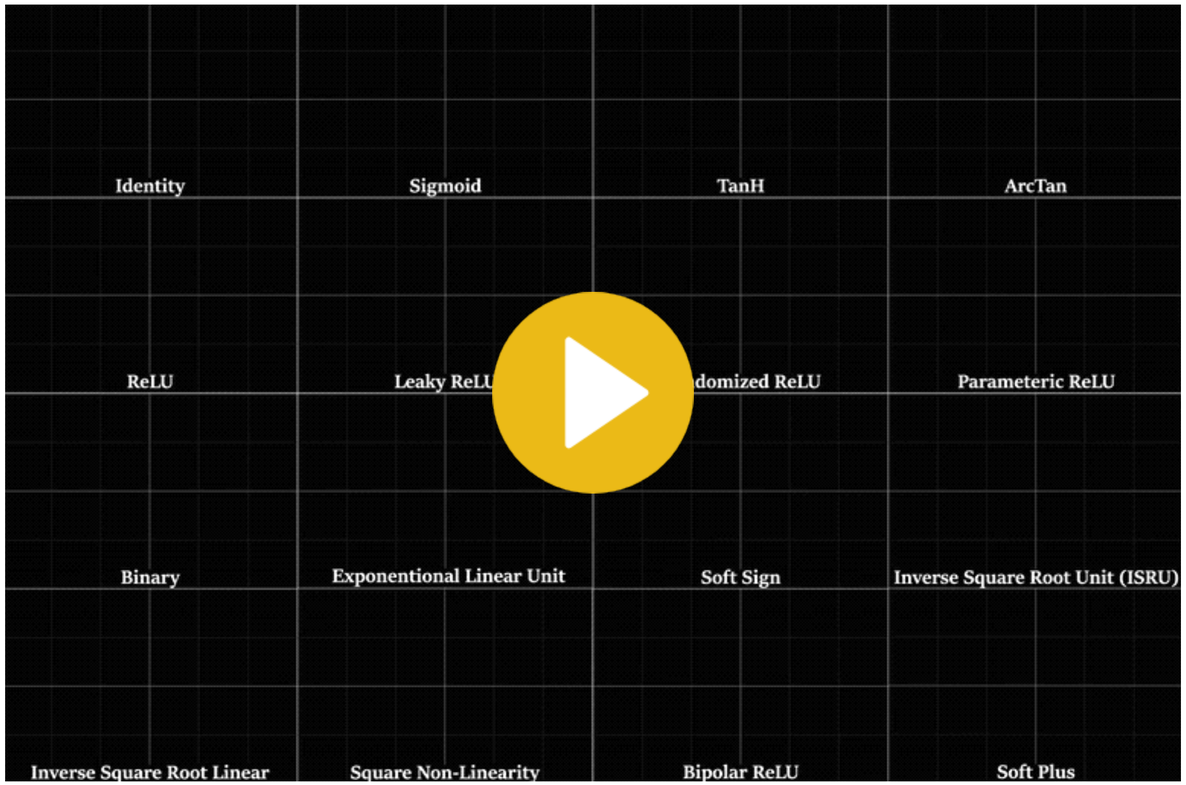


參考來源： Towards Data Science

AlexNet-ReLU

CUPOY

部分Github Code可能會用 Leaky ReLU取代ReLU，主要是由於用 Leaky ReLU 可改善 ReLU負值部分訊息消失的問題。



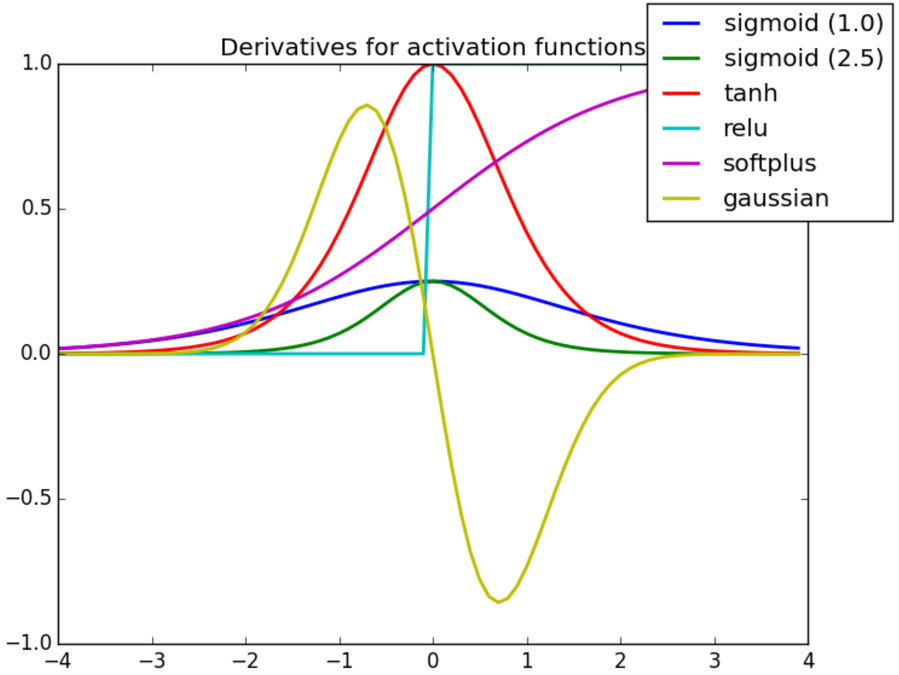
參考來源： MLFromScratch

AlexNet-ReLU

ReLU相較於Sigmoid的好處為『比較不容易造成梯度消失』，可以從下方導函數圖了解

Back Propagation 透過鏈鎖率來計算損失函數的梯度，由於Sigmoid的**導函數**最大值為**0.25**，當神經網路搭得比較深時，容易造成梯度消失 。

$$\frac{\partial C}{\partial w^{(1)}} = \underbrace{\frac{\partial C}{\partial a^{(4)}}\frac{\partial a^{(4)}}{\partial z^{(4)}}}_{\text{From } w^{(4)}} \underbrace{\frac{\partial z^{(4)}}{\partial a^{(3)}}\frac{\partial a^{(3)}}{\partial z^{(3)}}}_{\text{From } w^{(3)}} \underbrace{\frac{\partial z^{(3)}}{\partial a^{(2)}}\frac{\partial a^{(2)}}{\partial z^{(2)}}}_{\text{From } w^{(2)}} \frac{\partial z^{(2)}}{\partial a^{(1)}}\frac{\partial a^{(1)}}{\partial z^{(1)}}\frac{\partial z^{(1)}}{\partial w^{(1)}}$$



Activation Fucntion導函數

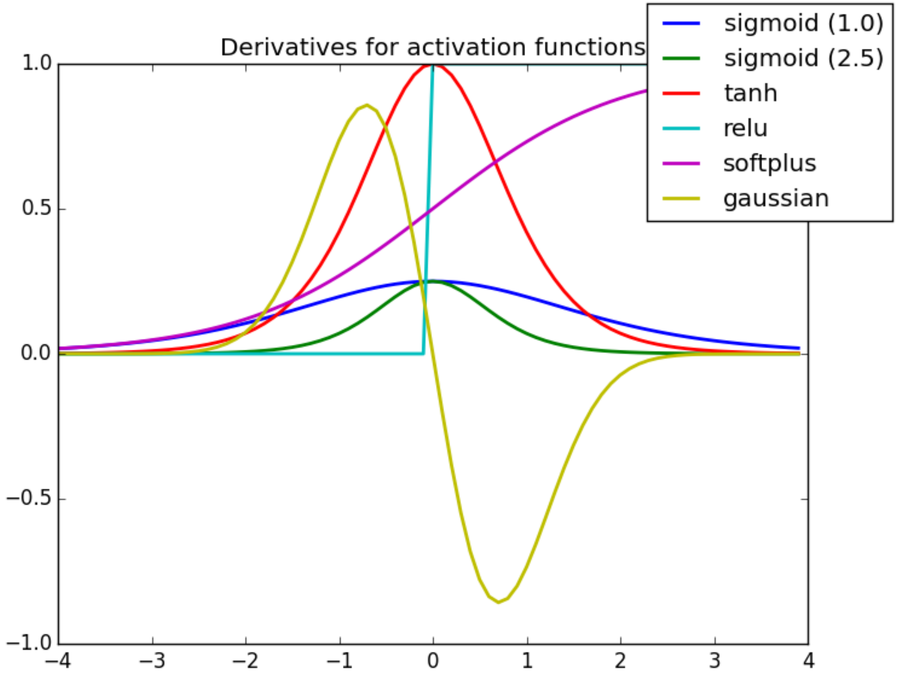
參考來源： Towards Data Science

AlexNet-ReLU

ReLU相較於Sigmoid的好處為『比較不容易造成梯度消失』，可以從下方導函數圖了解

Back Propagation 透過鏈鎖率來計算損失函數的梯度，由於Sigmoid的**導函數**最大值為**0.25**，當神經網路搭得比較深時，容易造成梯度消失 。

$$\frac{\partial C}{\partial w^{(1)}} = \underbrace{\frac{\partial C}{\partial a^{(4)}}\frac{\partial a^{(4)}}{\partial z^{(4)}}}_{\text{From } w^{(4)}} \underbrace{\frac{\partial z^{(4)}}{\partial a^{(3)}}\frac{\partial a^{(3)}}{\partial z^{(3)}}}_{\text{From } w^{(3)}} \underbrace{\frac{\partial z^{(3)}}{\partial a^{(2)}}\frac{\partial a^{(2)}}{\partial z^{(2)}}}_{\text{From } w^{(2)}} \frac{\partial z^{(2)}}{\partial a^{(1)}}\frac{\partial a^{(1)}}{\partial z^{(1)}}\frac{\partial z^{(1)}}{\partial w^{(1)}}$$



Activation Fucntion導函數

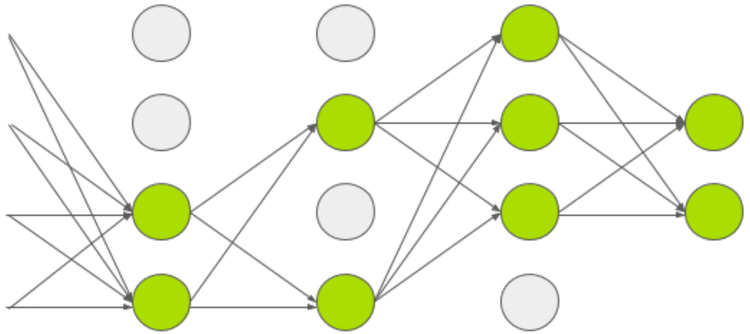
參考來源： Towards Data Science

AlexNet-Dropout

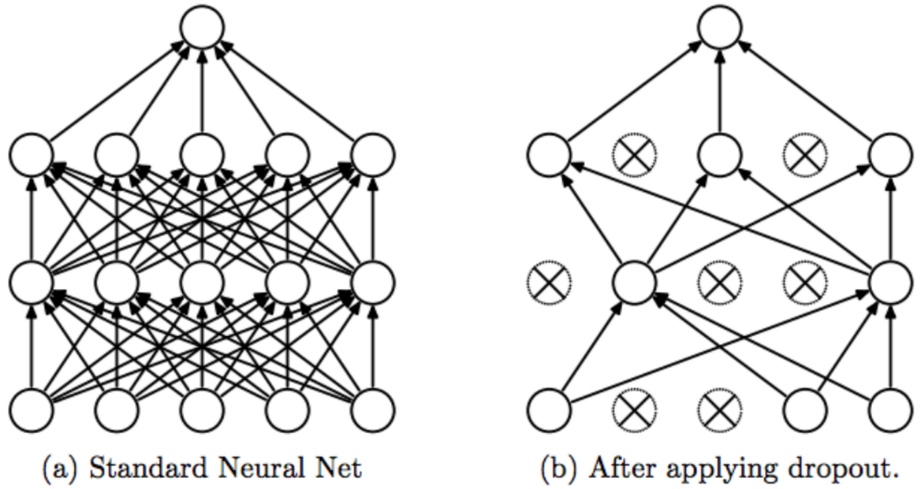
CUPOY

Dropout觀念相當直觀，就是在每次 Forward Propagation 時隨機關閉特定比例的神經元，避免模型 Overfitting。

AlexNet 中使用比例為 0.5，就是隨機關閉一半的神經元，目前較為常見的比例介於 0.1 - 0.3。



Dropout 動態示意圖

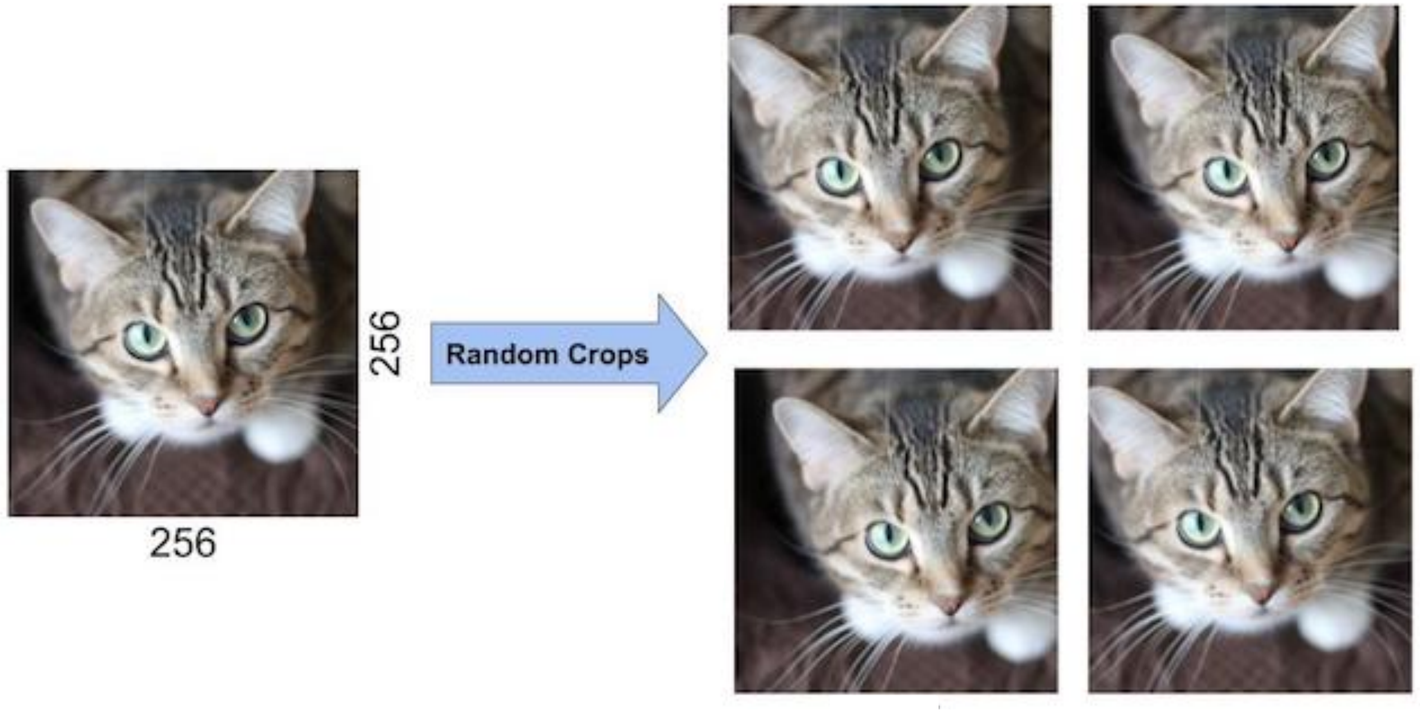


Dropout

參考來源： Learn OpenCV / Towards Data Science

AlexNet-Augmentation

Dropout觀念相當直觀，就是在每次 Forward Propagation 時隨機關閉特定比例的神經元，避免模型 Overfitting。



參考來源： Learn OpenCV

AlexNet-Local Response Normalization

LRN 顧名思義也是一種 Normalization 的方式，不過在近期研究中，發現添加 LRN 結構在模型中並不會提高準度，反而會增加記憶體使用量，因此已經沒有什麼文獻在使用 LRN，如有興趣的學員們可以參考這篇：

LRN 說明

$$b^i_{x,y} = a^i_{x,y} / (k + \alpha \sum_{j=max(0,i-n/2)}^{j=min(N-1,i+n/2)} {a^j_{x,y}}^2)^\beta$$

where

$b^i_{x,y}$ – regularized output for kernel i at position x, y

$a^i_{x,y}$ – source ouput of kernel i applied at position x, y

N – total number of kernels

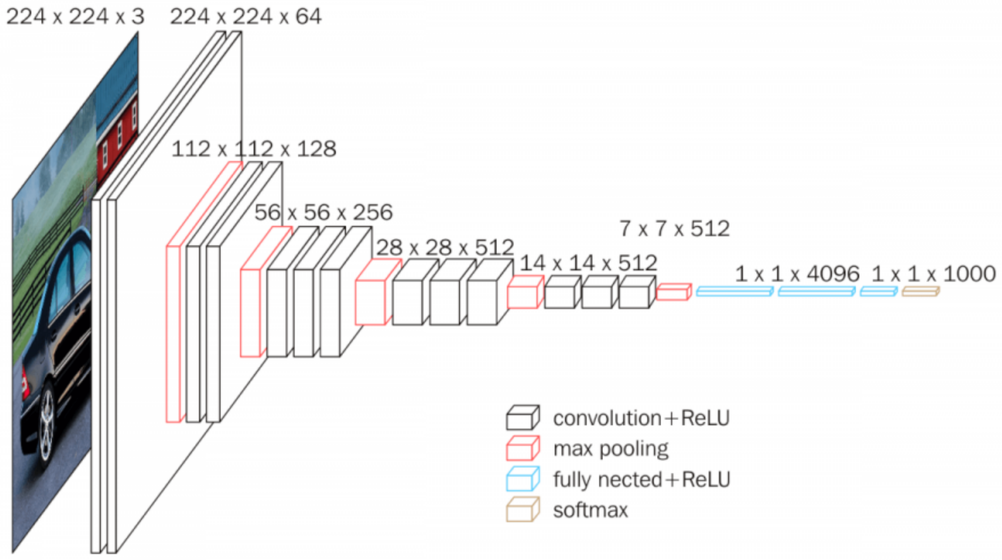
n – size of the normalization neigbourhood

$\alpha, \beta, k, (n)$ – hyperparameters

參考來源： Stackexchange

Vgg

Vgg算是相當經典的架構，不少CV框架都是使用Vgg為Backbone(Vgg論文)，Vgg其實有相當多種版本，然而Vgg-16、Vgg-19效果最好，因此以這兩種版本最為常見。



Vgg - 16 架構

參考來源： Neurohive

Vgg

CUPOY

Structure

論文中有 11-19 層的版本。

3*3 Kernel

全部使用 3*3 卷積核。

Vgg- 3*3 卷積核

CUPOY

Vgg 論文中提到，3*3 卷積核能在參數更少的狀態下，達到跟 5*5 或是更大卷積核一樣的效果，兩層 3*3 卷積核的 Receptive Field 與一層的5*5卷積核一樣，然而參數卻是 0.72 倍。

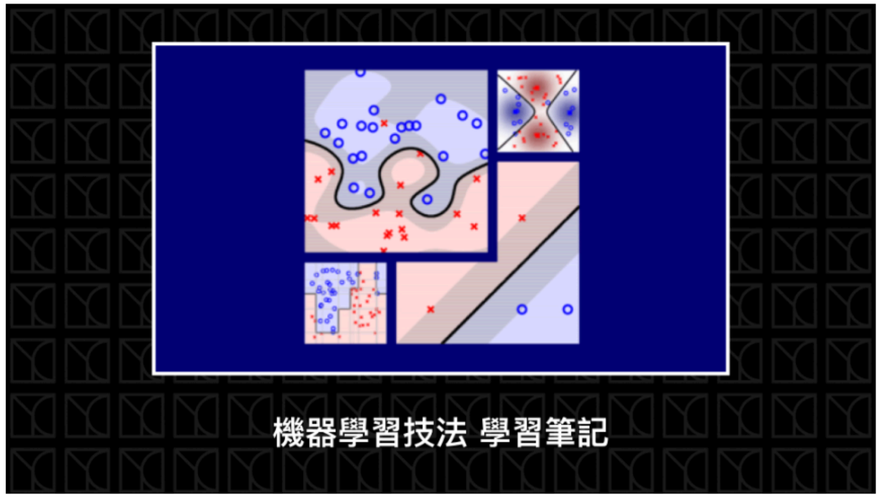
兩層 3*3 卷積核　　3*3*2 ＝18

一層 5*5 卷積核　　5*5*1 = 25

推薦延伸閱讀

FLOPS 與 GOPS 計算

連結



機器學習技法 學習筆記 (7)：Radial Basis Function Network與Matrix Factorization

YC Chen 2017-04-22 AI,ML 機器學習技法

RBF

連結

解題時間 Let’s Crack It

CUPOY

請跳出 PDF 至官網 Sample Code & 作業開始解題