DSP final

Visualizing and Understanding Convolutional Networks

(from DSP Chap 9.)

Matthew D. Zeiler (New York University)

Rob Fergus (New York University)

學號：r05922130

系級：資工所碩一

姓名：王瀚磊

1. Abstract

這篇paper提出一個可以有效提升Convolution Neural Network(CNN)的方法：將每個neural的output做視覺化。隨著CNN的發展，在圖型辨識方面可以達到越來越高的準確率，然而programmer往往不太清楚每個參數調整後，對這個CNN實際上的影響，因此在模型訓練的時候往往都是trial-and-error，透過不停地調整參數進而達到較高的準確率。這篇論文提出一個有效的方法去理解CNN架構中每一個neural對於影像辨識的實際意義，讓programmer可以有效地調整參數來達成最好的performance。

1. Introduction

在這篇文章一開始提及，CNN這項技術是由Yann LeCun在1989年所提出，他當時提出三個可以增強CNN準確率的方法：大量的labeled training data做監督式學習、強大效能的GPU藉以訓練複雜的模型、對模型做regularization例如dropout。

本文作者進而提出一種視覺化技術找出讓每個neural最excited的filter，以及能夠將input照片通過每層neural後顯示出來的照片透過視覺化技術將其轉換成人類看得懂的照片。如此一來我們就可以透過這些影像了解每個neural是藉由照片中的何種特徵辨認圖片。

此外這篇論文還有做了一個有趣的小實驗，他將照片中的不同區域遮擋後丟入CNN做判別，找出當哪一個區塊被遮擋後最能影響判斷的結果。

1. Approach

首先通過監督的方式利用Stochastic Gradient Descent來訓練CNN model，為了瞭解CNN每一層layer中的neural對於辨識影像的判斷基準，我們需要理解每個neural最能被激活的feature長什麼樣子。

簡單介紹一下這邊所訓練的CNN model架構，他在每層會做不同大小filter的convolution，並通過一個activation function：ReLu ( relu(x) = max(x,0) )。在部分layer中會在通過一個maxpooling，最後一層layer則做 N classifier。訓練過程則採用cross-entropy作為loss function，並且用back propagation 更新參數。

我們利用DeconvNet來將另neural最能被激活的feature轉換到pixel domain(0~255)觀察。其flow chart如圖1所示，分成三個步驟：Unpooling、Rectification、Filtering。

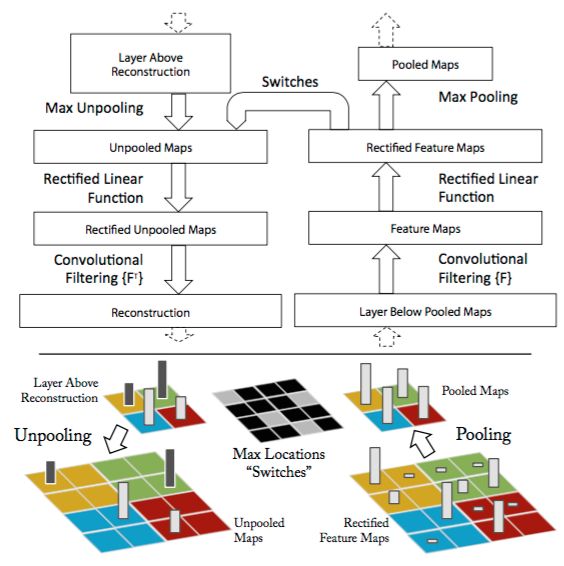


圖1. DeconvNet架構

* Unpooling：

上述提及CNN架構中會使用maxpooling，因此我們在做反向時需要利用unpooling技術才能一層一層往回將feature map推回到input image的大小。其實它所採用的原理很簡單，在做convolution的maxpooling時將每個window中的max value儲存起來，這樣在做unpooling時就可以透過查表找出max value應該放在何處，而window中其他的value則補0。（*在這部分我記得在ML課程的時候有聽老師提到其實這邊可以填0以外的值，例如做一些interpolation來改進影像在reconstruct所產生的error。*）

* Rectification：

在CNN架構中因為activation function為ReLu (non-linear)，可以確保輸出的值沒有負值。在做deconvNet時一樣需要確保output值為正數，這樣在最後做視覺化的時候才能產生看得懂的影像，因此這部分同樣通過一個ReLu (non-linear) function。

* Filtering：

在CNN架構中每一層neural即對影像做convolution，這步驟會將影像的size縮小。因此在做deconvNet時要做de-convolution。在李宏毅老師上課的投影片有提到，其實我們在做de-convolution的時候其實就是在做convolution，只是乘上不同的weight而已。而這邊的weight不同意思即為filter不同，即將row以及column都做flip。（*然而這邊內文所提到的上下左右顛倒即為deconvolution的filter我認為有點奇怪。*）

1. Training Details

在這個章節中會詳細介紹上述所提到的訓練模型。這篇文章所使用的架構與Krizhevsky et al., 2012相似，不過他們在第3,4,5層採用稀疏架構，因為他們用兩個GPU做training；而這篇文章採用一個GPU直接train。（當時的電腦沒有現在power！哈哈）

本文的training data set來自ImageNet,2012 (130萬張圖片，1000多種class)。他所做的preprocessing為擷取每張照片的中心256x256之RGB，減去其mean值，在做sub-sampling (10個224x224之照片)。採用SVG更新參數，batch\_size為128, learning rate為0.01, 動量係數為0.9。（這邊也有第二個疑問，內文提到採用SGD而非Adam等optimizer，怎麼會有動量呢？）而Dropout用在Flatten Layer之前，係數為0.5。詳細的架構圖如圖2.所示。

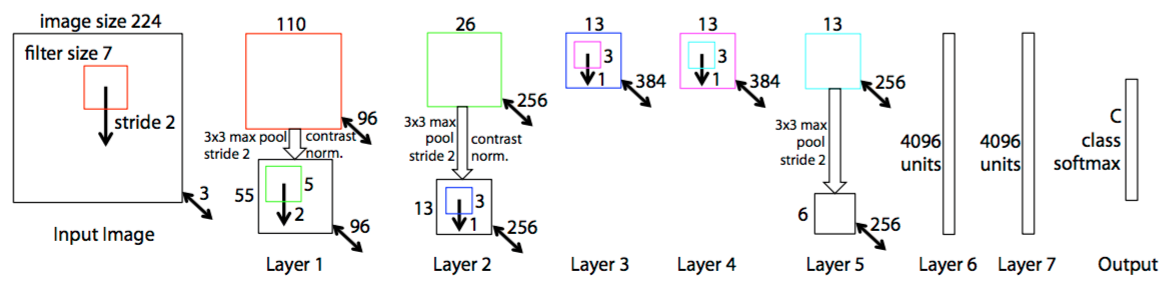


圖2. CNN model structure

然而他提及一個關鍵的操作，在第一層輸出的feature map中時常過大，如圖3的(a)所示。為了避免這種狀況，作者採用以下方法：當Root Mean Square Error(RMSE)超過0.1時對其做normalization。之所以說是很關鍵的原因為第一層的變化範圍很大(-128~128)。而如Yann Lecun所提及的，training data越多越能增進accuracy，這邊一樣做了一些水平移動以及翻轉照片增加training data。

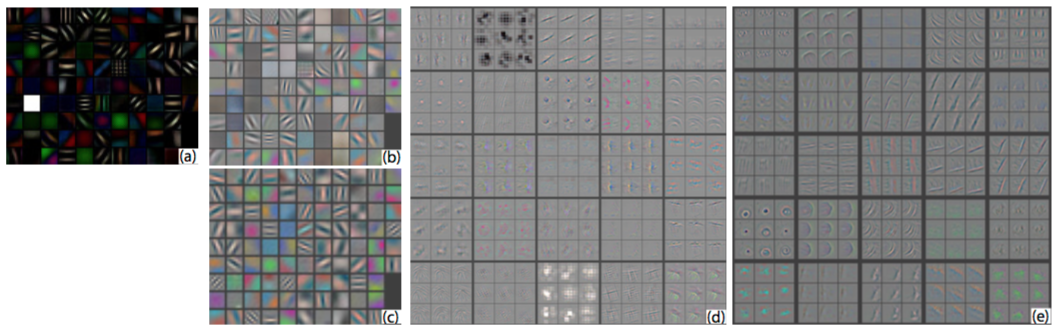


圖3. (a)沒做normalization之layer1 (b) Krizhevsky et al.之layer1 (c) 本文之layer1 (d) Krizhevsky et al.之layer2 (e) 本文之layer 2

1. ConvNet Visualization

這個章節為上述所提及之方法的result。這邊作者有三個重點：Feature Visualization、Feature Evolution during Training、Feature Invariance。

* Feature Visualization

圖4為在訓練過程中，由特定feature做deconvNet所得到的reconstruction image，每個layer分別展示了9個最容易被激活的feature以及將他們轉到pixel domain後的模樣。從這些圖片我們可以觀察到以下敘述：在第一層基本上只做了顏色的判斷；第二層開始判斷一些照片中的contour搭配顏色的組合；第三層則是開始變得觀察細微的contour，可以發現照片中的輪廓基本上都被描述出來了；第四層以及第五層則是做到最細緻的contour以及開始會做不同class之間差異的顯示，從圖4可以觀察到當layer到達第五層時，他在判斷貓狗的時候都可以把他們的頭部描述出來，而在判斷鍵盤的時候也能將鍵盤的輪廓表現出來。

* Feature Evolution during Training

圖4還可以觀察到在訓練過程中，特定feature所獲得的權重的改變。當輸入最能激活的feature發生變化時，其所對應的輪廓會開始變化。然而這樣的狀況在本文中說明只要在training的epoch越高時，這些feature會越趨穩定，而越高層所需train的epoch則需要越多。文章中認為只有在所有layer都成功收斂的時候才能做好正確的分類。

* Feature Invariance

圖5則展示了在照片經過平移、旋轉、縮放的時候，不同layer具有的feature invariant。文中認為CNN無法對旋轉產生feature invariant，除非照片具有對稱性。



圖4. Visualization of features in a fully trained model in different layers.其中每個layer顯示九個最能激活的filter

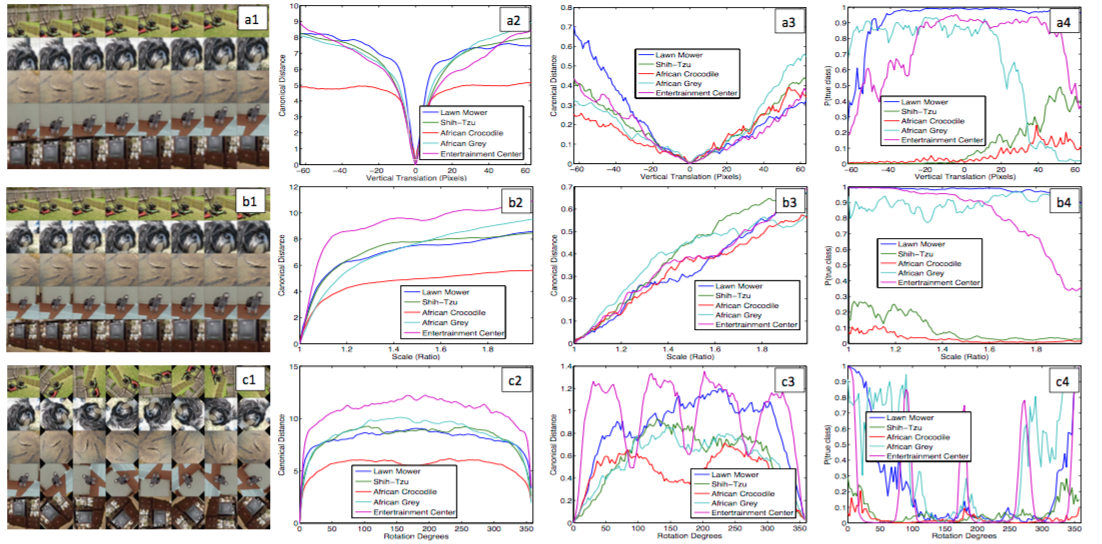


圖5. 圖片的(a)平移(b)旋轉以及(c)縮放所造成的feature invariance，(col2,3)為原始影像以及變形的影像分別在不同layer中feature間的euclidean distance，(col4)真實類別在分類中輸出的信心指數

1. Architecture Selection

觀察Krizhevsky的model以及利用本文的deconvNet技術可以觀察到一些問題。如圖3(a)以及(d)所示，第一層混雜了大量的high/low frequency information，缺少median frequency information；第二層由於window slide為4，產生了一些混亂的feature。為了解決這些問題作者將第一層window size由11x11條整成7x7，且window slide調整為2。新的結果為圖3之(c,e)所示。

1. Occlusion Sensitivity

當model提高準確率後作者想到一個問題：這些classifier究竟是用什麼資訊分類？是圖的上下文還是pixel value？因此作者使用一個灰色的mask遮住圖片中的不同部分藉以觀察分類氣輸出結果，當關鍵的地方被遮住後分類結果會急劇下降。其結果如圖6所示。

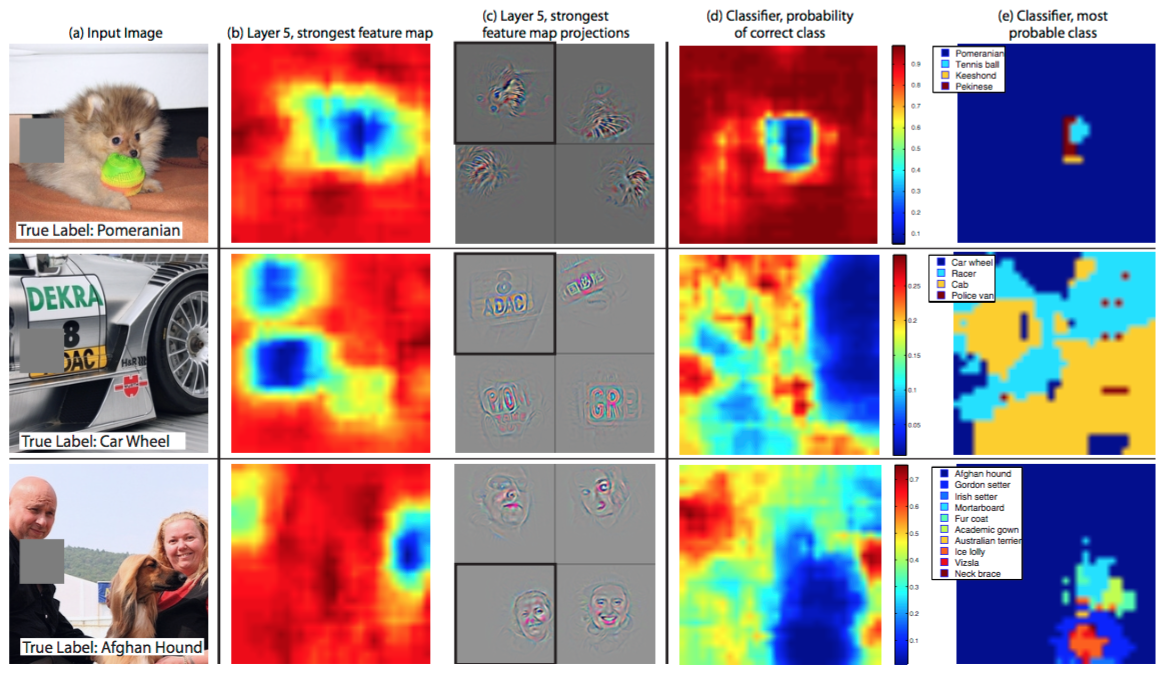


圖6. 遮擋不同地方對應到classifier所輸出的結果。(b)saliency map (c)利用deconvnet轉換至pixel domain (d)遮擋分佈後對應分類正確性的結果 (e)分類結果對應分佈

1. Correspondence Analysis

與其他許多已知的classifier model不同，CNN沒有一套有效理論來分析特定物體部位之間的關係。（例如：如何解釋人臉眼睛和鼻子在空間位置上的關係），但他很可能隱含是的計算了這些特徵。為了驗證這個假設，這篇論文隨機選擇五張狗狗的正面照，並遮擋一部分，如圖8所示。對於每張圖分別計算原始圖片和被遮擋圖片所產生的feature之間的差（相減），再針對所有結果計算hamming distance，值越小代表這些部位間存在越緊密的關係，其結果如表1顯示。



圖7. 遮擋不同部位，分別對應右眼左眼以及鼻子，其餘為隨機遮擋。

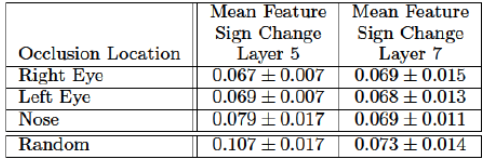


表1. 測試不同部位之間的相關性。在第五層中眼睛和鼻子的得分更低，代表開始產生相關性，而第七層值都差不多，代表已經不再關注局部特徵

1. Experiments
2. Image Net 2012

這個data set包含了130萬筆training data、5萬筆validation以及10萬筆testing data。這邊作者將他們利用deconvNet技術調整完Krizhevsky et al., 2012的model後與之比較testing data上的正確率差異，其結果如表2所示。

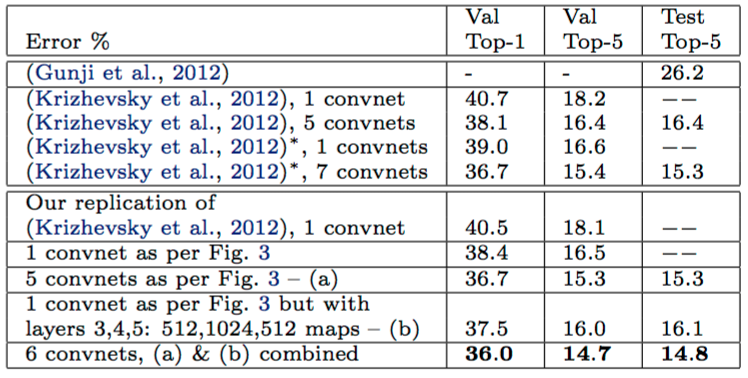


表2. ImageNet 2012 classification error rate

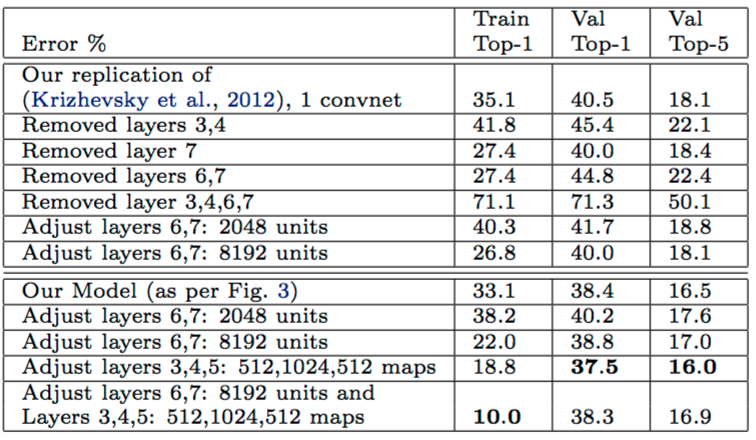


表3. ImageNet 2012 classification error rate with architecture change

此外，作者還特別針對ConvNet的結構做調整觀察其對於準確率之影響，他所得到的結論為CNN model的深度與分類效果有關聯。越深效果越好，且擴大hidden layer的unit個數也是有好的幫助，只是需要擔心overfitting，結果如圖3所示。

1. Feature Generalization

為了測試期generalization特性，這邊作者測試了Caltech-101、Caltech-256、PASCAL VOC 2012 三種data set。以下分別說明。

* + - Caltech-101：

這邊他們與Fei-fei et al., 2006的方法做比較，並成功超越Bo et al., 2013的成績，其詳細結果如表4所示。

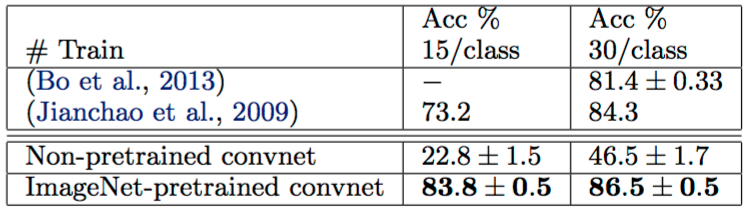


表4. Caltech-101 classification accuracy

* + - Caltech-256：

這邊採用的是Griffin et al., 2006的方法，結果如表5所示，並已超越19%之準確率超越當時最好的成績。

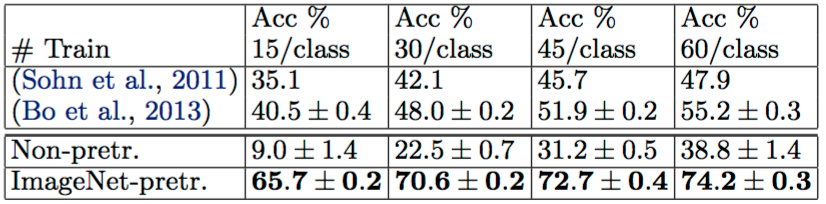


表5. Caltech-256 classification accuracy

* + - PASCAL VOC 2012：

文中敘述這邊用標準的方法訓練softmax classification。由於這個data set有可能是multi label，而本文作者提出的方法為single label，因此在分類上無法擊敗歷史最好的成績，然而在某些class的表現還是有取得優勢的，如表6所示。

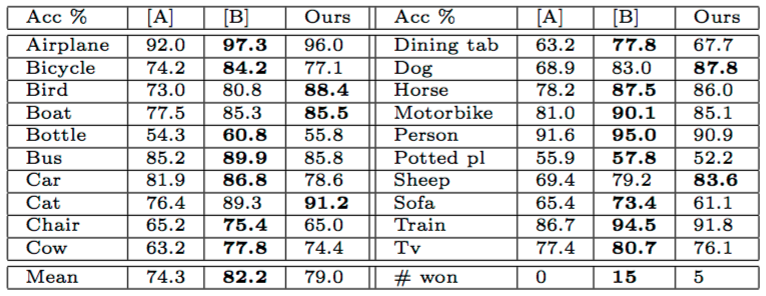


表6. PASCAL 2012 classification results

1. Feature Analysis

這邊想測試的事情為本文提出的架構所學習出的feature是否適用於SVM，其結果如表7所示，很明顯是可以共用的。此外作者透過不同的layer數目再次驗證了當深度增加的時，model可以學到更好的feature。

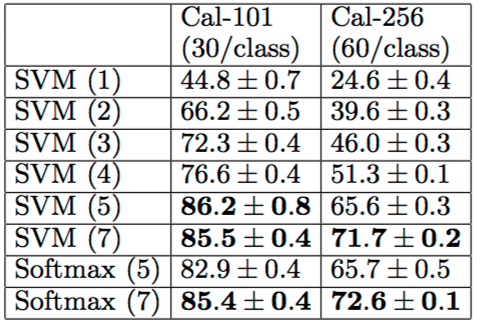


表7. SVM與Softmax連接不同layer數目的分類結果

1. Conclusion

這篇文章主要介紹了一種將CNN model的每個neural產生出來的結果做deconvNet將其轉為pixel domain，使我們人類看得懂意義的圖片之方法。並且可藉由這項技術調整model使其improve。此外作者還做了一個小實驗藉由遮擋影像中不同的位置找出CNN model在判斷照片時所需仰賴的feature。且作者也額外驗證了平移、縮放的feature invariant，對於旋轉則除非照片本身具對稱性，否則不具feature invariant。在實驗這個章節中，作者將其與不同的data set做比對，結果證實確實對於accuracy是有所提升的。

1. Reference
2. Zeiler M.D., Fergus R. (2014) Visualizing and Understanding Convolutional Networks. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8689. Springer, Cham
3. <https://www.zybuluo.com/lutingting/note/459569>
4. http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML\_2017/Lecture/auto.pdf