

# 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

## 0. 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。

## 2. 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 ( $\text{relu}(x) = \max(x, 0)$ )。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。

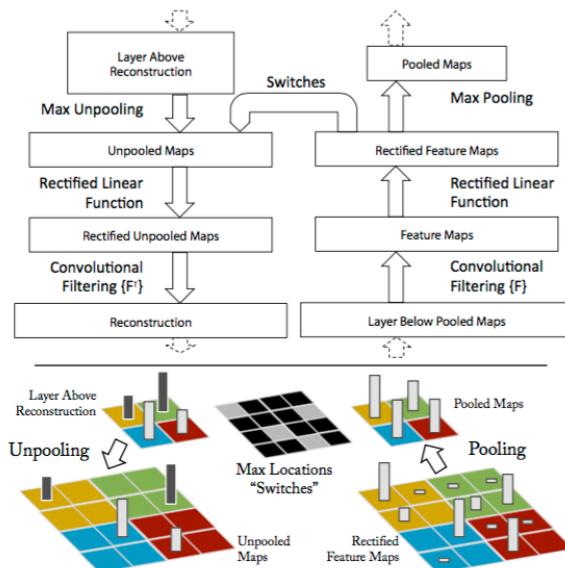


圖 1. DeconvNet 架構

- 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。)

### 3. 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 SGD 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。

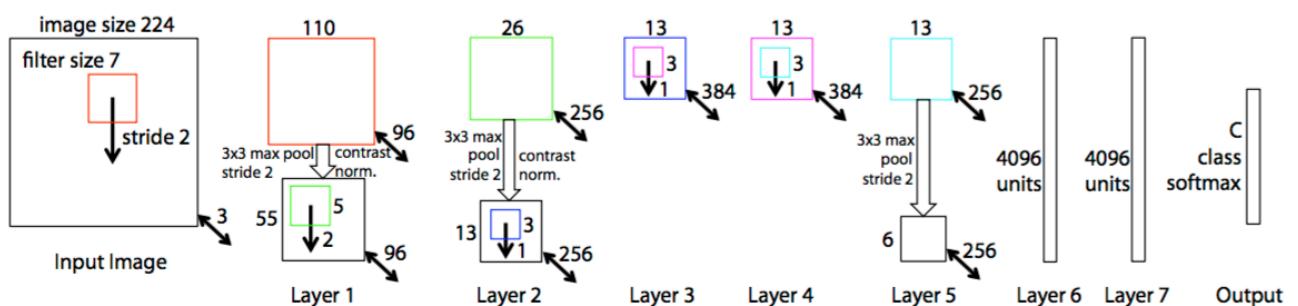


圖 2. CNN model structure

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。

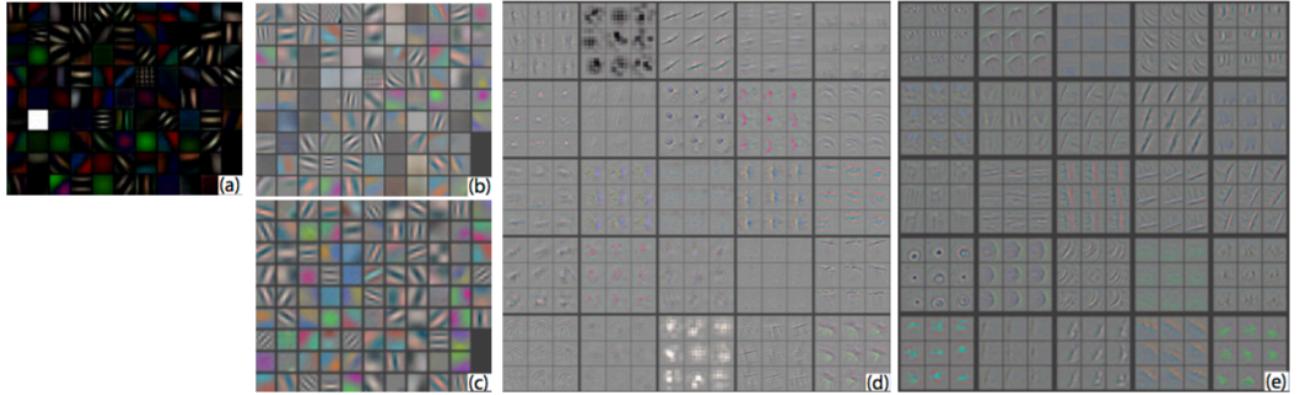


圖 3. (a)沒做 normalization 之 layer1 (b) Krizhevsky et al. 之 layer1 (c) 本文之 layer1 (d) Krizhevsky et al. 之 layer2 (e) 本文之 layer 2

## 4. 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

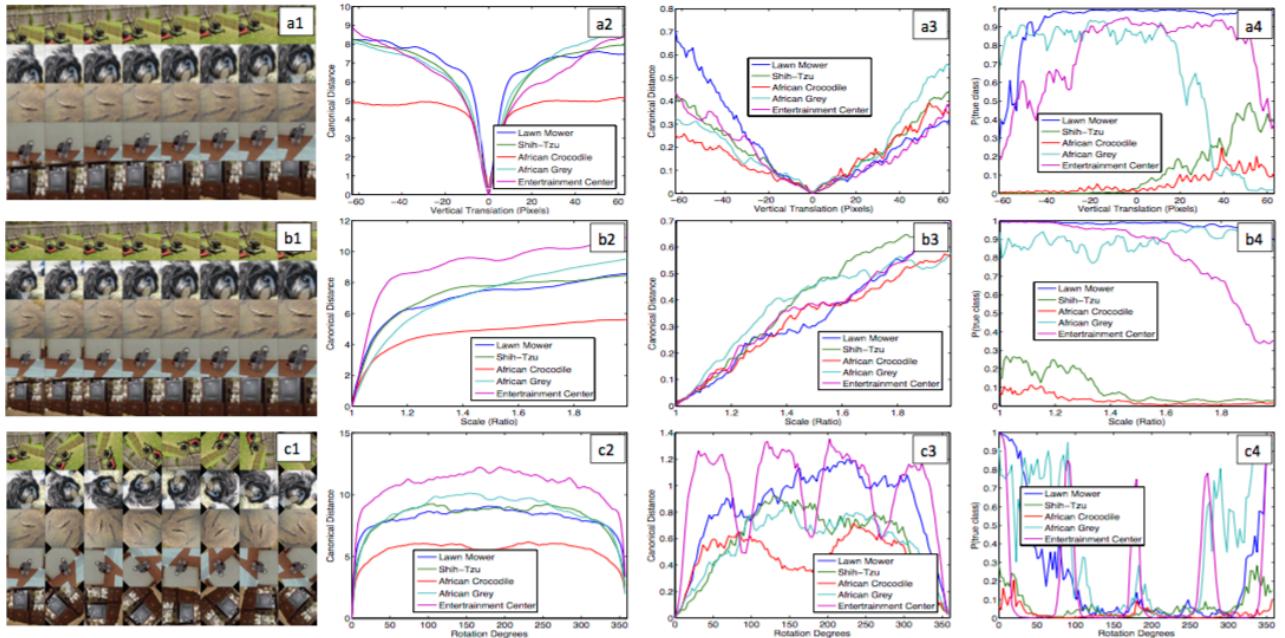


圖 5. 圖片的(a)平移(b)旋轉以及(c)縮放所造成的 feature invariance , (col2,3)為原始影像以及變形的影像分別在不同 layer 中 feature 間的 euclidean distance , (col4)真實類別在分類中輸出的信心指數

#### A. 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  $11 \times 11$  條整成  $7 \times 7$ ，且 window slide 調整為 2。新的結果為圖 3 之(c,e)所示。

#### B. 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。

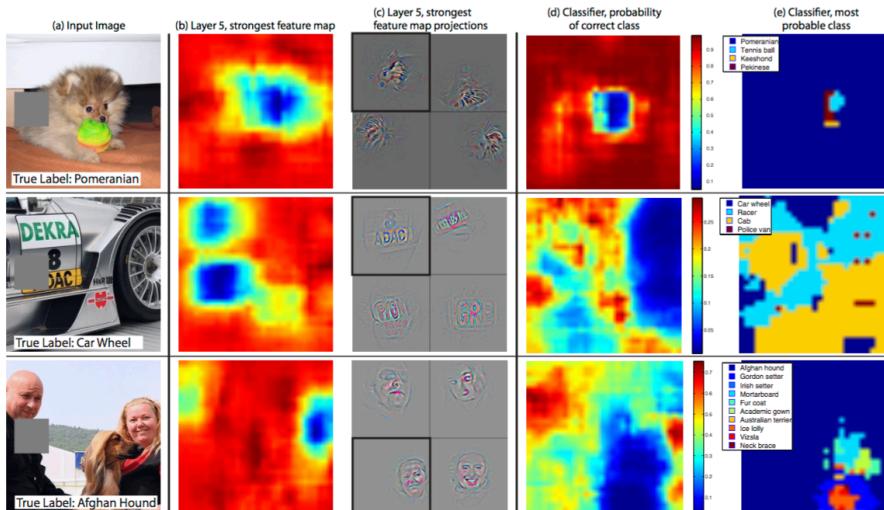


圖 6. 遮擋不同地方對應到 classifier 所輸出的結果 。(b)saliency map (c)利用 deconvnet 轉換至 pixel domain (d)遮擋分佈後對應分類正確性的結果 (e)分類結果對應分佈

### C. 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。

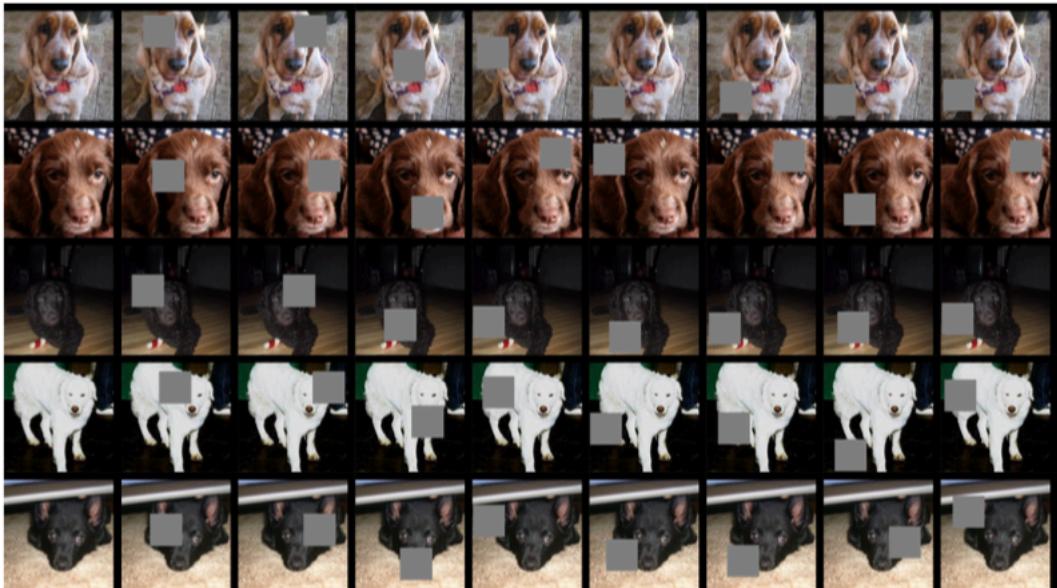


圖 7. 遮擋不同部位，分別對應右眼左眼以及鼻子，其餘為隨機遮擋。

Occlusion Location	Mean Feature Sign Change Layer 5	Mean Feature Sign Change Layer 7
Right Eye	$0.067 \pm 0.007$	$0.069 \pm 0.015$
Left Eye	$0.069 \pm 0.007$	$0.068 \pm 0.013$
Nose	$0.079 \pm 0.017$	$0.069 \pm 0.011$
Random	$0.107 \pm 0.017$	$0.073 \pm 0.014$

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

## 5. Experiments

### A. 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。

Error %	Val Top-1	Val Top-5	Test Top-5
(Gunji et al., 2012)	-	-	26.2
(Krizhevsky et al., 2012), 1 convnet	40.7	18.2	---
(Krizhevsky et al., 2012), 5 convnets	38.1	16.4	16.4
(Krizhevsky et al., 2012)*, 1 convnets	39.0	16.6	---
(Krizhevsky et al., 2012)*, 7 convnets	36.7	15.4	15.3
Our replication of (Krizhevsky et al., 2012), 1 convnet	40.5	18.1	---
1 convnet as per Fig. 3	38.4	16.5	---
5 convnets as per Fig. 3 – (a)	36.7	15.3	15.3
1 convnet as per Fig. 3 but with layers 3,4,5: 512,1024,512 maps – (b)	37.5	16.0	16.1
6 convnets, (a) & (b) combined	<b>36.0</b>	<b>14.7</b>	<b>14.8</b>

表 2. ImageNet 2012 classification error rate

Error %	Train Top-1	Val Top-1	Val Top-5
Our replication of (Krizhevsky et al., 2012), 1 convnet	35.1	40.5	18.1
Removed layers 3,4	41.8	45.4	22.1
Removed layer 7	27.4	40.0	18.4
Removed layers 6,7	27.4	44.8	22.4
Removed layer 3,4,6,7	71.1	71.3	50.1
Adjust layers 6,7: 2048 units	40.3	41.7	18.8
Adjust layers 6,7: 8192 units	26.8	40.0	18.1
Our Model (as per Fig. 3)	33.1	38.4	16.5
Adjust layers 6,7: 2048 units	38.2	40.2	17.6
Adjust layers 6,7: 8192 units	22.0	38.8	17.0
Adjust layers 3,4,5: 512,1024,512 maps	18.8	<b>37.5</b>	<b>16.0</b>
Adjust layers 6,7: 8192 units and Layers 3,4,5: 512,1024,512 maps	<b>10.0</b>	38.3	16.9

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。

## B. 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。

# Train	Acc % 15/class	Acc % 30/class
(Bo et al., 2013)	—	81.4 ± 0.33
(Jianchao et al., 2009)	73.2	84.3
Non-pretrained convnet	22.8 ± 1.5	46.5 ± 1.7
ImageNet-pretrained convnet	<b>83.8 ± 0.5</b>	<b>86.5 ± 0.5</b>

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。

# Train	Acc % 15/class	Acc % 30/class	Acc % 45/class	Acc % 60/class
(Sohn et al., 2011)	35.1	42.1	45.7	47.9
(Bo et al., 2013)	40.5 ± 0.4	48.0 ± 0.2	51.9 ± 0.2	55.2 ± 0.3
Non-pretr.	9.0 ± 1.4	22.5 ± 0.7	31.2 ± 0.5	38.8 ± 1.4
ImageNet-pretr.	<b>65.7 ± 0.2</b>	<b>70.6 ± 0.2</b>	<b>72.7 ± 0.4</b>	<b>74.2 ± 0.3</b>

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。

Acc %	[A]	[B]	Ours	Acc %	[A]	[B]	Ours
Airplane	92.0	<b>97.3</b>	96.0	Dining tab	63.2	<b>77.8</b>	67.7
Bicycle	74.2	<b>84.2</b>	77.1	Dog	68.9	83.0	<b>87.8</b>
Bird	73.0	80.8	<b>88.4</b>	Horse	78.2	<b>87.5</b>	86.0
Boat	77.5	85.3	<b>85.5</b>	Motorbike	81.0	<b>90.1</b>	85.1
Bottle	54.3	<b>60.8</b>	55.8	Person	91.6	<b>95.0</b>	90.9
Bus	85.2	<b>89.9</b>	85.8	Potted pl	55.9	<b>57.8</b>	52.2
Car	81.9	<b>86.8</b>	78.6	Sheep	69.4	79.2	<b>83.6</b>
Cat	76.4	89.3	<b>91.2</b>	Sofa	65.4	<b>73.4</b>	61.1
Chair	65.2	<b>75.4</b>	65.0	Train	86.7	<b>94.5</b>	91.8
Cow	63.2	<b>77.8</b>	74.4	Tv	77.4	<b>80.7</b>	76.1
Mean	74.3	<b>82.2</b>	79.0	# won	0	<b>15</b>	5

表 6. PASCAL 2012 classification results

## C. 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。

	Cal-101 (30/class)	Cal-256 (60/class)
SVM (1)	44.8 ± 0.7	24.6 ± 0.4
SVM (2)	66.2 ± 0.5	39.6 ± 0.3
SVM (3)	72.3 ± 0.4	46.0 ± 0.3
SVM (4)	76.6 ± 0.4	51.3 ± 0.1
SVM (5)	<b>86.2 ± 0.8</b>	65.6 ± 0.3
SVM (7)	<b>85.5 ± 0.4</b>	<b>71.7 ± 0.2</b>
Softmax (5)	82.9 ± 0.4	65.7 ± 0.5
Softmax (7)	<b>85.4 ± 0.4</b>	<b>72.6 ± 0.1</b>

表 7. SVM 與 Softmax 連接不同 layer 數目的分類結果

## 6. 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。

## 7. Reference

- [1] 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
- [2] <https://www.zybuluo.com/lutingting/note/459569>
- [3] [http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML\\_2017/Lecture/auto.pdf](http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2017/Lecture/auto.pdf)