

Class Activation Map Easier and Easier

C.E 18011573 이장후 / Sejong Artificial Intelligence : S team



Topics

1. Background
2. Class Activation Map
3. My interest



Background

기본적인 내용과 Convolution Neural Network 에 대한
Visualization Work 들을 간단히 짚고 넘어갑시다.



Filter & Activation Map

Filter 의 정의

촬영 시 화면상에서 원하지 않는 부분을 제거하기 위해 사용하는 장치.

- 녹음 시 특정한 주파수의 소리를 줄이고 나머지 소리를 높일 때 사용하는 장치.
- 들어오는 빛의 성질을 변화시켜 영상 효과를 주는 카메라 장비 ex. Bayer Filter

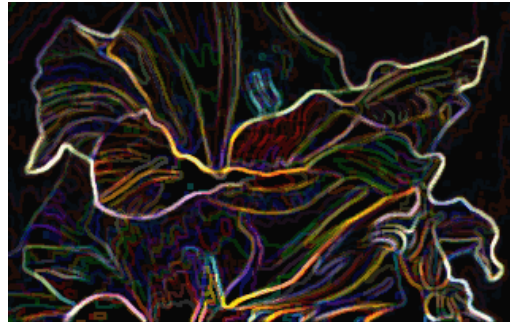
간단히 말해서, Image 에서 **Filter** 은 **이미지의 특징을 추출하는 역할**
추출된 이미지의 특징은 **Filter** 에 대한 **Activation Map** 이라고 함.



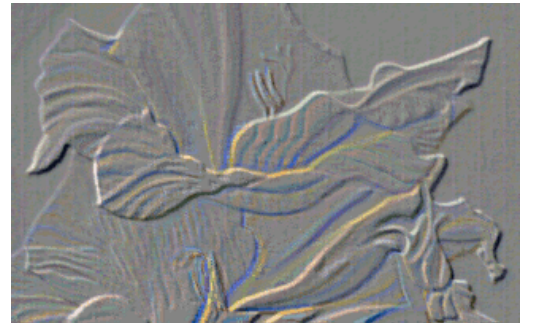
Original Image



Gaussian Blur Filter



Find Edges filter



Glowing Edges filter

<https://www.pcmag.com/encyclopedia/term/44794/image-filter>

Filter & Activation Map in ConvNet

Convolutional Neural Network 이 영상처리에서 가지는 가장 큰 의미는?

“사람이 수학적으로 만든 filter 을 쓰지 말고, 데이터를 바탕으로 특징 추출을 자동적으로 하도록 만드는 것”

-1	0	+1
-2	0	+2
-1	0	+1

x filter

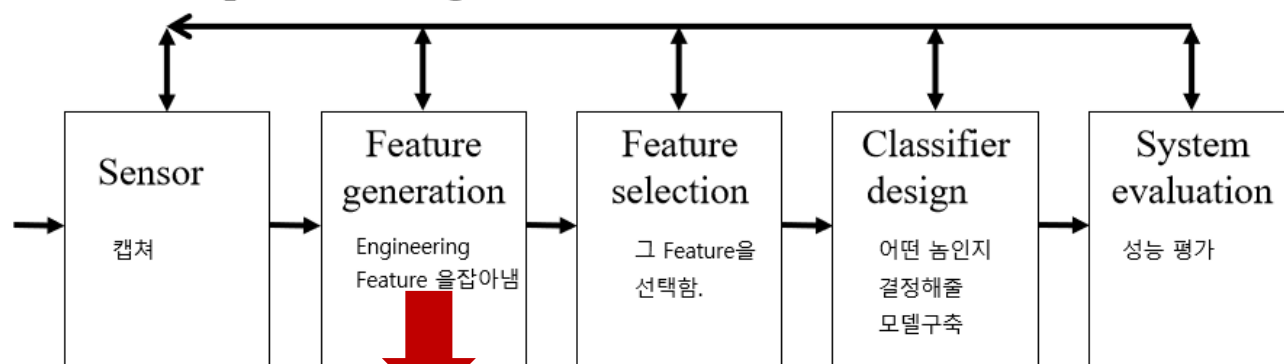
+1	+2	+1
0	0	0
-1	-2	-1

y filter

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A}$$



- Basic pattern recognition flowchart 전통적인 방법



(class)	area	height	width	number	number	(cx,cy)	best	least
character				#holes	#strokes	center	axis	inertia
'A'	medium	high	3/4	1	3	1/2,2/3	90	medium
'B'	medium	high	3/4	2	1	1/3,1/2	90	large
'8'	medium	high	2/3	2	0	1/2,1/2	90	medium
'0'	medium	high	2/3	1	0	1/2,1/2	90	large
'1'	low	high	1/4	0	1	1/2,1/2	90	low
'W'	high	high	1	0	4	1/2,2/3	90	large
'X'	high	high	3/4	0	2	1/2,1/2	?	large
'*	medium	low	1/2	0	0	1/2,1/2	?	large
'_'	low	low	2/3	0	1	1/2,1/2	0	low
'/'	low	high	2/3	0	1	1/2,1/2	60	low

Table 4.1: Example features for a sample character set.

Sejong Univ. DMS Lab.

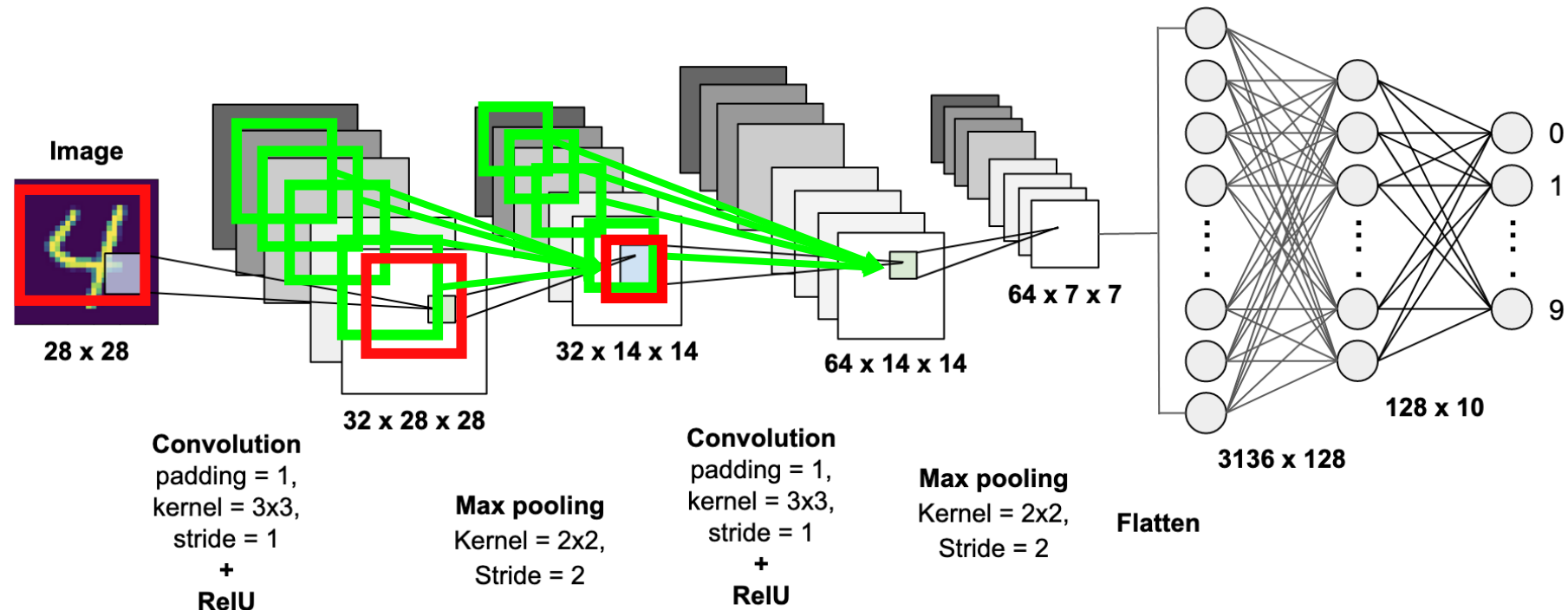
https://en.wikipedia.org/wiki/Sobel_operator



Filter & Activation Map in ConvNet

Convolutional Neural Network 의 가장 마지막 Layer 쪽으로 갈수록, 점점 더 “추상화된” 특징을 추출한다.

- 뒤쪽 layer 일수록, 원본 이미지의 ‘더 넓은 부분을’ 보고 있음.
- ‘더 많은 Channel’ 의 특성 중에서 선택한 것이기 때문. 즉, Activation Map 들을 여러 번 Filtering



How to Visualize ConvNet?

Convolutional Neural Network 을 Visualization 하려는 시도들은 크게 아래와 같이 구분할 수 있음.
발전은, CNN 자체를 설명하기 위한 방향에서, **특정 input 에 대한 원인 해석 (Attribution)** 을 하는 방향으로.



Activation Map 과 관련

- [CNN 자체를 설명하기 위함]
- Raw Activation Visualization
 - Activation with Gradient Ascent

- [모델의 예측 결과를 설명하기 위함]
- **Class Activation Map (CAM)**
 - Grad-CAM



Filter 의 가중치와 관련

- [CNN 자체를 설명하기 위함]
- Raw Filter Visualization



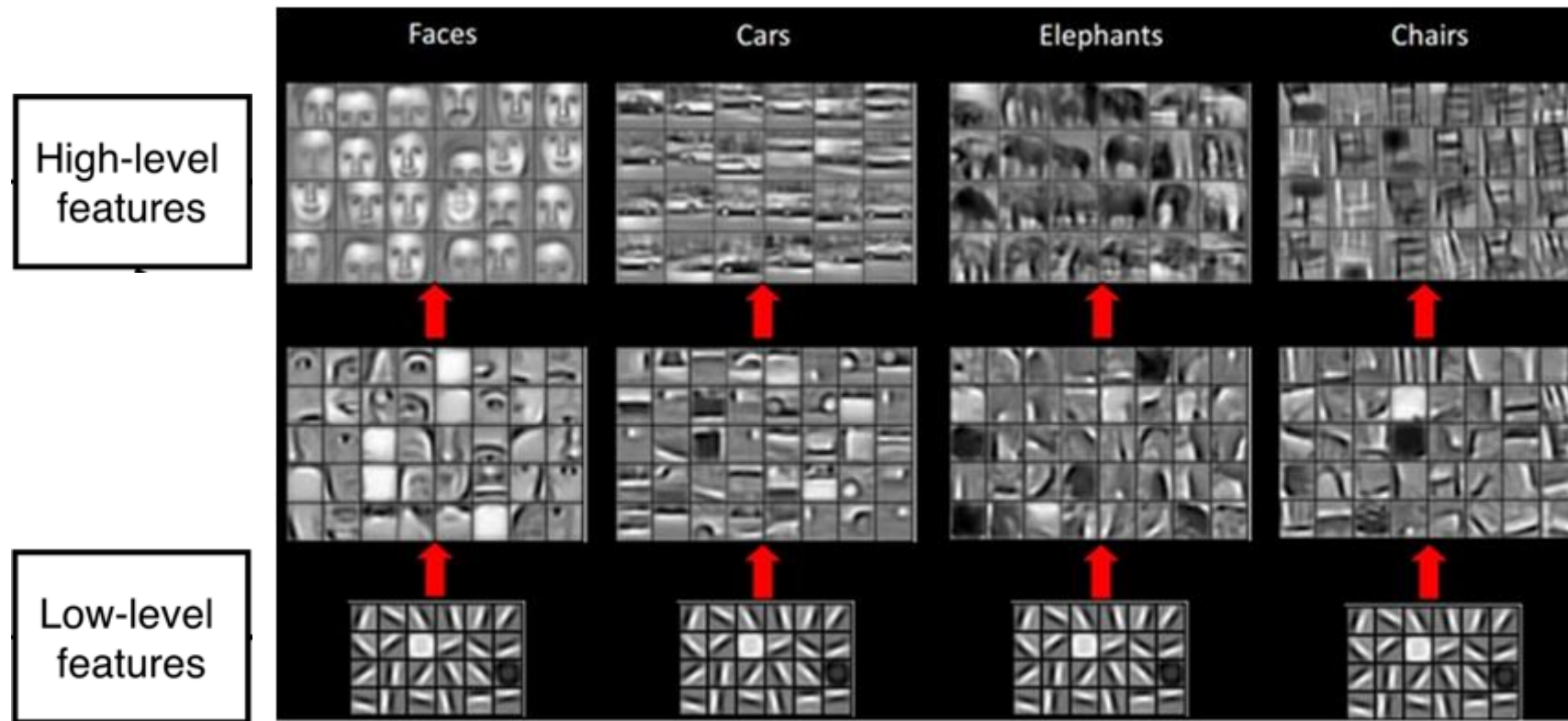
Other Methods

- [CNN 자체를 설명하기 위함]
- F.C Layer Visualization
 - Saliency Map

- [모델의 예측 결과를 설명하기 위함]
- Deconvnet, 이미지 일부분 가리기

Visualize : Raw Activation in ConvNet

- 추상화되어 간다는 것은 확인 가능
- 하지만, 이 그림처럼 사람이 이해할 수 있는 방향으로 추상화된다는 보장은 없음



Visualize : Raw Activation in ConvNet

- 추상화한 모델 : VGG16.
- 추상화 결과를 확인하기 어려운 대표적인 예
- Low Level Feature 이 명확히 “특정 방향의 edge” 가 검출되었다는 것을 알 수 있음

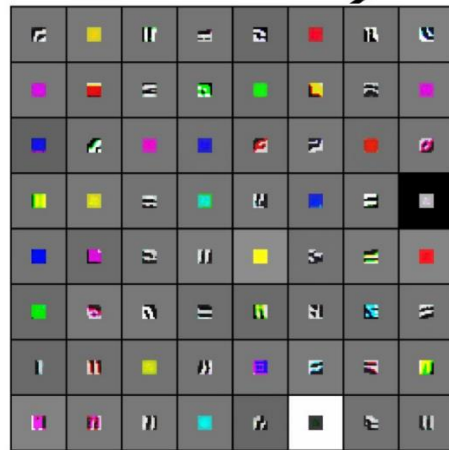
Preview

[Zeiler and Fergus 2013]

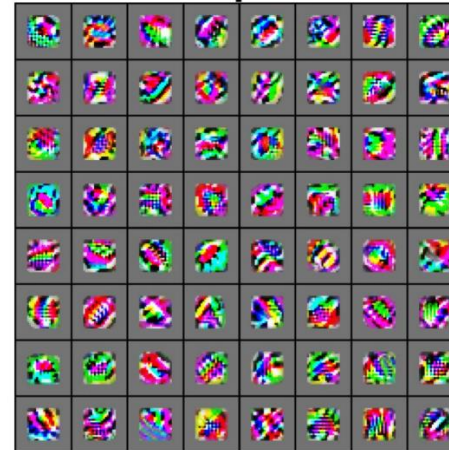
Visualization of VGG-16 by Lane McIntosh. VGG-16 architecture from [Simonyan and Zisserman 2014].



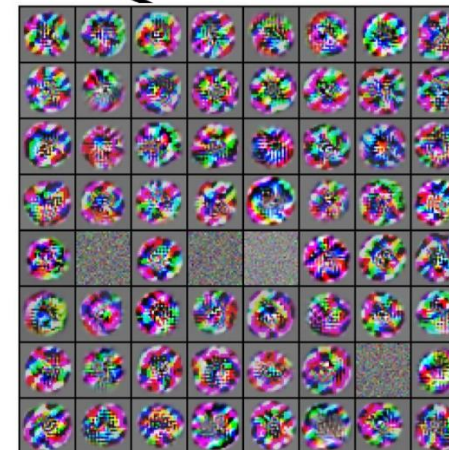
Filter 의 개수



VGG-16 Conv1_1



VGG-16 Conv3_2

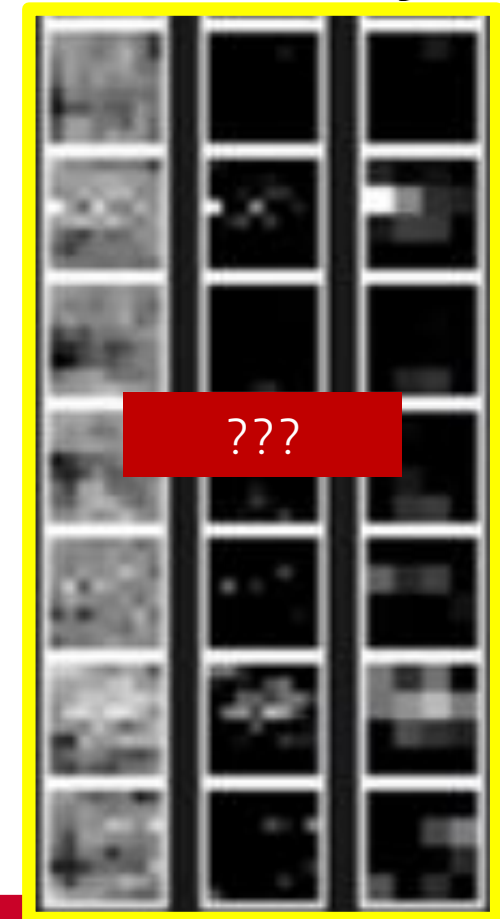
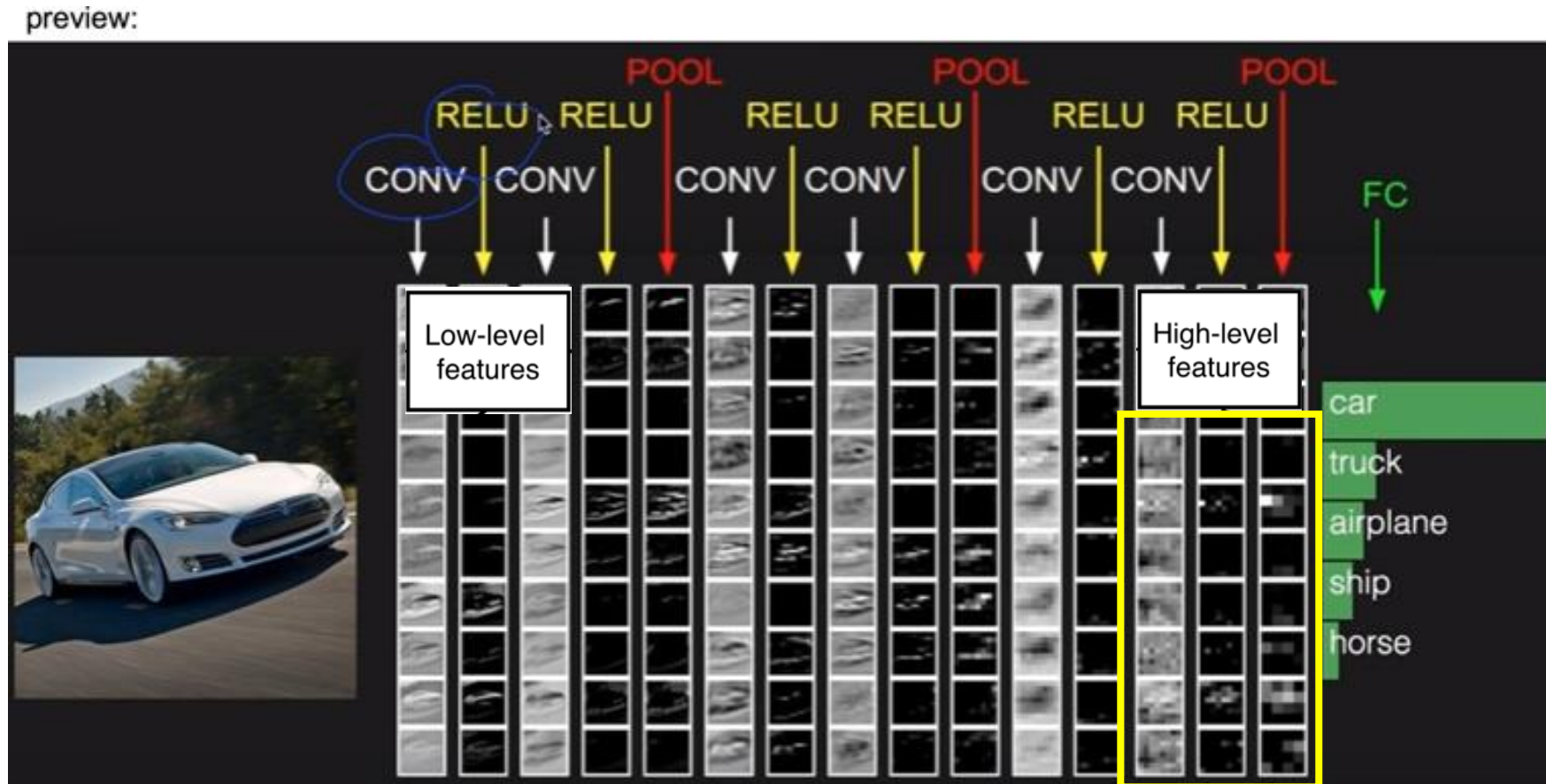


VGG-16 Conv5_3

Visualize : Raw Activation in ConvNet

Activati
on Map
과 관련

- 추상화 결과를 확인하기 어려운 대표적인 예 2
- Low Level 에서 명확히 “Edge 중심으로” 가 검출되었다는 것만 알 수 있음



Visualize : Raw Filter in ConvNet

단순히 Activation Map 을 Visualization 하는 방법은 지나친 추상화의 문제와, input 이 필요하다는 문제. Filter 의 가중치를 보고 이 Filter 이 어떤 역할을 하는 Filter 인지 알아낼 수 있다는 생각을 이용.

아래 그림의 Filter 은 무엇을 하는 Filter 일까?






-1	0	+1
-2	0	+2
-1	0	+1

x filter

+1	+2	+1
0	0	0
-1	-2	-1

y filter

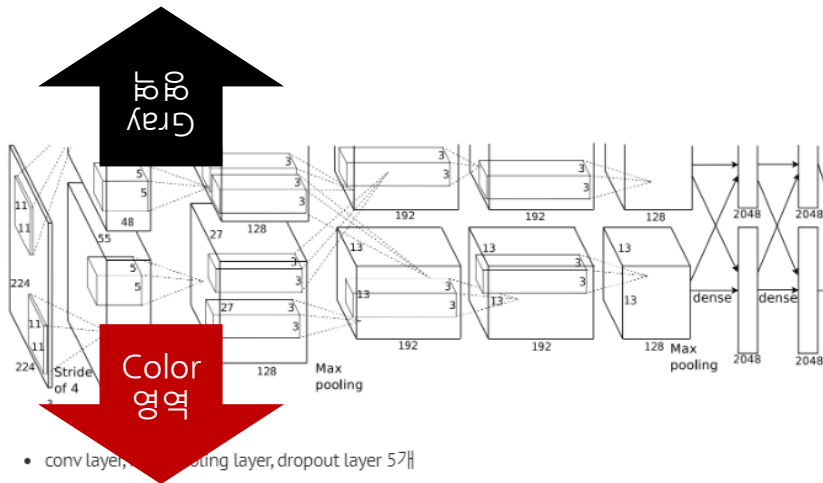
Answer : **Edge Detection Filter**

Operation	Kernel w	Image result $g(x,y)$
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	

Visualize : Raw Filter in ConvNet

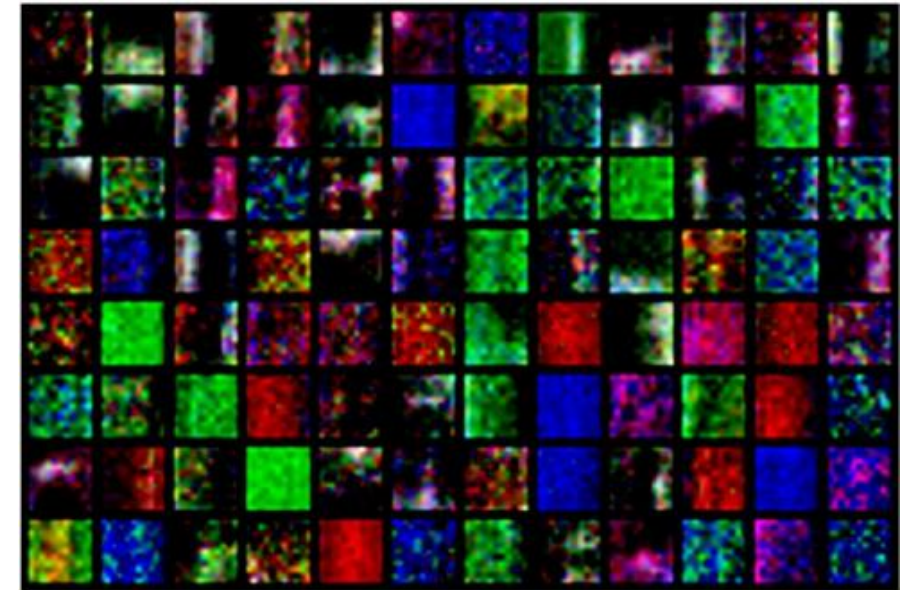
당연히 발생하는 문제점

- Filter 의 생김새만 보고, 어떤 기능을 하는 Filter 인지 직관적으로 파악하기 어렵다. (그리고 일단 너무 작다)
- 레이어가 쌓임 : 이미 한 번 특징을 뽑은 것에서 다시 특징을 뽑는 필터를 보는 것 : 의미파악이 불가능.
- Activation Map 에 비해 Filter 을 Visualization 하기에는 양이 너무 많다. (ex.1 Map 당 512 channel filter)
- ConvNet 에서 사람이 이해하기 쉬운 모양의 Filter 만 생겨나라는 법은 없다.



- conv layer, pooling layer, dropout layer 57%
- fully connected layer 37%
- nonlinearity function : ReLU
- batch stochastic gradient descent

Visualization of 96 filters of the first convolution layer. The filters of size $11 \times 11 \times 3$ are learned on the $227 \times 227 \times 3$ input images.



<https://youtu.be/6wcs6szJWMY>, https://www.researchgate.net/figure/Visualization-of-96-filters-of-the-first-convolution-layer-The-filters-of-size-11-11_fig2_279215570



Visualize : Raw Filter in ConvNet

결국 Filter 을 Visualization 하는 방법은, ConvNet 이 동작을 잘 하고 있긴 한건가 확인하는 데 사용.

Visualize the
filters/kernels
(raw weights)

We can visualize
filters at higher
layers, but not
that interesting

(these are taken
from ConvNetJS
CIFAR-10
demo)

Weights:


이건 뭔지 알겠다.

layer 1 weights

$16 \times 3 \times 7 \times 7$

Weights:


layer 2 weights

$20 \times 16 \times 7 \times 7$

Weights:


layer 3 weights

$20 \times 20 \times 7 \times 7$

<https://youtu.be/6wcs6szJWM>



Visualize : Gradient Ascent with Activation (1)

- Gradient Ascent : 특정 Activation Map 까지 경사 상승을 하는 경우.
- $\frac{d(Activation_{layer\ L, channel\ k})}{d(Imagepixel_{ijc})}$ 를 계산, $imagepixel_{ijc} + r \frac{d(Activation_{layer\ L, channel\ k})}{d(Imagepixel_{ijc})}$ 를 해 줌.
- 위 과정을 반복적, input image 를 갱신해 나감. $imagepixel = imagepixel_{ijc} + r \frac{d(Activation_{layer\ L, channel\ k})}{d(Imagepixel_{ijc})}$
- 그렇게 생성된 최종 input image 는 아래와 같은 모양들을 가지게 됨.

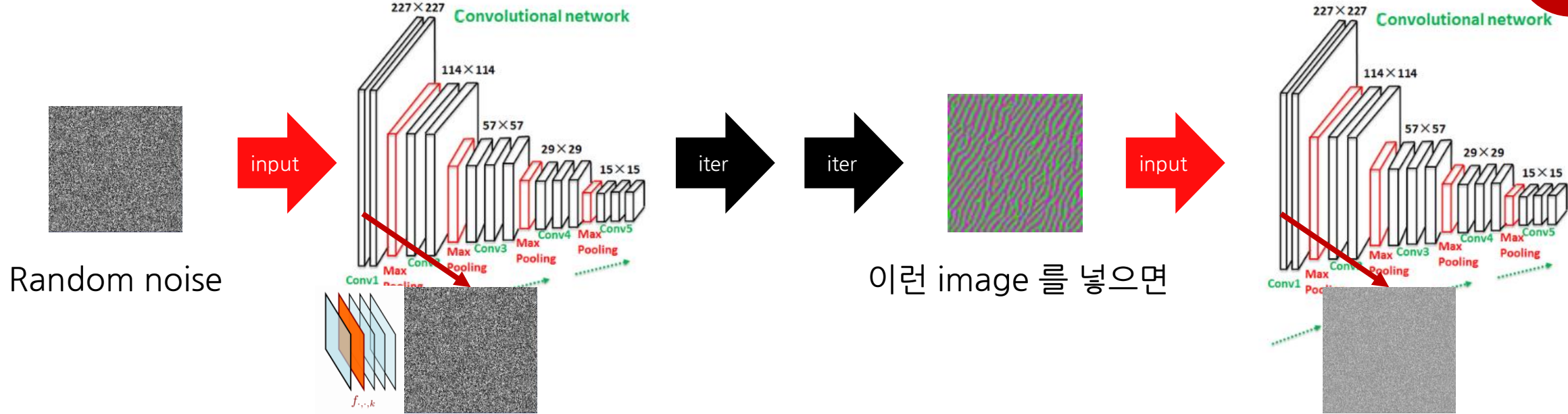
How neural networks build up their understanding of images



<https://distill.pub/2017/feature-visualization/>

Visualize : Gradient Ascent with Activation (2)

Activati
on Map
과 관련



Ex. Conv1 block, 1st layer, 2nd channel feature map : Target

이 Activation map 의 각 pixel 을 가장 많이 활성화시키는
Random noise 를 만드는 것이 이 Visualization 의 목적.

이 Conv1 block, 1st layer, 2nd channel feature map 을 만드는 filter 들
과 “아다리”가 잘 맞으면 Activation map 은 활성화 될 것이다.

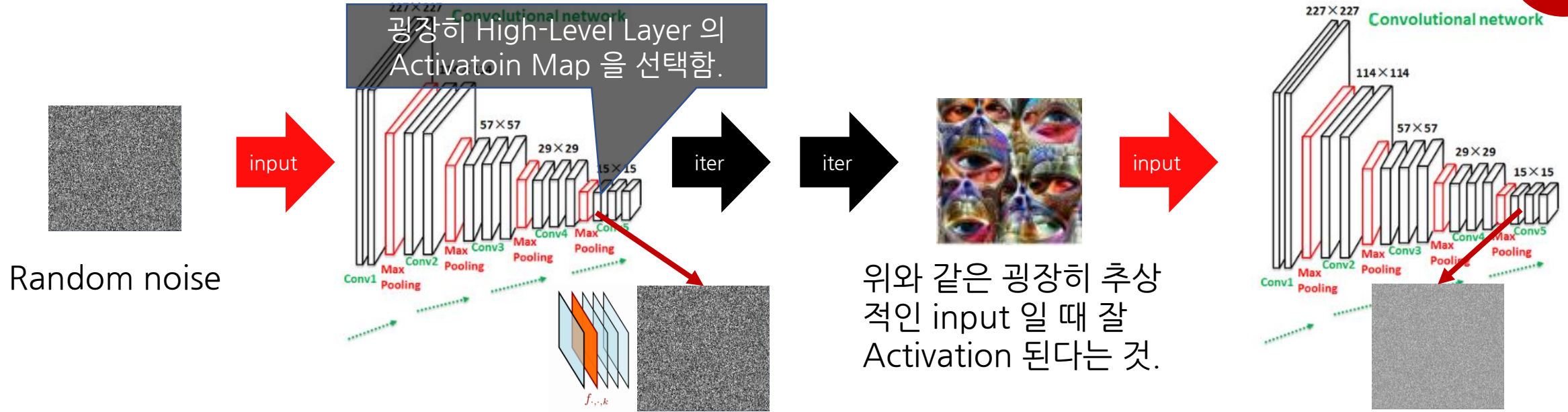
이런 Activation 이 나옴.
Random noise 에서 활성화된 것보
다 많이 활성화된 것을 볼 수 있음.

https://www.researchgate.net/figure/The-CNN-DCN-architecture-for-Conv5-of-VGG16_fig6_315769531



Visualize : Gradient Ascent with Activation (3)

Activati
on Map
과 관련



Ex. Conv4 block, 1st layer, 2nd channel feature map : Target

이 Activation map 의 각 pixel 을 가장 많이 활성화시키는 Random noise 를 만드는 것이 이 Visualization 의 목적.
이 Conv4 block, 1st layer, 2nd channel feature map 을 만드는 filter 들과 “아다리”가 잘 맞으면 Activation map 은 활성화 될 것이다.

이런 Activation 이 나옴.
Random noise 에서 활성화된 것보다 많이 활성화된 것을 볼 수 있음.

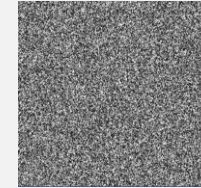
https://www.researchgate.net/figure/The-CNN-DCN-architecture-for-Conv5-of-VGG16_fig6_315769531



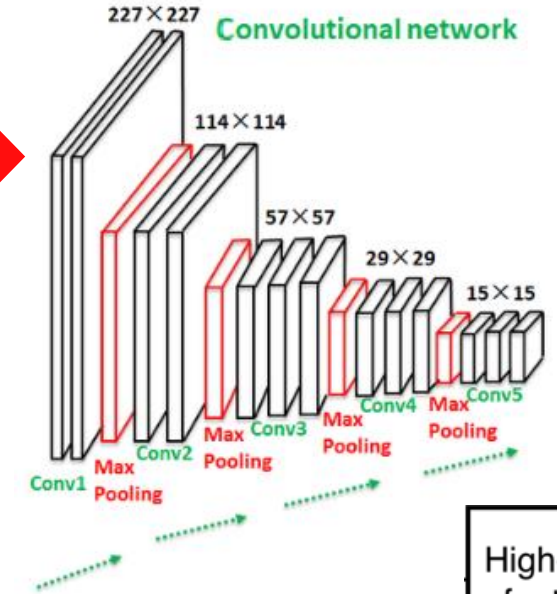
Visualize : Gradient Ascent with Activation (3)

Activati
on Map
과 관련

Random noise
gonna be...

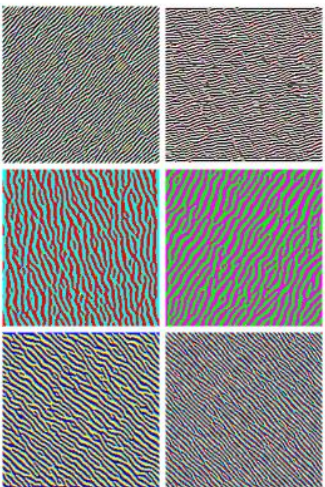


input



feature map : Target
이 Low Level 일 때

Random noise 는 이렇게 됨



Edges (layer conv2d0)



Textures (layer mixed3a)



Patterns (layer mixed4a)



Parts (layers mixed4b & mixed4c)



Objects (layers mixed4d & mixed4e)

feature map : Target
이 High Level 일 때

Random noise 는 이렇게 됨

Low-level
features

High-level
features

https://www.researchgate.net/figure/The-CNN-DCN-architecture-for-Conv5-of-VGG16_fig6_315769531



CAM

Learning Deep Features for Discriminative Localization
Class Activation Map, Bolei Zhou et al. CVPR 2016



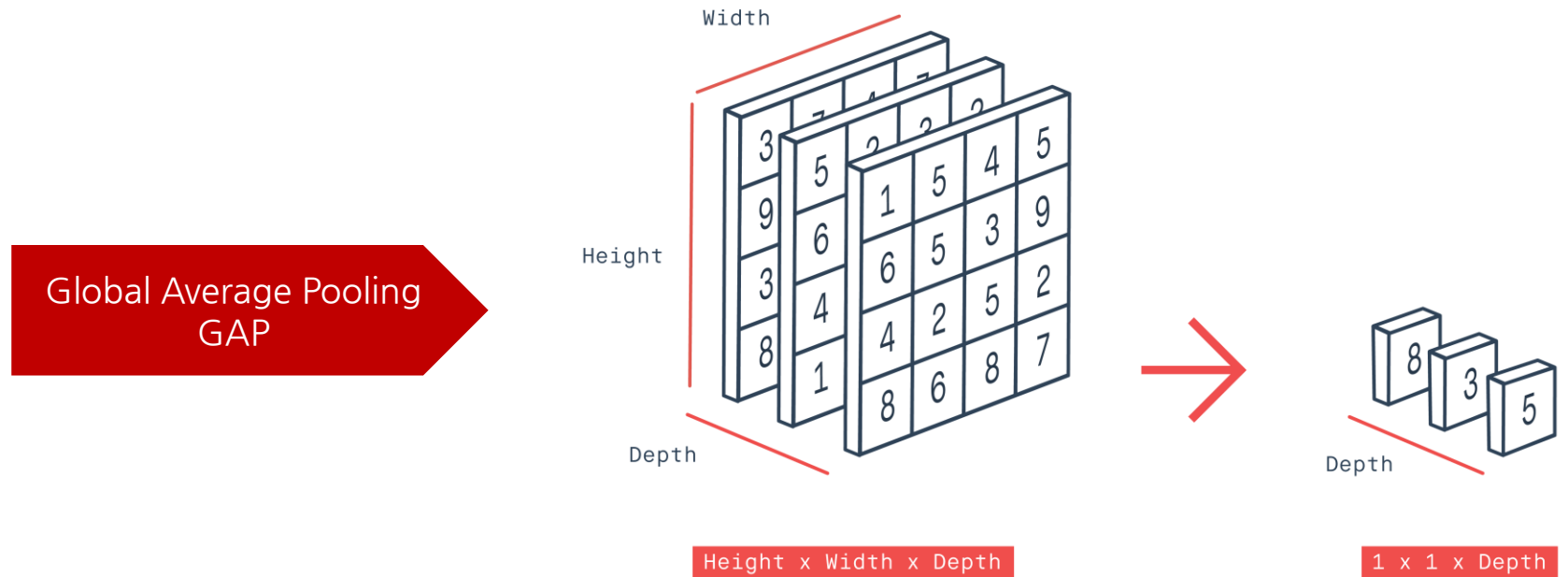
Abstraction

Global Average Pooling 이 CNN 의 Localization 에 미치는 영향에 대한 탐구.

논문 제목 : Learning Deep Features for Discriminative Localization

저자 : Bolei Zhou et al.

출판 : CVPR, 2016



Conclusion

단순히 image 에 대한 label 만으로, (bounding box 에 대한 annotation 없이) 학습한 CNN 모델이 객체 탐 지 문제도 함께 수행할 수 있다는 것에 대한 가능성을 검증.

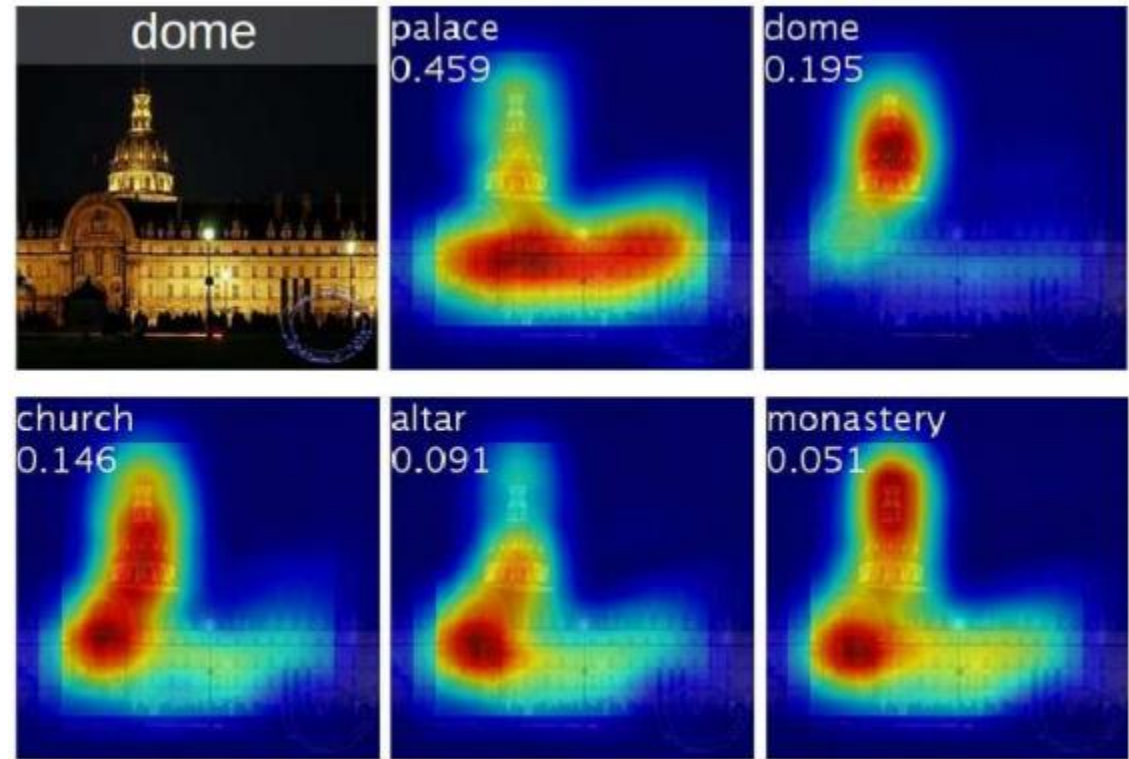
Heuristics : CAM 기반으로 Bbox 를 만드는 약간의 기교 추가 버전
Full supervision : 아예 Bbox 정보를 이용해 학습시킨 경우

Table 3. Localization error on the ILSVRC test set for various weakly- and fully- supervised methods.

Method	supervision	top-5 test error
GoogLeNet-GAP (heuristics)	weakly	37.1
GoogLeNet-GAP	weakly	42.9
Backprop [23]	weakly	46.4
GoogLeNet [25]	full	26.7
OverFeat [22]	full	29.9
AlexNet [25]	full	34.2

정확히 어떻게 Bbox 정보를 주고 비교했는지는 나와있지 않음.

왜 CNN 이 그렇게 판단을 했는지 클래스별로 아주 쉽게 highlighting 하는 방법 : CAM (Class Activation Map) 을 제시함.



Introduction : Relative Work

- Bounding Box 에 대한 정보를 아예 주지 않고, (이미지,라벨) 쌍만 이용해서 학습을 시킨 Convolutional Neural Network. 이러한 모델도 Object Detection 문제 (Localization 문제) 에 사용할 수 있다는 것을 Fully Connected Layer 이 아닌 Max Pooling Layer 을 이용해서 보인 논문 (is object localization for free? CVPR 2015)
- 하지만, Max Pooling 은 Map 에 여러 물체가 있어도 가장 큰 값 딱 하나에 맞추어지는 반면, 이 논문에서 강조하는 Average Pooling 은 Map 전체를 골고루 관찰하는 경향이 강하기 때문에, Localization 문제를 풀기 위해서는, GMP 보다는 GAP 가 더 타당하다고 주장.
- Global Average Pooling 에 대해서는 완전히 처음 제시되는 것이 아님. (Network In Network, ICLR 2014) Fully Connected Layer 을 대체한다는 관점은 NIN 논문에서 처음 등장함.
- 하지만 해당 논문에서는 그냥 Average Pooling 을 사용하면 Overfitting 을 피할 수 있다는 관점만 가져가지만, 이 논문에서는 Fully Connected 대신 삽입된 GAP 가 Localization 에 영향을 미치는지를 실험함.

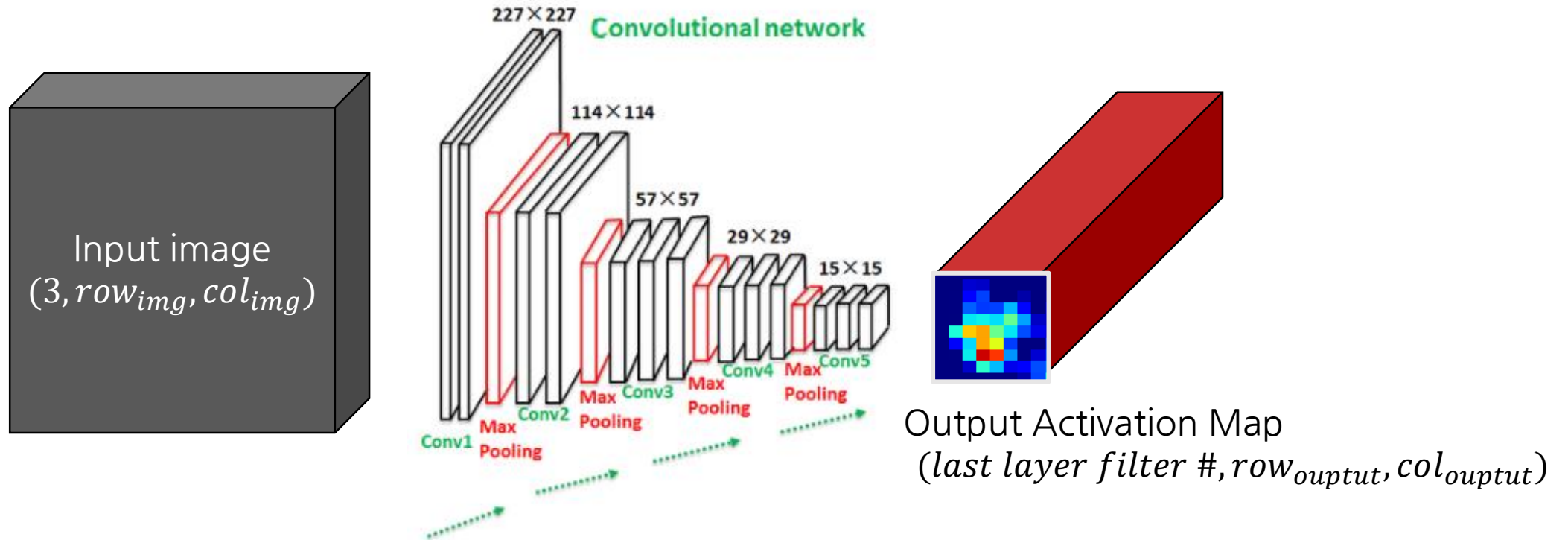
참고 : 논문에서의 표현

- 물체 인식 문제 : Localize, Localization Problem
- Bounding Box 등의 정보가 없는 단순 (이미지, 라벨) 쌍 : image-level labels



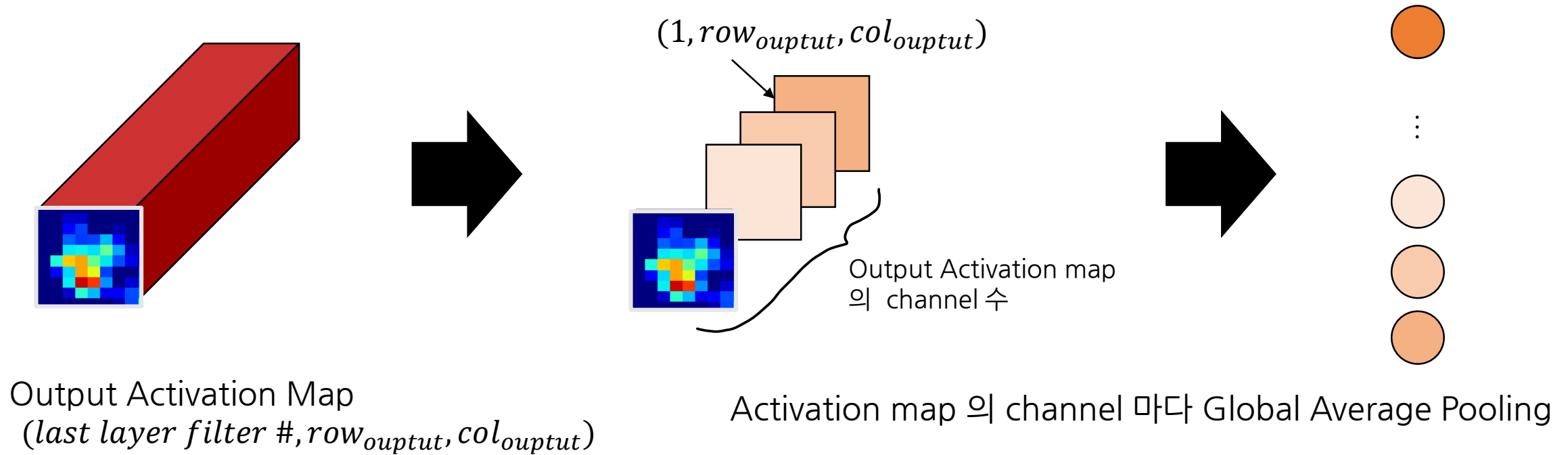
How It Works

우리가 알고 있는 일반적인 ConvNet 의 앞부분 그림.



How It Works

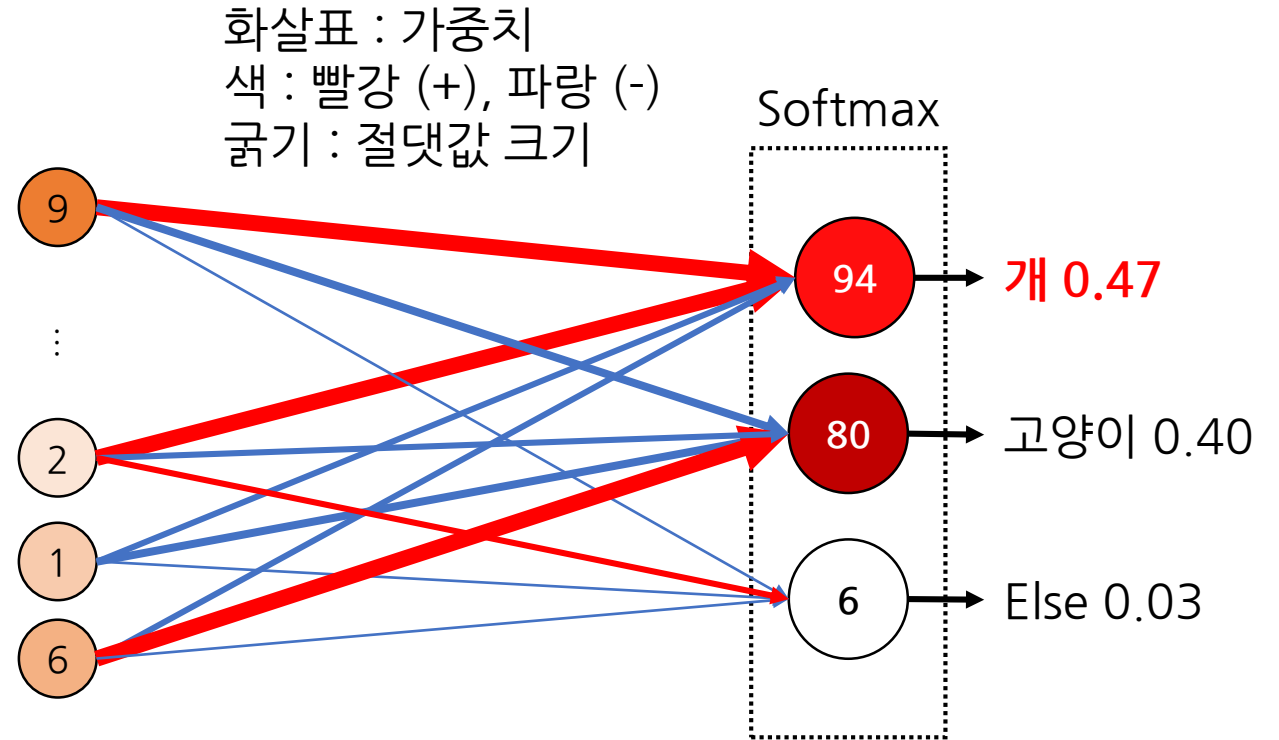
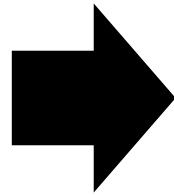
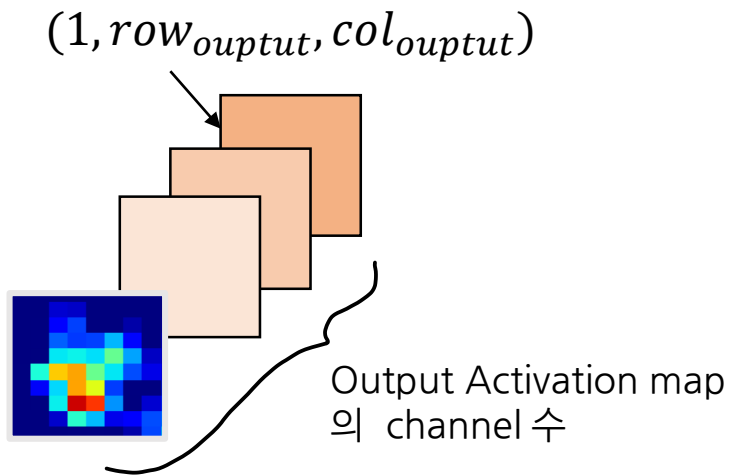
원래 보통 Localization 문제를 풀거나, 일반적인 Classification 문제를 풀 때 이렇게 나온 Output Activation Map 을 Flatten 해서 FC Layer 로 만든 뒤, FC Layer 을 여러 겹 쌓아 Bounding Box 를 Regression 하거나 Classify 했지만...



How It Works

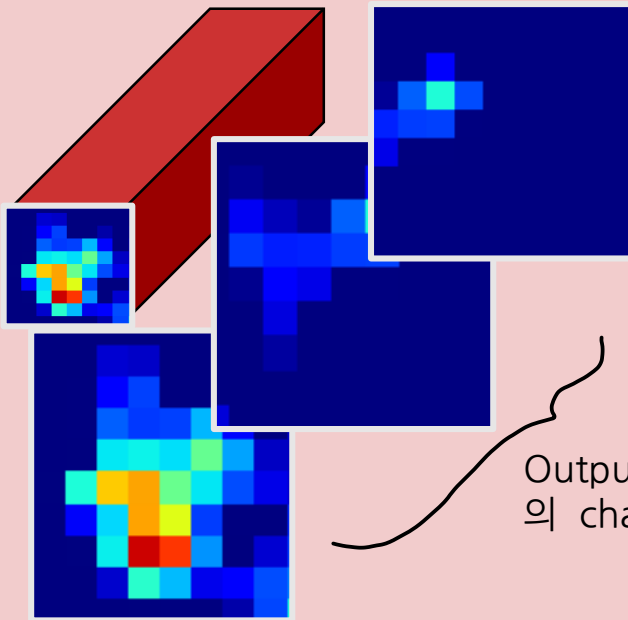
단지 Activation map 의 GAP 값을 이용해서, output layer 에 연결하고 Classification 을 수행함

param : # channel * # class



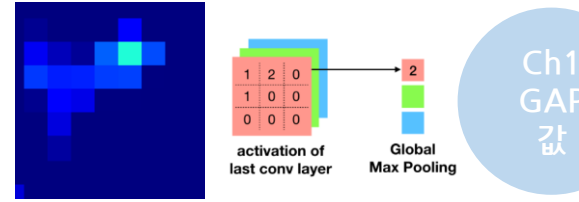
Why It Works

Output Activation Map
(last layer filter #, row_{output} , col_{output})

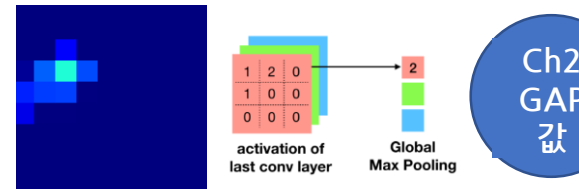


Output Activation map
의 channel 수

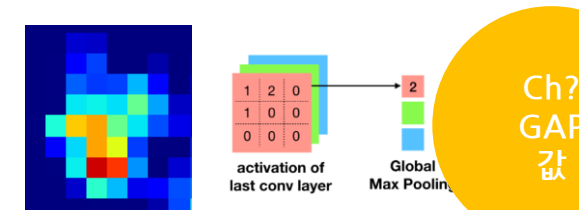
<https://www.saagie.com/blog/object-detection-part2/>



이 channel 은 결과적으로
Input Image 에서 강아지 꼬리를
activation 하는 경향이 강함

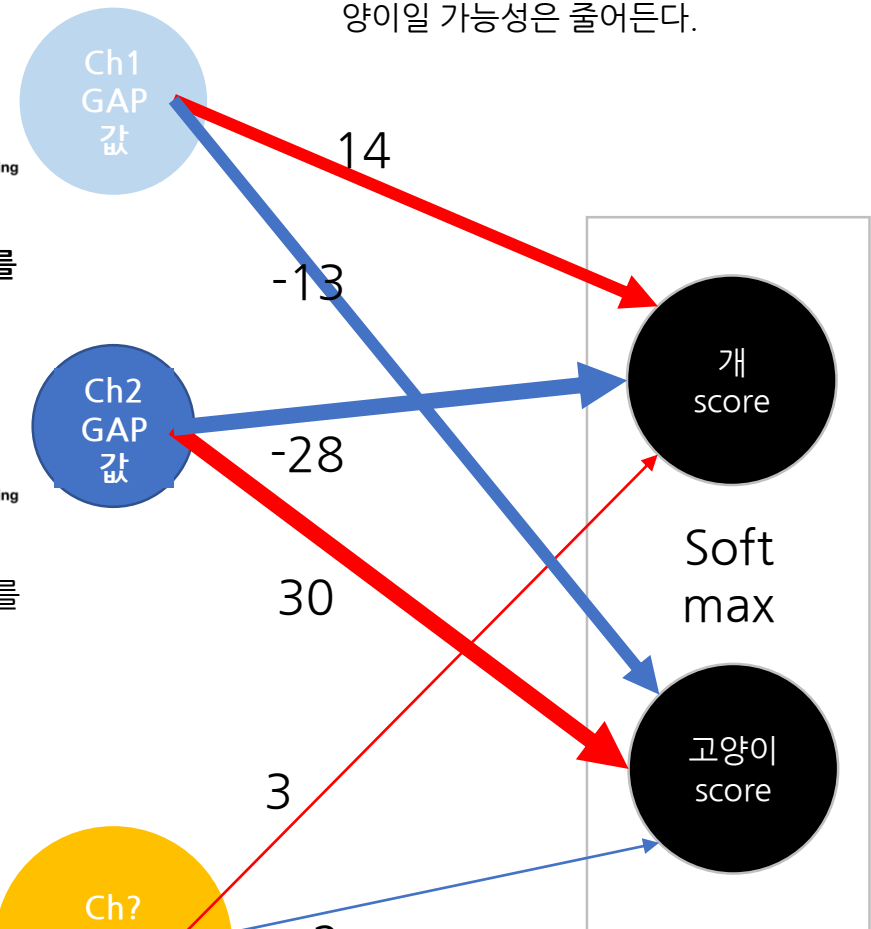


이 channel 은 결과적으로
Input Image 에서 고양이 꼬리를
activation 하는 경향이 강함.



이 channel 은 결과적으로
Input Image 에서 커다란 강아지 몸통을
activation 하는 경향이 강함.

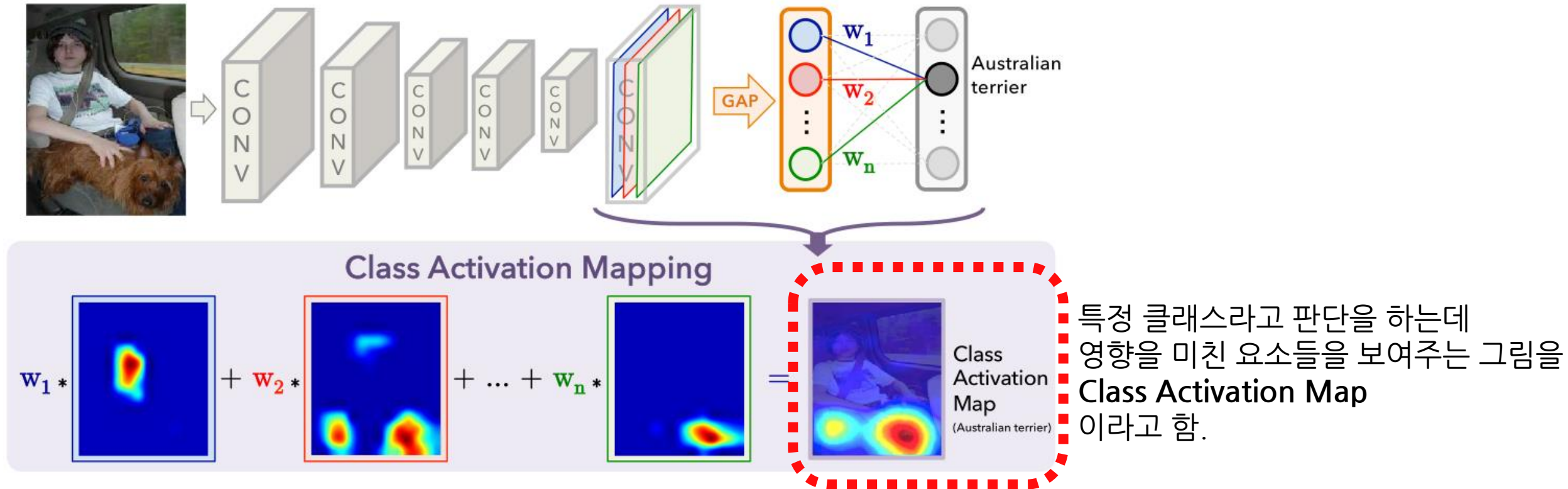
해석 : 강아지 꼬리가 감지된다면 (그 이미지를 넣
었을 때 activation map 의 GAP 값이 높다면) 고
양이일 가능성은 줄어든다.



해석 : Activation 되는 영역이 작더라도, 고양이의 꼬
리와 같은 확실한 근거라면 결과에 큰 영향을 미쳐야
하고, Activation 되는 영역이 커서 GAP 값이 크다는
이유만으로 다른 요인들이 무시되면 안 된다.

How It Works

정리하면, Final Layer 의 Activation Map 에 대한 GAP 값이 있을 때
그 GAP 값과 특정 클래스(ex.강아지) 와 연결된 가중치는
해당 Activation Map 이 특정 클래스에 미치는 전반적인 영향과 중요도를 나타냄.



Why It Works : with math (1)

- 왜 (가중치) = (특정 Activation Map 의 GAP 가 특정 클래스에 미치는 영향) 이라고 할 수 있는지
- 왜 (Class Activation Map) = (가중치 * (특정 Activation Map)) 이라고 할 수 있는지
특정 클래스라고 판단을 하는데 영향을 미친 요소들을 보여주는 그림을 Class Activation Map 이라고 함.

$$F_k = \sum_{x,y} f_k(x,y)$$

small f_k 는 k 번째 activation map. (x,y) 는 해당 activation map의 pixel
Large F_k 는 GAP.
평균인데 N 으로 나누지 않은 건 그냥 논문 표기 따르기 위함.

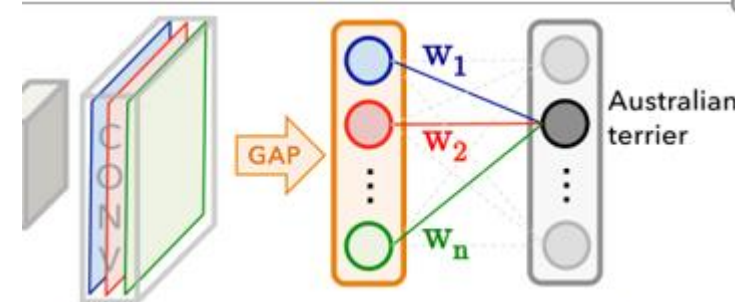
$$S_c, \text{ is } \sum_k w_k^c F_k$$

S_c 는 class c 에 해당하는 softmax layer 에 들어가는 값
 W 는 당연히 가중치인데, k 번째 activation map \rightarrow class c 로 가는 가중치
예를 들어, 3번째 class activation map 의 GAP 값과 강아지 클래스를 연결하는 가중치
즉, 특정 클래스 c 를 만드는 것에 k 번째 GAP 가 미치는 영향을 w 라고 정의함.

w_k^c indicates the *importance* of F_k for class c .

참고 : Softmax function

$$\frac{\exp(S_c)}{\sum_c \exp(S_c)}$$



Why It Works : with math (2)

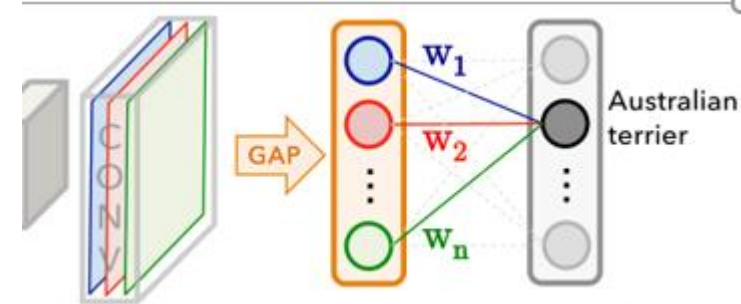
- 왜 (가중치) = (특정 Activation Map 의 GAP 가 특정 클래스에 미치는 영향) 이라고 할 수 있는지
- 왜 (Class Activation Map) = (가중치 * (특정 Activation Map)) 이라고 할 수 있는지

$$F_k = \sum_{x,y} f_k(x,y) \quad S_c, \text{ is } \sum_k w_k^c F_k$$

w_k^c indicates the *importance* of F_k for class c .

위 두 식을 이용하고, sigma 의 특성을 생각하면

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x,y)$$



Why It Works : with math (3)

- 왜 (가중치) = (특정 Activation Map 의 GAP 가 특정 클래스에 미치는 영향) 이라고 할 수 있는지
- 왜 (Class Activation Map) = (가중치 * (특정 Activation Map)) 이라고 할 수 있는지

$$F_k = \sum_{x,y} f_k(x,y) \quad S_c, \text{ is } \sum_k w_k^c F_k$$

w_k^c indicates the *importance* of F_k for class c .

위 두 식을 이용하고, sigma 의 특성을 생각하면

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x,y) = \sum_{x,y} \sum_k w_k^c f_k(x,y)$$

Sigma 의 특성을 또다시 생각하면



How It Works : Classification

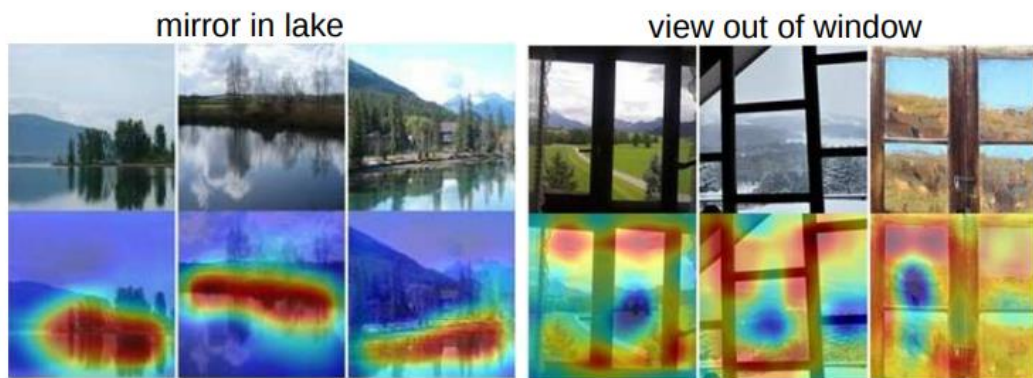


Figure 10. Informative regions for the concept learned from weakly labeled images. Despite being fairly abstract, the concepts are adequately localized by our GoogLeNet-GAP network.

Table 1. Classification error on the ILSVRC validation set.

Networks	top-1 val. error	top-5 val. error
VGGnet-GAP	33.4	12.2
GoogLeNet-GAP	35.0	13.2
AlexNet*-GAP	44.9	20.9
AlexNet-GAP	51.1	26.3
GoogLeNet	31.9	11.3
VGGnet	31.2	11.4
AlexNet	42.6	19.5
NIN	41.9	19.6
GoogLeNet-GMP	35.6	13.9

Why It Works : with math (4)

- 왜 (가중치) = (특정 Activation Map 의 GAP 가 특정 클래스에 미치는 영향) 이라고 할 수 있는지
- 왜 (Class Activation Map) = (가중치 * (특정 Activation Map)) 이라고 할 수 있는지

w_k^c indicates the *importance* of F_k for class c .

$$F_k = \sum_{x,y} f_k(x,y)$$

small f_k 는 k 번째 activation map. (x,y) 는 해당 activation map의 pixel
Large F_k 는 GAP.
평균인데 N 으로 나누지 않은 건 그냥 논문 표기 따르기 위함.

$$M_c(x,y) = \sum_k w_k^c f_k(x,y)$$

위 식을 고려할 때
 M_c 는 특정 클래스 c (ex. 강아지 클래스) 의
Class Activation Map 이라고 정의 가능.

Why It Works : with math (5)

- 왜 (가중치) = (특정 Activation Map 의 GAP 가 특정 클래스에 미치는 영향) 이라고 할 수 있는지
- 왜 (Class Activation Map) = (가중치 * (특정 Activation Map)) 이라고 할 수 있는지

w_k^c indicates the *importance* of F_k for class c .

$$M_c(x, y) = \sum_k w_k^c f_k(x, y)$$

M_c 는 특정 클래스 c (ex. 강아지 클래스) 의 Class Activation Map 이라고 정의 가능.

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x, y) = \sum_{x,y} \sum_k w_k^c f_k(x, y).$$

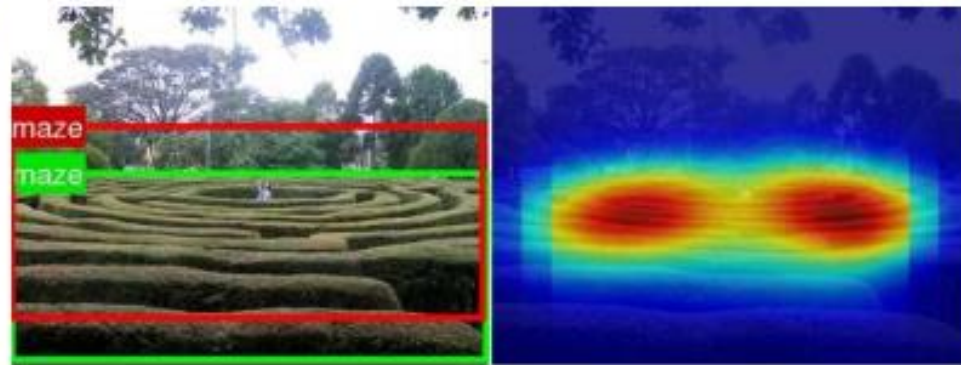
이 부분에 M_c 를 대입

$$S_c = \sum_{x,y} M_c(x, y)$$

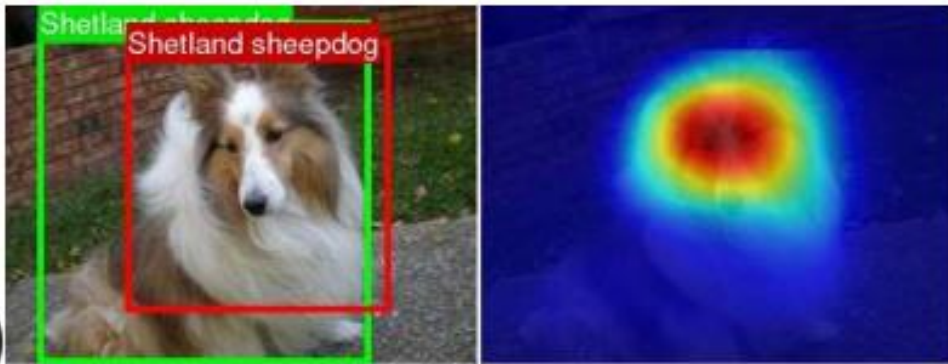
directly indicates the importance of the activation at spatial grid (x, y) leading to the classification of an image to class c . 즉, 왜 Class Activation Map 이 해당 Class 에 대한 중요도를 나타낼 수 있는가

How It Works : Localization

- 어느 부분이 Class Score 에 어떻게 영향을 미치는지 알 수 있다면, 그를 이용하여 Detection 을 수행할 수 있기 때문에, Class Activation Map 을 이용해서 Localization 을 시도하고, Bounding Box Label 과 성능을 비교.



Green : 예측
Red : 정답



a)

How It Works : Localization

- 약간의 성능 낙차가 있지만,
- 성능이 떨어져 보이는 것은 그냥 Bounding Box 특성일 가능성 “또한 용량을 엄청 줄이는 것에 비해 이정도 성능 하락은 감수할 만 하다”

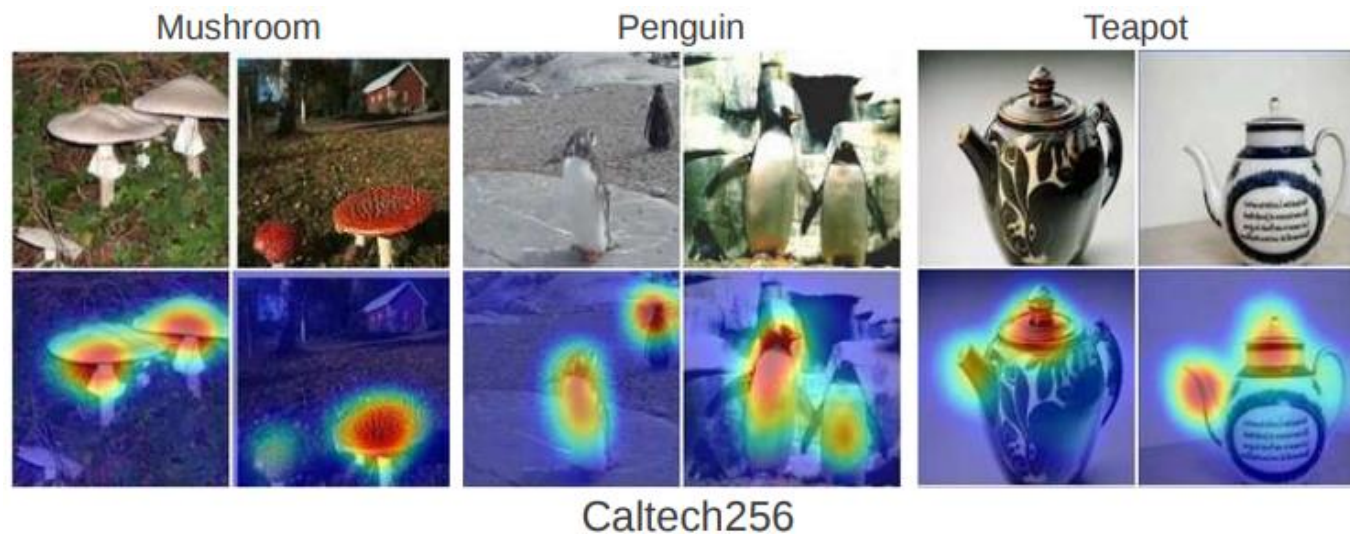
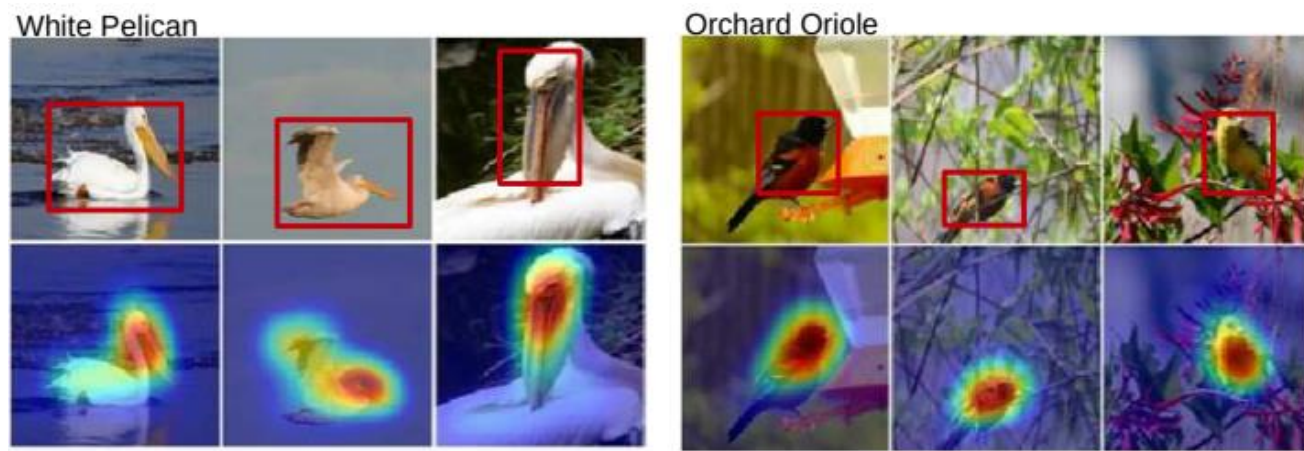


Table 3. Localization error on the ILSVRC test set for various weakly- and fully- supervised methods.

Method	supervision	top-5 test error
GoogLeNet-GAP (heuristics)	weakly	37.1
GoogLeNet-GAP	weakly	42.9
Backprop [23]	weakly	46.4
GoogLeNet [25]	full	26.7
OverFeat [22]	full	29.9
AlexNet [25]	full	34.2



Contribution

END-TO-END, SINGLE-FORWARD-PASS

- 방금 넣은 input 에 대해서 어떤 부분을 모델이 주목해서 보았는가를 하나의 모델에 구현했다는 것에 의의가 있음.
- 반복적인 forward-backward 작업이 필요가 없음
- 간단한 개조로 CNN 모델 전반에 쉽게 적용 가능함
- Localization 과 Classification 을 큰 성능 하락 없이 효과적으로 할 수 있음을 보여줌



Source Code

- <https://github.com/ProtossDragoon/CoMoLab/tree/master/CV/CAM>
- Keras implementation, tensorflow 2.0 backend (google COLAB env)



Source Code : def VGGCAM()

• 평범한 VGG-16 모델의 앞부분

VGG CAM Model

```
In [0]: def VGGCAM(nb_classes, num_input_channels=1024, input_tensor=None, input_shape=(224, 224, 3)):
```

```
    if input_tensor is None:
        img_input = layers.Input(shape=input_shape)
    else:
        img_input = input_tensor
```

```
    # Block 1
```

```
    x = layers.Conv2D(64, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block1_conv1')(img_input)
    x = layers.Conv2D(64, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block1_conv2')(x)
    x = layers.MaxPooling2D((2, 2), strides=(2, 2), name='block1_pool')(x)
```

```
    # Block 2
```

```
    x = layers.Conv2D(128, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block2_conv1')(x)
    x = layers.Conv2D(128, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block2_conv2')(x)
    x = layers.MaxPooling2D((2, 2), strides=(2, 2), name='block2_pool')(x)
```

```
    # Block 3
```

```
    x = layers.Conv2D(256, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block3_conv1')(x)
    x = layers.Conv2D(256, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block3_conv2')(x)
    x = layers.Conv2D(256, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block3_conv3')(x)
    x = layers.MaxPooling2D((2, 2), strides=(2, 2), name='block3_pool')(x)
```

```
    # Block 4
```

```
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block4_conv1')(x)
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block4_conv2')(x)
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block4_conv3')(x)
    x = layers.MaxPooling2D((2, 2), strides=(2, 2), name='block4_pool')(x)
```

```
    # Block 5
```

```
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block5_conv1')(x)
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block5_conv2')(x)
    x = layers.Conv2D(512, (3, 3),
                      activation='relu',
                      padding='same',
                      name='block5_conv3')(x)
```

Source Code : def VGGCAM()

VGG-16 모델의 Top (Fully Connect) 제거
해당 부분에 마지막 Conv 붙임

마지막 Conv 에 Average Pooling 2D

- Kernel size 를 해당 tensor 의 크기로 설정
- (물론 GlobalAveragePool2D써도됨)

그리고 마지막 output layer 로 연결되는
Fully Connect Layer

```
# Block 5
x = layers.Conv2D(512, (3, 3),
                  activation='relu',
                  padding='same',
                  name='block5_conv1')(x)
x = layers.Conv2D(512, (3, 3),
                  activation='relu',
                  padding='same',
                  name='block5_conv2')(x)
x = layers.Conv2D(512, (3, 3),
                  activation='relu',
                  padding='same',
                  name='block5_conv3')(x)

# Add another conv layer with ReLU + GAP
x = layers.Conv2D(num_input_channels, (3, 3),
                  activation='relu',
                  padding='same',
                  name='conv_cam')(x)
x = layers.AveragePooling2D((14, 14), name='avg_pool')(x)
x = Flatten(name='flatten')(x)

# Add the W layer
x = Dense(nb_classes, activation='softmax', name='fc_cam')(x)

# Ensure that the model takes into account
# any potential predecessors of 'input_tensor'.
if input_tensor is not None:
    inputs = keras_utils.get_source_inputs(input_tensor)
else:
    inputs = img_input

# Create model.
model = models.Model(inputs, x, name='vgg16')

return model
```

My Interest

Autonomous Vehicle



Autonomous Vehicle

- 올해 목표

30kg~50kg 정도 되는 모빌리티를 만들어서
자전거도로에서 자율주행 시키기 :
가을에 뚝섬유원지부터 반포 새빛섬까지 손 안 대고 가보기

다양한 하드웨어 위에서 인공지능 모델 돌려보기
- 다선님과 스터디



Autonomous Vehicle

차량의 제어를 기계에 넘기게 된다면,
이 차량이 어떤 시각 정보를 근거로 특정한 판단을 하게 되었는지
“디버깅” 할 필요가 있다고 생각했음.

오른쪽 그림을 보면
Teapot class 로 추정을 하는 것에는
Teapot 전체가 아니라, 주둥이 손잡이 뚜껑을 보고
Teapot 이라고 판단을 내렸음을 알 수 있음



Autonomous Vehicle

- NAVER LABS 2017
- 차선변경 판단자동화



인지 기술 통한 차선 변경 판단
알고리즘 발표

READ MORE >

네이버랩스, IV 2017서 인지 기술 통한 차선변경 판단 알고리즘 발표

2017.06.26 | 김동환 / PDX

자율주행



네이버랩스에서 고안한 자동차 주행환경 인지 알고리즘이 지난 6월 14일 국제 지능형 자동차공학 학회인 IEEE IV (intelligent vehicles symposium) 2017에서 논문 형태로 발표됐다. 정성균, 김지원, 김수정, 민재식(이상 네이버랩스) 공저인 'End-to-End Learning of Image based Lane-Change Decision'이다.

IV는 학계와 글로벌 기업을 아우르는 규모의 미래 자동차 기술 공학회다. 네이버랩스가 이번에 이곳에 발표한 논문은 딥러닝을 통해 주행중인 자동차의 안전한 차선 변경을 돕는 방법을 다루고 있다. 차량 외부에 달린 카메라에서 얻어진 측후방 영상을 통해 인접 차선이 옮겨갈 만한 상황인지를 순간적으로 판단하는 알고리즘이 핵심 내용이다. 자율주행차뿐만 아니라 사람이 운전하는 일반 차량에도 적용 가능하다.

실제로 해당 알고리즘은 현재 네이버랩스의 자율주행차량에도 적용되어 있다. 네이버랩스 Mobility팀의 정성균 연구원은 "사람이 차선변경을 실행하기 직전에 사이드미러로 측후방 상태를 보고 판단하는데서 영감을 얻어 해당 알고리즘을 개발했다"고 말했다.

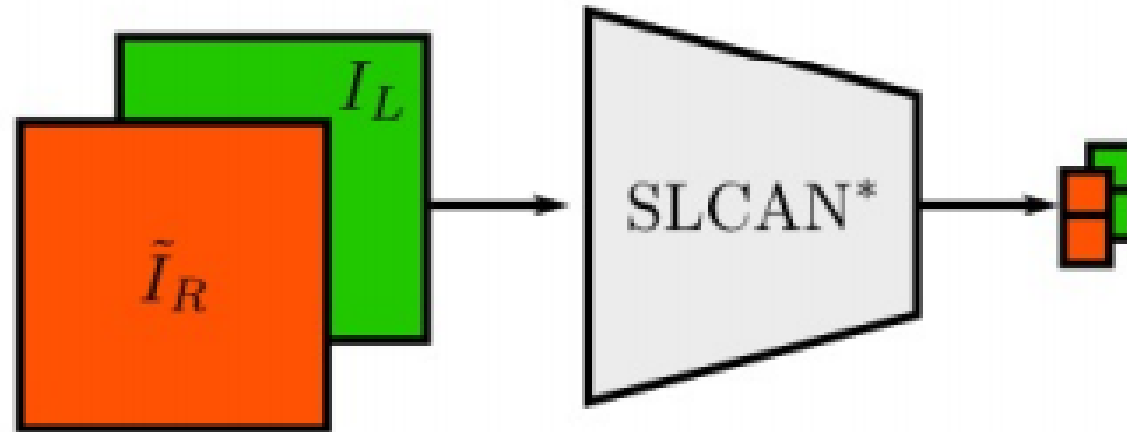
IV 2017 : <http://iv2017.org>

Paper URL: <http://arxiv.org/abs/1706.08211>



Autonomous Vehicle

- CNN 의 Binary Classification Model 의 구조.
- Input 이 들어가면



Autonomous Vehicle

- C1 : blocked
- C2 : free
- C3 : undefined

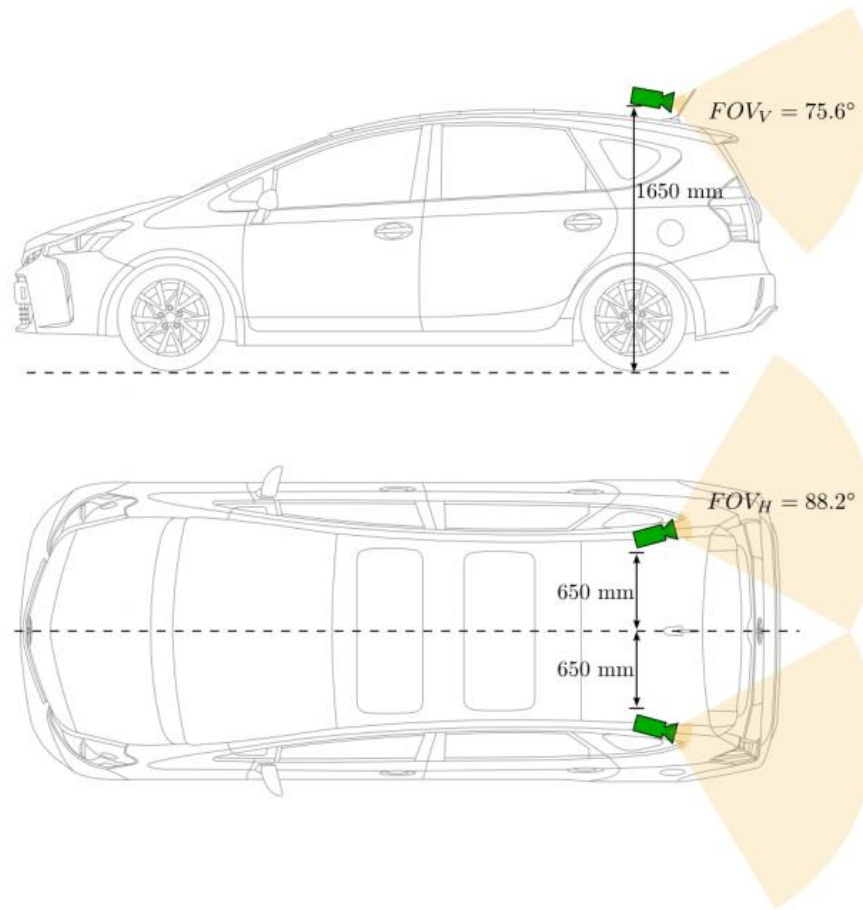


Fig. 2. Our research vehicle senses rear side space with cameras mounted at the both sides.



(a) BLOCKED



(b) FREE



(c) UNDEFINED

Fig. 3. Examples of the annotated images for left rear side view: (a) BLOCKED, (b) FREE, and (c) UNDEFINED

<http://arxiv.org/abs/1706.08211>



Autonomous Vehicle

- Saliency map 을 이용한 visualization
Saliency map 은 CAM 약간 이전의 논문에서 제안된 방법.
NAVER LABS 도 시간이 지나고 youtube 를 통해서
CAM 을 이용해서 보여 줌.
CAM 논문의 Saliency map 보다 더 성능이 좋다는 표

Table 2. Localization error on the ILSVRC validation set. *Backprop* refers to using [23] for localization instead of CAM.

Method	top-1 val.error	top-5 val. error
GoogLeNet-GAP	56.40	43.00
VGGnet-GAP	57.20	45.14
GoogLeNet	60.09	49.34
AlexNet*-GAP	63.75	49.53
AlexNet-GAP	67.19	52.16
NIN	65.47	54.19
Backprop on GoogLeNet	61.31	50.55
Backprop on VGGnet	61.12	51.46
Backprop on AlexNet	65.17	52.64
GoogLeNet-GMP	57.78	45.26

- Block 의 경우 차량을 보고 있음
- Free 의 경우 비어 있는 도로를 보고 있음

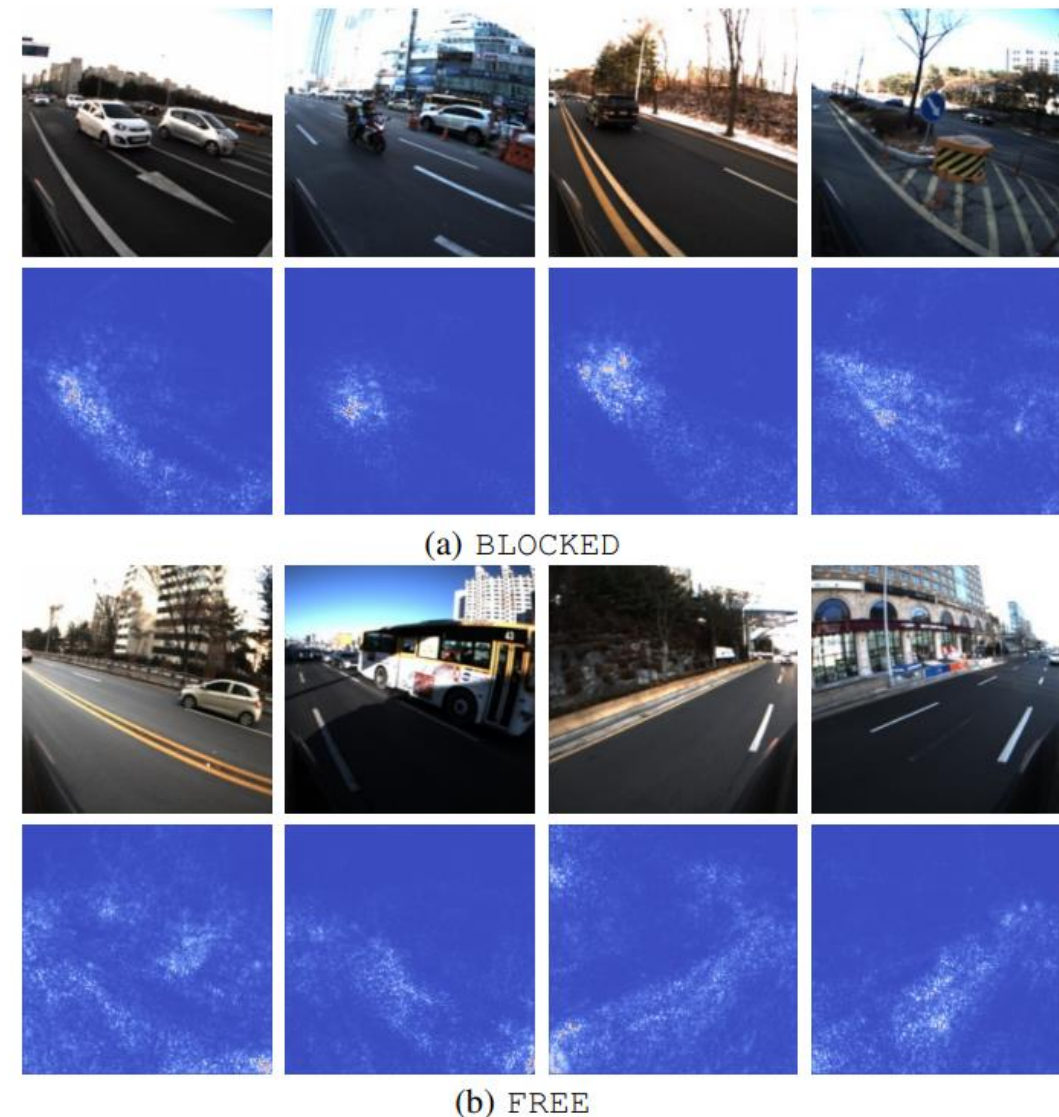


Fig. 8. Saliency maps [13] that show where on the image SLCAN focuses on when it classifies an image. It is apparent that SLCAN focuses on blocking obstacles when they exist and on the road surface when the lane is free.

아쉬운 점

- 지원님이 뭐 해보라고 제안해 주셨는데 그거 뭔지 모르겠음. 찾아보지 못함.
물론, 했어도 이해 못했을 듯.
- 신호처리와 영상처리 관점에서 Max Pooling 과 Average Pooling 이 가지는 의미를 분석해서 설명하려고 했지만, 아직 해당 내용을 올바르게 이해하지 못했기에 추가하지 못함.
- CAM은 자그마치 4년 전 논문으로, CAM 의 일반화 논문인 Grid-CAM 을 비롯하여 조금 더 최신 논문을 리뷰하지 못함.



Reference

- 한국어 : Interpretable Machine Learning 개요: (2) 이미지 인식 문제에서의 딥러닝 모델의 주요 해석 방법 :
<http://research.sualab.com/introduction/2019/10/23/interpretable-machine-learning-overview-2.html>
- 한국어 : youtube KoreaUniv DSBA CNN Localization 2 (CAM, grad CAM, PDA) :
<https://youtu.be/aGIEVeaKLgY>
- 한국어 : 논문 요약 Network In Network : <https://arclab.tistory.com/162>
- 논문 :
http://cnnlocalization.csail.mit.edu/Zhou_Learning_Deep_Features_CVPR_2016_paper.pdf
- 한국어 : NAVER LABS, CNN 기반 차선 변경 판단 알고리즘 :
<https://www.naverlabs.com/storyDetail/35>



- <https://arxiv.org/abs/1905.04899>

cutmix

- <https://www.google.com/search?q=cutmix&oq=cutmix&aqs=chrome.69i59j0l6j69i61.891j0j7&sourceid=chrome&ie=UTF-8>

