CNN (Image Classification)

LG 인화원 교육
윤세영
KAIST 김재철AI대학원

KAIST

Important References

Stanford CS231n course
http://cs231n.stanford.edu/index.html
Lecture slides, Youtube video,

Coursera Deep Learning course by Andrew Ng
https://www.deeplearning.ai
Not free if you want to get certifications

PyTorch Deep Learning Mini Course
https://github.com/Atcold/PyTorch-Deep-Learning-Minicourse

Many source codes in Github

목차

Computer Vision (시각지능)

본 강의에서는 Convolutional Neural Network (CNN) 기반의

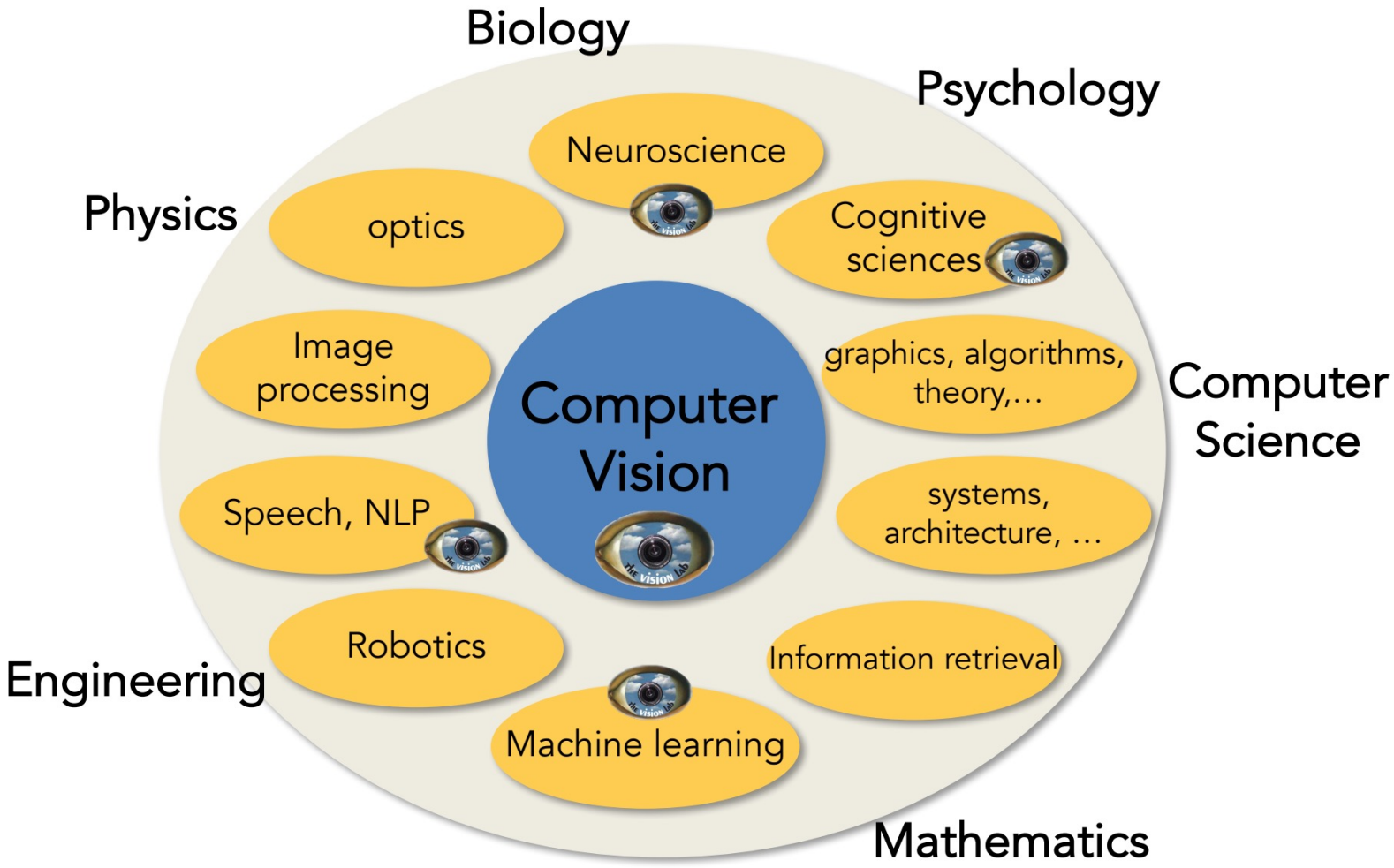
- Image Classification
- Object Detection
- Segmentation

을 다룰 예정입니다.

Application 으로는 (from wikipedia),

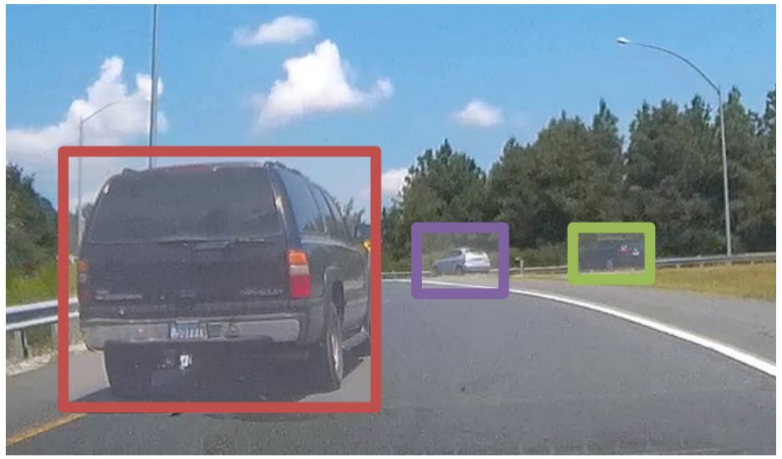
- Automatic inspection, e.g., in manufacturing applications;
- Assisting humans in identification tasks, e.g., a species identification system;
- Controlling processes, e.g., an industrial robot;
- Detecting events, e.g., for visual surveillance or people counting;
- Interaction, e.g., as the input to a device for computer-human interaction;
- Modeling objects or environments, e.g., medical image analysis or topographical modeling;
- Navigation, e.g., by an autonomous vehicle or mobile robot; and
- Organizing information, e.g., for indexing databases of images and image sequences.

Computer Vision (시각지능)

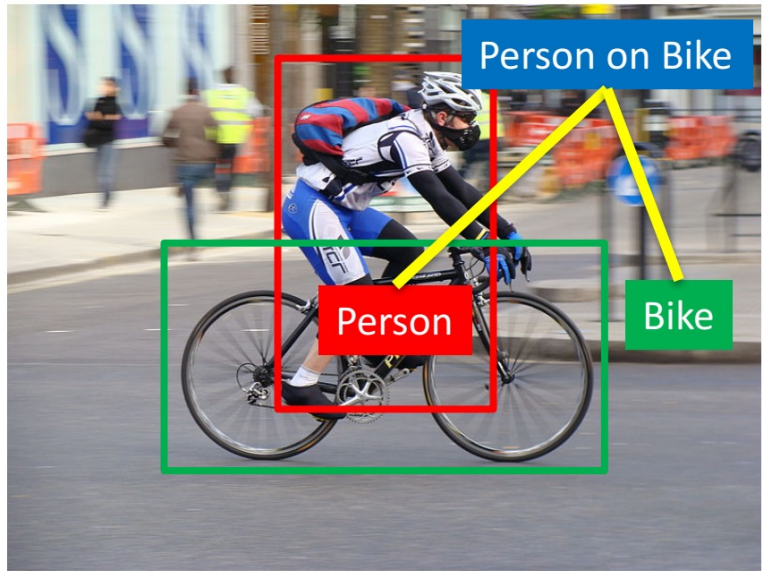


Li et al, CS231 Lecture note

Computer Vision Tasks



- Object detection
- Action classification
- Image captioning
- …



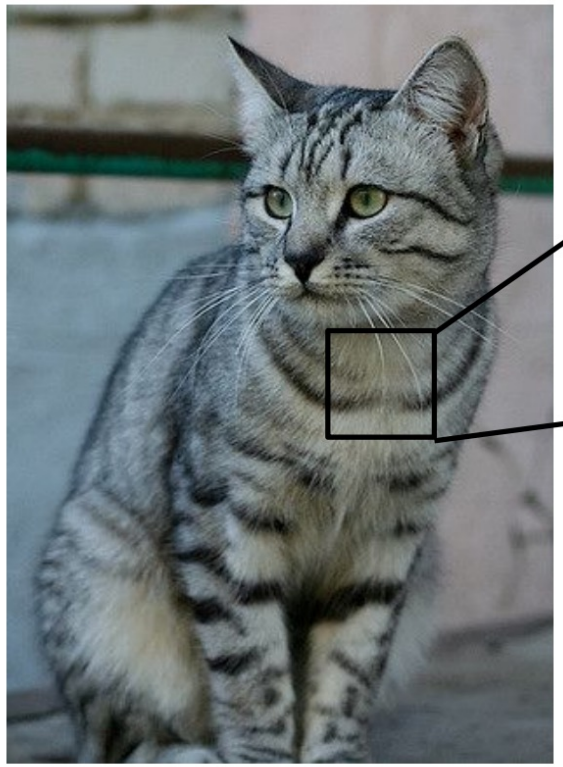
Li et al, CS231 Lecture note

Image Classification (CV의 시작점)

Li et al, CS231 Lecture note

Human Vision vs. Computer Vision

The Problem: Semantic Gap



This image by Nikita is licensed under CC-BY 2.0

```
[[105 112 108 111 104  99 106  99  96 103 112 119 104  97  93  87]
 [ 91  98 102 106 104  79  98 103  99 105 123 136 110 105  94  85]
 [ 76  85  90 105 128 105  87  96  95  99 115 112 106 103  99  85]
 [ 99  81  81  93 120 131 127 100  95  98 102  99  96  93 101  94]
 [106  91  61  64  69  91  88  85 101 107 109  98  75  84  96  95]
 [114 108  85  55  55  69  64  54  64  87 112 129  98  74  84  91]
 [133 137 147 103  65  81  80  65  52  54  74  84 102  93  85  82]
 [128 137 144 140 109  95  86  70  62  65  63  63  60  73  86 101]
 [125 133 148 137 119 121 117  94  65  79  80  65  54  64  72  98]
 [127 125 131 147 133 127 126 131 111  96  89  75  61  64  72  84]
 [115 114 109 123 150 148 131 118 113 109 100  92  74  65  72  78]
 [ 89  93  90  97 108 147 131 118 113 114 113 109 106  95  77  80]
 [ 63  77  86  81  77  79 102 123 117 115 117 125 125 130 115  87]
 [ 62  65  82  89  78  71  80 101 124 126 119 101 107 114 131 119]
 [ 63  65  75  88  89  71  62  81 120 138 135 105  81  98 110 118]
 [ 87  65  71  87 106  95  69  45  76 130 126 107  92  94 105 112]
 [118  97  82  86 117 123 116  66  41  51  95  93  89  95 102 107]
 [164 146 112  80  82 120 124 104  76  48  45  66  88 101 102 109]
 [157 170 157 120  93  86 114 132 112  97  69  55  70  82  99  94]
 [130 128 134 161 139 100 109 118 121 134 114  87  65  53  69  86]
 [128 112  96 117 150 144 120 115 104 107 102  93  87  81  72  79]
 [123 107  96  86  83 112 153 149 122 109 104  75  80 107 112  99]
 [122 121 102  80  82  86  94 117 145 148 153 102  58  78  92 107]
 [122 164 148 103  71  56  78  83  93 103 119 139 102  61  69  84]]
```

What the computer sees

An image is just a big grid of numbers between [0, 255]:

e.g. 800 x 600 x 3
(3 channels RGB)

고양이가 옆으로 이동하면? 20도 기울어졌으면? 사진이 어두워지면?

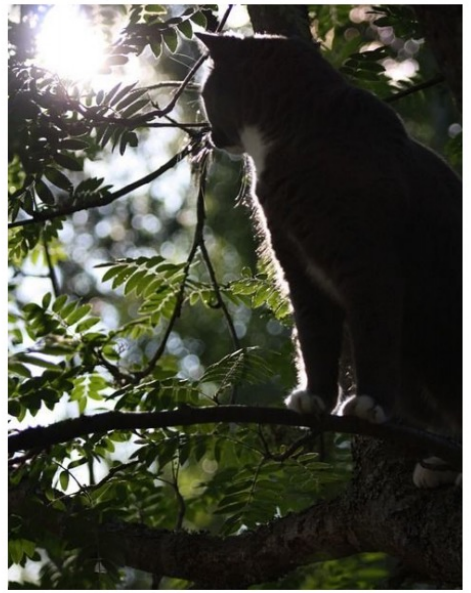
Li et al, CS231 Lecture note

Challenges: Illumination

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

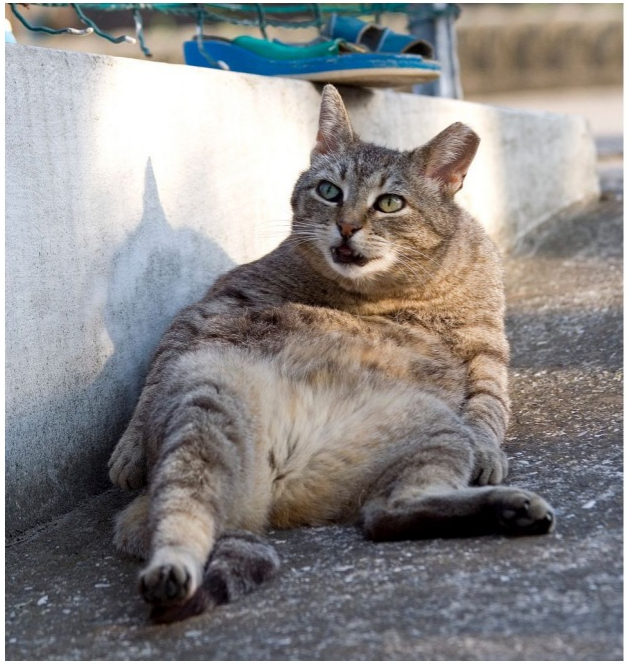
This image is CC0 1.0 public domain



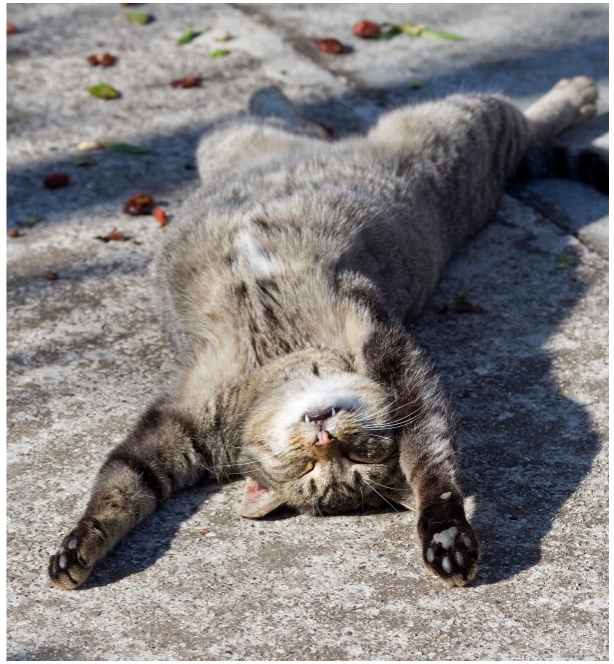
This image is CC0 1.0 public domain

Li et al, CS231 Lecture note

Challenges: Deformation

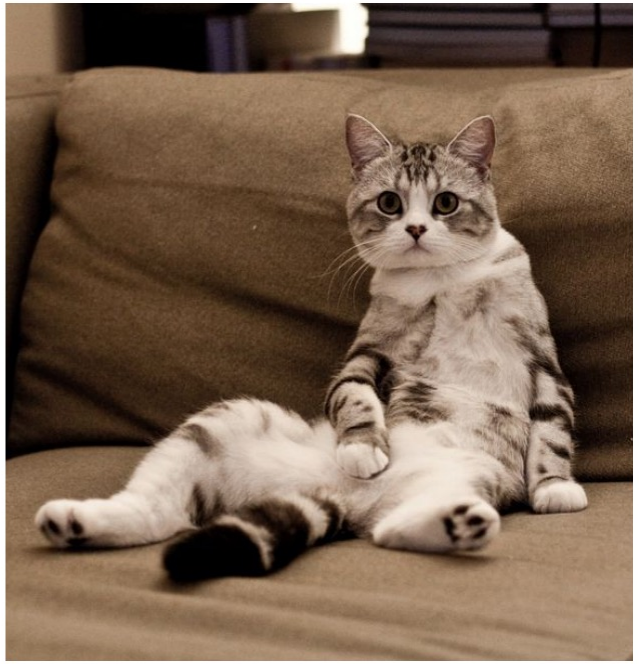


This image by Umberto Salvagnin is licensed under CC-BY 2.0



This image by Umberto Salvagnin is licensed under CC-BY 2.0

This image by sare bear is licensed under CC-BY 2.0



This image by Tom Thai is licensed under CC-BY 2.0

Challenges: Occlusion

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

This image by jonsson is licensed under CC-BY 2.0

Li et al, CS231 Lecture note

Challenges: Background Clutter

This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

Li et al, CS231 Lecture note

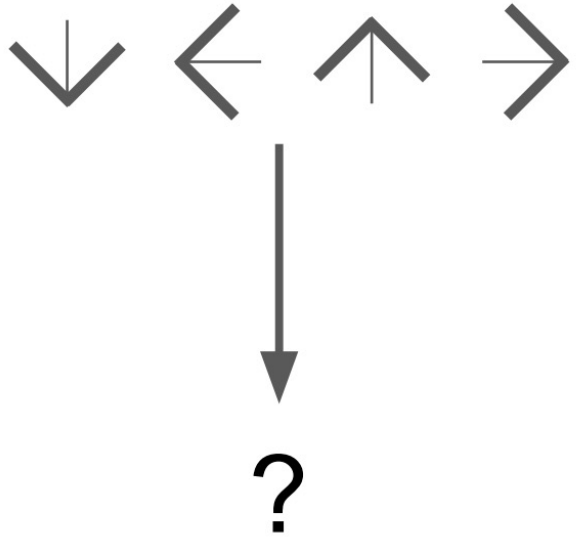
Challenges: Intraclass variation

Li et al, CS231 Lecture note

기존 CV 기법

Find edges

Find corners



?

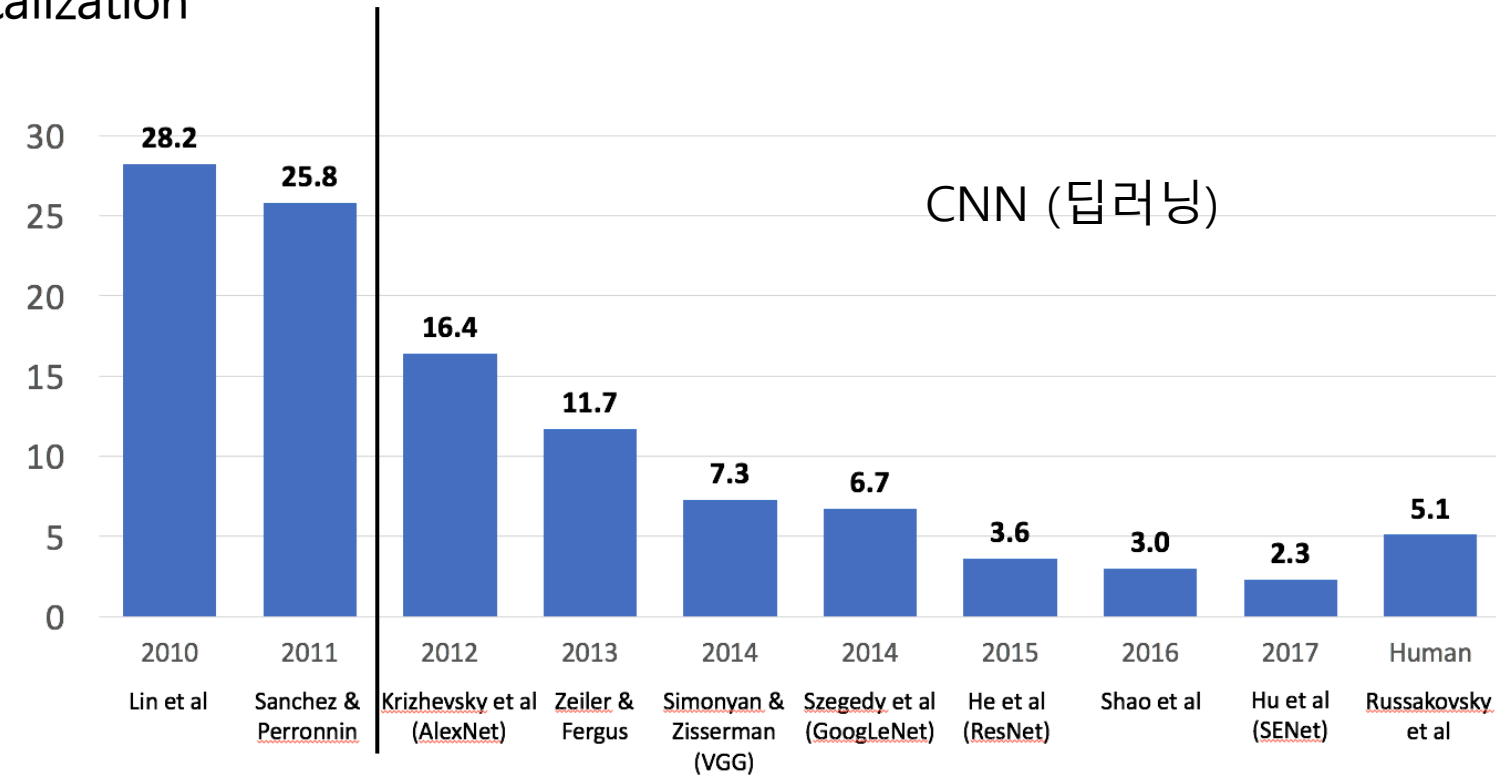
Li et al, CS231 Lecture note

목차

CNN (Deep Learning) 은 왜 갑자기 관심을 얻었나?

ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- Image classification
- Single-object localization
- Object Detection

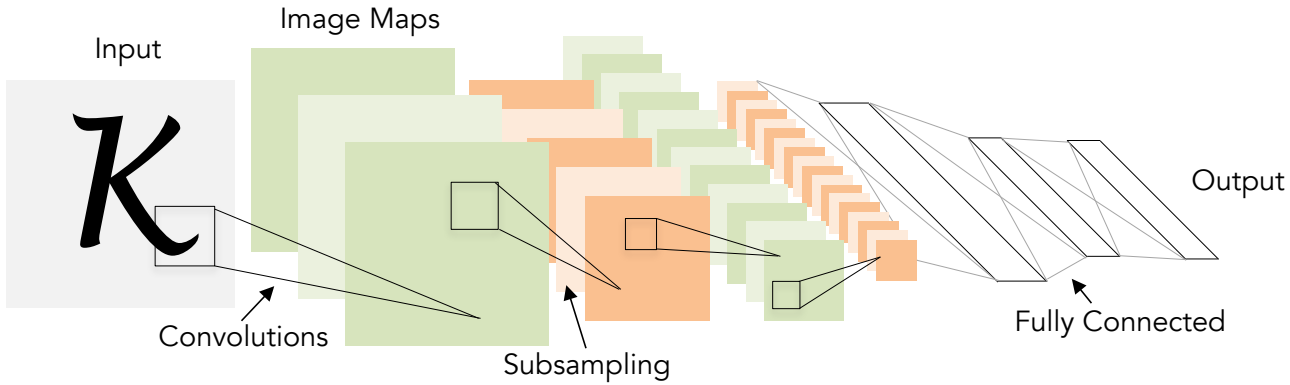


Ex) Image Classification Challenge: 1,000 object classes 1,431,167 images

Russakovsky et al. IJCV 2015
Li et al, CS231 Lecture note

CNN의 탄생

1998
LeCun et al.



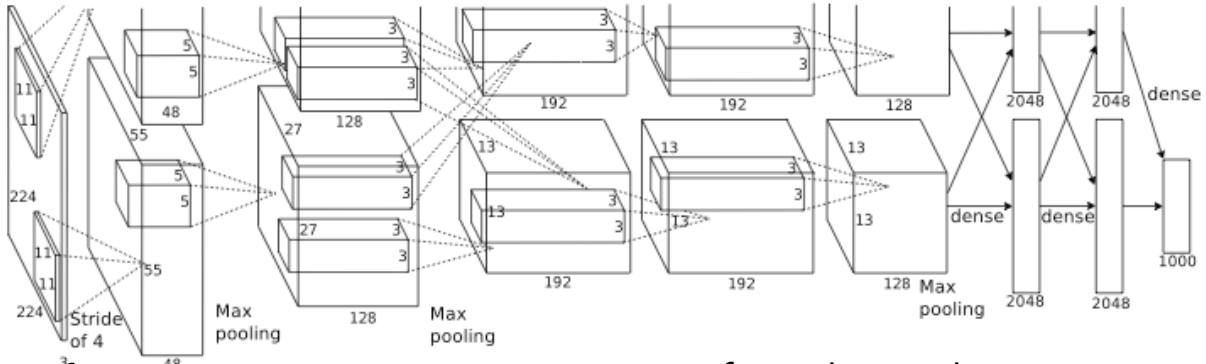
# of transistors

10^6

# of pixels used in training

10^7 NIST

2012
Krizhevsky et al.



# of transistors

10^9

GPUs

# of pixels used in training

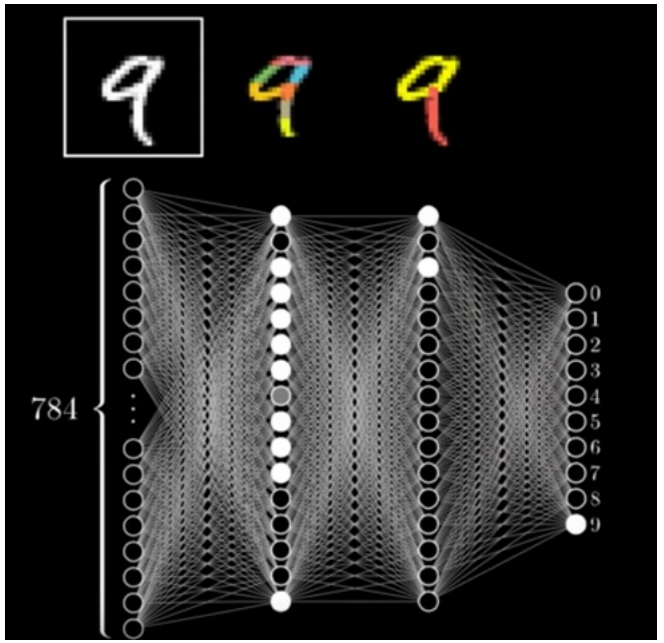
10^{14} IMAGENET

Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Li et al, CS231 Lecture note

왜 CNN이 필요한가?

Fully Connected Neural Net 으로 사진을 분류하면 어떤 문제가 발생하나?

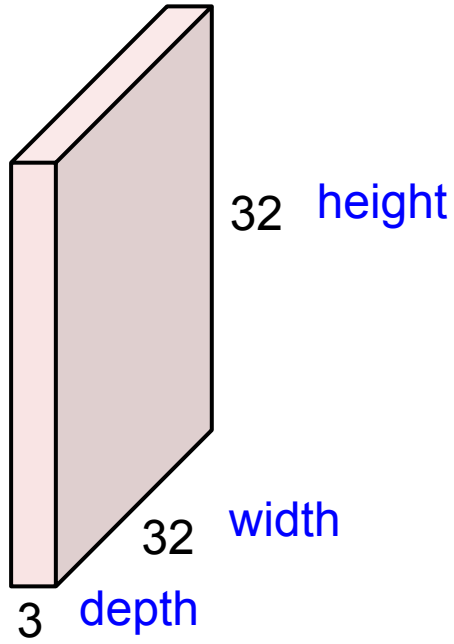


이미지의 특징을 활용하기 어렵다.

무엇보다 학습해야 하는 파라미터의 수가 너무 많다!

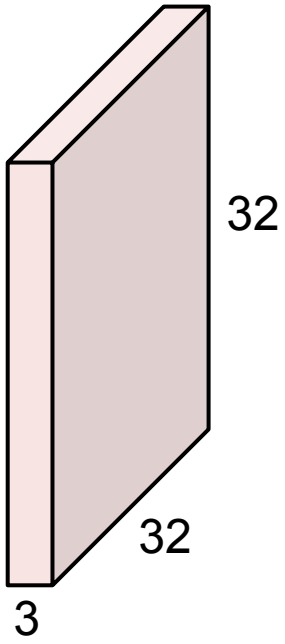
Convolutional Layer (1/5)

32x32x3 image -> preserve spatial structure

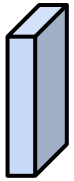


Convolutional Layer (2/5)

32x32x3 image

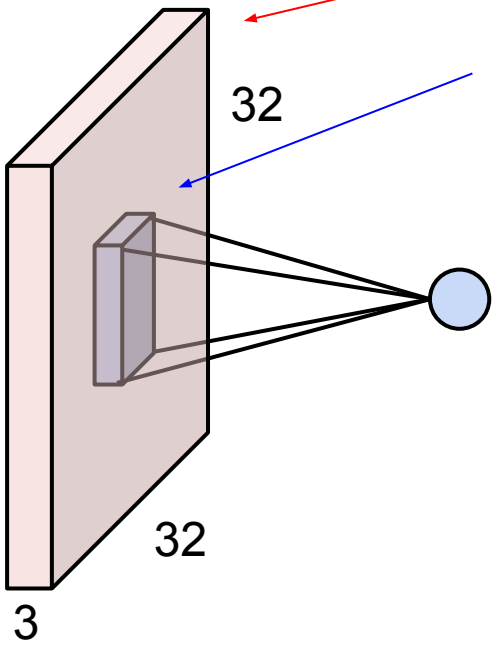


5x5x3 filter



Convolve the filter with the image
i.e. “slide over the image spatially,
computing dot products”

Convolutional Layer (3/5)



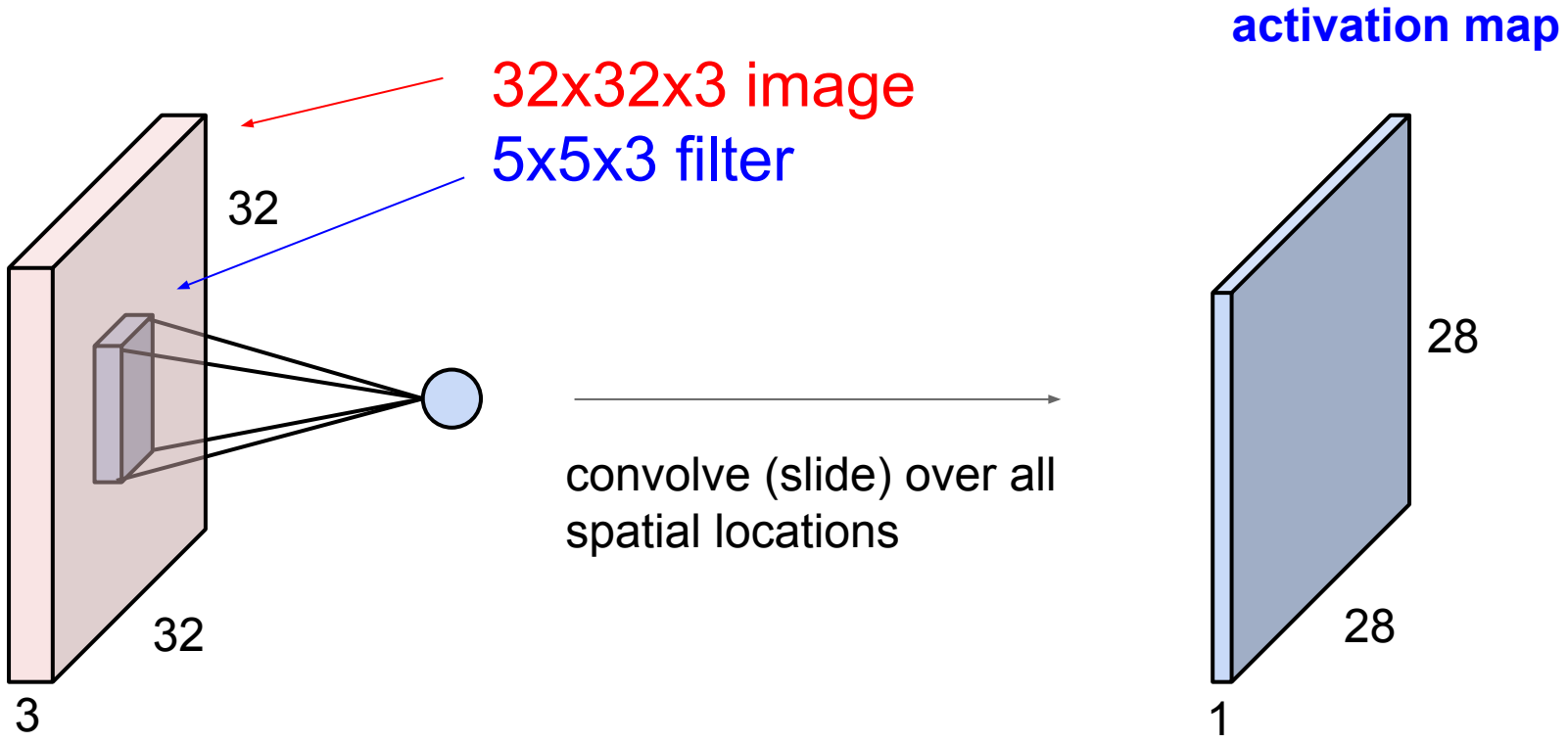
32x32x3 image

5x5x3 filter w

1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

Convolutional Layer (4/5)



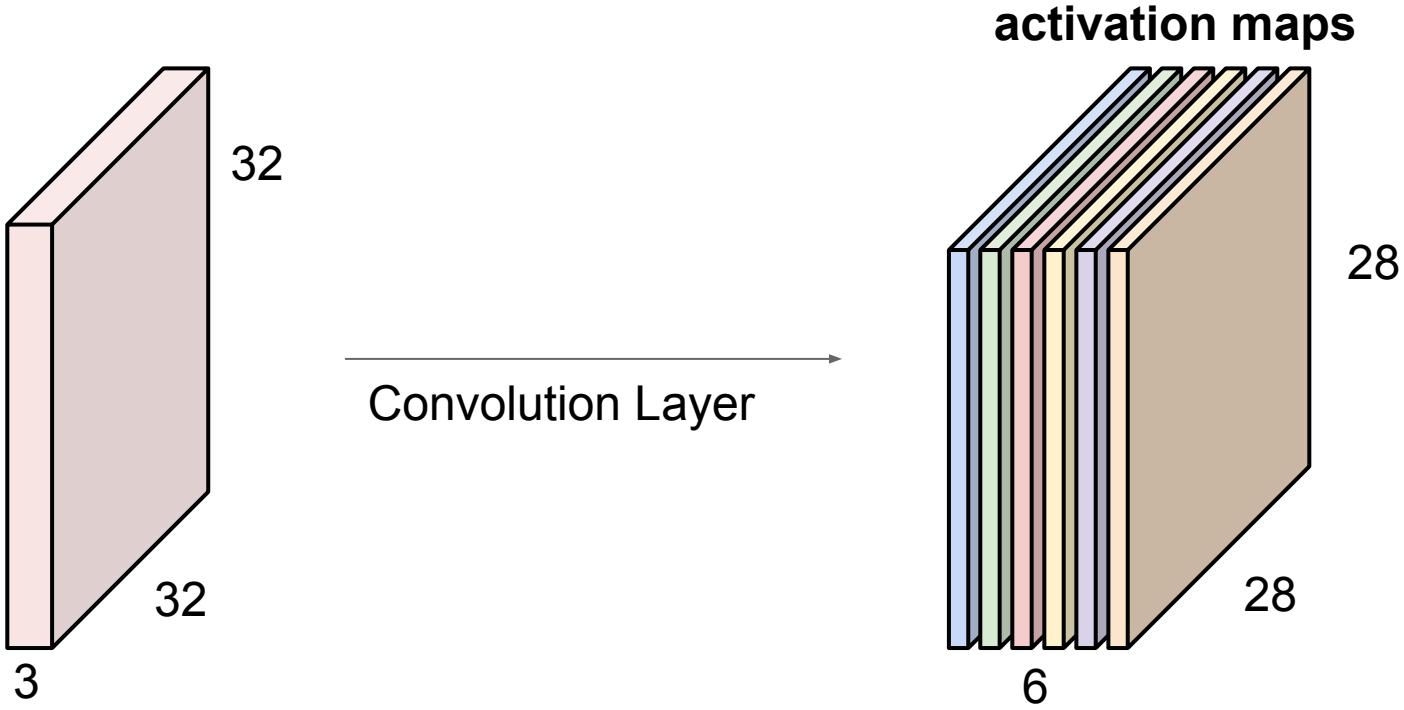
같은 parameter를 재사용하면서 parameter의 수를 절약!

이미지의 local한 정보를 충분히 활용 가능!

Li et al, CS231 Lecture note

Convolutional Layer (5/5)

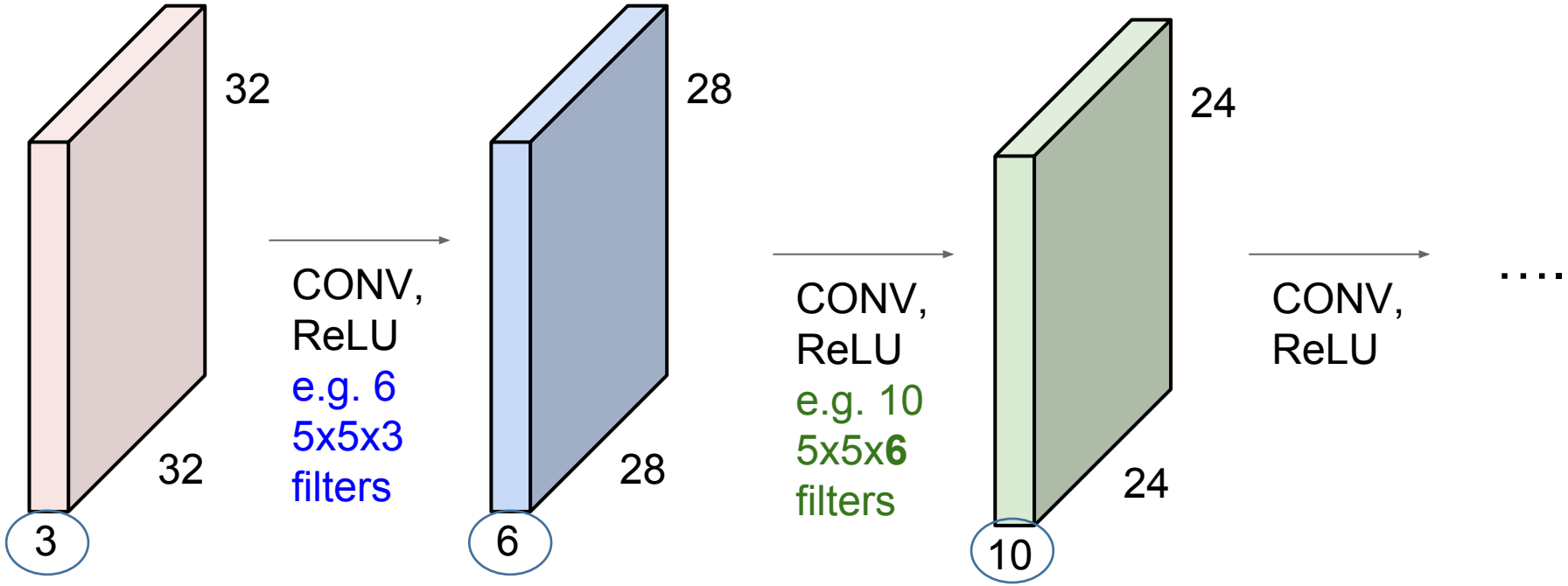
For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



We stack these up to get a "new image" of size 28x28x6!

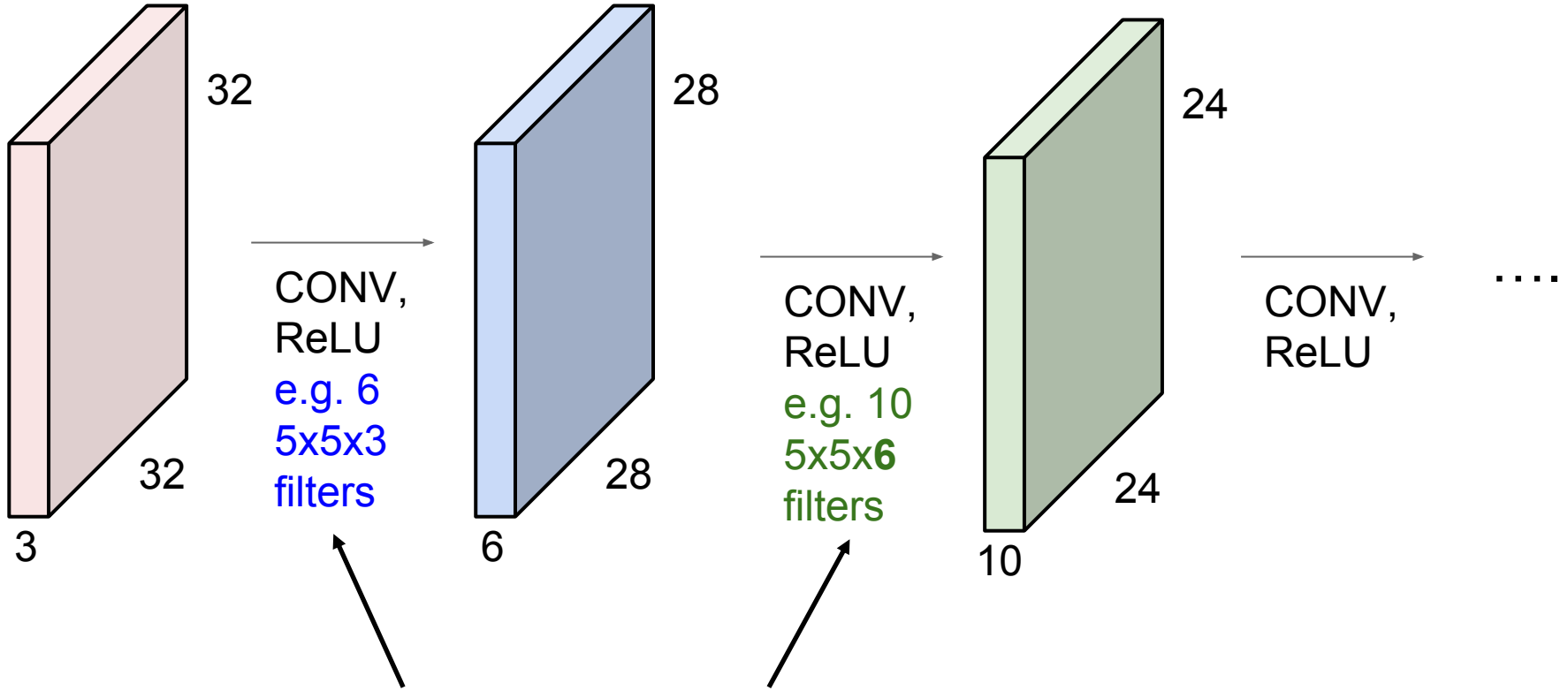
Convolutional Neural Network

CNN: Convolutional layer를 활용한 neural network



Depth = Image의 수 = Channel 의 수

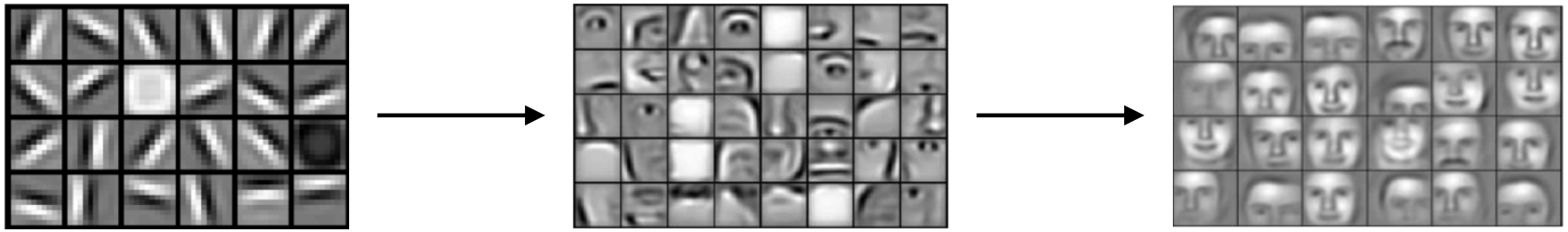
Quiz

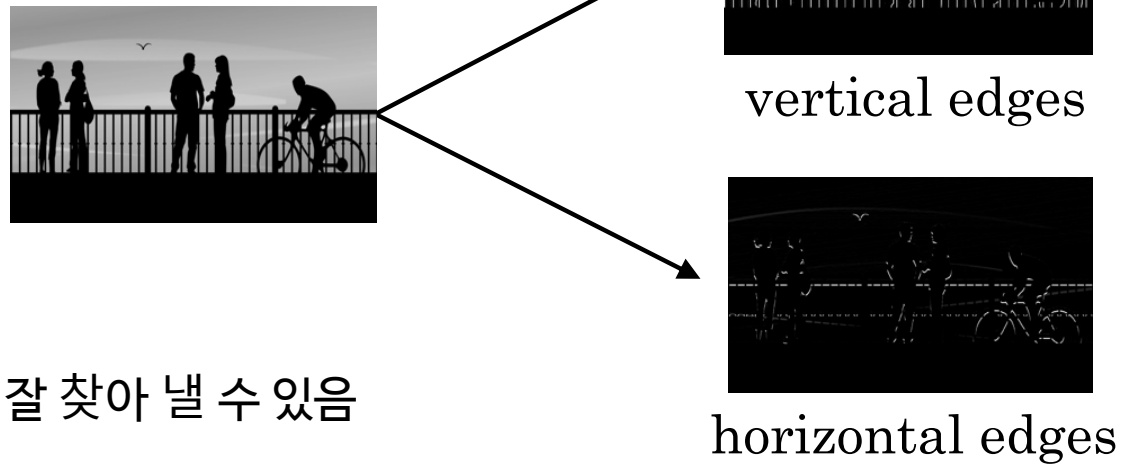


각 각 몇 개의 parameter 들이 필요한가?

Edge Detection and Convolutional layer

Computer vision에서 edge는 매우 중요한 정보임





Convolutional layer 들이 edge를 잘 찾아 낼 수 있음

예) Vertical Edge Detector

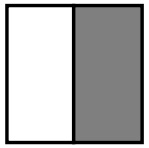
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

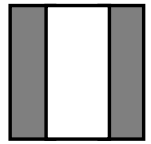
1	0	-1
1	0	-1
1	0	-1

=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0



*



주의사항: 실제 CNN이 이와 같은 filter를 가지고 있거나 사용자가 직접 filter 정보를 입력하지 않음! CNN은 학습을 통하여 적절한 filter를 찾아냄

Quiz

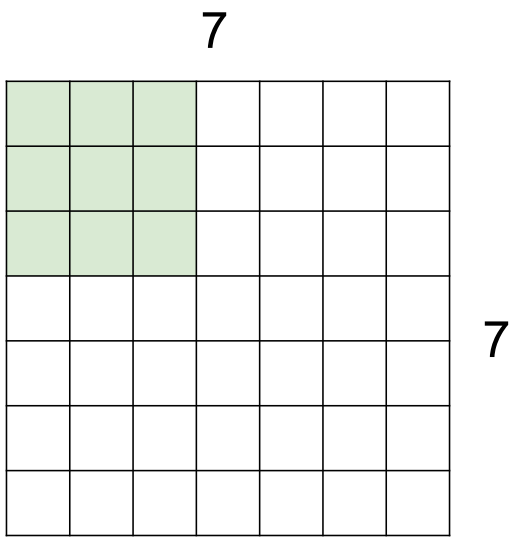
Q1. Convolution layer 에서 filter size가 커지면 장점이 무엇일까요?

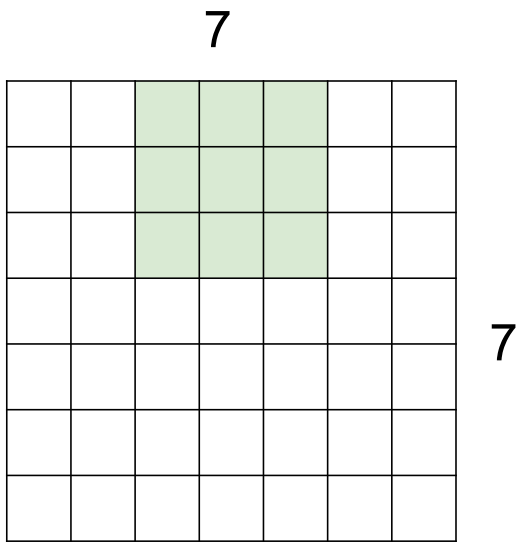
Q2. Convolution layer 에서 filter size가 커지면 단점이 무엇일까요?

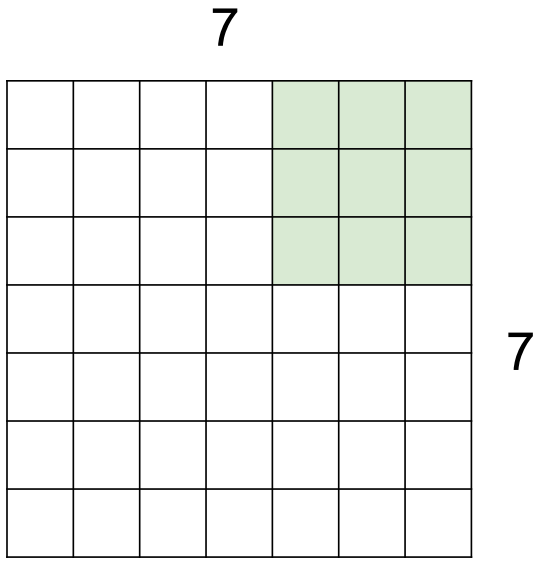
Q3. Convolution layer 에서 filter 값들은 누가 어떻게 정해야 하는 것인가요?

Stride

Stride를 통하여 CNN의 Dimension을 조절 할 수 있음



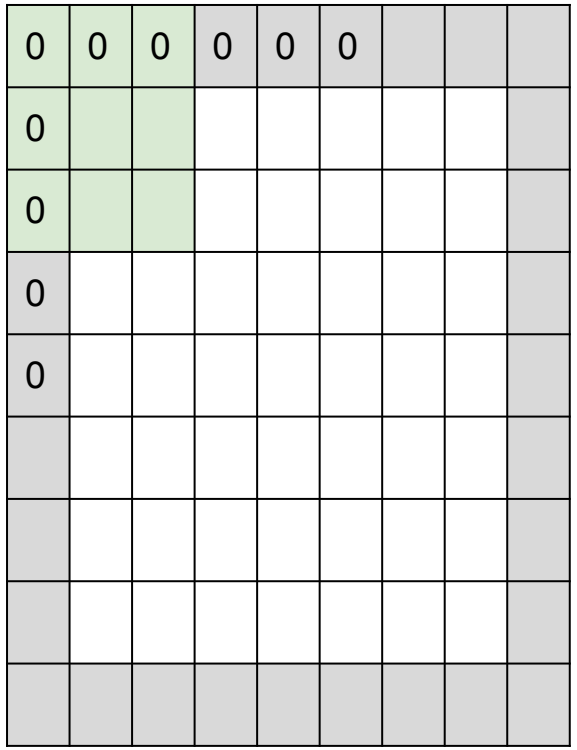




Li et al, CS231 Lecture note

Zero-padding

상황에 따라 0으로 가득 찬 column과 row를 위아래 양옆으로 추가하여 dimension 조절



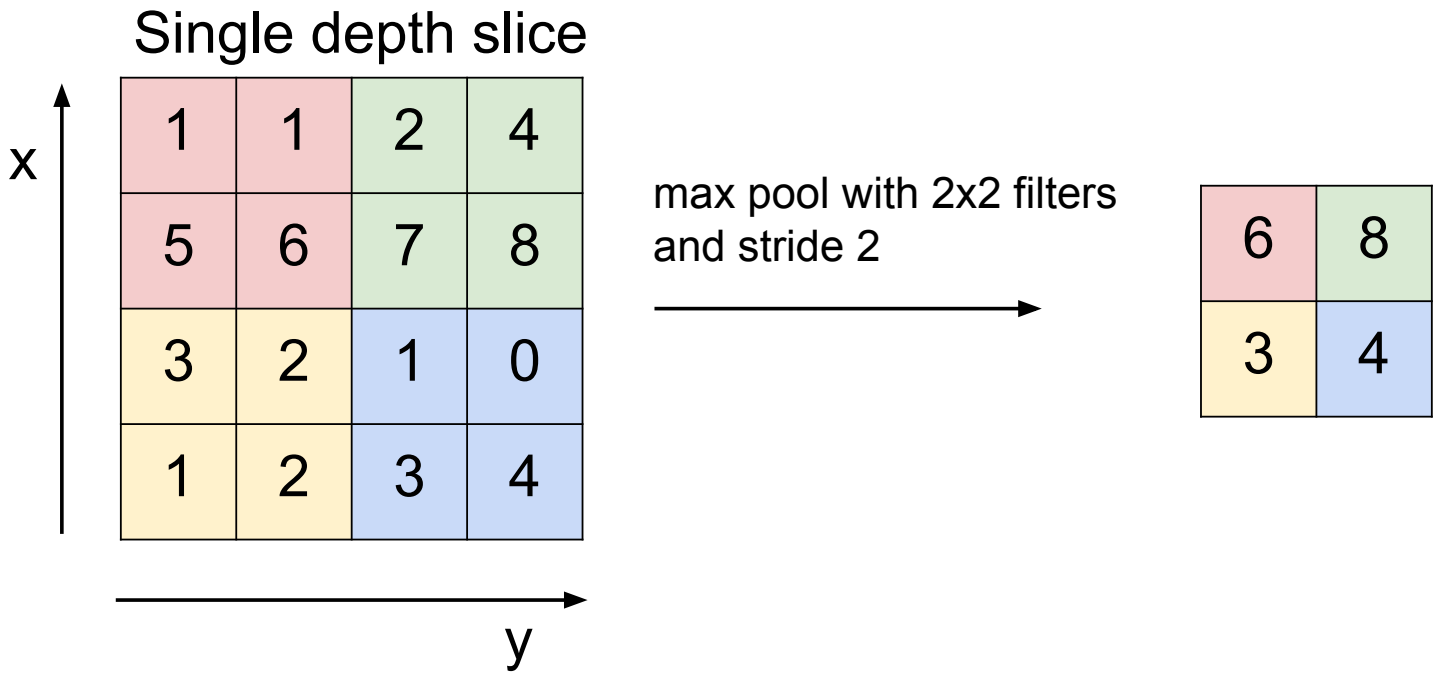
Li et al, CS231 Lecture note

Max-Pooling

다음 layer로 전달되는 dimension이 작으면 작을수록 다음과 같은 점에서 유리함

- 계산 시간이 단축됨
- Image가 더 압축되며 중요한 정보만 남게 됨

Stride와 padding을 활용하여 우리가 원하는 만큼 dimension을 조절 할 수 있으나 Stride는 규칙적으로 column과 row를 소거하는 방식으로 중요한 정보를 지울 가능성이 존재



Li et al, CS231 Lecture note

목차

AlexNet (1/2)

Convolutional layer를 깊게 쌓고 Convolution 후에는 pooling(down sampling) layer를 추가하는 design을 시도한 architecture. 구조에 필요한 연산량과 변수가 많아 두 개의 GPU를 사용해 분산처리 했으며 이전 모델들에 비해 성능을 크게 향상시킴.

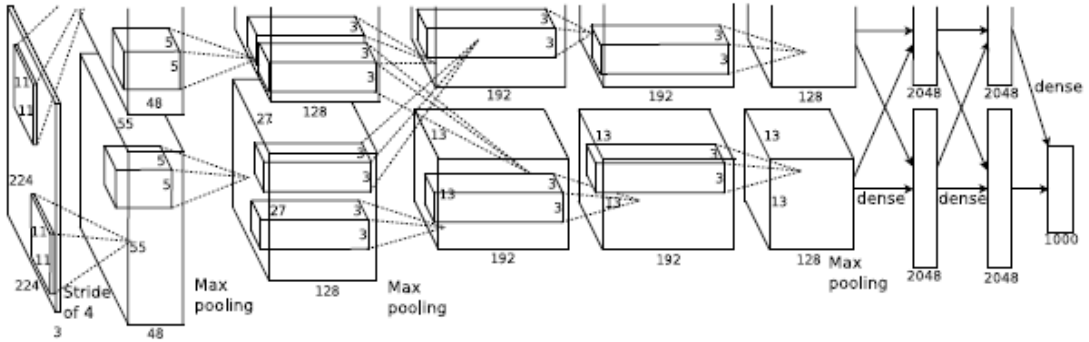


Figure 1. An illustration of the architecture of AlexNet, explicitly showing the delineation of responsibilities between the two GPUs.

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Table 1
Comparison of results on ILSVRC 2010 test set

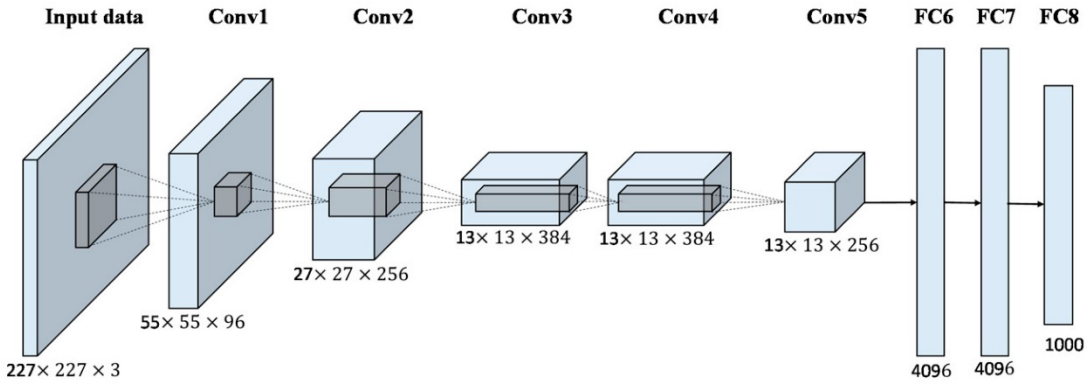


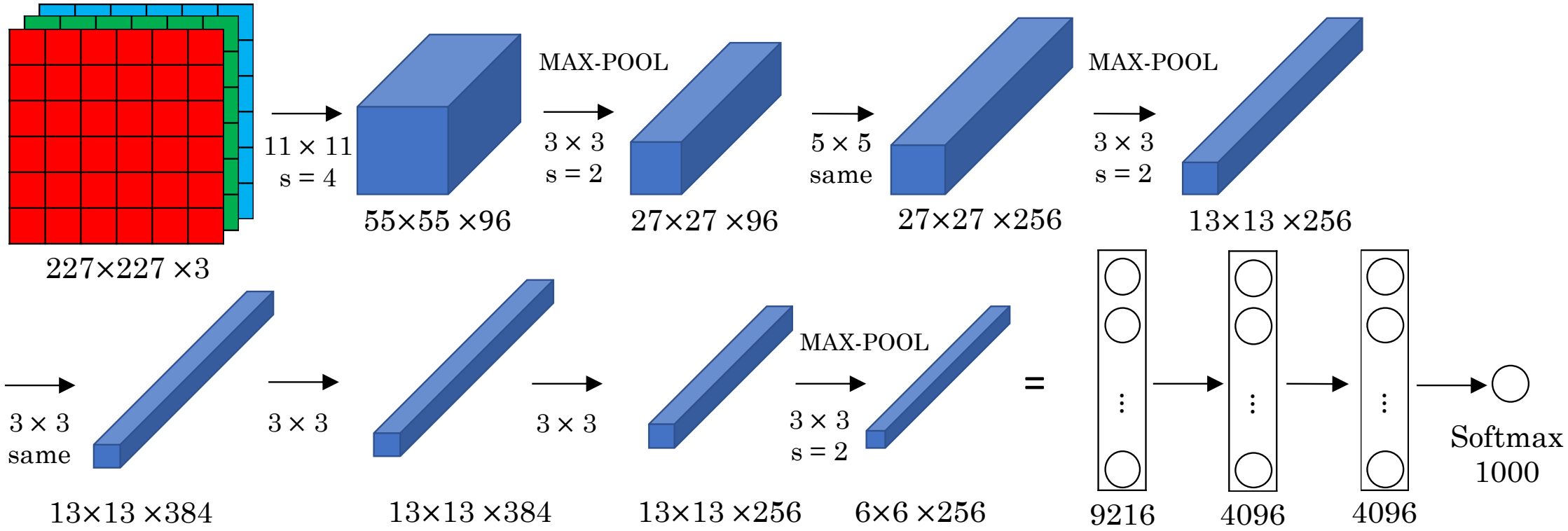
Figure 2. The architecture design of AlexNet with one GPU.

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Table 2
Comparison of error rates on ILSVRC 2012 validation and test sets.

출처 : Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012. Figure 2 from http://www.mdpi.com/2072-4292/9/8/848

AlexNet (2/2)


227×227 ×3
11 × 11
s = 4
55×55 ×96
MAX-POOL
3 × 3
s = 2
27×27 ×96
5 × 5
same
27×27 ×256
MAX-POOL
3 × 3
s = 2
13×13 ×256
3 × 3
same
13×13 ×384
3 × 3
13×13 ×384
3 × 3
13×13 ×256
MAX-POOL
3 × 3
s = 2
6×6 ×256
=
9216
4096
4096
Softmax
1000

Softmax

Want to interpret raw classifier scores as **probabilities**

$$s = f(x_i; W)$$

$$P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$

Softmax Function

$$L_i = -\log P(Y = y_i|X = x_i)$$

	Unnormalized log-probabilities / logits	exp	unnormalized probabilities	normalize	probabilities	compare	Correct probs
	Probabilities must be >= 0				Probabilities must sum to 1		
cat	**3.2**		**24.5**		**0.13**		**1.00**
car	5.1		164.0		0.87		0.00
frog	-1.7		0.18		0.00		0.00

Kullback–Leibler divergence

$$D_{KL}(P||Q) = \sum_y P(y) \log \frac{P(y)}{Q(y)}$$

SVM loss function

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Li et al, CS231 Lecture note

VGGNet (1/2)

AlexNet에 비해 더 깊은 신경망이다. 층은 19개로 증가시키고, FNN 역시 3개나 존재한다. 그래서 연산량과 변수의 수가 AlexNet에 비해 매우 많다. 그러나 구조가 간단하고 이해가 쉽고 변형을 시키며 테스트하는 것에 용이하다.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 1. VGGNet의 배열을 나타낸 그림이다.filter 크기는 3과 1을 사용했으며, 크기가 1인 filter는 network의 nonlinearity의 표현력을 증가시키기 위해 사용했다.

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Table1. Number of parameters (in millions)

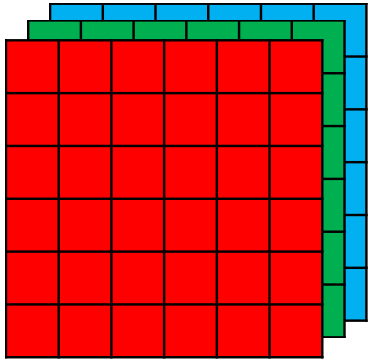
VGGNet 학습 방법

Deep network를 학습할 때, vanishing/exploding gradient issue가 생겨 학습이 안될 수도 있다. 하지만, 11개의 층을 가진 A 구조를 먼저 학습시키고 학습된 A구조에서 첫 4개의 층과 마지막 FNN의 값을 initialization해서 학습시켜 위의 문제를 해결했다.

출처 : Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

VGGNet (2/2)

CONV = 3×3 filter, s = 1, same MAX-POOL = 2×2 , s = 2



224×224 ×3

→ [CONV 64] ×2 → 224×224×64 → POOL → 112×112 ×64 → [CONV 128] ×2 → 112×112 ×128 → POOL → 56×56 ×128

→ [CONV 256] ×3 → 56×56 ×256 → POOL → 28×28 ×256 → [CONV 512] ×3 → 28×28 ×512 → POOL → 14×14×512

→ [CONV 512] ×3 → 14×14 ×512 → POOL → 7×7×512 → FC 4096 → FC 4096 → Softmax 1000

3 * 3 Filter

Q) 왜 3*3 을 사용하였나?

3개의 3*3 layer 를 통과하면 7*7 영역이 한 neuron 의 effective receptive field 가 된다. 즉 7*7과 비슷한 효과를 얻을 수 있다. 또한, 더 깊은 구조로 더 많이 non-linearity를 제공 가능하다.

7*7 filter 와 3*3 filter 를 3개 사용하는데 필요한 parameter 들의 수를 비교해보면 3*3쪽이 더 적은 parameter를 사용한다.

감사합니다!