Convolutional Neural Network

서울대학교병원 정보화실

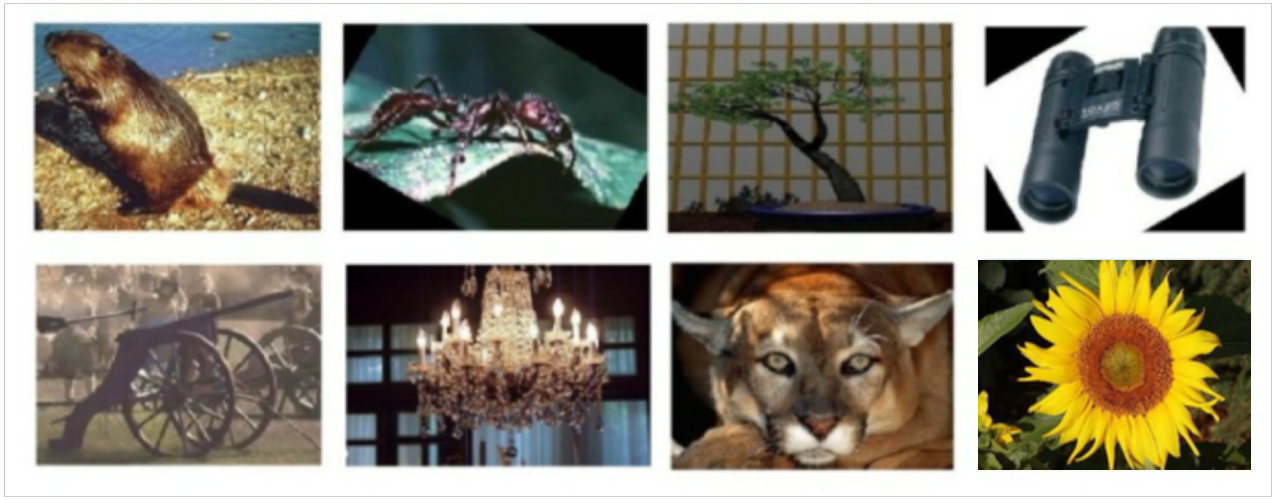
고태훈 (taehoonko@snuh.org)

최세원 (swc@snuh.org)

Motivation:
Why is CNN good?

Computer vision

- **Computer vision (CV)**
  - the design of computers that can process visual data and accomplish some given task
- **Object recognition**
  - Given some input images, identify which object it contains

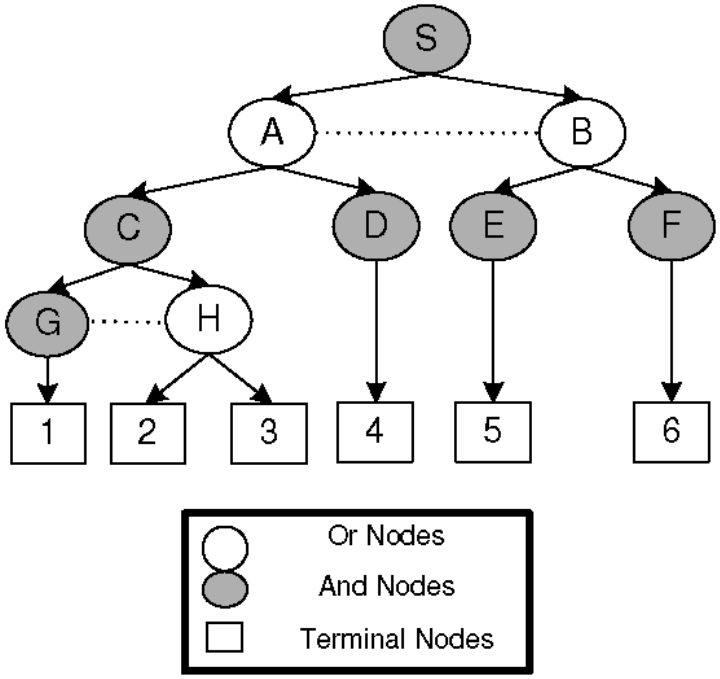


<Example images from Caltech 101 dataset>
(http://www.vision.caltech.edu/Image_Datasets/Caltech101/)

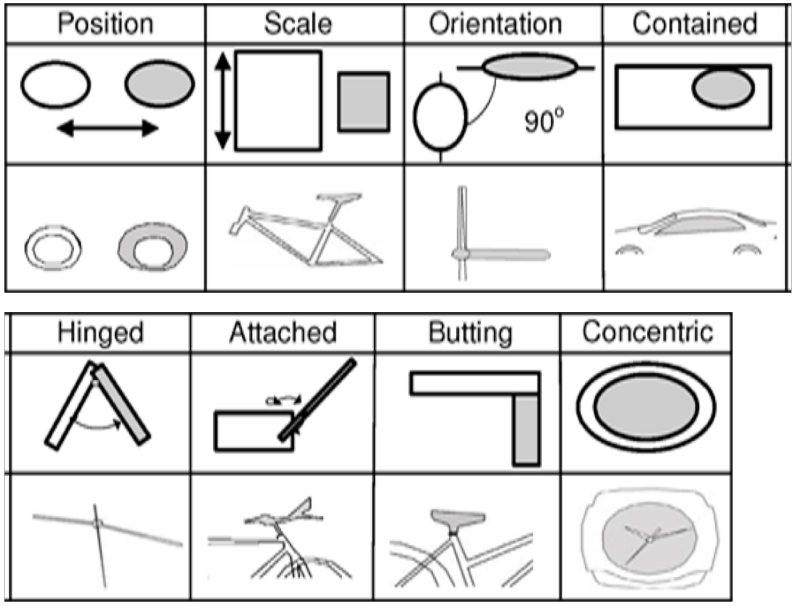
Which is the "sun flower"?

Object recognition

- **First way: Using hand-crafted feature extractor**
  - Users should design the method for representing image.



And-Or graph: The input image is decomposed to some object categories

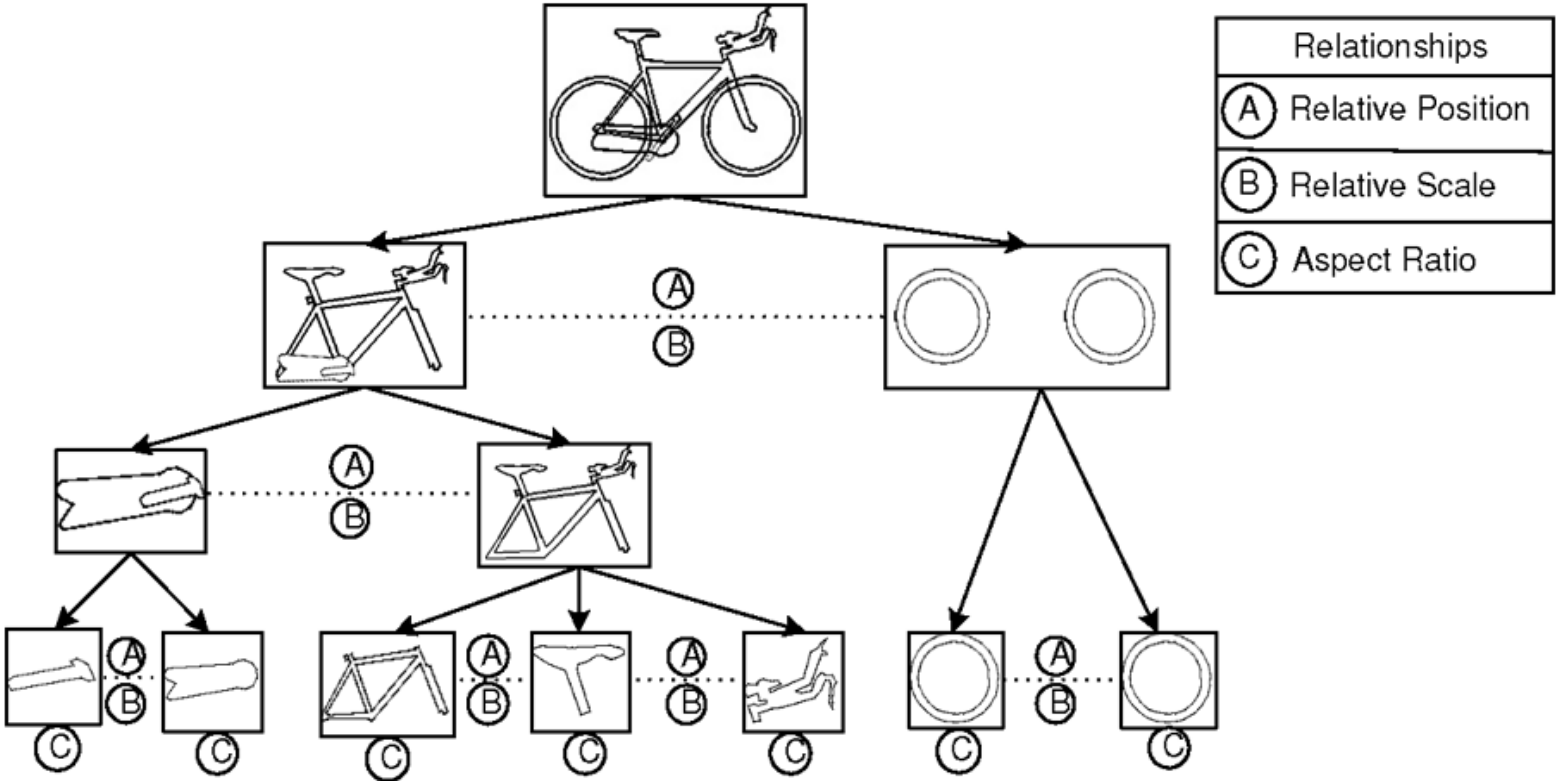


Relationship: Users define the relationship between categories (or nodes)

Porway, J., Yao, B., & Zhu, S. C. (2008). Learning compositional models for object categories from small sample sets. *Object categorization: computer and human vision perspectives*.

Object recognition

- **First way: Using hand-crafted feature extractor**
  - Users should design the method for representing image.

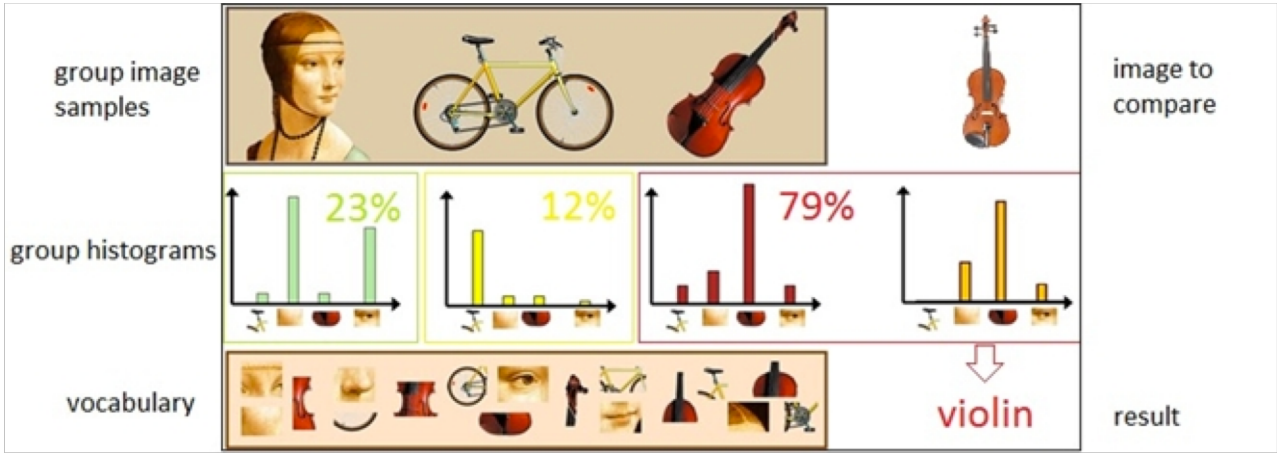


A parse graph of a bike: Nodes are composed via certain relation s to form the representation of the bike.

Porway, J., Yao, B., & Zhu, S. C. (2008). Learning compositional models for object categories from small sample sets. *Object categorization: computer and human vision perspectives*.

Object recognition

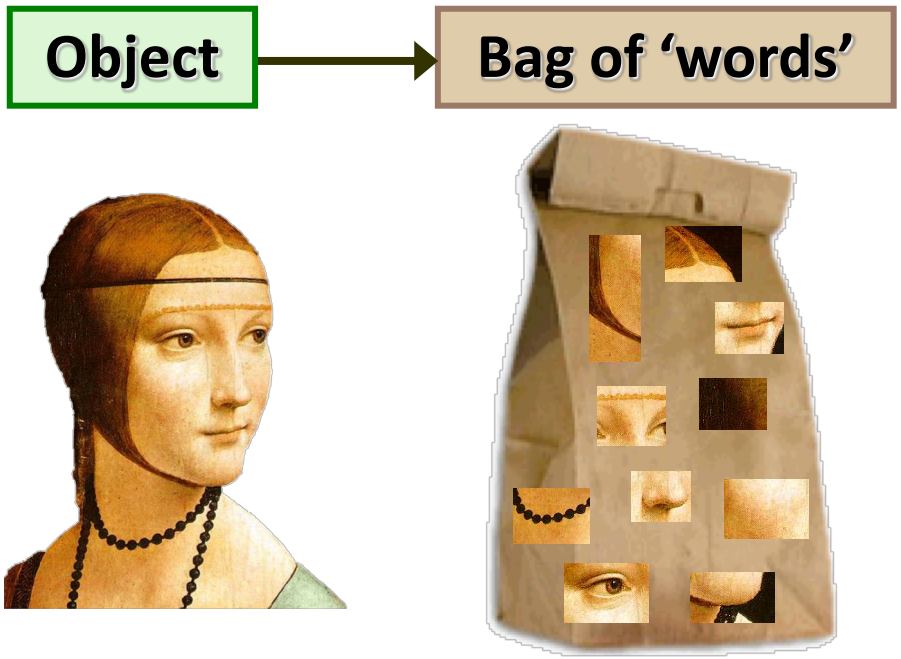
- In **computer vision**, the **bag-of-words model** (BoW model) can be applied to image classification, by treating image features as words.
- In document classification, a bag-of-words is a sparse vector of occurrence counts of words; that is, a sparse histogram over the vocabulary. In computer vision, a *bag of visual words* is a vector of occurrence counts of a vocabulary of local image features.

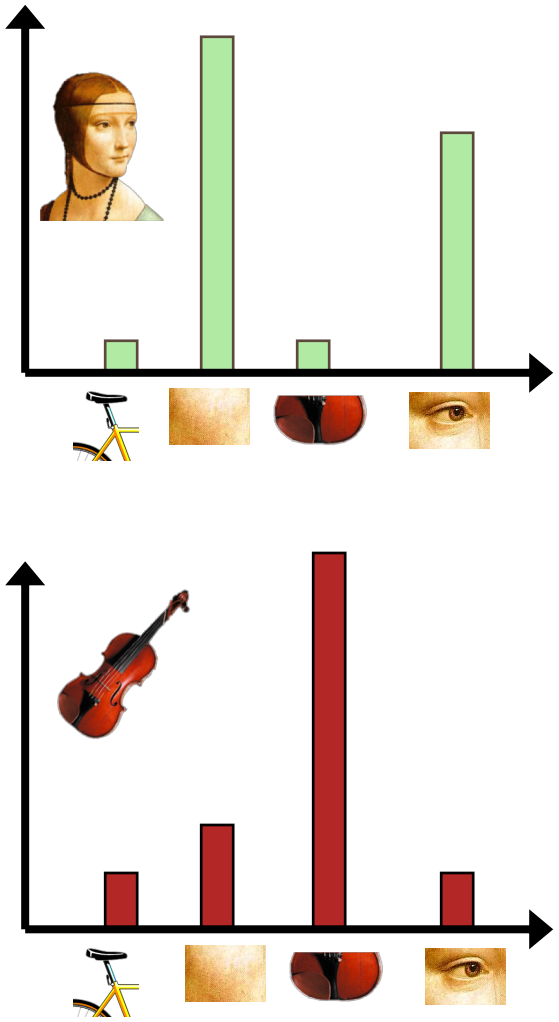


https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision
http://vgg.fiit.stuba.sk/2015-02/bag-of-visual-words-in-opencv/

Object recognition

- **Second way: Bag-of-words representation**
  - 이미지를 여러 (visual) words를 이용하여 표현
  - Histogram representation

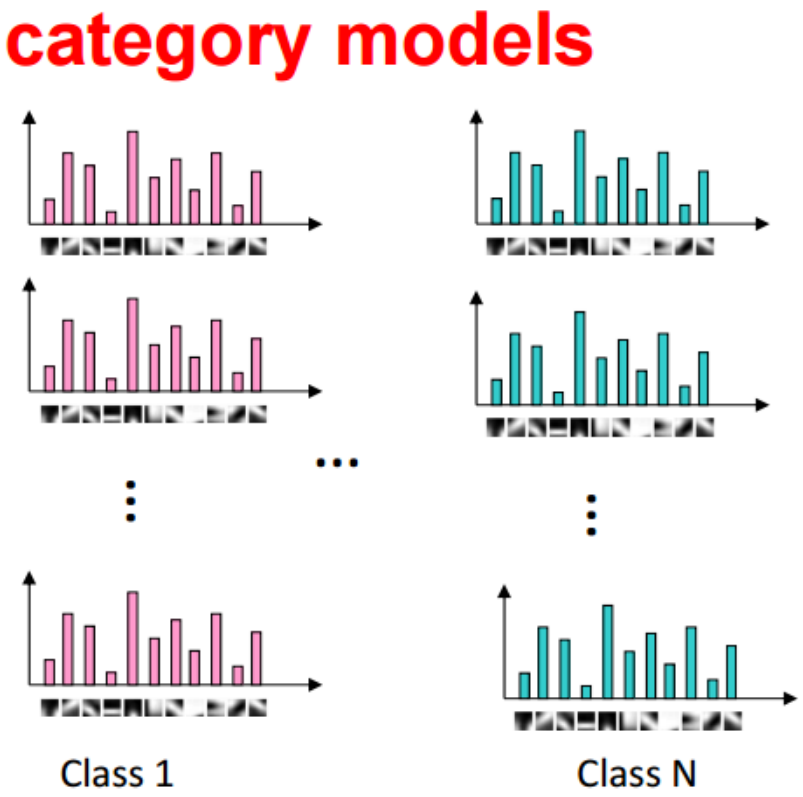


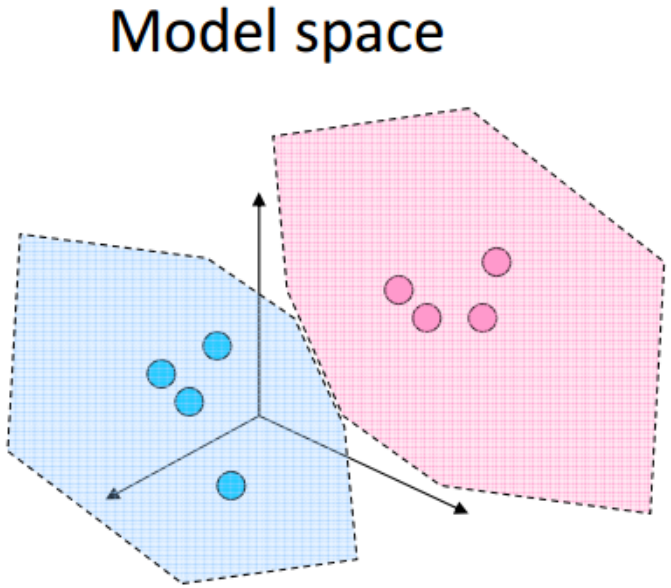


Source: Fei Fei Li's lecture notes

Object recognition

- **Second way: Bag-of-words representation**
  - Images ➔ Bag-of-words (BoW) ➔ Machine learning model





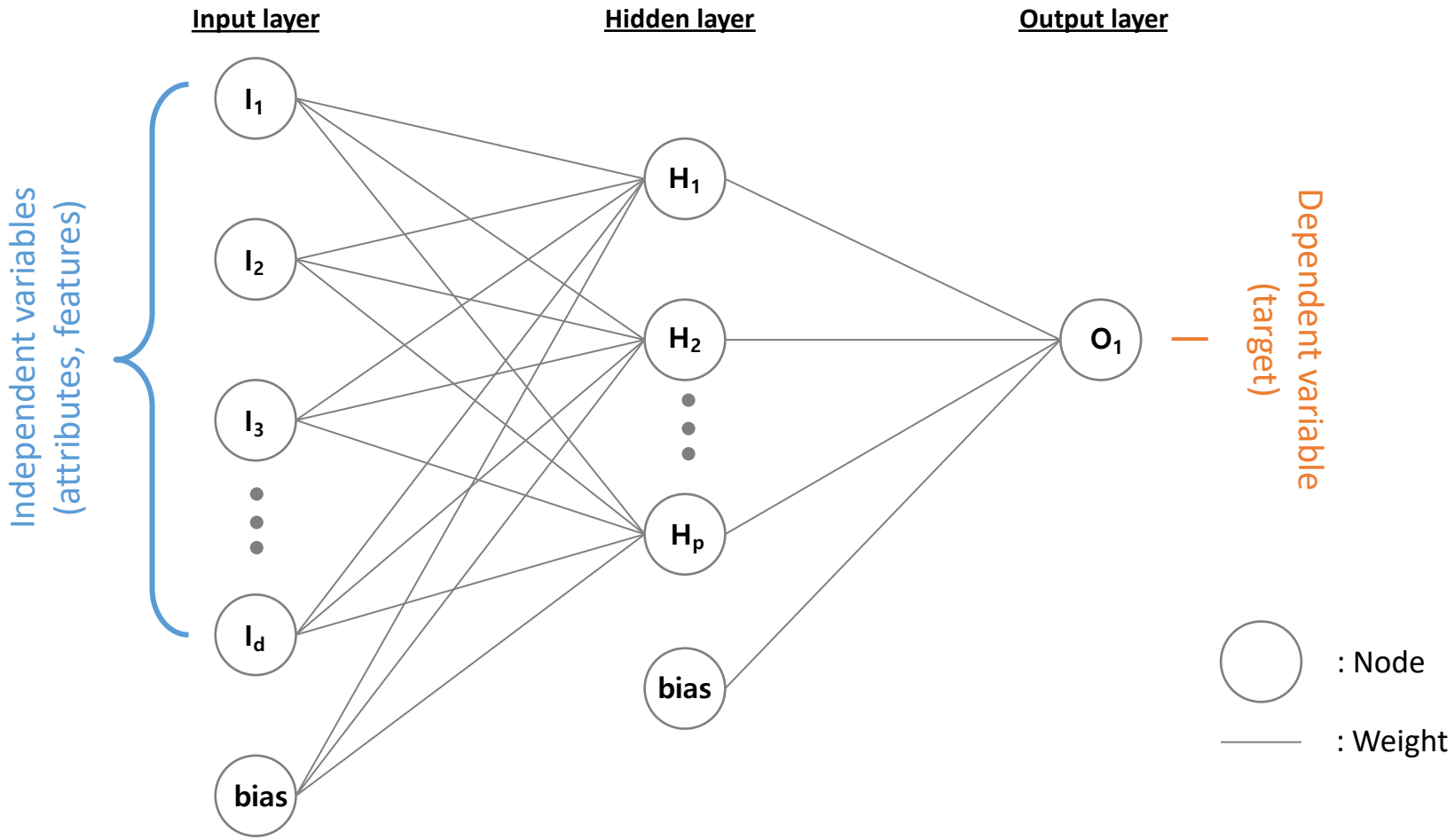
Source: Fei Fei Li's lecture notes

Object recognition

- **First way: Using hand-crafted feature extractor**
  - Intuitive
  - Its performance may be poor for unseen data which is more complex than training points.
- **Second way: Bag-of-words representation**
  - More complex patterns can be expressed / extracted than the first method.
  - It cannot completely satisfy
    - view-point invariance
    - scale invariance

Object recognition

- **Third way: Neural networks**
  - Feed-forward network?



Object recognition

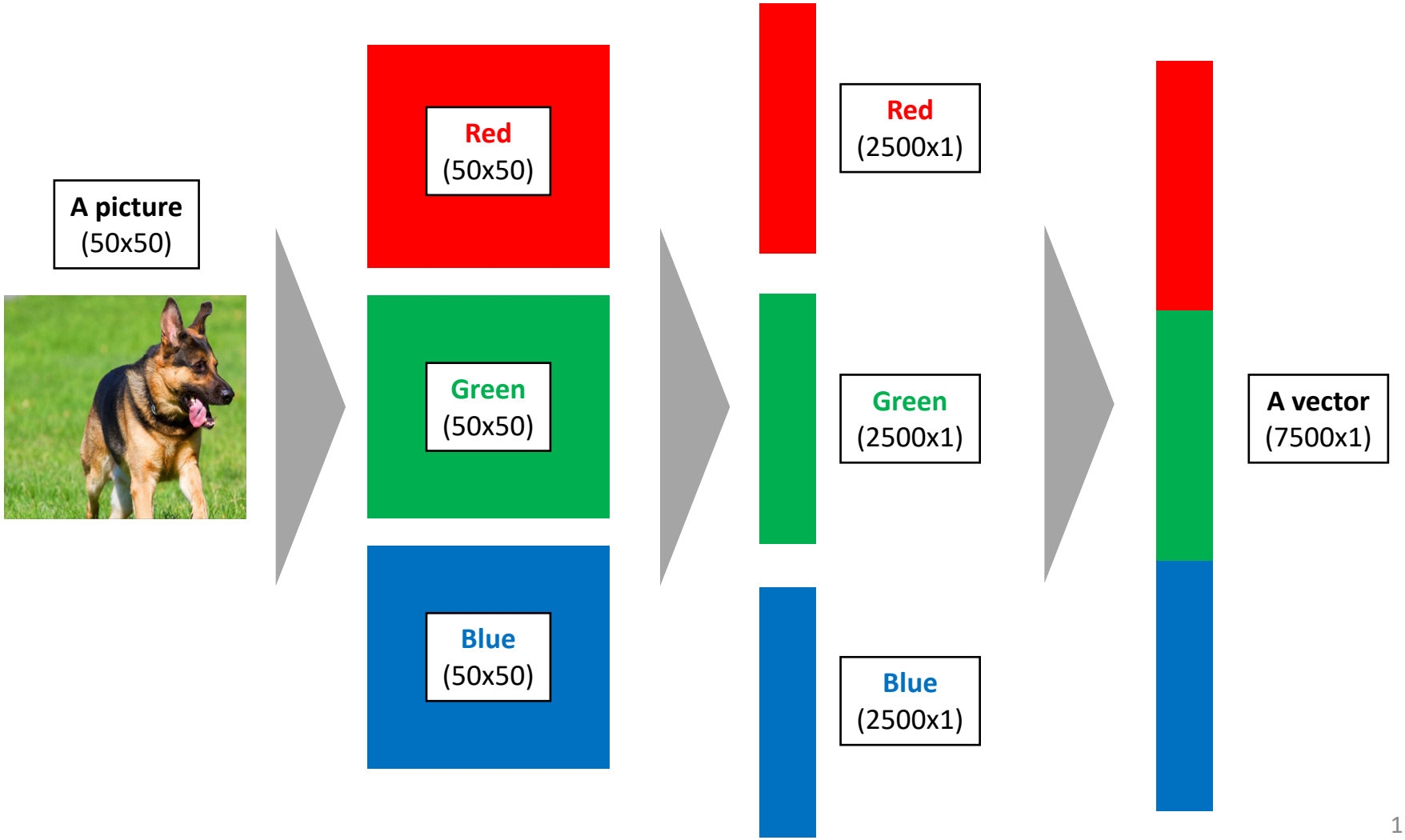
- **Feed-forward network를 사용하려면**
  - 이미지에서 각 픽셀 단위에서 RGB (Red, Green, Blue) 값을 추출한 후 이를 flat한 벡터로 변환해야 함

사진이 100 x 100 인 경우,
10,000 (단위) * 3 (색 혹은 채널)
= 30,000차원의 벡터를 생성

➔ Input layer의 차원이
30,000차원

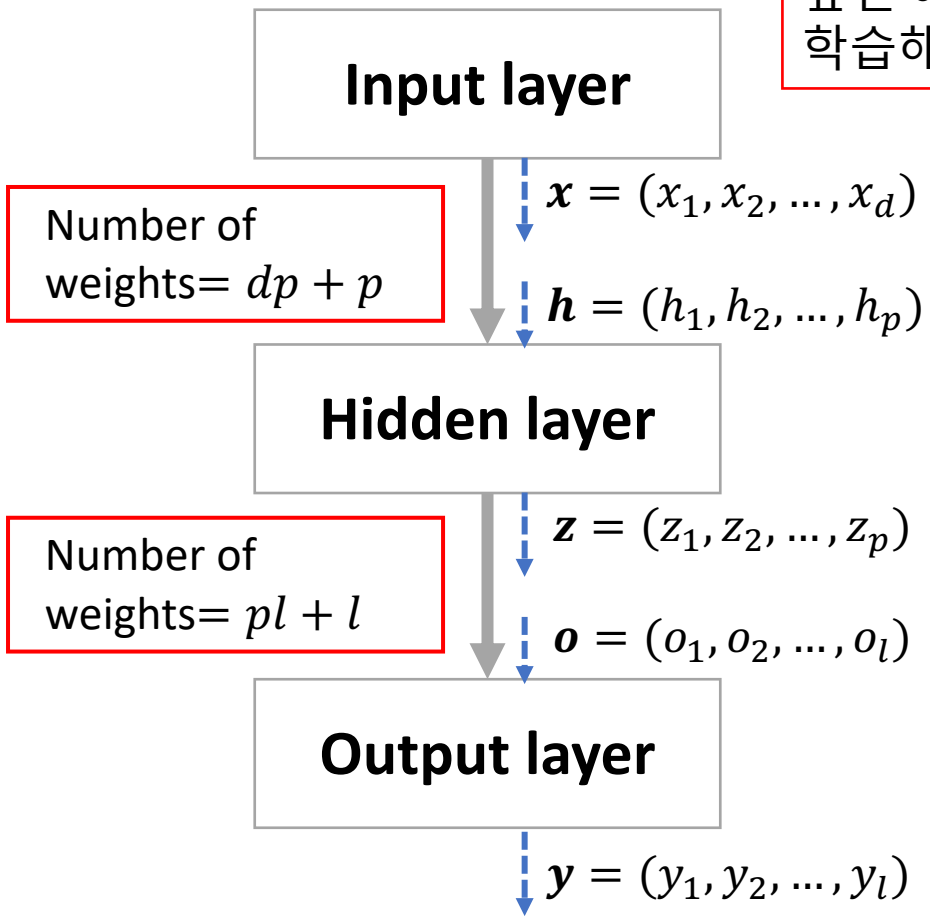
Features = Data representation

- **Example: Representing pictures**
  - A picture → 7500-dimensional vector (with minimum value 0 and maximum value 255)



Feed-forward network (revisited)

앞선 예제의 경우, 입력층이 30,000차원이면
학습해야 하는 가중치의 수가 너무 많아진다.



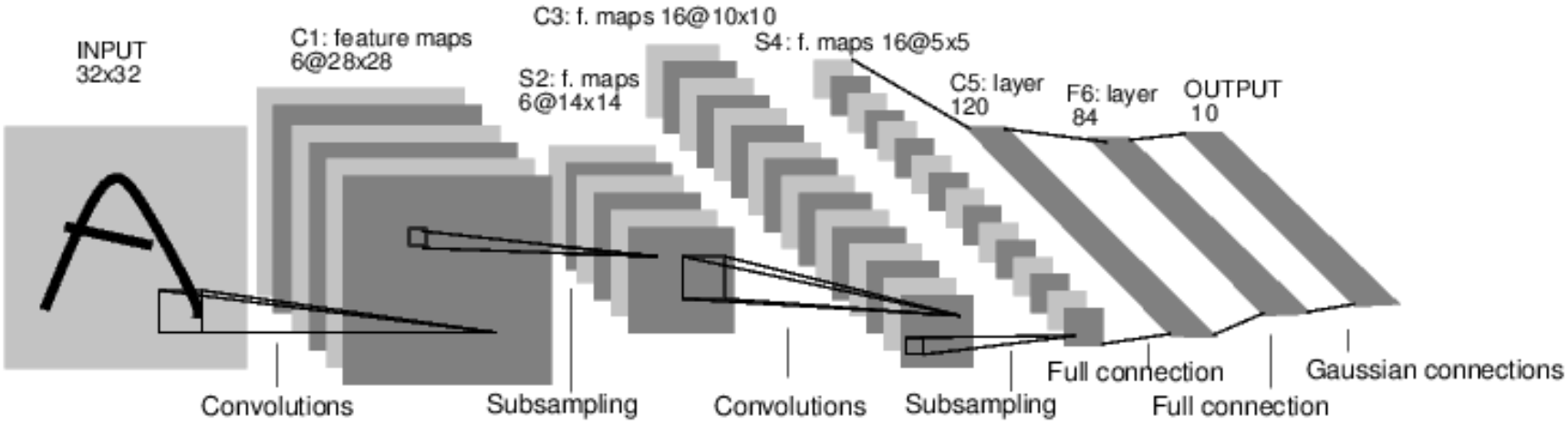
$$z_j = g(h_j) = g\left(\sum_{i=0}^{d} w_{ji}^{(1)} x_i\right)$$

$$y_k = f(o_k) = f\left(\sum_{j=0}^{p} w_{kj}^{(2)} z_j\right)$$

$$= f\left(\sum_{j=0}^{p} w_{kj}^{(2)} g\left(\sum_{i=0}^{d} w_{ji}^{(1)} x_i\right)\right)$$

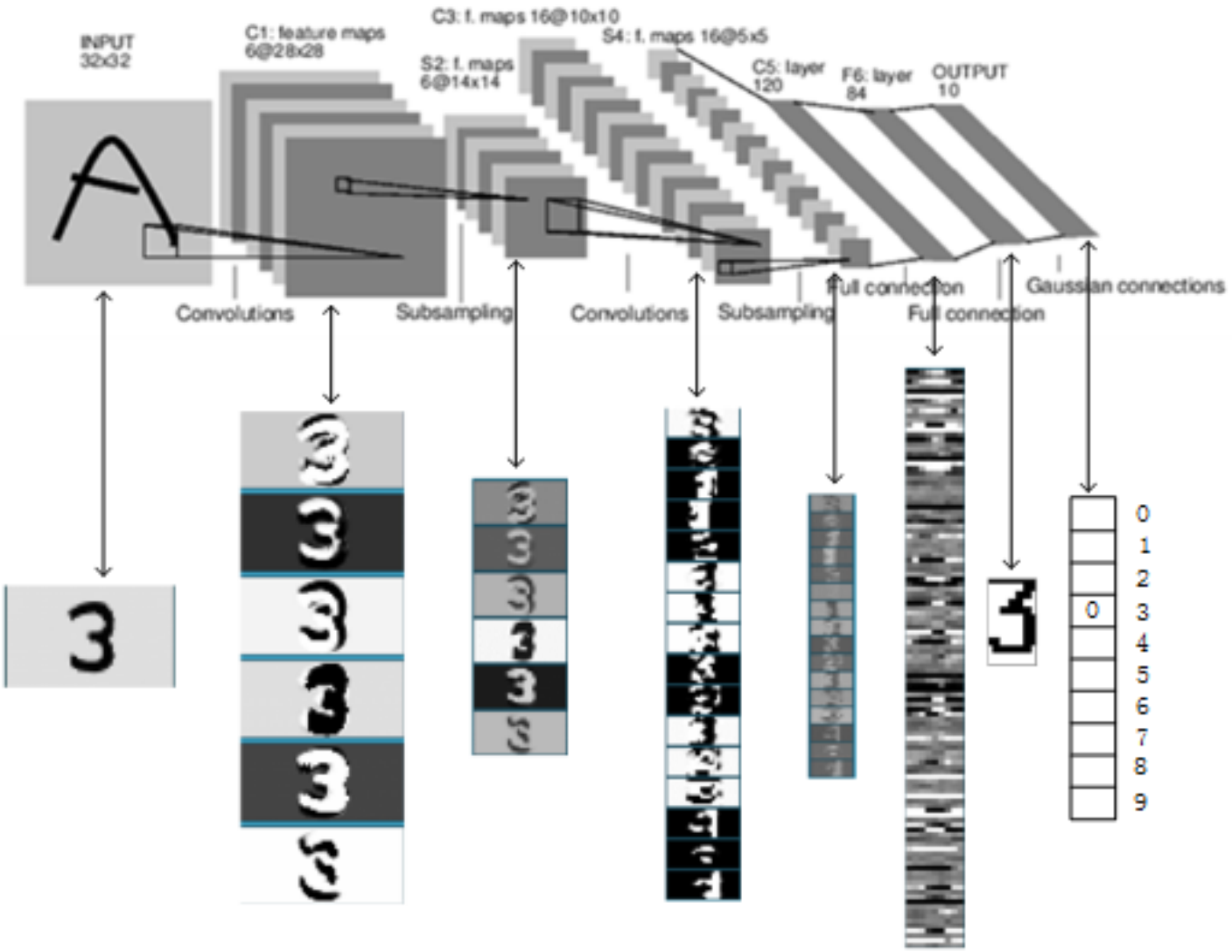
Object recognition

- **Third way: Neural networks**
  - **LeNet**
    - Proposed by **Yann LeCun**
    - Designed for hand-written and machine-printed character recognition
    - **The first successful applications of convolutional neural networks (CNNs)**
    - First version is proposed in 1989. ➔ LeNet-5 is proposed in 1998.



Structure of LeNet-5

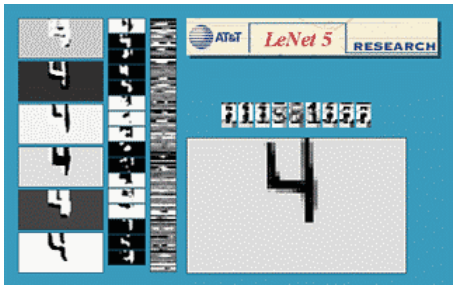
Process of LeNet-5 recognizing “3”

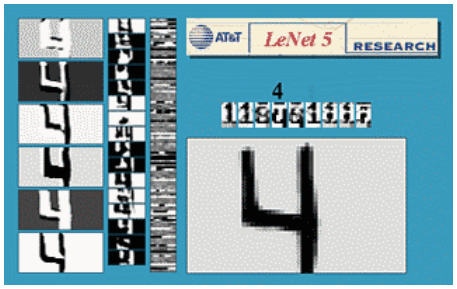


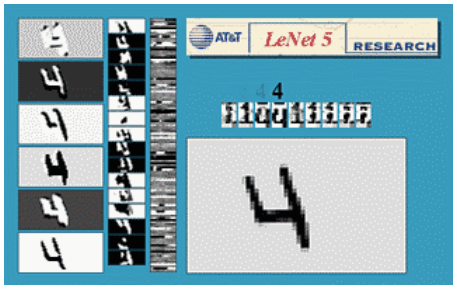
https://world4jason.gitbooks.io/research-log/content/deepLearning/CNN/Model%20&%20ImgNet/lenet.html

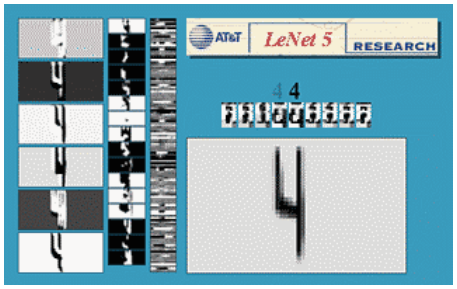
LeNet

- **LeNet is robust against the following variants.**
  - Shift
  - Scale
  - Rotation
  - Squeezing
  - Stroke width
  - …





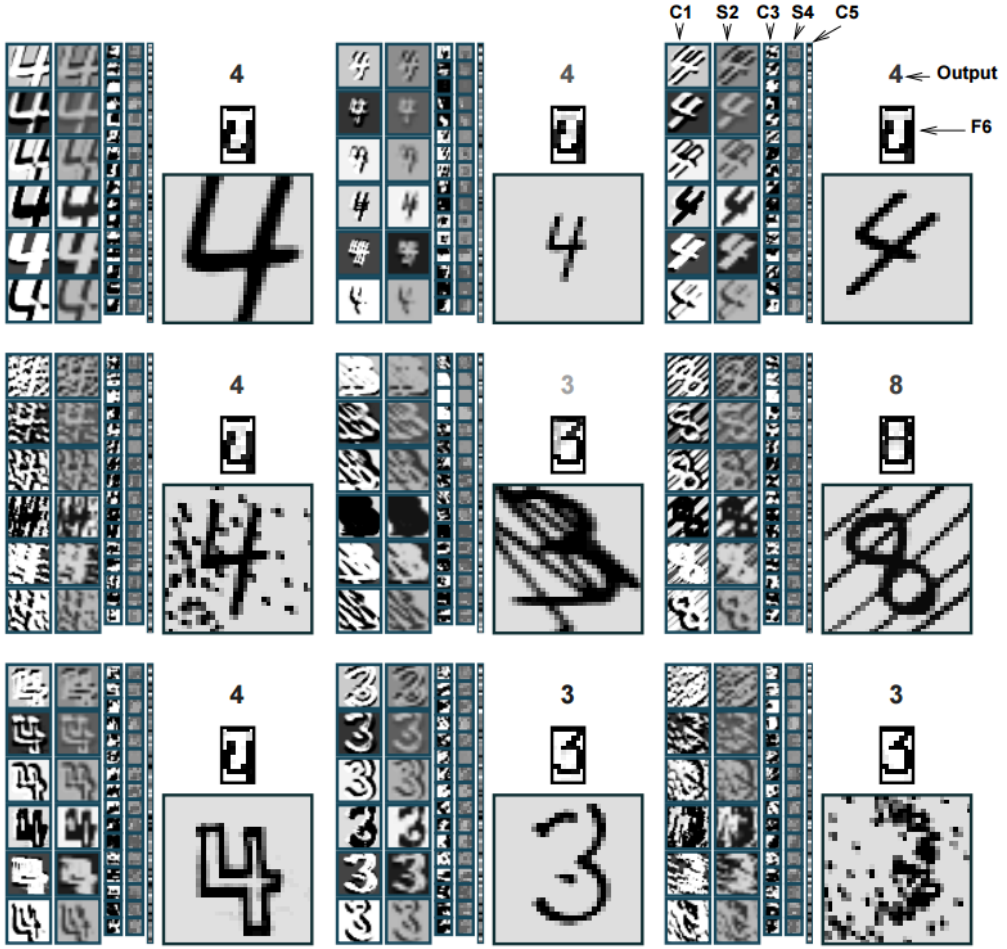




http://yann.lecun.com/exdb/lenet/index.html

LeNet

- **LeNet is robust against the following variants.**



LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

Structure of CNN

- **Convolution layer (+ ReLU)**
- **Pooling layer**
- **FC (Fully-connected)**

CNN Structures

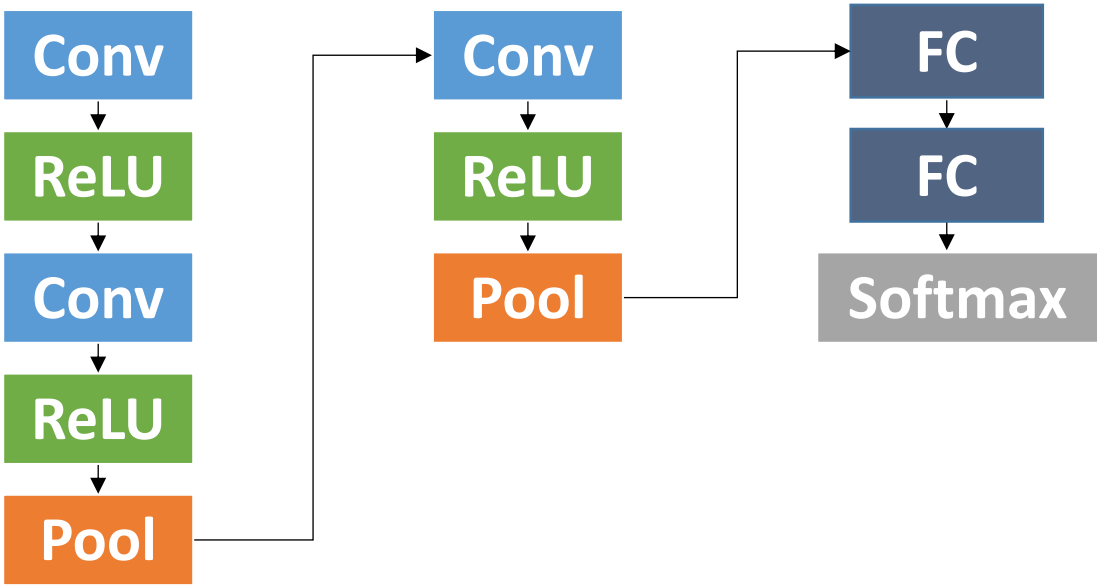
[(Conv → ReLU) * *k* → Pool] * *m*

→ (FC → ReLU) * *n* → Softmax

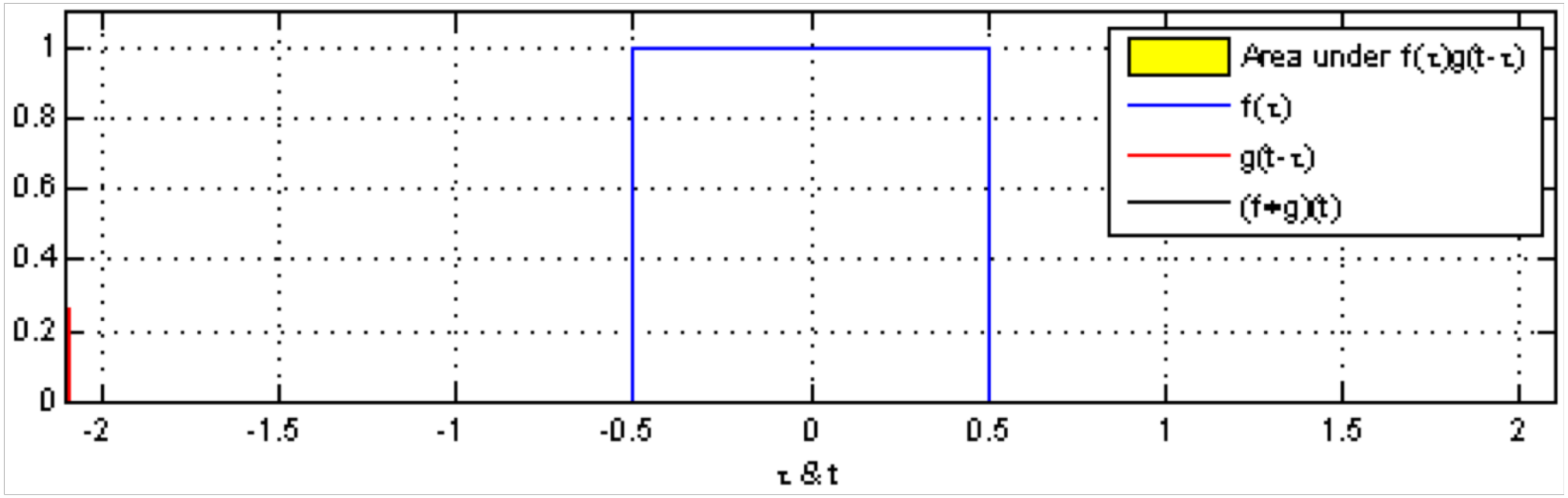
CNN Structures

[(Conv → ReLU) * *k* → Pool] * *m*

→ (FC → ReLU) * *n* → Softmax



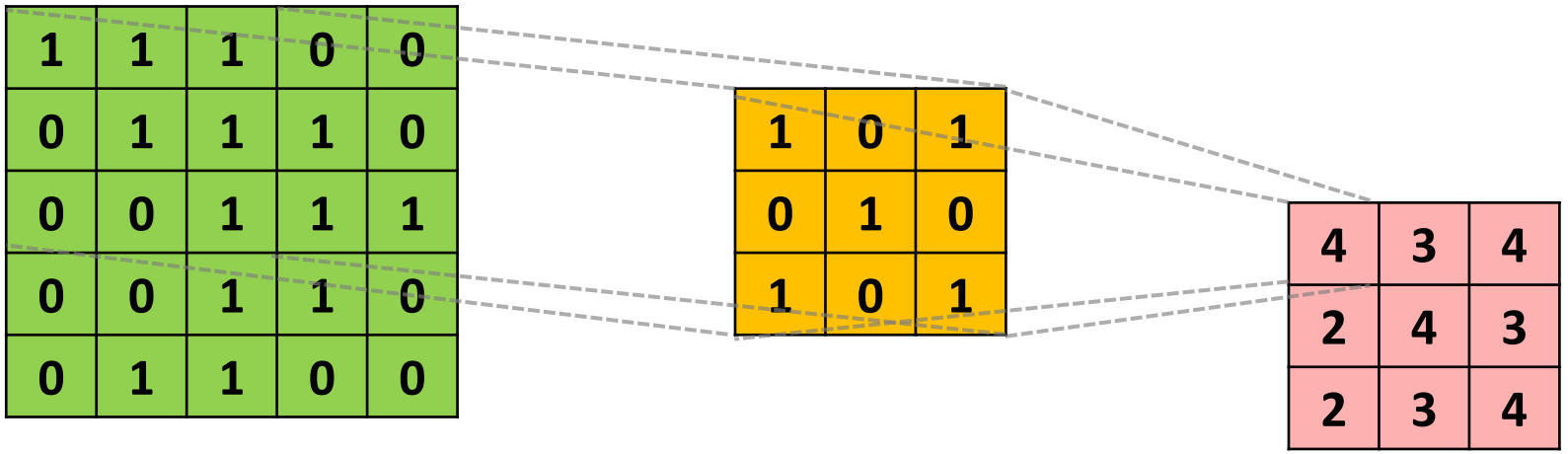
Convolution (합성곱) for 1-D



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

Convolution (합성곱) for 2-D

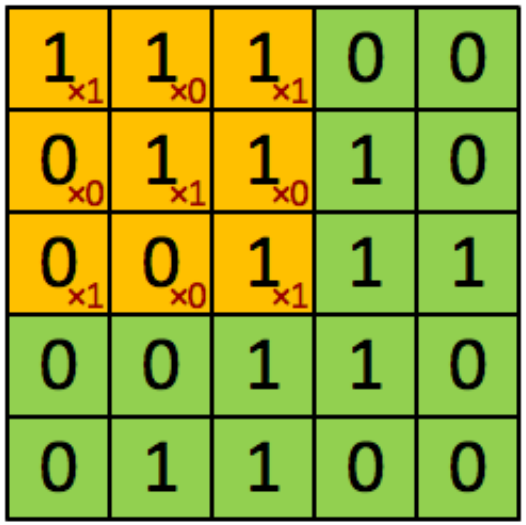
- **각 local 정보에 동일한 filter(or kernel)을 적용하여 값을 도출**
  - 5x5의 텐서 데이터에 3x3의 필터를 가로, 세로 1칸씩 움직이며 합성곱을 하면,

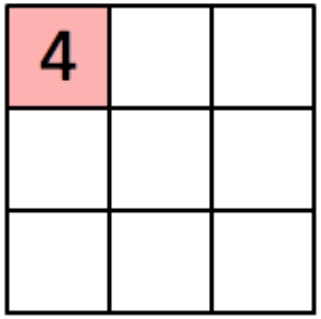


Tensor data with size 5x5x1 * **Filter (or kernel) with size 3x3x1** = **Convolved feature with size 3x3x1**

Convolution (합성곱) for 2-D

- **각 local 정보에 동일한 filter(or kernel)을 적용하여 값을 도출**
  - 5x5의 텐서 데이터에 3x3의 필터를 가로, 세로 1칸씩 움직이며 합성곱을 하면,

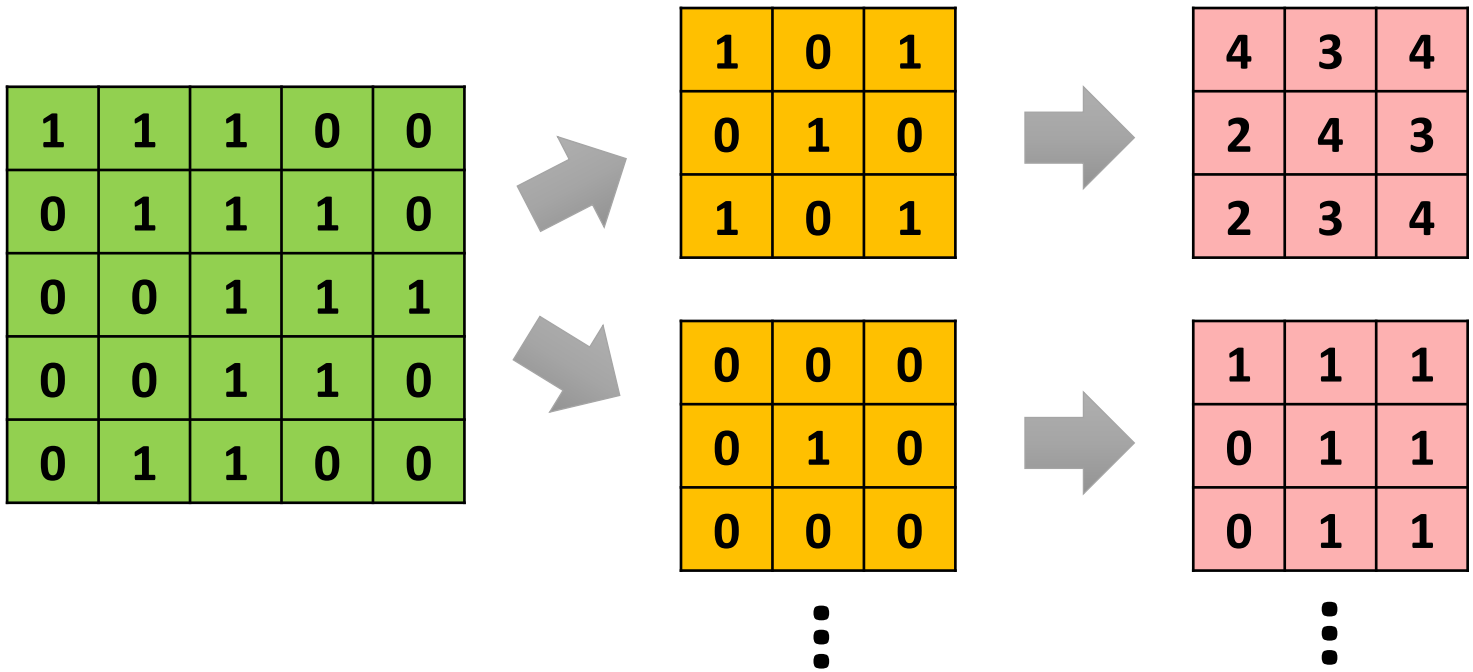




http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

Convolution (합성곱) for 2-D

- **여러 개의, 각기 다른 filter를 이용**
  - 5x5의 텐서 데이터에 3x3의 필터 *n*개를 가로, 세로 1칸씩 움직이며 합성곱을 하면,

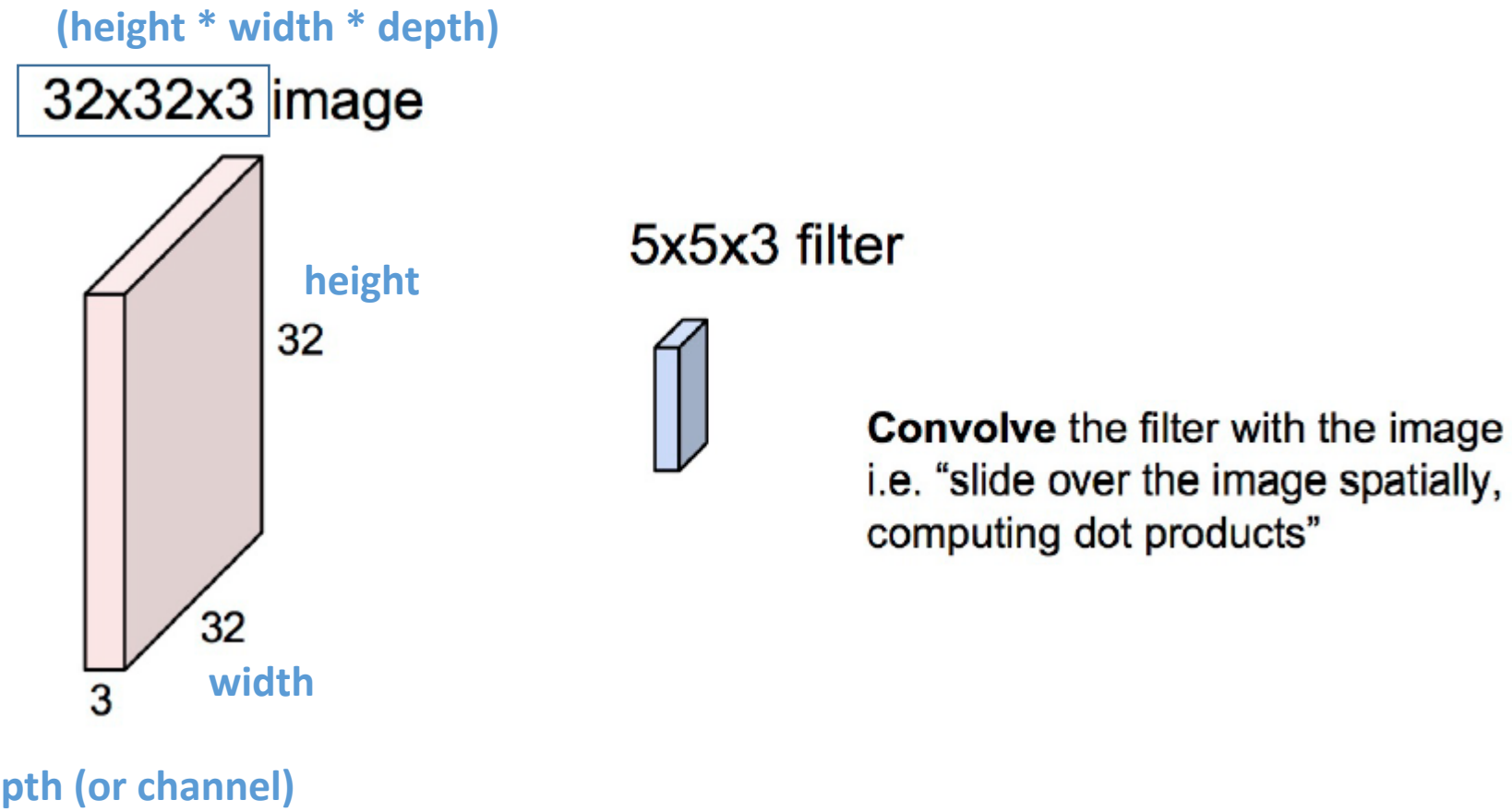


Tensor data with size 5x5x1 * **n filters (or kernels) with size 3x3x1** = **n convolved features with size 3x3x1**

5x5x1 tensor data → 3x3xn tensor data

Convolution for images

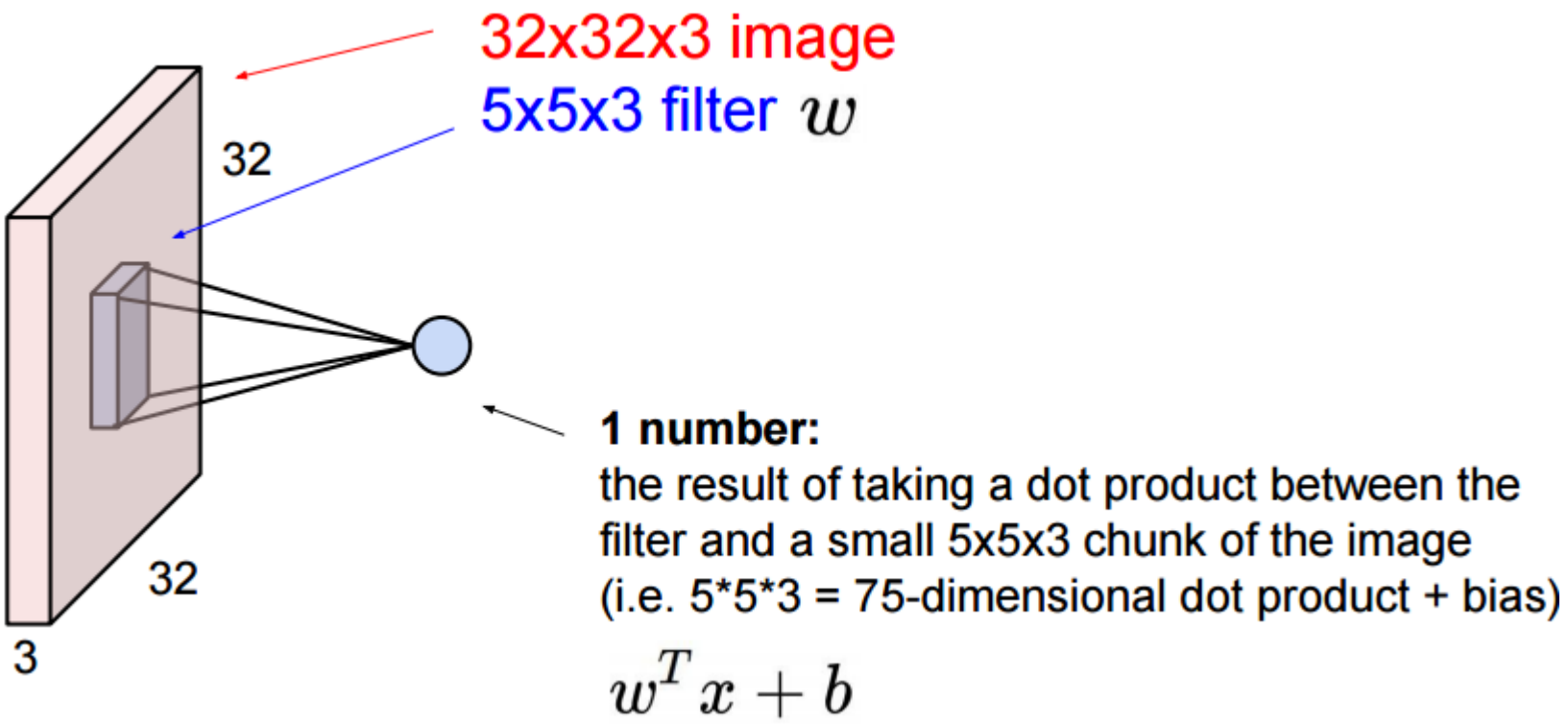
- 일반적인 image 데이터는 아래와 같이 32 x 32 pixel 에 3가지 색(Red / Green / Blue) channel 이 있다.
- 이 때, 특정 size 의 filter 가 전체 데이터를 돌면서 convolved feature 를 추출한다.



Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

Convolution for images

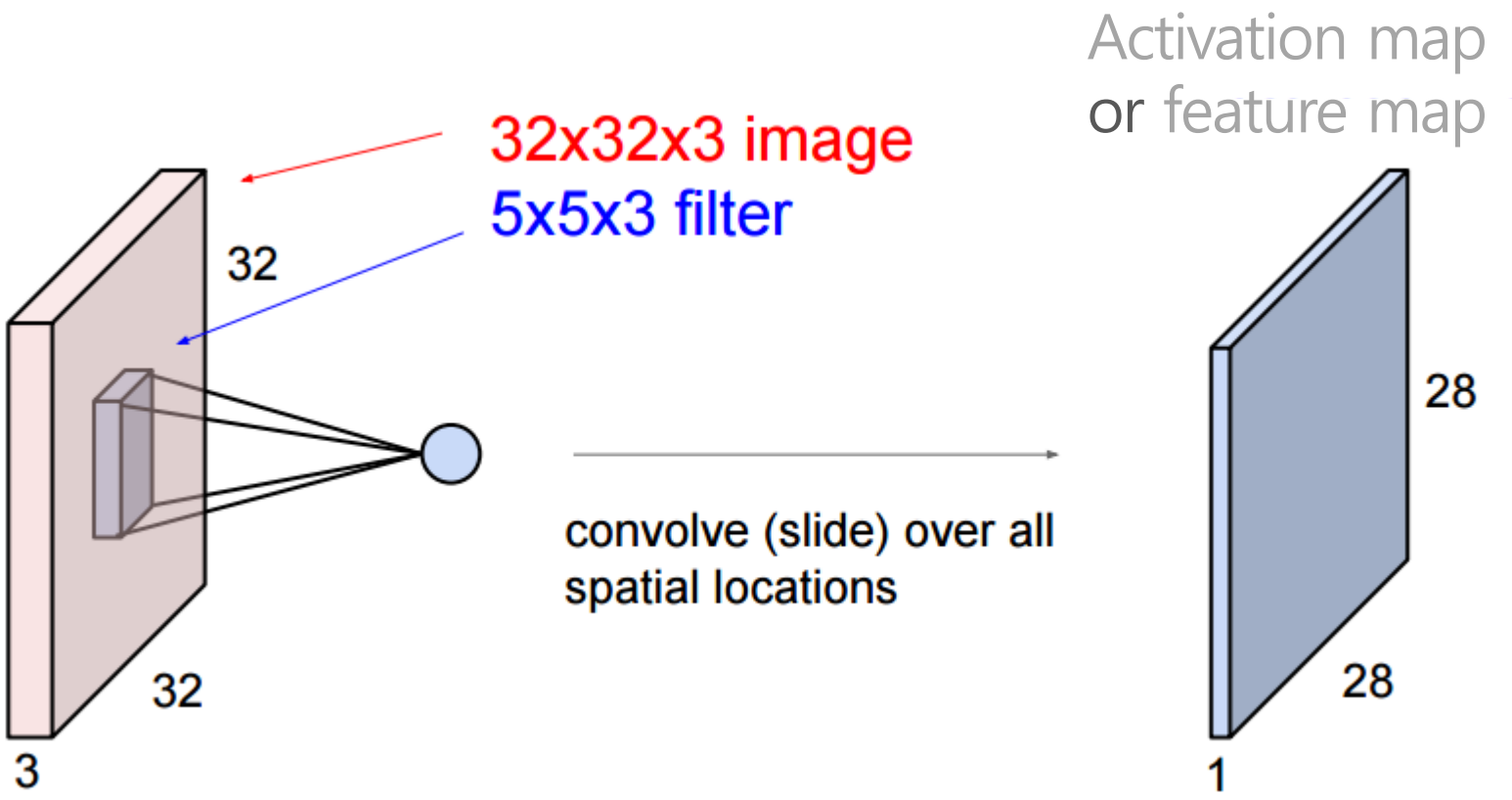
- 일반적인 image 데이터는 아래와 같이 32 x 32 pixel 에 3가지 색(Red / Green / Blue) channel 이 있다.
- 이 때, 특정 size 의 filter 가 전체 데이터를 돌면서 convolved feature 를 추출한다.



Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

Convolution for images

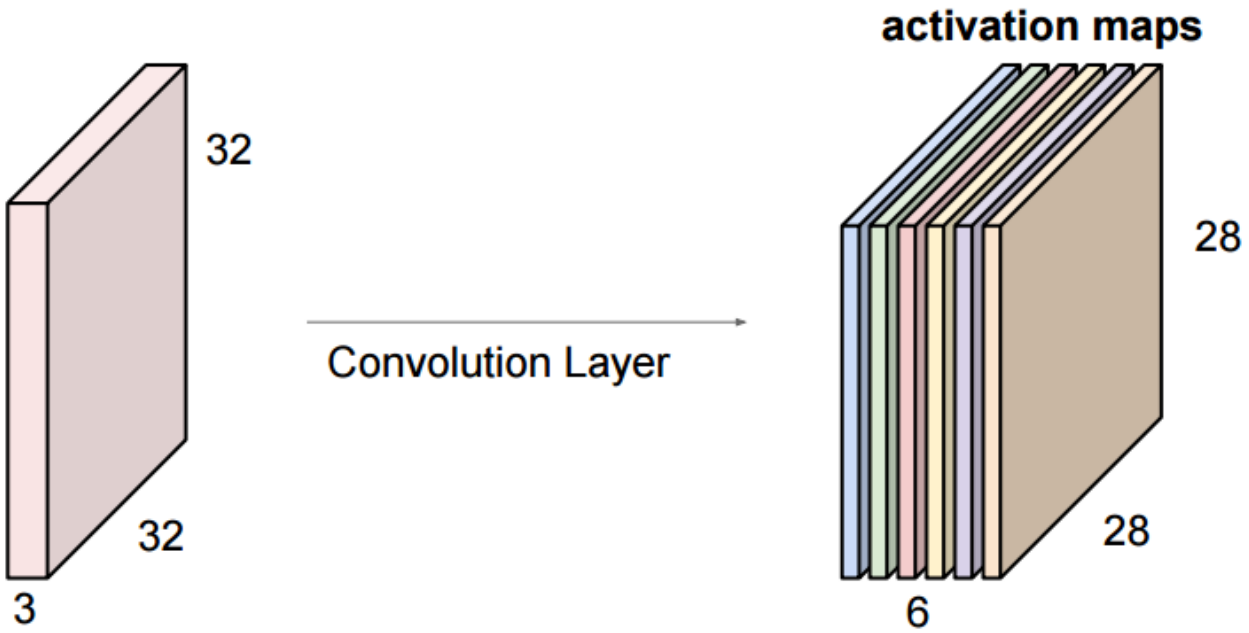
- 동일한 filter 를 사용해서 하나의 activation map 혹은 feature map 을 추출한다.



Convolution for images

- 서로 다른 5x5x3 filter 를 6개를 사용하면 아래와 같이 6개의 activation map 을 추출할 수 있다.
- 최종적으로 추출한 28 x 28 x 6을 새로운 convolutional layer 의 input 이 된다.

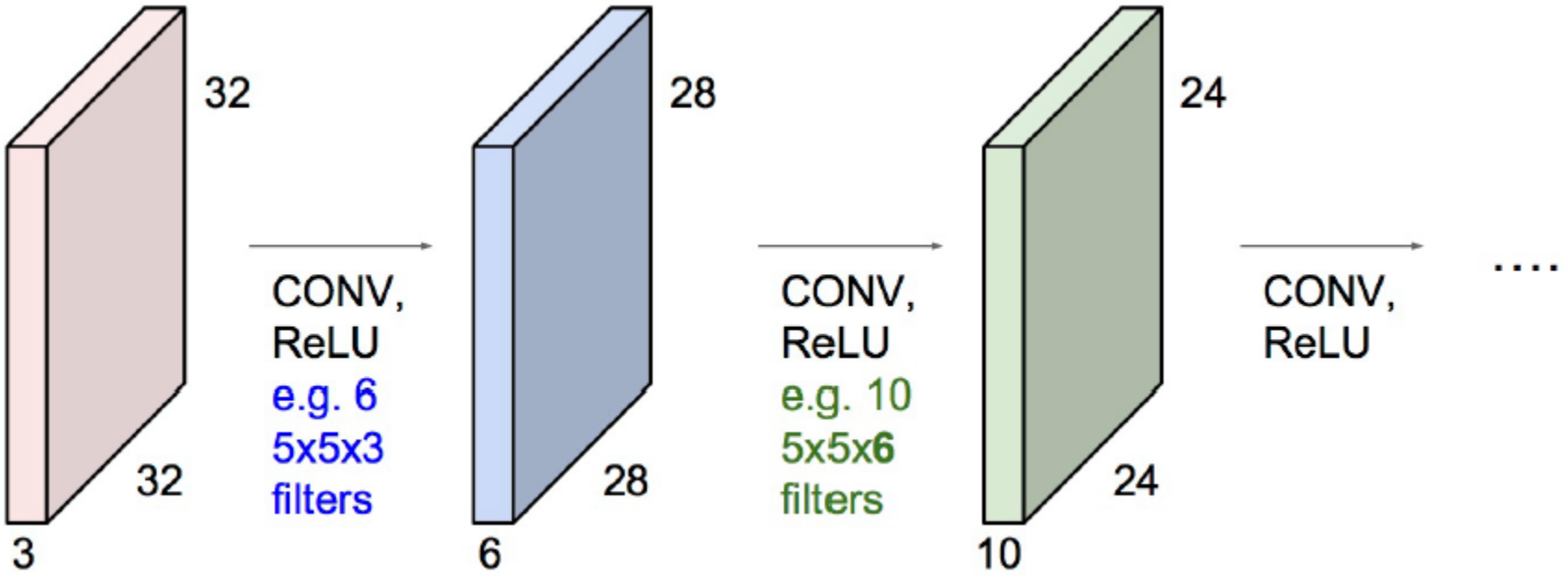
For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:



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

Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

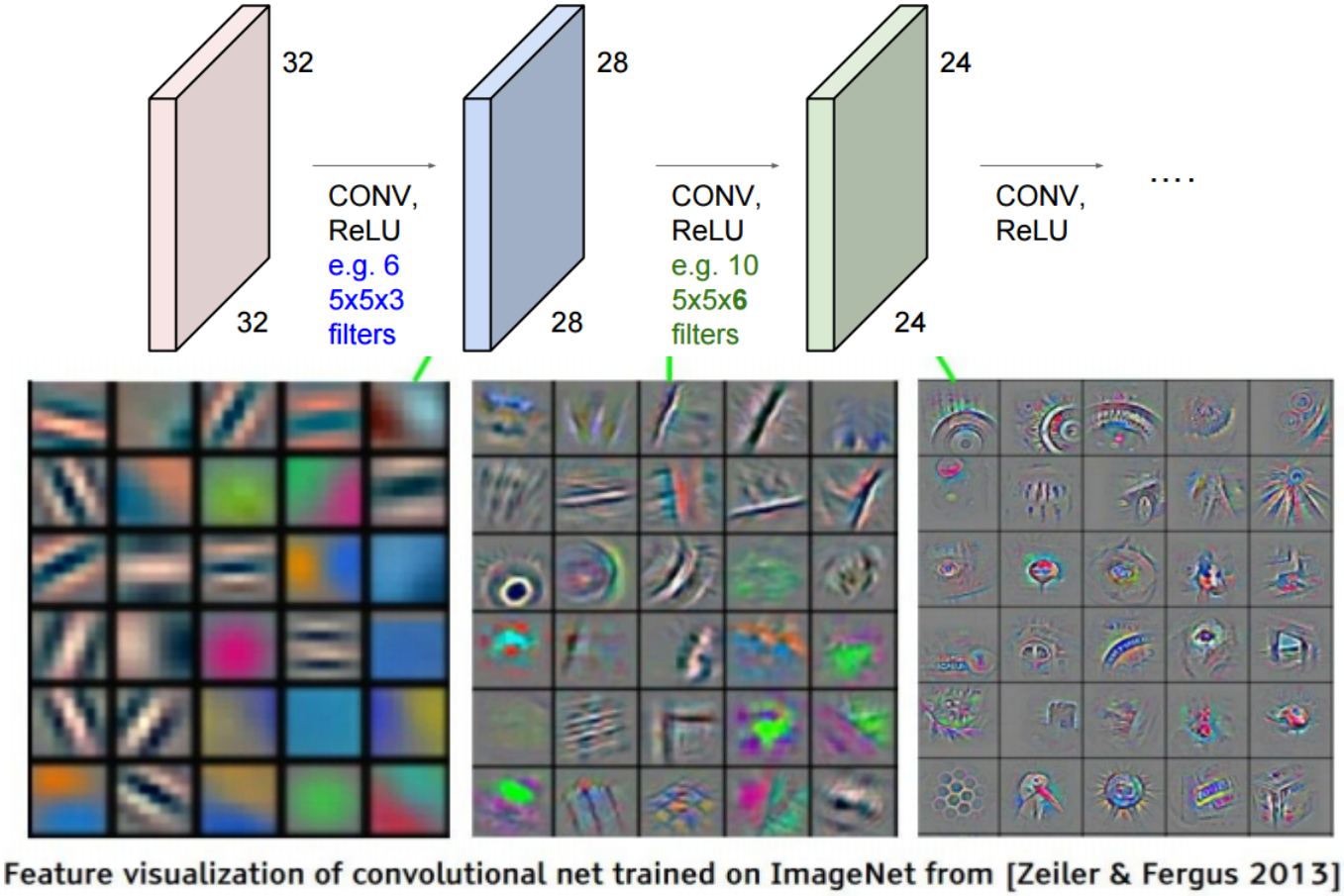
Conv → ReLU → Conv → ReLU → …



Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

Conv → ReLU → Conv → ReLU → …

- 초기에는 convolutional layer 에서는 매우 간단한 feature 만 추출하지만, convolution 단계를 거듭할 수록 복잡한 feature 를 추출할 수 있다.



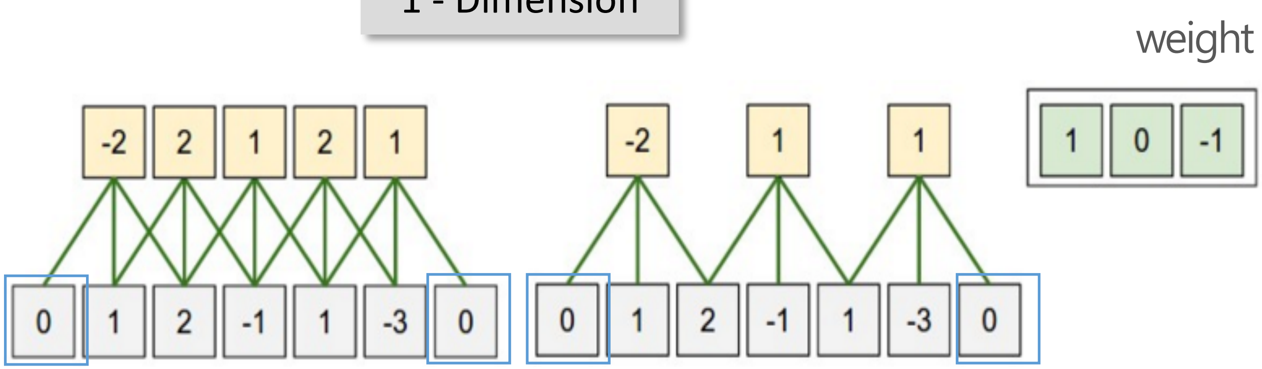
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

Hyper-parameters of convolution layer

- **Hyper-parameters (which should be defined by users)**
  - **Number of filters**: 얼마나 많은 필터를 사용하여 feature를 추출할 것인가
  - **Filter size**: 얼마나 local한 정보를 필터링할 것인가
  - **Stride**: 몇 걸음씩 이동할 것인가
  - **Zero padding**: border 정보를 유지 + output vector의 spatial size를 조절

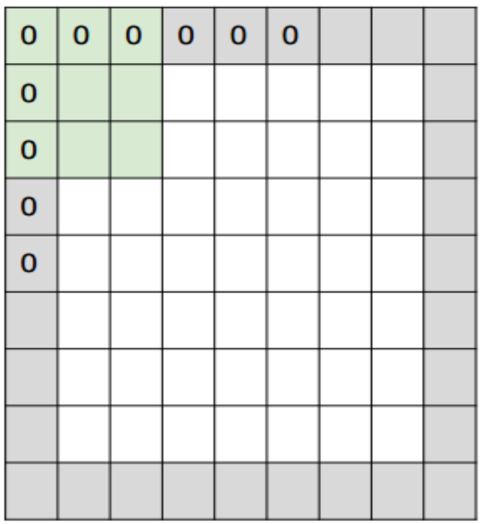
1 - Dimension



Region size : 3
Stride : 1
Zero padding : 1

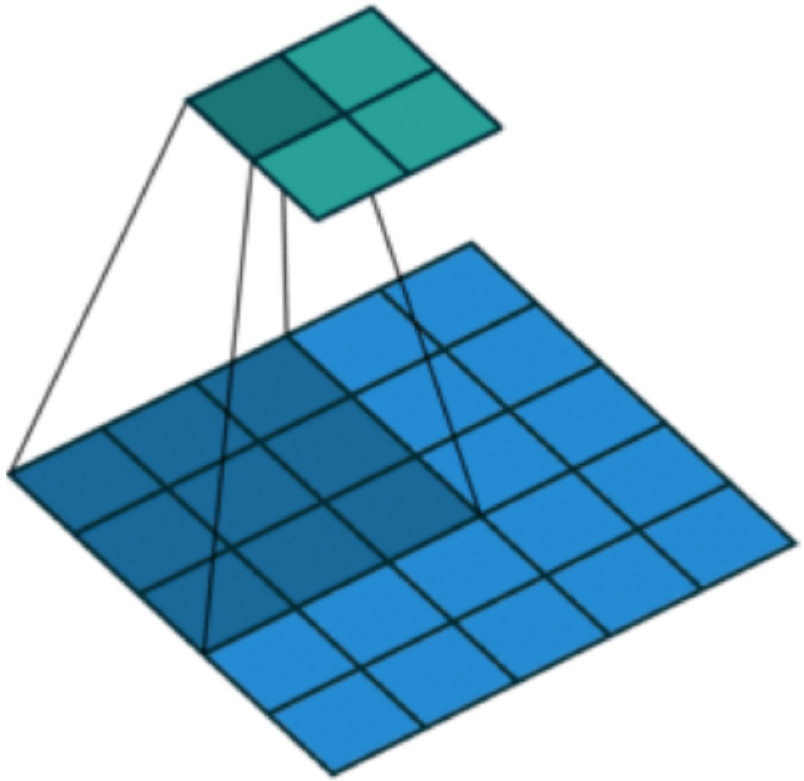
Region size : 3
Stride : 2
Zero padding : 1

2 - Dimension



Stride and zero-padding

- **Input: 5x5 ➔ W = 5**
- **Filter size: 3x3 ➔ F = 3**
- **Stride (S) = 2**
- **Zero-padding (P) = 0**



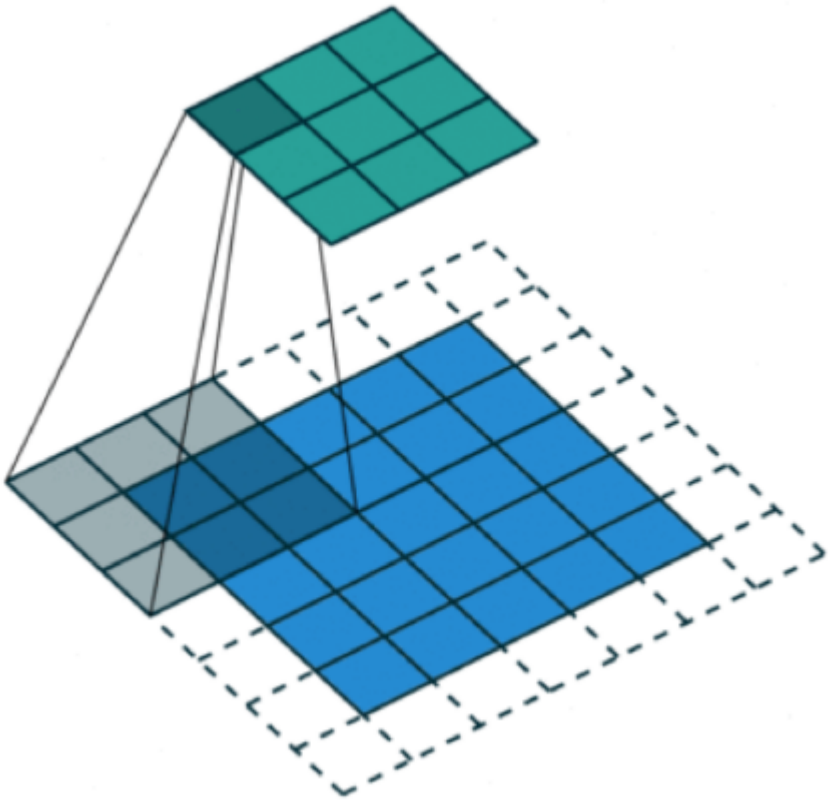
5x5x1 tensor

2x2x1 tensor

$$W' = \frac{W - F + 2P}{S} + 1 = \frac{5 - 3 + 2\times 0}{2} + 1 = 2$$

Stride and zero-padding

- **Input: 5x5**
- **Filter size: 3x3**
- **Stride = 2**
- **Zero-padding = 1**



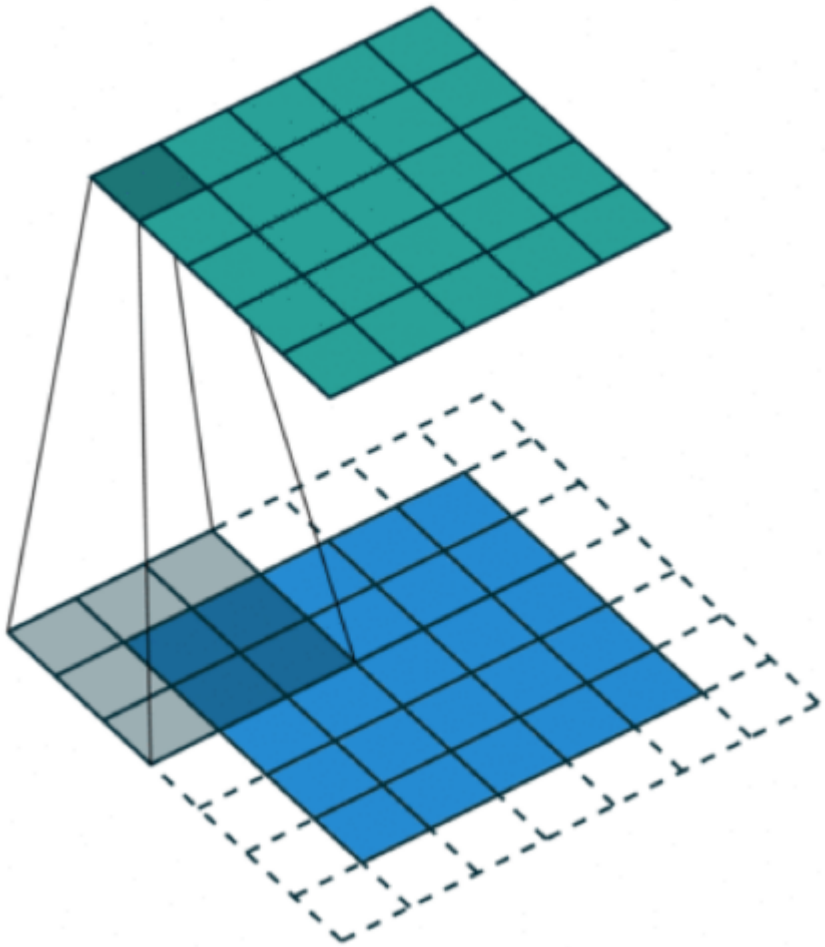
5x5x1 tensor

3x3x1 tensor

$$W' = \frac{W - F + 2P}{S} + 1 = \frac{5 - 3 + 2\times1}{2} + 1 = 3$$

Stride and zero-padding

- **Input: 5x5**
- **Filter size: 3x3**
- **Stride = 1**
- **Zero-padding = 1**



5x5x1 tensor

5x5x1 tensor

$$W' = \frac{W - F + 2P}{S} + 1 = \frac{5 - 3 + 2\times1}{1} + 1 = 5$$

Convolution layer

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

Convolution layer

Common settings:

K: Powers of 2 (e.g., 32, 64, 128, …)

- ✓ F=3, S=1, P=1
- ✓ F=5, S=1, P=2
- ✓ F=5, S=2, P=? (whatever fits)
- ✓ F=1, S=1, P=0

Summary. To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

Source : http://cs231n.stanford.edu/slides/winter1516_lecture7.pdf/

How to generate filters?

- **It's very simple!**
- **First, define hyper-parameters such as**
  - Number of filters, filter size,
  - Stride, zero-padding
- **And then,**
  - Initialize weights in filters
  - Train them via back-propagation.

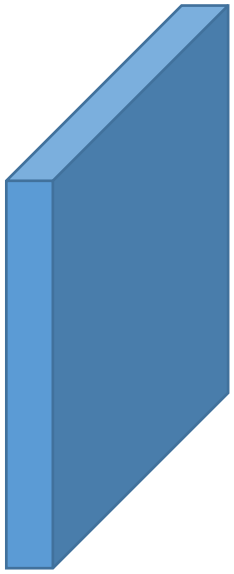
Convolution layer의 장점

동일한 filter를 각 tensor의 local에 똑같이 적용하므로 발생하는 장점들

- **Less parameters, high efficiency**
  - Weight sharing의 개념
  - (Fully-connected에 비하여) 매우 적은 가중치로도 다차원의 tensor를 학습
  - This helps preventing overfitting problem.
- **Local invariance**
  - The convolution filters are ‘sliding’ over the input image, the exact location of the object we want to find does not matter much.

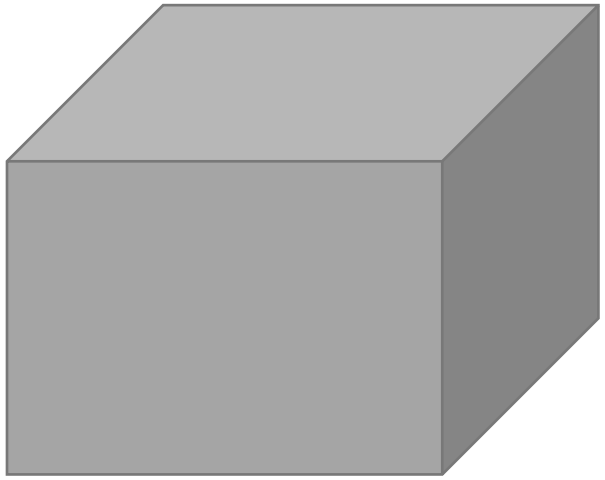
Number of parameters in convolution layer

[Input tensor(s)]



[Convolution filter(s)]

[Activation map(s)]



- ✓ filter: *(filter_height* x *filter_width* x *in_channels)* filters
- ✓ Number of filters: *out_channels*

for bias

Number of parameters in each filter: *filter_height* * *filter_width* * *in_channels* + 1

Total number of parameters: *out_channels* * (*filter_height* * *filter_width* * *in_channels* + 1)

We need activation function after Conv.

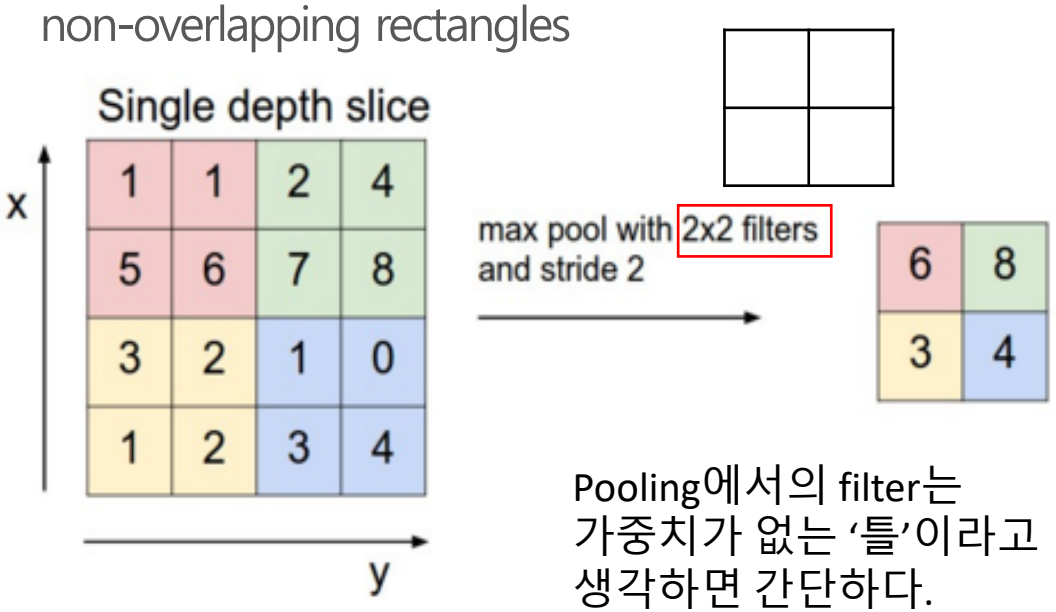
- **In many studies, a rectified linear unit (ReLU) is generally used for activating convolved features.**
- **Why?**
  - (Almost) the same reason as described in previous lectures

Structure of CNN

- Convolution layer (+ ReLU)
- **Pooling layer**
- **FC (Fully-connected)**

Pooling

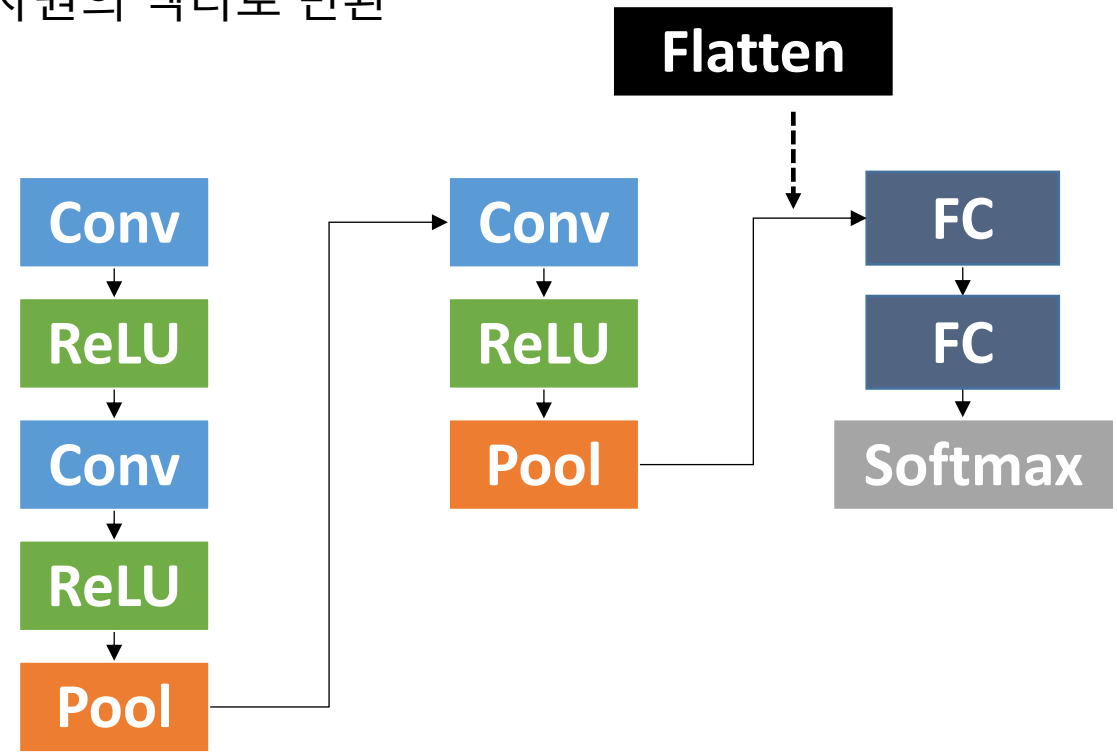
- **Pooling 은 유의미한 정보를 유지한 채 downsampling 기능을 함**
  - Intuition : 일단 feature 를 추출했으면, 정확한 위치는 중요하지 않고, 다른 feature 에 비해 상대적 정보가 중요하다.
  - Parameter 를 줄이기 위해 representation 의 size를 줄이는 기능을 함.
  - 이를 통해 overfitting을 줄여주는 역할
  - Types of pooling
    - Average pooling
    - L2-norm pooling
    - Max pooling



Pooling에서의 filter는 가중치가 없는 '틀'이라고 생각하면 간단하다.

Fully-connected

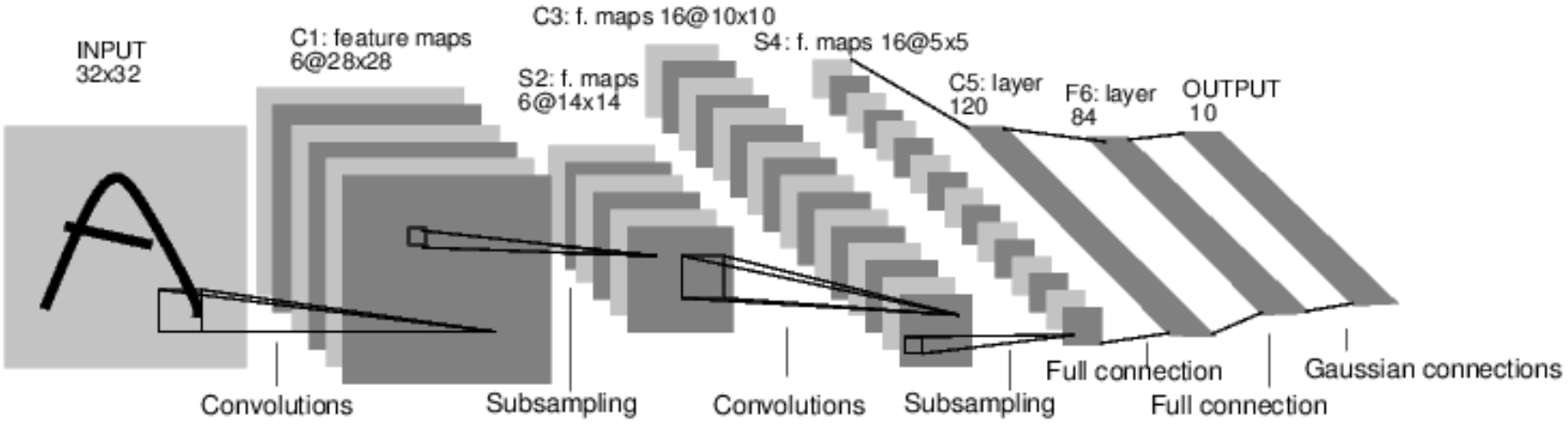
- **Output layer (일반적으로 softmax) 와 convolution / pooling layer를 연결**
- **Activation map을 평평하게 만든 후 (flatten), 일반적은 fully-connected network와 연결**
  - FC에 들어가기 전의 activation map 사이즈가 3x3x1,000 이라면, 이를 평평하게 펴서 9,000 차원의 벡터로 변환



Examples of CNNs

Convolutional neural network: Examples

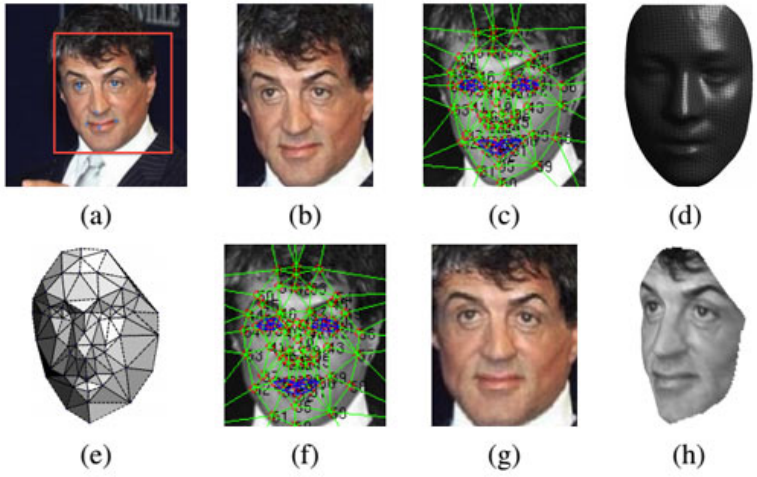
- **LeNet**
  - Proposed by Yann LeCun
  - CNN designed for hand-written and machine-printed character recognition
  - The first successful applications of CNN
  - First version is proposed in 1989.



Convolutional neural network: Examples

- **DeepFace**
  - Facebook's new AI research group reports a major improvement in face-processing software.
  - **Using Deep Learning**

Asked whether two unfamiliar photos of faces show the same person, a human being will get it right **97.53 percent** of the time. New software developed by researchers at Facebook can score **97.25 percent** on the same challenge, regardless of va riations in lighting or whether the person in the picture is directly facing the camer a.



(MIT Technology Review, Facebook Creates Software That Matches Faces Almost as Well as You Do, March 2014.)
http://www.technologyreview.com/news/525586/facebook-creates-software-that-matches-faces-almost-as-well-as-you-do/

Convolutional neural network: Examples

- **DeepFace**

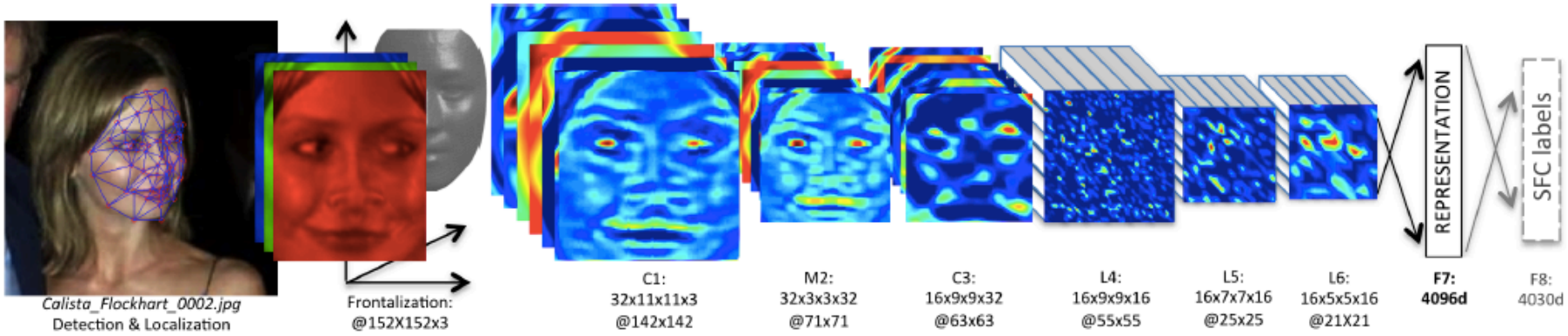


Figure 2. **Outline of the *DeepFace* architecture**. A front-end of a single convolution-pooling-convolution filtering on the rectified input, followed by three locally-connected layers and two fully-connected layers. Colors illustrate outputs for each layer. The net includes more than 120 million parameters, where more than 95% come from the local and fully connected layers.

(Facebooks' DeepFace facial recognition systems is one well-known deep learning application. Source: Facebook)

Convolutional neural network: Examples

- **ImageNet**
  - An image database designed for use in visual object recognition software research
  - http://image-net.org/
- **ILSVRC (ImageNet Large Scale Visual Recognition Challenge)**
  - an annual software contest by ImageNet project since 2006.
  - Object: Classifying and detecting objects and scenes correctly using ImageNet database.
  - ILSVRC 2016: http://image-net.org/challenges/LSVRC/2016/index

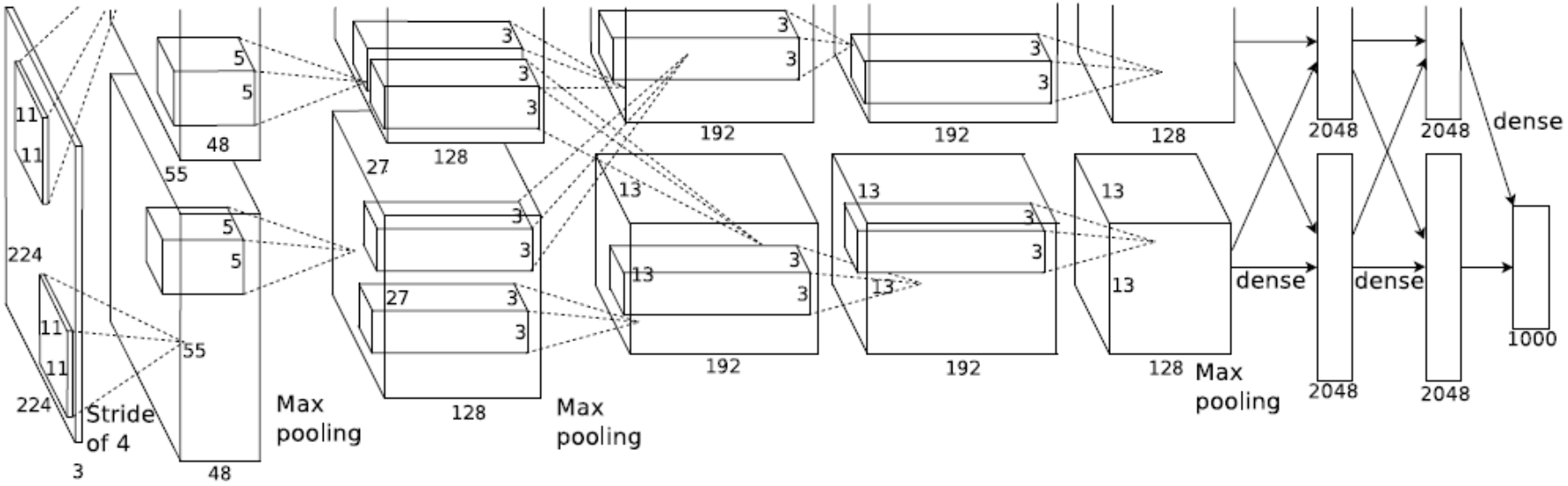
Convolutional neural network: Examples

- **AlexNet (Krizhevsky, Sutskever and Hinton): ILSVRC 2012 winner**
  - The first work that popularized CNN in computer vision
  - Significantly outperformed the second runner-up (top 5 error of 16% compared to runner-up with 26% error)
  - A similar architecture basic as LeNet, but deeper, bigger than it
  - Convolutional Layers stacked on top of each other

from: Prof. Sungroh Yoon's lecture note

AlexNet

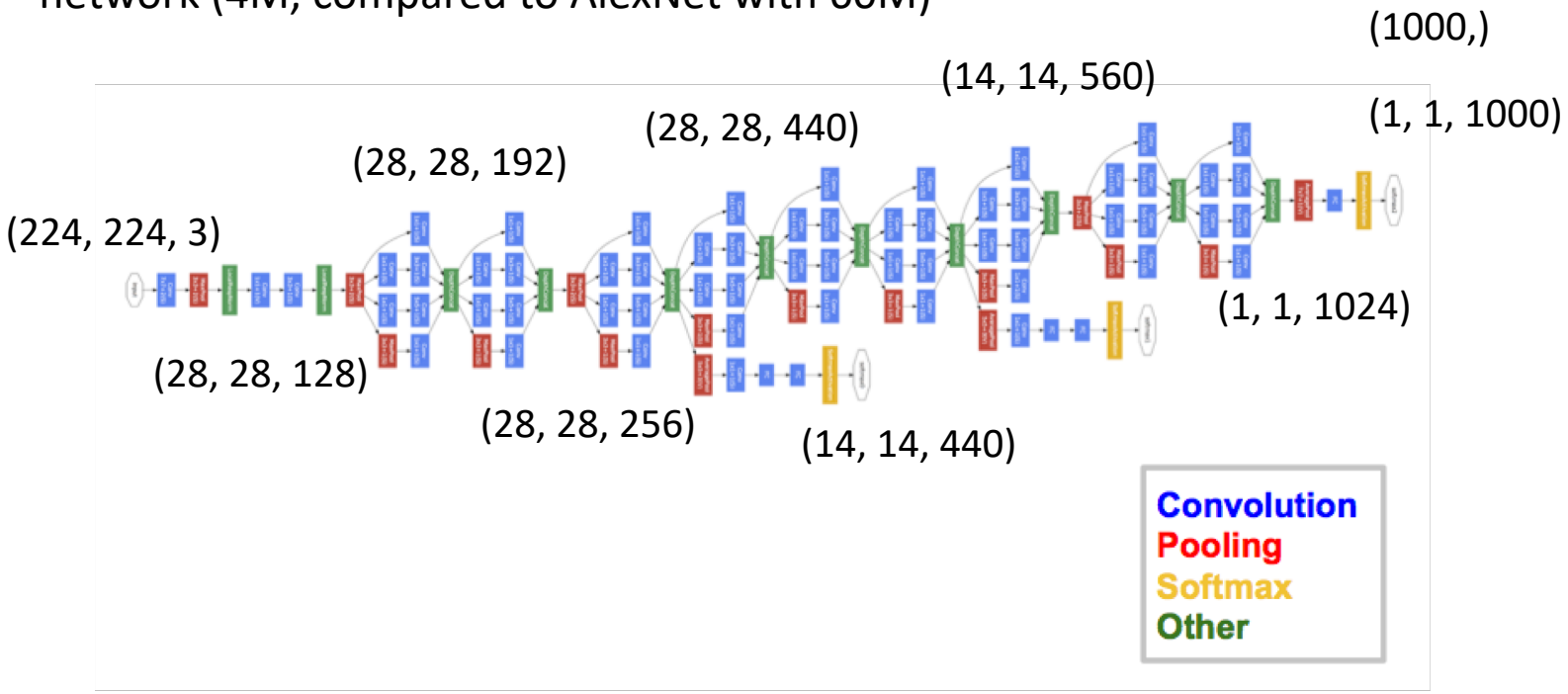
- 여러 개의 convolution / pooling 단계를 거치고 최종적으로 fully connected 된 layer 를 거침



2012 NIPS; ImageNet Classification with Deep Convolutional Neural Networks; Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

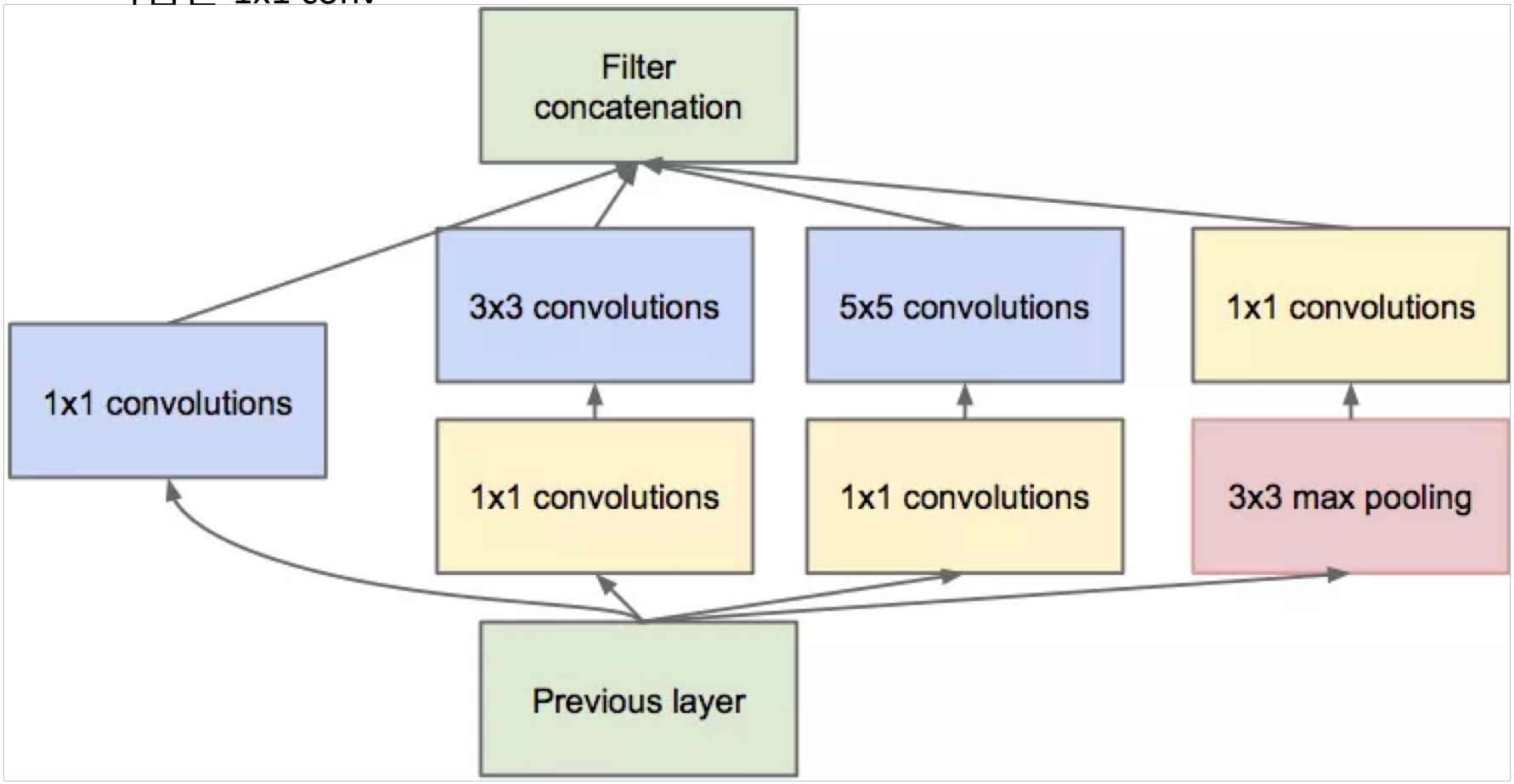
Convolutional neural network: Examples

- **ZF Net (Zeiler and Fergus): ILSVRC 2013 winner**
  - an improvement on AlexNet by tweaking the architecture hyperparameters
  - in particular by expanding the size of the middle convolutional layers
- **GoogLeNet (Szegedy et al.): ILSVRC 2014 winner**
  - has an Inception Module that dramatically reduced the number of parameters in the network (4M, compared to AlexNet with 60M)



Convolutional neural network: Examples

- **GoogLeNet: Inception block**
  - 핵심은 1x1 conv



https://learningcarrot.wordpress.com/2015/11/14/how-does-googlenet-need-only-small-number-of-parameters/

Convolutional neural network: Examples

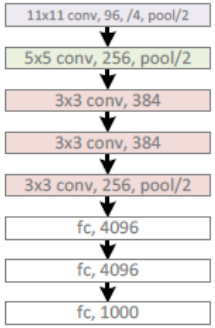
- **VGGNet (Simonyan and Zisserman): runner-up in ILSVRC 2014**
  - showed that the depth of the network is a critical component for good performance (contains 16 CONV/FC layers)
  - features an extremely homogeneous architecture that only performs 3x3 convolutions and 2x2 pooling from the beginning to the end
  - despite its slightly weaker classification performance, VGGNet features outperform those of GoogLeNet in multiple transfer learning tasks
  - currently the most preferred choice in the community when extracting CNN features from images

Convolutional neural network: Examples

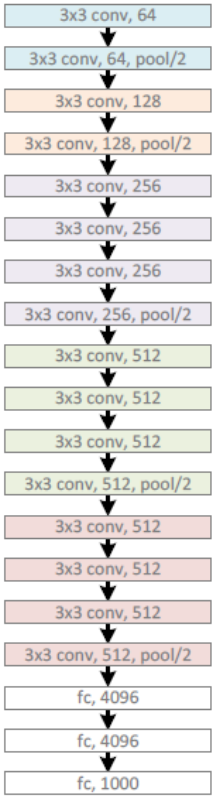
- **8 (AlexNet) → 19 (VGG) → 22 (GoogLeNet)**

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)



VGG, 19 layers
(ILSVRC 2014)



GoogleNet, 22 layers
(ILSVRC 2014)

ICCV15

Convolutional neural network: Examples

- **Deep residual network (ResNet)**

  : ILSVRC and COCO 2015 winner

  - He, Zhang, Ren & Sun

    (Microsoft Research Asia)

  - 1st places in all 5 main tracks

    – ImageNet classification

    – ImageNet detection: 16% better than 2nd

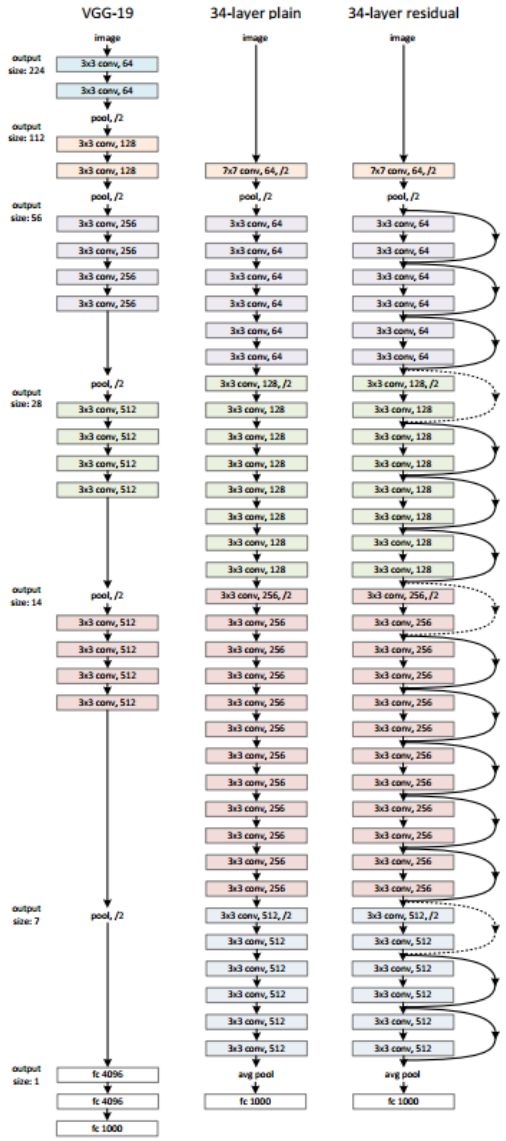
    – ImageNet localization: 27% better

    – COCO detection: 11% better

    – COCO segmentation: 12 % better

  - Revolution depth: 152 layers

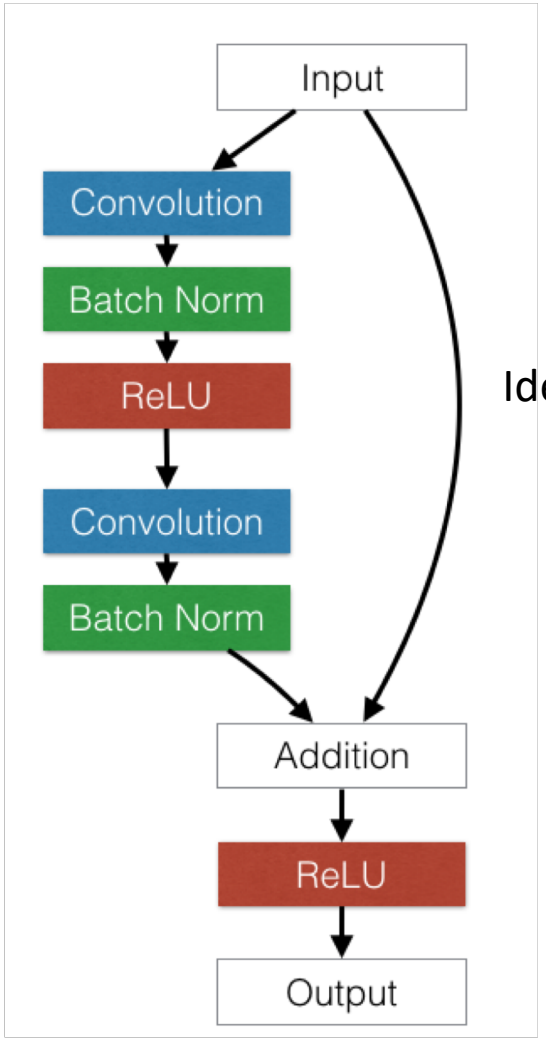
    (VGG is 19 layers)



He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

Convolutional neural network: Examples

- **ResNet: residual block**



Identity function

- Highway
- Short-cut
- Skip connection

http://torch.ch/blog/2016/02/04/resnets.html