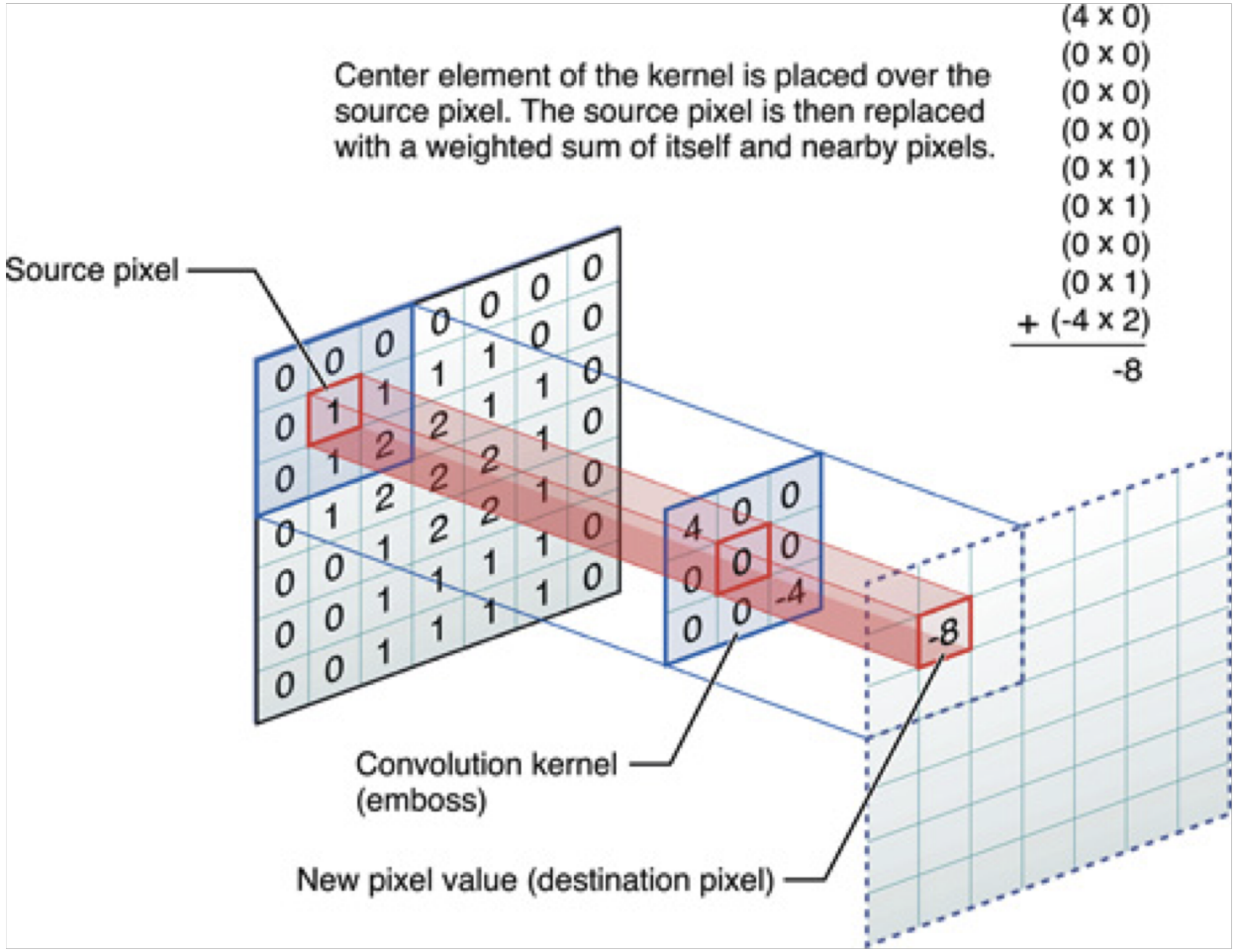
Published as a conference paper at ICLR 2015

VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* **& Andrew Zisserman+**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

Convolution Neural Net (CNN)

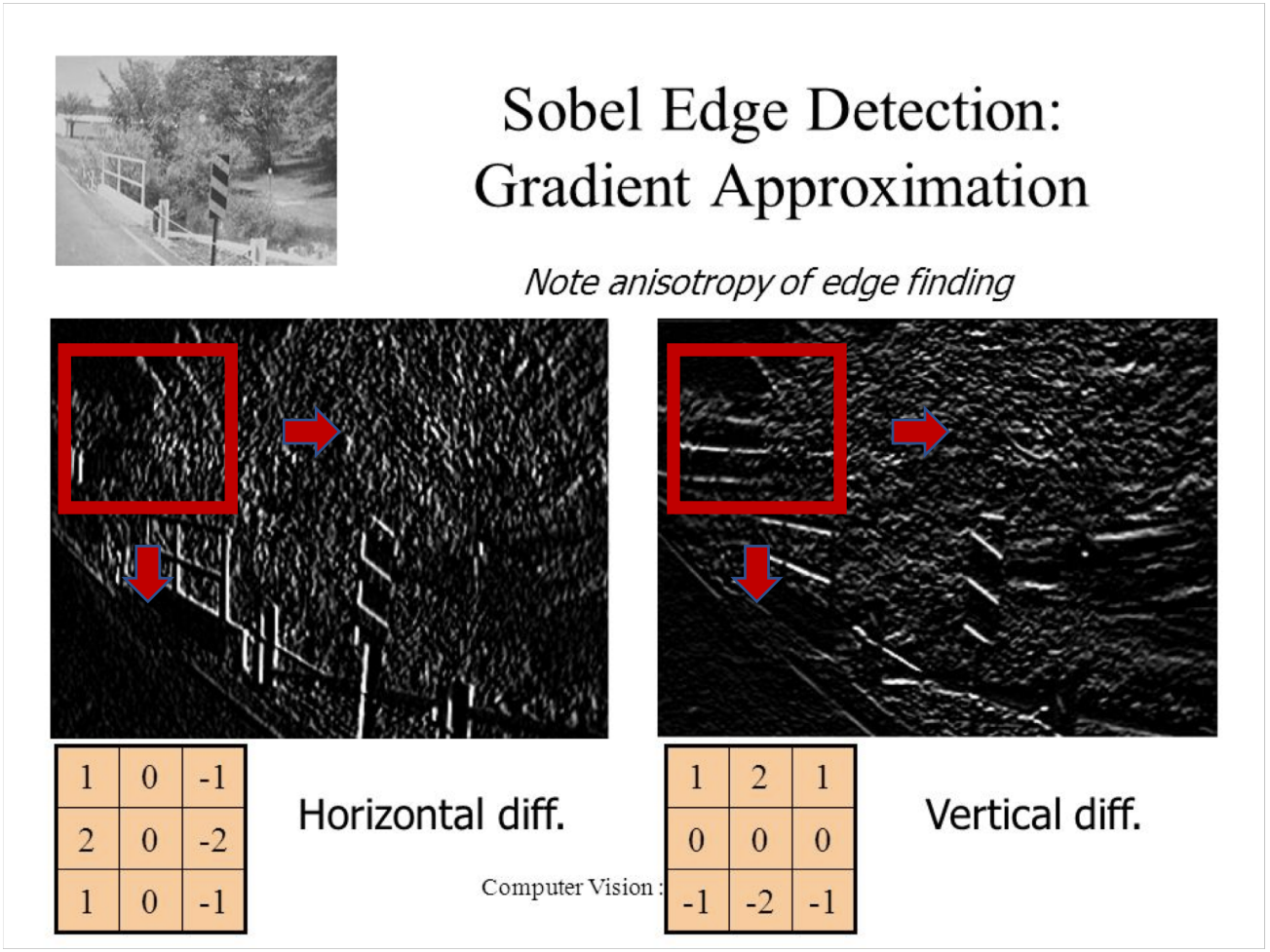
Convolution in computer vision



이미지 출처 : https://developer.apple.com/library/mac/documentation/Performance/Conceptual/vImage/Art/kernel_convolution.jpg

Convolution Neural Net (CNN)

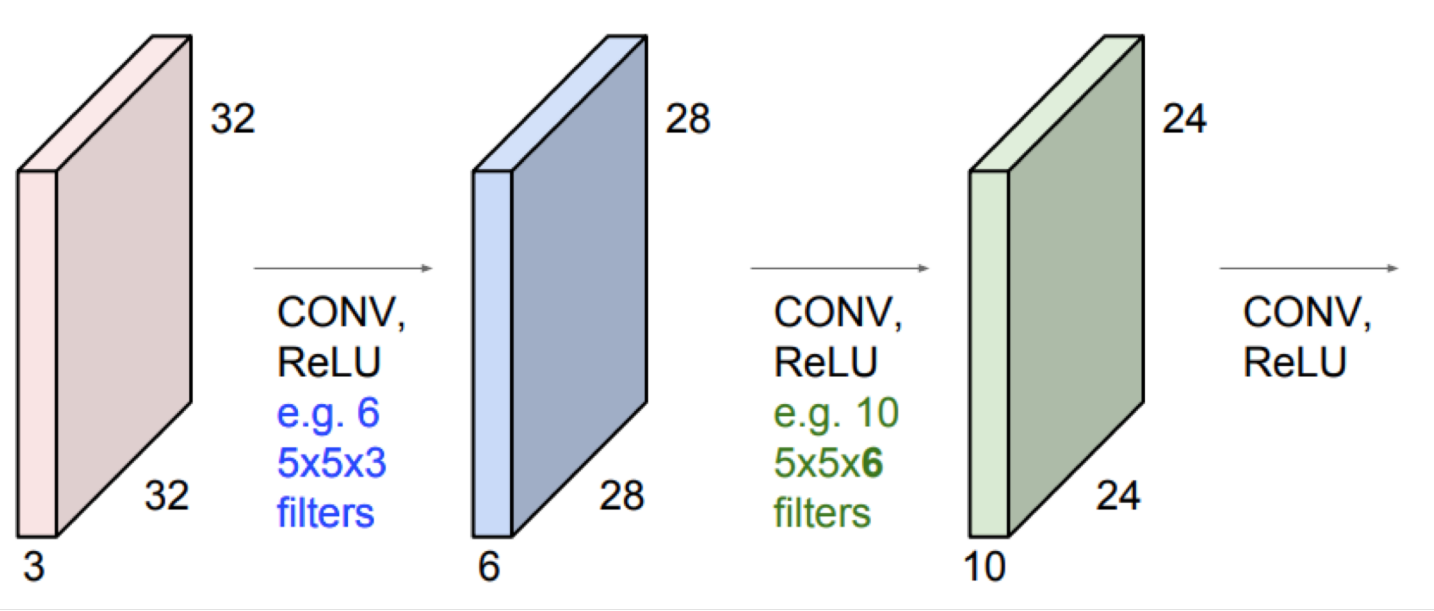
Convolution in computer vision

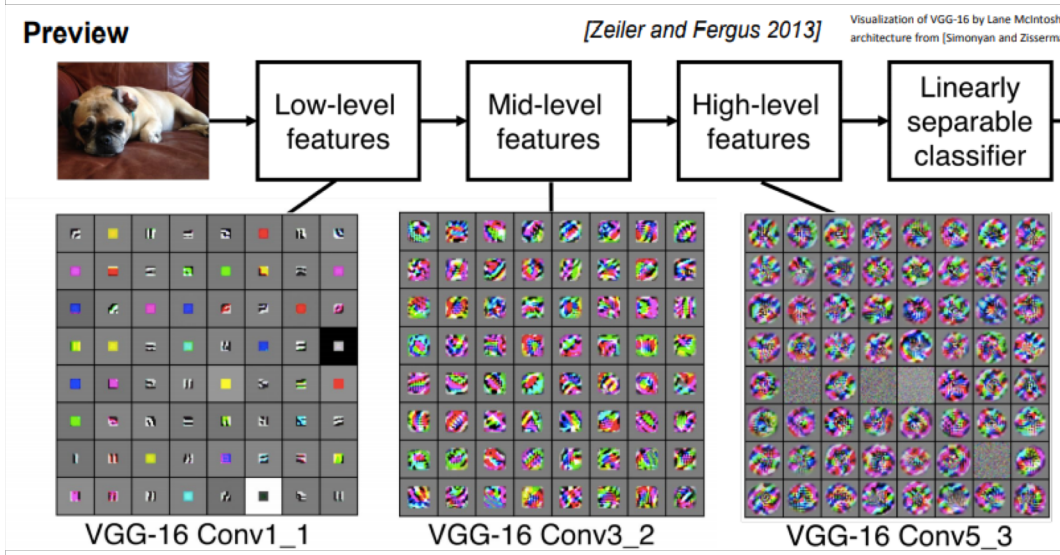


- Extract feature
  ex) edge, ridge, etc…

이미지 출처 : https://images.slideplayer.com/25/7974874/slides/slide_1.jpg

Convolution Neural Net (CNN)



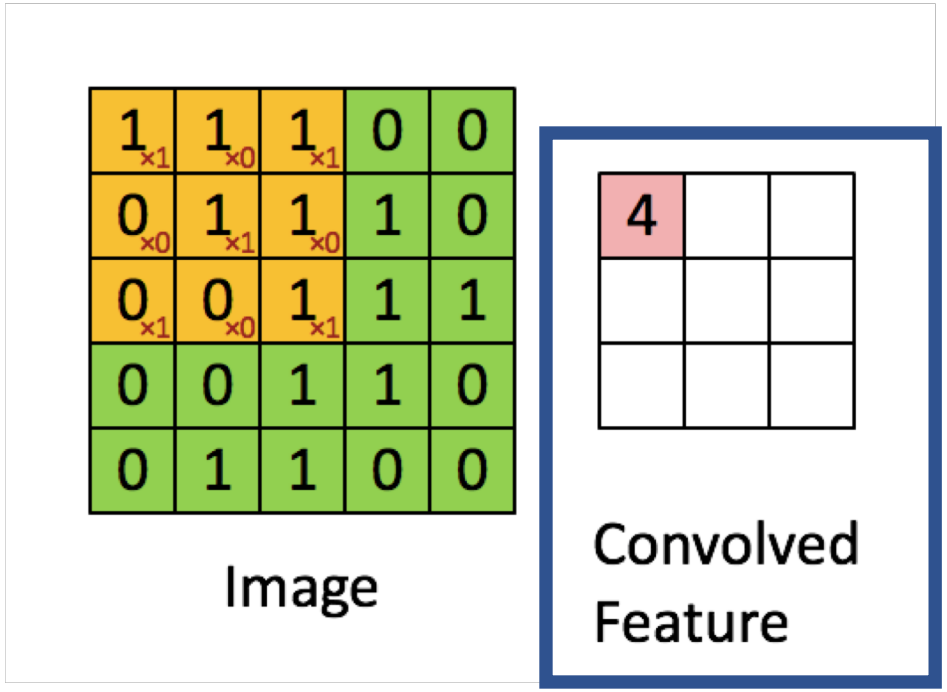


One filter => one activation map
6, 5x5 filter => get 6 activation map

이미지 출처 : Stanford lecture cs231n

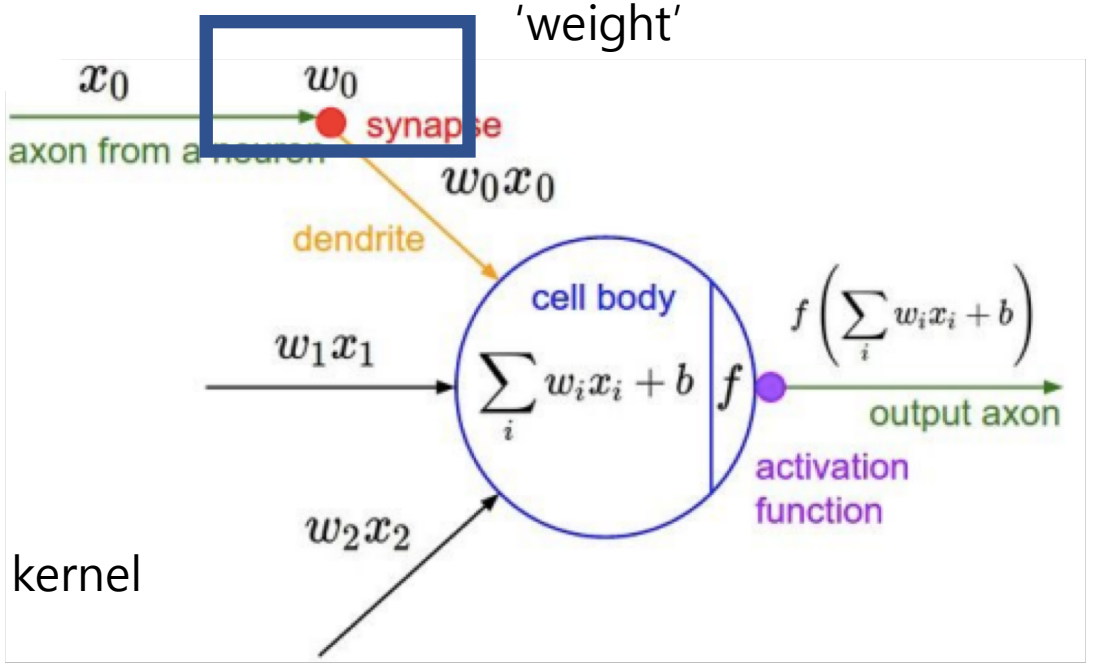
Convolution Neural Net (CNN)

Convolution Layer



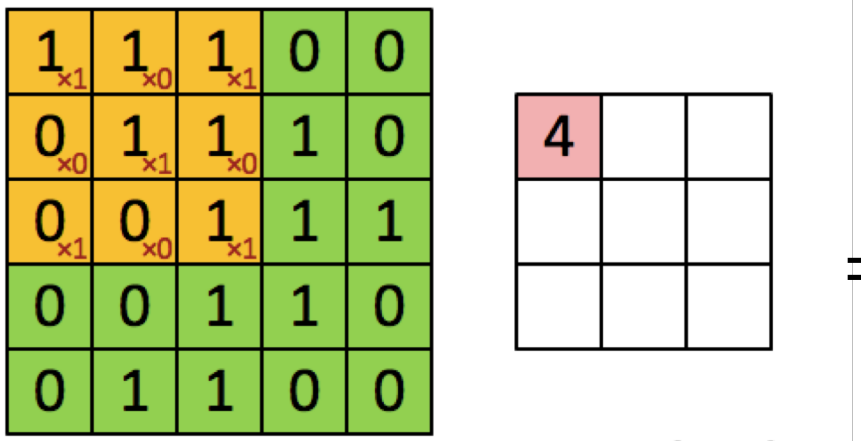
Train this kernel

Train parameter 'weight'



Convolution Neural Net (CNN)

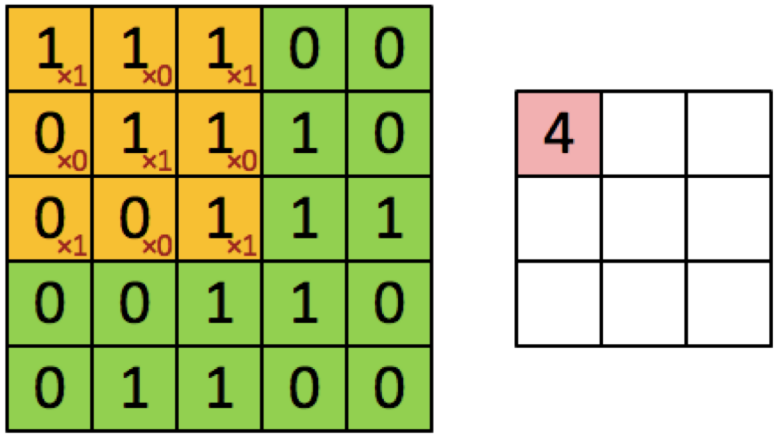
Convolution Layer



= Feature Map

Convolution Neural Net (CNN)

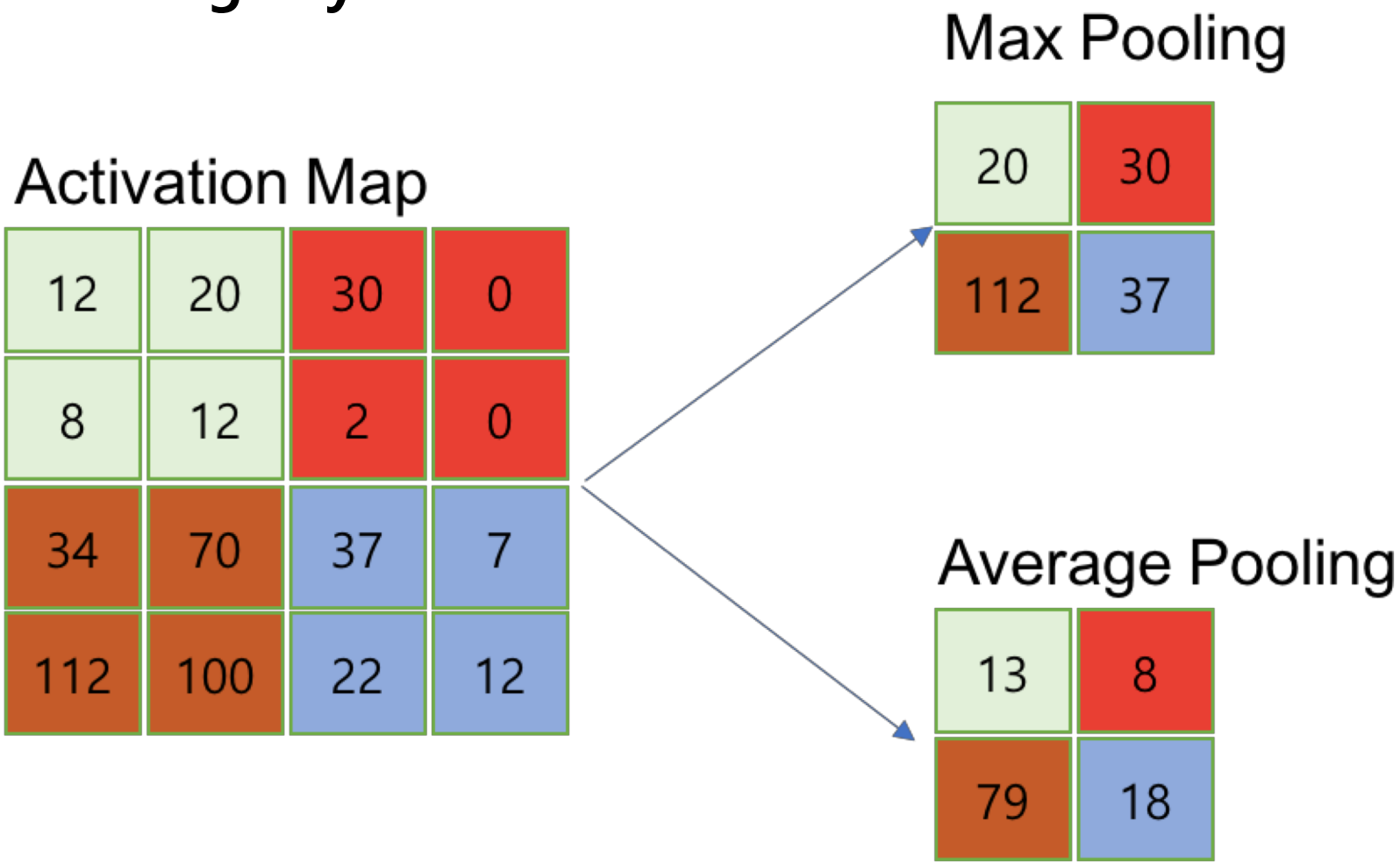
Convolution Layer



Feature Map x Activation function (Relu)
= Activate Map (Convolution layer output)

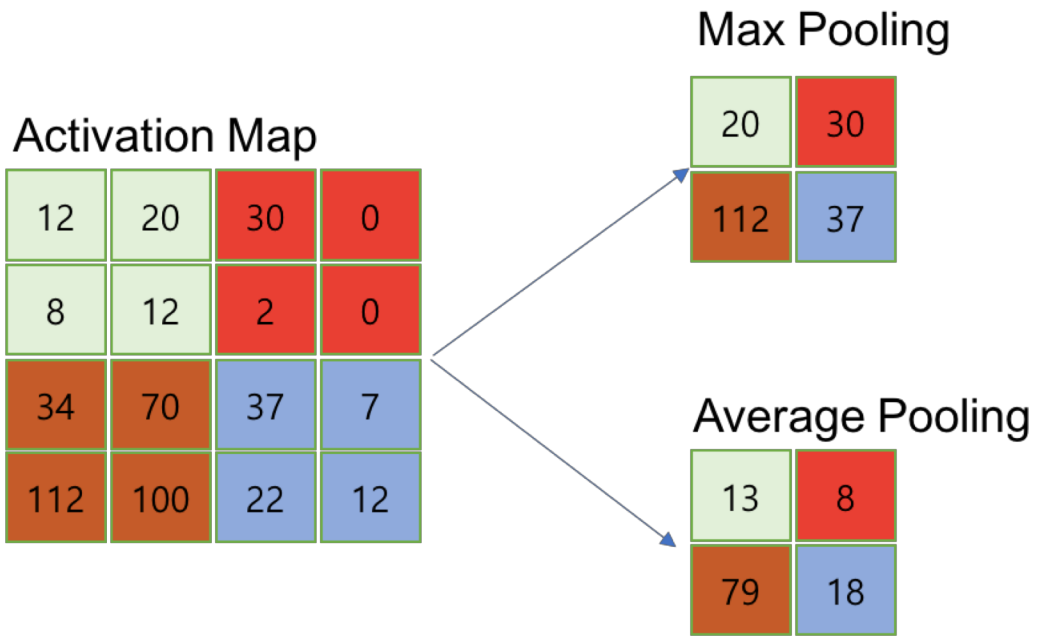
Convolution Neural Net (CNN)

Pooling Layer



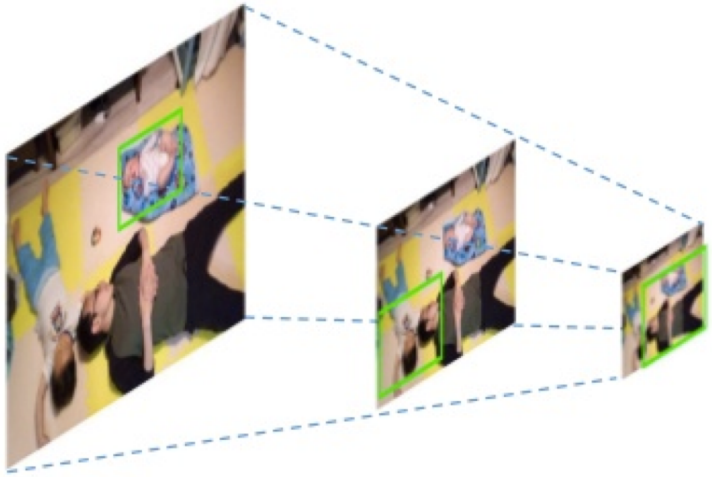
Convolution Neural Net (CNN)

Why Pooling?



Multi-Scaling (Pyramid Pooling)

- Find objects in images at different scales
- Combined with a **sliding window,** it can find objects in various locations **with the same window size**

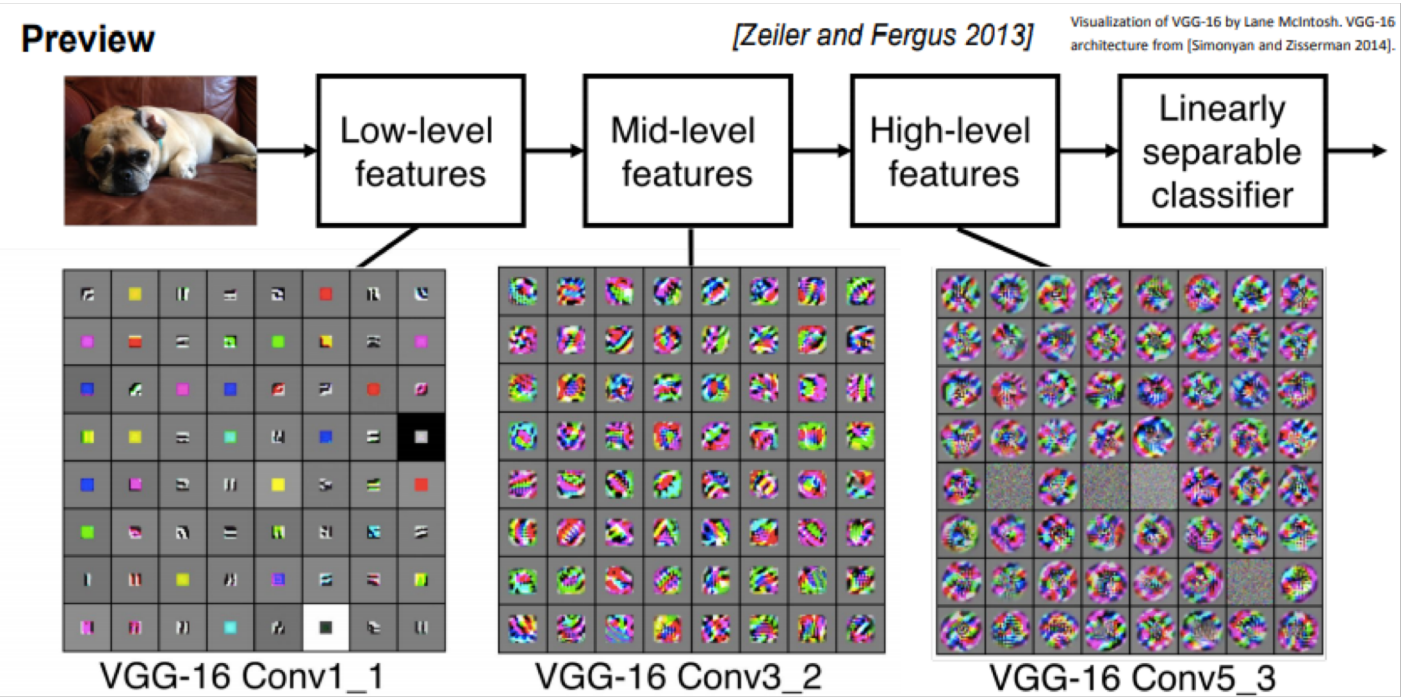


11

이미지 출처 : https://image.slidesharecdn.com/fpt17slidingwindowv1-180225213227/95/fpt17-an-object-detector-based-on-multiscale-sliding-window-search-using-a-fully-pipelined-binarized-cnn-on-an-fpga-11-638.jpg?cb=1519594472

Convolution Neural Net (CNN)

CNN Architecture



AlexNet

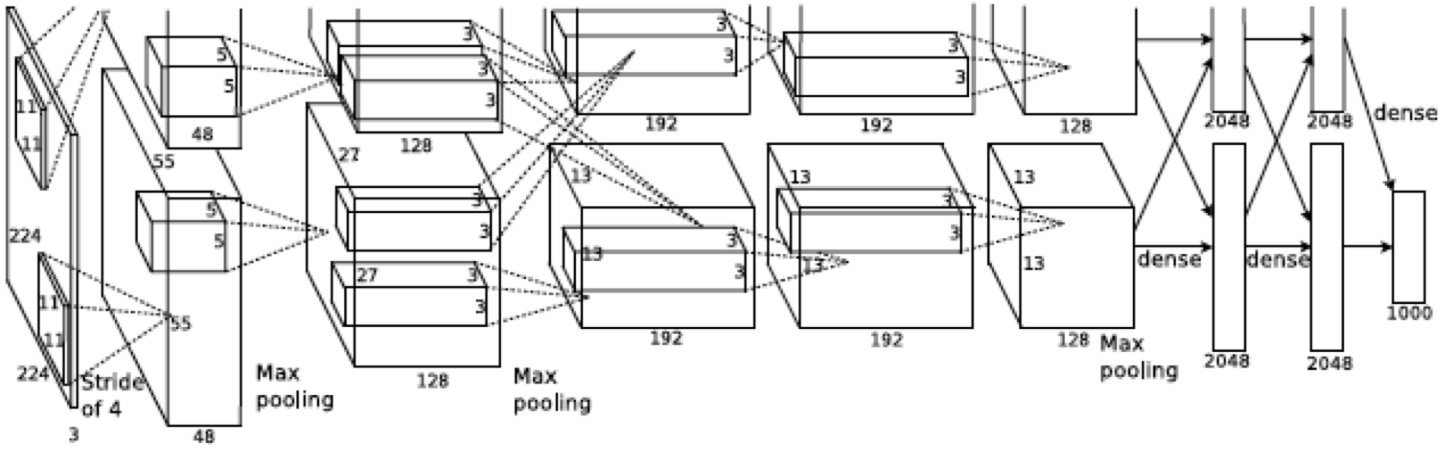


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

AlexNet

- CNN Architecture
- Activation function 'Relu'
- GPU
- Preventing overfitting
  - Dropout
  - Data augmentation

AlexNet

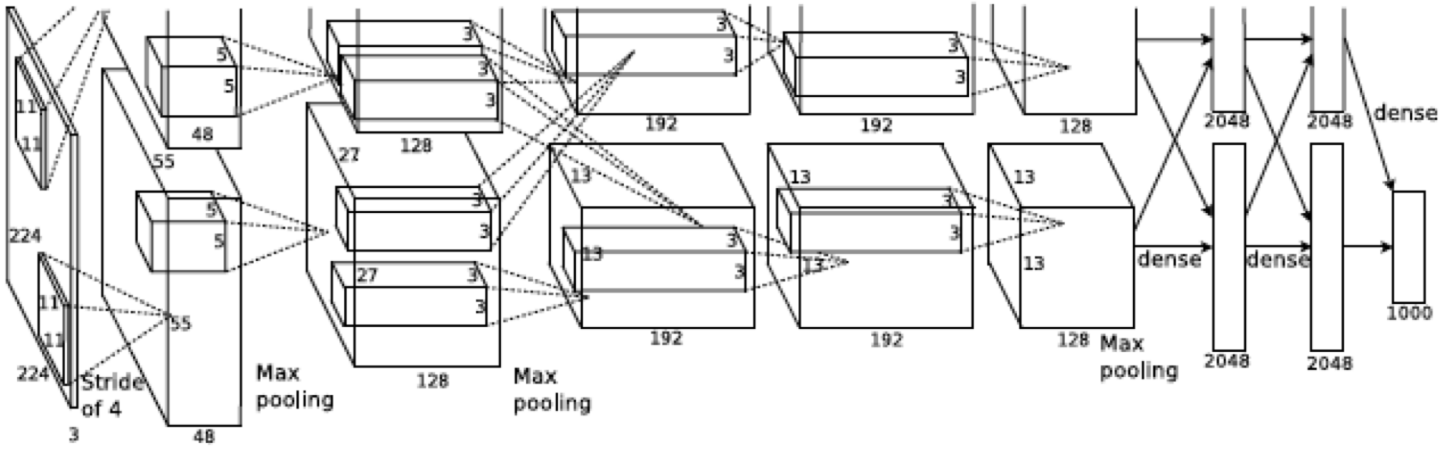


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

AlexNet Architecture

- 5 Convolution+pooling Layer
- 3 Fully Connected Layer
- Two-GPU net
  because of hardware limitation
  at the time

Why Deep learning

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. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

Pattern recognition
&
machine learning

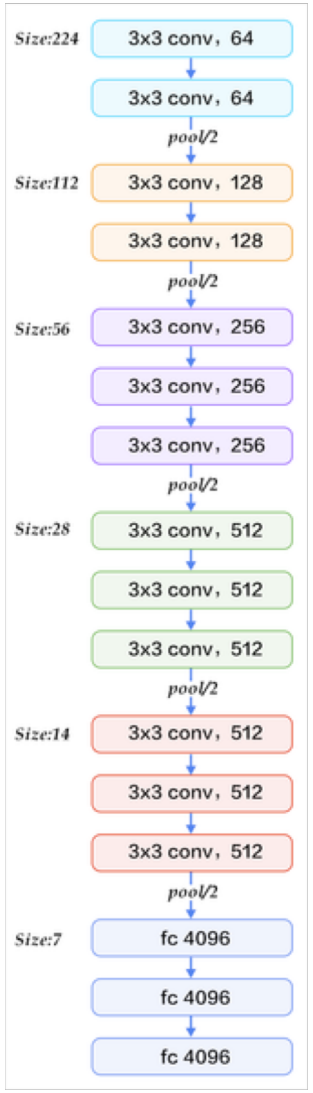
10.9% !

Deep learning

Top 5 test error

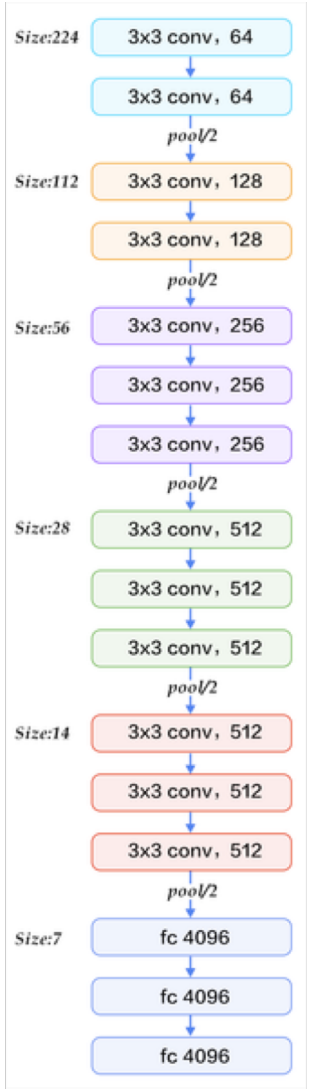
The Top-5 error rate is the percentage of test examples for which the correct class was not in the top 5 predicted classes.

VGG



- *Very Deep Convolutional Networks For Large Scale Image Recognition - Karen Simonyan and Andrew Zisserman; 2015*
- ILSVRC'14 2nd in classification, 1st in localization
- Smaller convolution filter
- Deeper than previous network

VGG

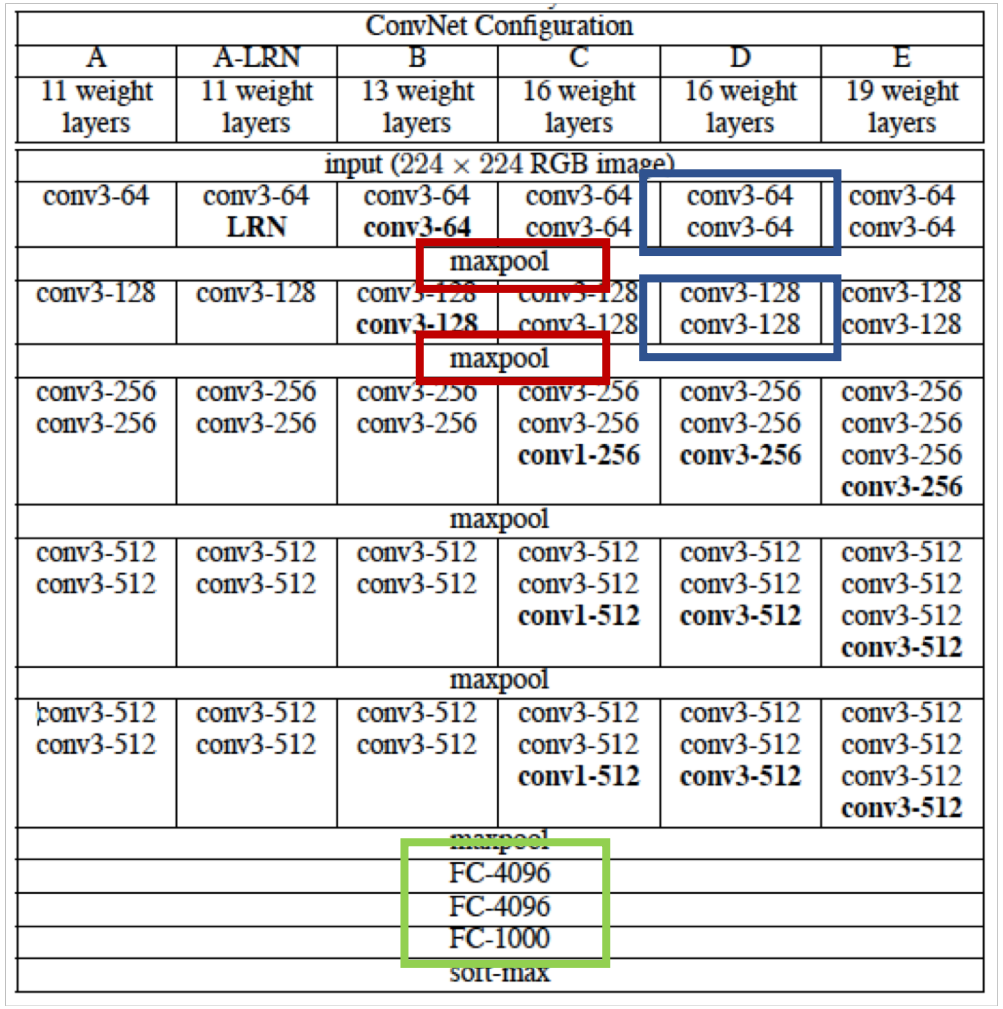


- **Smaller convolution filter**
  -AlexNet (2012)
  filter size = (11 x 11), stride = 4

  -ZFNet (2013)
  filter size = (7 x 7), stride =2

- **Deeper layer**
  -AlexNet (2012)
  layer depth = 8 layers

Architecture



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					

- **Convolution layer (blue)**
  - kernel size = (3x3), stride = 1
- **Max pooling layer (red)**
  - kernel size = (2x2), stride = 2
- **Convolution layer + max pooling layer**
  - get spatial resolution for the feature maps

Architecture

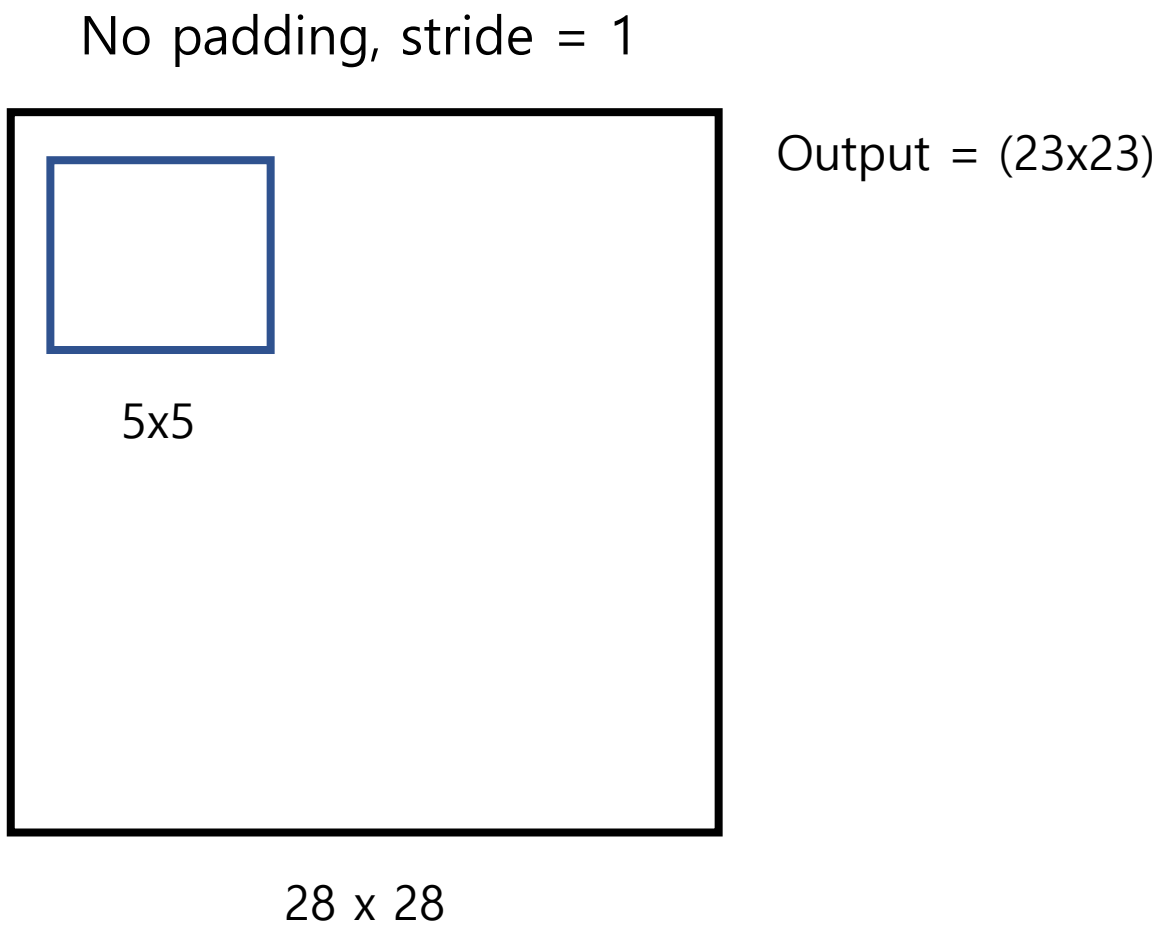
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	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					

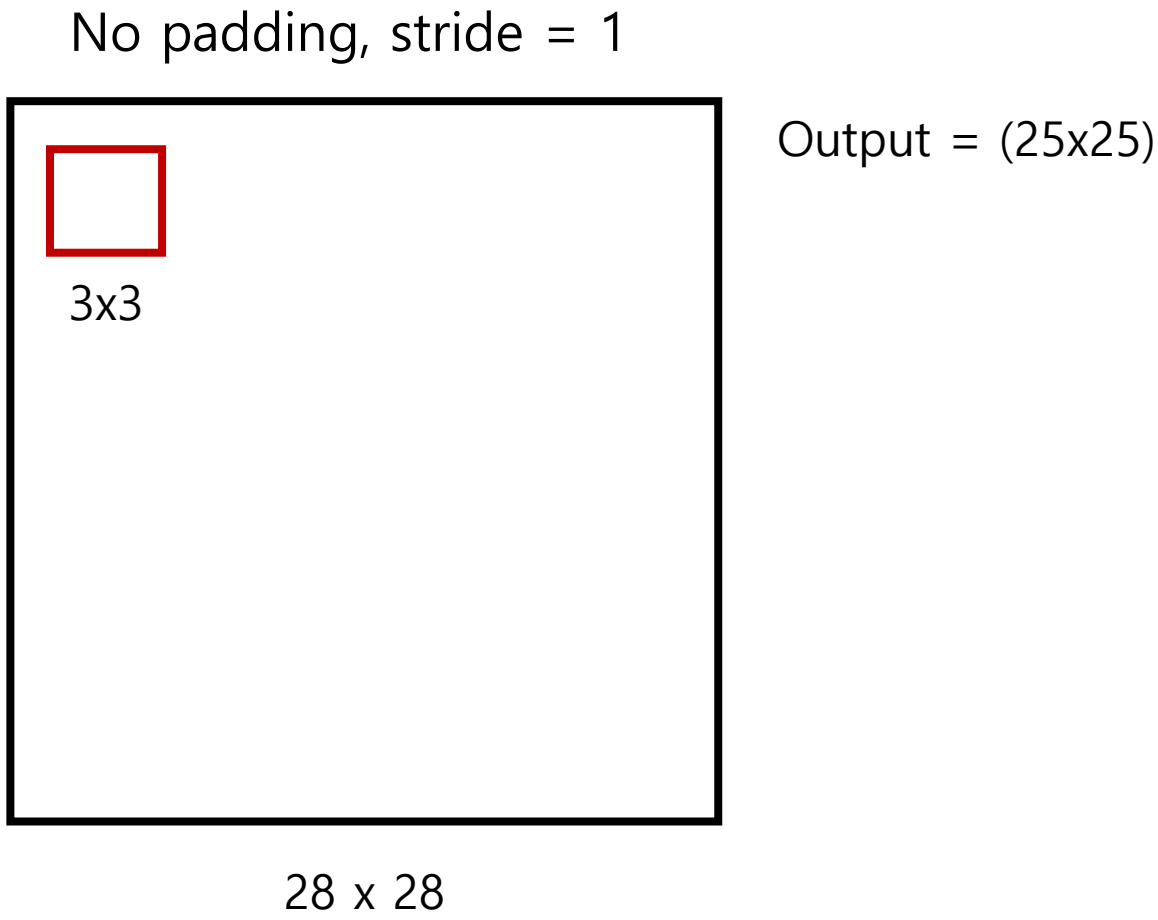
- **Activation function = 'Relu'**
  - for non-linearity
- **Padding = 1 pixel (convolution layer)**
- **No LRN (Local Response Normalization)**
  - not improve the performance
  - increase memory consumption and computation time
- **Parameter**

Table 2: **Number of parameters** (in millions).

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

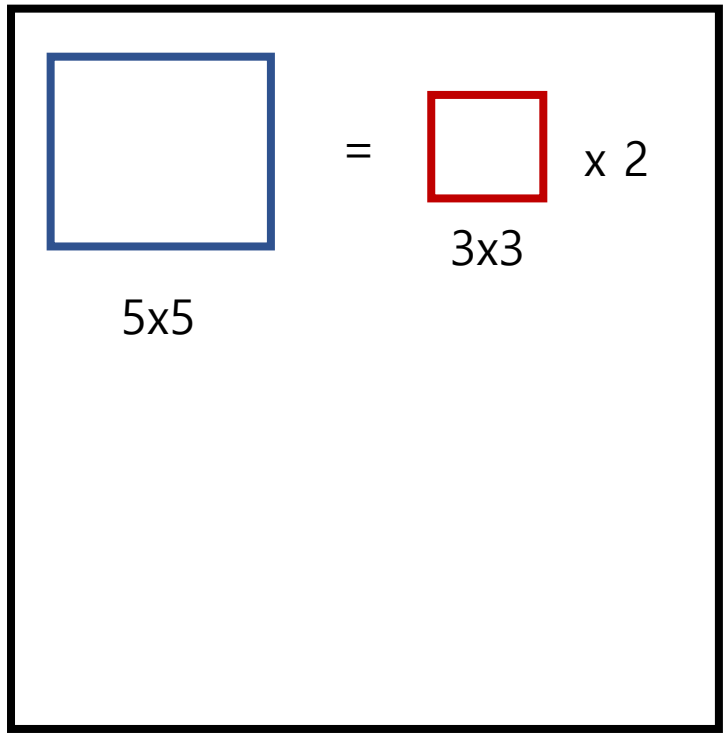
Smaller Convolution filter





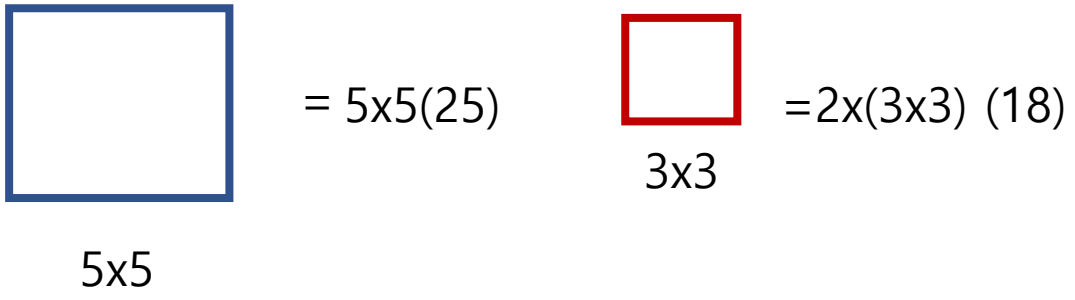
Smaller Convolution filter

No padding, stride = 1



Why use smaller filter instead of bigger one

- Makes the decision function more discriminative (increase non-linearity)
- Decrease the number of parameter



Deeper layer

Vanishing gradient (Fine tuning)

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**
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**
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**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

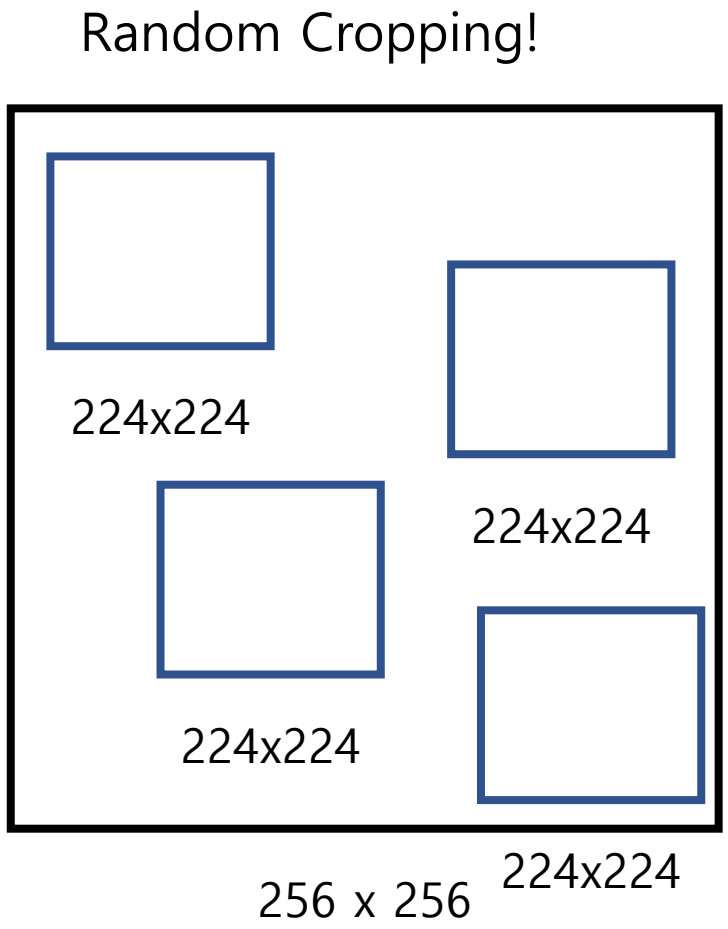
Use pretrained 'A' VGG model

- Net 'A' is simple model than other networks.

- initialized weights in first 4 convolution layers and last 3 FC layers using pretrained net 'A'

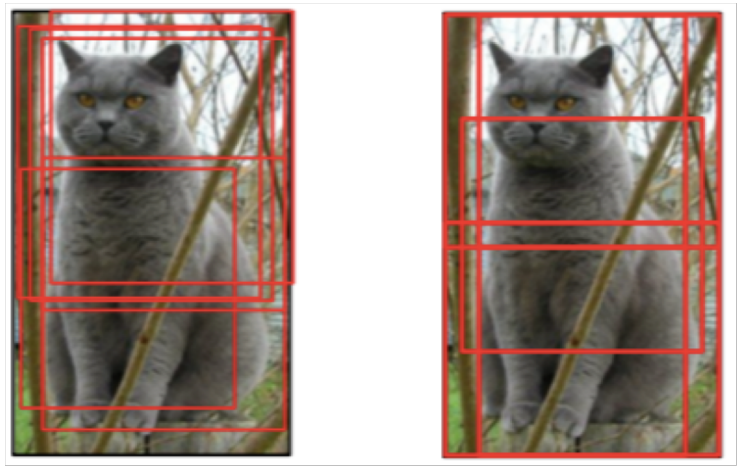
Train

Data augmentation



Random horizontal flipping & RGB color shift (Same as Alexnet's augmentation strategy)

- extracting random 224x224 patches



Train

Data augmentation

RGB color shift
(Same as Alexnet's augmentation strategy)

- change RGB color !

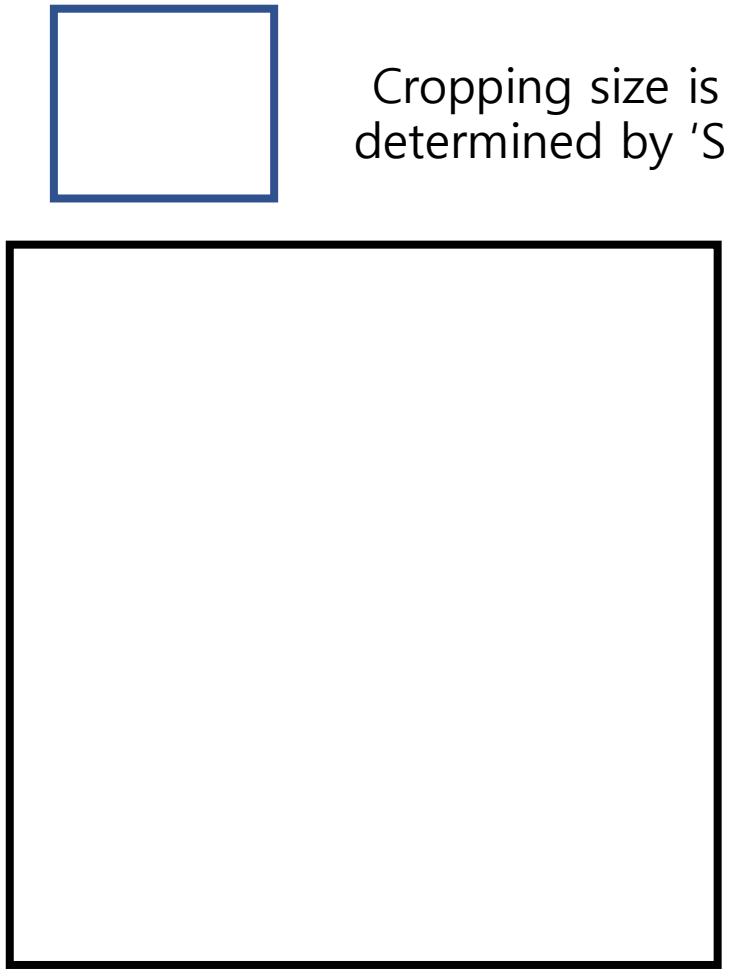
- PCA on the set of RGB pixel values

- P1, p2, p3 = eigenvectors from PCA
  lambda1, 2, 3 = eigenvalues from PCA
  a1, a2, a3 = Gaussian noise (0, 0.1)

$$I_{xy} = [I_{xy}^R,\ I_{xy}^G,\ I_{xy}^B]^T \qquad [\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$
$$\alpha_i \sim N(0,\ 0.1)$$

Train

Training scale



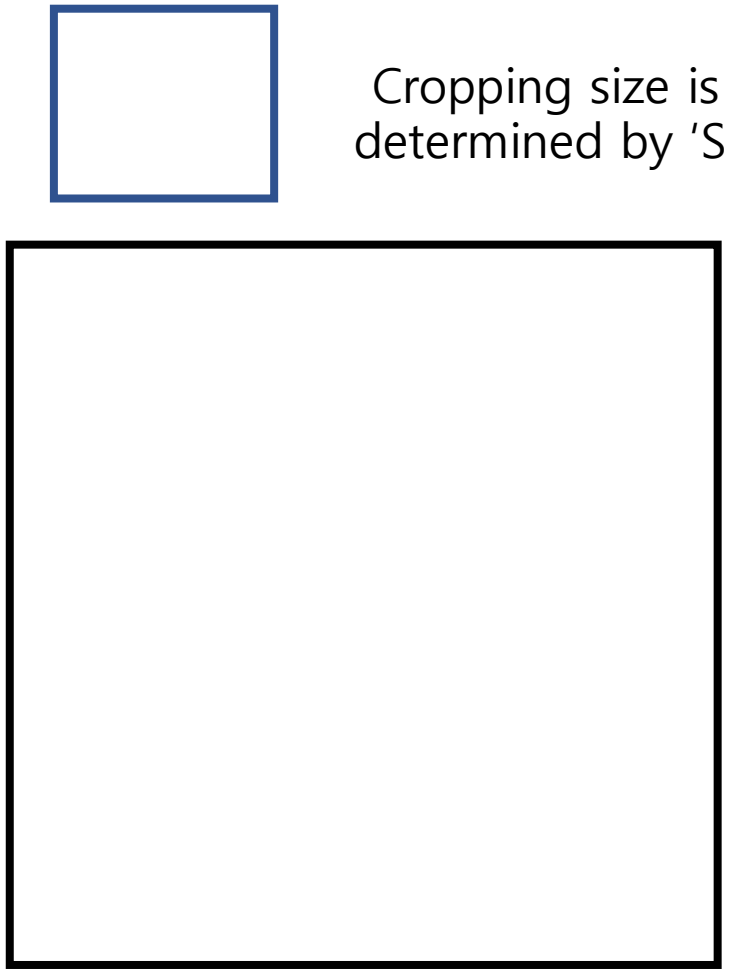
- 'S' is 'train scale' value

- 'S' >= 224
  If S == '224' :
    Crop whole image

  Elif S > '224':
    Crop small part of image

Train

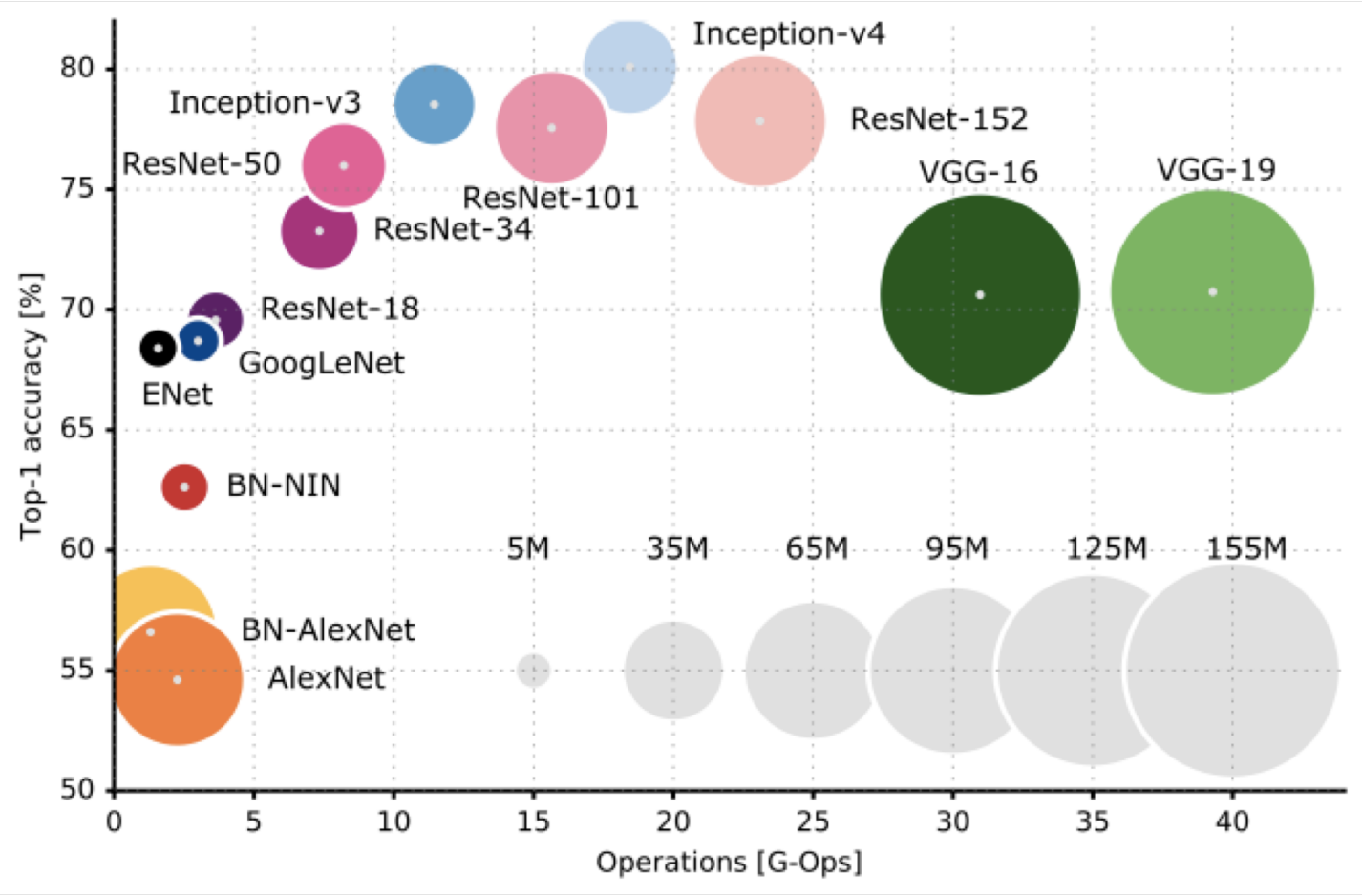
Training scale



- 'S' is 'train scale' value

- 'S' >= 224
  If S == '224' :
    Crop whole image

  Elif S > '224':
    Crop small part of image


Inception-v4
Inception-v3
ResNet-152
ResNet-50
ResNet-101
VGG-16
VGG-19
ResNet-34
ResNet-18
GoogLeNet
ENet
BN-NIN
5M
35M
65M
95M
125M
155M
BN-AlexNet
AlexNet
Top-1 accuracy [%]
80
75
70
65
60
55
50
0
5
10
15
20
25
30
35
40
Operations [G-Ops]