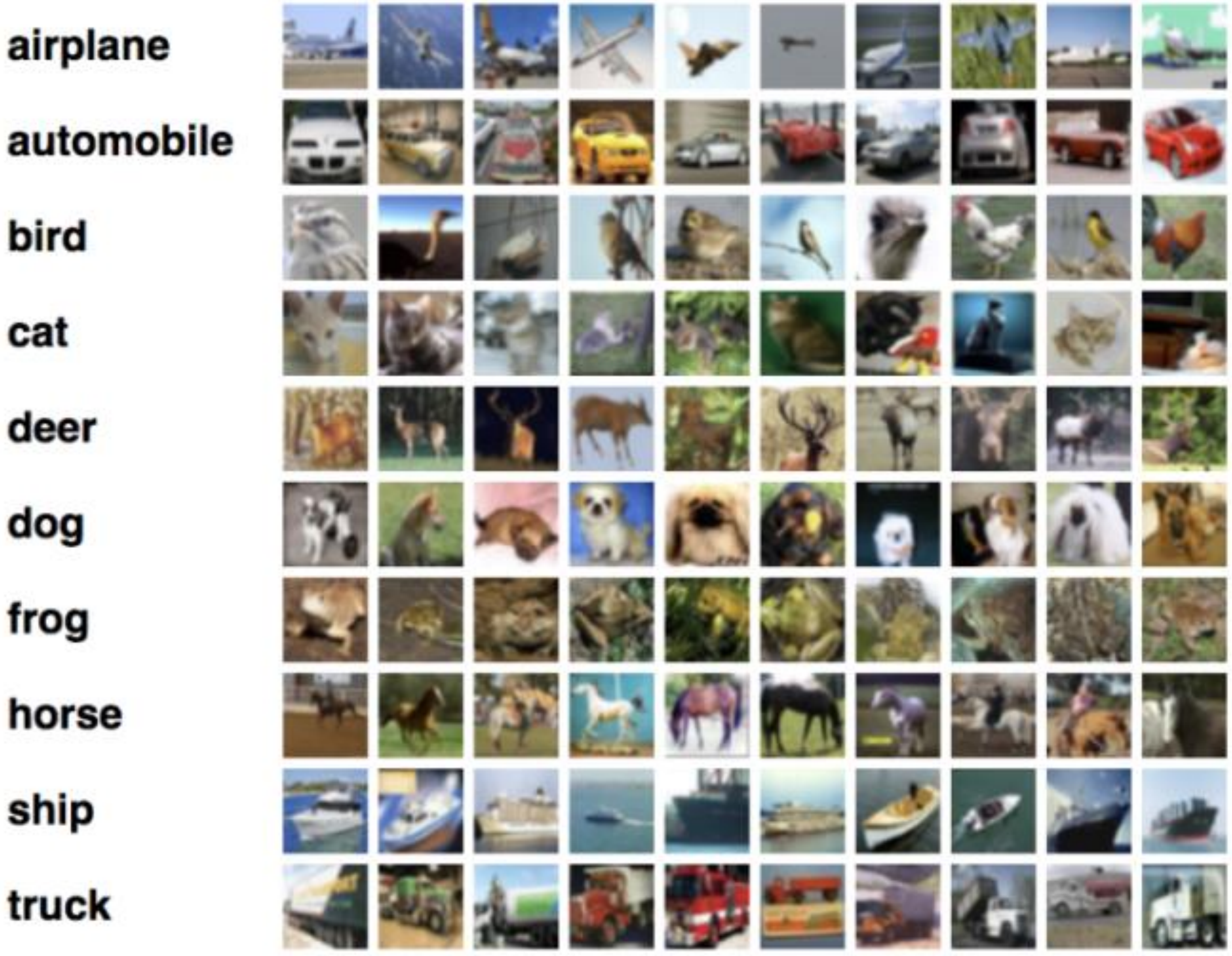
Convolutional Neural Network (CNN)

강사: 김 남 범 교수

LSVRC

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



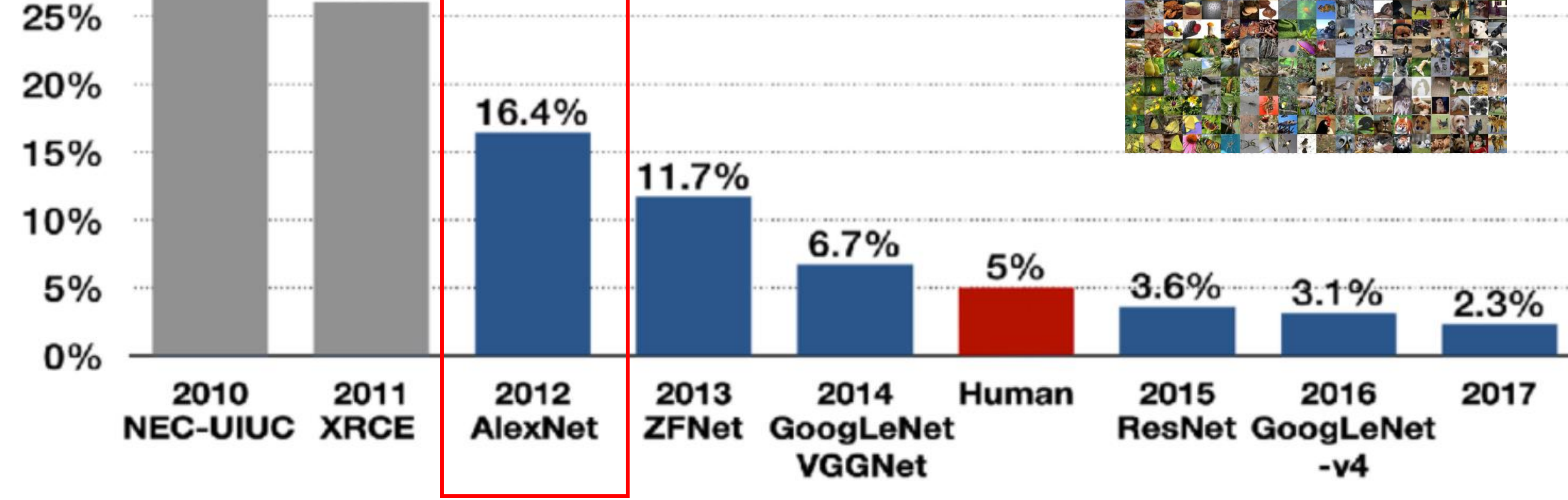
http://www.image-net.org/challenges/LSVRC/

LSVRC (Imagenet)

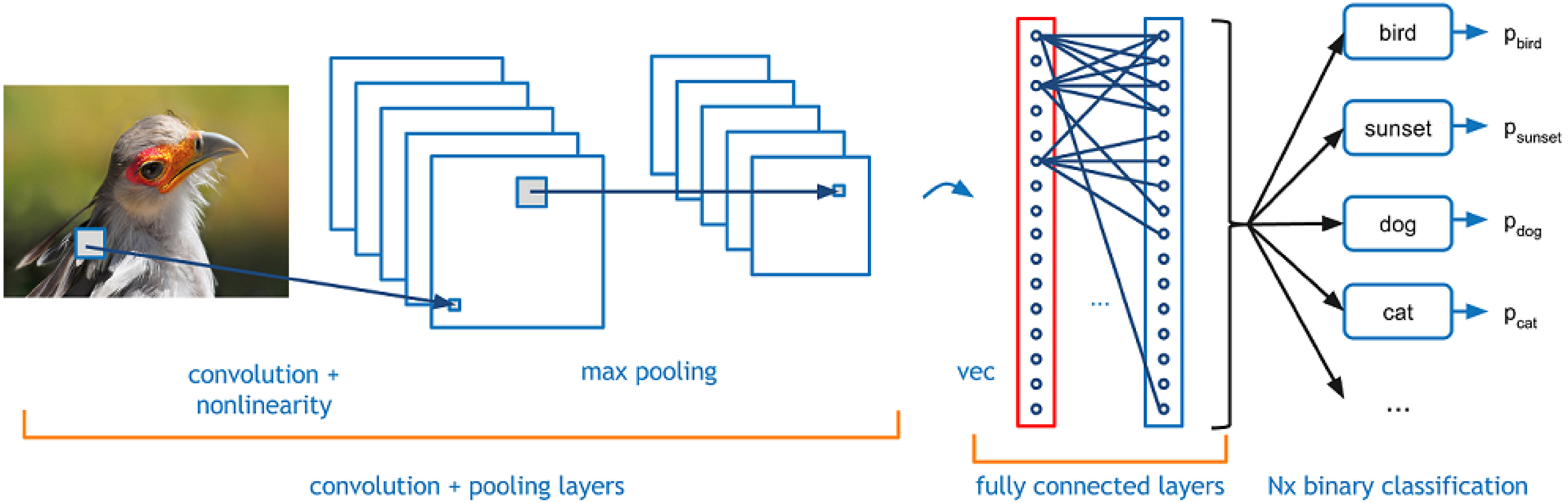
- *ReLU*
- *Dropout*
- *Max pooling*
- *Data augmentation*

Top-5 error

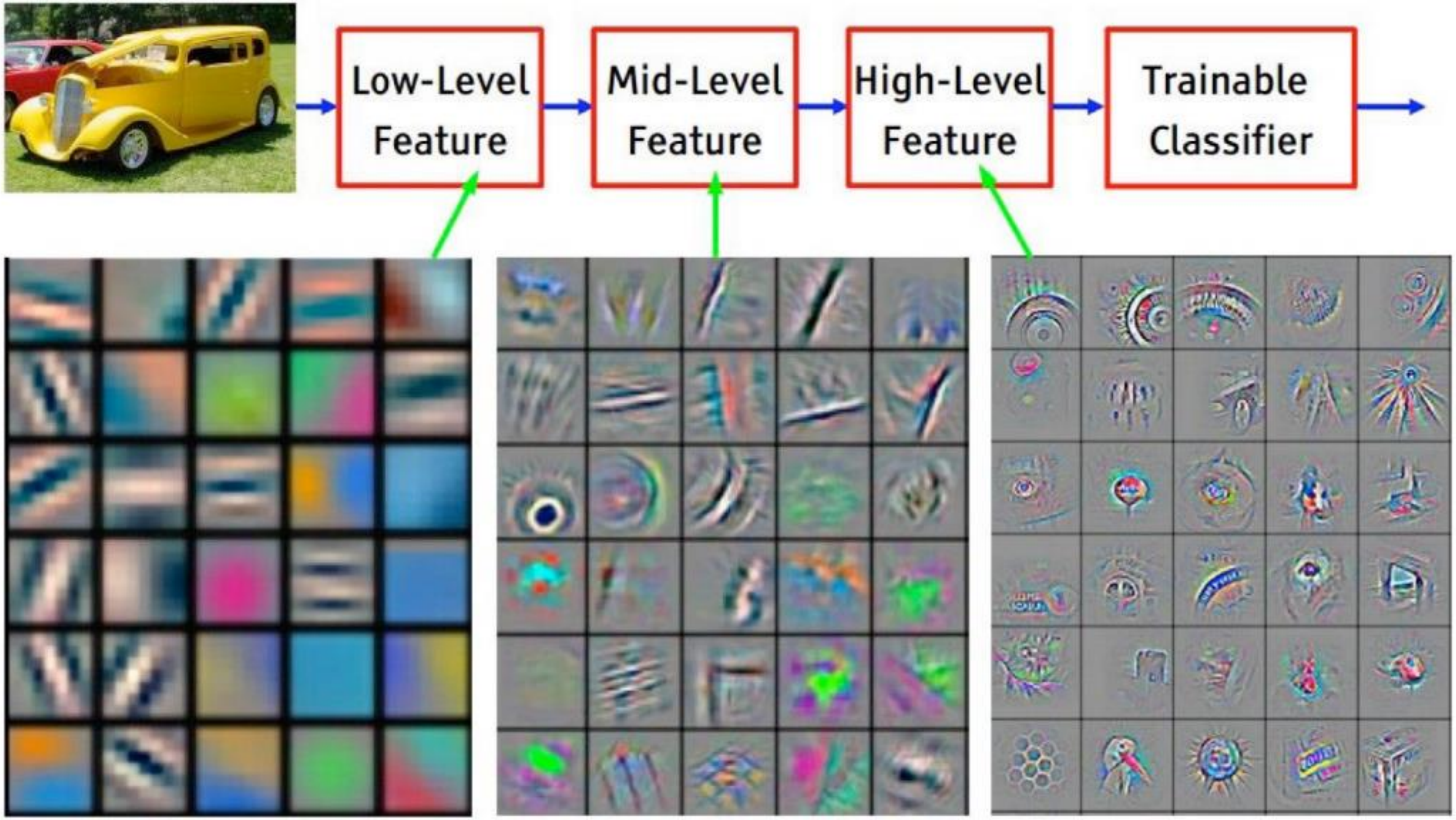
CNN + Deep learning



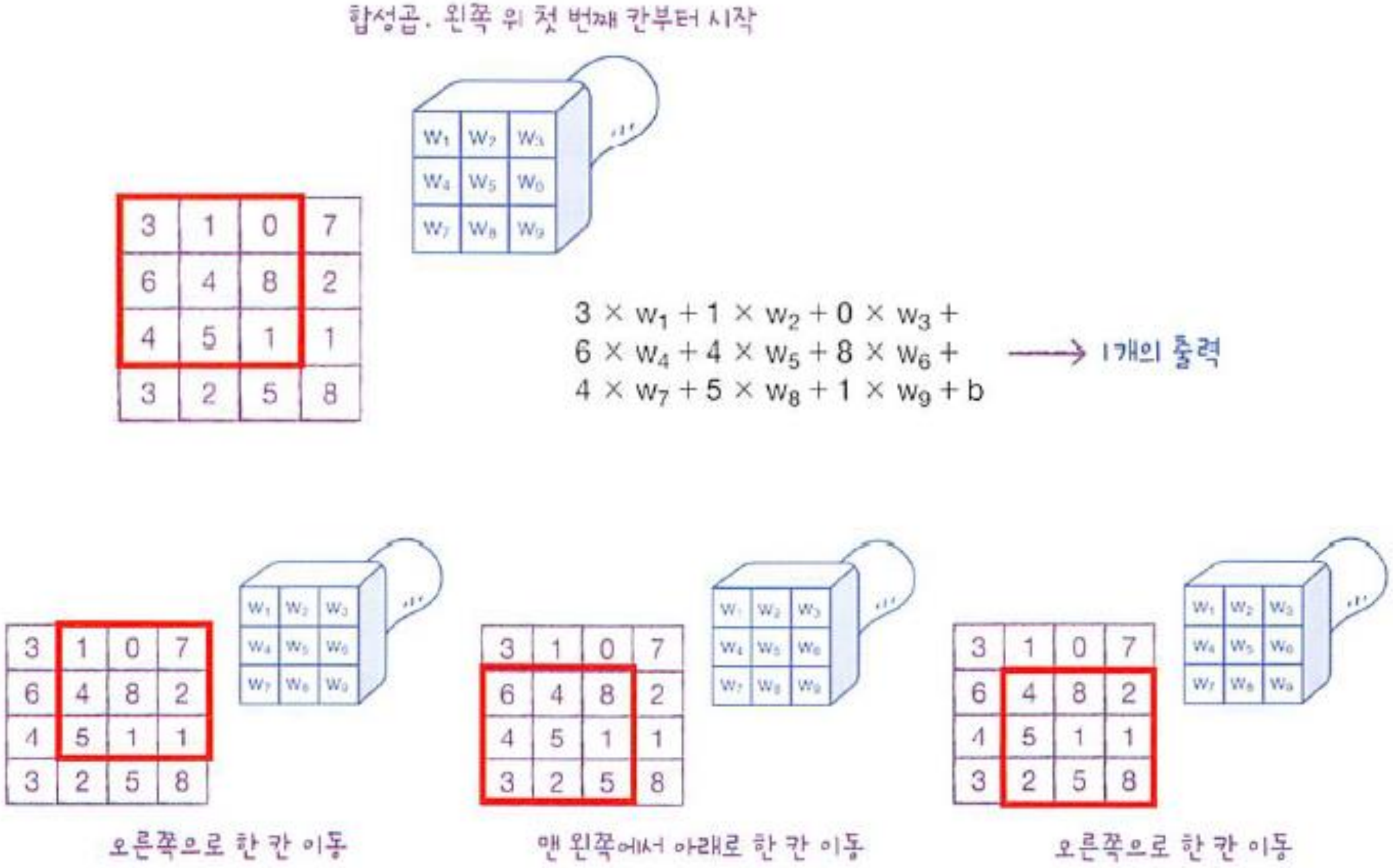
CNN의 구조


convolution +
nonlinearity
max pooling
vec
...
bird
sunset
dog
cat
...
p_{bird}
p_{sunset}
p_{dog}
p_{cat}
convolution + pooling layers
fully connected layers
Nx binary classification

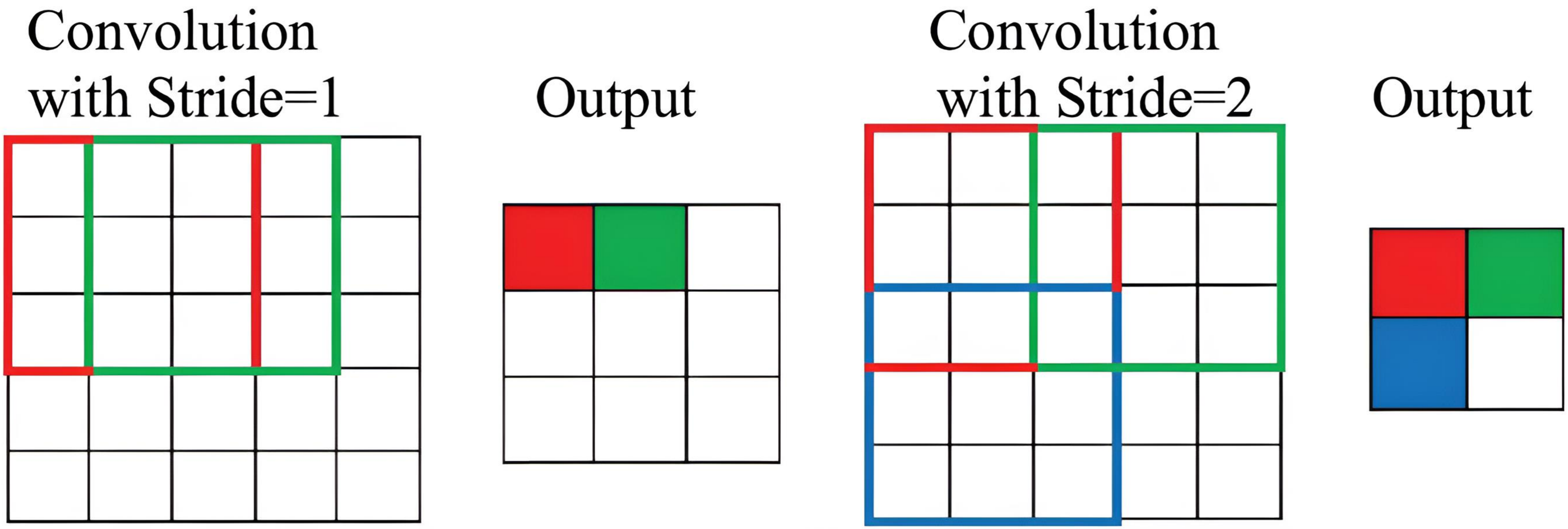
ZFNet (2013)


Low-Level
Feature
Mid-Level
Feature
High-Level
Feature
Trainable
Classifier

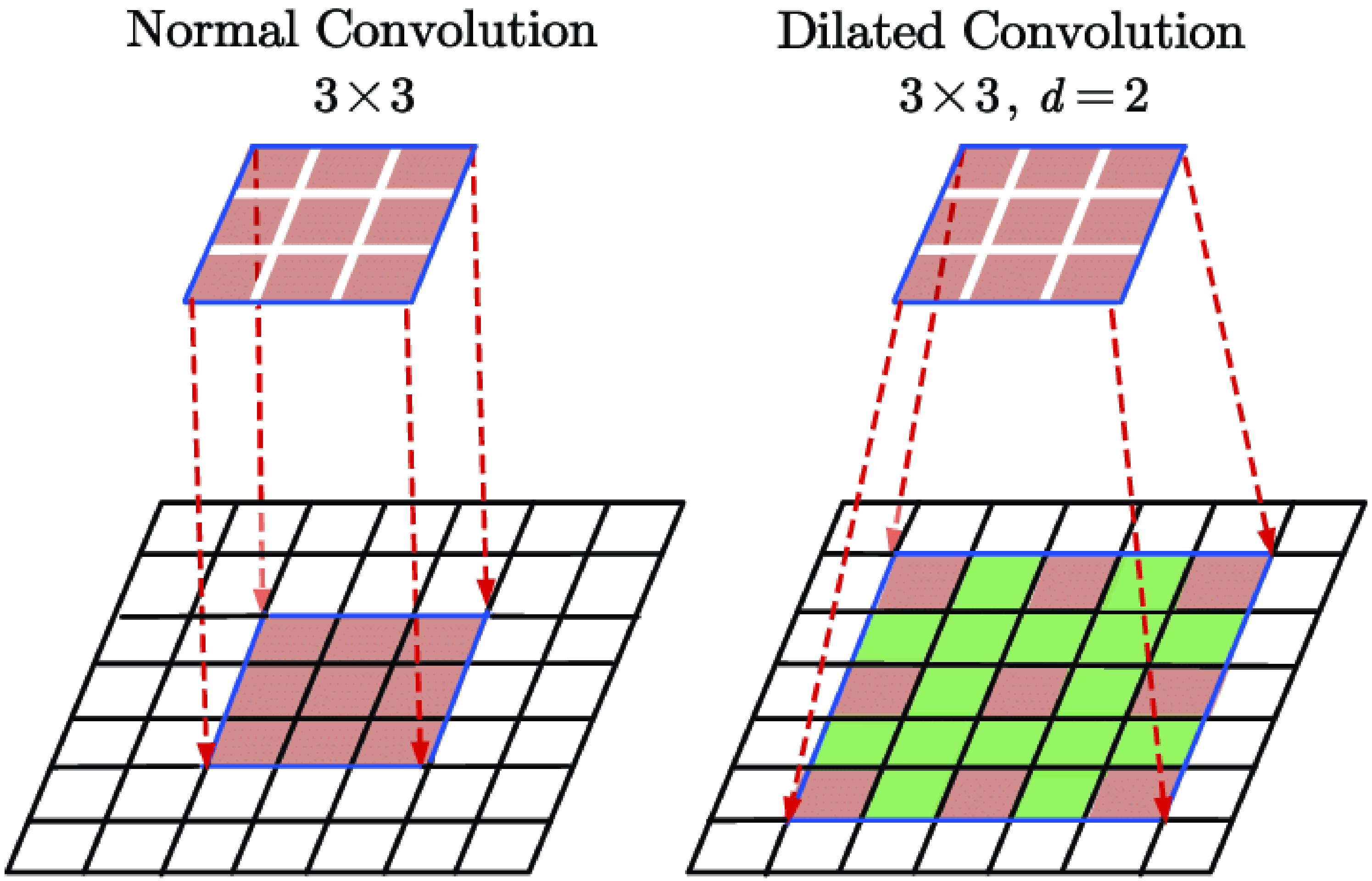
합성곱의 과정


합성곱. 왼쪽 위 첫 번째 칸부터 시작
$3 \times w_1 + 1 \times w_2 + 0 \times w_3 +$
$6 \times w_4 + 4 \times w_5 + 8 \times w_6 +$
$4 \times w_7 + 5 \times w_8 + 1 \times w_9 + b$
1개의 출력
오른쪽으로 한 칸 이동
맨 왼쪽에서 아래로 한 칸 이동
오른쪽으로 한 칸 이동

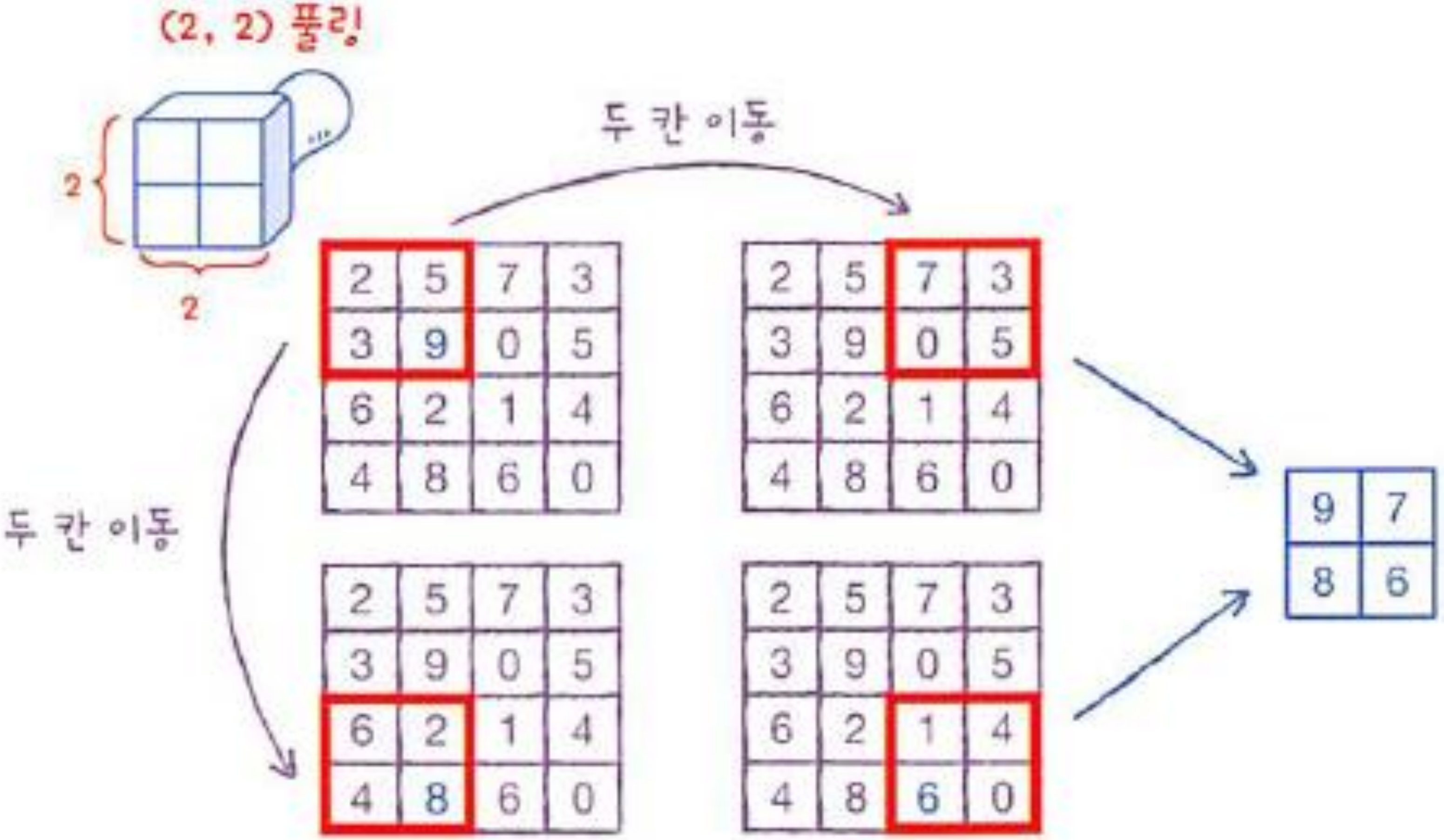
Stride


Convolution with Stride=1
Output
Convolution with Stride=2
Output

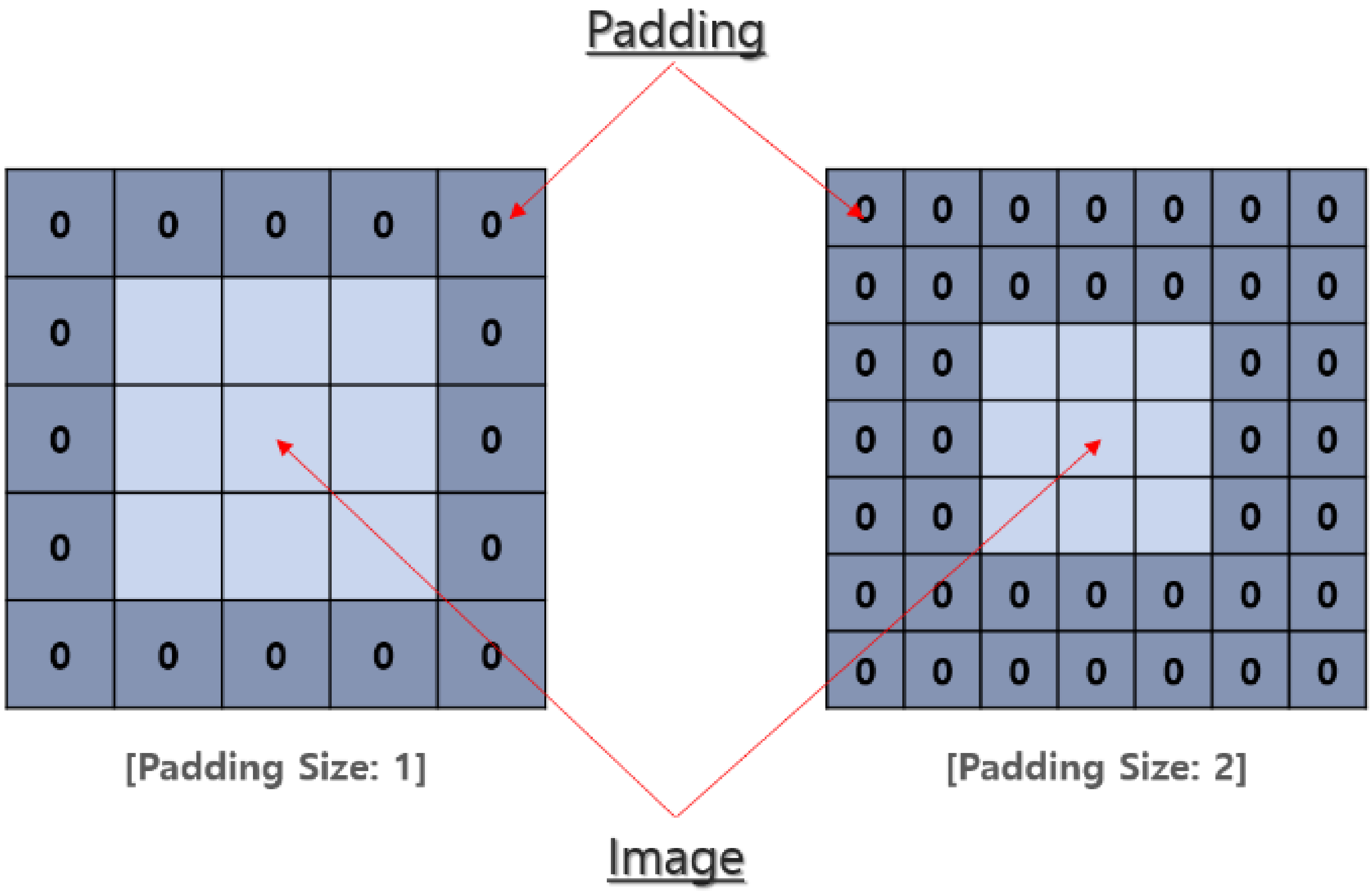
Dilation



풀링 (Pooling)



패딩 (Padding)


Padding
0 0 0 0 0
0 0
0 0
0 0
0 0 0 0 0
[Padding Size: 1]
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
[Padding Size: 2]
Image

합성곱 연산 후 결과영상의 크기

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

n_{in}: number of input features

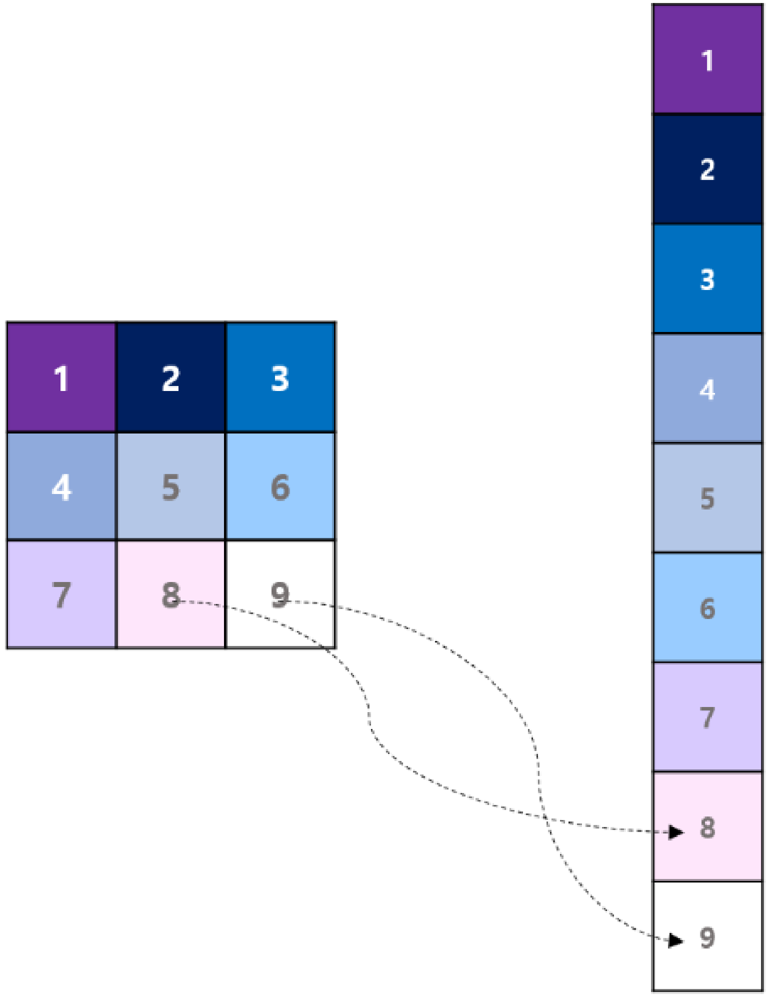
n_{out}: number of output features

k: convolution kernel size

p: convolution padding size

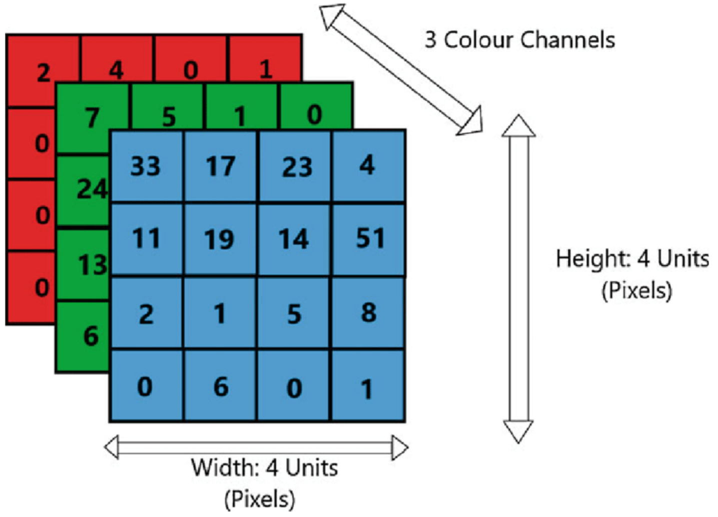
s: convolution stride size

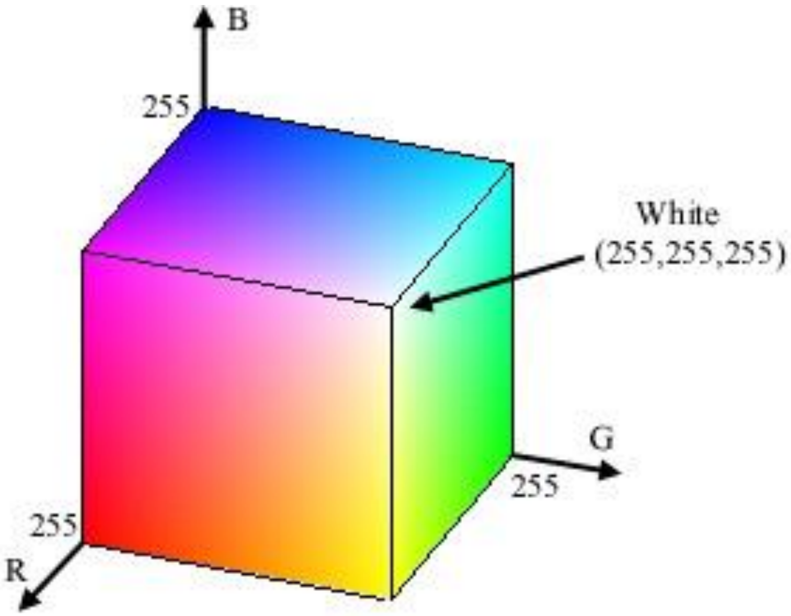
Flattening layer



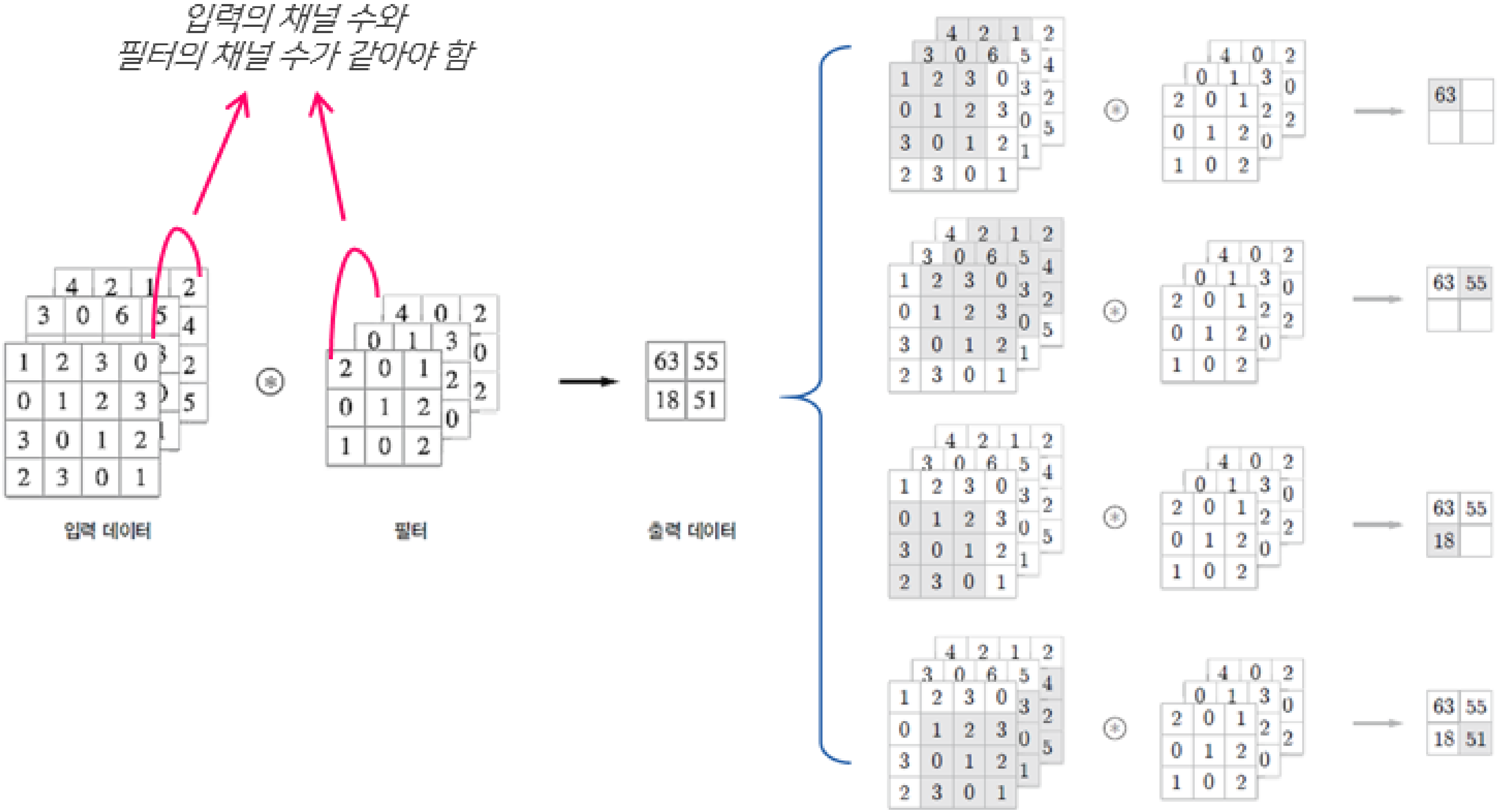
https://yjjo.tistory.com/8

RGB 칼라 영상

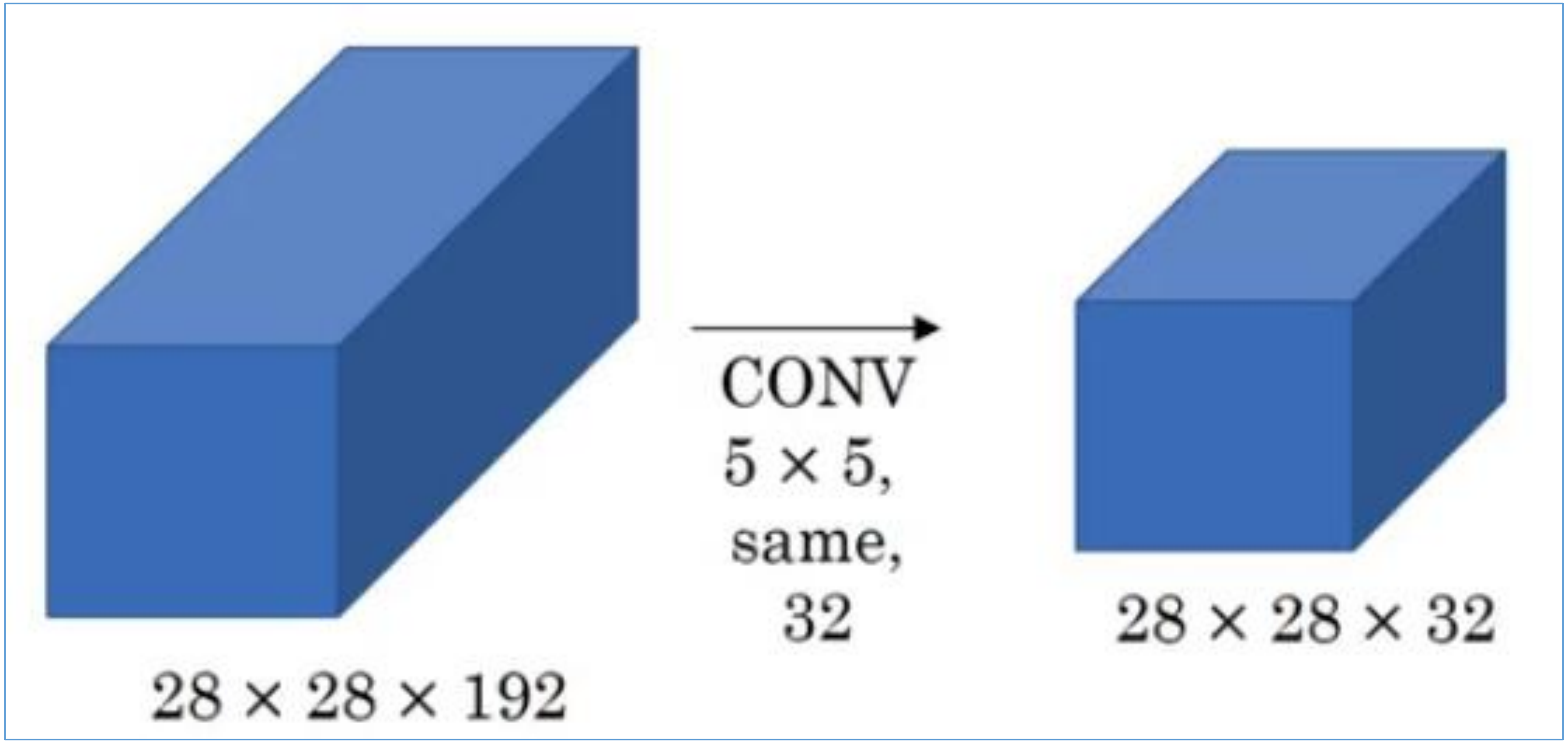

3 Colour Channels
2 4 0 1
0
0
0
7 5 1 0
24
13
6
33 17 23 4
11 19 14 51
2 1 5 8
0 6 0 1
Height: 4 Units
(Pixels)
Width: 4 Units
(Pixels)


B
255
White
(255,255,255)
G
255
255
R

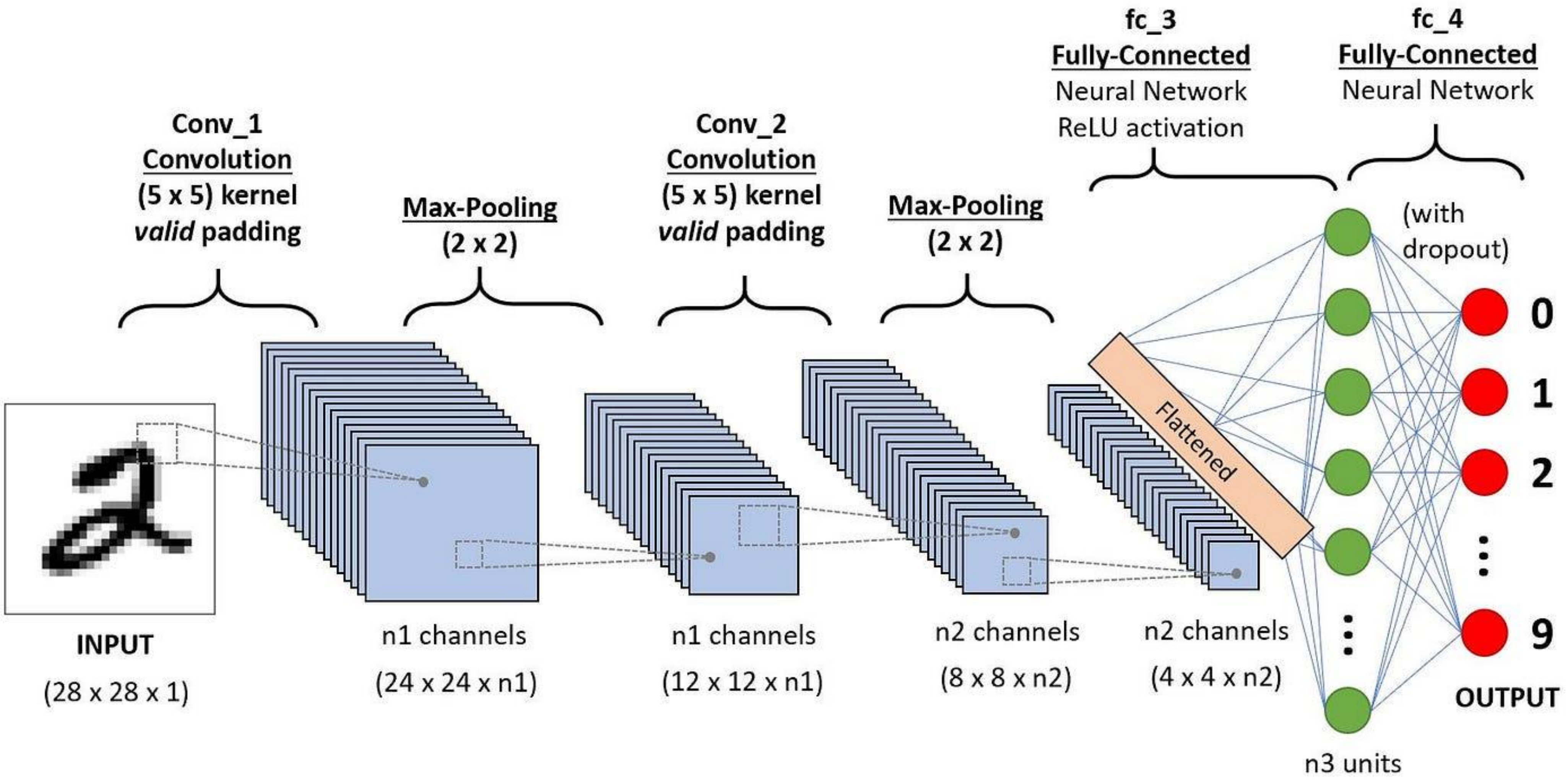
RGB 영상의 합성곱



CNN 블럭



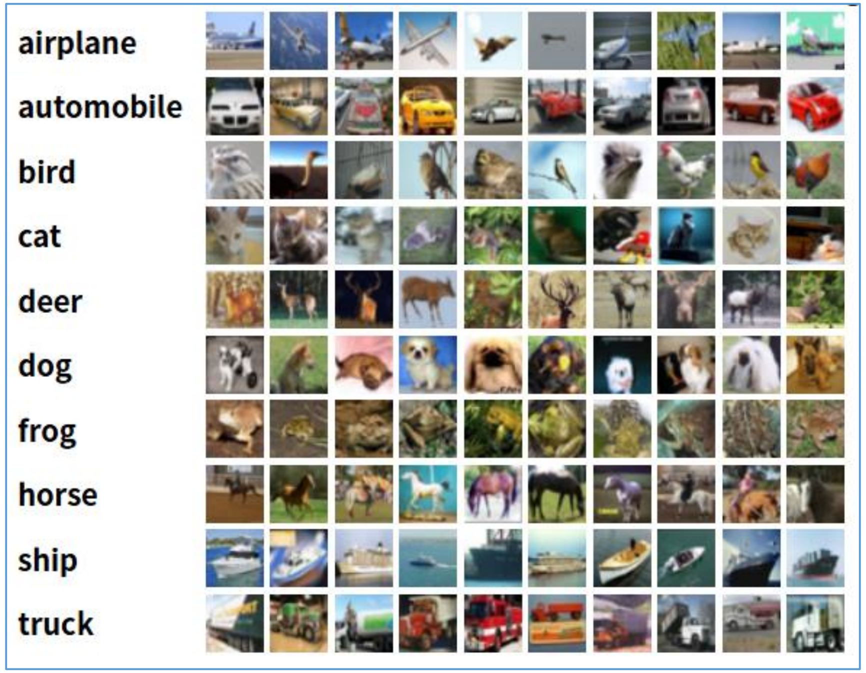
일반적인 CNN 구조



fc_3
Fully-Connected
Neural Network
ReLU activation
fc_4
Fully-Connected
Neural Network
Conv_1
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
Conv_2
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
(with
dropout)
Flattened
0
1
2
9
INPUT
(28 x 28 x 1)
n1 channels
(24 x 24 x n1)
n1 channels
(12 x 12 x n1)
n2 channels
(8 x 8 x n2)
n2 channels
(4 x 4 x n2)
n3 units
OUTPUT

CIFAR10 dataset

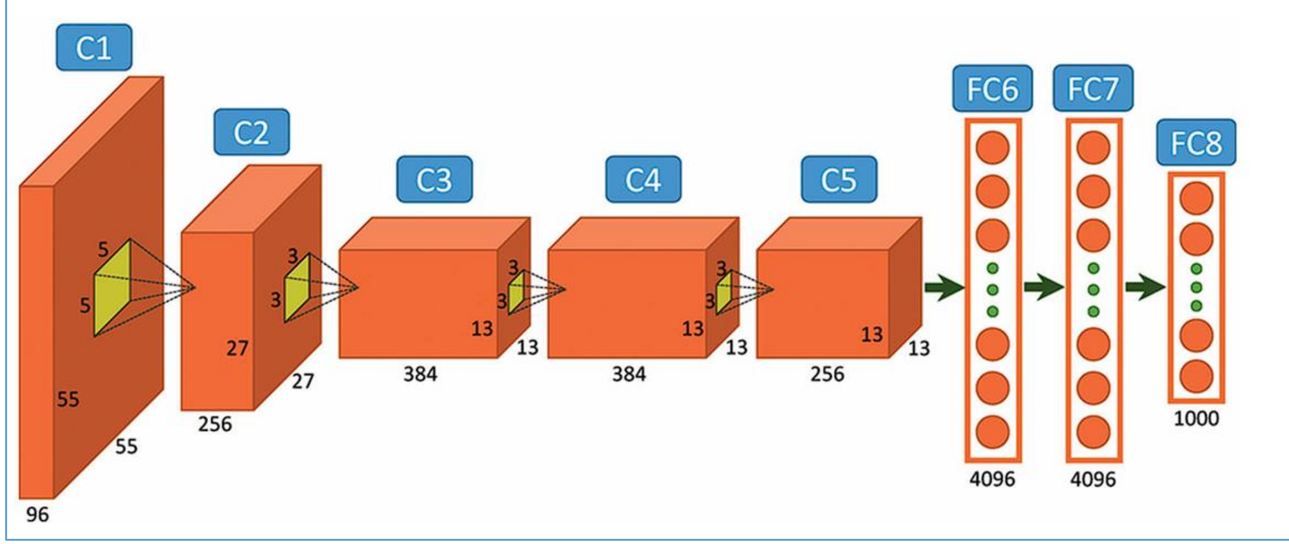
- CIFAR-10 and CIFAR-100 were created by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton (University of Toronto)
- **Data Format**:
  - Total Images: **60,000** images.
  - Training Set: **50,000** images.
  - Test Set: **10,000** images.
  - Image Dimensions: **32x32 pixels**.
  - Number of Channels: **3 (RGB)**.
  - Image Values: Pixel values are integers in the range **[0, 255]**.

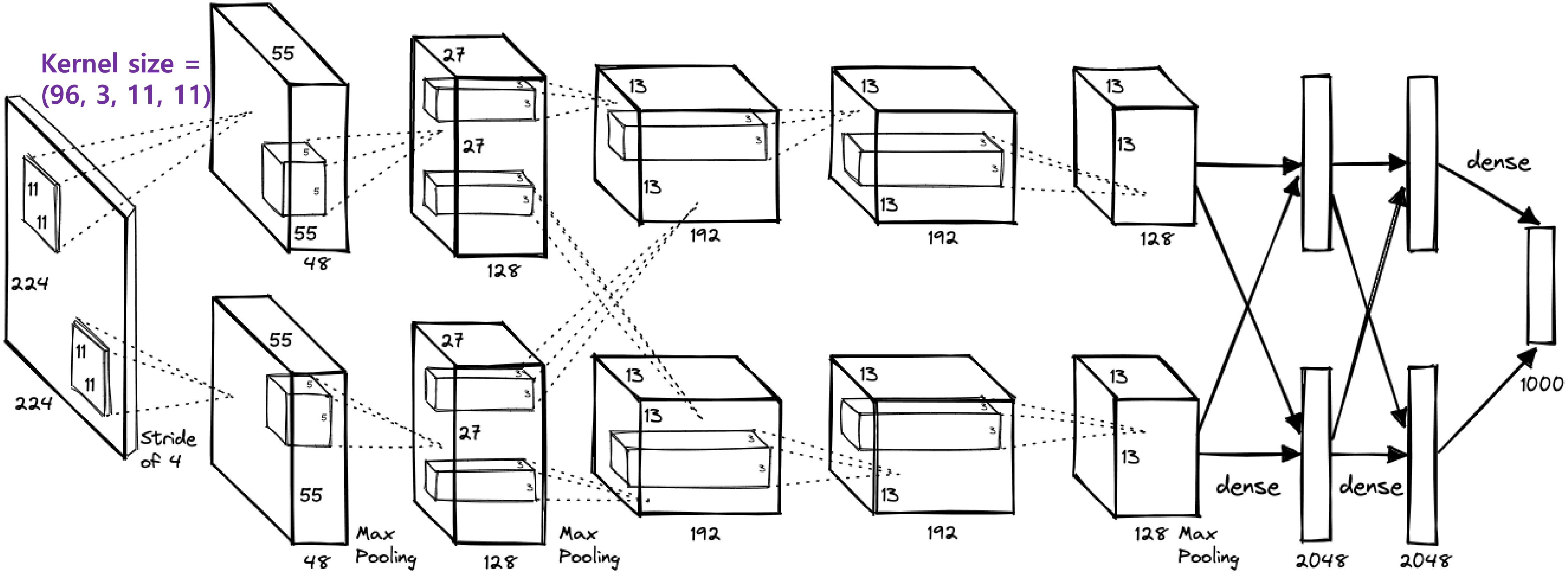


https://www.cs.toronto.edu/~kriz/cifar.html

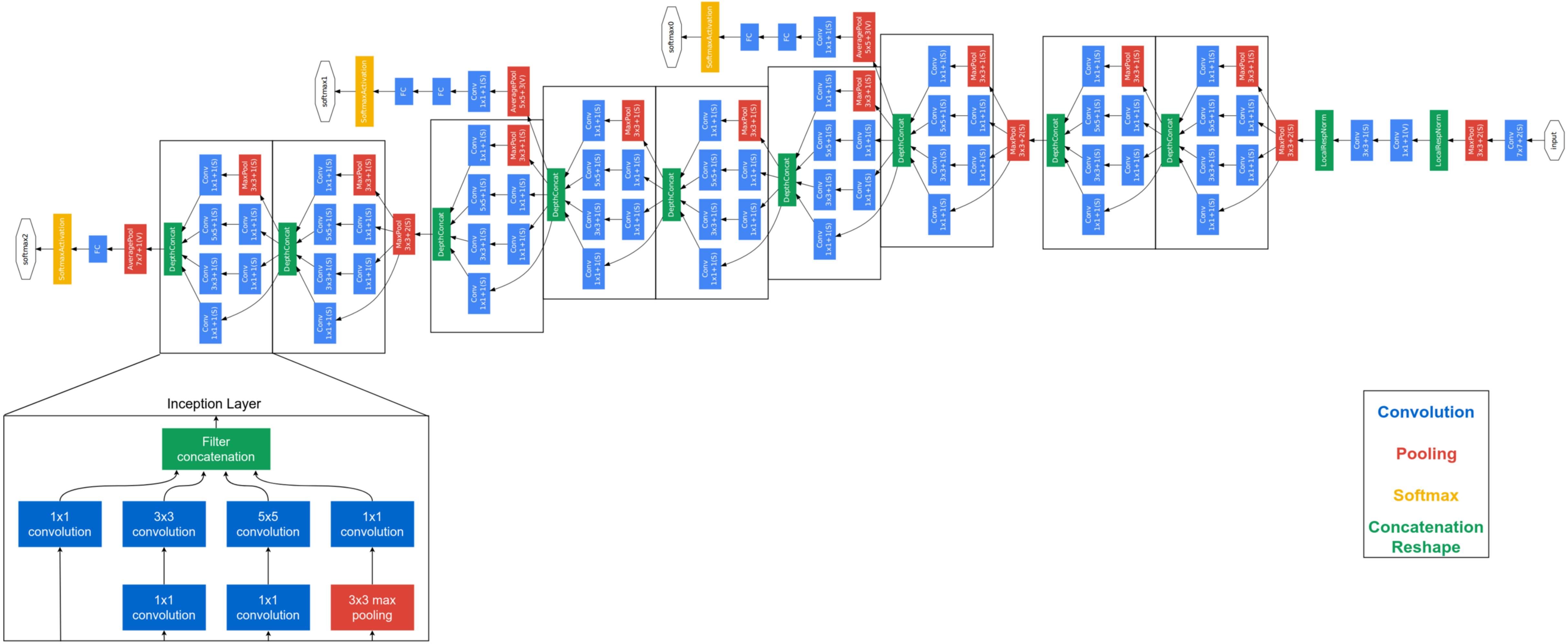
Alexnet (2012)

- *Tanh → ReLU*
- *Dropout*
- *Average pooling → Max pooling*
- *Data augmentation*
- *Local response normalization (LRN)*

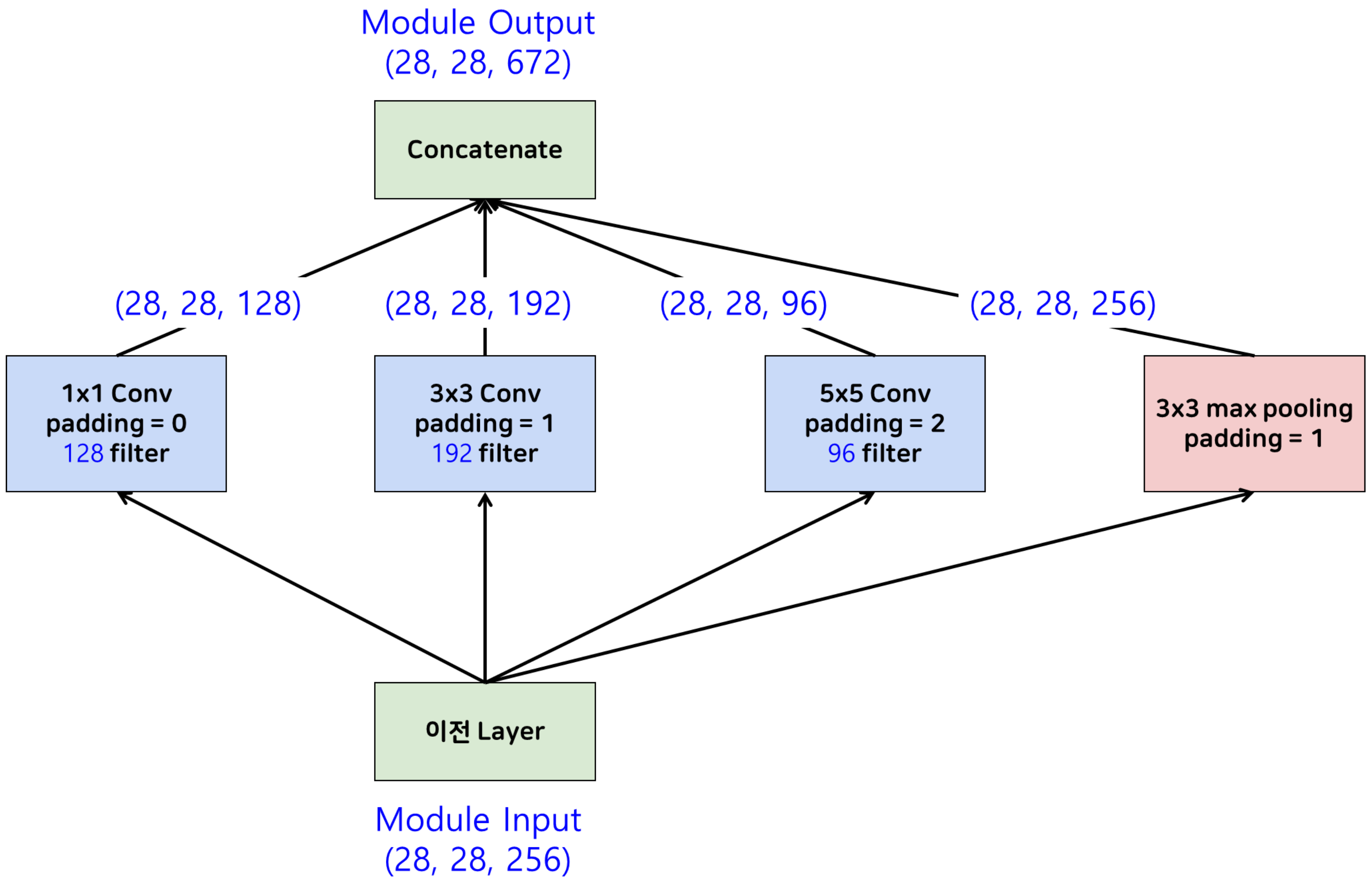




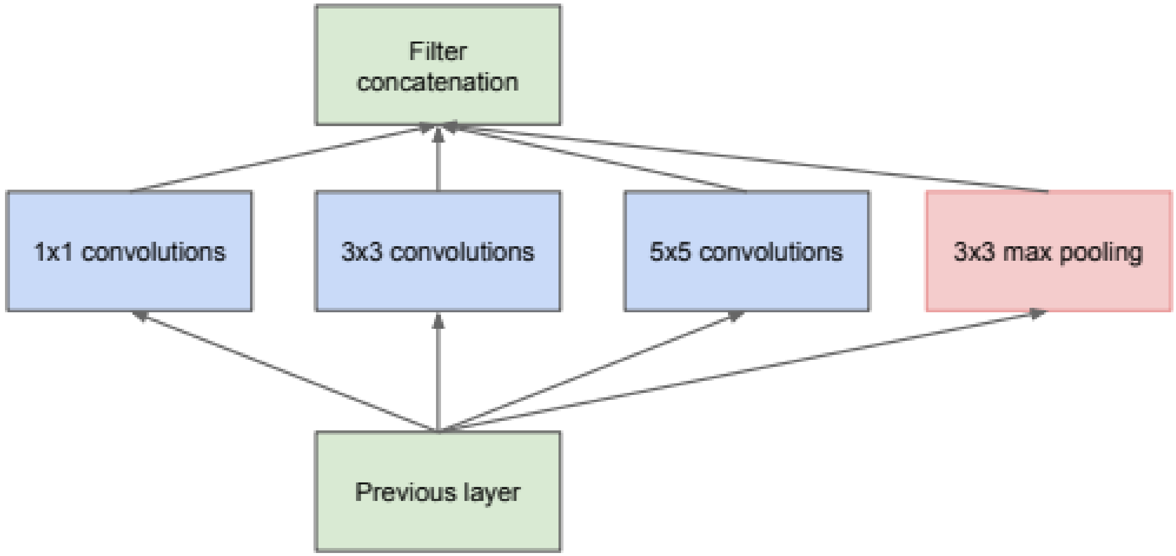
GoogleNet architecture (2014)



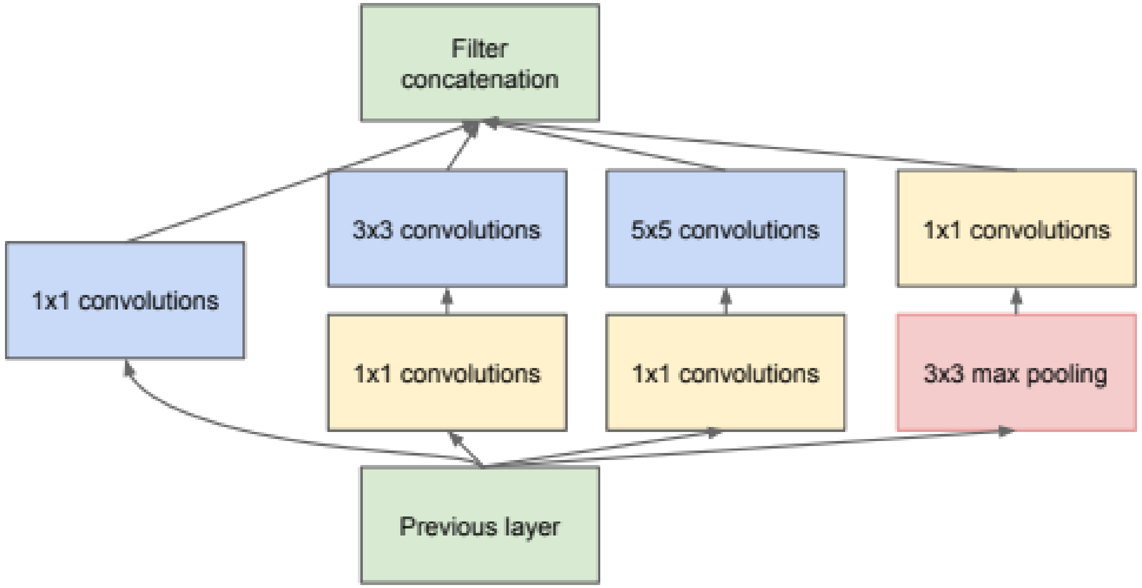
Naive Inception



GoogleNet : Inception module

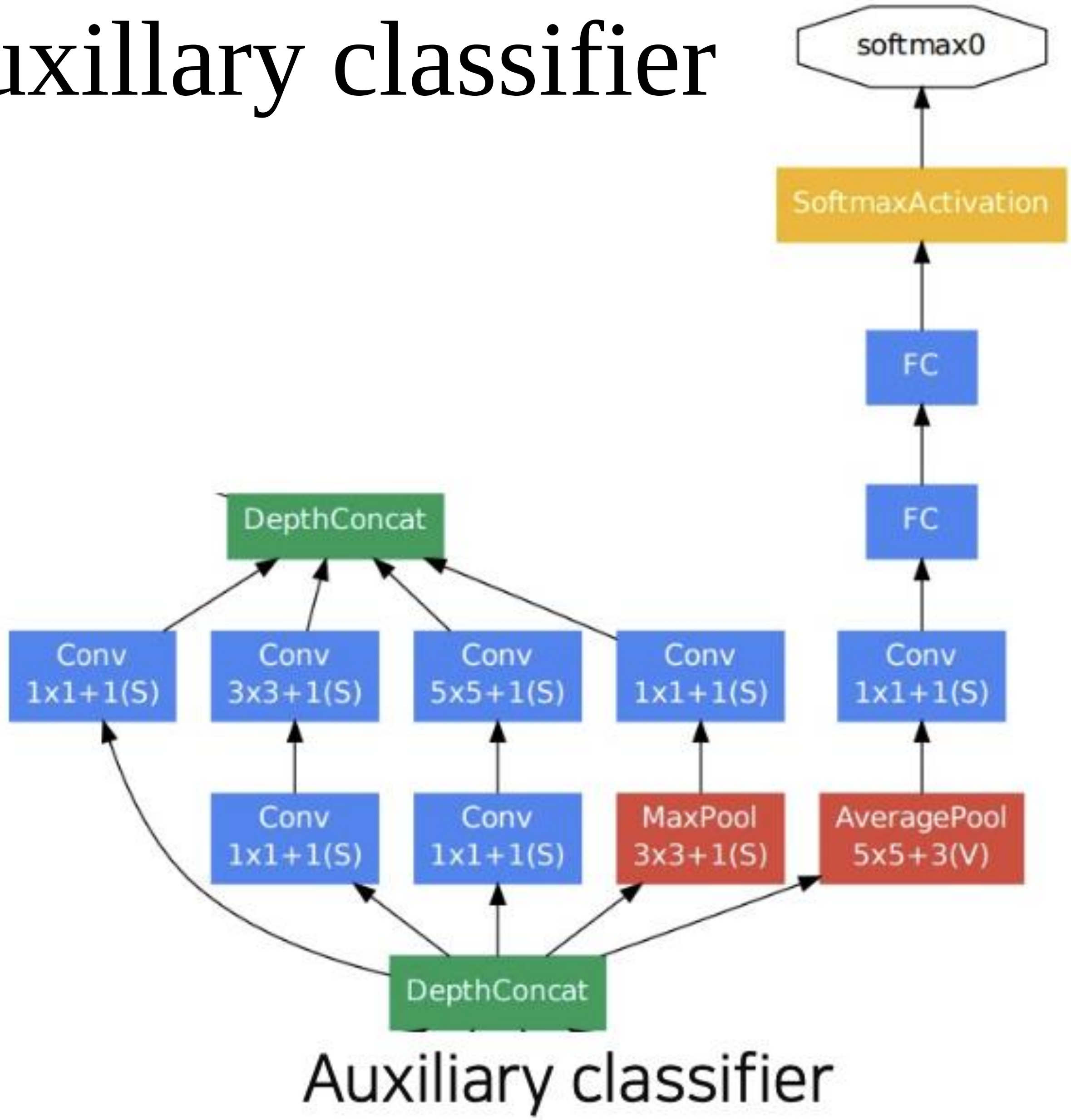


(a) Inception module, naïve version

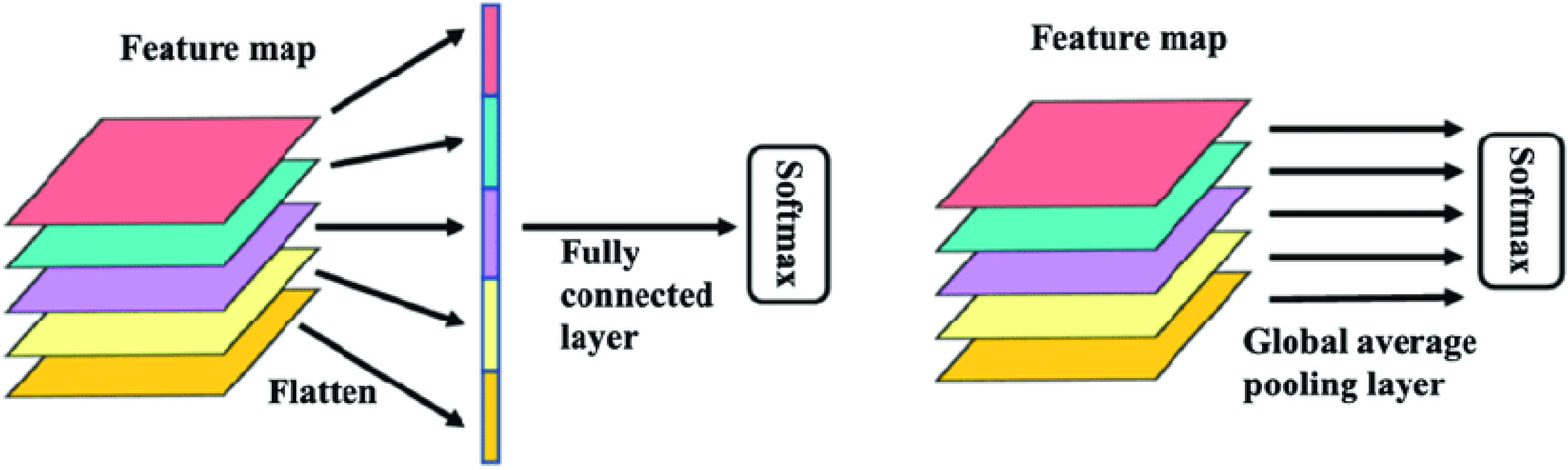


(b) Inception module with dimension reductions

GoogleNet : Auxillary classifier



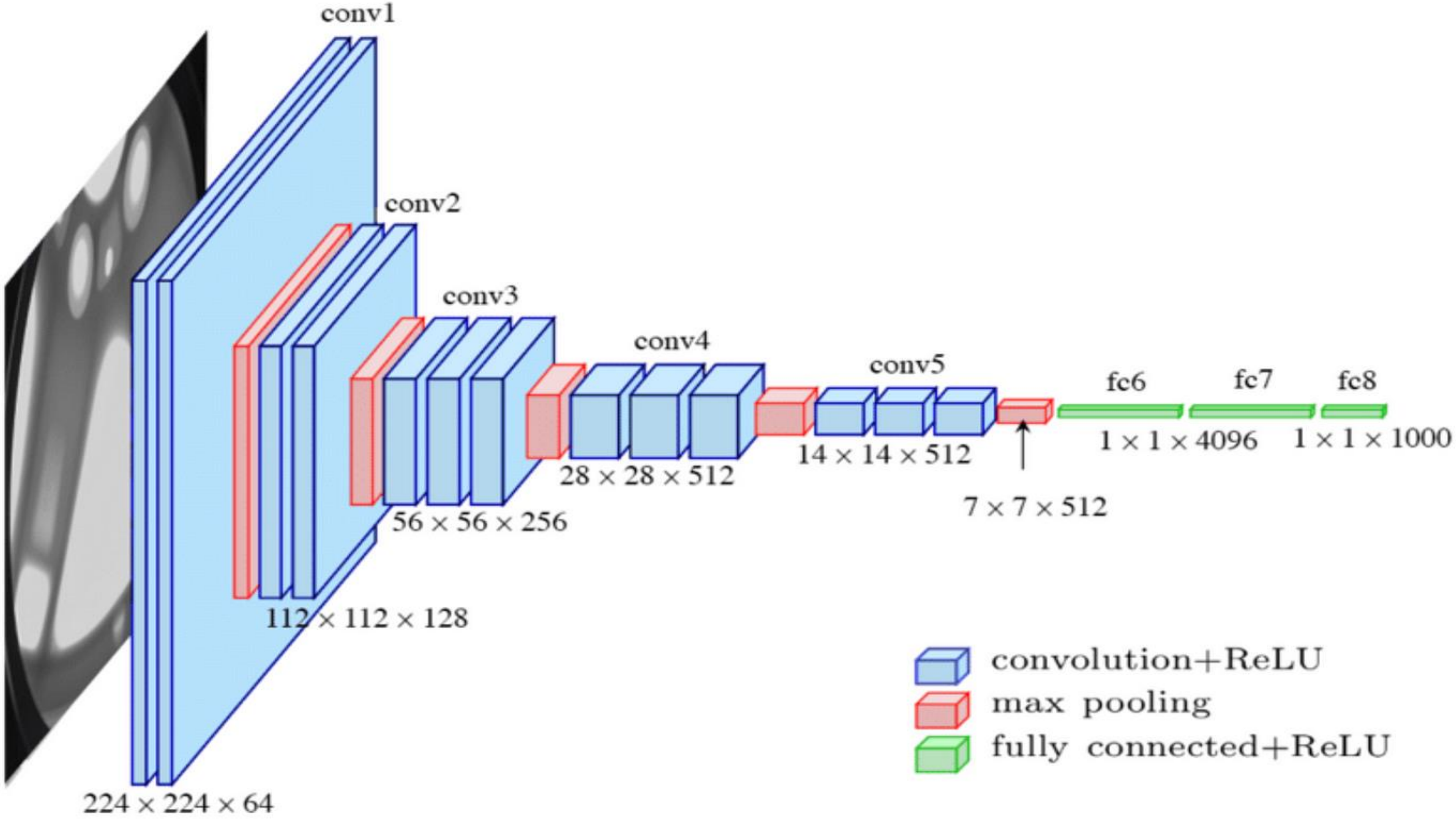
GoogleNet : global average pooling



(a) Fully connected layer

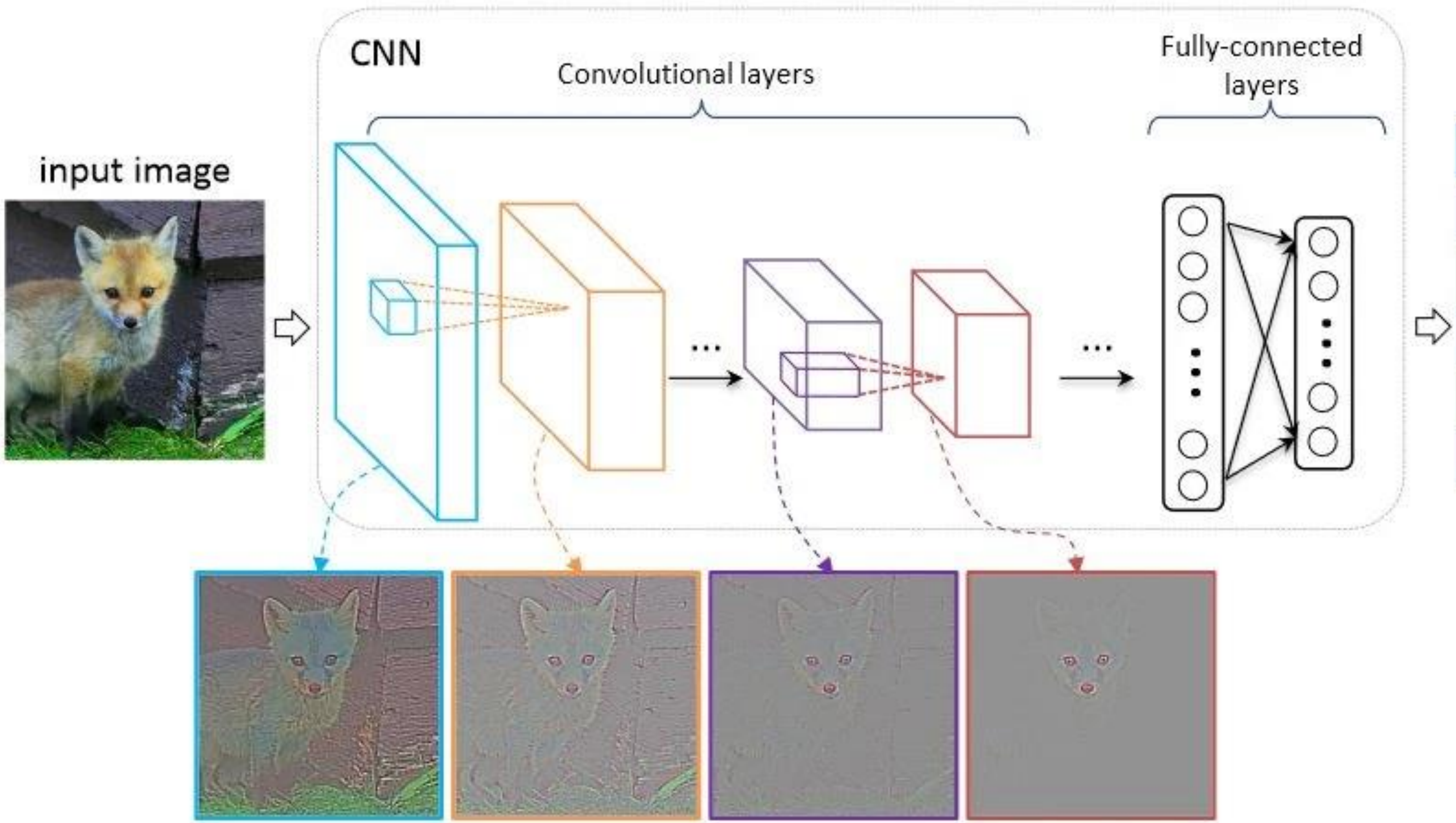
(b) Global average pooling layer

VGG16 (2014)



Title: Very Deep Convolutional Networks for Large-Scale Image Recognition (2014)

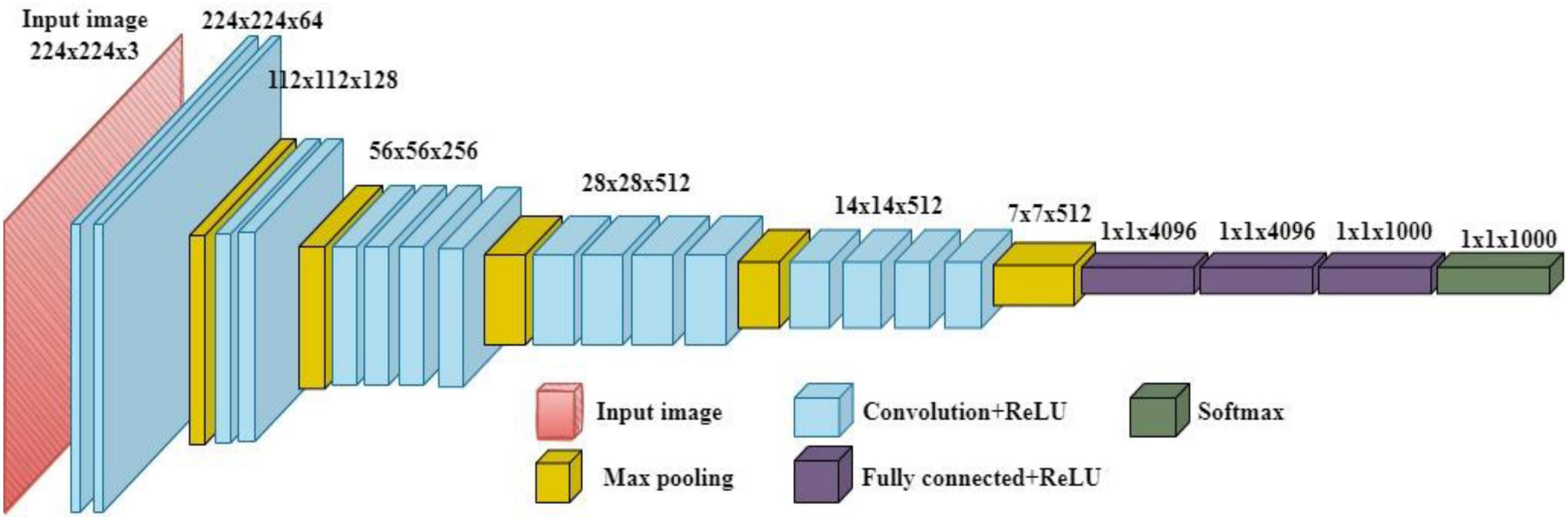
VGG16



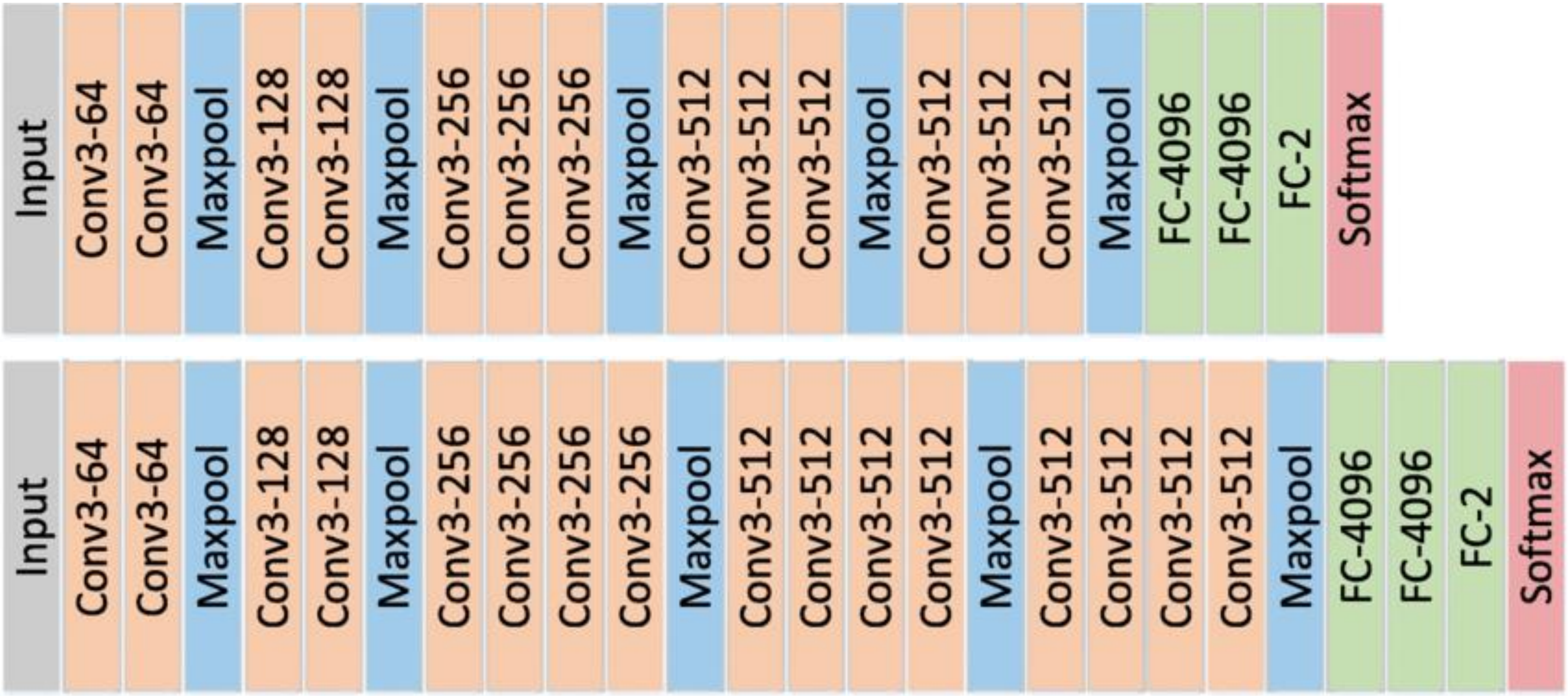
class name	prob
kit fox	0.5956
red fox	0.3576
grey fox	0.0439
coyote	0.0013
Arctic fox	0.0003

Deep Learning Bible – 2, https://wikidocs.net/book/7887

VGG19

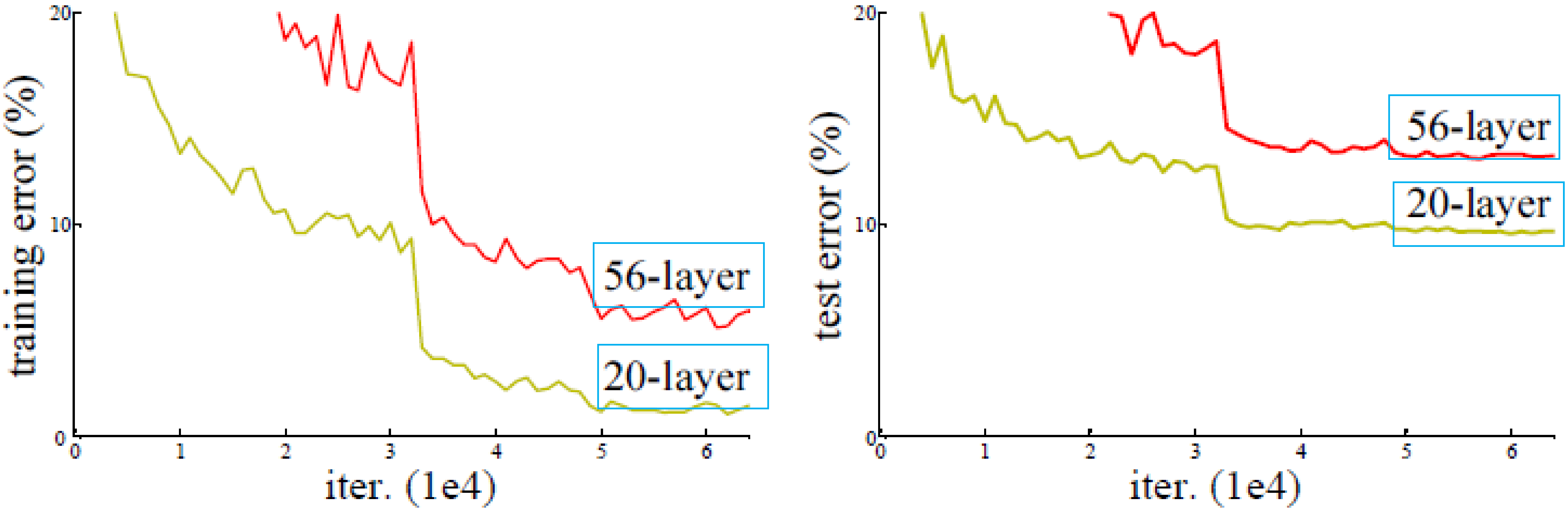

Input image
224x224x3
224x224x64
112x112x128
56x56x256
28x28x512
14x14x512
7x7x512
1x1x4096
1x1x4096
1x1x1000
1x1x1000
Input image
Convolution+ReLU
Softmax
Max pooling
Fully connected+ReLU

VGG16 vs VGG19



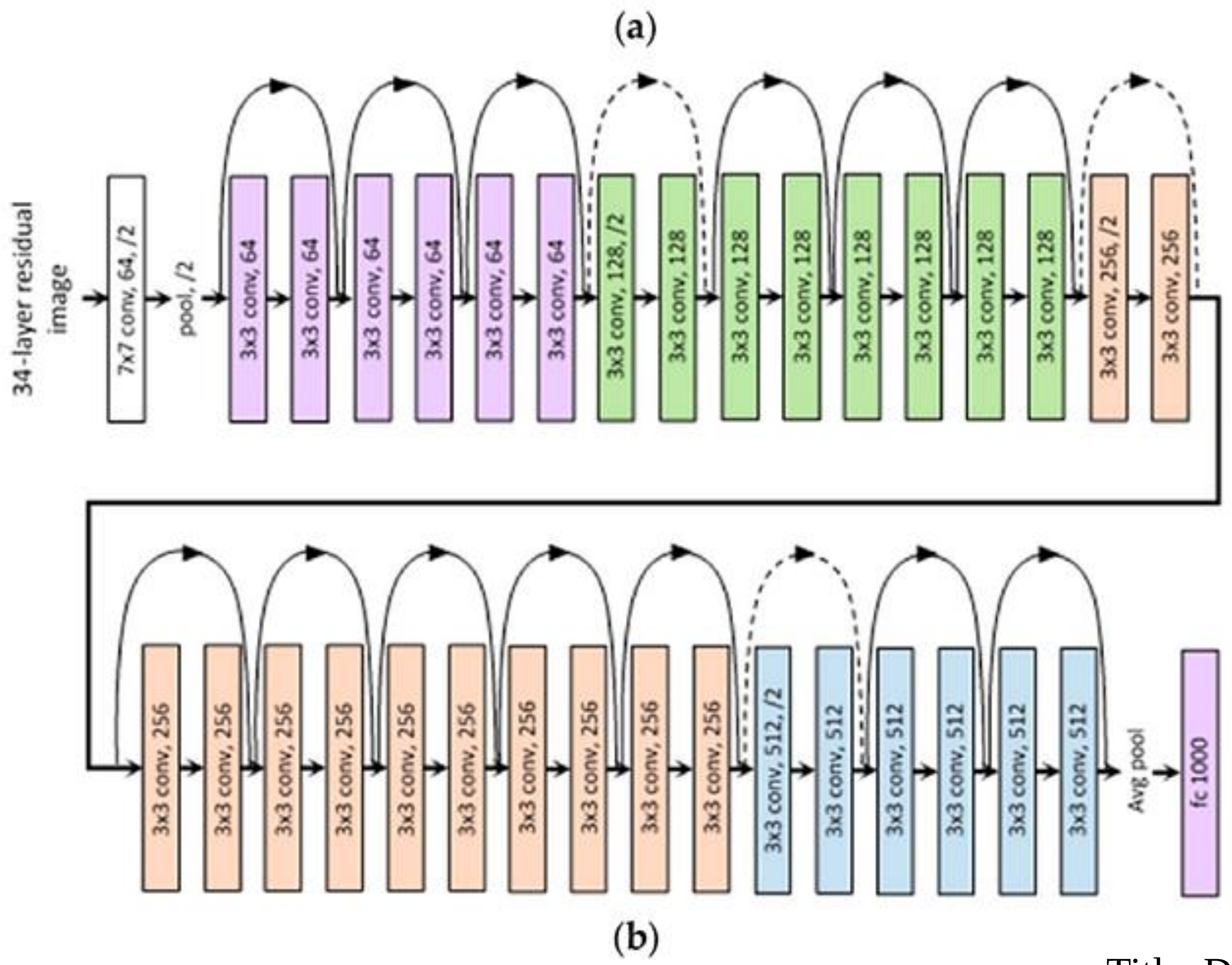
Network structures of VGG16 (top) and VGG19 (bottom)

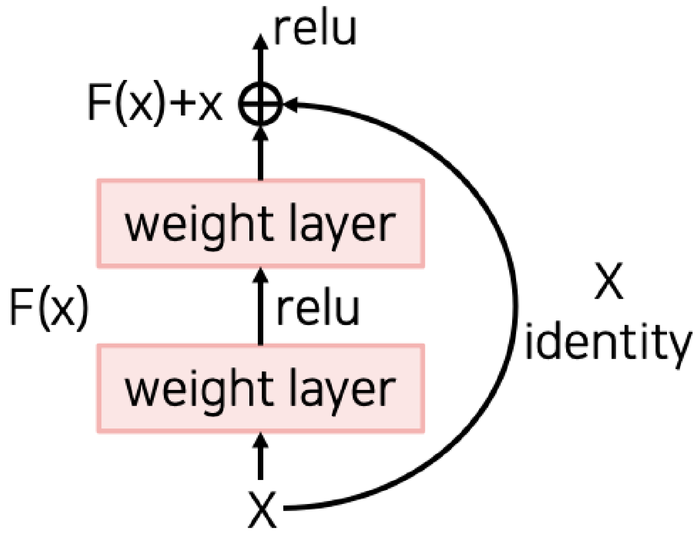
Degradation problem



Not equal overfitting
Not equal vanishing gradient

Resnet architecture (2015)



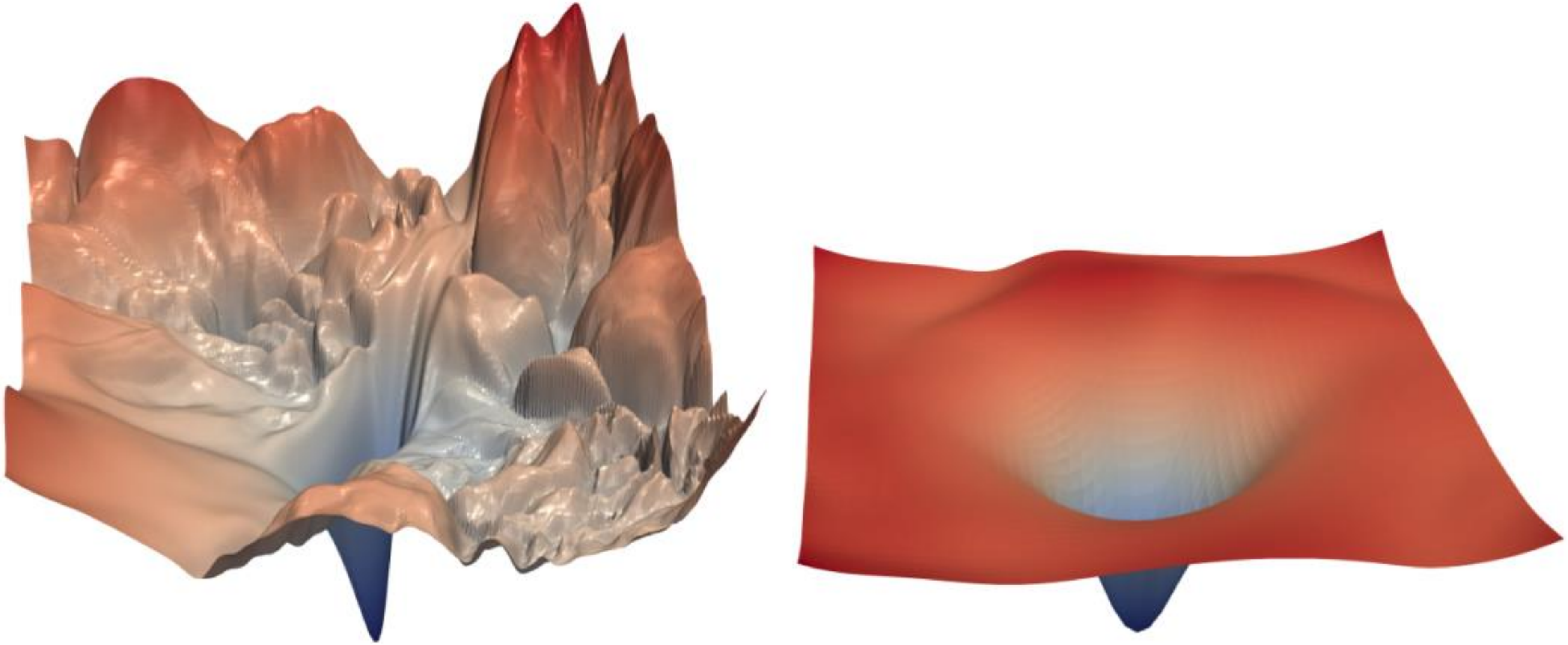


"Residual" block

Foundation

Title: Deep Residual Learning for Image Recognition (2015)

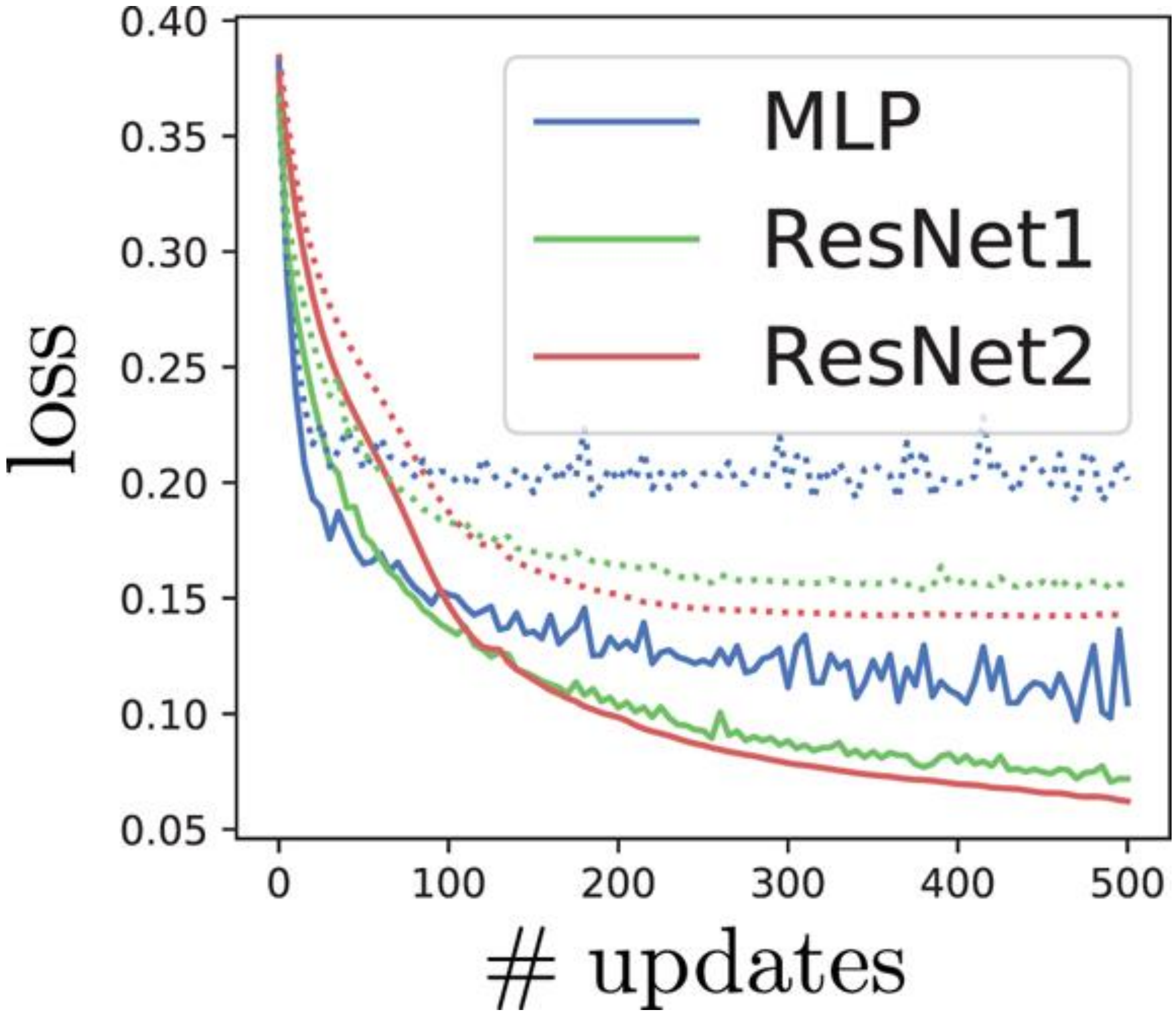
The loss surfaces of ResNet-56 with and without skip connections



(a) without skip connections (b) with skip connections

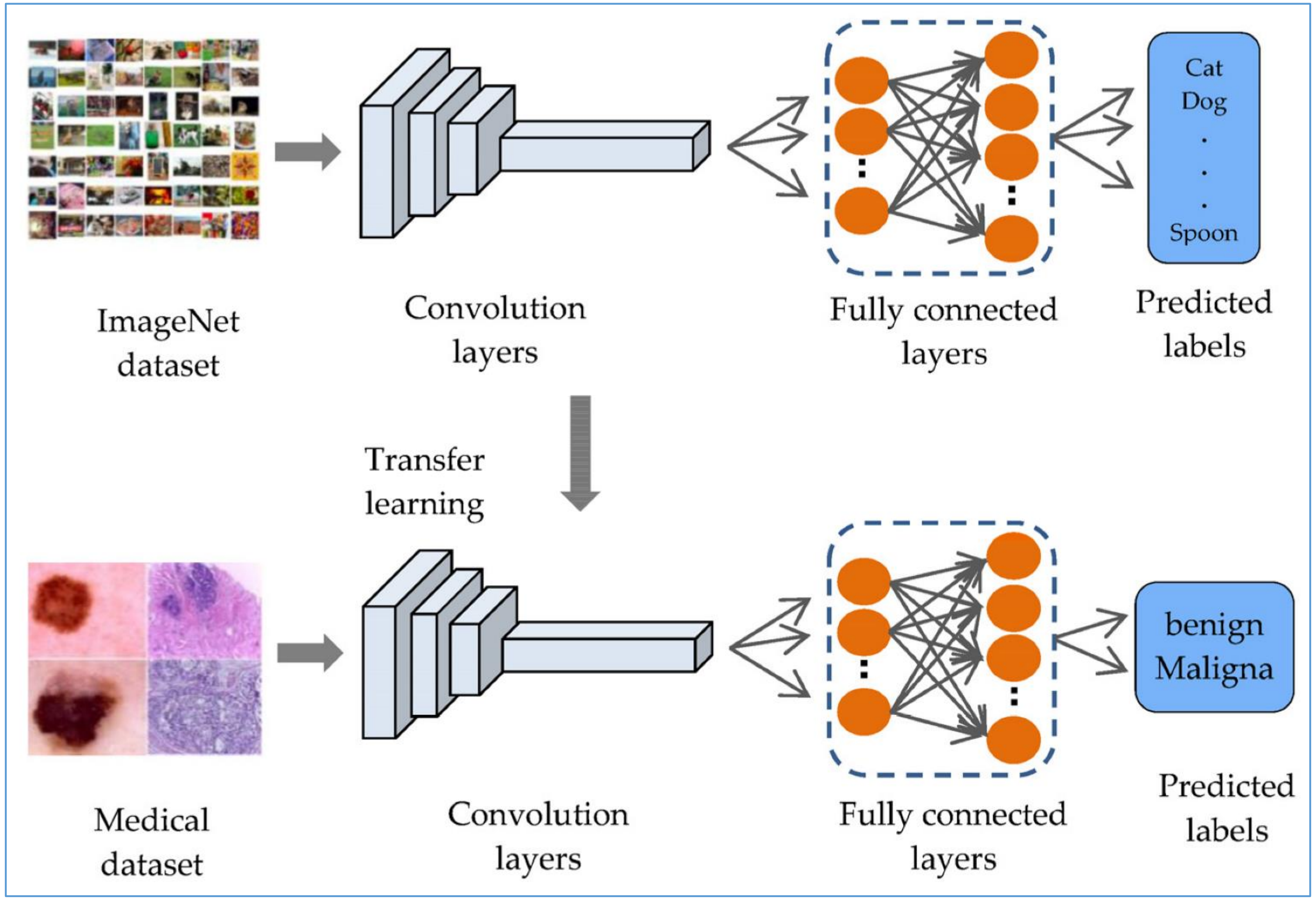
Visualizing the loss landscape of neural nets, Dec 2018

Training loss (solid lines) and **test loss** (dotted lines) of the 10-layer DNNs with the ReLU activation.



전이 학습 (Transfer learning)

- 기존의 모델이 학습한 특징 (Feature)를 새로운 모델에서 재사용하여 학습 효율성을 높이고 데이터 요구량을 줄이는 것



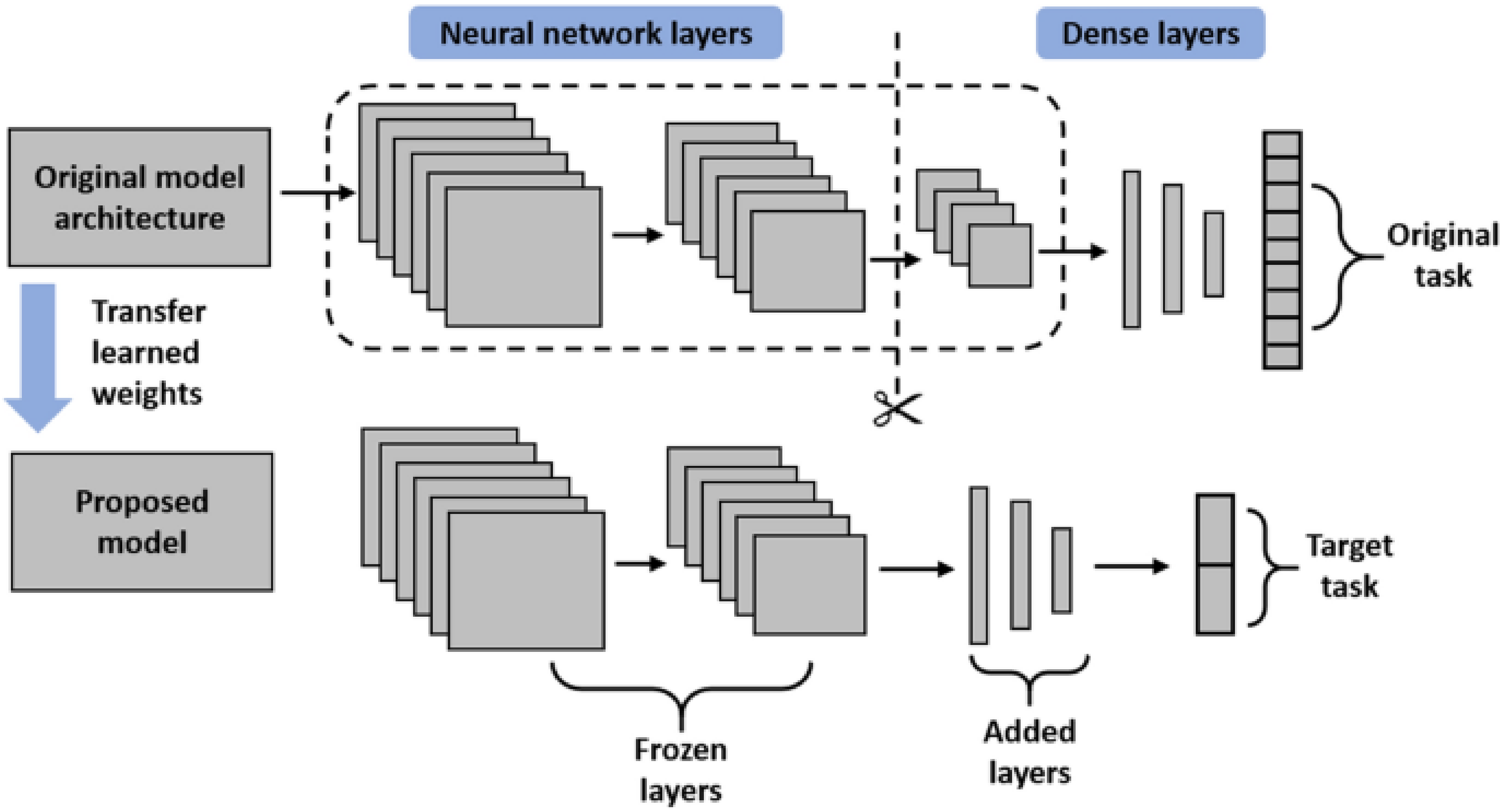
전이 학습의 특징

- 특징 재사용
  - 기존에 학습된 모델이 생성한 특징(Feature)을 새로운 태스크에서 초기 가중치로 사용
- 적은 데이터 요구
  - 전이 학습을 통해 새로운 태스크에서 대량의 데이터 없이도 좋은 성능을 달성 할 수 있음

전이 학습의 장단점

- 장점
  - 데이터 부족 문제 해결
  - 학습 시간 단축
  - 더 나은 초기화로 학습 최적화 가능
- 단점
  - 원본 태스크와 새 태스크 간의 유사성이 낮으면 효과가 제한적
  - 큰 사전 학습 모델은 컴퓨팅 자원을 많이 요구

일반적인 전이 학습 구조


Neural network layers
Dense layers
Original model architecture
Original task
Transfer learned weights
Proposed model
Target task
Frozen layers
Added layers

파인 튜닝 (Fine-tuning)

구분	전이 학습	파인튜닝
적용 범위	일반적인 개념, 특정 태스크가 아님	특정 태스크에 최적화된 추가 학습 단계
가중치 변경 여부	가중치를 고정하거나 특정 레이어만 사용	가중치를 미세 조정
목적	기존 지식 활용 및 새로운 태스크 초기화	새로운 태스크에 맞춘 성능 극대화
작업 범위	사전 학습 모델의 "기반" 사용	모델의 세부 "조정"