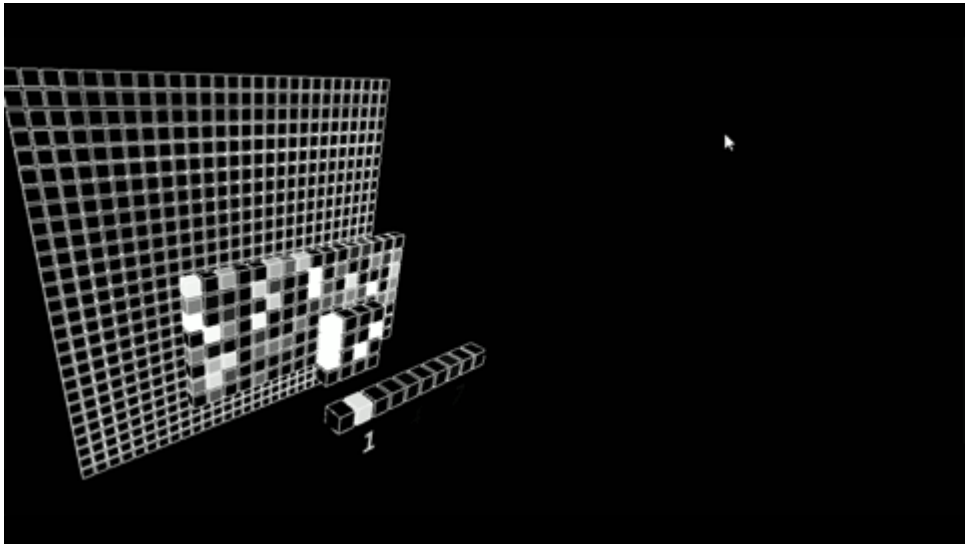
M8. Convolutional Neural Network

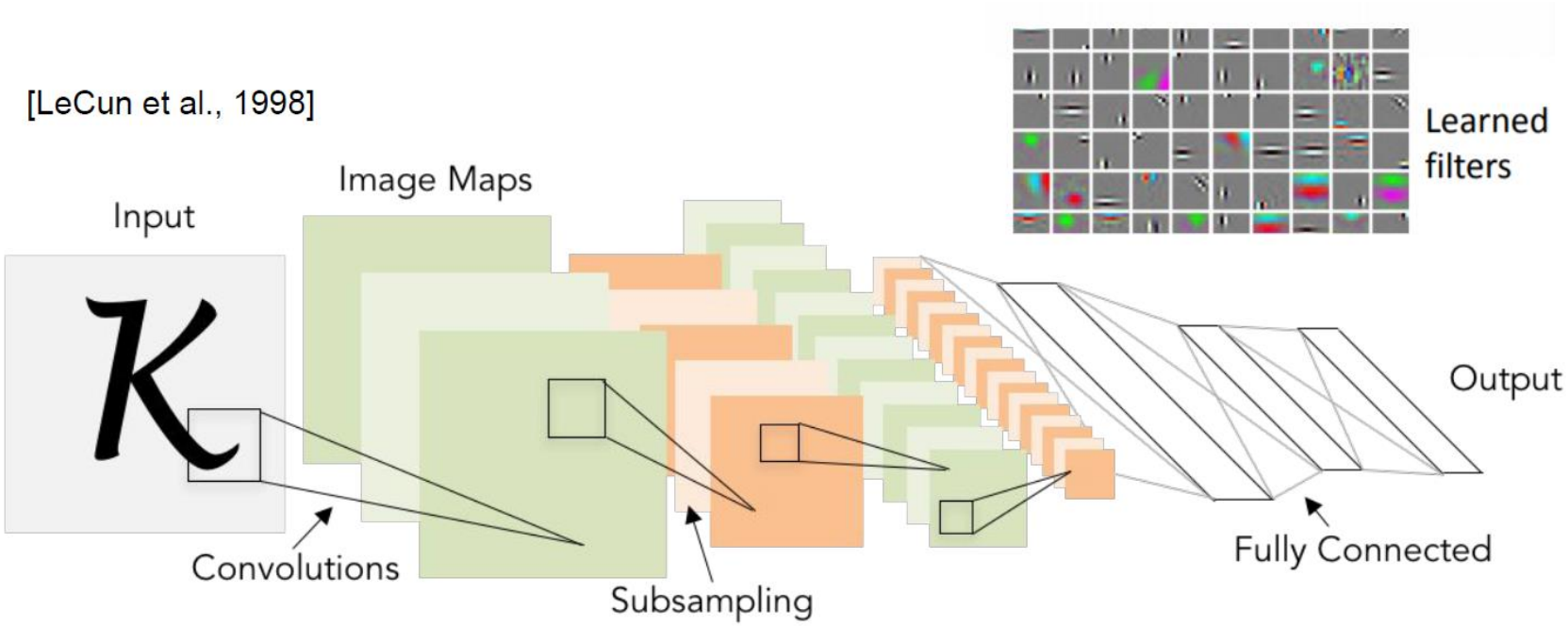


이미지 처리에 특화된 네트워크

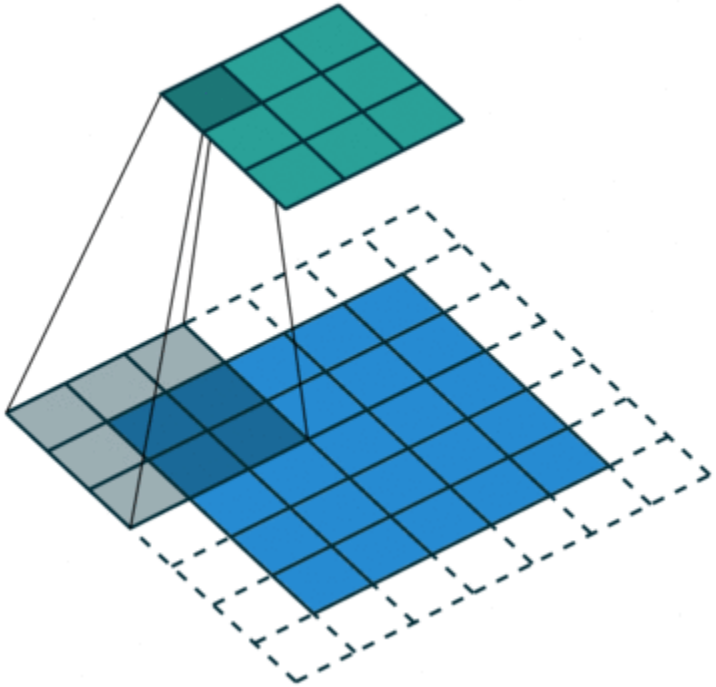
- Convolution
- Padding
- Pooling

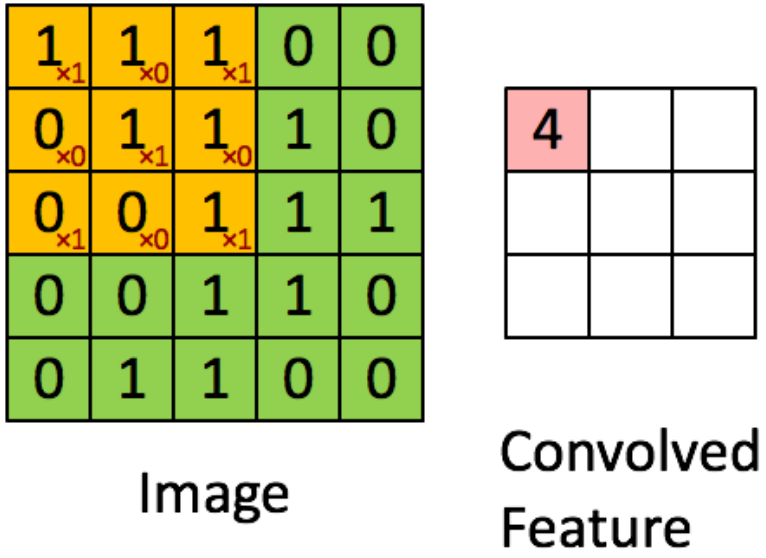
Convolutional Neural Networks 정의

Convolution + Subsampling(Pooling) + Full Connection

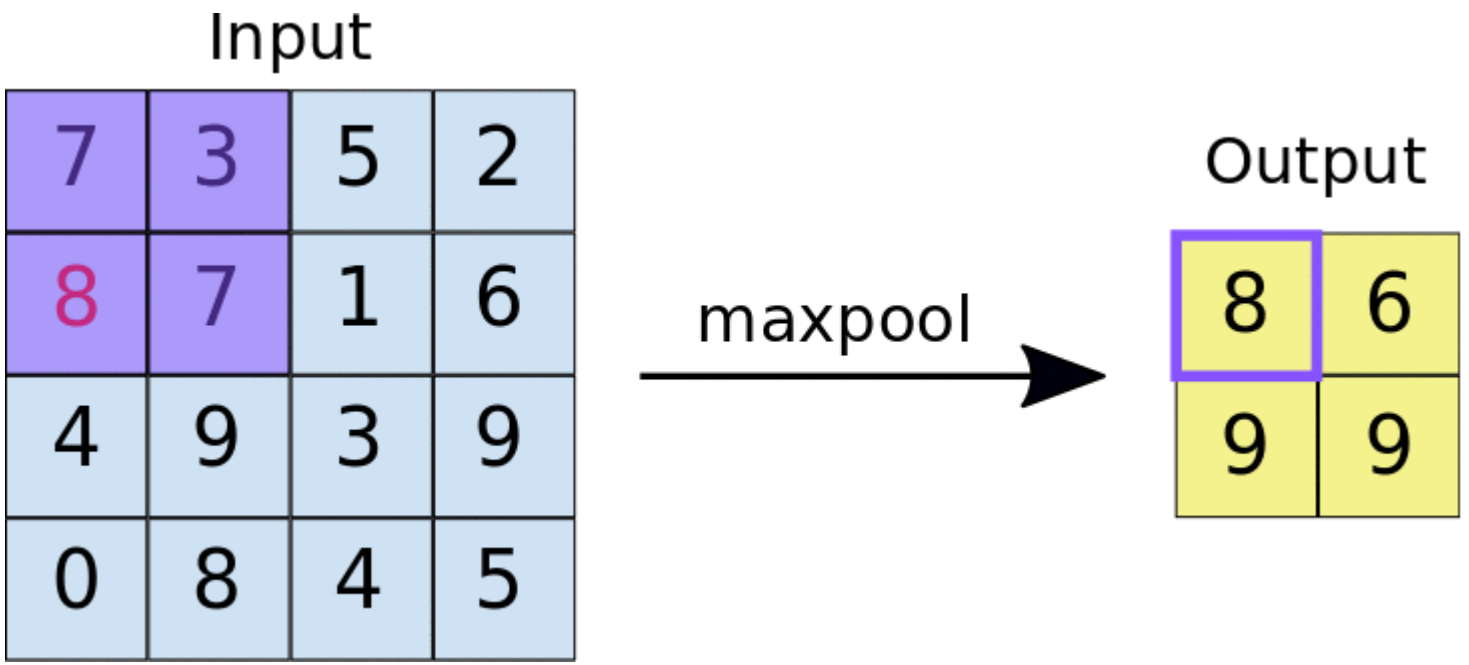


이미지로부터 특징을 추출하는 Convolution 연산





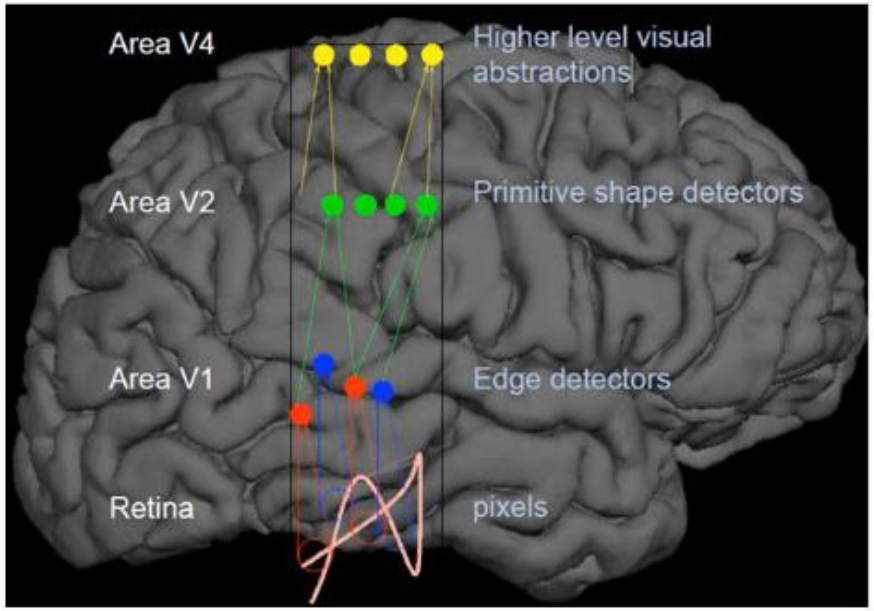
차원은 축소하는 Subsampling(Pooling)

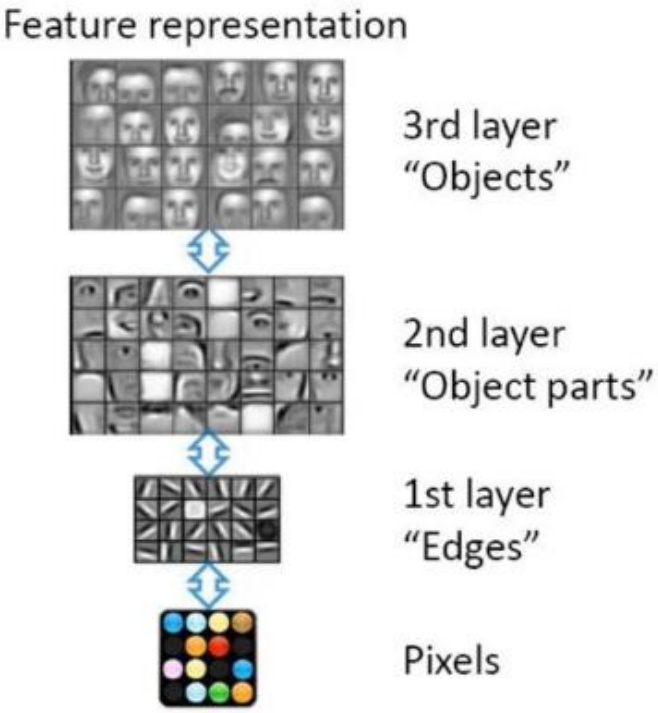


Convolutional Neural Networks (a.k.a CNN, ConvNet)

CNN을 쓰는 이유?

- 생물학적인 방법에서 고안
- 계층적으로 표현(representations)을 학습 (features)

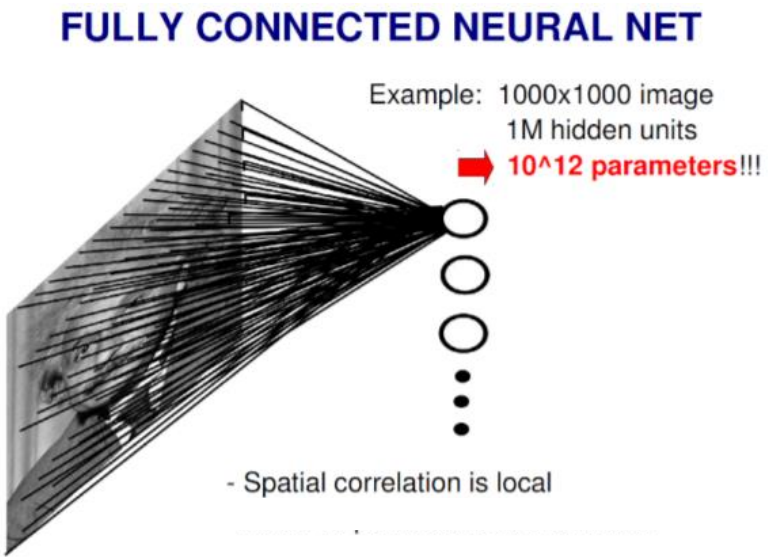


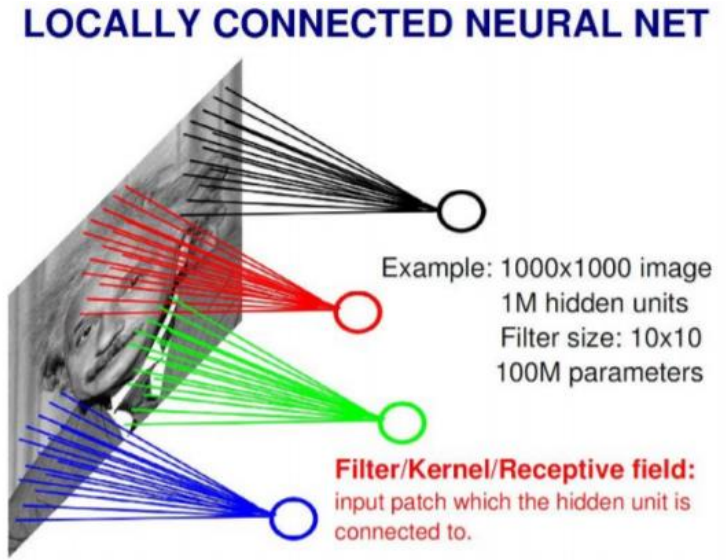


Convolutional Neural Networks (a.k.a CNN, ConvNet)

CNN을 쓰는 이유?

- 부분적으로 인식(Receptive field)하는 것으로 공간구조를 유지
- Convolution 연산은 이미지의 공간적인 부분 상관관계 득성을 이용
  - 사람의 눈이 인식하는 것처럼 득정 위치와 해당 주변부에 집중

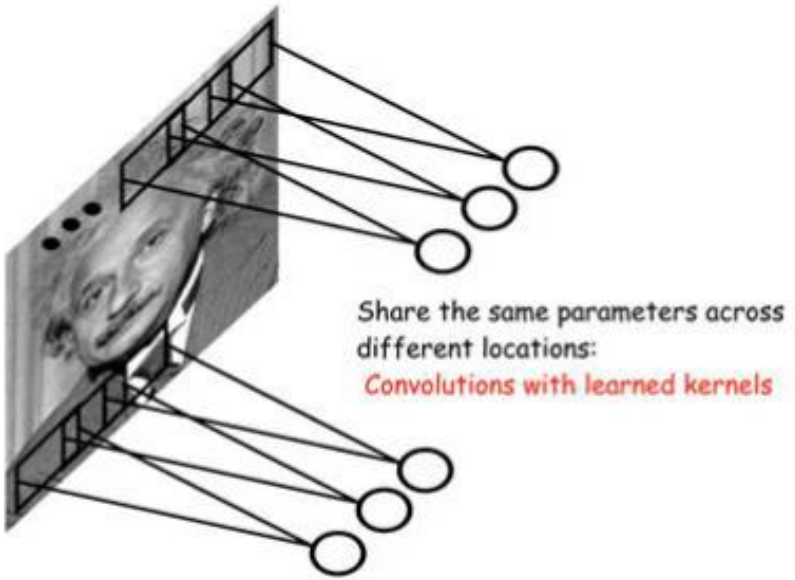


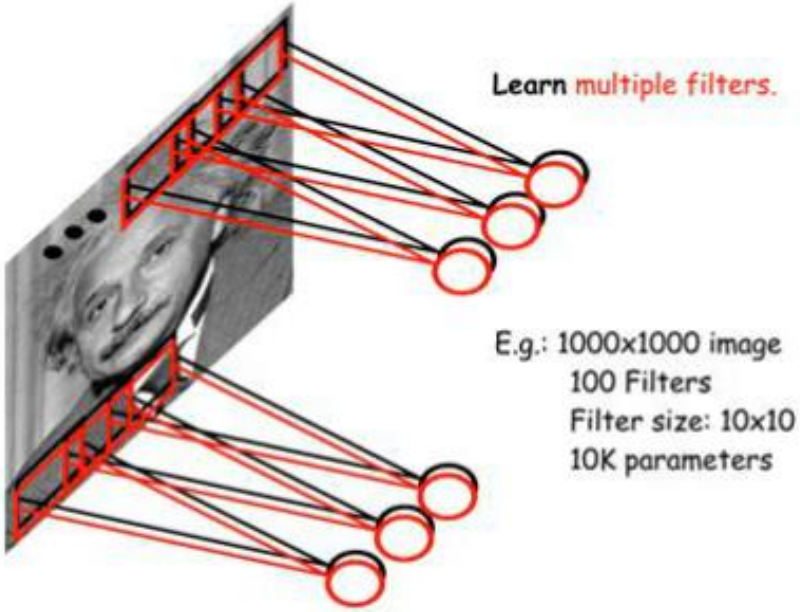


Convolutional Neural Networks (a.k.a CNN, ConvNet)

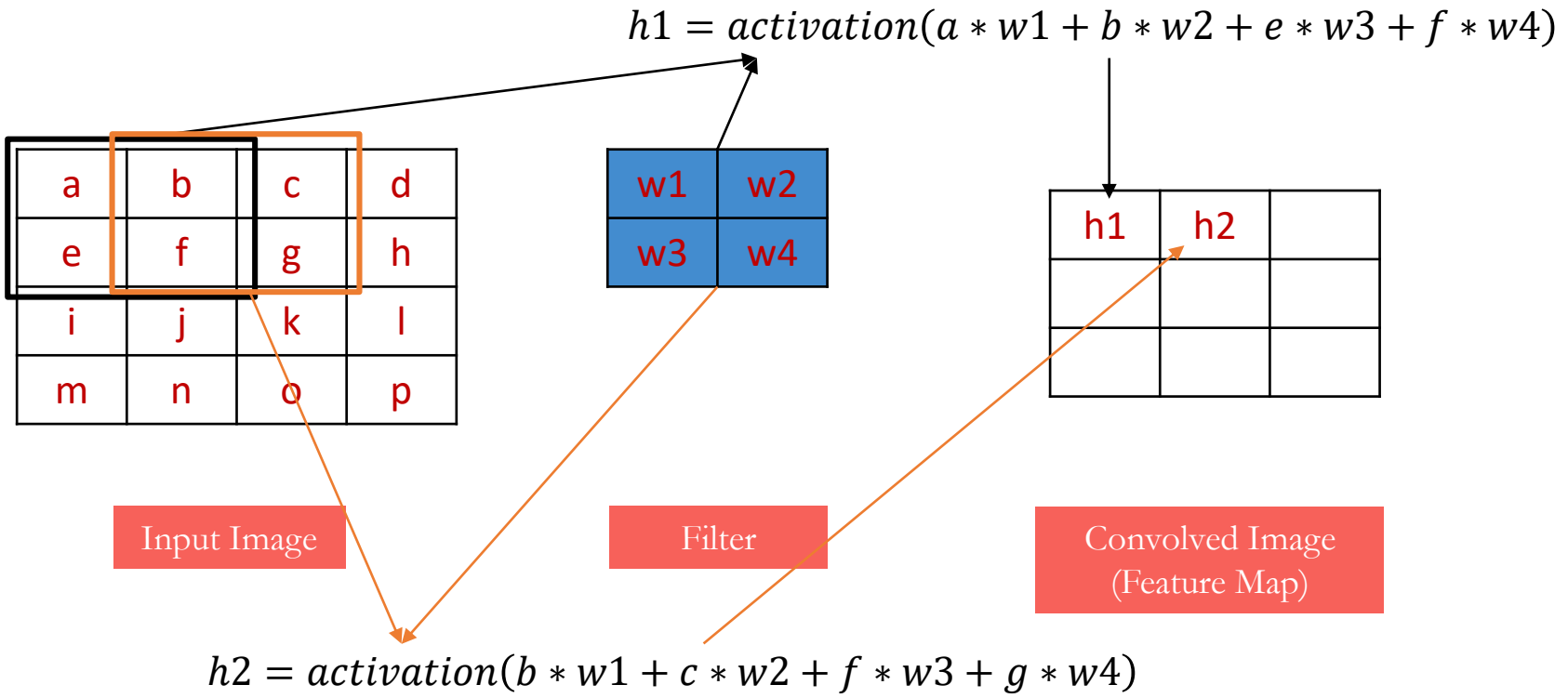
CNN을 쓰는 이유?

- Weight를 공유
- FCN에 비하여 parameter 수가 크게 감소하여, overfitting을 줄여줌





Convolution 연산은 적은 수의 parameter 만을 필요로 한다

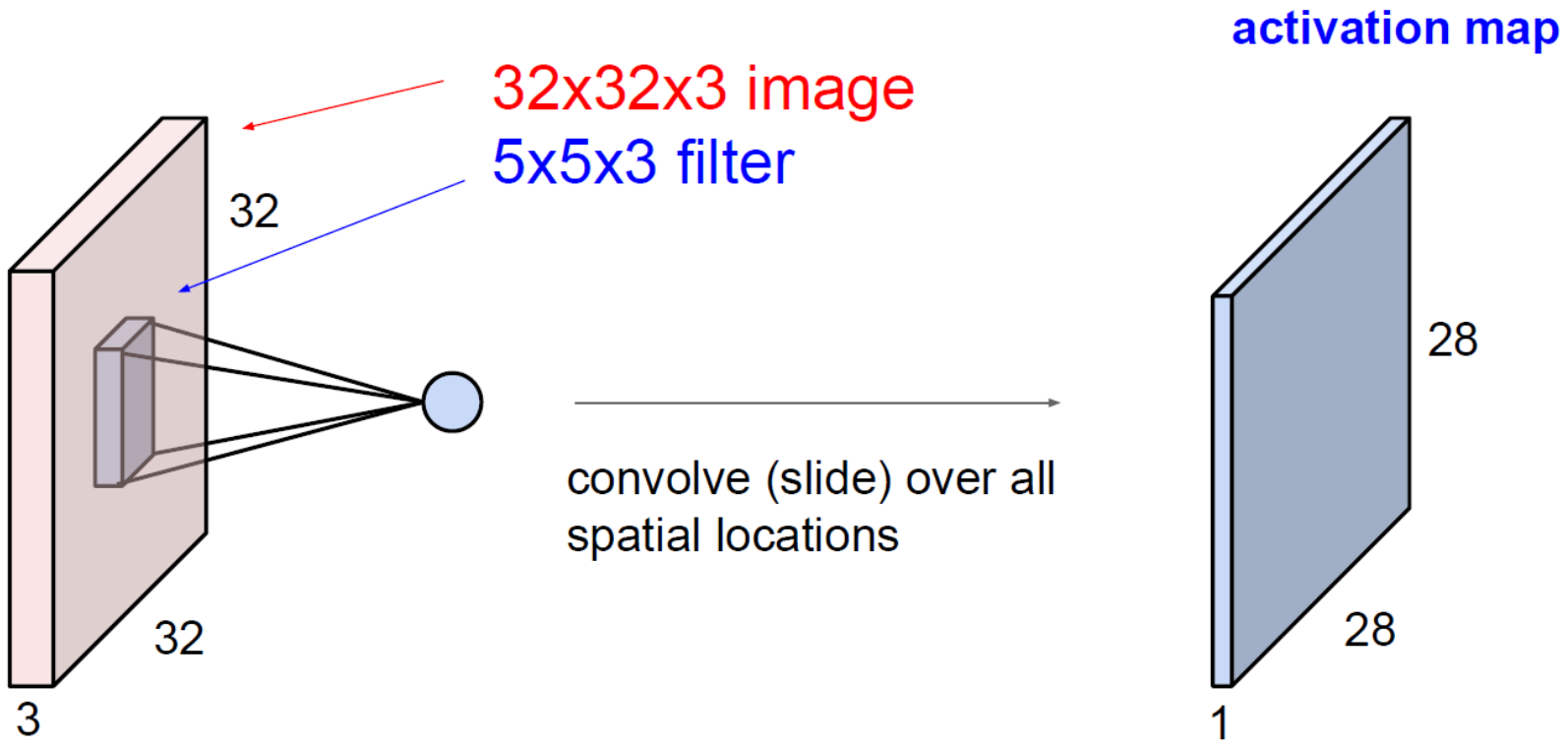


Number of Parameters for one feature map = 4

Number of Parameters for 100 feature map = 4*100

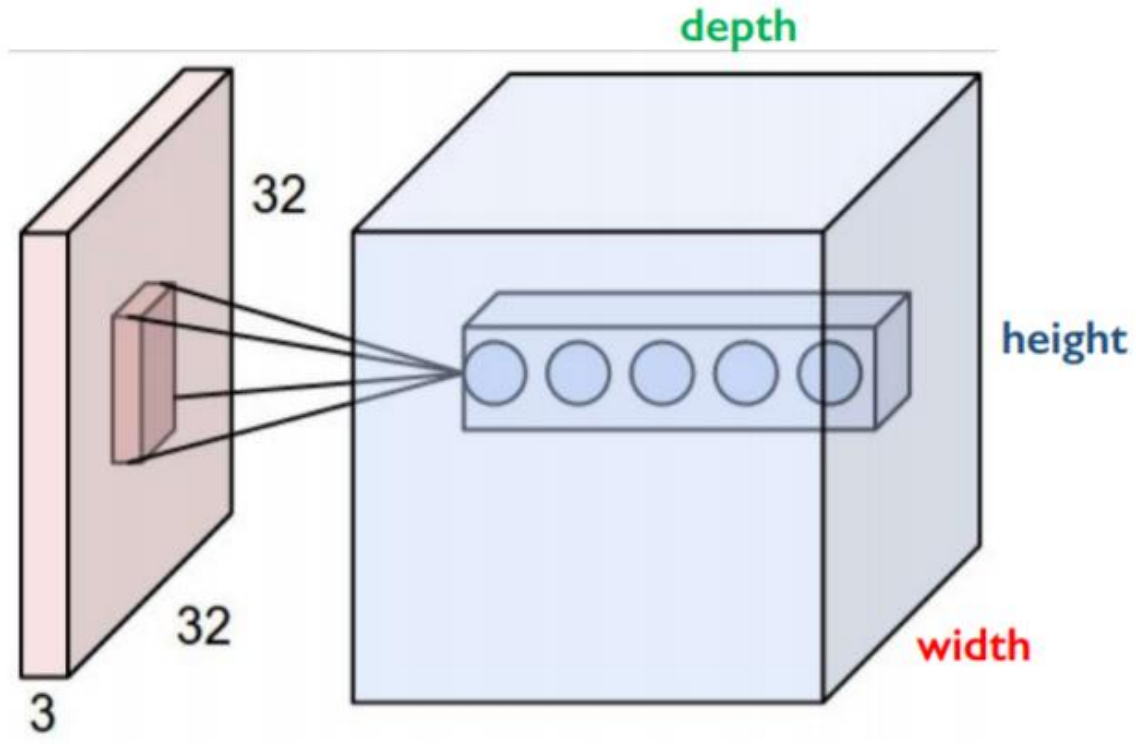
※ 이 두 경우 bias를 적용하지 않았습니다

Convolution은 통해 feature를 추출



출처: CS231n_2018_lecture05

feature를 여러 장 추출하면 이미지로부터 풍부한 정보를 꺼낼 수 있다



Layer Dimensions:

h x w x d

where h and w are spatial dimensions
d (depth) = number of filters

Stride:

Filter step size

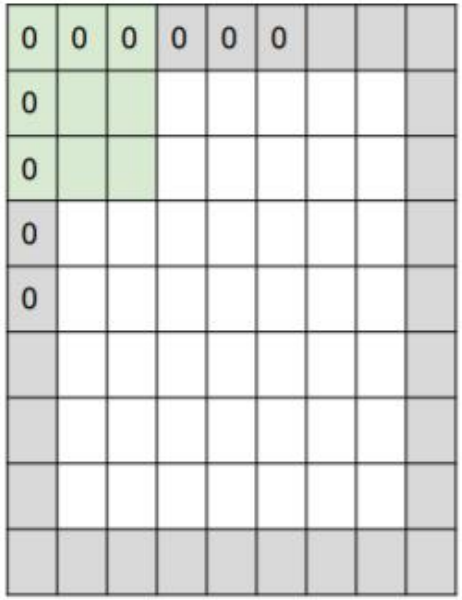
Receptive Field:

Locations in input image that
a node is path connected to

출처: CS231n_2018_lecture05

Padding : 이미지 주변에 계산과는 무관한 테두리를 덧붙여 output의 사이즈를 조정

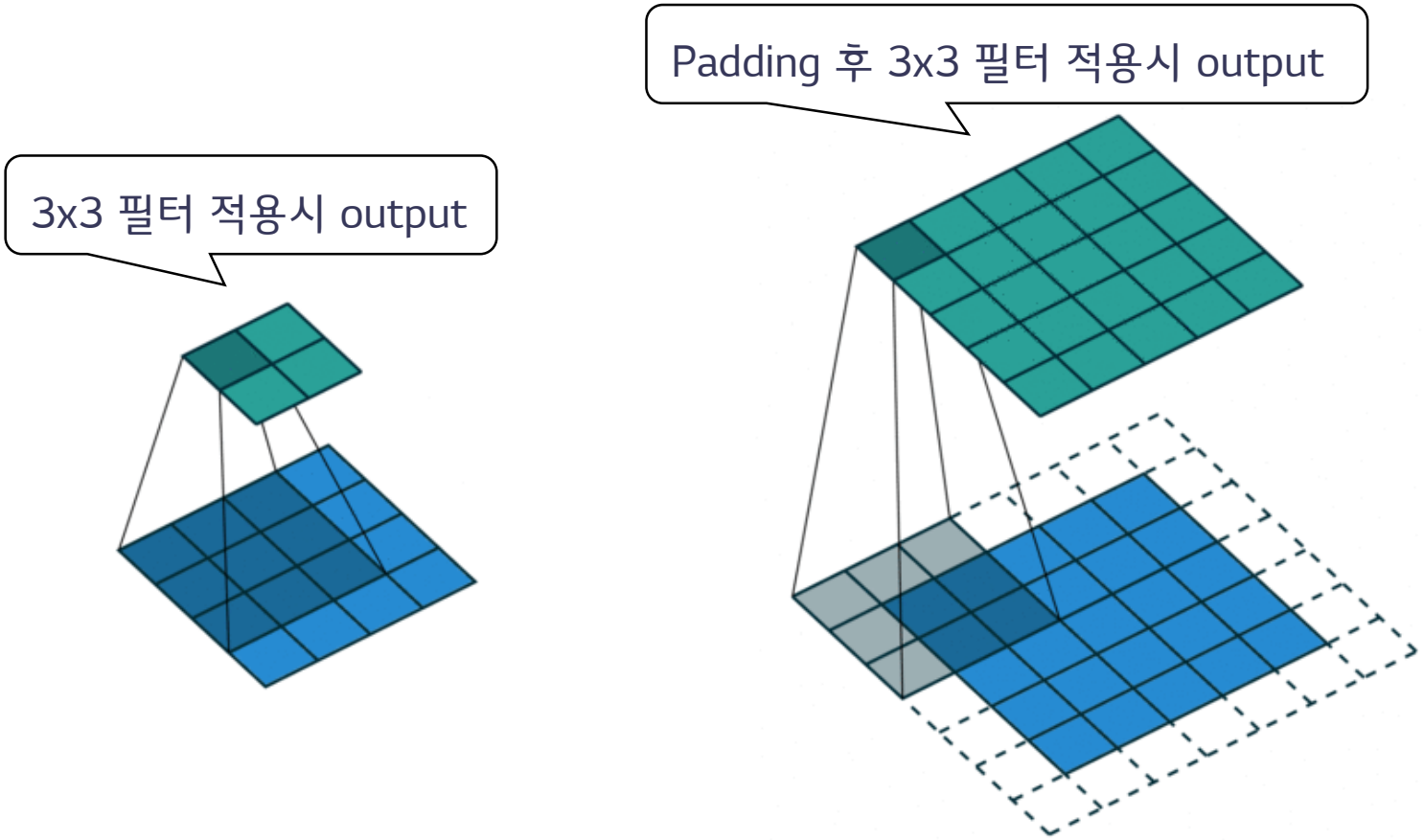
In practice: Common to zero pad the border



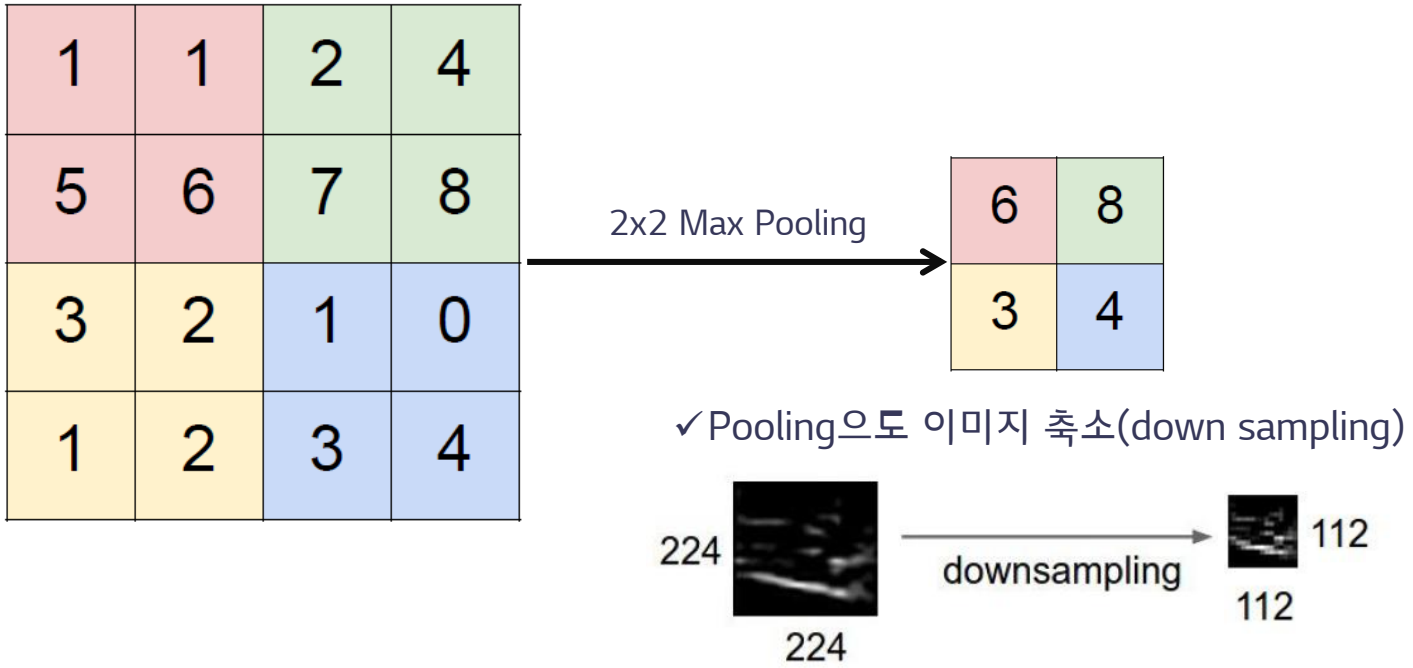
e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(recall:)
(N - F) / stride + 1

Padding : 이미지 주변에 계산과는 무관한 테두리를 덧붙여 output의 사이즈를 조정

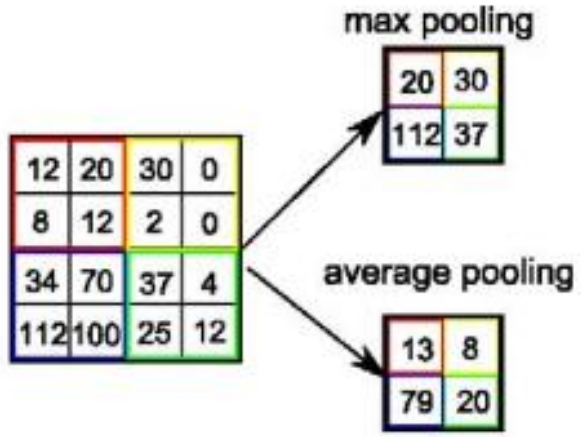


Pooling 연산 : Max Pooling



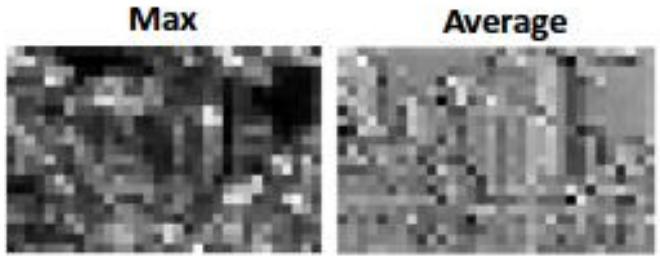
- Pooling 연산: Filter를 이용하여 결과들을 합침
  - Conv layer의 출력에서 정보를 단순화
  - 차원은 줄임
  - 공간적인 의미도는 유지

Pooling 연산 : Max Pooling, Average Pooling



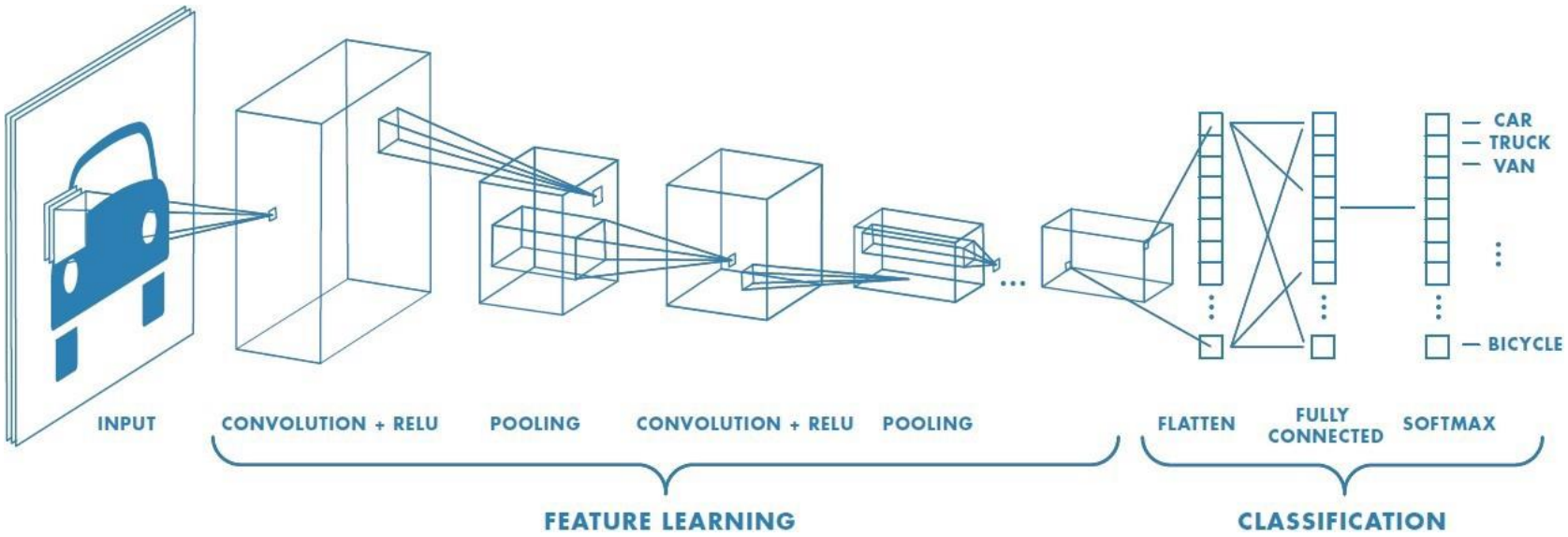
Produces a volume of size $W_2 \times H_2 \times D_2$ where:

- $W_2 = (W_1 - F)/S + 1$
- $H_2 = (H_1 - F)/S + 1$
- $D_2 = D_1$

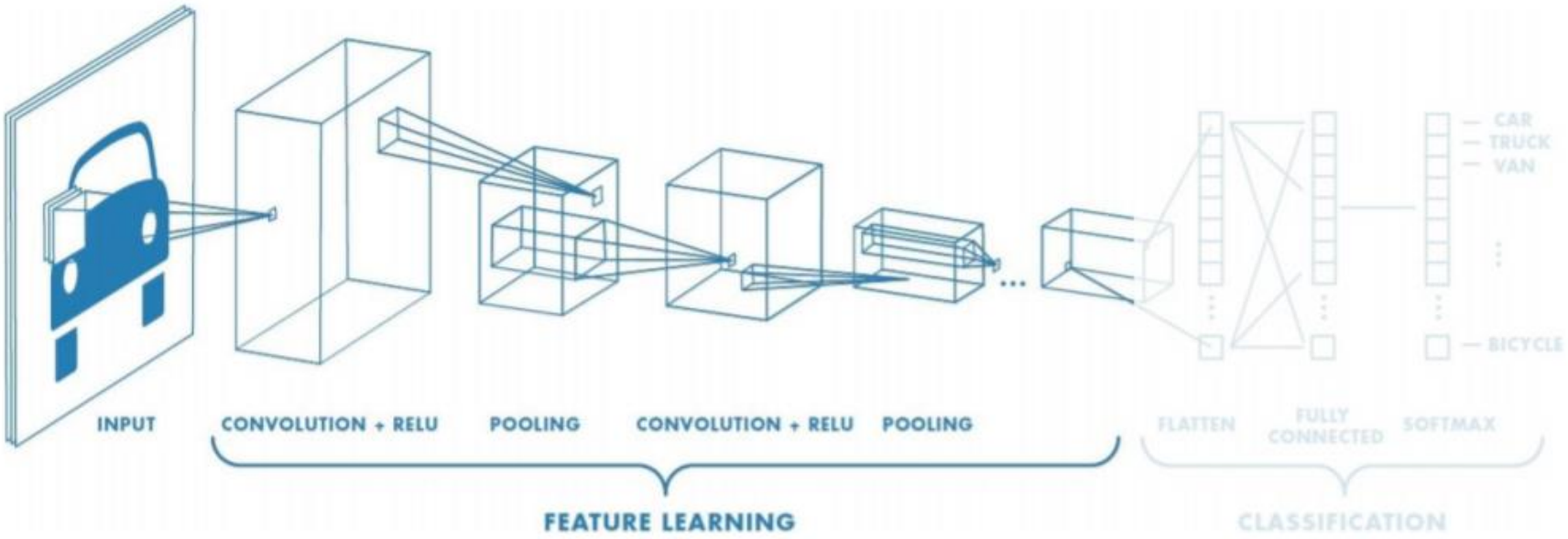


- Max pooling 연산은 edge와 같은 가장 중요한 feature 들을 추출하는데 반해,
- Average pooling 연산은 비교적 smooth한 feature를 추출한다.

CNN for Classification

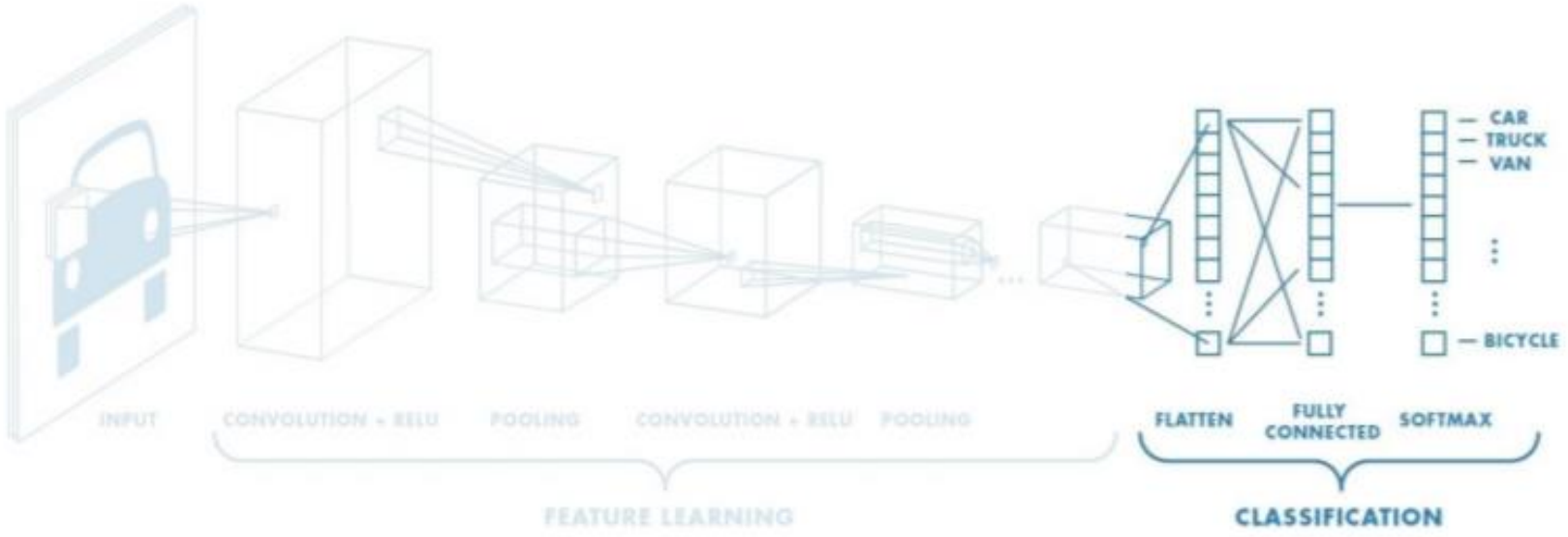


Feature Learning



1. 이미지로부터 convolution 연산을 통해 특징을 학습
2. Activation function을 거쳐 non-linearity 적용
3. Pooling으로 공간정보를 유지하면서도 차원을 축소

CNN for Classification



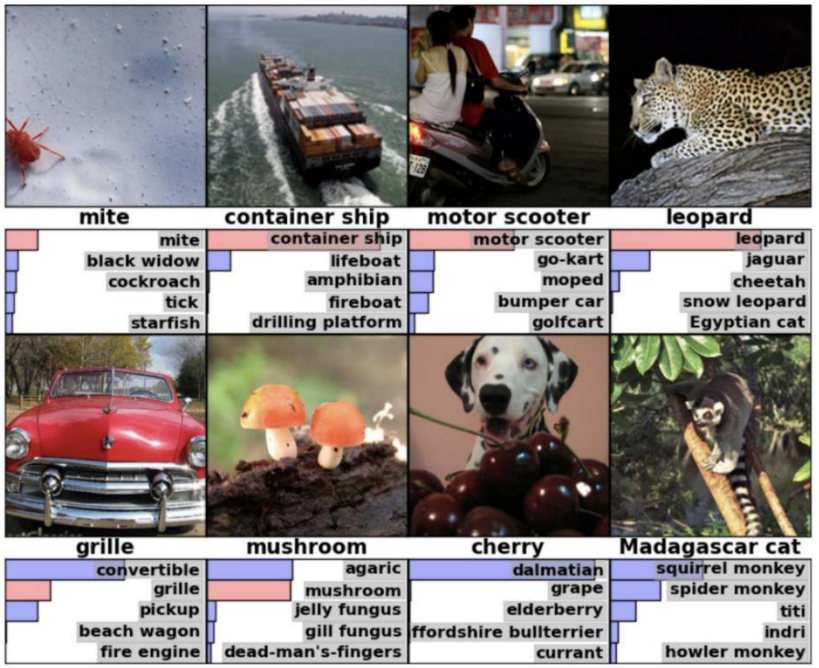
- Convolution과 Pooling으로 도출된 고차원의 feature를
- 분류를 하기 위해 납작하게 1차원으로 펼쳐서
- Class 수만큼 분류되도록 fully-connected로 연결!
- Class 수만큼의 probability로 표현

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

ImageNet Challenge : 글로벌 이미지 분듀 대회

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

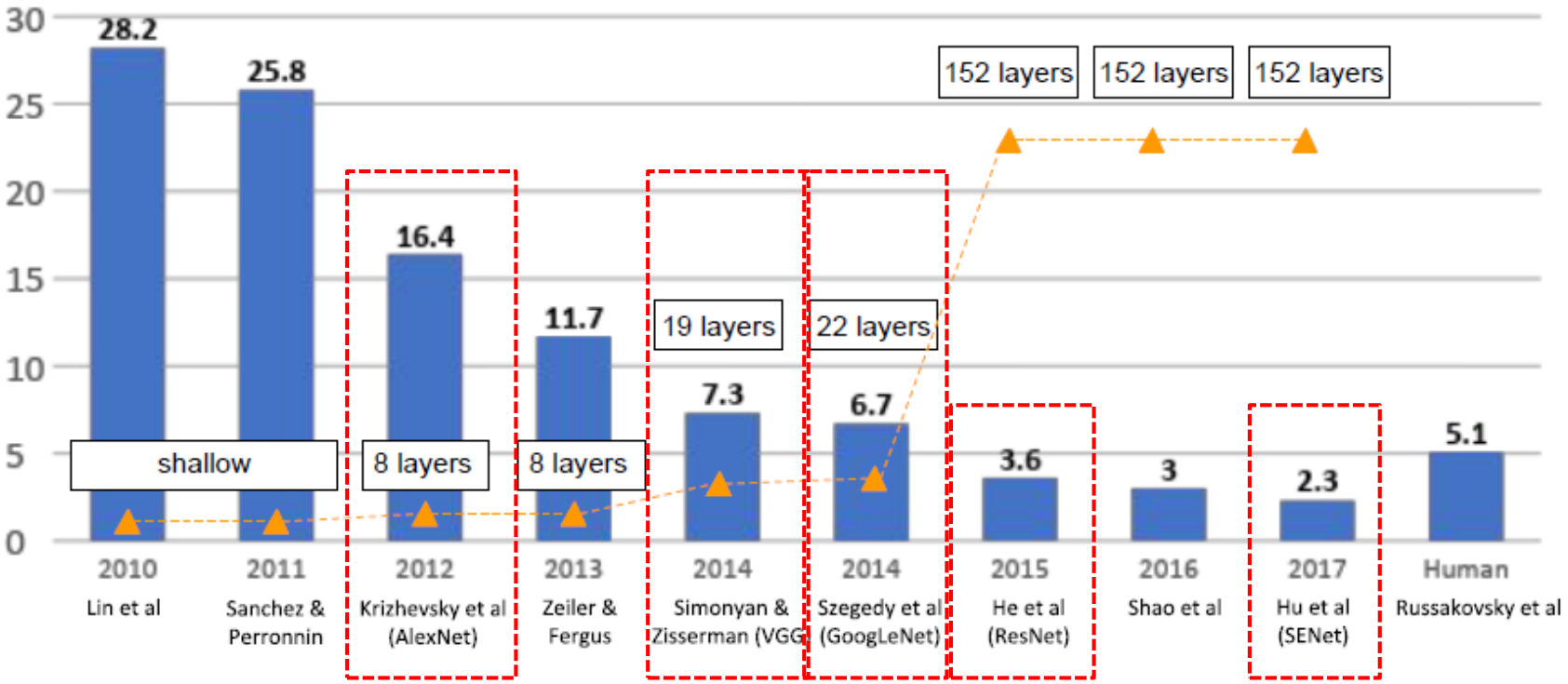


Classification task: produce a list of object categories present in image. 1000 categories.
"Top 5 error": rate at which the model does not output correct label in top 5 predictions

Other tasks include:
single-object localization, object detection from video/image, scene classification, scene parsing

ImageNet 챌린지에서 우승한 주요 CNN 아키텍처

AlexNet | VGGNet | GoogLeNet | ResNet | DenseNet



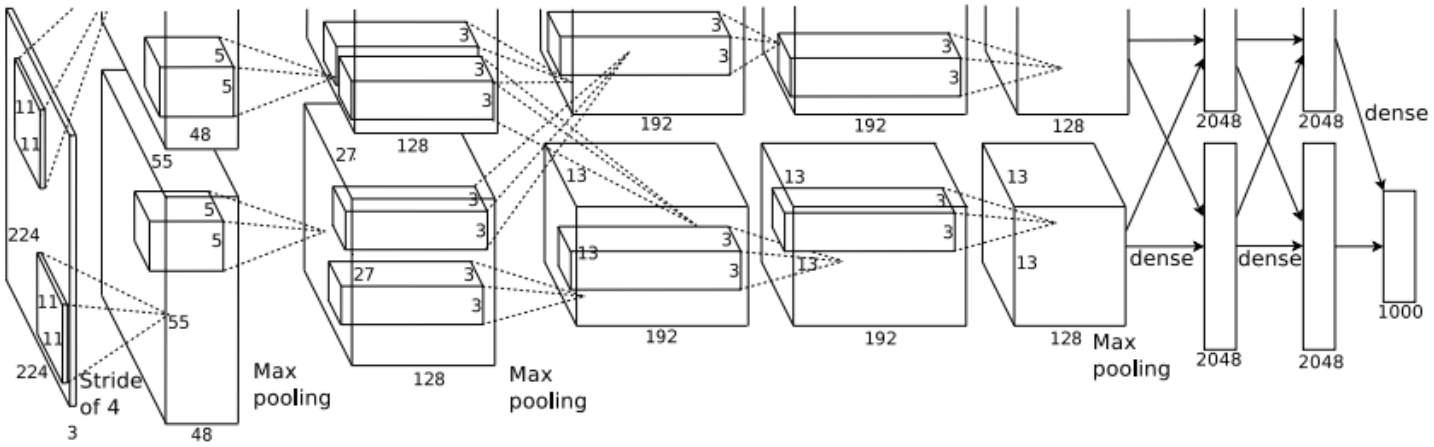
AlexNet : ImageNet 대회 첫 딥러닝(CNN기반) 모델 적용, 오듀륻 크게 개선

2012 ILSVRC Top-5 accuracy : 84.7% (2nd place is with 73.8%)

특징

- 처음으도 ReLU 적용
- Dropout 적용
- 2개의 path도 나누어 처리, 중간과정에 cross 공유 (자원제약)

Architecture:
CONV1
MAX POOL1
NORM1
CONV2
MAX POOL2
NORM2
CONV3
CONV4
CONV5
Max POOL3
FC6
FC7
FC8

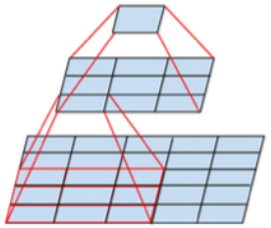


ImageNet classification with deep convolutional neural networks, by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton (2012).

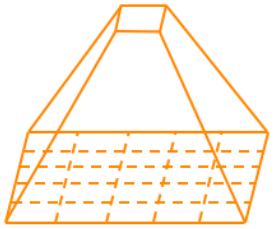
VGGNet : 더 깊은 신경망은 더 좋은 성능을 보인다! (Depth is matter!)

특징

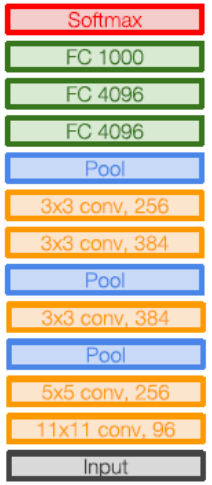
- 더 작은 필터 (3x3 conv)
- 더 깊은 네트워크 (11층, 13층, 16층, 19층)



two successive
3x3 convolutions



5x5 convolution



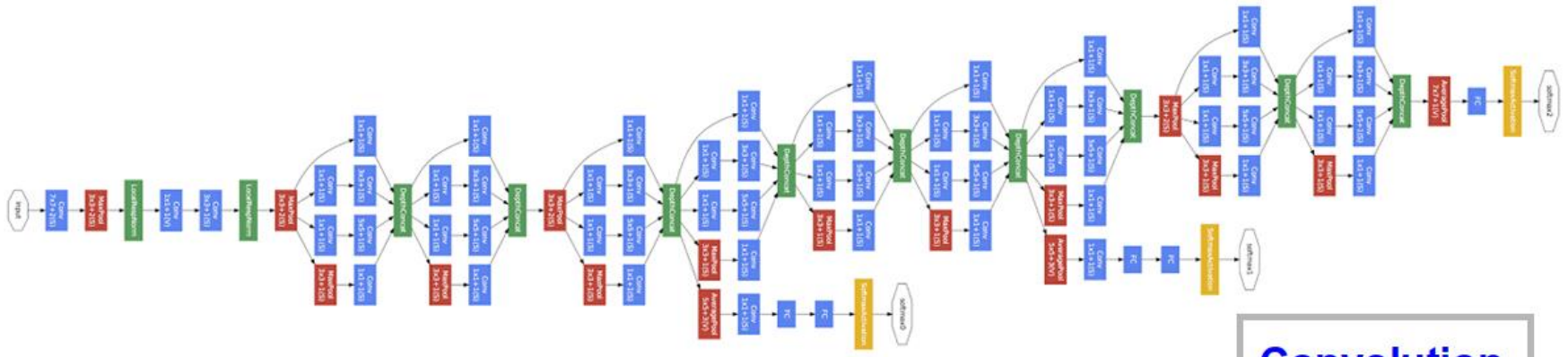
AlexNet

VGG16

VGG19

GoogleNet : 다양한 conv 연산의 활용

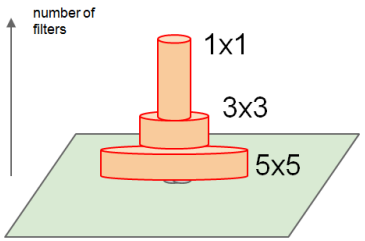
’14 ILSVRC 우승 Top-5 error : 6.7% (VGG: 7.3 %)

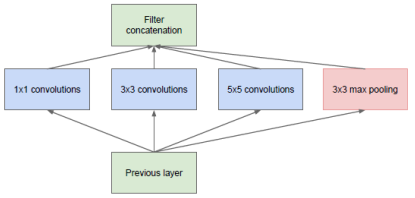


Convolution
Pooling
Softmax
Other

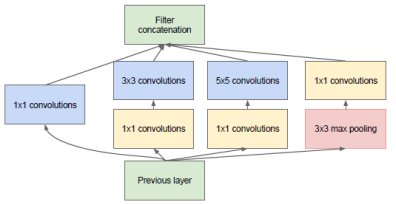
특징

- 더 깊고 더 복잡하다...! 22층!
- 다양한 사이즈의 conv연산을 한번에 적용한 Inception module
- 한번에 좁은 영역, 넓은 영역을 동시에 보는 효과
- 1x1 conv도 차원축소 효과
- Auxiliary classifier 적용으로 vanishing gradient 문제 해결





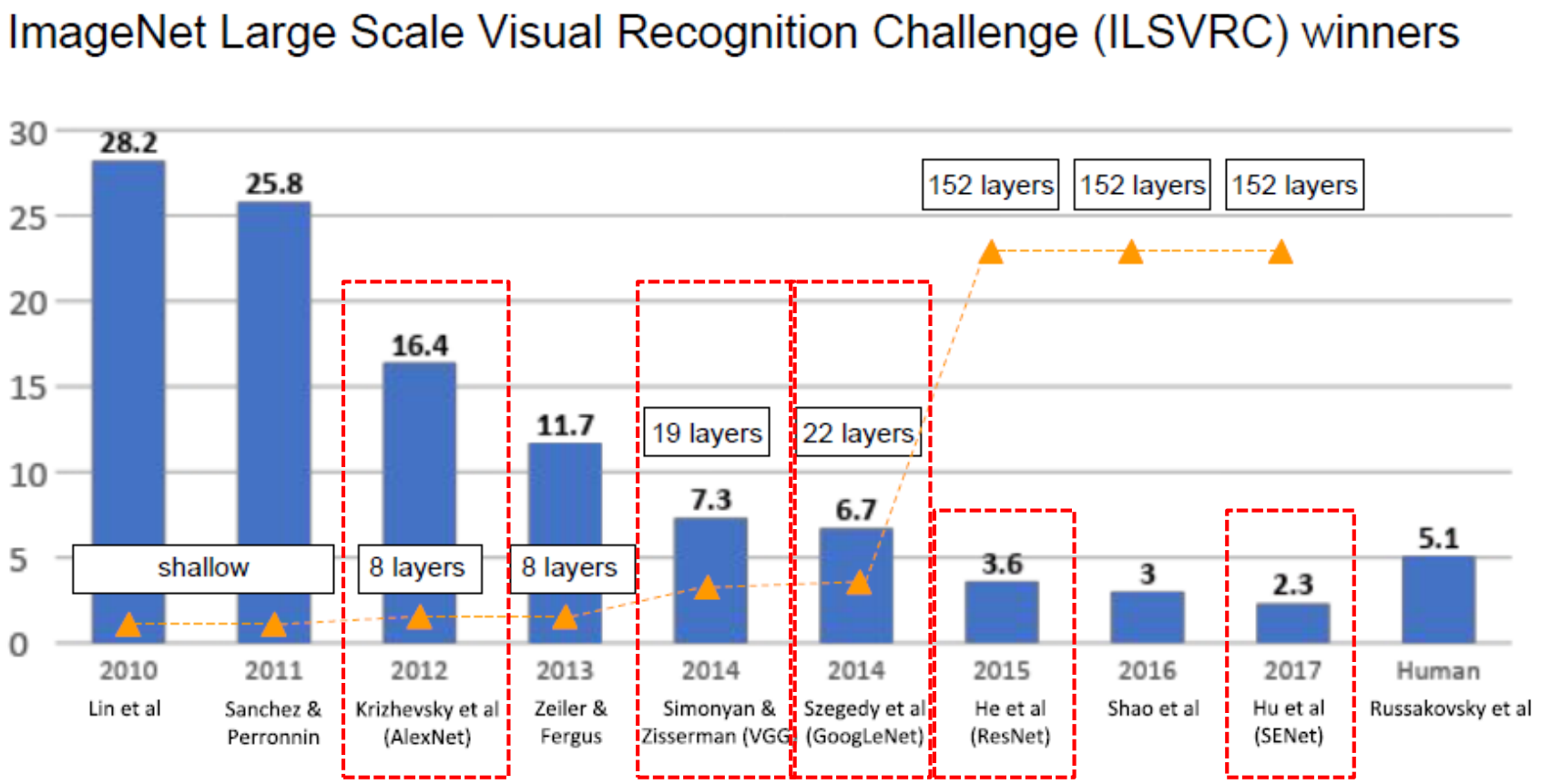
(a) Inception module, naïve version



(b) Inception module with dimensionality reduction

Going deeper with convolutions, by Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich (2014).

이제부턴 인공지능이 사람의 성능을 뛰어넘기 시작합니다...



ResNet : ???:"야 그렇게 몇층씩 늘려서 되겠냐? 팍팍 쌓자!"

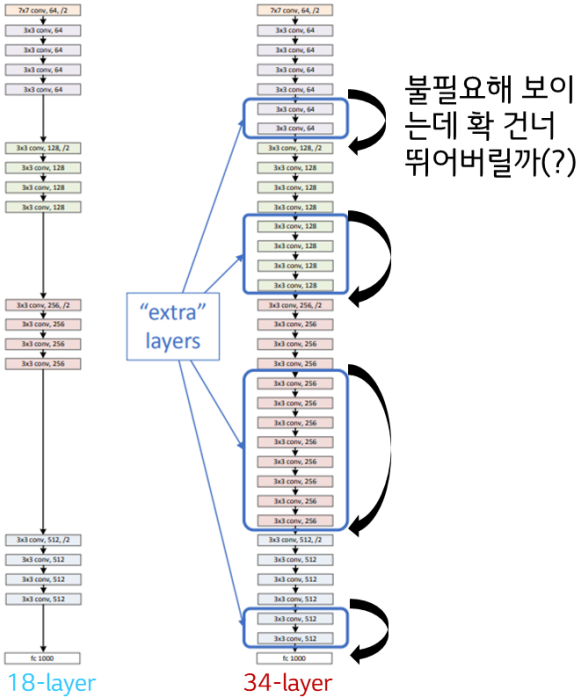
'15 ILSVRC Top-5 accuracy : 3.57%

특징

- 최초로 사람의 성능을 뛰어넘은 인공지능 모델
- 무려 152층!!
- 무작정 깊다고 좋은것도 아니었다
  → Residual connection 적용, 일종의 지름길 효과

MSRA @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: *"Ultra-deep"* (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

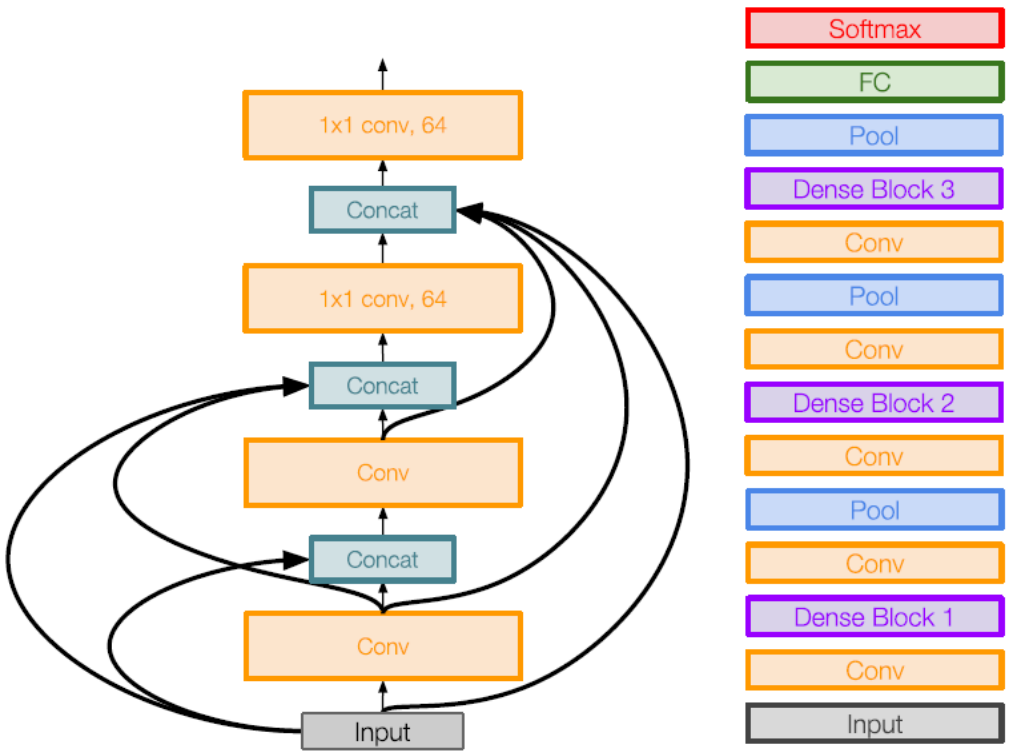


Deep Residual Learning for Image Recognition, by Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2015).

DenseNet : 지름길이 좋다고?? 그럼 모든 부분에 넣자!

특징

- 가능한 모든 부분에 shortcut 적용
- 입력 데이터의 흐름 및 gradient의 흐름을 원활하게 하는 효과



Dense Block