• Neural Network : fully connected layer,
1D vector ← weighted sum, W

{ + convolutional layer : 2D input $\xrightarrow{\text{kernel}}$ 2D output
{ + max pooling layer

$\boxed{\text{o kernel, o padding, o stride}}$

In-Class 20: Convolutional Neural Network

[SCS4049] Machine Learning and Data Science

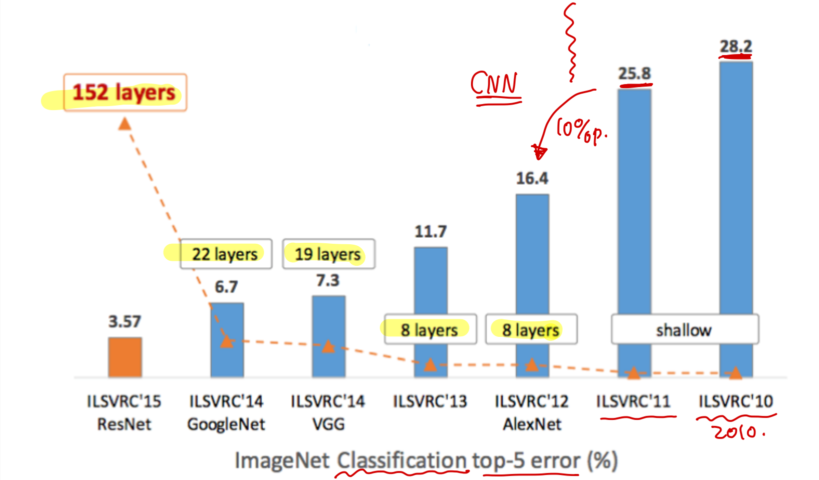
Seongsik Park (s.park@dgu.edu)

AI Department, Dongguk University

size kernel.
↓
input → conv. layer → output
size   o padding   size.
         o stride

ImageNet challenge

Image Classification



The architecture of the visual cortex

Convolution : 주어진 image에, 주어진 kernel같은 모양(?)이 얼마나 들어있는가?
heat map 변환

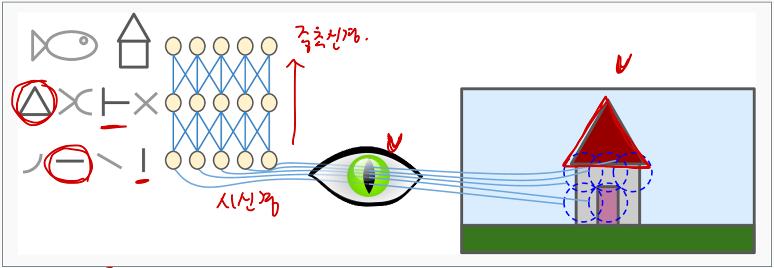
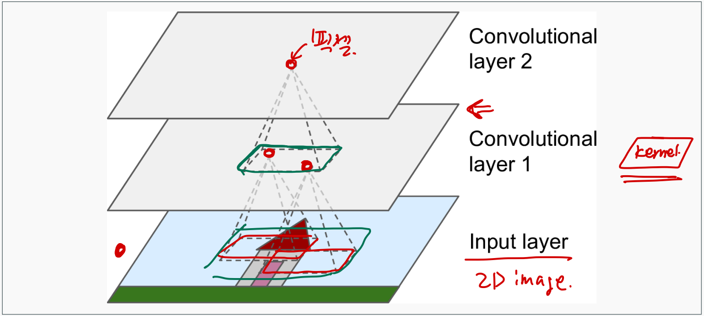


Figure 14-1. Local receptive fields in the visual cortex

"착시"

In particular, they showed that many neurons in the visual cortex have a small **local receptive field**, meaning they react only to visual stimuli located in a limited region of the visual field (see Figure 14-1, in which the local receptive fields of five neurons are represented by dashed circles). The receptive fields of different neurons may overlap, and together they tile the whole visual field.

Convolutional layer



feature 특징,특성.

Figure 14-2. CNN layers with rectangular local receptive fields

The most important building block of a CNN is the **convolutional layer**:6 neurons in the first convolutional layer are not connected to every single pixel in the input image (like they were in previous chapters), but only to pixels in their receptive fields (see Figure 14-2). In turn, each neuron in the second convolutional layer is connected only to neurons located within a small rectangle in the first layer. This architecture allows the network to concentrate on small low-level features in the first hidden layer, then assemble them into larger higher-level features in the next hidden layer, and so on.

Convolutional neural network

- 컨볼루션 신경망(CNN)은 다층 신경망(multi-layer neural networks)의 특별한 형태
- CNN은 시각 피질의 국소 수용역(Local Receptive Field, LRF)과 시신경 세포의 방향 선택적 성격의 발견에 의해서 신경 생물학적인 영감으로 만들어짐

이미지→특징,특성

- 내재적이면서 자동적으로 유의미한 특징(feature)들을 추출하는 신경망 구조
- CNN 은 이미지로부터 위상학적 성질들을 추출할 수 있는 피드포워드(feed-forward) 신경망
- 다른 모든 신경망들처럼 CNN 은 변형된 역전파 알고리즘을 사용하여 학습
- CNN 은 최초한의 전처리 만으로도 픽셀 이미지들로부터 시각적 패턴을 직접 인식할 수 있도록 설계

⇒ 극도로 변화가 많은 패턴들도 인식 가능함 (예: 손글씨)

이미지 인식,분류.

Convolution

Given a filter kernel $\mathcal{H}$, the convolution of the kernel with image $\mathcal{F}$ is an image $\mathcal{R}$. The (i, j)-th component of $\mathcal{R}$ is given by

$$R_{ij} = \sum_{u,v} H_{i-u,j-v} F_{uv}. \tag{1}$$

- **kernel** of the filter: the pattern of weights used for a linear filter
- **convolution**: the process of applying the filter

This operation is called **convolution**

$$R(f) = (h * f) \tag{2}$$

- *commutative*: $(g * h)(x) = (h * g)(x)$
- *associative*: $(f * (g * h)) = ((f * g) * h)$
- *distributive*: $f * (g + h) = f * g + f * h$

Padding (border effects)

- **zero**: set all pixels outside the source image to 0
- **constant**: set all pixels outside the source image to a specified border value   따로 명시한 특정값
- **clamp**: repeat edge pixels indefinitely
- **wrap**: loop “around” the image in a “toroidal” configuration
- **mirror**: reflect pixels across the image edge
- **extend**: extend the signal by subtracting the mirrored version of the signal from the edge pixel value

1,1 1,2

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}$$

2,1 2,2

3x3

2x2
kernel

2x2. 같게
output

$$\begin{bmatrix} 1 \\ 4 \\ 7 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} * \begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

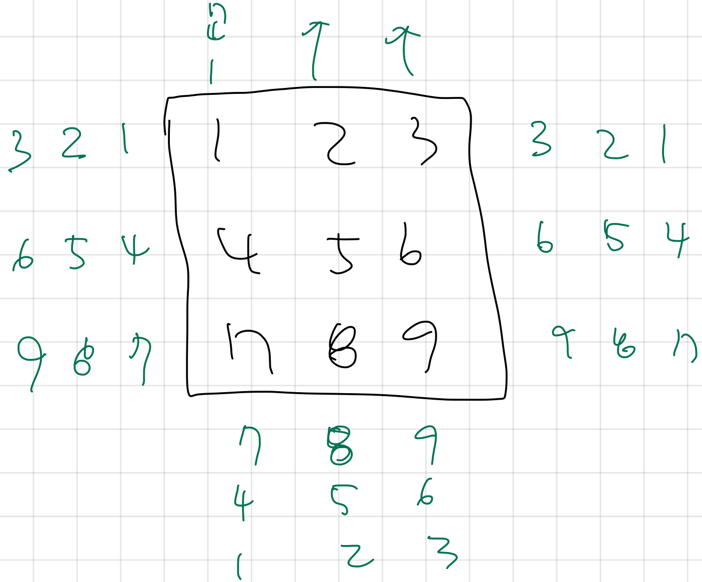
$\boxed{?}$

$\boxed{7 \mid 8 \mid 9}$

3x3
4x4.

2x2
kernel

3x3
output.



Zero padding

Constant

clamp

mirror

⋮

Padding (border effects)

바둑판식.
동일한이미지반복.

대칭한이미지반복.

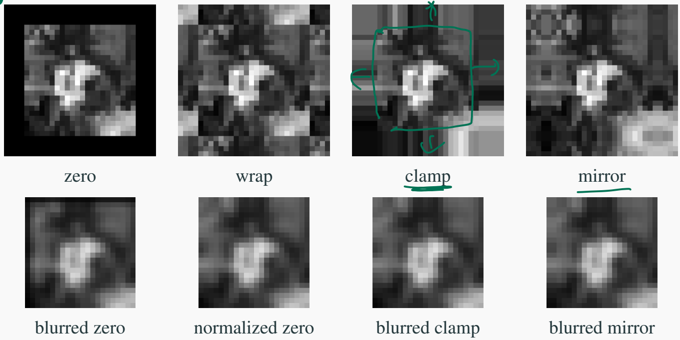


Figure 3.13 *Border padding (top row) and the results of blurring the padded image (bottom row). The normalized zero image is the result of dividing (normalizing) the blurred zero-padded RGBA image by its corresponding soft alpha value.*

Convolution: connections and padding

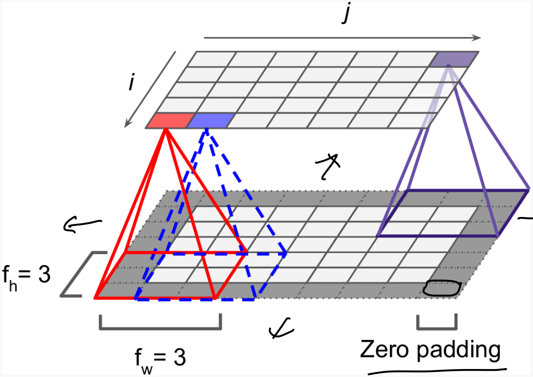


Figure 14-3. Connections between layers and zero padding

5x7 output.

3x3 kernel.

5x7 input

Convolution: stride and dimensional reduction

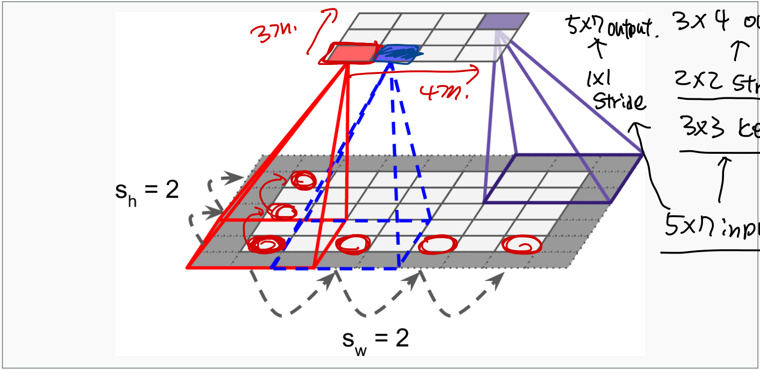


Figure 14-4. Reducing dimensionality using a stride of 2

Convolution → ~~한~~ 픽셀씩 움직이면서 계산.

Stride

$$\underbrace{\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix}}_{4\times 3} * \underbrace{\begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix}}_{2\times 2} =$$

padding X
Stride 1x1
3x2

padding 0
Stride 1x1
4x3

padding 0
Stride 2x1
2x3

padding 0
Stride 3x2
2x2

$$\begin{array}{cc} 1,1 & 1,2 \\ & \end{array}$$

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 4 & 5 & 6 \\ 0 & 7 & 8 & 9 \\ 0 & 10 & 11 & 12 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & -1 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} 6 & 9 \\ 20 & 46 \end{bmatrix}$$

$$\begin{array}{cc} 2,1 & 2,2 \end{array}$$

$$\begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}$$

MNIST

input

28x28

→ Conv. layer →

kernel=10

size=3,3

padding=0, zero.

stride=2,2

output.

14x14 x10

conv. 결과 이미지 size

kernel의 개수.

→ 1D. vector

1960x1

→ NN

3x3

10개.

Convolution: example

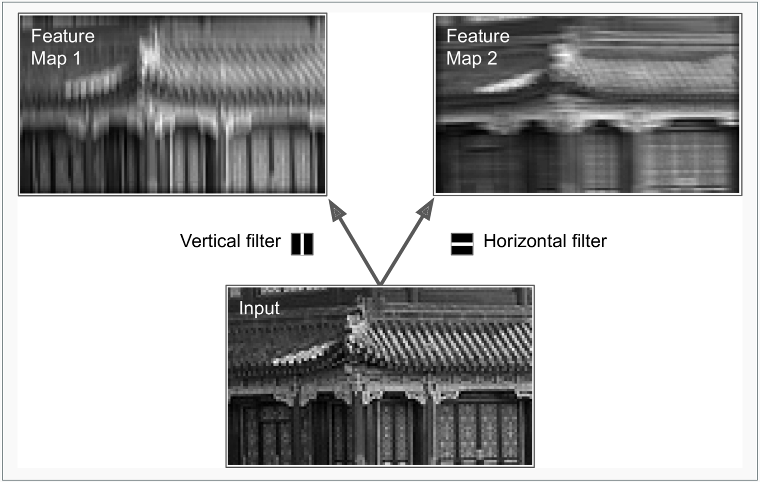
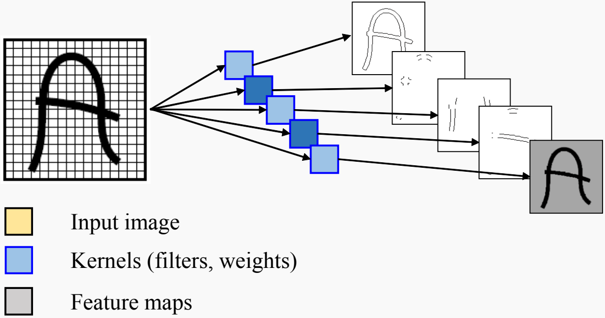


Figure 14-5. Applying two different filters to get two feature maps

Convolution: filter effect

Feature extraction by different kernels



- 컨볼루션은 입력 이미지의 다른 부분들에 존재하는 동일한 특징들을 찾아냄
- Feature map에 있는 뉴런은 template or kernel과 일치되는 경우에 활성화

CNN: convolution layers with multiple feature maps

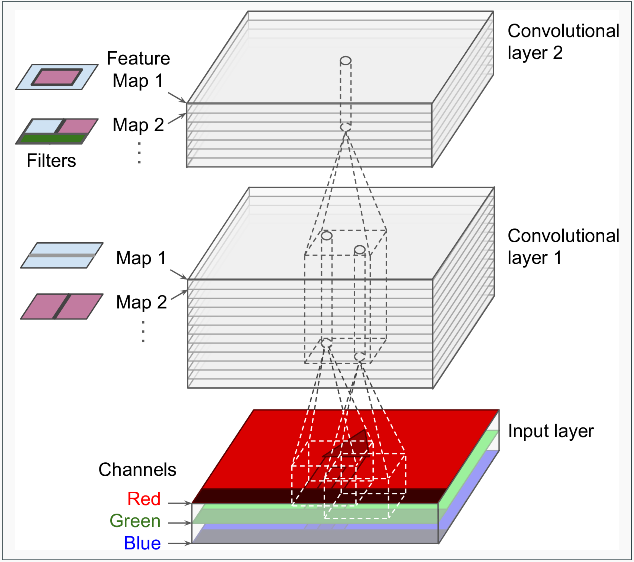
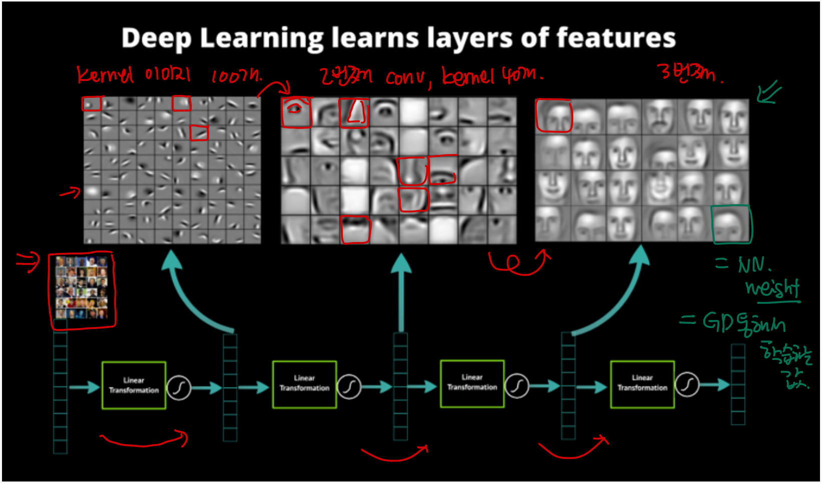


Figure 14-6. Convolution layers with multiple feature maps, and images with three color channels

CNN: hierarchical learning of features

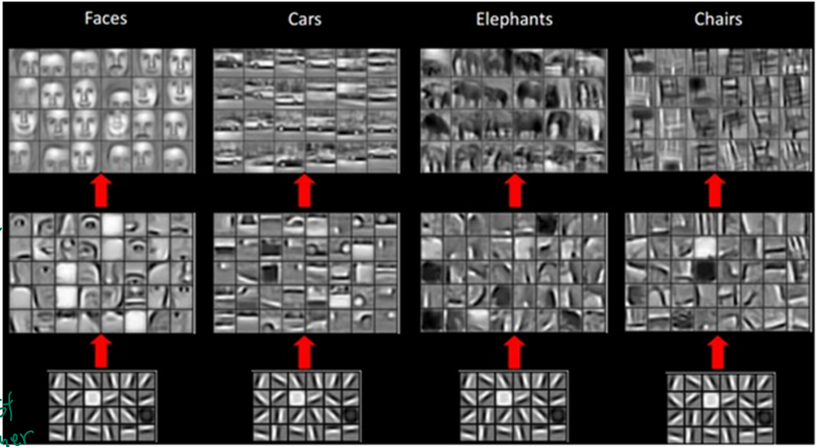


CNN: hierarchical learning of features

3rd

2nd

1st layer

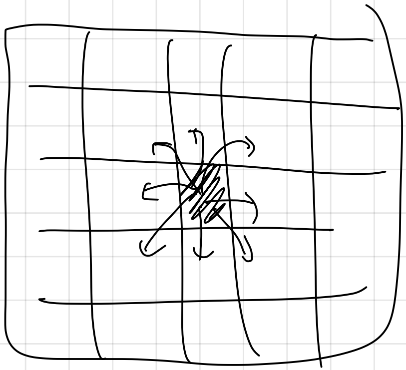


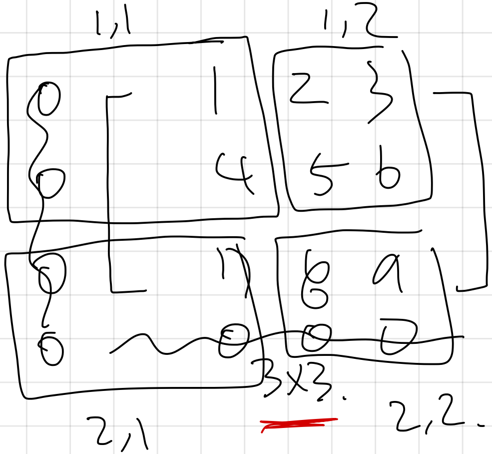
Flatten

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \quad \begin{bmatrix} 9 & 10 \\ 11 & 12 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{bmatrix}$$

$2 \times 2 \times 3$

2D 이미지.





$\rightarrow$ max pooling $\rightarrow$ $\begin{bmatrix} 4 & 6 \\ 7 & 9 \end{bmatrix}$

2x2

size = 2x2

stride = 2x2 (default = size)

padding = 0.

CNN: dimensional reduction by pooling layer

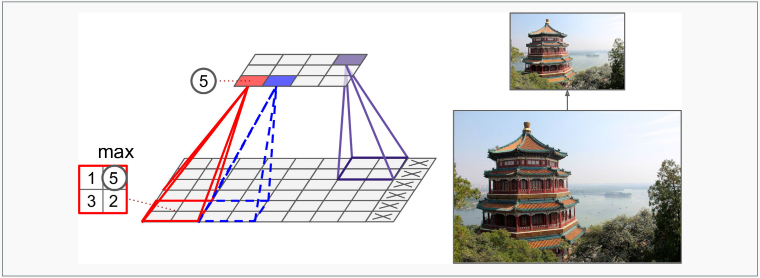


Figure 14-8. Max pooling layer (2 × 2 pooling kernel, stride 2, no padding)

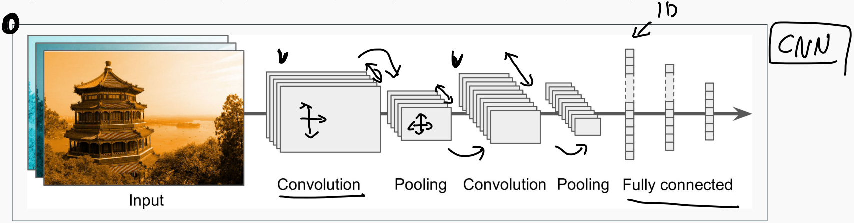


Figure 14-11. Typical CNN architecture

Appendix

Reference and further reading

- “Chap 14 | Deep Computer Vision Using Convolutional Neural Network” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow
- “Lecture 10 | Convolutional Neural Networks” of Kwang Il Kim, Machine Learning (2019)