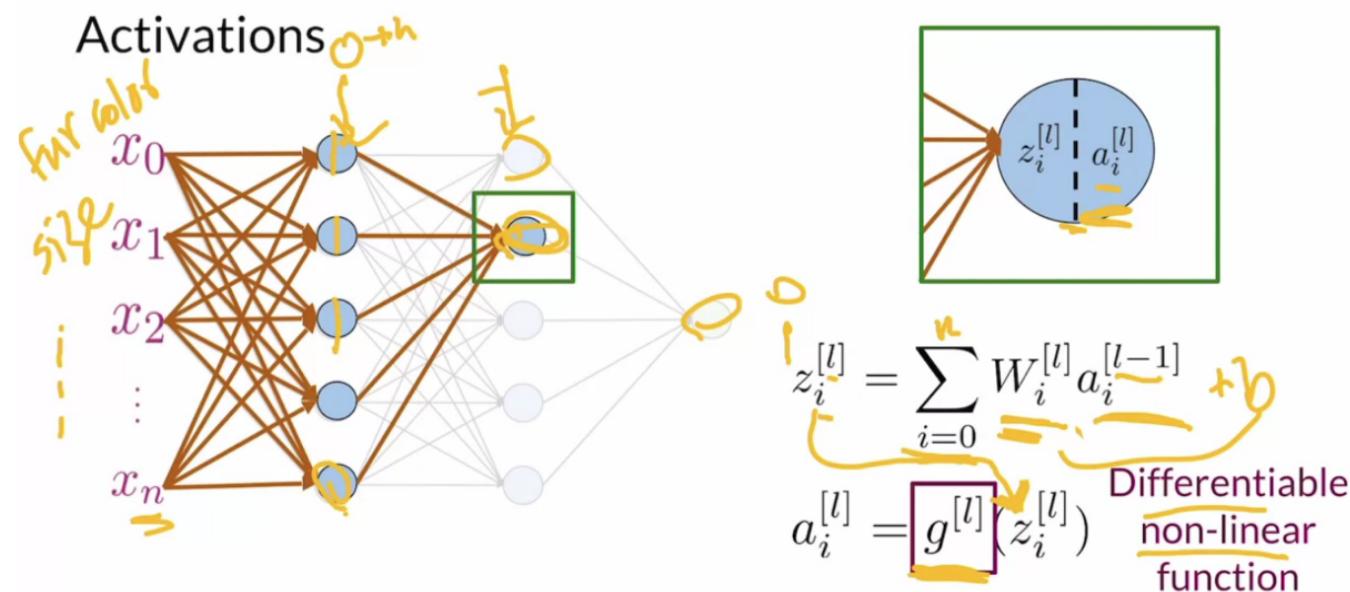
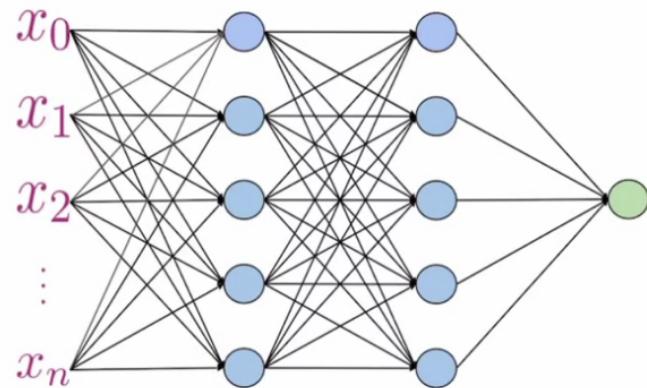


Deep Convolutional GANs

Activations (Basic Properties)

- 활성화(activation)란?
- 비선형(non-linear) 미분 활성화에 대한 추론

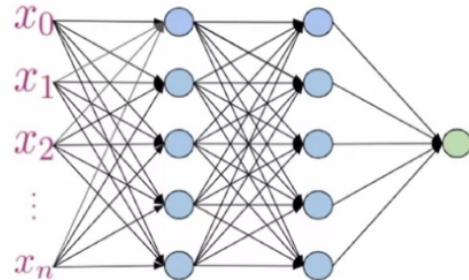


$$a_i^{[l]} = g^{[l]}(z_i^{[l]})$$

Differentiable
non-linear
function

1. Differentiable for backpropagation

2. Non-linear to compute complex features, **if not:**



$$WX + b$$

Linear
regression

요약

- 활성화 함수는 비선형(non-linear)이며 미분가능(differentiable)해야함
- 미분가능(differentiable)은 역전파(backpropagation)을 위함이다.
- 비선형(non-linear)은 복잡한 함수를 근사하기 위함이다.

Common Activation Functions

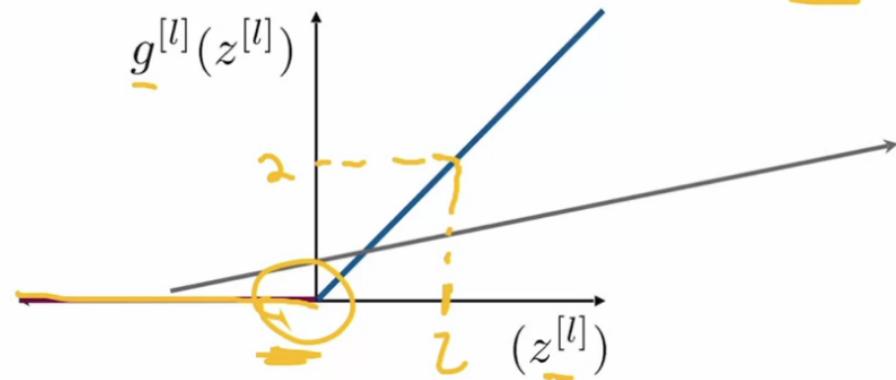
- ReLU
- Leaky ReLU
- Sigmoid
- Tanh

Activations: ReLU

ReLU = Rectified Linear Unit

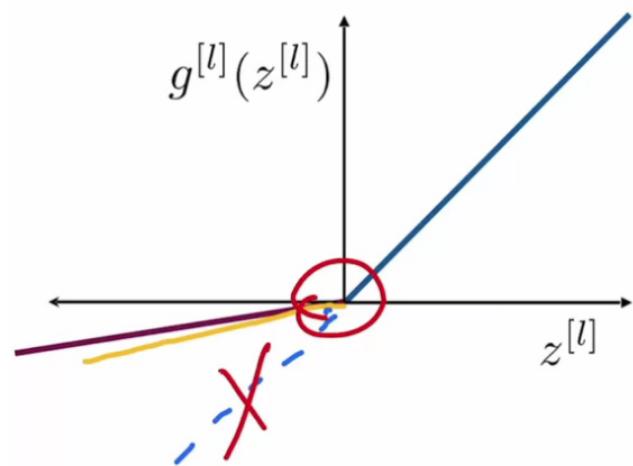
↓ *ReLU*

$$g^{[l]}(z^{[l]}) = \max(0, z^{[l]})$$



Dying ReLU
problem

Activations: Leaky ReLU



Activations: Sigmoid

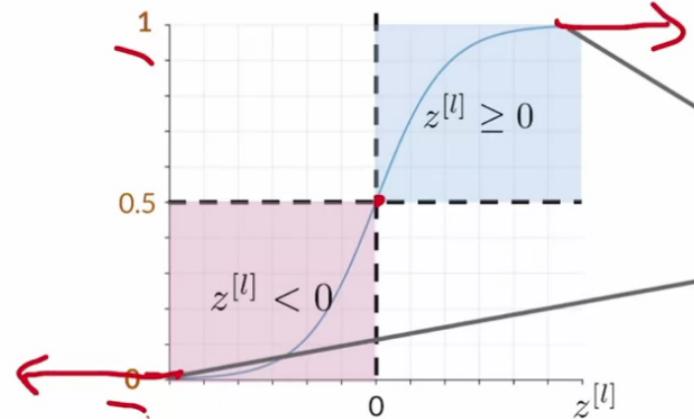
$$g^{[l]}(z^{[l]}) = \max(\underline{az}^{[l]}, \underline{z}^{[l]})$$

0.1

↓

Solves the dying
ReLU problem

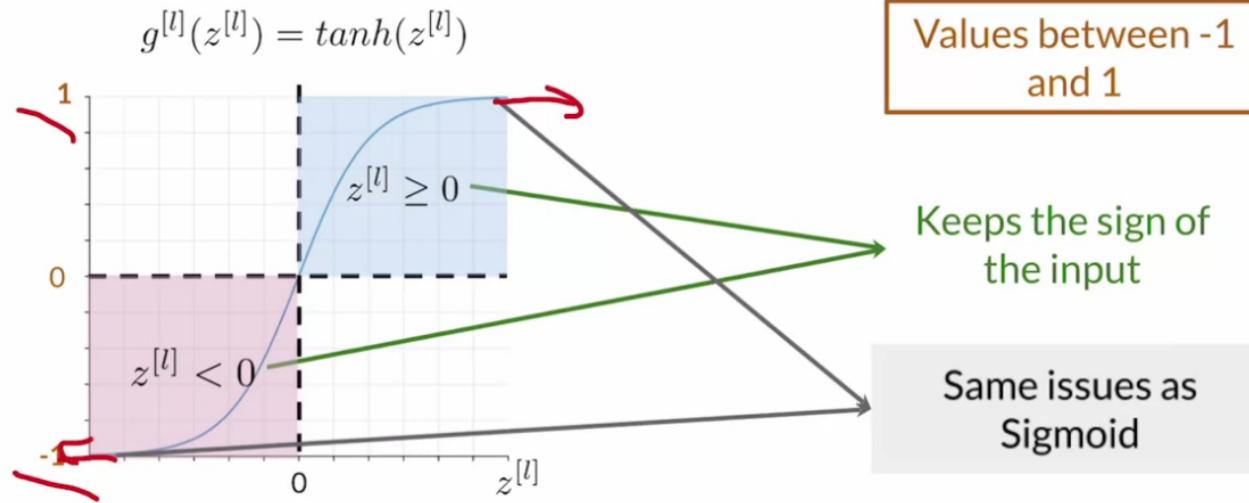
$$g^{[l]}(z^{[l]}) = \frac{1}{1 + e^{-z^{[l]}}}$$



Values between 0
and 1

Vanishing gradient
and saturation
problems

Activations: Tanh



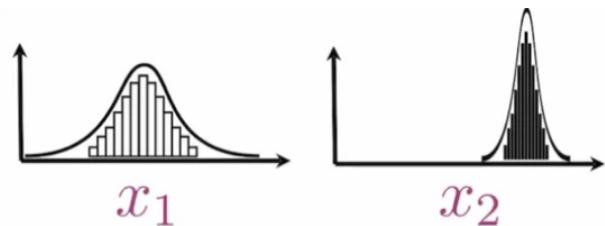
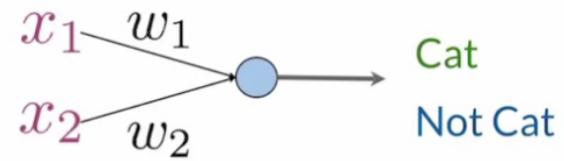
요약

- ReLU 활성화는 죽어가는 ReLU의 영향을 받는다.
- Leaky ReLU는 죽어가는 ReLU 문제를 해결한다.
- Sigmoid 와 Tanh는 소멸하는 그라디언트와 saturation 영향을 받는다.

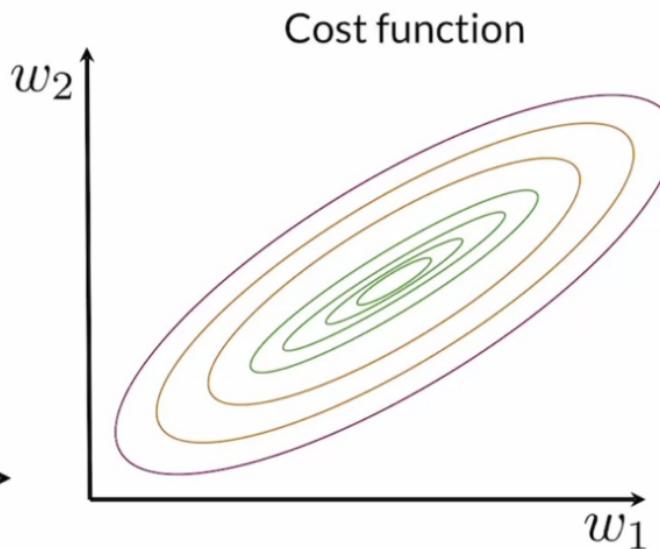
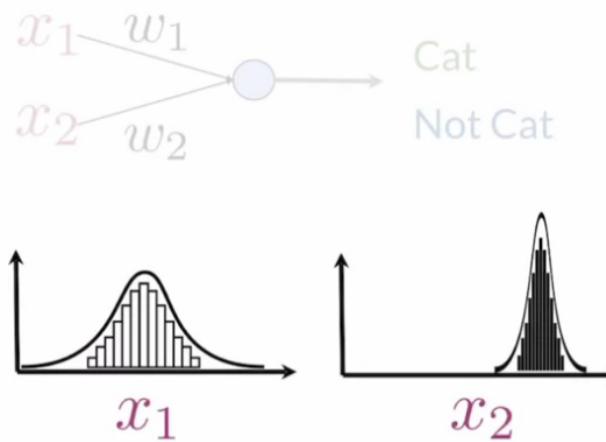
배치 정규화(Batch Normalization)

- 정규화는 어떻게 모델을 돋는가?
- 내부 공변량 이동(covariate shift)
- 배치 정규화(batch normalization)

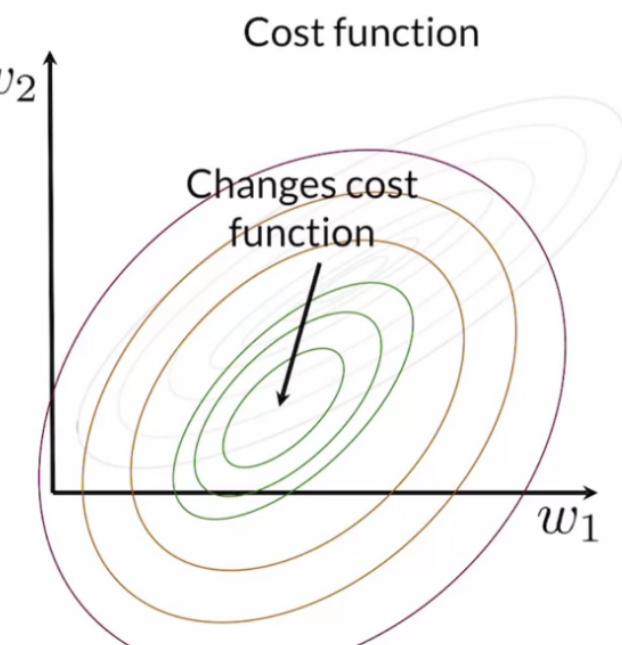
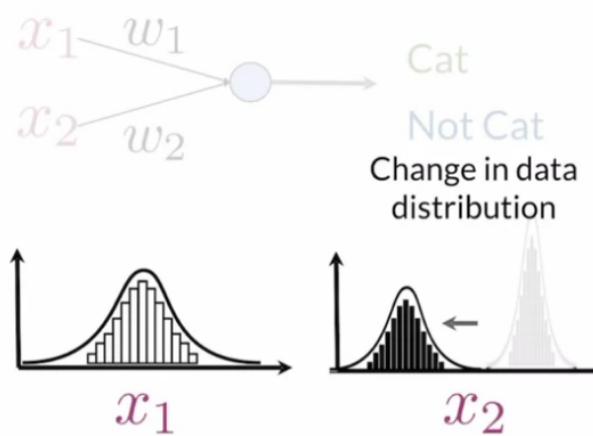
서로 다른 분포



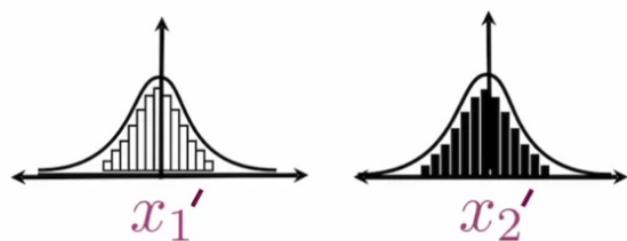
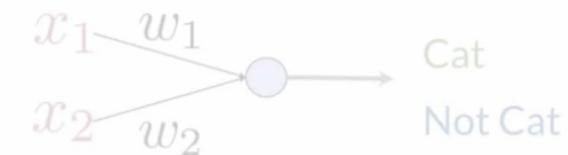
Covariate Shift



Covariate Shift



정규화와 그 효과

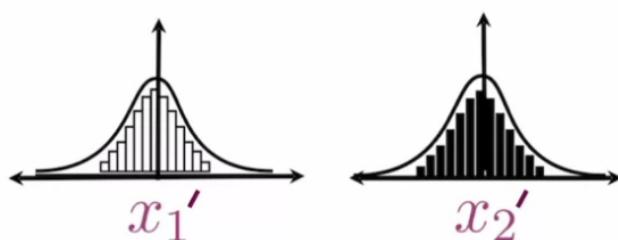
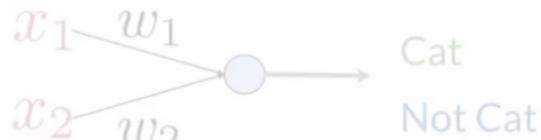


$$x_1' \quad x_2'$$

Around mean at 0
and std. at 1

Training data uses
batch stats

Test data uses
training stats



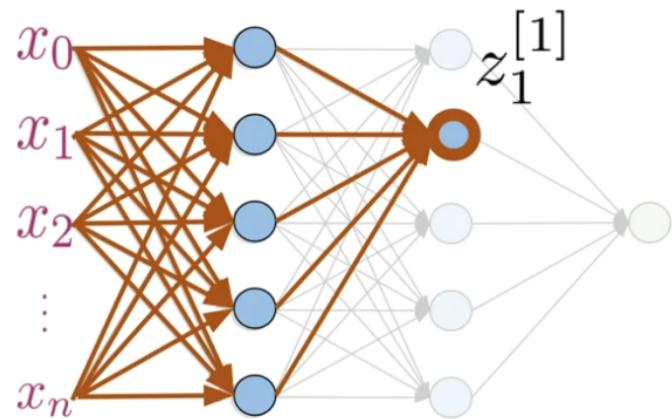
$$x_1' \quad x_2'$$

Around mean at 0
and std. at 1

Reduction of
covariate shift

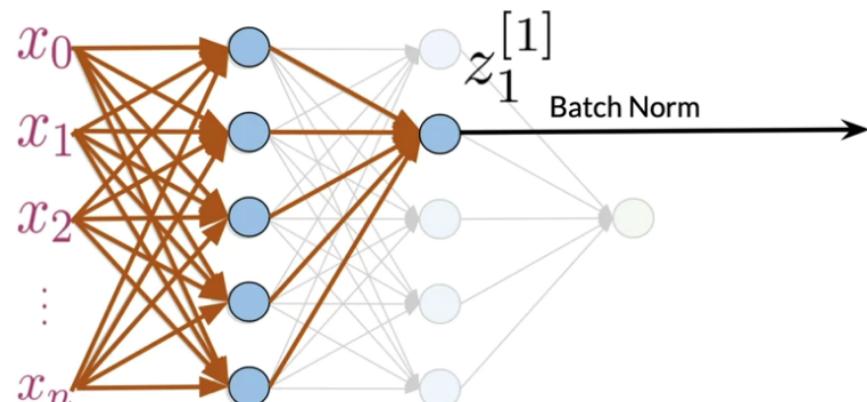
3.54

내부 공변량 이동

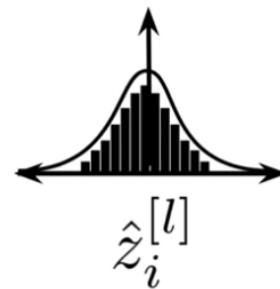


Changes in weights
 ↓
 Changes in activation distribution

배치 정규화



Normalizes the input for each neuron



요약

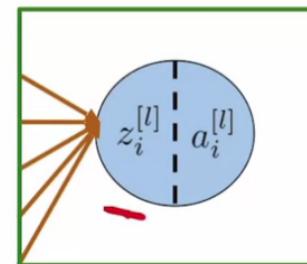
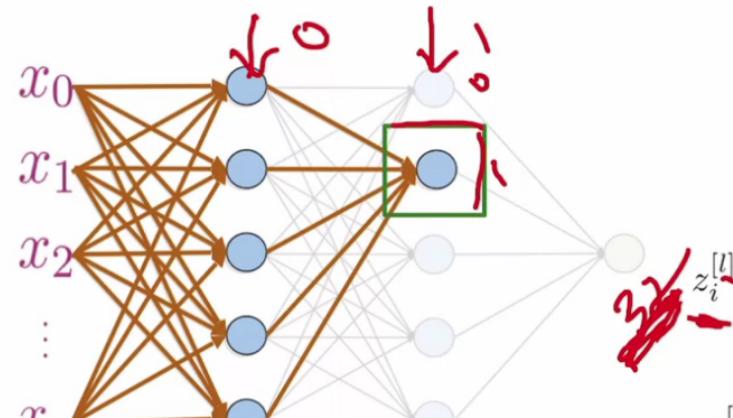
- 배치 정규화는 비용함수를 부드럽게 해준다.
- 배치 정규화는 내부 공변량 이동을 감소시킨다.
- 배치 정규화는 학습의 속도를 높여준다.

배치 정규화 (과정)

- 훈련을 위한 배치 정규화
- 시험을 위한 배치 정규화

배치 정규화: 훈련

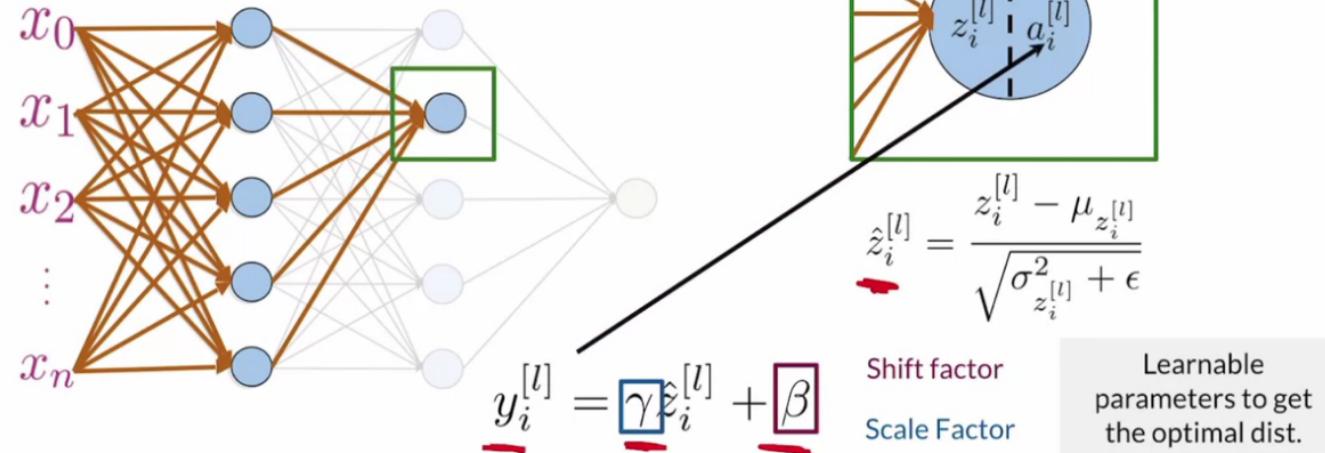
Batch Normalization: Training



$$z_i^{[l]} = \sum_{i=0} W_i^{[l]} a_i^{[l-1]} \rightarrow \text{For every example in the batch}$$
$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mu_{z_i^{[l]}}}{\sqrt{\sigma_{z_i^{[l]}}^2 + \epsilon}}$$

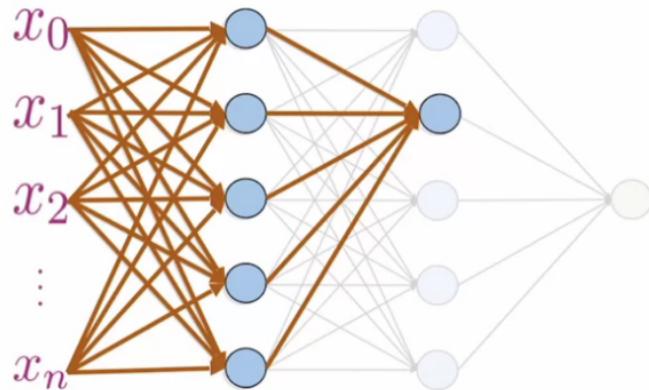
Batch mean of $z_i^{[l]}$
Batch std of $z_i^{[l]}$

Batch Normalization: Training



배치 정규화: 시험

Batch Normalization: Test



$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mathbb{E}(z_i^{[l]})}{\sqrt{\text{Var}(z_i^{[l]}) + \epsilon}}$$

Running mean and running std from training

Frameworks like Tensorflow and Pytorch keep track of them

요약

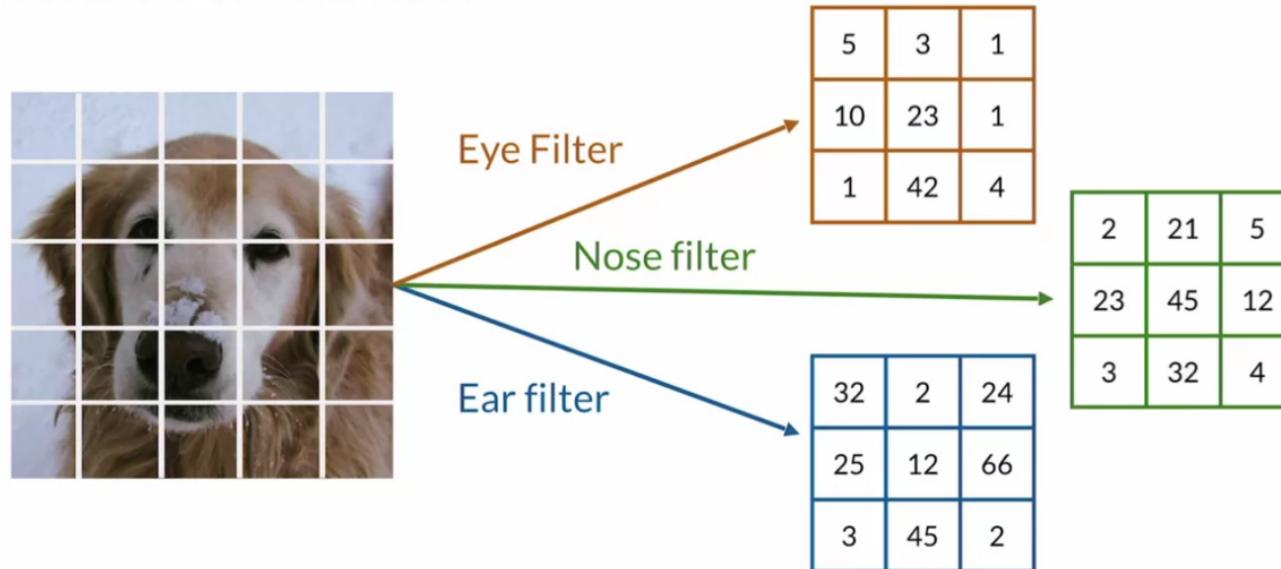
- 배치 정규화는 학습 가능한 이동 및 축척 요소를 도입합니다.
- 시험하는 동안, 훈련시 얻어진 통계치가 사용된다.
- 프레임워크가 이 모든 과정을 담당한다.

컨볼루션 복습

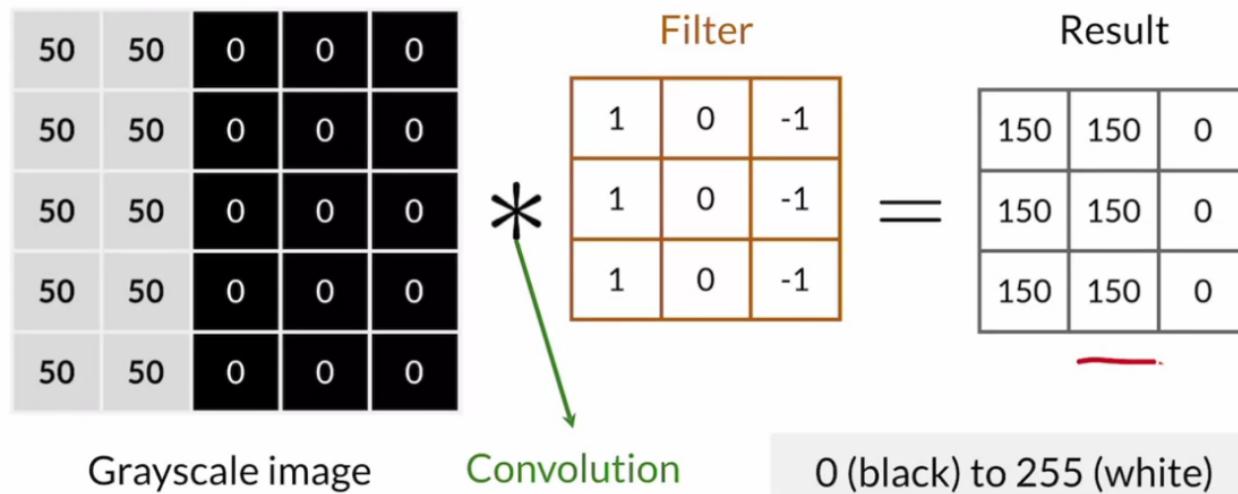
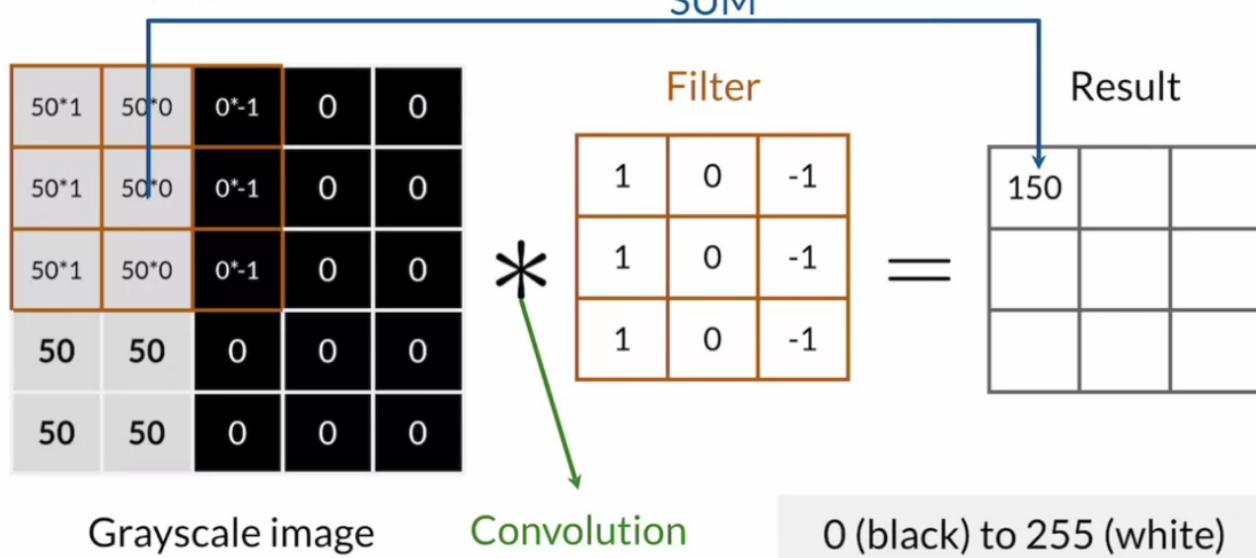
- 컨볼루션이란?
- 어떻게 동작하는가?

컨볼루션이란?

What is a convolution?



What is a convolution?



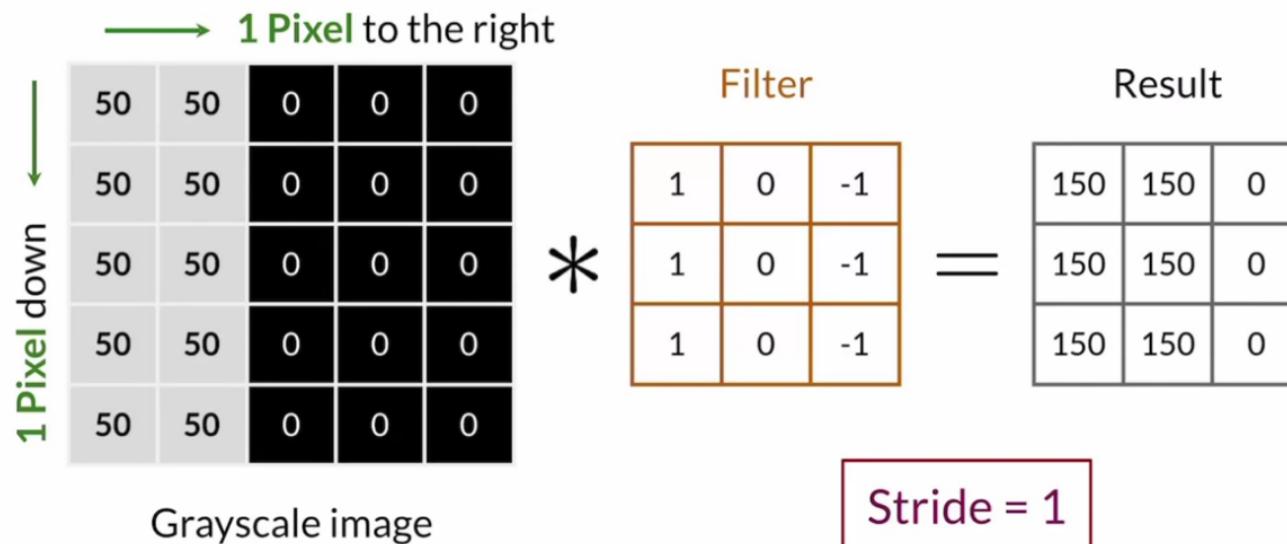
요약

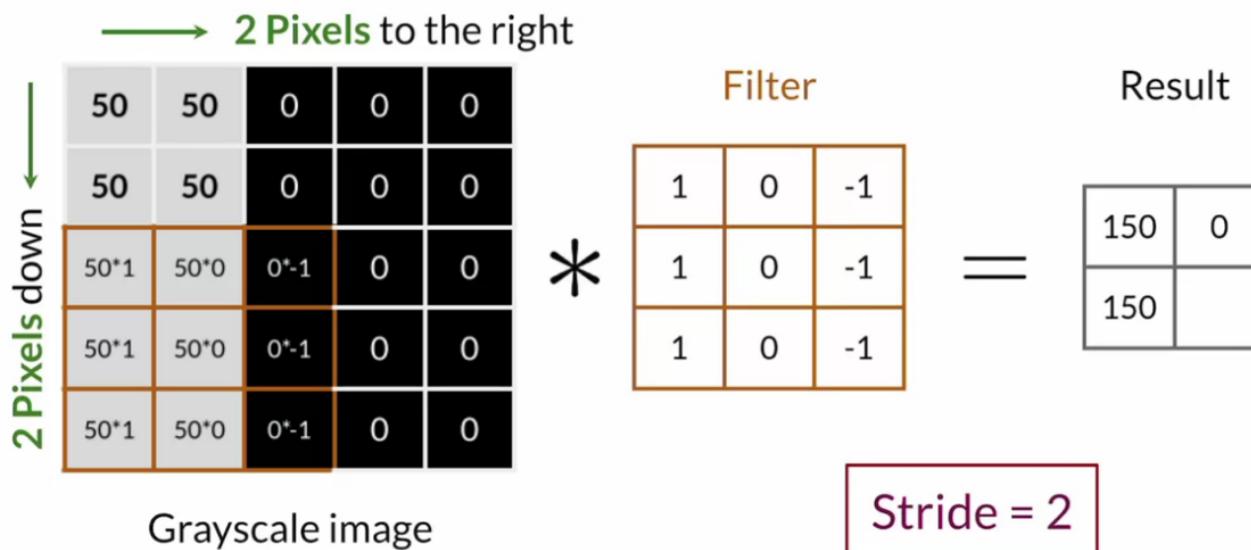
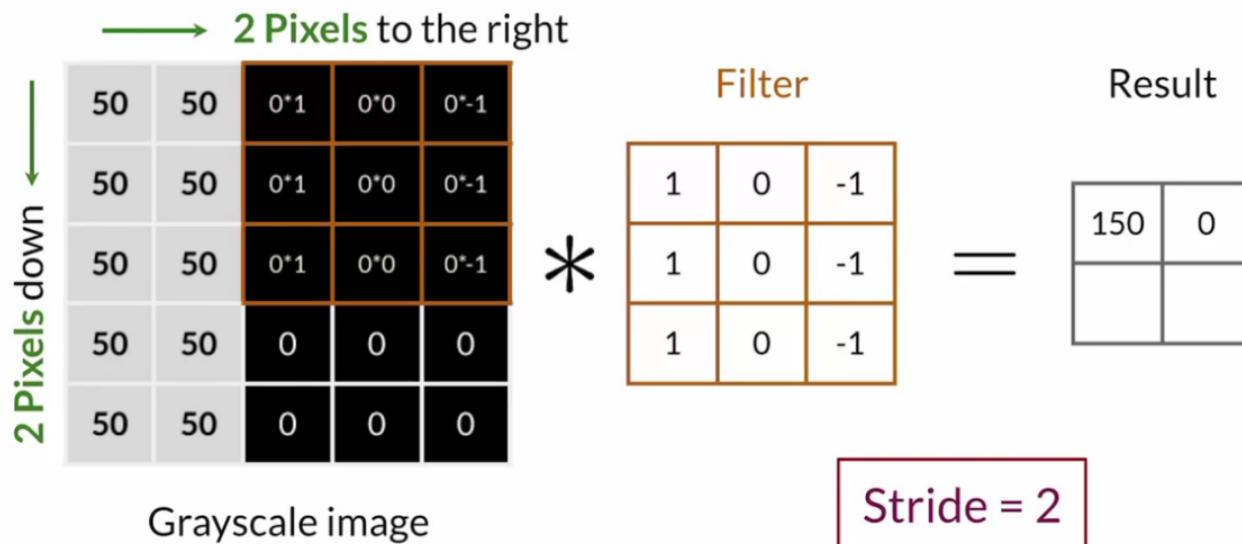
- 컨볼루션은 이미지를 처리할 때 유용한 층이다.
- 유용한 특징을 감지하기 위해 이미지를 스캔한다.
- 단지 요소별(element-wise) 곱셈과 덧셈으로 수행한다.

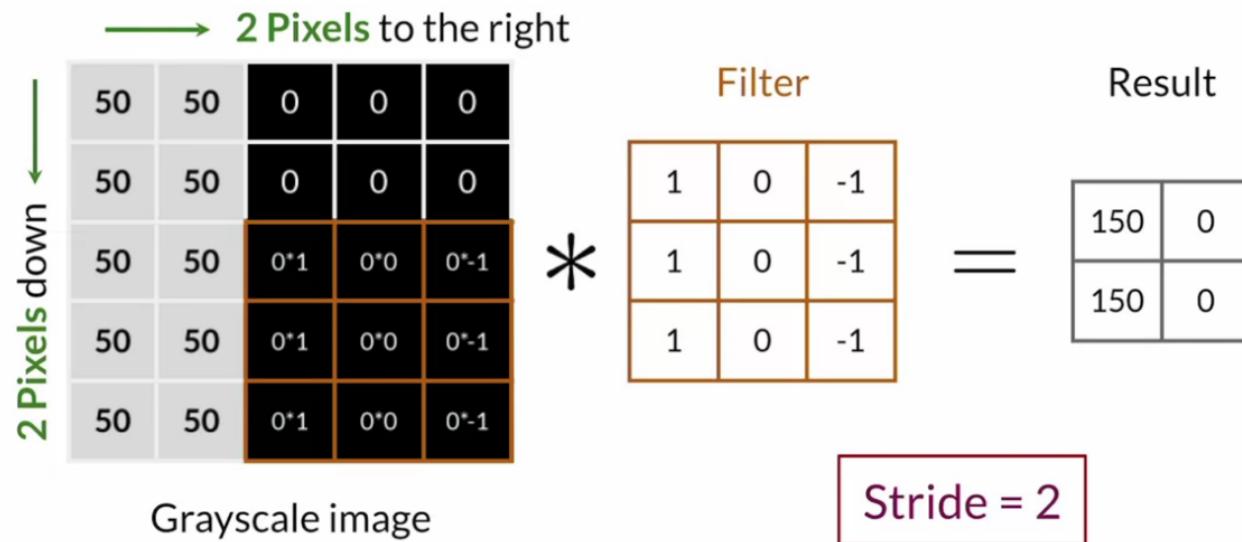
패딩(padding)과 보폭(stride)

- 패딩과 보폭
- 패딩의 직관

보폭(Stride)

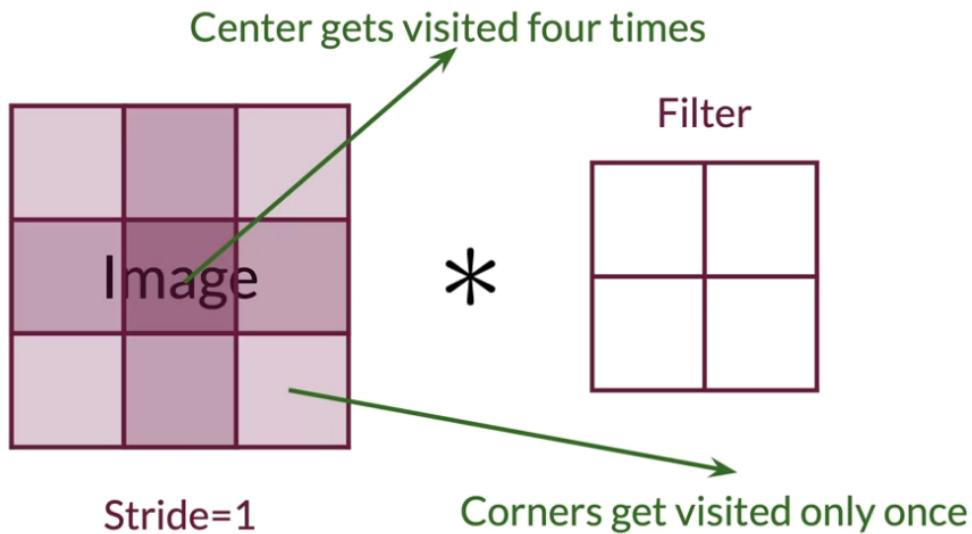




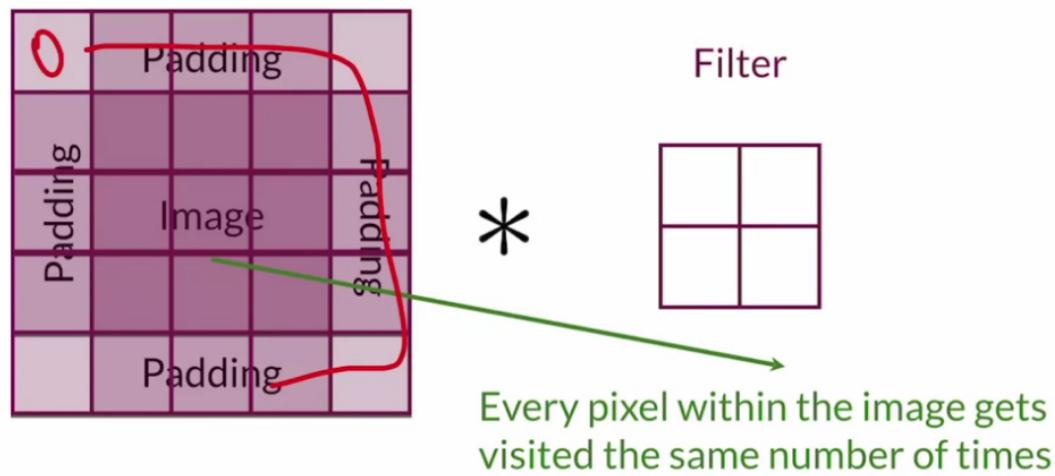


패딩(Padding)

Padding



Padding



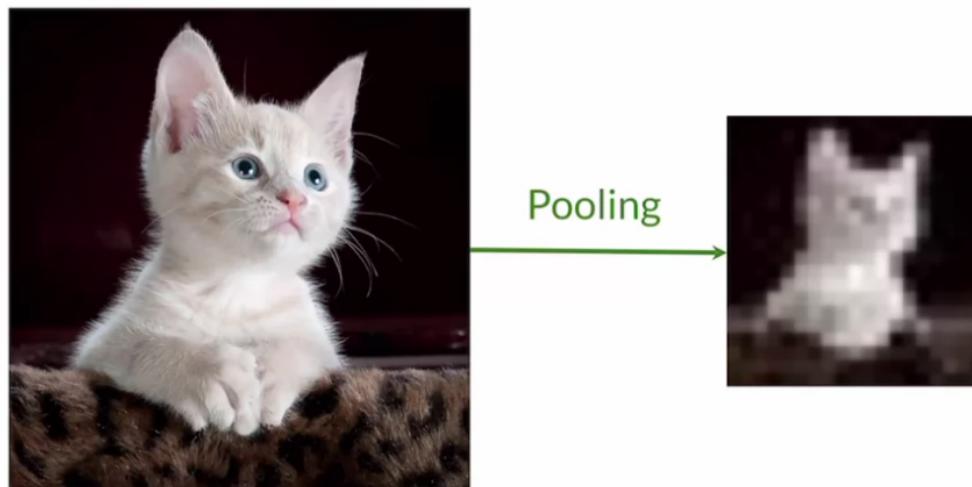
요약

- 보폭은 필터가 이미지를 어떻게 스캔할지를 결정한다.
- 패딩은 이미지의 프레임과 같다.
- 패딩은 가장자리와 중앙에서 유사한 중요도를 갖게 한다.

풀링(pooling)과 업샘플링(upsampling)

- 풀링
- 업샘플링과 풀링과의 관계

풀링



최대 풀링

0	20	30	45
8	1	23	43
0	4	40	20
10	6	43	42

2x2 pooling with stride = 2

Max pooling

20	

4x4 input to 2x2 output

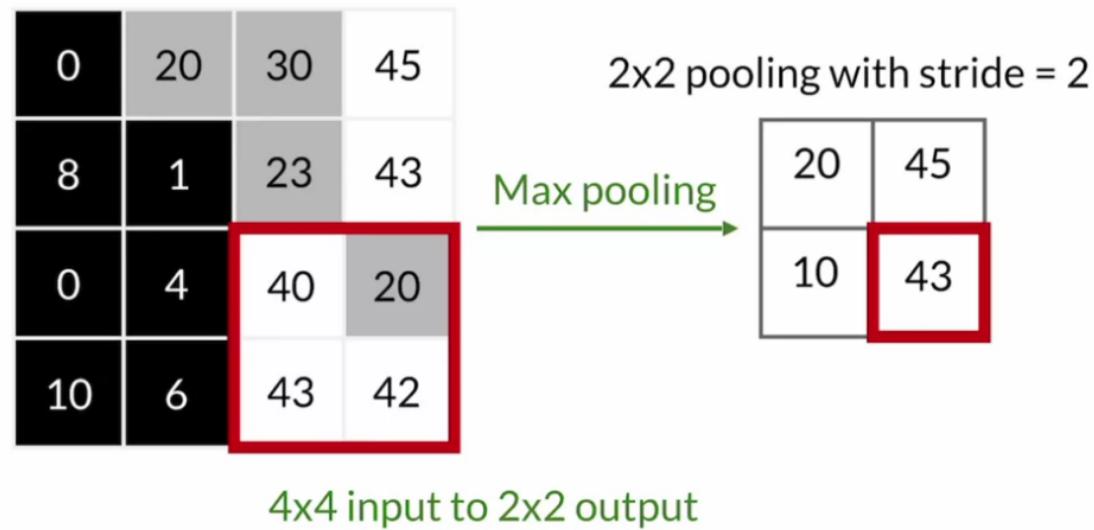
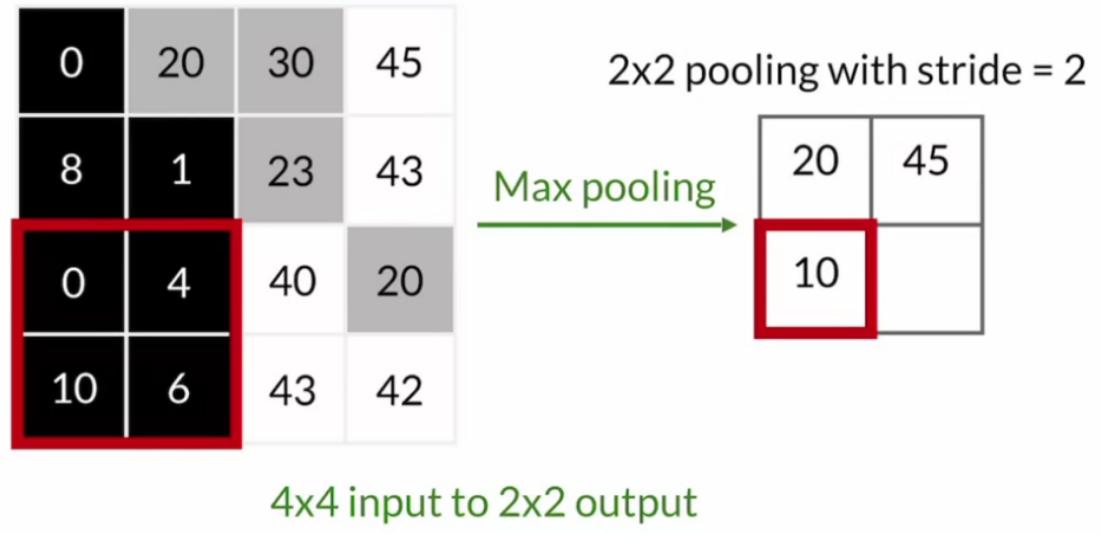
0	20	30	45
8	1	23	43
0	4	40	20
10	6	43	42

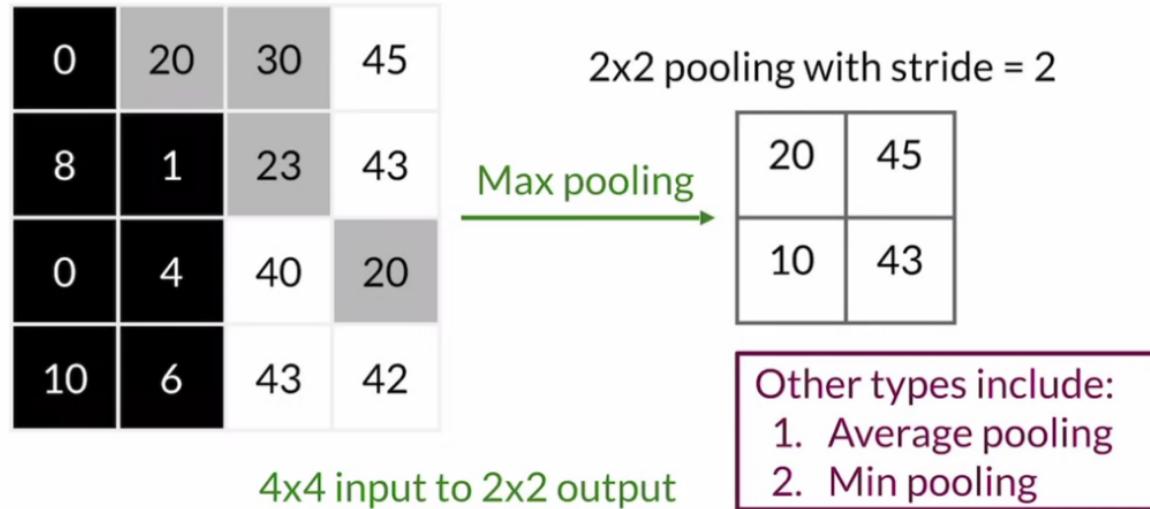
2x2 pooling with stride = 2

Max pooling

20	45

4x4 input to 2x2 output



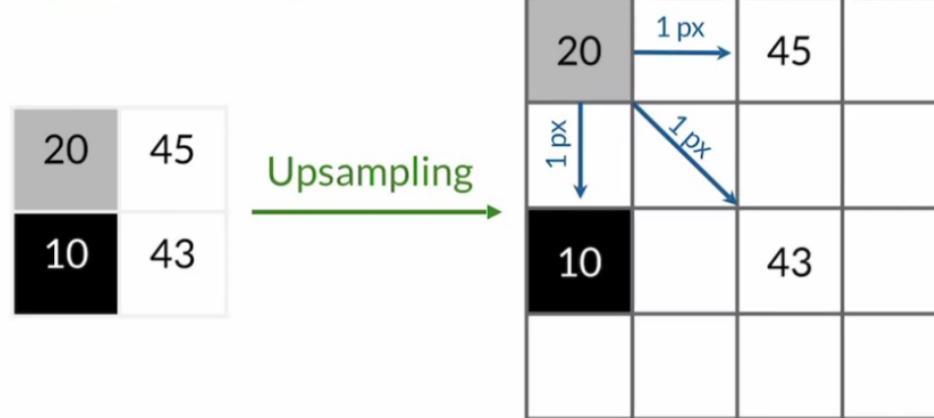


업샘플링

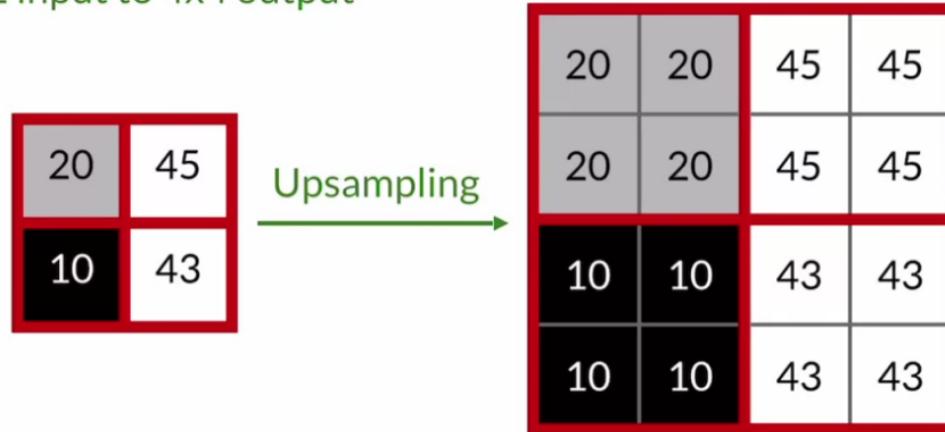


업샘플링 : 최근접 이웃(Nearest Neighbor)

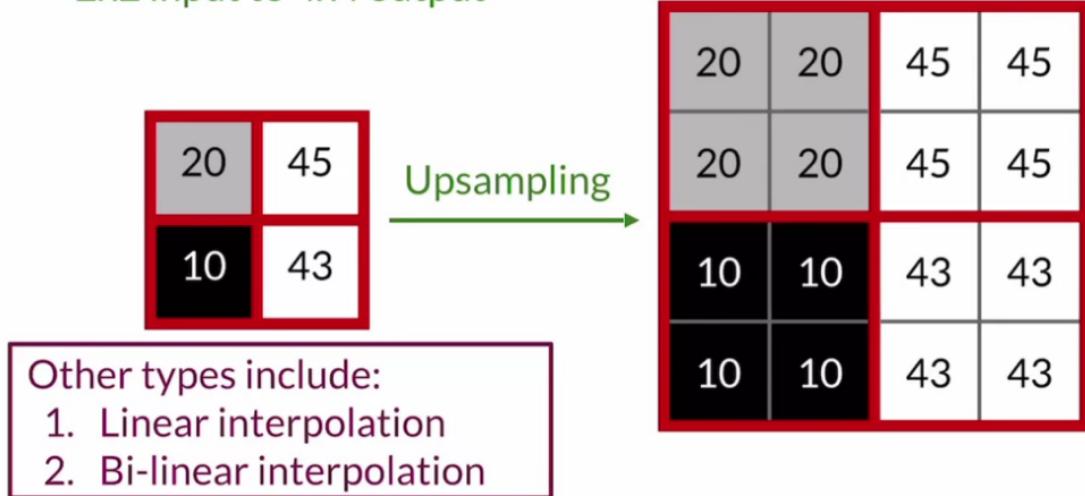
2x2 input to 4x4 output



2x2 input to 4x4 output



2x2 input to 4x4 output



요약

- 풀링은 이력의 크기를 줄인다.
- 업샘플링은 입력의 크기를 키운다.
- 학습할 매개변수가 없다.

전치된 컨볼루션(Transposed convolution)

- 전치된 컨볼루션은 업샘플링이다.
- 전치된 컨볼루션의 문제점들

전치된 컨볼루션

$$\begin{array}{c}
 \begin{array}{|c|c|} \hline 1 & 4 \\ \hline 0 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 2 & 2 \\ \hline 1 & 1 \\ \hline \end{array} = \begin{array}{|c|c|} \hline 1^*2 & 1^*2 \\ \hline 1^*1 & \\ \hline \end{array}
 \end{array}$$

Input Filter

Stride = 1

$$\begin{array}{c}
 \begin{array}{|c|c|} \hline 1 & 4 \\ \hline 0 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 2 & 2 \\ \hline 1 & 1 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline 1^*2 & 1^*2 + 4^*2 & 4^*2 \\ \hline 1^*1 & 1^*1 + 4^*1 & 4^*1 \\ \hline \end{array}
 \end{array}$$

Input Filter

Stride = 1

$$\begin{array}{|c|c|} \hline 1 & 4 \\ \hline 0 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 2 & 2 \\ \hline 1 & 1 \\ \hline \end{array}$$

Input Filter

=

$$\begin{array}{|c|c|c|} \hline 1*2 & 1*2+4*2 & 4*2 \\ \hline 1*1+0*2 & 1*1+4*1 & 4*1 \\ \hline 0*1 & 0*1 & \\ \hline \end{array}$$

Stride = 1

$$\begin{array}{|c|c|} \hline 1 & 4 \\ \hline 0 & 2 \\ \hline \end{array} * \begin{array}{|c|c|} \hline 2 & 2 \\ \hline 1 & 1 \\ \hline \end{array}$$

Input Filter

=

$$\begin{array}{|c|c|c|} \hline 1*2 & 1*2+4*2 & 4*2 \\ \hline 1*1+0*2 & 1*1+4*1 & 4*1+2*2 \\ \hline 0*1 & 0*1+2*1 & 2*1 \\ \hline \end{array}$$

Stride = 1

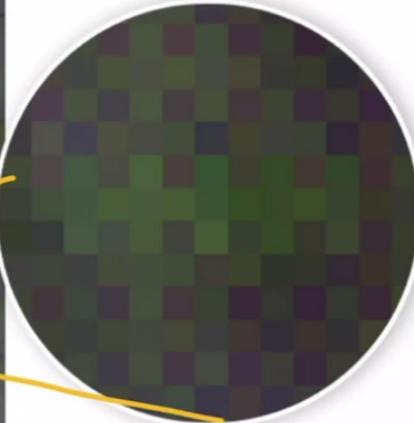
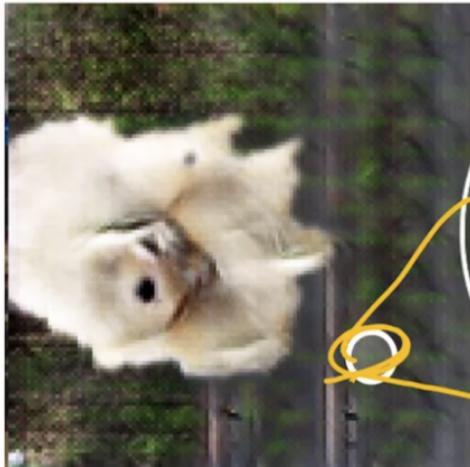
Influence from every value in the input

Input Filter Stride = 1

Influenced by just one value

Input Filter Stride = 1

전치된 컨볼루션의 문제



Checkerboard
Pattern

Available from: <http://doi.org/10.23915/distill.00003>

요약

- 전치된 컨볼류션은 업샘플링이다.
- 이들은 학습할 매개변수를 가진다.
- 문제점: 채커보드 패턴을 만든다.

In []: