가천대 회화·조소과 AI 특강

2021-06-22

조형래

A Neural Algorithm of Artistic Style

A Neural Algorithm of Artistic Style

A Neural Algorithm of Artistic Style

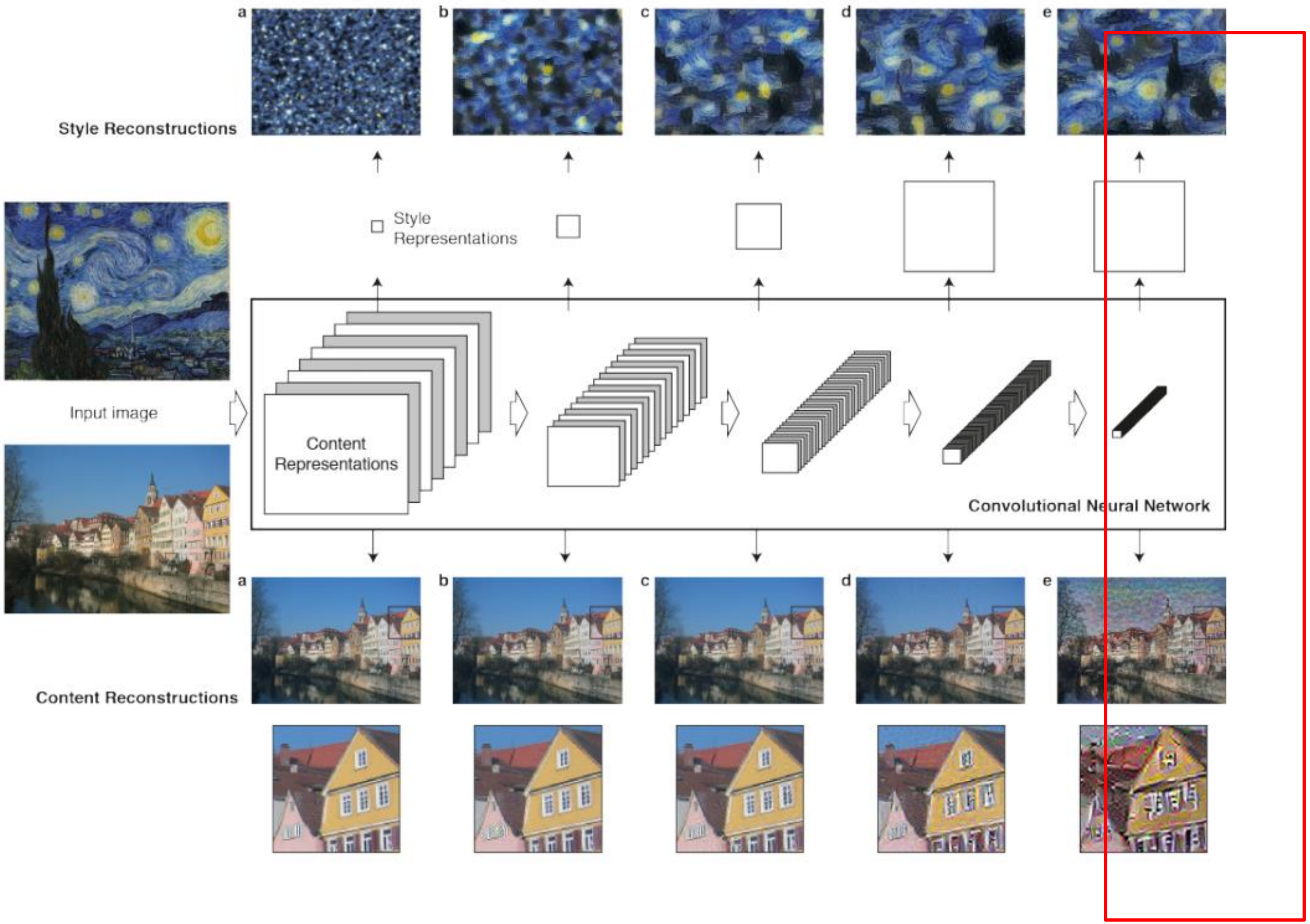
Leon A. Gatys,[1,2,3*] Alexander S. Ecker,[1,2,4,5] Matthias Bethge[1,2,4]

[1]Werner Reichardt Centre for Integrative Neuroscience
and Institute of Theoretical Physics, University of Tübingen, Germany
[2]Bernstein Center for Computational Neuroscience, Tübingen, Germany
[3]Graduate School for Neural Information Processing, Tübingen, Germany
[4]Max Planck Institute for Biological Cybernetics, Tübingen, Germany
[5]Department of Neuroscience, Baylor College of Medicine, Houston, TX, USA
[*]To whom correspondence should be addressed; E-mail: leon.gatys@bethgelab.org

cs.CV] 2 Sep 2015

논문: https://arxiv.org/pdf/1508.06576.pdf

CNN model: feature extraction



- content image(우측 하단 image) 의 경우, layer가 깊어질수록 원본 대비 detail한 pixel information은 소실되지만 high-level image, 즉 전체적인 윤곽인 건물의 모습은 유지된다.
- style image(우측 상단 Image)의 경우, layer가 깊어질수록 style image 원본에 가까워지게 된다.
- 이러한 현상이 발생하는 이유는 같은 layer의 feature map의 channel간 correlation(Gram Matrix)으로 정의하였기 때문

Gram Matrix; 한 Conv레이어 안에 있는 모든 피쳐 벡터들의 내적. 하나의 피쳐가 다른 피쳐와 얼마나 비슷하고 다른지를 나타내는 내적공간

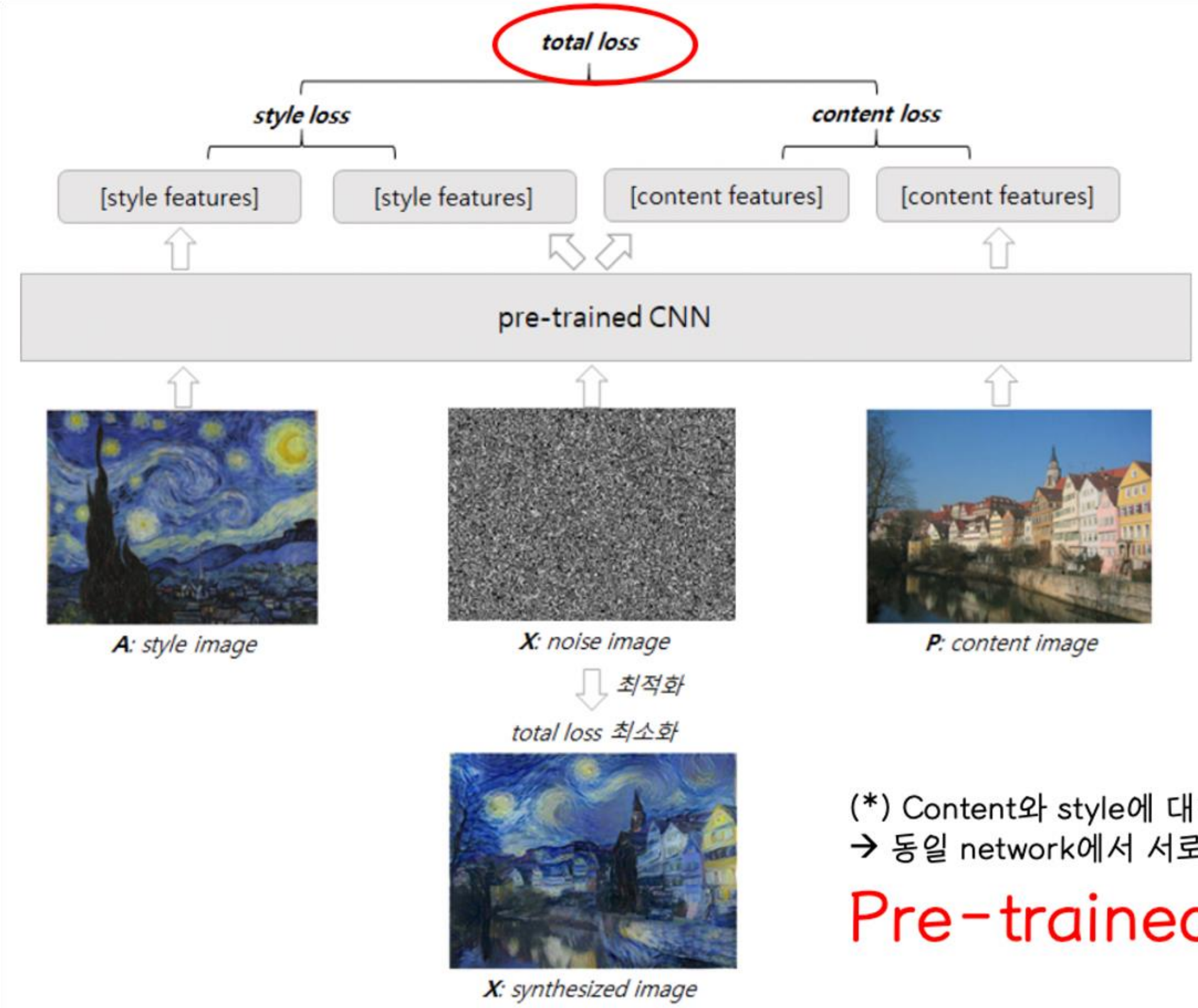
A Neural Algorithm of Artistic Style

Content image와 Style image가 주어졌을 때, 윤곽과 형태는 Content image와 유사하게 보존하면서
텍스쳐나 스타일만 원하는 Style image와 유사하게 바꾸는 것

$$x = \arg\max_{x} \alpha \mathcal{L}_{content}(x, A) + \beta \mathcal{L}_{style}(x, B)$$

optimization problem)
- loss function Lcontent(x,A)와 x와 B 간의 style이 얼마나 다른지 표현하는
loss function Lstyle(x,A)를 minimize하는 x를 찾는 것
- Total cost; 기존에 구한 content, style cost에 특정 상수값(alpha, beta)을
곱한 후 더하여 계산한다.

Algorithm Concept



Content of Content image

+

Style of Style image

=

Generated image

Pre-trained된 CNN
model의 feature 이용

(*) Content와 style에 대한 representation 분리
→ 동일 network에서 서로 다른 content와 style을 reconstruction하는 것이 가능

Pre-trained CNN model optimization

Optimization Problem

content 와 style 을 〉 reconstruction 해서 혼합.

optimization 과정 ; 임의의 image → input
(이미지의 parameter의 style이나 content에 대한 loss를 minimize 하는 것)

input image를 x라고 할 때.

목표: x와 A 간의 content 가 얼마나 다른지 표현하는 loss function $\mathcal{L}_{content}(x, A)$ 와 / x와 B 간의 style이 얼마나 다른지 표현하는 loss function $\mathcal{L}_{style}(x, A)$를 minimize 하는 x를 찾는 것.

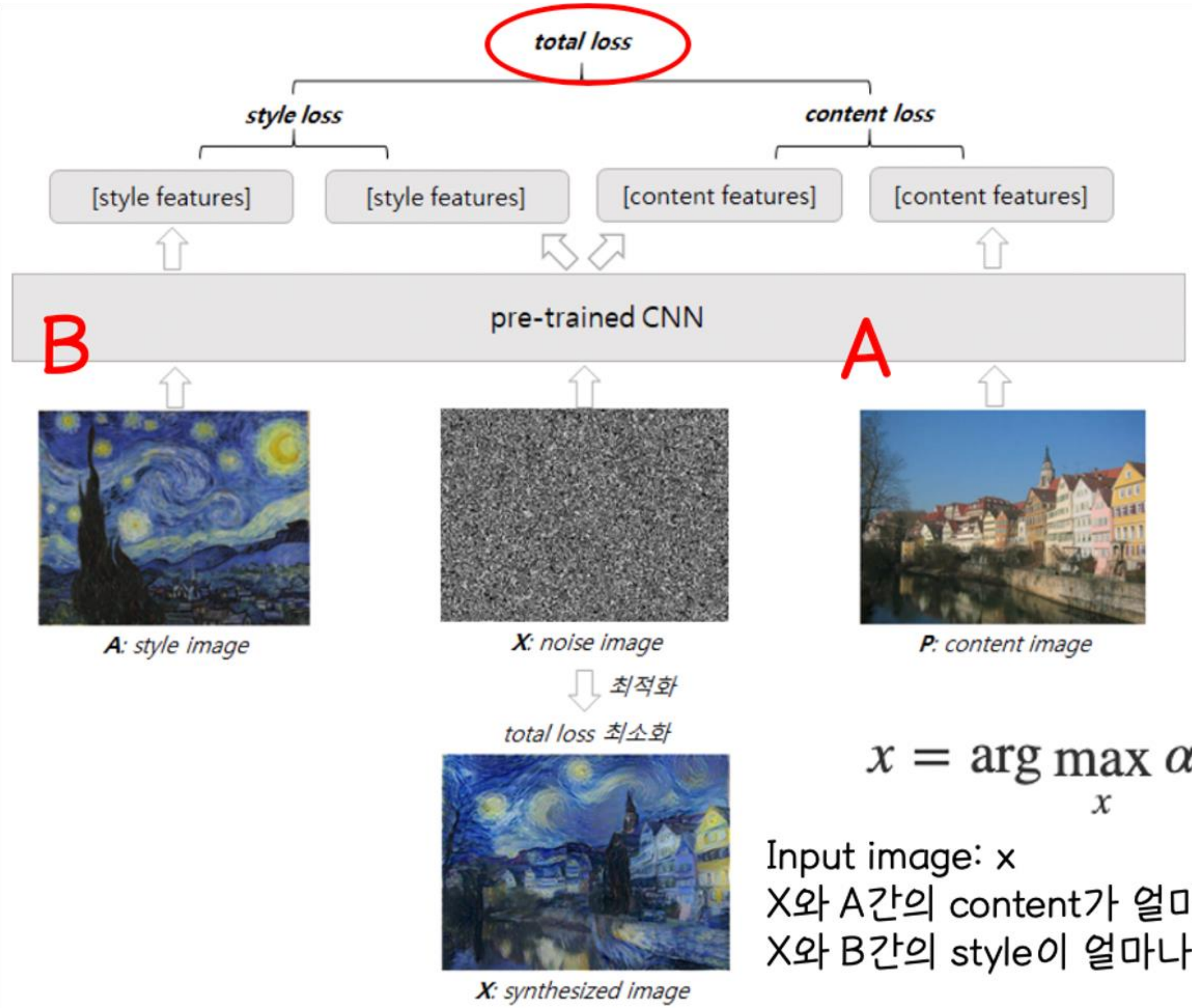
optimization problem:

$$x = \arg\max_{x} \alpha \mathcal{L}_{content}(x, A) + \beta \mathcal{L}_{style}(x, B)$$

α, β (상수)

→ x에 대한 gradient를 구하고 gradient decent optimization 하는 것.

Optimization Problem



A input의 content

+

B input의 style

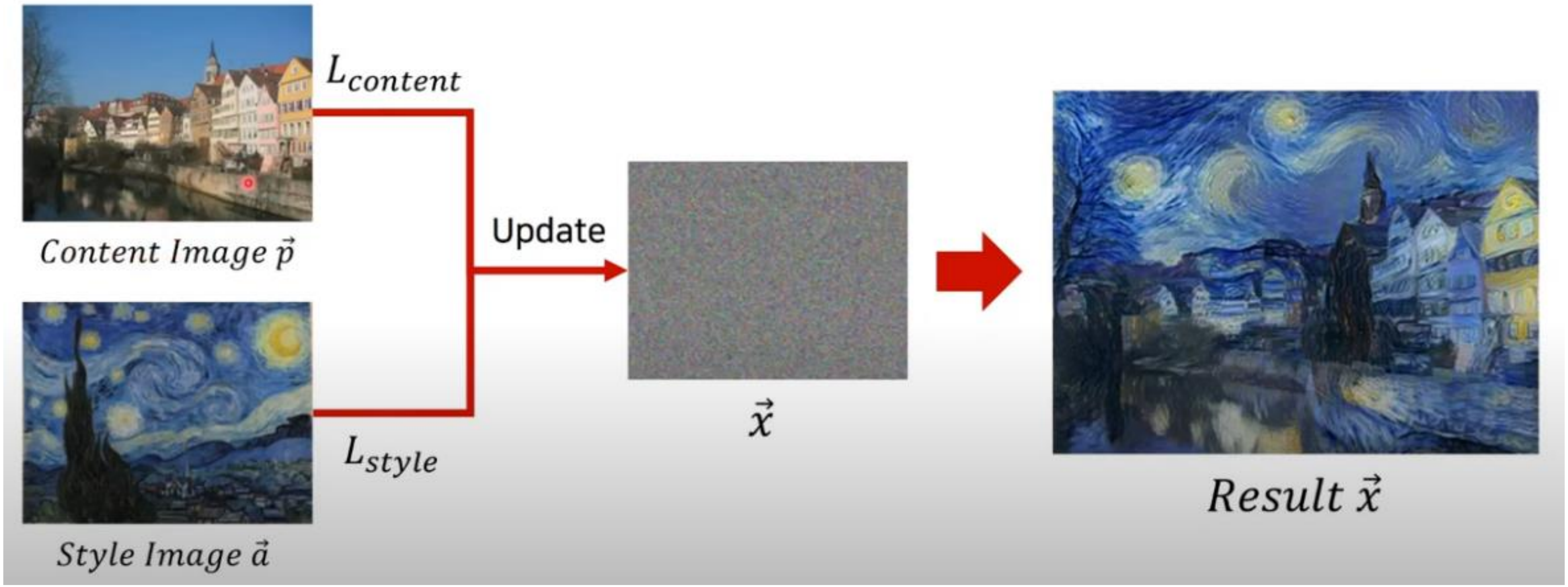
Input의 gradient 구하기

$$x = \arg\max_{x} \alpha \mathcal{L}_{content}(x, A) + \beta \mathcal{L}_{style}(x, B)$$

Input image: x
X와 A간의 content가 얼마나 다른지 표현하는 loss function과
X와 B간의 style이 얼마나 다른지 표현하는 loss function을 minimize하는 x

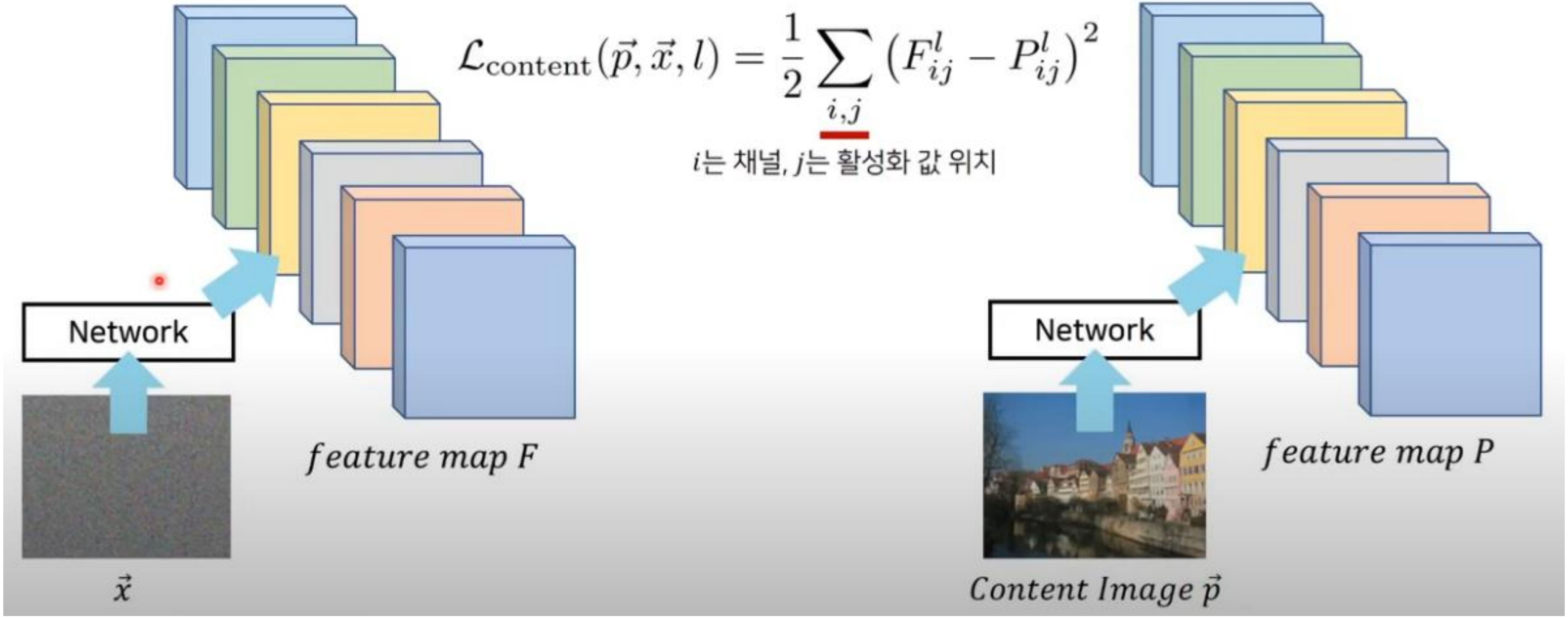
Optimization Problem

$$L_{total}(\vec{p},\vec{a},\vec{x}) = \alpha L_{content}(\vec{p},\vec{x}) + \beta L_{style}(\vec{a},\vec{x})$$



Content loss

두 이미지의 특징 activation 값이 동일하게 만든다.



Content loss

$\boxed{l\text{번째 content loss}}$

l번째 layer의 feature representation

- $\mathcal{L}_{content}(p, x, l) = \frac{1}{2}\sum_{ij}(F_{ij}^{l} - P_{ij}^{l})^2$

  이미지

  i번째 필터의 j번째 output

feature map : $F^l \in R^{N_l \times M_l}$

각각의 필터의 가로 세로를 곱한 값 (각 필터의 output 개수)

l번째 레이어의 필터 개수

⇒ p와 x에 대해 각각 feature map P^l, F^l 계산하고.

이들의 차 ($\|P^l - F^l\|_F$) 을 loss로 선택.

이 error를 각각의 layer에 대해 정의

- content 이미지 p의 l번째 layer의 representation을 사용해 image reconstruction을 할 때.

  l번째 layer에서 복원한 이미지 → x^l 라 하고

  x^l: loss를 minimize

$$x^l = \arg\max_x \mathcal{L}_{content}(p, x, l)$$

→ x^l을 random image로 initialize 하고

$\frac{\mathcal{L}_{content}(p,x,l)}{x}$ 을 계산 (gradient method)

Loss는 layer의 각각의 activation의 미분한 결과.

→ $\frac{\partial \mathcal{L}_{content}(p,x,l)}{\partial F_{ij}^{l}} = (F_{ij}^{l} - P_{ij}^{l})_{ij}$ if $F_{ij}^{l} > 0$, otherwise, 0

전체 gradient를 back-propagation 계산하여 복원한 이미지. (각각 conv1-1, con2-2, con3-1, conv4-1, con5-1 에서 loss를 계산하여 복원)

$$\mathcal{L}_{content}(p, x, l) = \frac{1}{2}\sum_{ij}(F_{ij}^{l} - P_{ij}^{l})^2.$$

$$x^l = \arg\max_x \mathcal{L}_{content}(p, x, l).$$

$$\frac{\partial \mathcal{L}_{content}(p, x, l)}{\partial F_{ij}^{l}} = (F_{ij}^{l} - P_{ij}^{l})_{ij} \text{ if } F_{ij}^{l} > 0, \text{ otherwise, } 0.$$

Style loss

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l$$

$$\mathcal{L}_{style}(a, x) = \sum_{l=0}^{L} w_l E_l$$

layer l의 style loss contribution

weight(자유롭게 1이라).

loss에 영향을 주는 layer의 개수

이미지 a, x 간의 style이 얼마나 다른지를 나타내는 style loss

$\mathcal{L}$ style

$$\Rightarrow E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

layer l의 style loss contribution

$$\Rightarrow \frac{\partial E_l}{\partial F_{ij}^l} = \frac{1}{N_l^2 M_l^2} \sum_{i,j} ((F^l)^T (G_{ij}^l - A_{ij}^l))_{ji}$$

if $F_{ij}^l > 0$ otherwise 0.

$\mathcal{L}$style(a, x)를 minimize 하는 x를 찾는 back-propagation

Total loss

$$\mathcal{L}_{total}(\vec{p},\vec{a},\vec{x}) = \alpha\mathcal{L}_{content}(\vec{p},\vec{x}) + \beta\mathcal{L}_{style}(\vec{a},\vec{x})$$

Total loss.

α와 β의 파라미터 조정에 따라 결과가 다르다.

$$\boxed{\mathcal{L}_{total}(p,a,x) = \alpha\mathcal{L}_{content}(p,x) + \beta\mathcal{L}_{style}(a,x)}$$

X 축이 α/β.

Y 축은 순서대로 conv1_1, conv2_1, conv3_1, .. conv5_1.

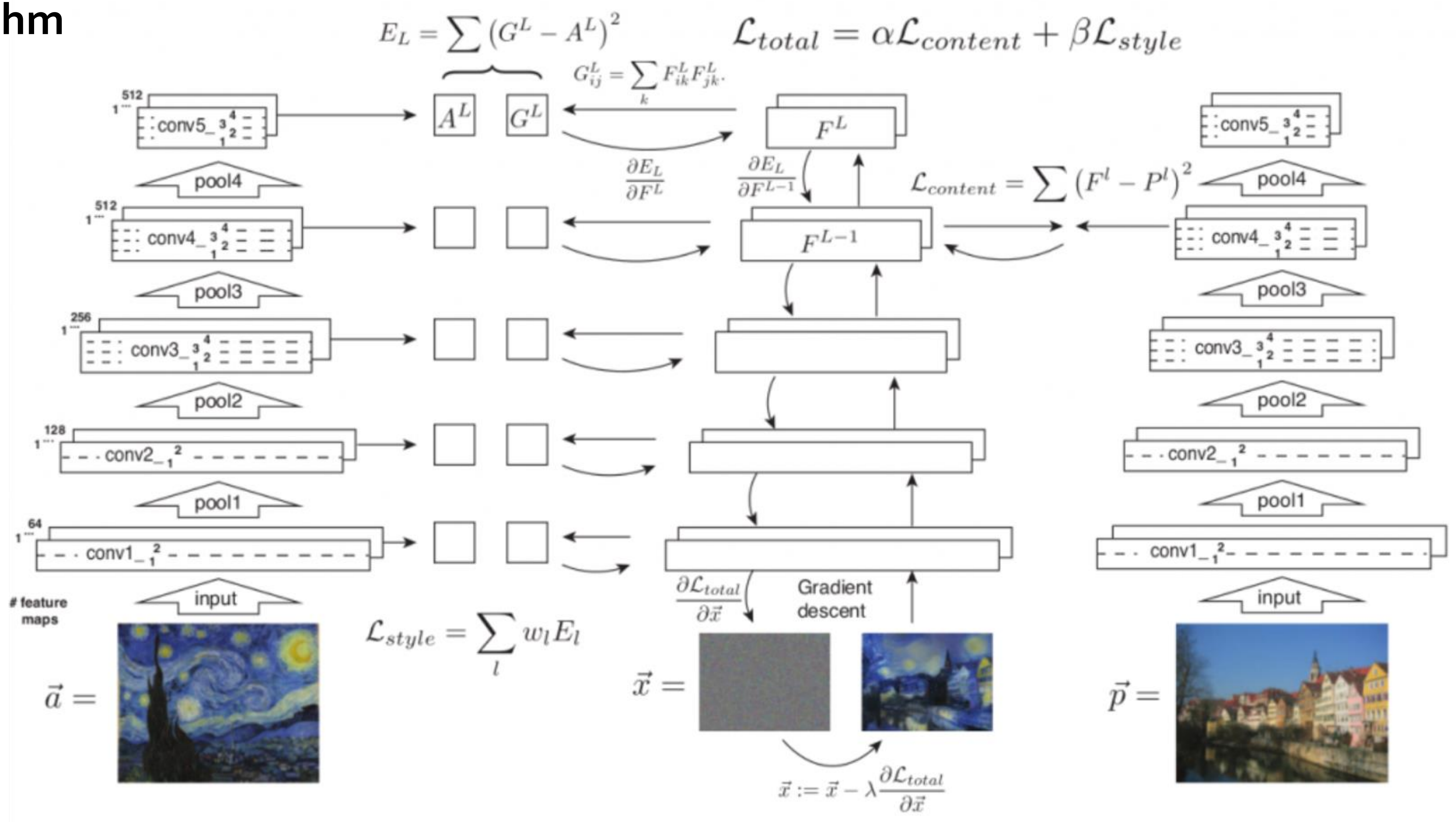
Conv 값의 조정에 따라.(Layer의 쌓는 정도)

style과 content의 적당한 trade-off가 조정됨.

Layer를 많이 쌓고, β값이 커질수록 style 값이 커진다.

Layer 적게 쌓고, β값이 작을수록 Content 값이 커진다.

Algorithm



$E_L = \sum (G^L - A^L)^2$
$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$
$G^L_{ij} = \sum_k F^L_{ik} F^L_{jk}.$
A^L
G^L
F^L
$\frac{\partial E_L}{\partial F^L}$
$\frac{\partial E_L}{\partial F^{L-1}}$
$\mathcal{L}_{content} = \sum (F^l - P^l)^2$
F^{L-1}
conv5_1 2 3 4
pool4
conv4_1 2 3 4
pool3
conv3_1 2 3 4
pool2
conv2_1 2
pool1
conv1_1 2
input
512
256
128
64
feature maps
$\frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$
Gradient descent
$\mathcal{L}_{style} = \sum_l w_l E_l$
$\vec{a} =$
$\vec{x} =$
$\vec{p} =$
$\vec{x} := \vec{x} - \lambda \frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$

CNN model: VGG19

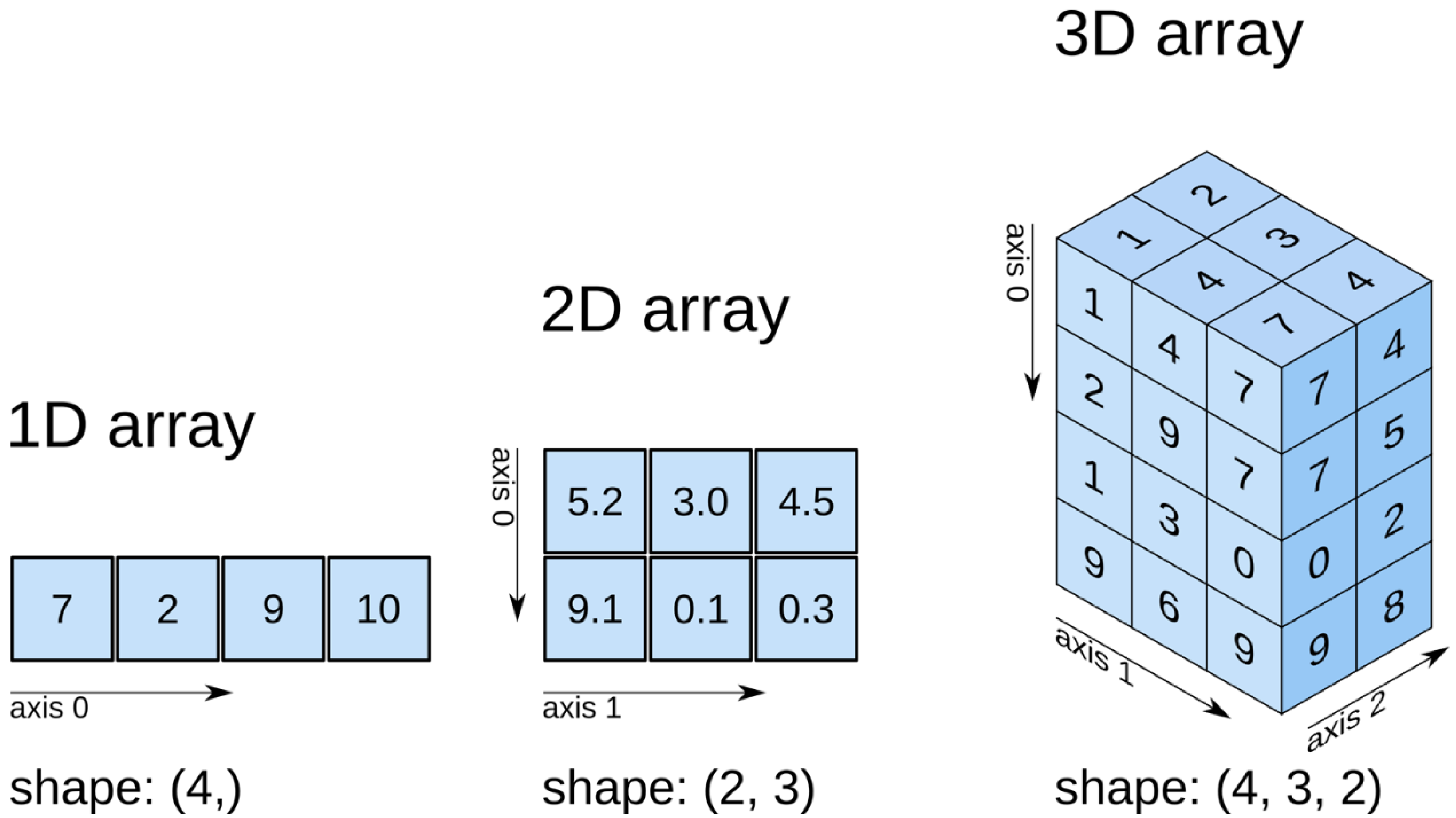
- CNN 모델로 VGG 19(총 16개의 convolution layer, 5개의 pooling layer, 3개의 fully connected layer로
  구성)네트워크 사용
- 16개의 convolution layer에서 생성되는 feature map을 사용해 style loss와 content loss를 계산한다.
- fully connected layer는 사용하지 않고
- image reconstruction에는 max pooling보다는 average pooling을 사용함으로서 그림이 조금 더 자연스럽고 좋아 보이는 결과로 나오기 때문

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					

Generated image 얻는 방법

1. Content image를 로드
2. Style image를 로드
3. Generated image를 random 값으로 초기화
4. pre-trained된 VGG model을 로드
5. VGG model에 입력 이미지로 content image를 입력 및 실행 후(forward-propagation), content cost를 계산
6. VGG model에 입력 이미지로 style image를 입력 및 실행 후(forward-propagation), style cost를 계산
7. total cost를 계산
8. total cost를 minimize하기 위해 optimizer를 정의
9. 학습을 최대 iteration만큼 수행하고, 각 iteration별로 generated image를 저장
10. 마지막 iteration때 생성된 generated image로 최종 결과를 확인

Array(배열)



1D array
7
2
9
10
axis 0
shape: (4,)
2D array
5.2
3.0
4.5
9.1
0.1
0.3
axis 0
axis 1
shape: (2, 3)
3D array
axis 0
axis 1
axis 2
shape: (4, 3, 2)

실습

Figure 7. Photorealistic style transfer. The style is transferred from a photograph showing New York by night onto a picture showing London by day. The image synthesis was initialised from the content image and the ratio α/β was equal to 1×10^{-2}

10의 -1승=1/10
10의 -6승=1/1,000,000
10의-2승=1/100

실습

스타일)클레, 모네, 두부
!wget --quiet -P /tmp/nst/ https://img.hani.co.kr/imgdb/resize/2007/1008/04544065_20071008.JPG
!wget --quiet -P /tmp/nst/ https://gonggam.korea.kr/goNewsRes/attaches/2021.01/11/9f1566d2fb982c6b2508792ed06b98ee.jpg
!wget --quiet -P /tmp/nst/ https://src.hidoc.co.kr/image/lib/2020/8/19/1597827889881_0.jpg

content) 시골풍경1, 시골풍경2, 홍콩빌라
!wget --quiet -P /tmp/nst/ https://cdn.crowdpic.net/detail-thumb/thumb_d_634DD73647A7DC190A415272700700F9.jpg
!wget --quiet -P /tmp/nst/ https://i.ytimg.com/vi/h8qIiEzDQqI/hqdefault.jpg
!wget --quiet -P /tmp/nst/ https://ncache.ilbe.com/files/attach/new/20190913/377678/10047983340/11198785795/8a01e183215b11eb92e84037ef5df4f6_11198785930.jpg

https://github.com/artjow/-AI-/blob/main/ART/Neural_Style_Transfer_with_Eager_Execution_(1).ipynb

https://github.com/artjow/-AI-/blob/main/ART/Style_Transfer_Tutorial.ipynb