

꼼꼼한 딥러닝 논문 리뷰와 코드 실습

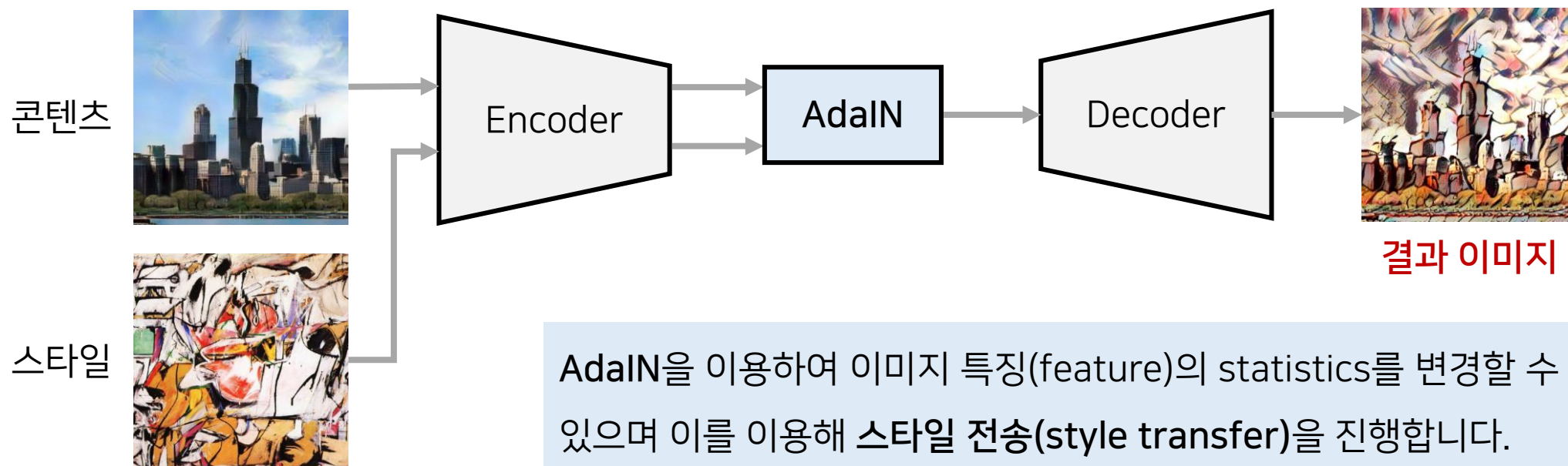
Deep Learning Paper Review and Code Practice

나동빈(dongbinna@postech.ac.kr)

Pohang University of Science and Technology

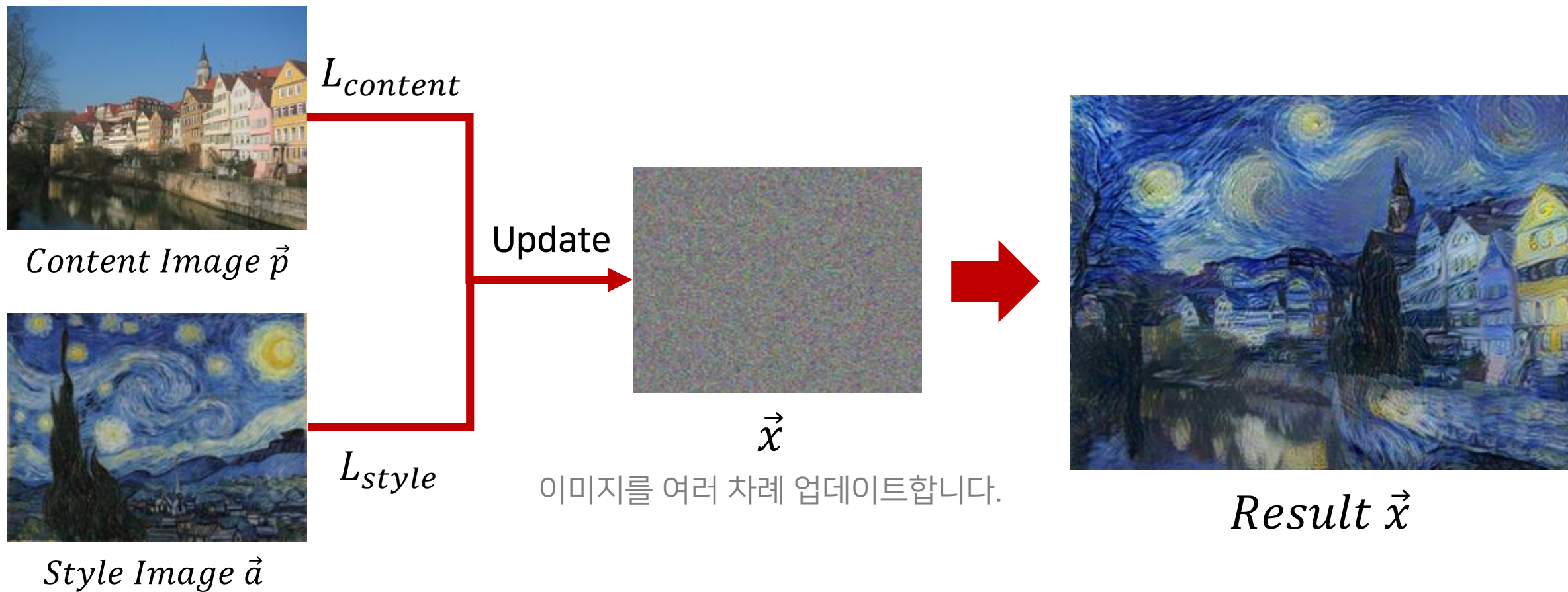
논문 소개: AdaIN Style Transfer (ICCV 2017)

- 본 논문에서 제안한 메서드: Adaptive Instance Normalization (AdaIN)
- 본 논문의 **장점**
 - ① 자신이 원하는 임의의(arbitrary) 스타일 이미지로부터 스타일 정보를 가져올 수 있습니다.
 - ② 실시간으로(real-time) 빠르게 스타일 전송(style transfer)을 진행할 수 있습니다.



연구 배경: Image Style Transfer Using Convolutional Neural Networks (CVPR 2016)

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



연구 배경: Style Reconstruction과 Content Reconstruction 초창기 방법 (CVPR 2016)

[Style Reconstructions]

- Gram Matrix는 채널의 크기만큼 커지게 됩니다.

(a) conv1_1

(b) conv1_1, conv2_1

(c) conv1_1, conv2_1, conv3_1

(d) conv1_1, conv2_1, conv3_1, conv4_1,

(e) conv1_1, conv2_1, conv3_1, conv4_1, conv5_1

[Content Reconstructions]

- 깊어질수록 구체적인 픽셀 정보는 소실됩니다.

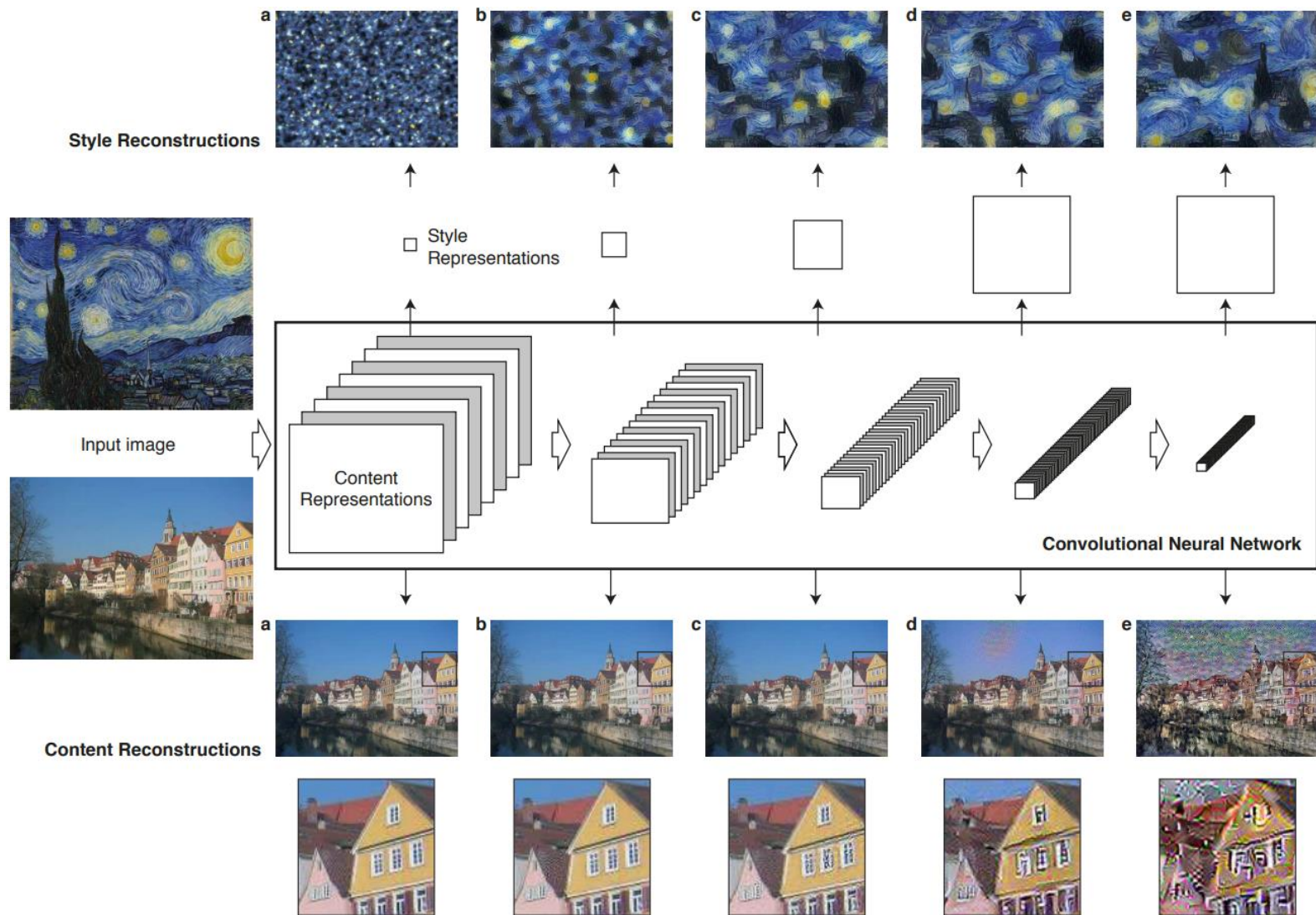
(a) conv1_2

(b) conv2_2

(c) conv3_2

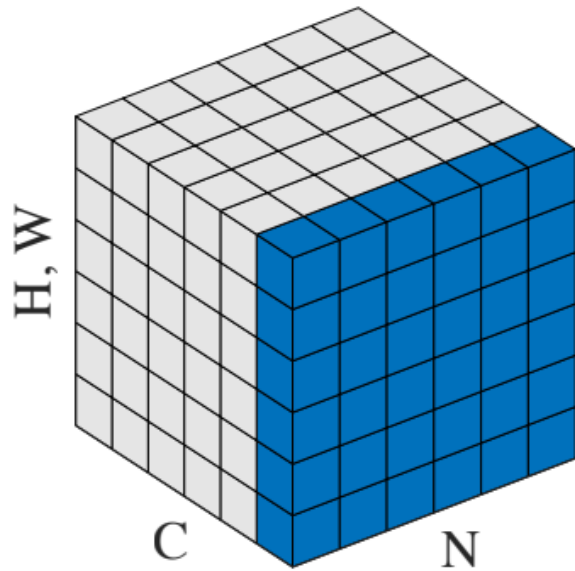
(d) conv4_2

(e) conv5_2



연구 배경: 배치 정규화 (Batch Normalization)

- 채널별로(channel-wise) 정규화를 수행합니다.
 - 각 채널에 적용할 파라미터인 γ 와 β 를 학습합니다.



텐서(Tensor)

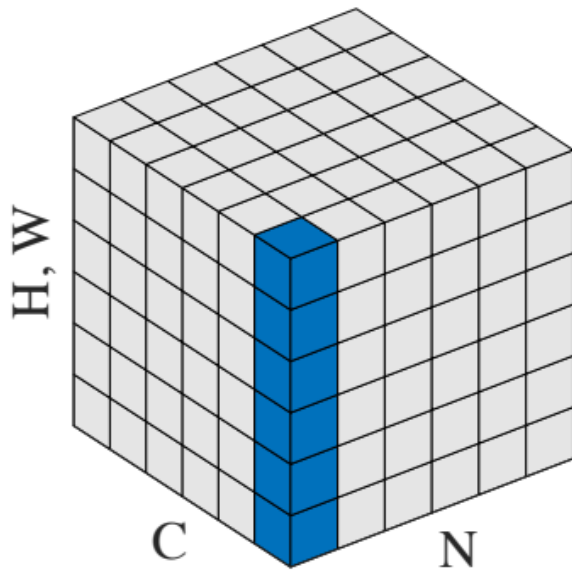
$$\text{BN}(x) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta$$

$$\mu_c(x) = \frac{1}{NHW} \sum_{n=1}^N \sum_{h=1}^H \sum_{w=1}^W x_{nchw}$$

$$\sigma_c(x) = \sqrt{\frac{1}{NHW} \sum_{n=1}^N \sum_{h=1}^H \sum_{w=1}^W (x_{nchw} - \mu_c(x))^2 + \epsilon}$$

연구 배경: 인스턴스 정규화 (Instance Normalization)

- 개별적인 **이미지(인스턴스)** 대하여 각 채널별로(channel-wise) 정규화를 수행합니다.
 - 개별적인 샘플(이미지)에 대하여 수행한다는 점이 배치 정규화(BN)과의 차이점입니다.



텐서(Tensor)

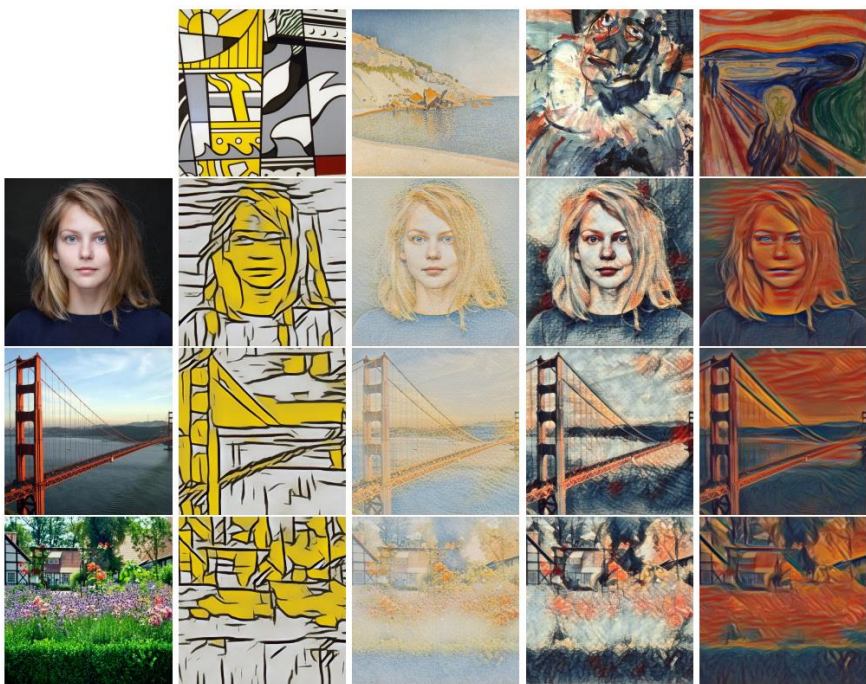
$$\text{IN}(x) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta$$

$$\mu_{nc}(x) = \frac{1}{HW} \sum_{h=1}^H \sum_{w=1}^W x_{nchw}$$

$$\sigma_{nc}(x) = \sqrt{\frac{1}{HW} \sum_{h=1}^H \sum_{w=1}^W (x_{nchw} - \mu_{nc}(x))^2 + \epsilon}$$

연구 배경: 조건부 인스턴스 정규화 (Conditional Instance Normalization)

- Instance Normalization (IN)을 수행할 때 조건에 따라 다른 feature statistics를 갖도록 합니다.



$$\text{CIN}(x; s) = \gamma^s \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta^s$$



CIN을 이용한 스타일 전송(style transfer)을 위해
2FS만큼의 추가적인 파라미터가 필요합니다.

*F: Feature Map의 개수

- 서로 다른 affine parameters(γ^s 와 β^s)를 적용해 완전히 다른 스타일(style)의 이미지를 만들 수 있습니다.
 - Dumoulin의 논문에서 CIN을 제안했고 32개의 스타일(s)에 대하여 성공적으로 학습했습니다.

메인 아이디어: Adaptive Instance Normalization (AdaIN)

- 연구 동기: CIN으로 feature statistics를 변경하는 것만으로 style transfer의 효과를 낼 수 있음
 - 그렇다면 미리 고정된 style을 사용하지 말고 임의의(arbitrary) 이미지로부터 스타일을 가져오자!
- AdaIN을 이용해 다른 원하는 이미지에서 스타일(style) 정보를 가져와 적용할 수 있습니다.
 - 학습시킬 파라미터가 필요하지 않습니다. (γ 와 β 사용하지 않음)
 - feed-forward 방식의 style transfer 네트워크에서 사용되어 좋은 성능을 보입니다.

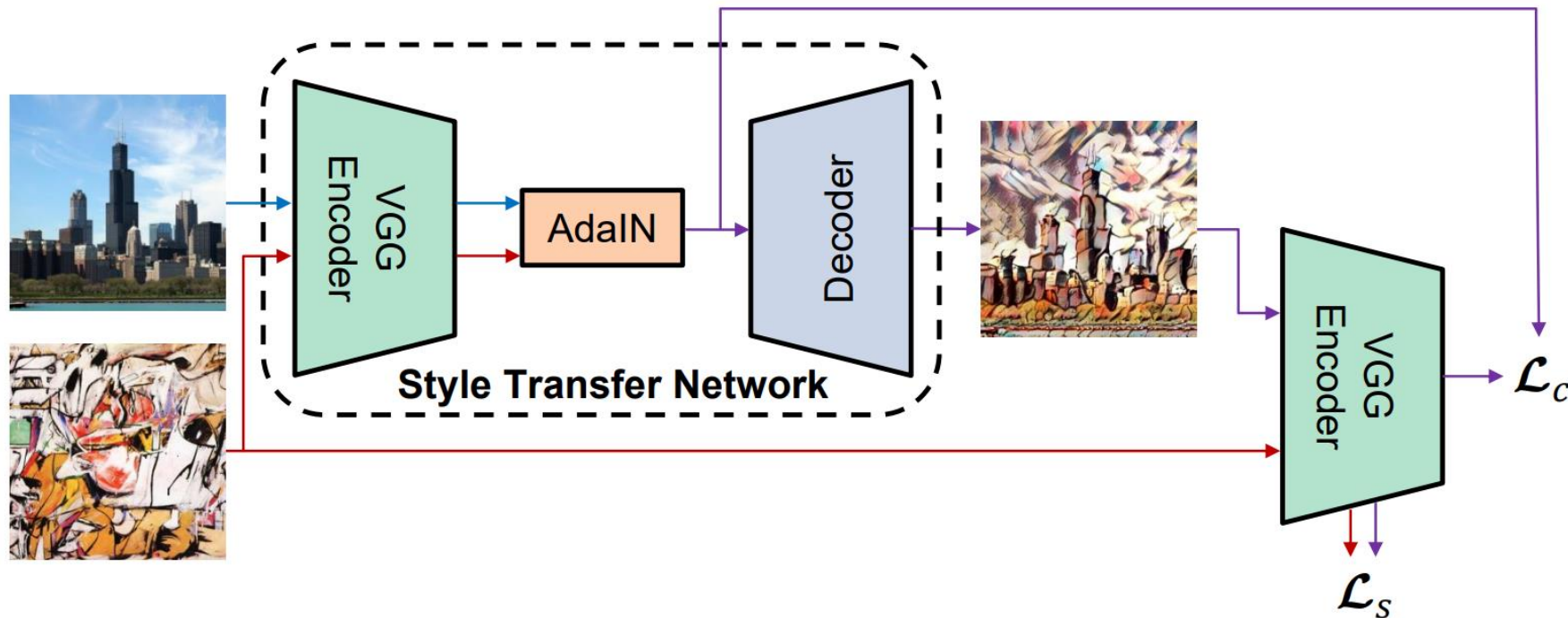
$$\text{AdaIN}(x, y) = \sigma(y) \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \mu(y)$$

x : 콘텐츠 이미지(content image)

y : 스타일 이미지(style image)

Adaptive Instance Normalization (AdaIN)

- 네트워크 중간에 있는 AdaIN이 스타일 전송(style transfer) 역할을 수행합니다.



- VGG Encoder: 특징 추출 목적으로 사전 학습되어 고정된(fixed) 네트워크
- Decoder: 학습할 네트워크이며 결과 이미지를 생성하는 역할 수행

Adaptive Instance Normalization (AdaIN)

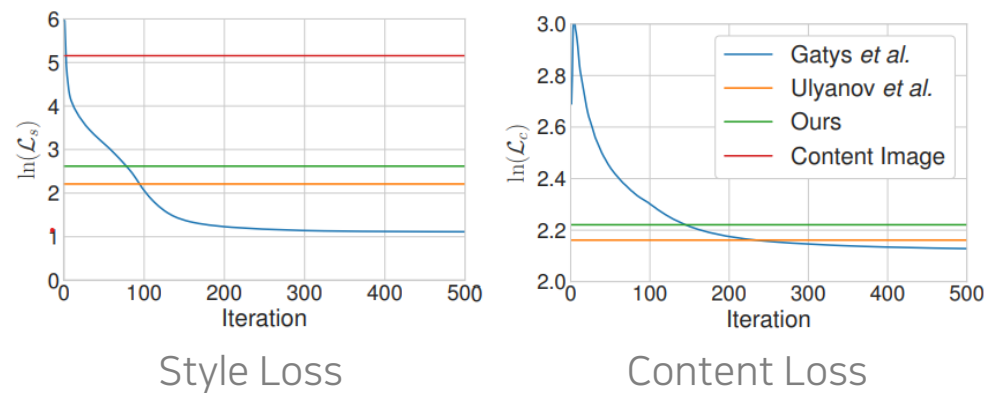
- 최종 목적 함수: $\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_s$
- 콘텐츠 손실: $\mathcal{L}_c = \|f(g(t)) - t\|_2$
- 스타일 손실: $\mathcal{L}_s = \sum_{i=1}^L \underbrace{\|\mu(\phi_i(g(t))) - \mu(\phi_i(s))\|_2}_{i\text{번째 스타일 feature}} + \sum_{i=1}^L \|\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))\|_2$

$$\left[\begin{array}{ll} t = \text{AdaIN}(f(c), f(s)) & \longrightarrow \text{Style Transfer를 수행한 feature 정보} \\ T(c, s) = g(t) & \longrightarrow \text{디코더(decoder)를 거쳐 생성된 결과 이미지} \end{array} \right.$$

AdaIN Style Transfer 결과

Method	Time (256px)	Time (512px)	# Styles
Gatys <i>et al.</i>	14.17 (14.19)	46.75 (46.79)	∞
Chen and Schmidt	0.171 (0.407)	3.214 (4.144)	∞
Ulyanov <i>et al.</i>	0.011 (N/A)	0.038 (N/A)	1
Dumoulin <i>et al.</i>	0.011 (N/A)	0.038 (N/A)	32
Ours	0.018 (0.027)	0.065 (0.098)	∞

[Table] Speed comparison



Style

Content

Ours

Chen and Schmidt

Ulyanov et al.

Gatys et al.

Runtime Control ① Content-style Trade-off

$$T(c, s, \alpha) = g((1 - \alpha)f(c) + \alpha \text{AdaIN}(f(c), f(s)))$$

원래 Style Transfer에서 사용되는 부분



$\alpha = 0$



$\alpha = 0.25$



$\alpha = 0.5$



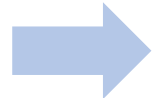
$\alpha = 0.75$



$\alpha = 1$



Style



α 가 0에 가까울수록 콘텐츠(content) 이미지에 가까운 결과 이미지가 생성됩니다.

Runtime Control ② Style Interpolation

$$T(c, s_{1,2,...K}, w_{1,2,...K}) = g\left(\sum_{k=1}^K w_k \text{AdaIN}(f(c), f(s_k))\right)$$

- K개의 스타일 이미지에 존재하는 스타일을 골고루 섞어 Style Transfer를 수행할 수 있습니다.
- feature maps 사이의 Interpolation으로 간주할 수 있습니다.

