Review of

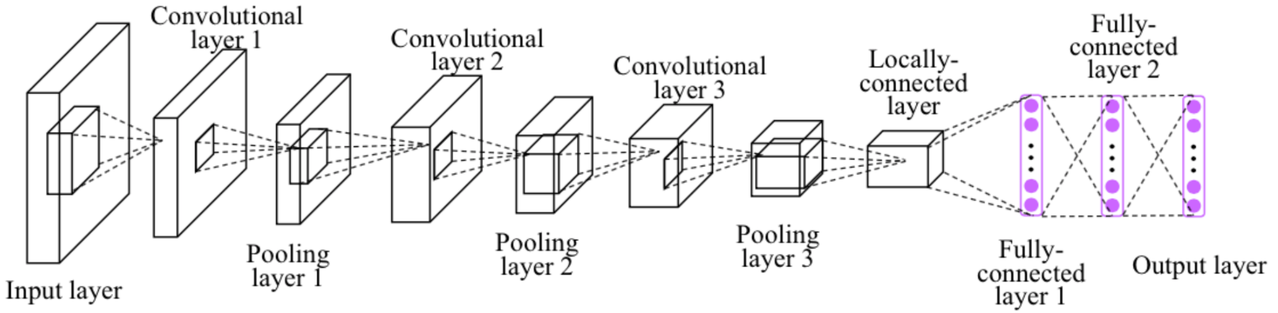
Image Style Transfer using Neural Networks

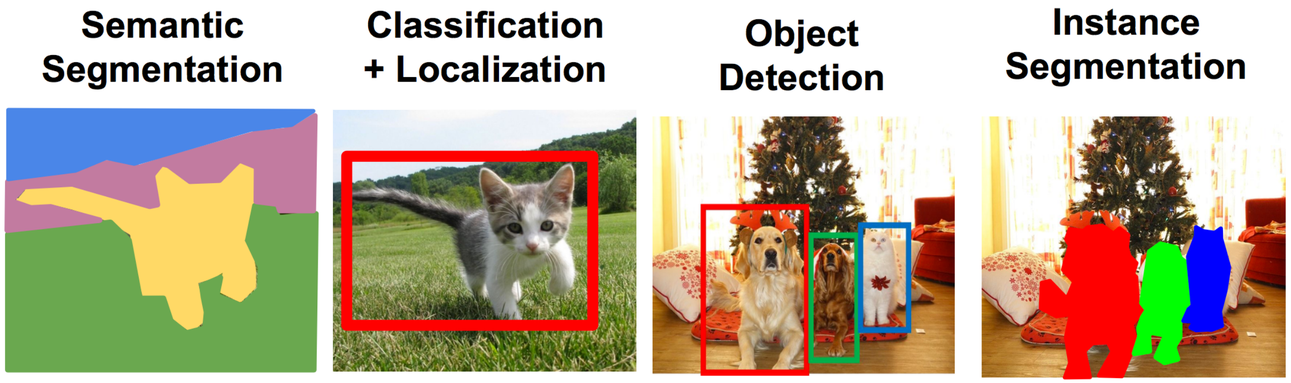
POSTECH 창의IT융합공학과 통합과정
엑셈-포스텍 R&D 센터
이도엽

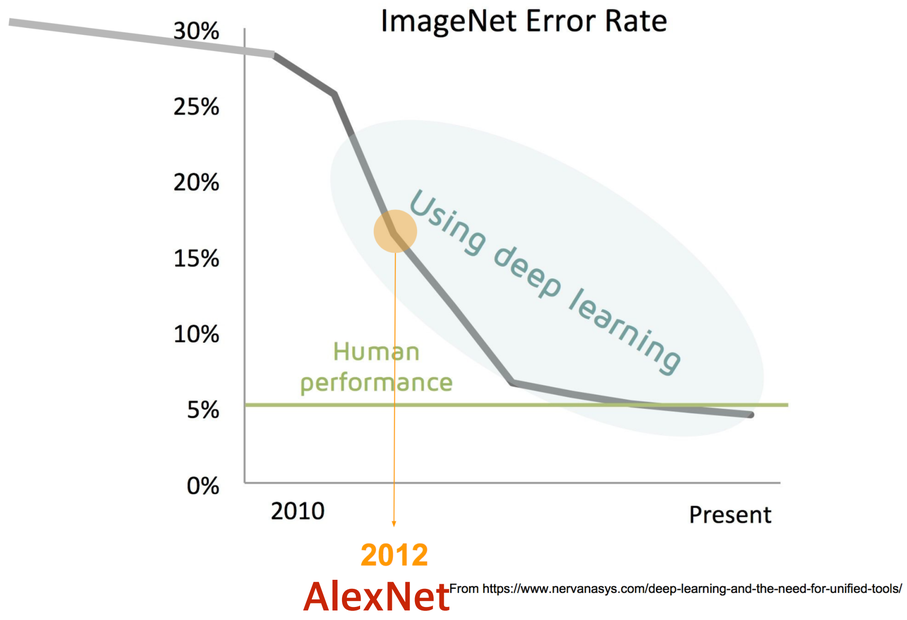
Image Style Transfer using ConvNet

Introduction

- Convolutional Neural Network (ConvNet) 기반의 딥러닝 모델은 이미지 처리 분야에서 높은 성능을 보여주었음
- ConvNet의 Hidden layer에서 이미지를 잘 나타내는 특징(representative feature)을 학습하여 표현함
- 과거에는 이미지의 특징을 찾는 Feature Encoding에 집중하여 연구를 진행해왔음







Content Generation with NN

- **이미지의 특징을 잘 계산할 수 있다면, 특징을 바탕으로 이미지를 다시 만들어 수 있지 않을까 (Reconstruction)?**
- **Feature Inversion**

  : 특정한 ConvNet feature vector가 주어졌을 때 1) 주어진 특징과 잘 맞고 (match), 2) 어색하지 않고 자연스러운 이미지를 생성하는 것

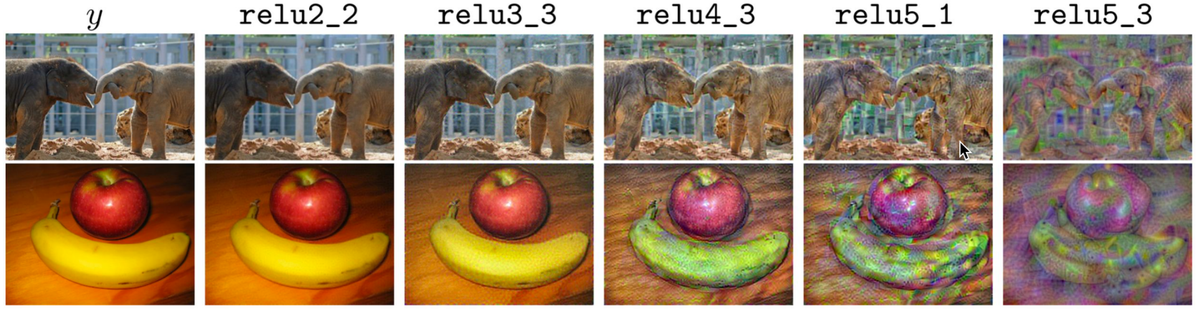
$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathbb{R}^{H \times W \times C}}{\operatorname{argmin}} \ell(\Phi(\mathbf{x}), \Phi_0) + \lambda \mathcal{R}(\mathbf{x})$$

$$\ell(\Phi(\mathbf{x}), \Phi_0) = \|\Phi(\mathbf{x}) - \Phi_0\|^2$$

$$\mathcal{R}_{V^\beta}(\mathbf{x}) = \sum_{i,j} \left((x_{i,j+1} - x_{ij})^2 + (x_{i+1,j} - x_{ij})^2 \right)^{\frac{\beta}{2}}$$

: Total Variation Regularizer

<Reconstructing from Different Layers of VGG-16>



Mahendran and Vedaldi, “Understanding Deep Image Representations by Inverting Them”, CVPR 2015

Basic Assumption of Style Transfer

- 사진이나 그림은 표현의 대상인 **내용(Content)**과 표현 방식인 **스타일(Style)**이 존재하며 이는 **구분 가능**하다고 전제함
- **Question: 내용은 그대로 유지하면서 스타일만 바꿀 수 있지 않을까?**

내용: 하늘, 강, 건물들…
스타일: 경계가 뚜렷, 사실적, 낮 …

+

내용: 마을, 성당, 별, 하늘 …
스타일: 추상적인, 고흐 풍..…

=

내용: 하늘, 강, 건물들…
스타일: 추상적인, 고흐 풍…

General Approach for Neural Style Transfer

- Content 이미지와 Style 이미지를 Input Image로 받고, 목적에 잘 맞는 새로운 이미지를 찾는 방식으로 이미지 생성
- 학습의 대상은 ConvNet 파라미터가 아닌 모델이 <u>생성하고자 이미지</u> (input X)

Content Input

Generated Image (X)

Style Transfer Model (Image Generator)

Style Input

Learn "X" with minimum loss

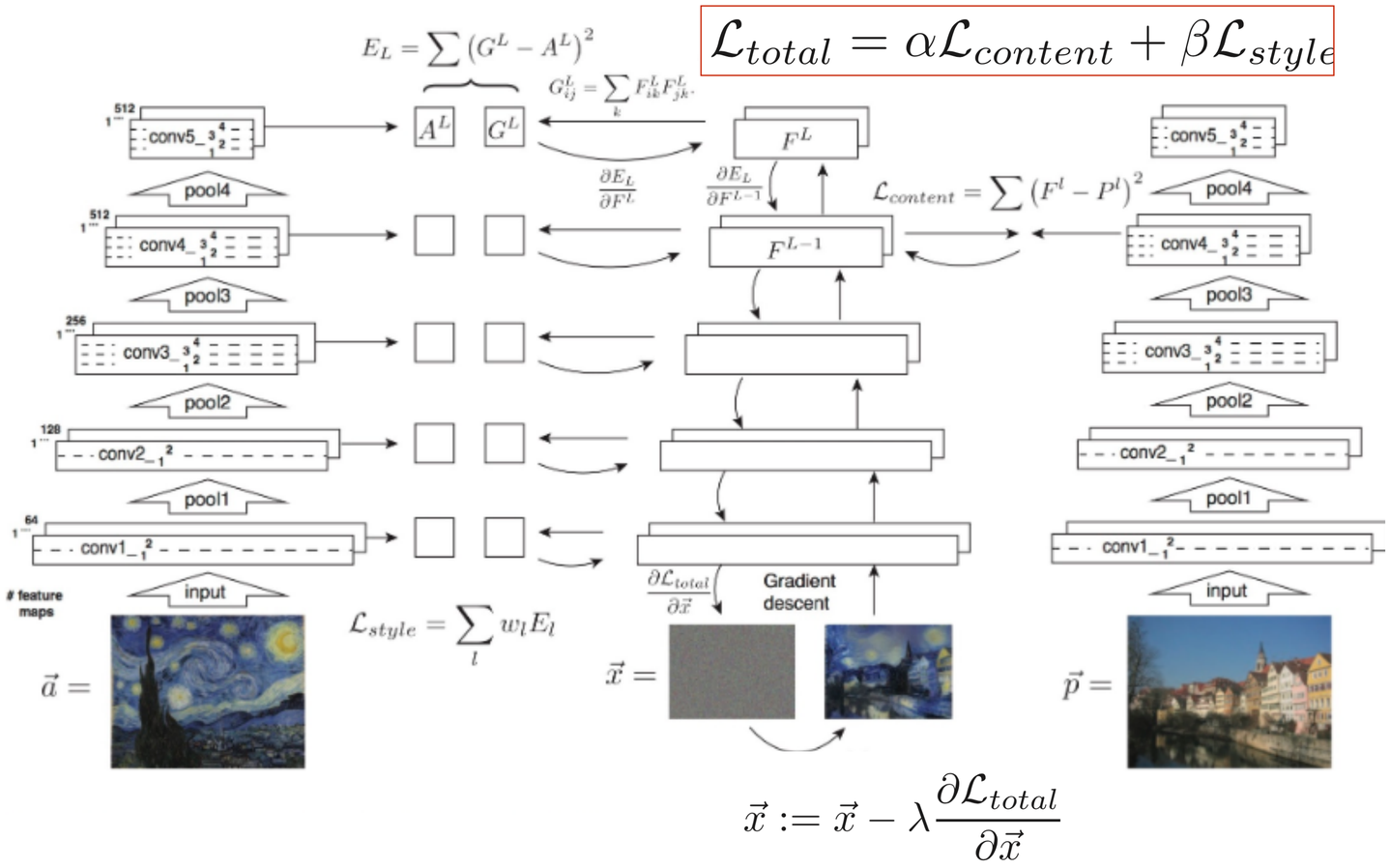
- X's content is similar with Content Input
- X's style is similar with Style Input

$\vec{x} =$

Loss Function를 어떻게 정의할 것인가?

Neural Style Transfer Framework

- 이미지와 관련하여 미리 학습된 ConvNet 모델을 이용하여 Style Transfer를 진행함 (주로 VGG-16, VGG-19 이용)
- 생성한 이미지에 대하여 **Content Loss**와 **Style Loss**를 정의하고, 둘의 합을 **Total Loss**로 정의
- 입력된 Content & Style Image에 대하여 Total Loss를 최소화하는 이미지를 반복적으로 학습하여 생성함



Content Representation: Loss Function

- l 번째 layer의 Content Loss는 아래와 같이 정의됨
- Content image의 feature map과 Input image의 feature map 사이의 SSE (Sum of Square Error)

<l-th Content Loss Definition>

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

$$\frac{\partial \mathcal{L}_{\text{content}}}{\partial F_{ij}^{l}} = \begin{cases} \left(F^{l} - P^{l} \right)_{ij} & \text{if } F_{ij}^{l} > 0 \\ 0 & \text{if } F_{ij}^{l} < 0 \end{cases}$$

$\vec{x}$: Input Image (모델이 생성하는 이미지, 학습의 대상)

$\vec{p}$: Input Content Image

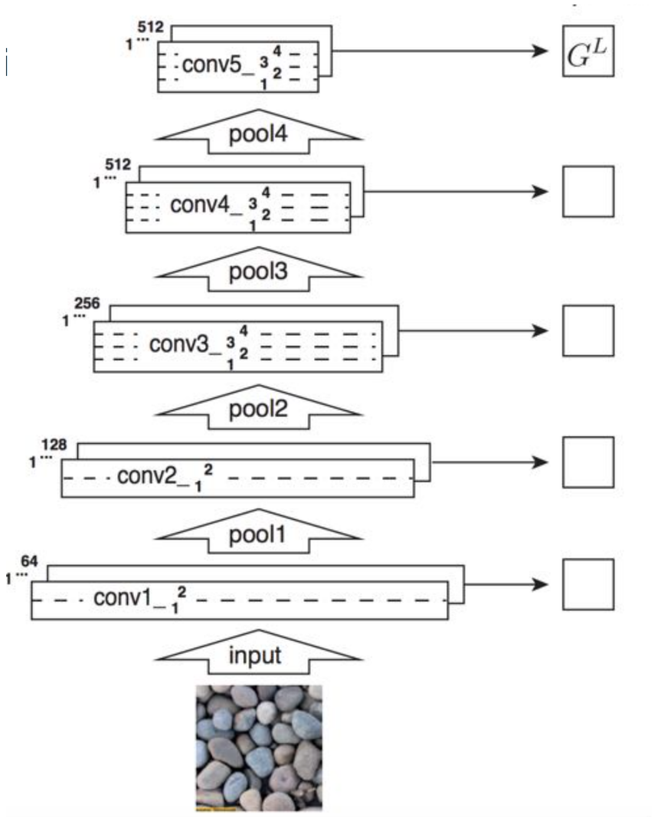
M_l : l-th layer의 feature map size (height * width)

N_l : l-th layer의 feature map 개수

F_{ij}^{l} : l-th layer에서
i-th feature map의
j-th 위치에 있는 activation 값

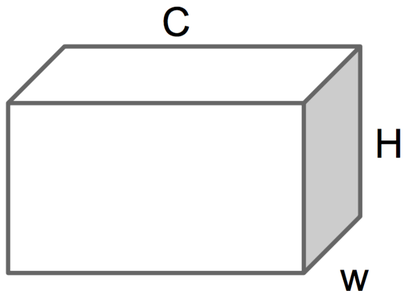
Style Representation: Gram Matrix

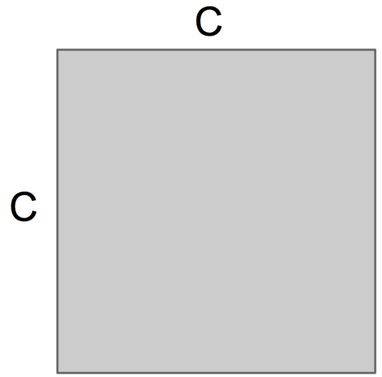
- Feature maps 사이의 Correlation을 나타내는 Gram Matrix를 통해 이미지의 Style을 표현함
- Input image와 Style image의 Gram matrix를 유사하게 만들도록 style loss를 설정하고 input image X를 학습함



$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}$$

l-th layer에서 i 번째 feature와 j 번째 feature 사이의 correlation을 의미함.
(각각의 feature map을 vector 형태로 reshape)





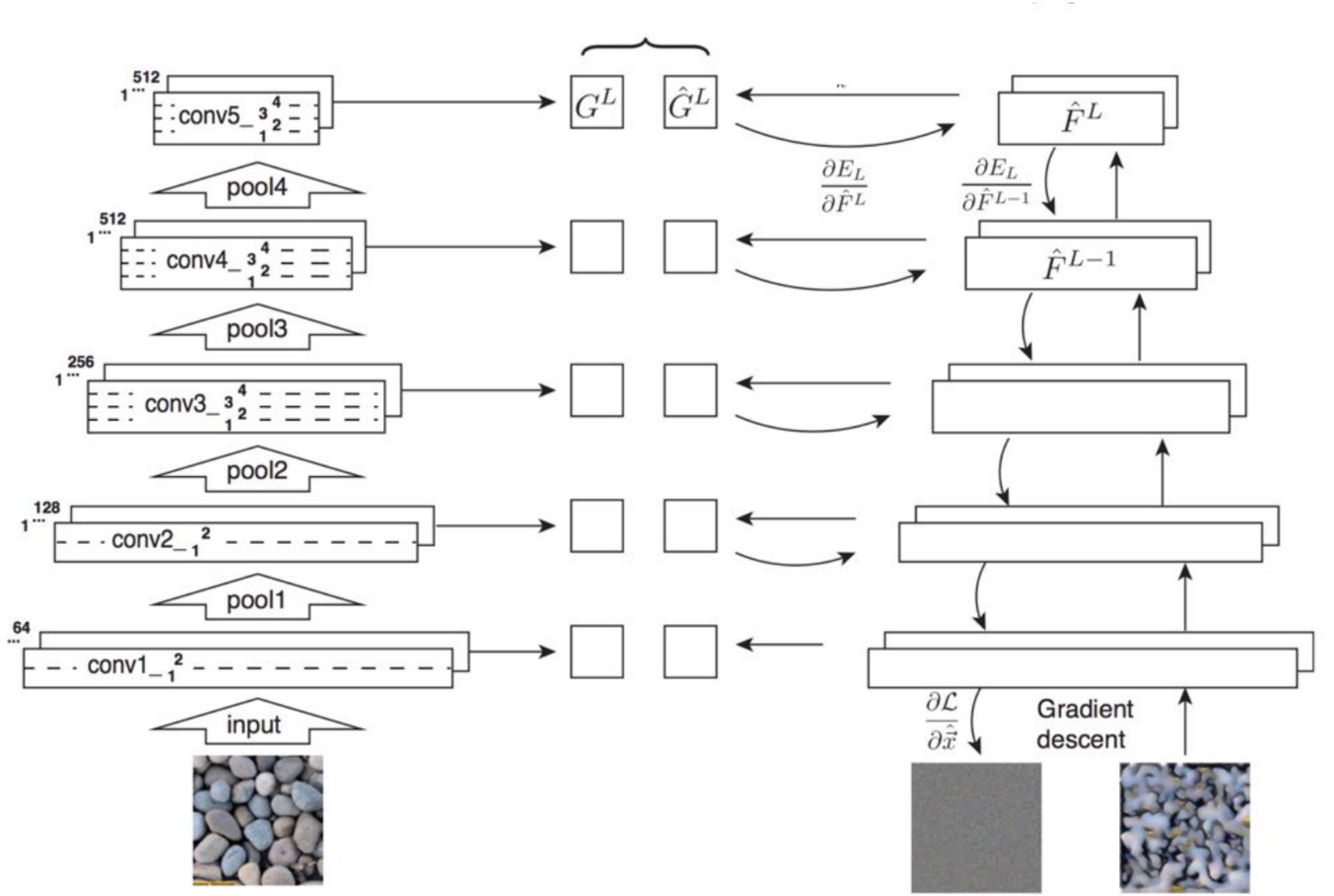
Efficient to compute; reshape features from

C x H x W to =C x HW

then compute G = FF^T

Style Representation: Loss Function

- Input image X와 Style Input Image의 Gram matrix의 차이를 Style Loss로 정의함
- Layer 별로 style loss를 계산하고, 모든 layer의 style loss 가중치 합을 전체 style loss로 정의함



<l-th Style Loss Definition>

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

<Overall Style Loss Definition>

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

Loss Function Summary

- Total loss는 Content loss와 Style loss의 가중치 합으로 정의됨
- Content와 Style 사이의 일치 정도를 weight으로 조절하며 trade-off를 진행함

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

Input X와 Content Input의
Feature map 값 차이

Input X와 Style Input의
Feature map의 Gram matrix 차이

Learning? ↓

random noise image X

$$\vec{x} := \vec{x} - \lambda \frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$$

X를 random noise로 초기화한 후,
total loss를 최소화하는 방향으로 반복적으로 X를 학습 (픽셀 단위)

trained image X

Results

- 다양한 Style Image들에 대하여 원활하게 transfer하는 것을 확인할 수 있음
- Content & Style loss 사이의 weight 변수를 조절함에 따라 content & style matching 정도 조절 가능
  (일반적으로 alpha/beta = 0.001)

More weight to content loss

More weight to style loss

The Effect of Matching Content Representation in Different Layers

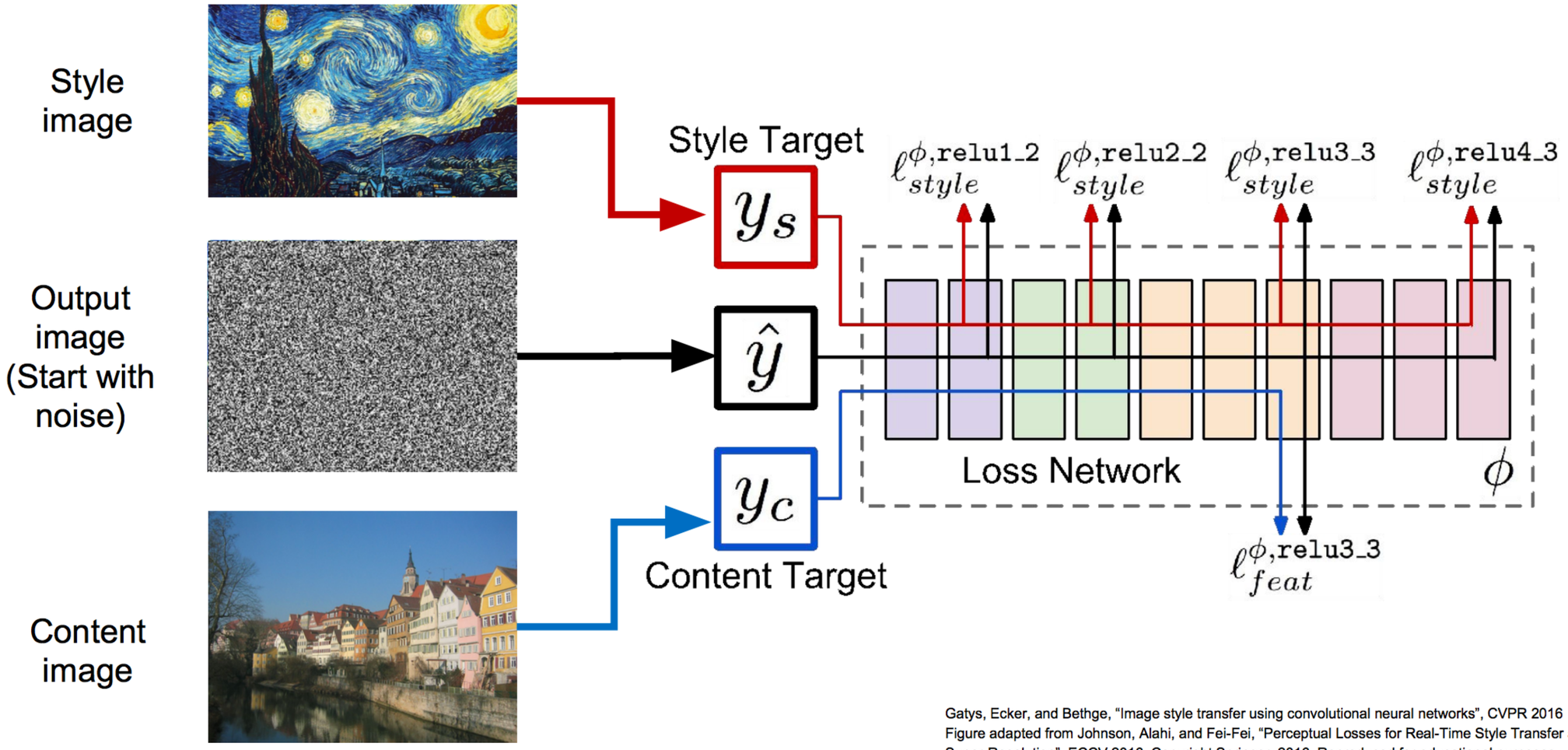
- 상위 layer의 feature map을 content representation으로 사용할 수록 왜곡 정도가 심해지는 것을 확인 가능

Photorealistic Style Transfer Result

- 해당 연구는 **Artistic style transfer**를 목적으로 만들어졌지만, **photorealistic style transfer**에도 이용할 수 있음

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}

Summary



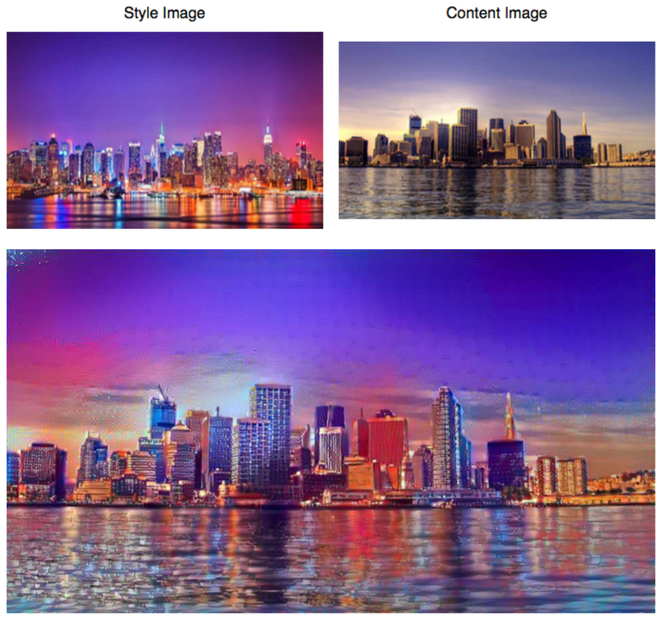
Gatys, Ecker, and Bethge, "Image style transfer using convolutional neural networks", CVPR 2016
Figure adapted from Johnson, Alahi, and Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution", ECCV 2016. Copyright Springer, 2016. Reproduced for educational purposes.

Deep Photo Style Transfer

Photographic Style Transfer

- **사진 스타일 변경 (Photographic style transfer)는 특정 사진의 내용은 유지한 채 스타일을 다르게 변경하는 것을 의미함**
  - 조도 (illumination), 시각 (time of day), 날씨 (weather) 등이 전통적으로 사진 스타일 변경의 대상이었음
- **기존 Neural style transfer를 사진에 적용할 경우, 사진이 그림 (painting) 처럼 변화하는 문제가 있음**
  - 사진이 나타내는 대상의 구조를 선명하게 유지해야함 (Structure preservation)
  - 사진 속의 대상이 나타내는 의미를 중심으로 스타일을 변경해야함 (하늘 스타일 A-> 하늘 스타일 B, 건물 스타일 A-> 건물 스타일 B 등)

<사진의 스타일 변경 후 그림이 되는 문제>

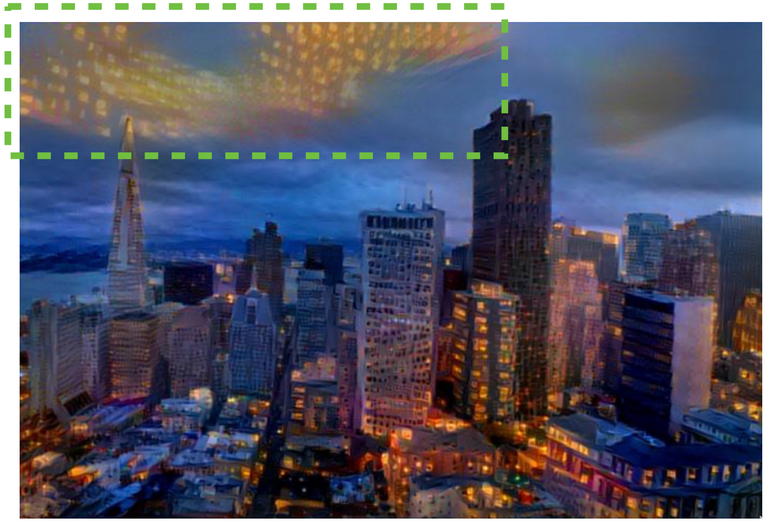


<의미(semantic)가 반영 없이 스타일이 변경되는 경우>


Input image


Reference style image

하늘에 건물의 스타일이 반영



Differences between Neural and Photo Style Transfer

❑ 전체 구조는 유지한 상태하고 **Loss Function을 변경**하여 기존 문제를 해결하고 **Photo Style Transfer** 구현

<Loss Function of Neural Style Transfer>

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

Input X와 Content Input의 Feature map 값 차이

Input X와 Style Input의 Feature map의 Gram matrix 차이

<Loss Function of Deep Photo Style Transfer>

$$\mathcal{L}_{\text{total}} = \sum_{l=1}^{L} \alpha_\ell \mathcal{L}_c^\ell + \Gamma \sum_{\ell=1}^{L} \beta_\ell \mathcal{L}_{s+}^\ell + \lambda \mathcal{L}_m$$

Input X와 Content Input의 Feature map 값 차이

(Semantic을 고려한) Input X와 Style Input의 Feature map의 Gram matrix 차이

(Photorealism Regularization) 스타일 변형 결과를 사진으로 유지하기 위한 Loss Function

Photorealism Regularization

- **Matting Laplacian (Levin et. al., 2008)을 이용한 Photorealism Regularization Term을 Loss에 추가함으로써 Photorealism을 구현함**
- **Style Transfer 이 후 생성되는 Output Image의 왜곡(distortion)에 cost를 추가하는 의미가 있음**
- **Color Space에서의 Affine Transformation을 통해 사진 안의 객체의 윤곽선이 왜곡되는 것을 방지함**
  - RGB 채널의 값들이 다양하더라도 윤곽의 표현은 RGB 채널에 관계없이 동일한 의미를 가지고 있음
  - RGB 채널의 locally affine transformation을 통하여 grayscale matte를 표현하는 방법임

$$\mathcal{L}_m = \sum_{c=1}^{3} V_c[O]^T \mathcal{M}_I V_c[O]$$

$\mathcal{M}_I$: Input Image I에 의해 유도되는 Matting Laplacian

$V_c[O]$: output 이미지의 c 채널의 결과를 벡터화

→ **Minimize?**

"Seek an image transform
that is locally affine in color space, that is, a function such that for

each output patch, there is an affine function
that maps the input RGB values onto their output counterpars."

A.Levin,D.Lischinski,andY.Weiss.Aclosed-formsolution to natural image matting.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2):228–242, 2008. 2, 4

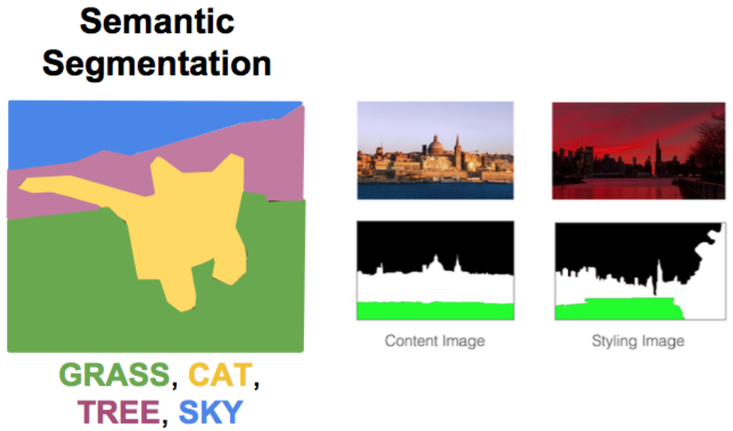
Augmented Style Loss with Semantic Segmentation

- **사진 내 객체들의 의미(semantic)을 반영하여 스타일 변경을 진행하는 것을 의미 (spill over 현상을 방지)**

  ex) Content Image의 하늘 -> Style Image의 하늘, Content Image의 건물 -> Style Image의 건물

- **Semantic Segmentation 방법을 통해 사진 내 Segmentation label을 구성함**
- **Segmentation 결과를 Mask로 활용하여 feature maps를 전처리한 후, Gram Matrix를 구하는 방식으로 변경**

<Semantic Segmentation 예시>



# of Segmentation Labels

Gram Matrix of Masked Features

$$\mathcal{L}_{s+}^{\ell} = \sum_{c=1}^{C} \frac{1}{2N_{\ell,c}^2} \sum_{ij} (G_{\ell,c}[O] - G_{\ell,c}[S])_{ij}^2$$

$$F_{\ell,c}[O] = F_{\ell}[O] M_{\ell,c}[I] \qquad F_{\ell,c}[S] = F_{\ell}[S] M_{\ell,c}[S]$$

"같은 Segmentation Label"끼리 Style Loss를 계산!

Semantic Segmentation Issues

- **Semantic Segmentation의 결과를 사용하는데 있어 몇 가지 이슈 사항이 존재함**

  Q) VGG 모델의 경우 layer가 깊어질수록 feature map의 크기가 감소하지 않나? (pooling layer의 영향)

  A) Linearly down sampling으로 mask 크기를 일정하게 줄임

  Q) Semantic Segmentation은 어떤 모델로 진행하나?

  A) DilatedNet (Chen et. al., 2016)을 이용하여 150개의 카테고리로 Segmentation 결과를 분류함

  Q) Content Image과 Style (reference) Image 사이의 Semantic label이 일치하지 않으면 어떻게 처리하나?

  A) Deep Photo Style Transfer에서 중요한 문제 중 하나! 짝이 맞지 않는 Semantic을 “Orphan Semantic Labels”이라고 지칭하는데, 이 현상을 최소화하기 위해 Reference Style Image의 Semantic Segmentation 결과만 Mask로 사용함.
  또한 성능 향상을 위해 150개의 모든 카테고리를 다 사용하지 않고 유사한 카테고리는 같은 Label로 통합함!
  (“lake”이나 “sea”나 물이라는 속성은 유사하므로 같은 카테고리로 통합)
  하지만 두 이미지 사이의 semantic label 사이의 관련성이 매우 낮다면, 좋지 않은 성능을 보여주는 것이 한계점!

L.-C.Chen,G.Papandreou,I.Kokkinos,K.Murphy,andA.L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016. 2, 4, 5

Review of Loss Function

- **결국 Style Loss를 Semantic mask를 이용하여 변경하고, Photorealism Regularization Term을 추가함:**

  1) 사진의 특성을 유지하여 Structure preservation을 지키고

  2) Semantic accuracy와 transfer faithfulness (얼마나 스타일을 잘 반영하는지)를 높임!

- **Random Noise Image에서 바로 결과 Image를 생성하지 않고 2-step optimization 방법 사용**

  - 1차적으로 기존 Neural Style Transfer 결과 이미지를 생성함

  - 기존 모델의 결과를 초기 Input X로 재설정한 후 Deep Photo Style Transfer 모델 학습

<Loss Function of Deep Photo Style Transfer>

$$\mathcal{L}_{\text{total}} = \sum_{l=1}^{L} \alpha_\ell \mathcal{L}_c^\ell + \Gamma \sum_{\ell=1}^{L} \beta_\ell \mathcal{L}_{s+}^\ell + \lambda \mathcal{L}_m$$

Input X와 Content Input의
Feature map 값 차이

(Semantic을 고려한)
Input X와 Style Input의
Feature map의 Gram matrix 차이

(Photorealism Regularization)
스타일 변형 결과를 사진으로 유지하기 위한
Loss Function

Results - Photorealism Regularization Parameter

- **Photorealism Regularization Term의 Weight을 증가시킬수록 사실적인 사진 이미지가 생성되는 것을 확인**

<Loss Function of Deep Photo Style Transfer>

$$\mathcal{L}_{\text{total}} = \sum_{l=1}^{L} \alpha_\ell \mathcal{L}_c^\ell + \Gamma \sum_{\ell=1}^{L} \beta_\ell \mathcal{L}_{s+}^\ell + \lambda \mathcal{L}_m$$

Input X와 Content Input의
Feature map 값 차이

(Semantic을 고려한)
Input X와 Style Input의
Feature map의 Gram matrix 차이

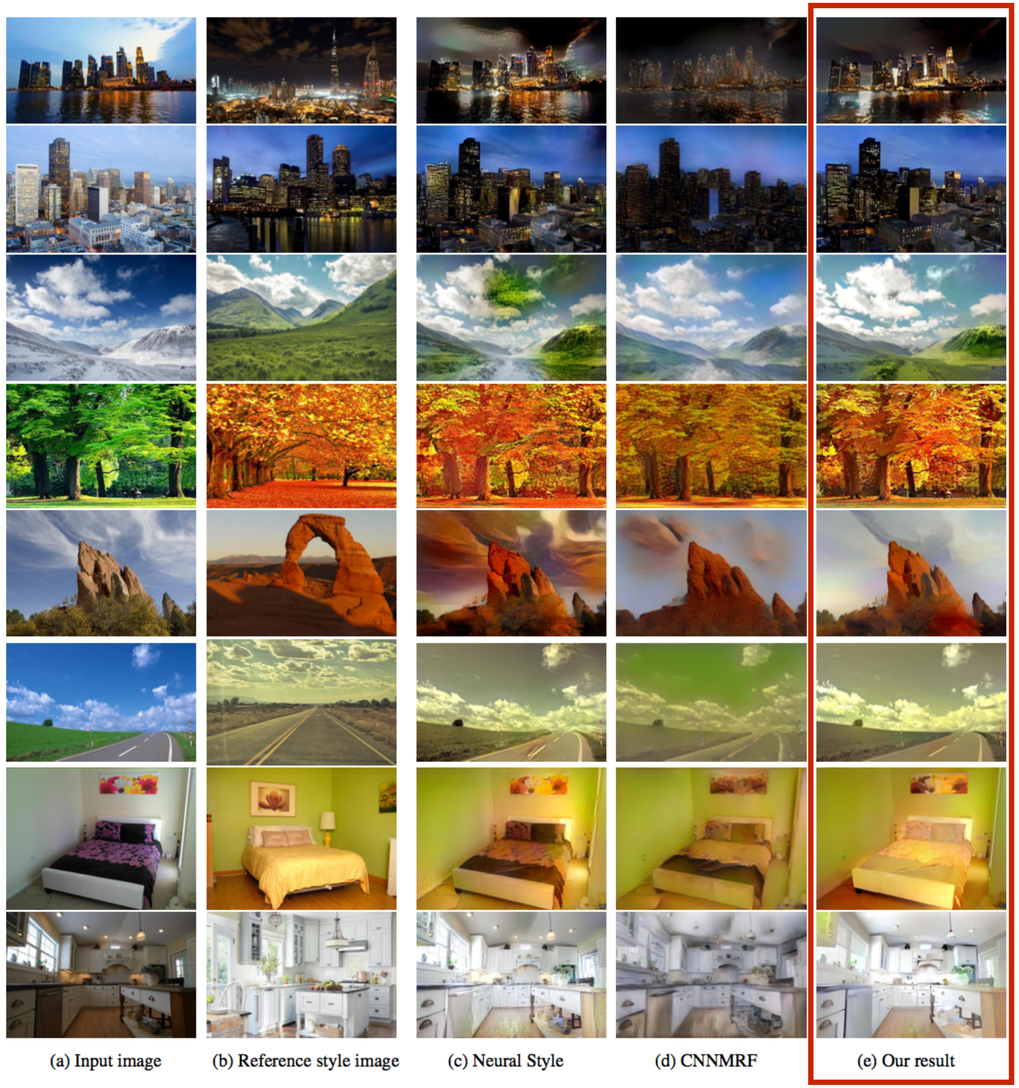
(Photorealism Regularization)
스타일 변형 결과를 사진으로 유지하기 위한
Loss Function

(a) Input and Style (b) $\lambda = 1$ (c) $\lambda = 10^2$ (d) $\lambda = 10^4$, our result (e) $\lambda = 10^6$ (f) $\lambda = 10^8$

Figure 3: Transferring the dramatic appearance of the reference style image ((a)-inset), onto an ordinary flat shot in (a) is challenging. We produce results using our method with different λ parameters. A too small λ value cannot prevent distortions, and thus the results have a non-photorealistic look in (b,c). Conversely, a too large λ value suppresses the style to be transferred yielding a half-transferred look in (e,f). We found the best parameter $\lambda = 10^4$ to be the sweet spot to produce our result (d) and all the other results in this paper.

Results - Structure Preservation

❑ 다른 Neural Style Transfer 모델들과 결과 비교 (선명도)

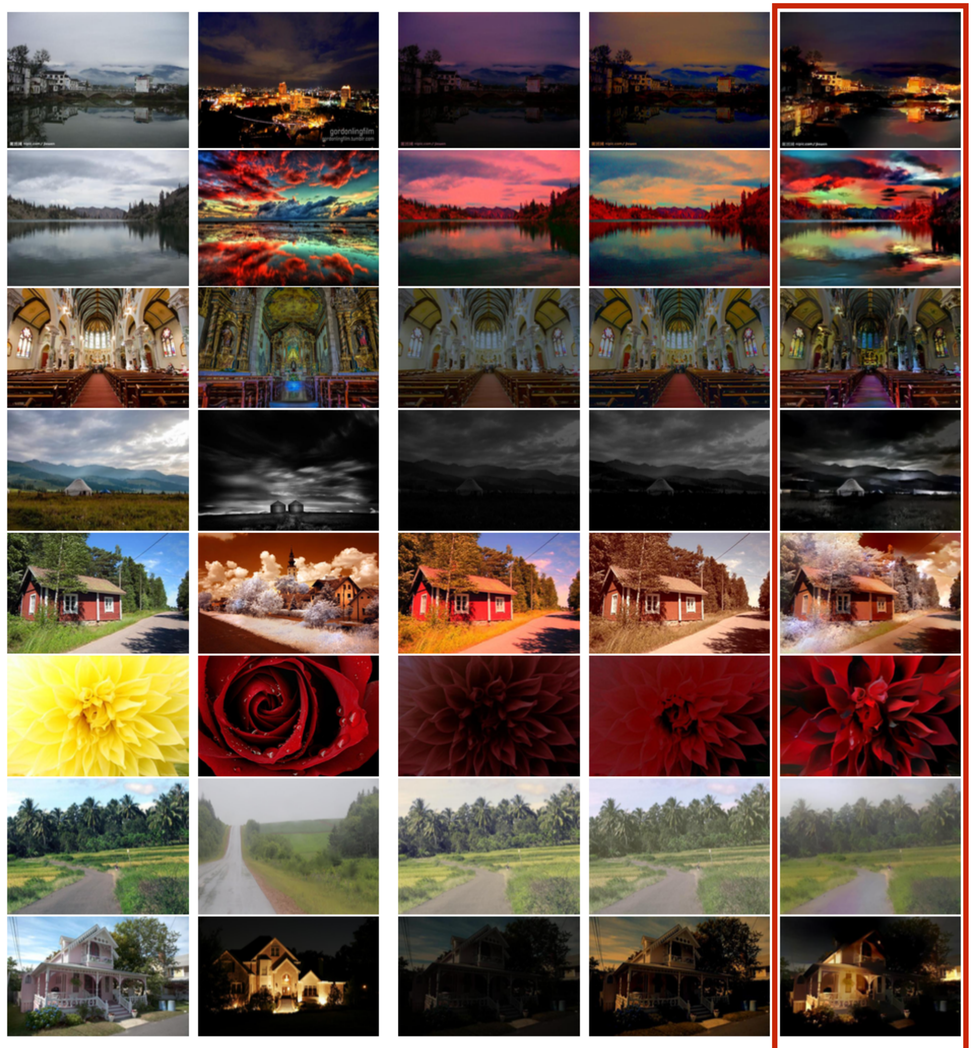


(a) Input image (b) Reference style image (c) Neural Style (d) CNNMRF (e) Our result

Figure 4: Comparison of our method against Neural Style and CNNMRF. Both Neural Style and CNNMRF produce strong distortions in their synthesized images. Neural Style also entirely ignores the semantic context for style transfer. CNNMRF tends to ignore most of the texture in the reference style image since it uses nearest neighbor search. Our approach is free of distortions and matches texture semantically.

Results: Transfer Faithfulness

- Content를 왜곡하지 않고, 색상만 변경하는 Global style transfer 방법들과 결과 비교



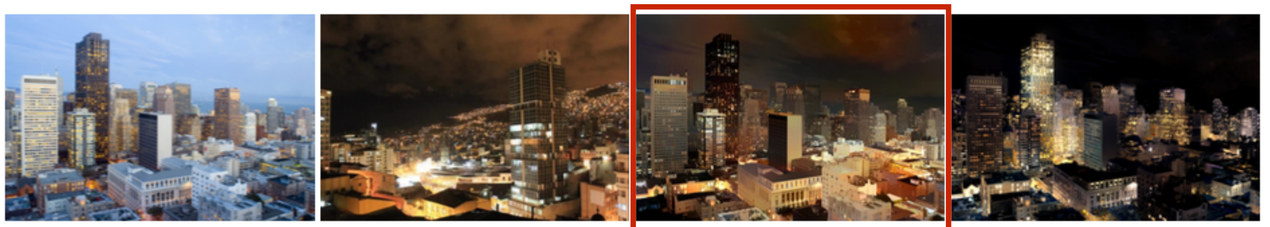
(a) Input image (b) Reference style image (c) Reinhard et al. [12] (d) Pitié et al. [11] (e) Our result

Figure 5: Comparison of our method against Reinhard et al. [12] and Pitié [11]. Our method provides more flexibility in transferring spatially-variant color changes, yielding better results than previous techniques.

Results: Time of Day and Manual Segmentation

- **기존 시각 (Time of Day)을 변경하는 모델들의 결과와 비교**

  (기존 방법은 특정 사진의 타임랩스 비디오를 촬영하여 시간의 변화를 도출함)

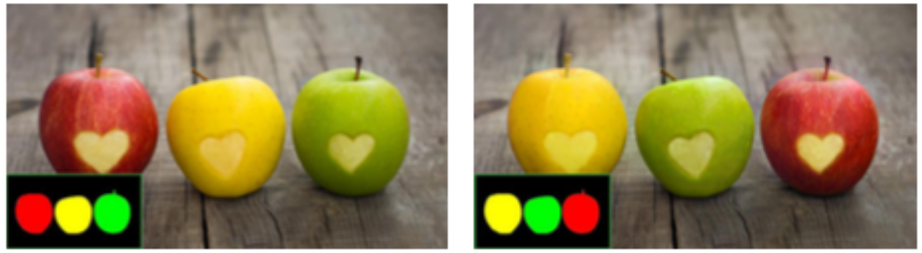


(a) Input image (b) Reference style image (c) Our result (d) Shih et al. [15]

Figure 6: Our method and the techique of Shih et al. [15] generate visually satisfying results. However, our algorithm requires a single style image instead of a full time-lapse video, and it can handle other scenarios in addition to time-of-day hallucination.

- **Semantic Segmentation을 직접 manual하게 설정할 경우, 원하는 객체에 원하는 Style을 직접 입힐 수 있음**

(a) Input (b) Reference style (c) Our result



(d) Input (e) Our result

Wrong Results

- Content Input과 Reference Style 사진 사이 Semantic label의 관련이 매우 낮을 경우, 좋지 않은 결과를 보여줌

(a) Input (b) Reference style (c) Our result

Figure 8: Failure cases due to extreme mismatch.

Evaluation of Transfer Model

- **Deep Photo Style Transfer 모델의 결과와 기존 방법들의 결과를 비교하기 위하여 35-40 명에게 설문조사 진행**
- **평가 기준은 아래 두 가지 항목이며, 4 Likert Scale로 항목 평가 진행**

  1) Photorealism: 얼마나 결과물이 사실적인가

  2) Style Faithfulness Preference: 결과물이 Style을 잘 반영하고, 충분히 선호되는가

Photorealism 항목은
기존 NN 기반 Style Transfer 방법보다
월등하게 높은 평가를 받았으며,

절대적인 평가
역시 **Photorealistic에 해당하는 점수**를 받음

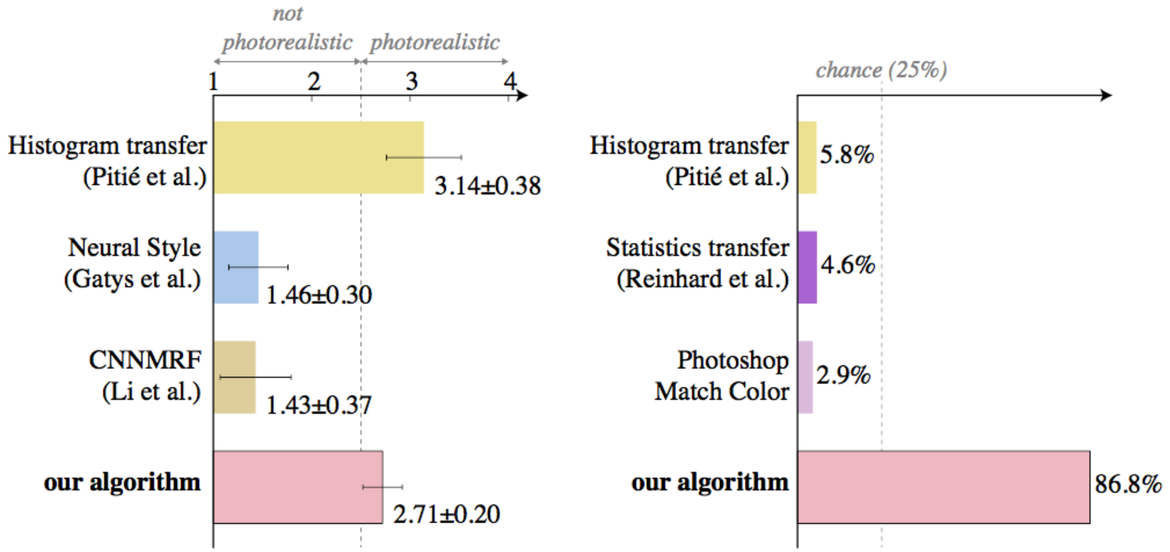


Figure 9: User study results confirming that our algorithm produces photorealistic and faithful results.

Style faithfulness preference 항목은
기존 통계적인 방법론과
Photoshop Match Color 기능보다
월등하게 높은 평가를 받음

Discussion

- **아래 항목들에 대한 추가 연구, 발전, 또는 검토 등이 필요할 것으로 보임**

  - Titan X를 사용하더라도 하나의 이미지를 처리하는데 5분 정도의 시간이 걸림. Style Transfer를 전문적인 이미지 처리 분야에서도 사용할 수 있지만, 재미나 놀이를 위해서도 많이 사용될텐데 처리 시간이 너무 긴 것은 아닌지?

  - Semantic Segmentation 결과에 매우 큰 영향을 받음. Label 역시 일반적인 카테고리만 존재. 특수한 Label을 가진 Semantic은 처리하지 못함. 결과적으로 Manual Segmentation을 이용하는 것의 성능이 가장 좋은 것으로 확인됨
  (개선의 여지가 있어 보임)

  - Photorealism Regularization Term에 대한 구체적인 근거를 조사한 후 자료 보완 필요
  (현재 사용하는 Matting Laplacian은 Input에 대하여 deterministic한 성질을 가지고 있음. 이 부분을 변경할 수 있을까? 변경했을 때 어떤 장점이 있을까?)

  - Gram Matrix가 Style을 나타내는 최선의 방법인가?

Thank You :)