Universal Style Transfer via Feature Transforms

Yijun Li
UC Merced
yli62@ucmerced.edu

Chen Fang
Adobe Research
cfang@adobe.com

Jimei Yang
Adobe Research
jimyang@adobe.com

Zhaowen Wang
Adobe Research
zhawang@adobe.com

Xin Lu
Adobe Research
xinl@adobe.com

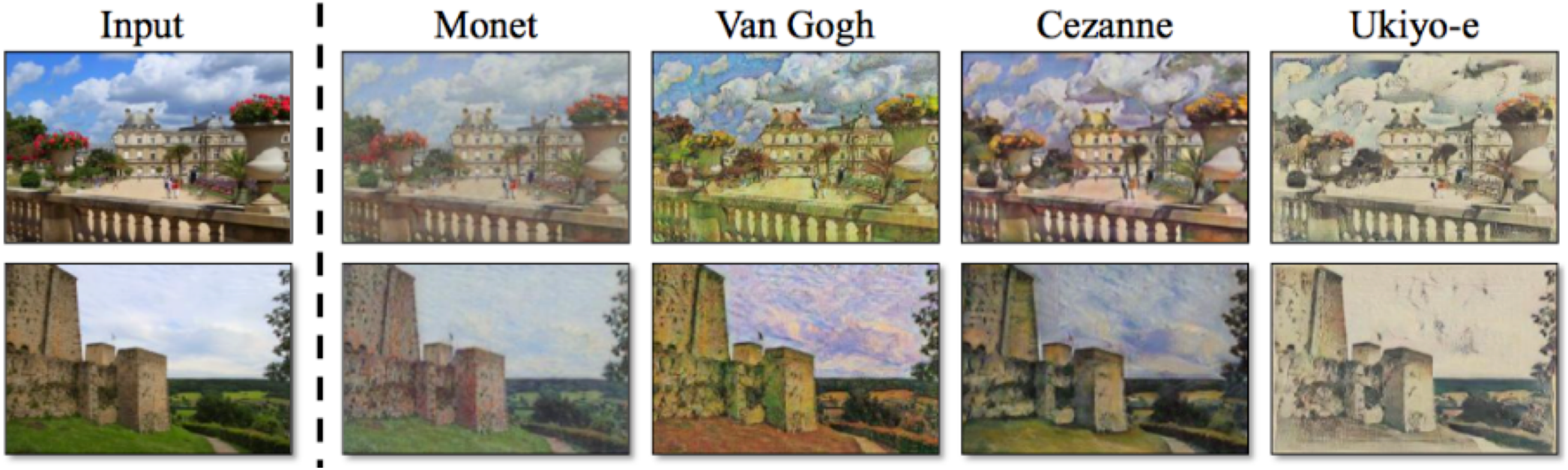
Ming-Hsuan Yang
UC Merced, NVIDIA Research
mhyang@ucmerced.edu

Presented by : Kangyeol Kim

DAVIAN Lab, Korea University

Style Transfer

WHAT?



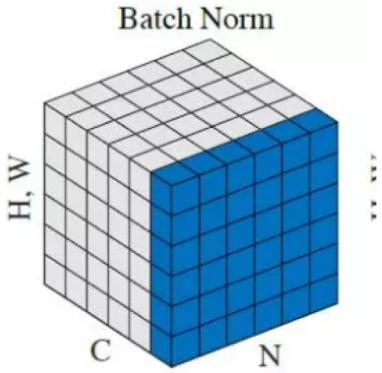
- Basically, Style Transfer aims to transfer the style of content image(i.e. Input) to target style(Monet, Van Gogh etc..) while preserving the core object of content image.

- There are several works before this paper such as AdaIN or so on.

RECAP Basis of AdaIN(Adaptive Instance Normalization); Why Instance Norm?

It has been known that the convolutional feature statistics of a DNN can capture the style of an image [16, 30, 33]. While Gatys *et al.* [16] use the second-order statistics as their optimization objective, Li *et al.* [33] recently showed that matching many other statistics, including channel-wise mean and variance, are also effective for style transfer. Mo-

Since BN normalizes the feature statistics of a batch of samples instead of a single sample, it can be intuitively understood as normalizing a batch of samples to be centered around a single style. Each single sample, however,



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

Computing **statistics over batch** means **nothing to styles** of images !!!

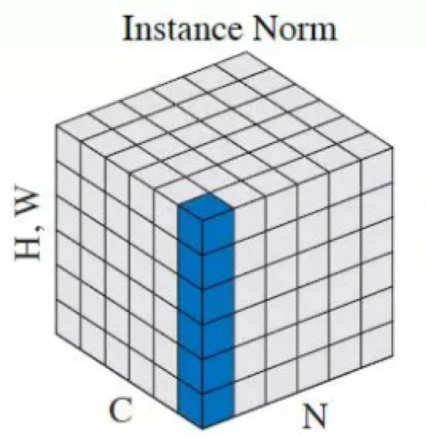
$$\mu_c(x) = \frac{1}{NHW}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{nchw} \qquad (2)$$

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

RECAP Basis of AdaIN(Adaptive Instance Normalization); Why Instance Norm?

lenges for training. On the other hand, IN can normalize the style of each individual sample to the target style. Training is facilitated because the rest of the network can focus on content manipulation while discarding the original style information. The reason behind the success of CIN also be-

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



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

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

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

Introduction (1/3)

midst of feed-forward passes. In each intermediate layer, our main goal is to transform the extracted content features such that they exhibit the same statistical characteristics as the style features of the same layer and we found that the classic signal *whitening* and *coloring* transforms (WCTs) on those features are able to achieve this goal in an almost effortless manner.

In this work, we first employ the VGG-19 network [26] as the feature extractor (encoder), and train a symmetric decoder to invert the VGG-19 features to the original image, which is essentially the image reconstruction task (Figure 1(a)). Once trained, both the encoder and the decoder are fixed through all the experiments. To perform style transfer, we apply WCT to one layer of content features such that its covariance matrix matches that of style features, as shown in Figure 1(b). The transformed features are then fed forward into the downstream decoder layers to obtain the stylized image. In addition to this single-level stylization, we further develop a multi-level stylization pipeline, as depicted in Figure 1(c), where we apply WCT sequentially to multiple feature layers. The multi-level algorithm generates stylized images with greater visual quality, which are comparable or even better with much less computational costs. We also introduce a control parameter that defines the degree of style transfer so that the users can choose the balance between stylization and content preservation. The entire procedure of our algorithm only requires learning the image reconstruction decoder with *no* style images involved. So when given a new style, we simply need to extract its feature covariance matrices and apply them to the content features via WCT. Note that this learning-free scheme is

Introduction (2/3)

The main contributions of this work are summarized as follows:

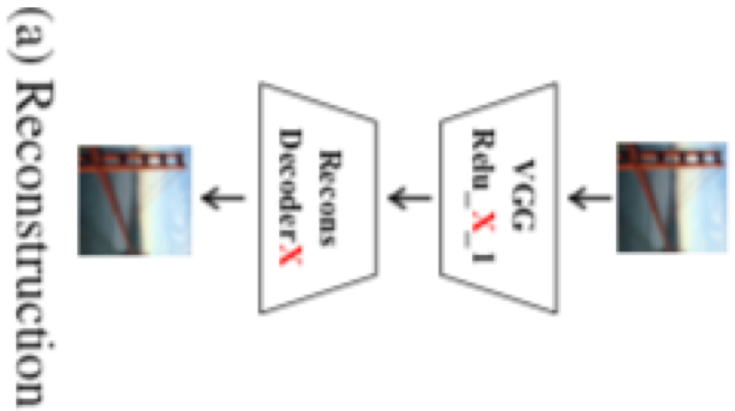
- We propose to use feature transforms, i.e., whitening and coloring, to directly match content feature statistics to those of a style image in the deep feature space.
- We couple the feature transforms with a pre-trained general encoder-decoder network, such that the transferring process can be implemented by simple feed-forward operations.
- We demonstrate the effectiveness of our method for universal style transfer with high-quality visual results, and also show its application to universal texture synthesis.

Different from the existing methods, our approach performs style transfer efficiently in a feed-forward manner while achieving generalization and visual quality on arbitrary styles. Our approach is closely related to [15], where content feature in a particular (higher) layer is adaptively instance normalized by the mean and variance of style feature. This step can be viewed as a sub-optimal approximation of the WCT operation, thereby leading to less effective results on both training and unseen styles. Moreover, our encoder-decoder network is trained solely based on image reconstruction, while [15] requires learning such a module particularly for stylization task. We evaluate the proposed algorithm with existing approaches extensively on both style transfer and texture synthesis tasks and present in-depth analysis.

Proposed methods (1/5)

$$L = \|I_o - I_i\|_2^2 + \lambda\|\Phi(I_o) - \Phi(I_i)\|_2^2 , \quad (1)$$

where I_i, I_o are the input image and reconstruction output, and Φ is the VGG encoder that extracts the Relu_X_1 features. In addition, λ is the weight to balance the two losses. After training, the decoder is fixed (i.e., will not be fine-tuned) and used as a feature inverter.



- **5 decoders are trained while we fix the encoders' parameters**
- **VGG_Relu_X_L(L=1,2,3,4,5) is used for network structure**
- **In the code, there are no NORMALIZATION layers in this architectures**

Proposed methods (2/5)

- Let I_c, f_c, I_s, f_s are content image, content feature, style image, style feature respectively, where $f_c, \in R^{C \times H_c W_c} f_s \in R^{C \times H_s W_s}$. First, make f_c, f_s zero-centered feature maps

Whitening transform : Removing style from content image.

- Recall matrix A is symmetric, A can be decomposed as QDQ^T, where Q is a orthogonal matrix whose columns are eigenvectors of A, D is a diagonal matrix whose entries are eigenvalues

- In this case, $A = f_c f_c^T$.(Note that it is a covariance matrix) Then, $f_c f_c^T = E_c D_c E_c^T$. Our goal is to remove style(i.e. normalizing feature statistics like AdaIN). Thus, convert f_c to $\widehat{f_c}$ where $\widehat{f_c}\widehat{f_c^T} = I \equiv \widehat{f_c} = E_c D_c^{-\frac{1}{2}} E_c^T f_c$. The authors can successfully remove style while preserving global content of images

Coloring transform : Adapt style from style image to $\widehat{f_c}$.

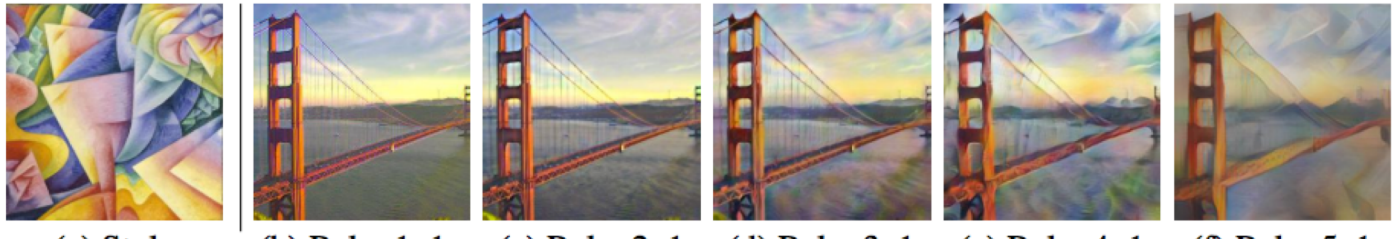
- We can align statistics of $\widehat{f_c}$ with $\widehat{f_s}$ which means painting style to $\widehat{f_c}$ by aligning statistics between them.
- In this case, $A = f_s f_s^T$. Then, $f_s f_s^T = E_s D_s E_s^T$. Our goal is to paint style by adjusting correlation Thus, convert $\widehat{f_c}$ to $\widehat{f_{cs}}$ where $\widehat{f_{cs}}\,\widehat{f_{cs}^T} = f_s f_s^T \equiv \widehat{f_{cs}} = E_s D_s^{\frac{1}{2}} E_s^T \widehat{f_c}$.
- Recenter $\widehat{f_{cs}}$ by adding mean of $\widehat{f_s}$. Now $\widehat{f_{cs}}$ gets mean and covariance matrix as $f_s f_s^T$
- Expressions:
  - $\widehat{f_{cs}}\,\widehat{f_{cs}^T} = E_s D_s^{\frac{1}{2}} E_s^T \widehat{f_c} \times \widehat{f_c^T} E_s D_s^{\frac{1}{2}} E_s^T \quad By,\ \widehat{f_c} \times \widehat{f_c^T} = I$
  - $\widehat{f_{cs}}\,\widehat{f_{cs}^T} = E_s D_s^{\frac{1}{2}} E_s^T E_s D_s^{\frac{1}{2}} E_s^T \qquad By, E_s^T E_s = I(Orthonormal\ matrix)$
  - $\widehat{f_{cs}}\,\widehat{f_{cs}^T} = E_s D_s^{\frac{1}{2}} D_s^{\frac{1}{2}} E_s^T = \mathrm{f_s f_s^T}$

Proposed methods (4/5)

Adjust value α to control the degree of stylization

$$\widehat{f_{cs}} = \alpha\widehat{f_{cs}} + (1-\alpha)f_c$$

Multi-level coarse-to-fine to get better results



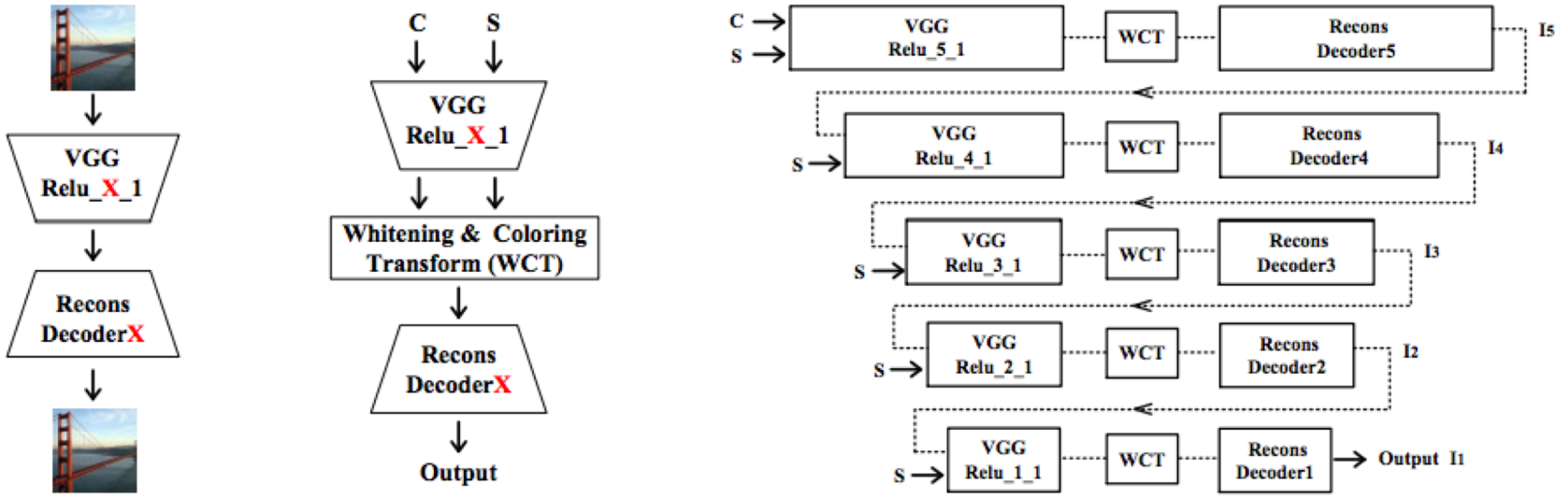
(a) Style (b) Relu_1_1 (c) Relu_2_1 (d) Relu_3_1 (e) Relu_4_1 (f) Relu_5_1

Figure 4: Single-level stylization using different VGG features. The content image is from Figure 2.

(a) I_5 (b) I_4 (c) I_1 (d) Fine-to-coarse

Proposed methods (5/5)

Architecture

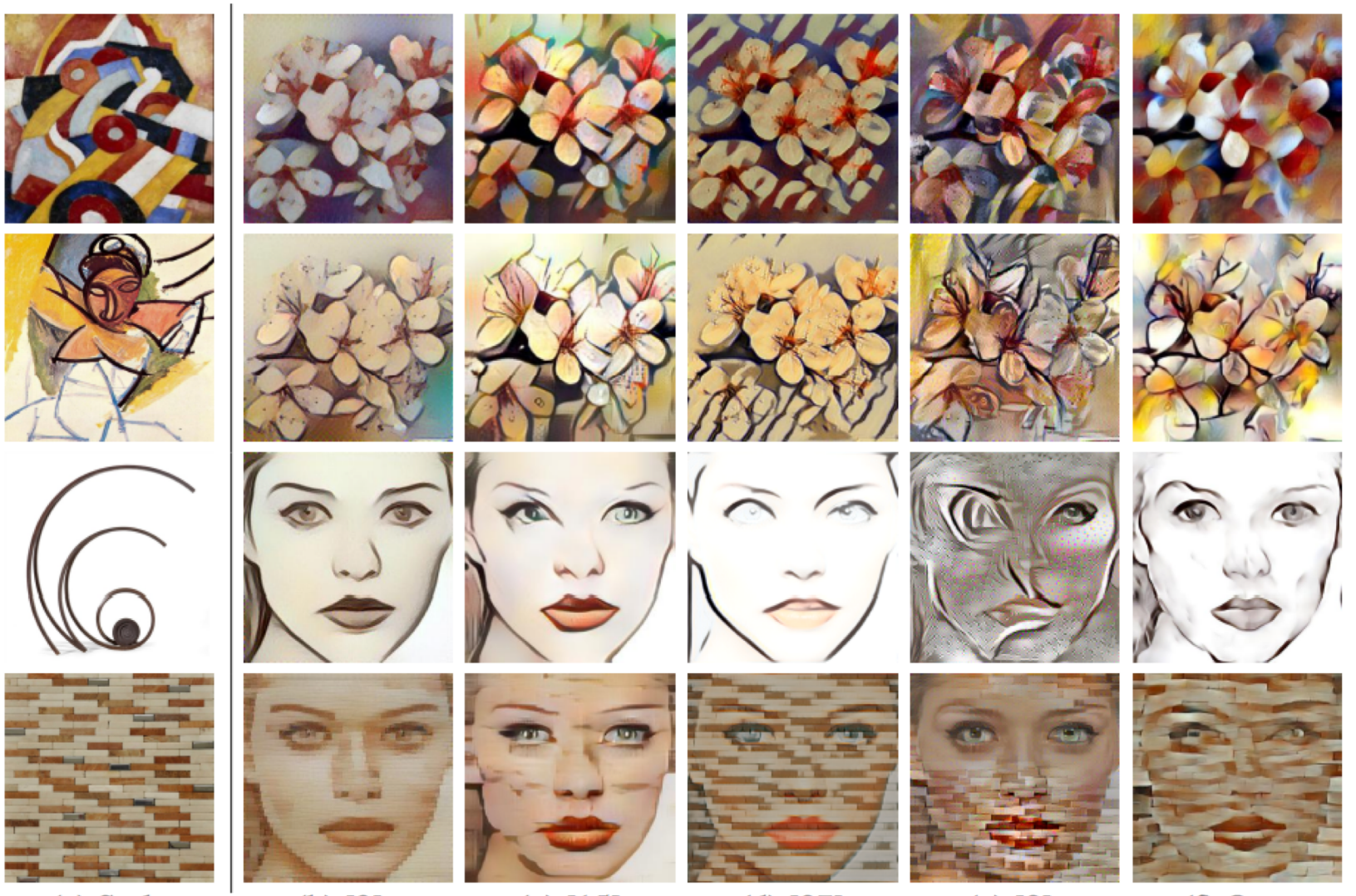


(a) Reconstruction (b) Single-level stylization (c) Multi-level stylization

Figure 1: Universal style transfer pipeline. (a) We first pre-train five decoder networks DecoderX (X=1,2,...,5) through image reconstruction to invert different levels of VGG features. (b) With both VGG and DecoderX *fixed*, and given the content image C and style image S, our method performs the style transfer through whitening and coloring transforms. (c) We extend single-level to multi-level stylization in order to match the statistics of the style at all levels. The result obtained by matching higher level statistics of the style is treated as the new content to continue to match lower-level information of the style.

Experiments (1/3)

Style Transfer.



(a) Style (b) [3] (c) [15] AdaIN (d) [27] (e) [9] (f) Ours

Experiments (2/3)

Quantitative results

Table 2: Quantitative comparisons between different stylization methods in terms of the covariance matrix difference (L_s), user preference and run-time, tested on images of size 256×256 and a 12GB TITAN X.

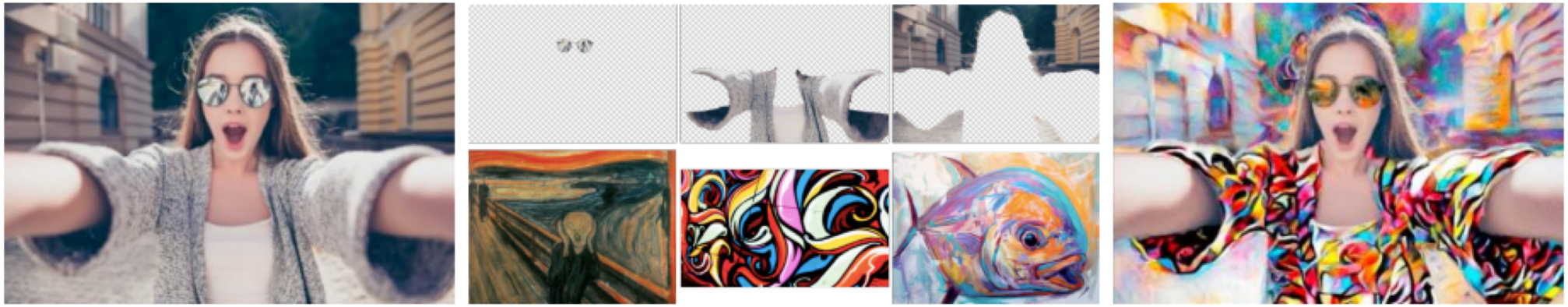
Style Image vs Styled Image

	Chen et al. [3]	Huang et al. [15]	TNet [27]	Gatys et al. [9]	Ours
$\log(L_s)$	7.4	7.0	6.8	6.7	**6.3**
Preference/%	15.7	24.9	12.7	16.4	**30.3**
Time/sec	2.1	0.20	0.18	21.2	**0.83**

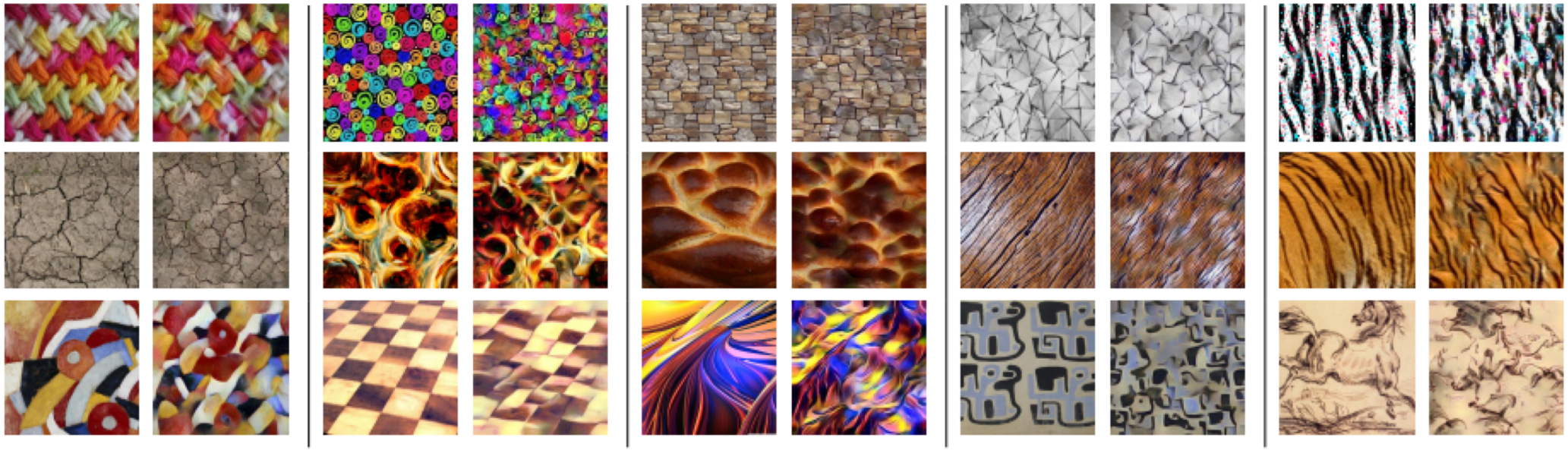
Figure 7: Controlling the stylization on the scale and weight.

Experiments (3/3)

Other results



(a) Content (b) Different masks and styles (c) Our result



+ **Combination of texture synthesis by convex combination of two Texture synthesis**

Image-to-Image Translation via Group-wise Deep Whitening-and-Coloring Transformation

Wonwoong Cho[1] Sungha Choi[1,2] David Keetae Park[1] Inkyu Shin[3] Jaegul Choo[1]

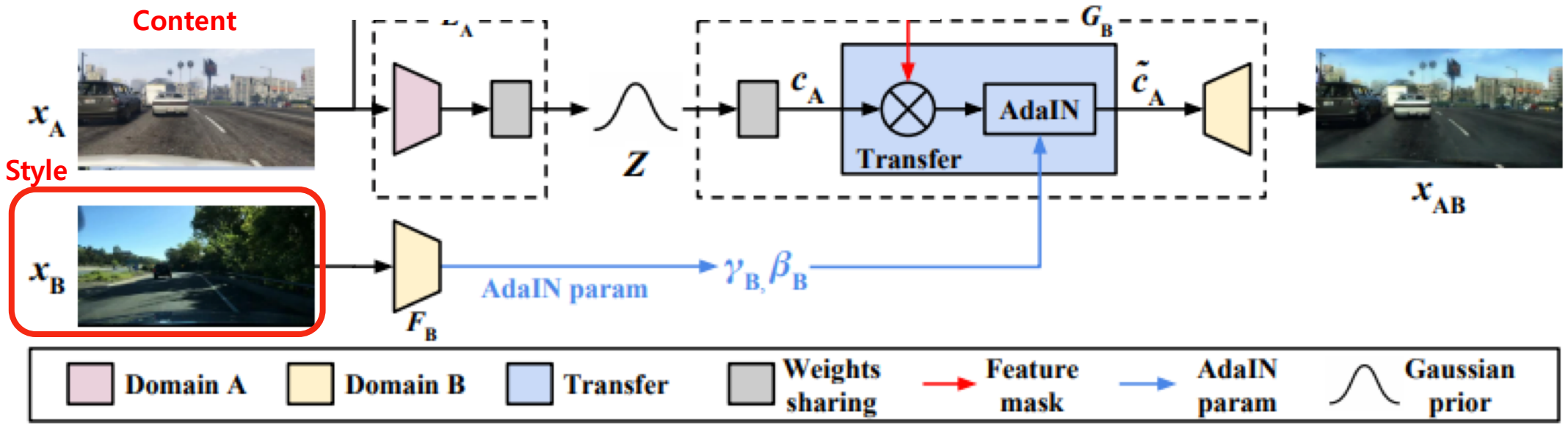
[1]Korea University [2]LG Electronics [3]Hanyang University

Presented by : Kangyeol Kim

DAVIAN Lab, Korea University

Background

Toward Various outputs in image-to-image translation



- To tackle unimodal problem in image-to-image translation as CycleGAN, DRIT and MUNIT propose new architecture to generate **multi-modal outputs from a single input image**
- We can use a **random noise** from gaussian distribution or an **exemplar image** as depicted above.

Motivation

to produce the final output. DRIT concatenates the encoded content and style feature vectors, while MUNIT exploits the adaptive instance normalization (AdaIN), a method first introduced in the context of style transfer. AdaIN matches two channel-wise statistics, the mean and variance, of the encoded content feature with the style feature, which is proven to perform well in image translation.

However, we hypothesize that matching only these two statistics may not reflect the target style well enough, ending up with the sub-optimal quality of image outputs on numerous occasions, as we confirm through our experiments in

Our model is mainly motivated by whitening-and-coloring transformation (WCT) [23], which utilizes the

The problem when applying WCT in image translation is that its time complexity is as expensive as $O(n^3)$ where n is the number of channels of a given activation map. Furthermore, computing the backpropagation with respect to singular value decomposition involved in WCT is non-trivial [30, 15]. To address these issues, we propose a novel deep whitening-and-coloring transformation that flexibly approximates the existing WCT based on deep neural networks. We further extend our method into group-wise deep whitening-and-coloring transformation (GDWCT), which

Proposed methods (1/6)

Overall architecture

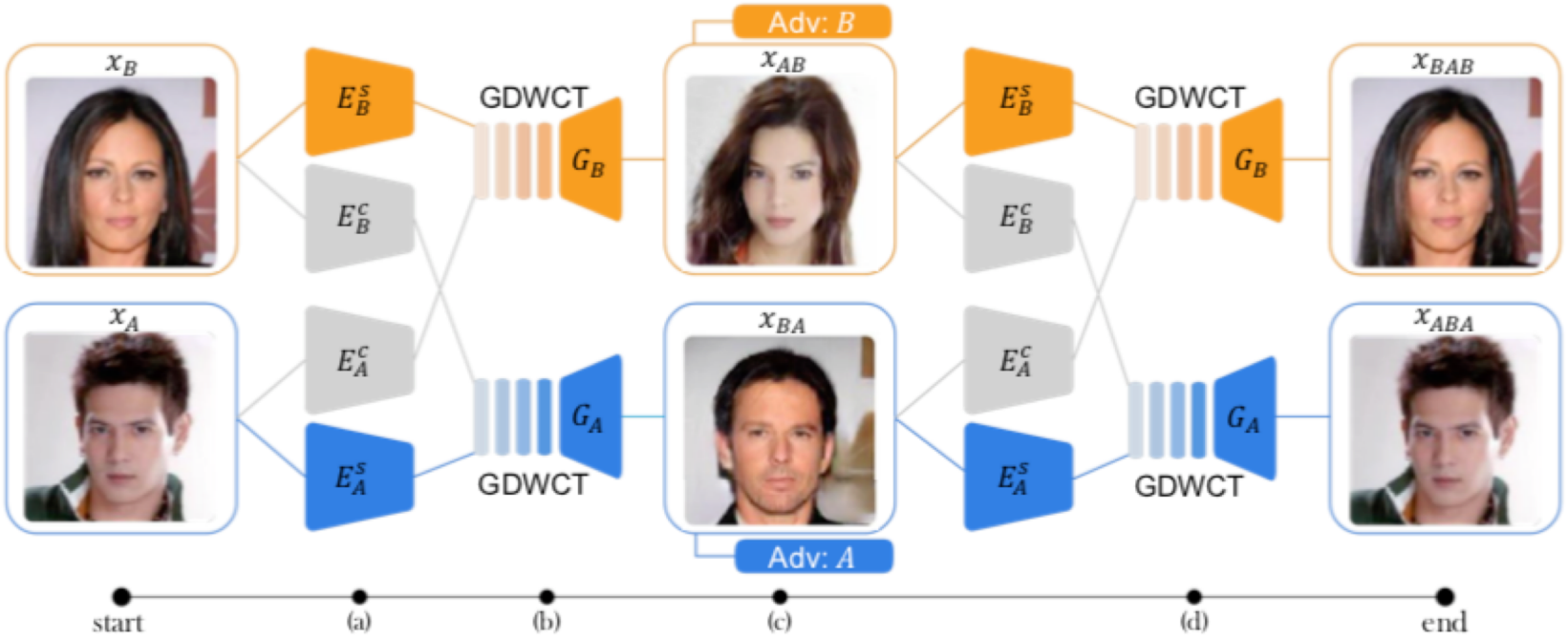


Figure 1: Overview of our model. (a) To translate from $\mathcal{A} \rightarrow \mathcal{B}$, we first extract the content feature $c_{\mathcal{A}}$ from the image $x_{\mathcal{A}}$ (i.e., $c_{\mathcal{A}} = E_{\mathcal{A}}^c(x_{\mathcal{A}})$) and the style feature $s_{\mathcal{B}}$ from the image $x_{\mathcal{B}}$ (i.e., $s_{\mathcal{B}} = E_{\mathcal{B}}^s(x_{\mathcal{B}})$). (b) The obtained features are combined in our GDWCT module while forwarded through the generator $G_{\mathcal{B}}$. (c) The discriminator $D_{\mathcal{B}}$ classifies whether the input $x_{\mathcal{AB}}$ is a real image of the domain $\mathcal{B}$ or not. (d) Similar to the procedures from (a) to (c), the generator $G_{\mathcal{B}}$ generates the reconstructed image $x_{\mathcal{BAB}}$ by combining the content feature $c_{\mathcal{BA}}$ and the style feature $s_{\mathcal{AB}}$.

Proposed methods (2/6)

- **Style consistency loss** – Styles of both style image and translated image are similar

$$\mathcal{L}_s^{\mathcal{A}\to\mathcal{B}} = \mathbb{E}_{x_{\mathcal{A}\to\mathcal{B}},x_{\mathcal{B}}}\ [\|E_{\mathcal{B}}^s(x_{\mathcal{A}\to\mathcal{B}}) - E_{\mathcal{B}}^s(x_{\mathcal{B}})\|_1]$$

- **Content consistency loss** – To maintain the content feature of the input image

$$\mathcal{L}_c^{\mathcal{A}\to\mathcal{B}} = \mathbb{E}_{x_{\mathcal{A}\to\mathcal{B}},x_{\mathcal{A}}}[\|E_{\mathcal{B}}^c(x_{\mathcal{A}\to\mathcal{B}}) - E_{\mathcal{A}}^c(x_{\mathcal{A}})\|_1]$$

- **Cycle consistency loss** and **identity loss** – To obtain high-quality images

$$\mathcal{L}_{cyc}^{\mathcal{A}\to\mathcal{B}\to\mathcal{A}} = \mathbb{E}_{x_{\mathcal{A}}}[\|x_{\mathcal{A}\to\mathcal{B}\to\mathcal{A}} - x_{\mathcal{A}}\|_1]$$
$$\mathcal{L}_i^{\mathcal{A}\to\mathcal{A}} = \mathbb{E}_{x_{\mathcal{A}}}[\|x_{\mathcal{A}\to\mathcal{A}} - x_{\mathcal{A}}\|_1]\,.$$

- **Adversarial Loss** – To minimize the discrepancy between dist(real) and dist(generated)

$$\mathcal{L}_{D_{adv}}^{\mathcal{B}} = \tfrac{1}{2}\,\mathbb{E}_{x_{\mathcal{B}}}[(D(x_{\mathcal{B}}) - 1)^2] + \frac{1}{2}\,\mathbb{E}_{x_{\mathcal{A}\to\mathcal{B}}}[(D(x_{\mathcal{A}\to\mathcal{B}}))^2]$$
$$\mathcal{L}_{G_{adv}}^{\mathcal{B}} = \tfrac{1}{2}\,\mathbb{E}_{x_{\mathcal{A}\to\mathcal{B}}}[(D(x_{\mathcal{A}\to\mathcal{B}}) - 1)^2]$$

Proposed methods (3/6)

Whitening transform

Recall whitening procedure is to transform content feature map f_c to $\widehat{f_c}$ where $\Sigma_c = f_c f_c^T$, $\widehat{f_c}\ \widehat{f_c^T} = I$. However, eigendecomposition is not only computationally intensive but also difficult to backpropagate the gradient signal

The authors propose deep whitening transformation(DWT):

$$R_w = E\left[\left|\left|\Sigma_c - I\right|\right|\right]$$

Then,

$$\widehat{f_c} = f_c - \mu_c$$

However, performing DWT to entire channels may excessively throw away the content feature. Thus, the authors propose group-DWT:

$$f_{ci} \in R^{G\times\left(\frac{C}{G}\right)\times BHW} \Rightarrow \Sigma_{ci} \in \mathrm{R}^{\mathrm{G}\times\left(\frac{\mathrm{C}}{\mathrm{G}}\right)\times\left(\frac{C}{G}\right)}$$

Proposed methods (4/6)

Coloring transform

Recall coloring procedure is to transform whitend feature map $\widehat{f_c}$ to $\widehat{f_{cs}}$ where $\widehat{f_{cs}}\widehat{f_{cs}^T} = f_s f_s^T, \widehat{f_{cs}} = E_s D_s^{\frac{1}{2}} E_s^T \widehat{f_c}$. However, considering same reasons as whitening transform, The authors propose deep coloring transformation(DCT):

$$MLP^{CT}(f_s) = S = UD = E_s D_s^{\frac{1}{2}}$$

where $U \in R^{C \times C}$,orthonormal matrix, $D \in R^{C \times C}$ is diagonal matrix whose diagonal entries correspond to the L_2 norm of each column vector of S. Additionally, add regularization term to ensure orthogonality of U.

$$R_c = E\left[\left||U^T U - I|\right|\right]$$

Then,

$$\widehat{f_{cs}} = UDU^T \widehat{f_c}$$

Due to expensive computational cost, the authors propose group-DCT:

$$Compute\ set\ of\ \{UDU^T\}_i \in R^{\left(\frac{C}{G}\right) \times \left(\frac{C}{G}\right)}$$

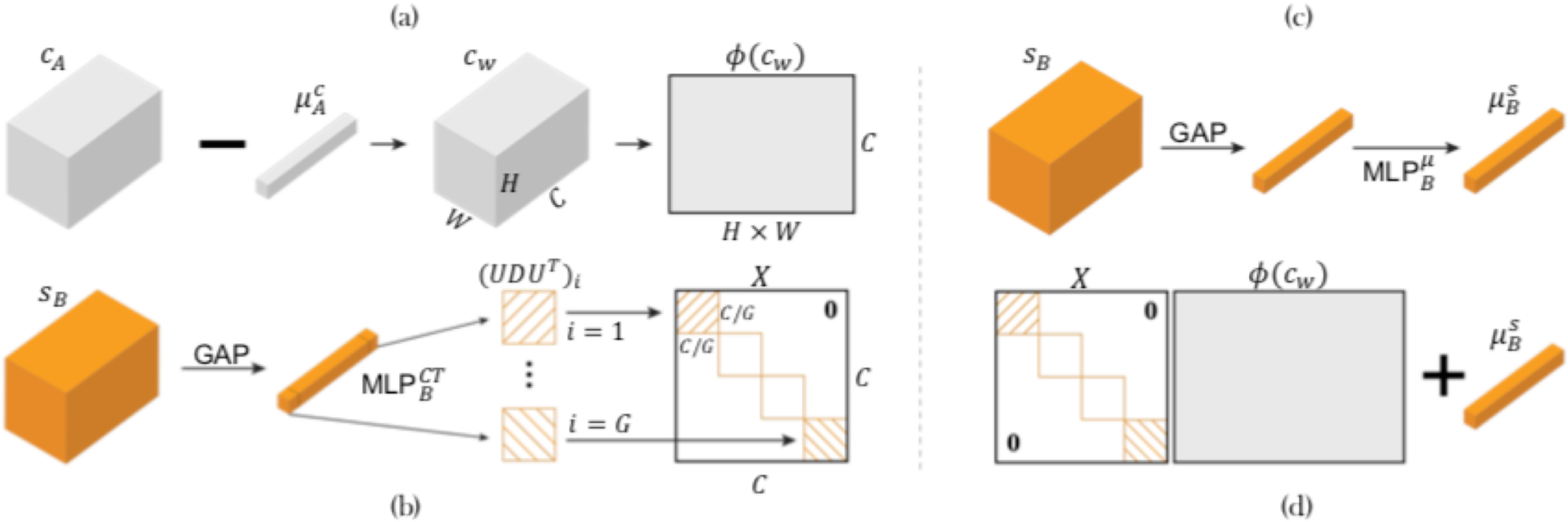


Figure 3: Details on the proposed GDWCT module. (a) The process for obtaining the whitened feature. Because the regularization term (Eq. (2)) encourages the zero-mean content feature $\bar{c}$ to be the whitened feature c_w, we just subtract the mean of the content feature μ_A^c from c_A. (b) The procedure of approximating the coloring transformation matrix (Section 3.3). (c) We obtain the mean of the style feature μ_B^s by forwarding it to the MLP layer MLP_B^μ. (d) Our module first multiply the whitened feature c_w with the group-wise coloring transformation matrix GDCT. We then add it with the mean of the style μ_B^s.

Proposed methods (6/6)

GDWCT overview

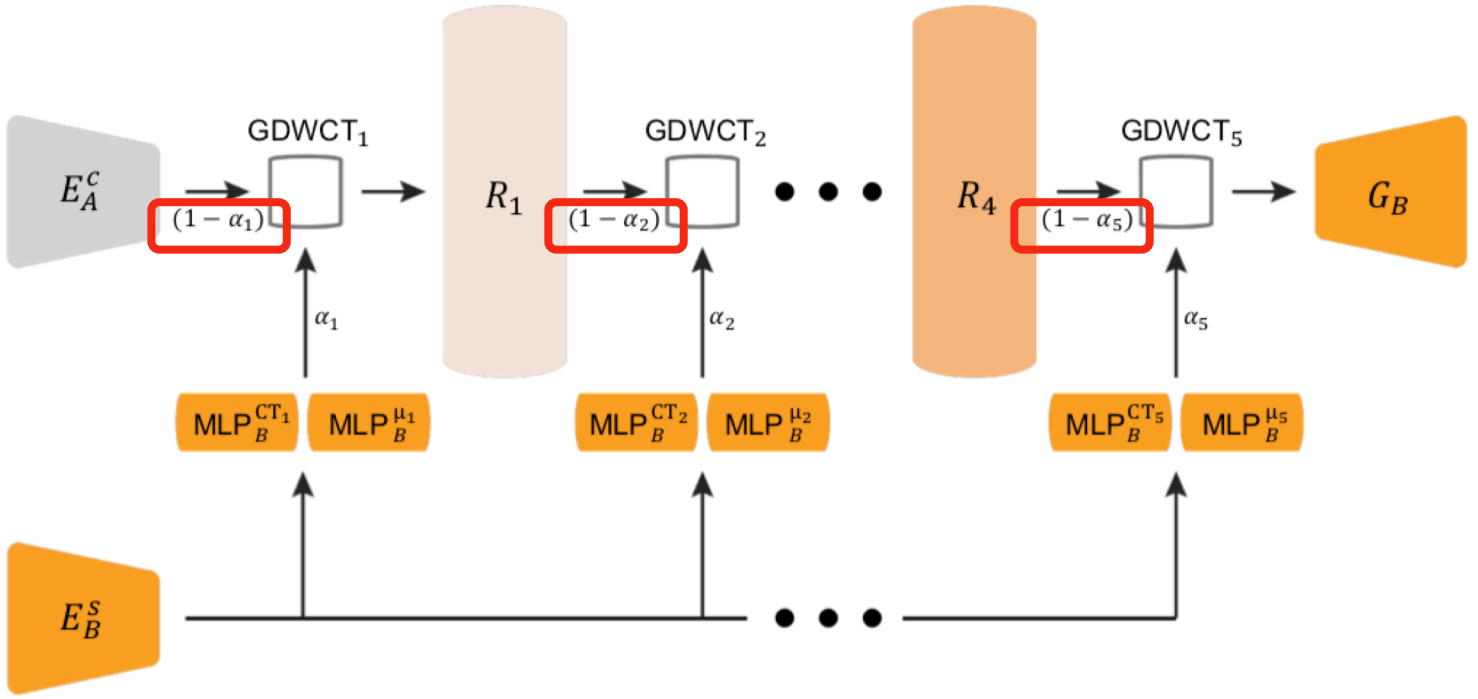
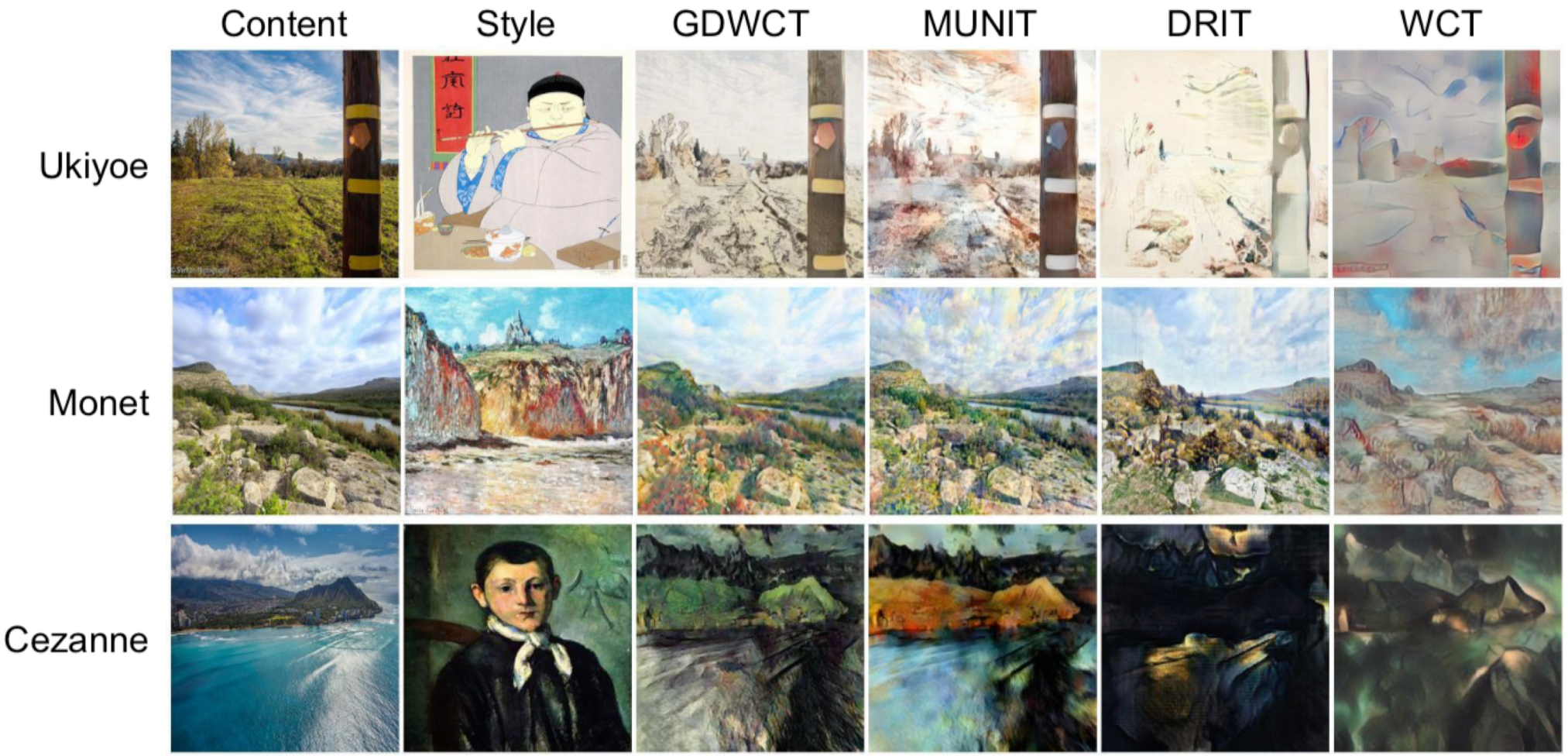


Figure 2: Image translation via the proposed GDWCT. We apply the style via multiple hops to apply the style from the low-level feature to the high-level feature.

Experiments (1/2)

Style transfer.



Experiments (2/2)

Effects of regularization. We verify the influences of the regularizations $\mathcal{R}_w$ and $\mathcal{R}_c$ on the final image output. Intuitively, a higher λ_w will strengthen the whitening transformation, erasing the style more, because it encourages the covariance matrix of the content feature to be closer to the identity matrix. Likewise, a high value of λ_c would result in a diverse level of style, since the intensity of the style applied during coloring increases as the eigenvectors of the style feature gets closer to orthogonal.

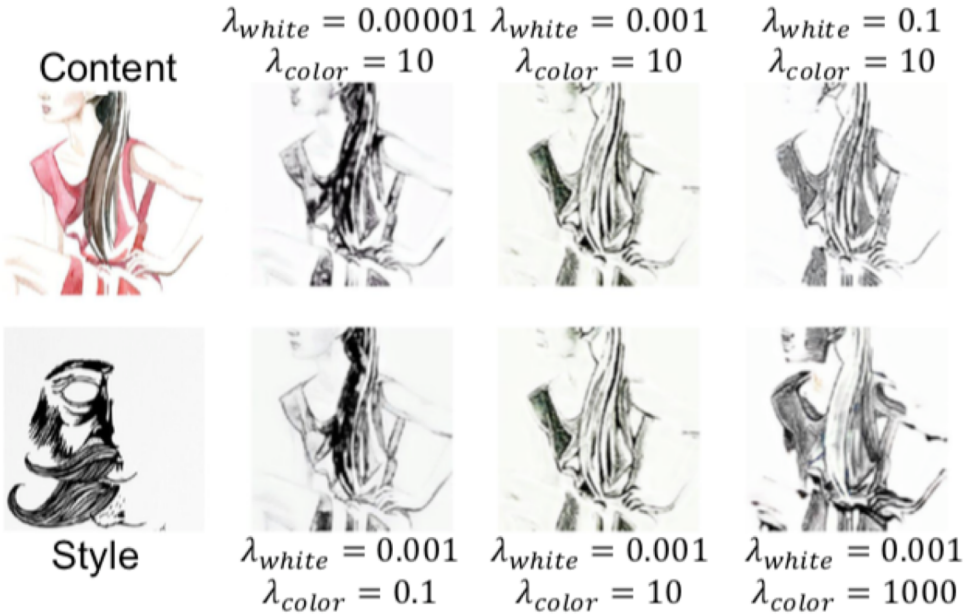


Figure 6: Visualization of the regularization influences.

Thank You