Arbitrary Style Transfer with Style-Attentional Networks

Dae Young Park[1,*] and Kwang Hee Lee[2,*,**]

Artificial Intelligence Research Institute, Korea
[1]likebullet86@gmail.com, [2]lkwanghee@gmail.com

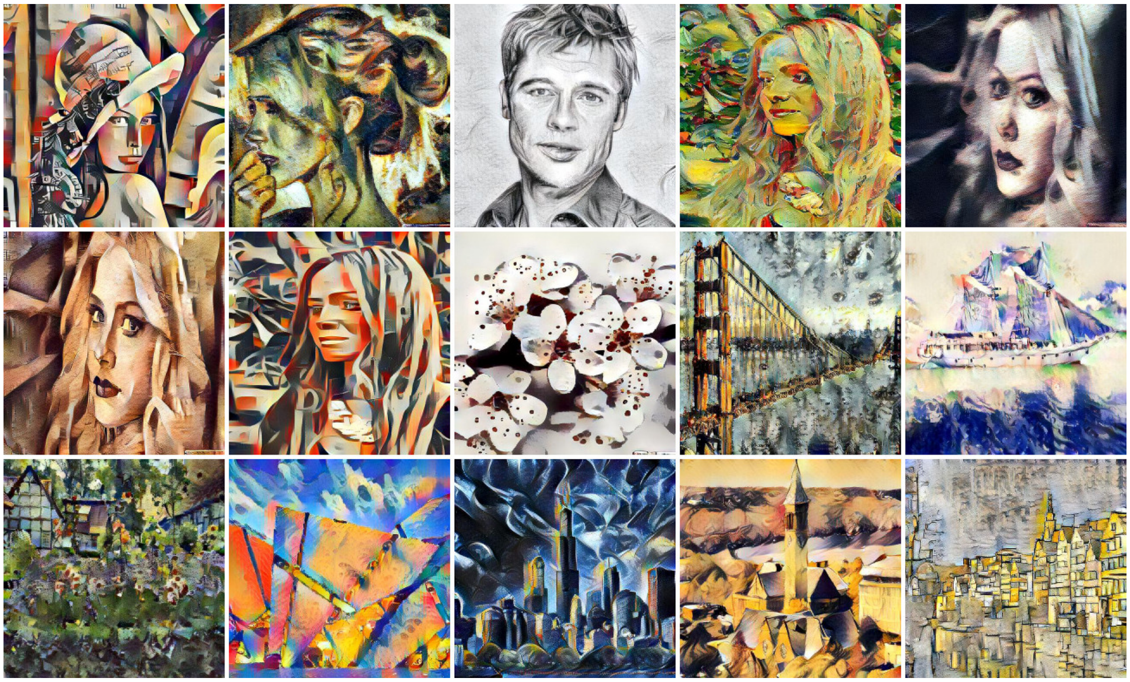


Figure 1: Result of the proposed SANet. We can transfer various styles to content images with high-quality.

Abstract

Arbitrary style transfer aims to synthesize a content image with the style of an image to create a third image that has never been seen before. Recent arbitrary style transfer algorithms find it challenging to balance the content structure and the style patterns. Moreover, simultaneously maintaining the global and local style patterns is difficult due to the patch-based mechanism. In this paper, we introduce a novel style-attentional network (SANet) that efficiently and flexibly integrates the local style patterns according to the semantic spatial distribution of the content image. A new identity loss function and multi-level feature embeddings enable our SANet and decoder to preserve the content structure as much as possible while enriching the style patterns. Experimental results demonstrate that our algorithm synthesizes stylized images in real-time that are higher in quality than those produced by the state-of-the-art algorithms.

1. Introduction

Artistic style transfer is a technique used to create art by synthesizing global and local style patterns from a given style image evenly over a content image while maintaining its original structure. Recently, the seminal work of Gatys et al. [5] showed that the correlation between features ex-

[*] indicates equal contribution
[**] The current affiliation of Kwang Hee Lee is Boeing Korea Engineering and Technology Center.

tracted from a pre-trained deep neural network can capture the style patterns well. The method by Gatys et al. [5] is flexible enough to combine the content and style of arbitrary images, but is prohibitively slow due to the iterative optimization process.

Significant efforts have been made to reduce the computational cost of this process. Several approaches [1, 8, 12, 22, 3, 14, 19, 26, 29] have been developed based on feedforward networks. Feedforward methods can synthesize stylized images efficiently, but are limited to a fixed number of styles or provide insufficient visual quality.

For arbitrary style transfer, a few methods [13, 7, 20] holistically adjust the content features to match the second-order statistics of the style features. AdaIN [7] simply adjusts the mean and variance of the content image to match those of the style image. Although AdaIN effectively combines the structure of the content image and the style pattern by transferring feature statistics, its output suffers in quality due to the over-simplified nature of this method. WCT [13] transforms the content features into the style feature space through a whitening and coloring process with the covariance instead of the variance. By embedding these stylized features within a pre-trained encoder–decoder module, the style-free decoder synthesizes the stylized image. However, if the feature has a large number of dimensions, WCT will accordingly require computationally expensive operations. Avatar-Net [20] is a patch-based style decorator module that maps the content features with the characteristics of the style patterns while maintaining the content structure. Avatar-Net considers not only the holistic style distribution, but also the local style patterns. However, despite valuable efforts, these methods still do not reflect the detailed texture of the style image, distort content structures, or fail to balance the local and global style patterns.

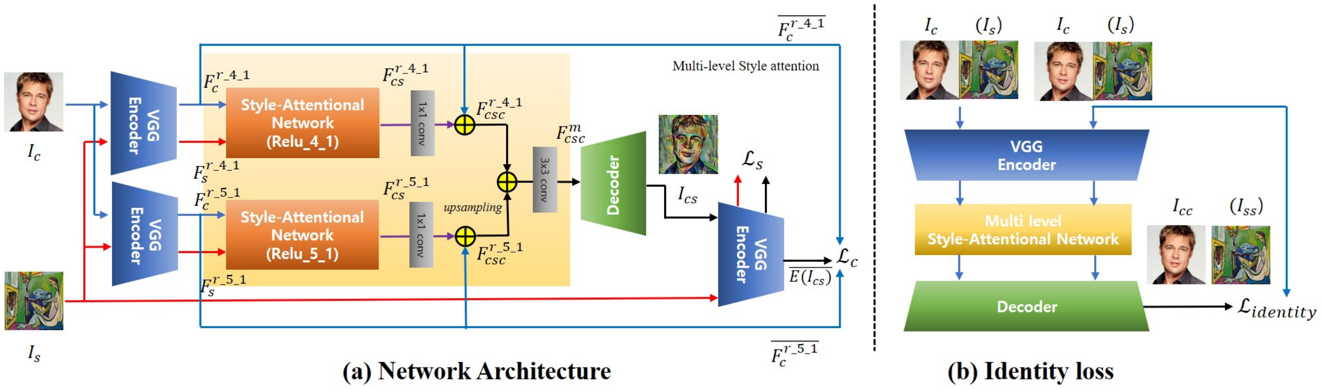
In this work, we propose a novel arbitrary style transfer algorithm that synthesizes high-quality stylized images in real time while preserving the content structure. This is achieved by a new style-attentional network (SANet) and a novel identity loss function. For arbitrary style transfer, our feedforward network, composed of SANets and decoders, learns the semantic correlations between the content features and the style features by spatially rearranging the style features according to the content features.

Our proposed SANet is closely related to the style feature decorator of Avatar-Net [20]. There are, however, two main differences: The proposed model uses 1) a learned similarity kernel instead of a fixed one and 2) soft attention instead of hard attention. In other words, we changed the self-attention mechanism to a learnable soft-attention-based network for the purpose of style decoration. Our SANet uses the learnable similarity kernel to represent the content feature map as a weighted sum of style features that are similar to each of its positions. Using the identity loss during the training, the same image pair are input and our model is trained to restore the same result. At inference time, one of the input images is replaced with the style image, and the content image is restored as much as possible based on the style features. Identity loss, unlike the content–style trade-off, helps to maintain the content structure without losing the richness of the style because it helps restore the contents based on style features. The main contributions of our work are as follows:

- We propose a new SANet to flexibly match the semantically nearest style features onto the content features.
- We present a learning approach for a feedforward network composed of SANets and decoders that is optimized using a conventional style reconstruction loss and a new identity loss.
- Our experiments show that our method is highly efficient (about 18–24 frames per second (fps) at 512 pixels) at synthesizing high-quality stylized images while balancing the global and local style patterns and preserving content structure.

2. Related Work

Arbitrary Style Transfer. The ultimate goal of arbitrary style transfer is to simultaneously achieve and preserve generalization, quality, and efficiency. Despite recent advances, existing methods [5, 4, 1, 8, 12, 22, 3, 6, 10, 11, 23, 24, 28, 18] present a trade-off among generalization, quality, and efficiency. Recently, several methods [13, 20, 2, 7] have been proposed to achieve arbitrary style transfer. The AdaIN algorithm simply adjusts the mean and variance of the content image to match those of the style image by transferring global feature statistics. WCT performs a pair of feature transforms, whitening and coloring, for feature embedding within a pre-trained encoder-decoder module. Avatar-Net introduced the patch-based feature decorator, which transfers the content features to the semantically nearest style features while simultaneously minimizing the difference between their holistic feature distributions. In many cases, we observe that the results of WCT and Avatar-Net fail to sufficiently represent the detailed texture or maintain the content structure. We speculate that WCT and Avatar-Net could fail to synthesize the detailed texture style due to their pre-trained general encoder–decoder networks, which are learned from general images such as MS-COCO datasets [15] with large differences in style characteristics. As a result, these methods consider mapping the style feature onto the content feature in the feature space, but there is no way to control the global statistics or content structure of the style. Although Avatar-Net can obtain the local style patterns through a patch-based style decorator, the scale of style patterns in the style images depends on the patch size. Therefore, the global and local style patterns cannot both be taken into consideration. In contrast, AdaIN transforms



(a) Network Architecture **(b) Identity loss**

Figure 2: Overview of training flow. (a) Fixed VGG encoder encoding content and style images. Two SANets map features from `Relu_4_1` and `Relu_5_1` features respectively. The decoder transforms the combined SANet output features to I_{cs} (Eq. 4). The fixed VGG encoder is used to compute $\mathcal{L}_c$ (Eq. 7) and $\mathcal{L}_s$ (Eq. 8). (b) The identity loss $\mathcal{L}_{identity}$ (Eq. 9) quantifies the difference between I_c and I_{cc} or between I_s and I_{ss}, where I_c (I_s) is the original content (style) image and I_{cc} (I_{ss}) is the output image synthesized from the image pair (content or style).

texture and color distribution well, but does not represent local style patterns well. In this method, there exists another trade-off between content and style due to a combination of scale-adapted content and style loss. In this paper, we try to solve these problems using the SANets and the proposed identity loss. In this way, the proposed style transfer network can represent global and local style patterns and maintain the content structure without losing the richness of the style.

Self-Attention Mechanism. Our style-attentional module is related to the recent self-attention methods [25, 30] for image generation and machine translation. These models calculate the response at a position in a sequence or an image by attending to all positions and taking their weighted average in an embedding space. The proposed SANet learns the mapping between the content features and the style features by slightly modifying the self-attention mechanism.

3. Method

The style transfer network proposed in this paper is composed of an encoder–decoder module and a style-attentional module, as shown in Fig. 2. The proposed feedforward network effectively generates high-quality stylized images that appropriately reflect global and local style patterns. Our new identity loss function helps to maintain the detailed structure of the content while reflecting the style sufficiently.

3.1. Network Architecture

Our style transfer network takes a content image I_c and an arbitrary style image I_s as inputs, and synthesizes a stylized image I_{cs} using the semantic structures from the former and characteristics from the latter. In this work, the pretrained VGG-19 network [21] is employed as encoder and a symmetric decoder and two SANets are jointly trained for arbitrary style transfer. Our decoder follows the settings of [7].

To combine global style patterns and local style patterns adequately, we integrate two SANets by taking the VGG feature maps encoded from different layers (`Relu_4_1` and `Relu_5_1`) as inputs and combining both output feature maps. From a content image I_c and style image I_s pair, we first extract their respective VGG feature maps $F_c = E(I_c)$ and $F_s = E(I_s)$ at a certain layer (e.g., `Relu_4_1`) of the encoder.

After encoding the content and style images, we feed both feature maps to a SANet module that maps the correspondences between the content feature map F_c and the style feature map F_s, producing following the output feature map:

$$F_{cs} = SANet\left(F_c, F_s\right) \tag{1}$$

After applying 1×1 convolution to F_{cs} and summing the two matrices element-wise as follows, we obtain F_{csc}:

$$F_{csc} = F_c + W_{cs} F_{cs}, \tag{2}$$

where "+" denotes element-wise summation.

We combine two the output feature maps from the two SANets as

$$F_{csc}^{m} = conv_{3\times3}(F_{csc}^{\text{r\_4\_1}} + upsampling(F_{csc}^{\text{r\_5\_1}})), \tag{3}$$

where $F_{csc}^{\text{r\_4\_1}}$ and $F_{csc}^{\text{r\_5\_1}}$ are the output feature maps obtained from the two SANets, $conv_{3\times3}$ denotes the 3×3 con-

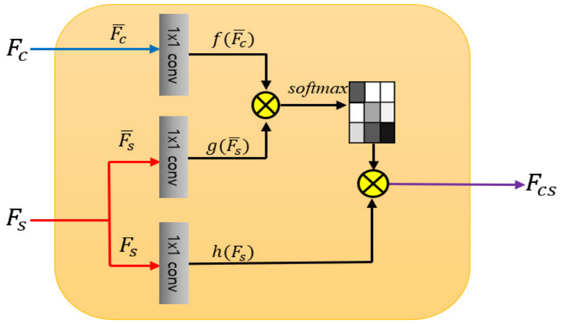


Figure 3: SANet.

volution used to combine the two feature maps, and $F_{csc}^{\text{r\_5\_1}}$ is added to $F_{csc}^{\text{r\_4\_1}}$ after upsampling.

Then, the stylized output image I_{cs} is synthesized by feeding F_{csc}^{m} into the decoder as follows:

$$I_{cs} = D\left(F_{csc}^{m}\right). \tag{4}$$

3.2. SANet for Style Feature Embedding

Figure 3 shows style feature embedding using the SANet module. Content feature maps F_c and style feature maps F_s from the encoder are normalized and then transformed into two feature spaces f and g to calculate the attention between $\overline{F_c^i}$ and $\overline{F_s^j}$ as follows:

$$F_{cs}^{i} = \frac{1}{C(F)} \sum_{\forall j} \exp(f(\overline{F_c^i})^T g(\overline{F_s^j}))h(F_s^j), \tag{5}$$

where $f(\overline{F_c}) = W_f \overline{F_c}$, $g(\overline{F_s}) = W_g \overline{F_s}$, and $h(F_s) = W_h F_s$. Further, $\overline{F}$ denotes a mean–variance channel-wise normalized version of F. The response is normalized by a factor $C(F) = \sum_{\forall j} \exp(f(\overline{F_c^i})^T g(\overline{F_s^j}))$. Here, i is the index of an output position and j is the index that enumerates all possible positions. In the above formulation, W_f, W_g, and W_h are the learned weight matrices, which are implemented as 1×1 convolutions as in [30].

Our SANet has a network structure similar to the existing non-local block structure [27], but the number of input data differ (the input of the SANet consists of F_c and F_s). The SANet module can appropriately embed a local style pattern in each position of the content feature maps by mapping a relationship (such as affinity) between the content and style feature maps through learning.

3.3. Full System

As shown in Fig. 2, we use the encoder (a pre-trained VGG-19 [21]) to compute the loss function for training the SANet and decoder:

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \lambda_s \mathcal{L}_s + \mathcal{L}_{identity}, \tag{6}$$

where the composers of content, style, and identity loss are $\mathcal{L}_c$, $\mathcal{L}_s$, and $\mathcal{L}_{identity}$, respectively, and λ_c and λ_s are the weights of different losses.

Similar to [7], the content loss is the Euclidean distance between the mean–variance channel-wise normalized target features, $\overline{F_c^{\text{r\_4\_1}}}$ and $\overline{F_c^{\text{r\_5\_1}}}$ and the mean–variance channel-wise normalized features of the output image VGG features, $\overline{E(I_{cs})^{\text{r\_4\_1}}}$ and $\overline{E(I_{cs})^{\text{r\_5\_1}}}$, as follows:

$$\mathcal{L}_c = ||\overline{E(I_{cs})^{\text{r\_4\_1}}} - \overline{F_c^{\text{r\_4\_1}}}||_2 + ||\overline{E(I_{cs})^{\text{r\_5\_1}}} - \overline{F_c^{\text{r\_5\_1}}}||_2. \tag{7}$$

The style loss is defined as follows:

$$\begin{aligned} \mathcal{L}_s = \sum_{i=1}^{L} & ||\mu(\phi_i(I_{cs})) - \mu(\phi_i(I_s))||_2 \\ & + ||\sigma(\phi_i(I_{cs})) - \sigma(\phi_i(I_s))||_2, \end{aligned} \tag{8}$$

where each ϕ denotes a feature map of the layer in the encoder used to compute the style loss. We use `Relu_1_1`, `Relu_2_1`, `Relu_3_1`, `Relu_4_1`, and `Relu_5_1` layers with equal weights. We have applied both the Gram matrix loss [5] and the AdaIN style loss [7], but the results show that the AdaIN style loss is more satisfactory.

When W_f, W_g, and W_h are fixed as the identity matrices, each position in the content feature maps can be transformed into the semantically nearest feature in the style feature maps. In this case, the system cannot parse sufficient style features. In the SANet, although W_f, W_g, and W_h are learnable matrices, our style transfer model can be trained by considering only the global statistics of the style loss $\mathcal{L}_s$.

To consider both the global statistics and the semantically local mapping between the content features and the style features, we define a new identity loss function as

$$\begin{aligned} \mathcal{L}_{identity} = & \lambda_{identity1}(||(I_{cc} - I_c)||_2 + ||(I_{ss} - I_s)||_2) \\ & + \lambda_{identity2} \sum_{i=1}^{L} (||\phi_i(I_{cc}) - \phi_i(I_c)||_2 \\ & + ||\phi_i(I_{ss}) - \phi_i(I_s)||_2), \end{aligned} \tag{9}$$

where I_{cc}(or I_{ss}) denotes the output image synthesized from two same content (or style) images, each ϕ_i denotes a layer in the encoder, and $\lambda_{identity1}$ and $\lambda_{identity2}$ are identity loss weights. The weighting parameters are simply set as $\lambda_c = 1$, $\lambda_s = 3$, $\lambda_{identity1} = 1$, and $\lambda_{identity2} = 50$ in our experiments.

The content and style losses control the trade-off between the structure of the content image and the style patterns. Unlike the other two losses, the identity loss is calculated from the same input images with no gap in style characteristics. Therefore, the identity loss concentrates

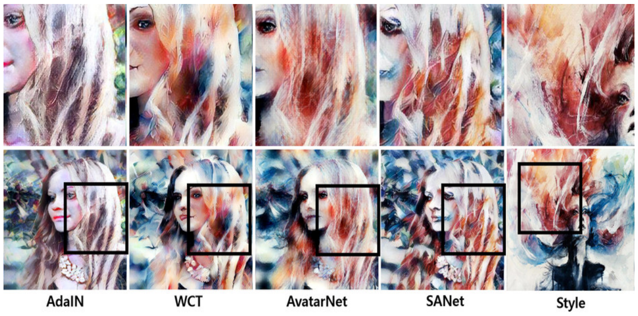


Figure 4: Result details. Regions marked by bounding boxes in the bottom row are enlarged in the top row for better visualization.

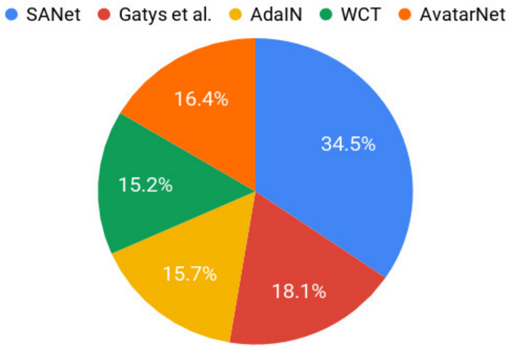


Figure 5: User preference result of five style transfer algorithms.

Method	Time (256 px)	Time (512 px)
Gatys et al. [5]	15.863	50.804
WCT [13]	0.689	0.997
Avatar-Net [20]	0.248	0.356
AdaIN [7]	0.011	0.039
ours (`Relu_4_1`)	0.012	0.042
ours (multi-level)	0.017	0.055

Table 1: Execution time comparison (in seconds).

on keeping the structure of the content image rather than changing the style statistics. As a result, the identity loss makes it possible to maintain the structure of the content image and style characteristics of the reference image simultaneously.

4. Experimental Results

Figure 2 shows an overview of our style transfer network based on the proposed SANets. The demo site will be release at https://dypark86.github.io/SANET/.

4.1. Experimental Settings

We trained the network using MS-COCO [15] for the content images and WikiArt [17] for the style images. Both datasets contain roughly 80,000 training images. We used the Adam optimizer [9] with a learning rate of 0.0001 and a batch size of five content–style image pairs. During training, we first rescaled the smaller dimension of both images to 512 while preserving the aspect ratio, then randomly cropped a region of size 256×256 pixels. In the testing phase, our network can handle any input size because it is fully convolutional.

4.2. Comparison with Prior Work

To evaluate the our method, we compared it with three types of arbitrary style transform methods: the iterative optimization method proposed by Gatys et al. [5], two feature transformation-based methods (WCT [13] and AdaIN [7]), and the patch-based method Avatar-Net [20].

Qualitative examples. In Fig. 11, we show examples of style transfer results synthesized by the state-of-the-art methods. Additional results are provided in the supplementary materials. Note that none of the test style images were observed during the training of our model.

The optimization-based method [5] allows arbitrary style transfer, but is likely to encounter a bad local minimum (e.g., rows 2 and 4 in Fig. 11). AdaIN [7] simply adjusts the mean and variance of the content features to synthesize the stylized image. However, its results are less appealing and often retain some of the color distribution of the content due to the trade-off between content and style (e.g., rows 1, 2, and 8 in Fig. 11). In addition, both AdaIN [7] and WCT [13] sometimes yield distorted local style patterns because of the holistic adjustment of the content features to match the second-order statistics of the style features, as shown in Fig. 11. Although Avatar-Net [20] decorates the image with the style patterns according to the semantic spatial distribution of the content image and applies a multi-scale style transfer, it frequently cannot represent the local and global style patterns at the same time due to its dependency on the patch size. Moreover, it cannot keep the content structure in most cases (column 4 in Fig. 11). In contrast, our method can parse diverse style patterns such as global color distribution, texture, and local style patterns while maintaining the structure of the content in most examples, as shown in Fig. 11.

Unlike other algorithms, our learnable SANets can flexibly parse a sufficient level of style features without maximally aligning the content and style features, regardless a large domain gap (rows 1 and 6 in Fig. 11). The proposed SANet semantically distinguishes the content structure and transfers similar style patterns onto the regions with the same semantic meaning. Our method transfers different styles for each type of semantic content. In Fig. 11 (row 3), the sky and buildings in our stylized image are stylized using different style patterns, whereas the results of other methods have ambiguous style boundaries between the sky and buildings.

We also provide details of the results in Fig. 4. Our results exhibit multi-scale style patterns (e.g., color distribution, bush strokes, and the white and red patterns of rough textures in the style image). Avatar-Net and WCT distort the brush strokes, output blurry hair texture, and do not preserve the appearance of the face. AdaIN cannot even preserve the color distribution.

User study. We used 14 content images and 70 style images to synthesize 980 images in total. We randomly selected 30 content and style combinations for each subject and showed them the stylized images obtained by the five comparison methods side-by-side in a random order. We then asked the subject to indicate his/her favorite result for each style. We collect 2,400 votes from 80 users and show the percentage of votes for each method in Fig. 5. The result shows that the stylized results obtained by our method are preferred more often than those of other methods.

Efficiency. Table 1 shows the run time performance of the proposed method and other methods at two image scales: 256 and 512 pixels. We measured the runtime performance, including the time for style encoding. The optimization-based method [5] is impractically computationally expensive because of its iterative optimization process. In contrast, our multi-scale models (`Relu_4_1` and `Relu_5_1`) algorithms run at 59 fps and 18 fps for 256- and 512-pixel images respectively, and the single-scale (only `Relu_4_1`) algorithms runs at 83 fps and 24 fps for 256- and 512-pixel images respectively. Therefore, our method could feasibly process style transfer in real time. Our model is 7–20 times faster than the matrix computation-based methods (WCT [13] and Avatar-Net [20]).

4.3. Ablation Studies

Loss analysis. In this section, we show the influence of content-style loss and identity loss. Figure 6 (a) shows the results obtained by fixing $\lambda_{identity1}$, $\lambda_{identity2}$, and λ_s at 0, 0, and 5, respectively, while increasing λ_c from 1 to 50. Figure 6 (b) shows the results obtained by fixing λ_c and λ_s at 0 and 5, respectively, and increasing $\lambda_{identity1}$ and $\lambda_{identity2}$ from 1 to 100 and from 50 to 5,000, respectively. Without the identity loss, if we increase the weight of the content loss, the content structure is preserved, but the characteristics of the style patterns disappear, because of the trade-off between the content loss and the style loss. In contrast, increasing the weights of identity loss without content loss preserves the content structure as much as possible while maintaining style patterns. However, distortion of the content structure cannot be avoided. We hence applied a combination of content-style loss and identity loss to maintain the content structure while enriching style patterns.

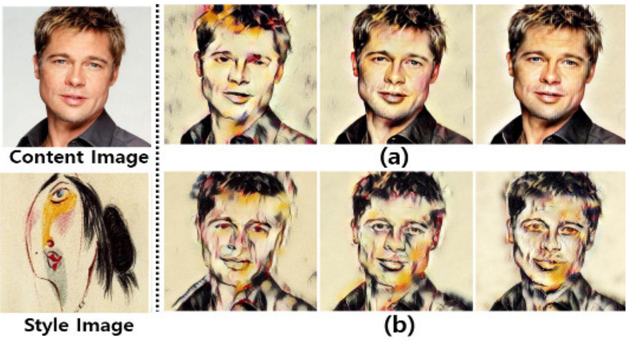


Figure 6: Content-style loss vs. identity loss. (a) Results obtained by fixing $\lambda_{identity1}$, $\lambda_{identity2}$, and λ_s at 0, 0, and 5, respectively, and increasing λ_c from 1 to 50. (b) Results obtained by fixing λ_c and λ_s at 0 and 5, respectively, and increasing $\lambda_{identity1}$ and $\lambda_{identity2}$ from 1 to 100 and from 50 to 5,000, respectively.

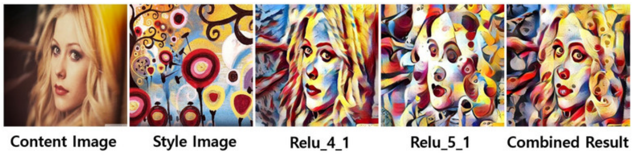


Figure 7: Multi-level feature embedding. By embedding features at multiple levels, we can enrich the local and global patterns for the stylized images.

Multi-level feature embedding. Figure 7 shows two stylized outputs obtained from `Relu_4_1` and `Relu_5_1`, respectively. When only `Relu_4_1` is used for style transfer, the global statistics of the style features and the content structure are maintained well. However, the local style patterns do not appear well. In contrast, `Relu_5_1` helps add the local style patterns such as circle patterns because the receptive field is wider. However, the content structures are distorted and textures such as brush strokes disappear. In our work, to enrich the style patterns, we integrated two SANets by taking VGG feature maps encoded from the different (`Relu_4_1` and `Relu_5_1`) layers as inputs and combining both output feature maps

4.4. Runtime Controls

In this section, we present the flexibility of our method through several applications.

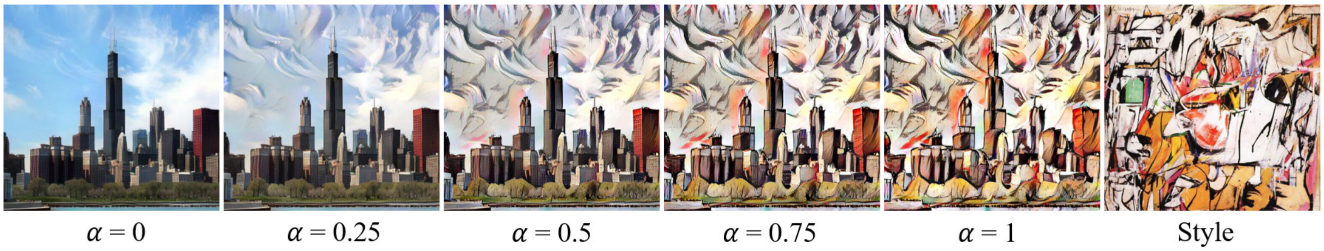


Figure 8: Content–style trade-off during runtime. Our algorithm allows this trade-off to be adjusted at runtime by interpolating between feature maps F^m_{ccc} and F^m_{csc}.

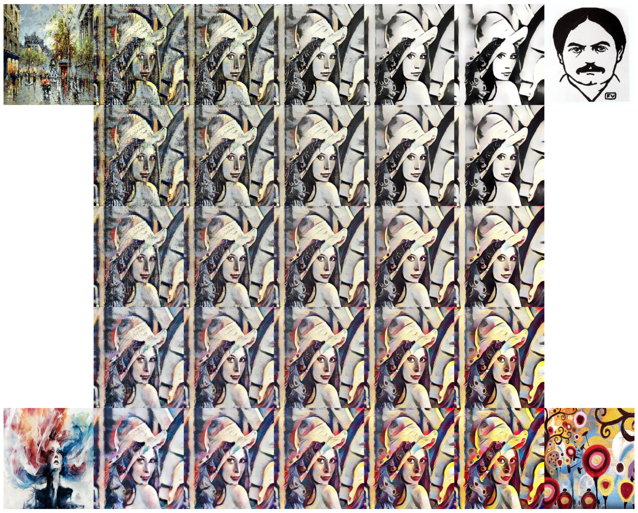


Figure 9: Style interpolation with four different styles.

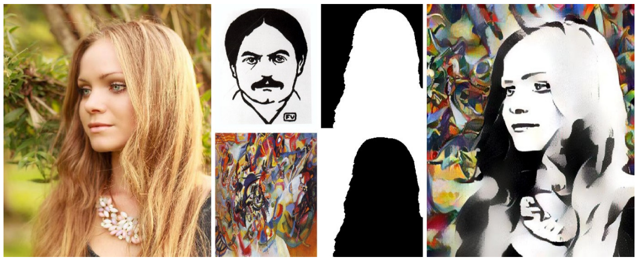


Figure 10: Example of spatial control. Left: content image. Middle: style images and masks. Right: stylized image from two different style images.

Content–style trade-off. The degree of stylization can be controlled during training by adjusting the style weight λ_s in Eq. 6 or during test time by interpolating between feature maps that are fed to the decoder. For runtime control, we adjust the stylized features $F^m_{csc} \leftarrow \alpha F^m_{csc} + (1-\alpha) F^m_{ccc}$ and $\forall \alpha \in [0, 1]$. Map F^m_{ccc} is obtained by taking two content images as input for our model. The network tries to reconstruct the content image when $\alpha = 0$, and to synthesize the most stylized image when $\alpha = 1$ (as shown in Fig. 8).

Style interpolation. To interpolate between several style images, a convex combination of feature maps F^m_{csc} from different styles can be fed into the decoder (as shown in Fig. 9).

Spatial control. Figure 10 shows an example of spatially controlling the stylization. A set of masks M (Fig. 10 column 3) is additionally required as input to map the spatial correspondence between content regions and styles. We can assign the different styles in each spatial region by replacing F^m_{csc} with $M \bigodot F^m_{csc}$, where $\bigodot$ is a simple mask-out operation.

5. Conclusions

In this work, we proposed a new arbitrary style transform algorithm that consists of style-attentional networks and decoders. Our algorithm is effective and efficient. Unlike the patch-based style decorator in [20], our proposed SANet can flexibly decorate the style features through learning using a conventional style reconstruction loss and identity loss. Furthermore, the proposed identity loss helps the SANet maintain the content structure, enriching the local and global style patterns. Experimental results demonstrate that the proposed method synthesizes images that are preferred over other state-of-the-art arbitrary style transfer algorithms.

Acknowledgments. This research is supported by the Ministry of Culture, Sports, and Tourism (MCST) and Korea Creative Content Agency (KOCCA) in the Culture Technology (CT) Research and Development Program 2019

References

[1] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua. StyleBank: An explicit representation for neural image style transfer. In *Proc. CVPR*, volume 1, page 4, 2017.

[2] T. Q. Chen and M. Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016.

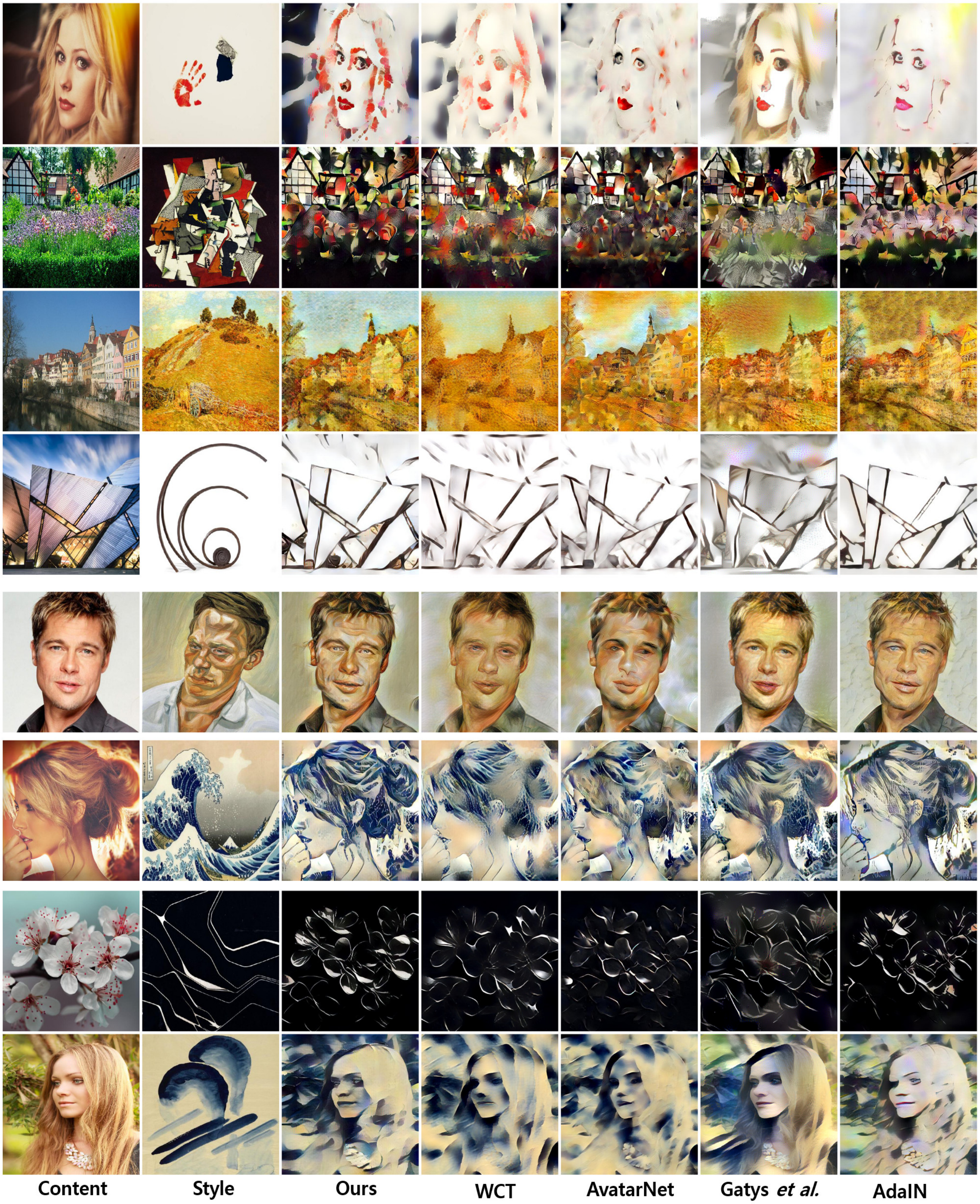


Figure 11: Example results for comparisons.

[3] V. Dumoulin, J. Shlens, and M. Kudlur. A learned representation for artistic style. In *Proc. ICLR*, 2017.

[4] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015.

[5] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proc. CVPR*, pages 2414–2423, 2016.

[6] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. Controlling perceptual factors in neural style transfer. In *Proc. CVPR*, 2017.

[7] X. Huang and S. J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proc. ICCV*, pages 1510–1519, 2017.

[8] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proc. ECCV*, pages 694–711. Springer, 2016.

[9] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] C. Li and M. Wand. Combining Markov random fields and convolutional neural networks for image synthesis. In *Proc. CVPR*, pages 2479–2486, 2016.

[11] C. Li and M. Wand. Precomputed real-time texture synthesis with Markovian generative adversarial networks. In *Proc. ECCV*, pages 702–716. Springer, 2016.

[12] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Diversified texture synthesis with feed-forward networks. In *Proc. CVPR*, 2017.

[13] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang. Universal style transfer via feature transforms. In *Advances in Neural Information Processing Systems*, pages 386–396, 2017.

[14] Y. Li, N. Wang, J. Liu, and X. Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017.

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Proc. ECCV*, pages 740–755. Springer, 2014.

[16] A. Paszke, S. Chintala, R. Collobert, K. Kavukcuoglu, C. Farabet, S. Bengio, I. Melvin, J. Weston, and J. Mariethoz. PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration, Available: https://github.com/pytorch/pytorch, May 2017.

[17] F. Phillips and B. Mackintosh. Wiki Art Gallery, Inc.: A case for critical thinking. *Issues in Accounting Education*, 26(3):593–608, 2011.

[18] E. Risser, P. Wilmot, and C. Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. *arXiv preprint arXiv:1701.08893*, 2017.

[19] F. Shen, S. Yan, and G. Zeng. Meta networks for neural style transfer. *arXiv preprint arXiv:1709.04111*, 2017.

[20] L. Sheng, Z. Lin, J. Shao, and X. Wang. Avatar-Net: Multi-scale zero-shot style transfer by feature decoration. In *Proc. CVPR*, pages 8242–8250, 2018.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[22] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proc. ICML*, pages 1349–1357, 2016.

[23] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, (2016).

[24] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proc. CVPR*, volume 1, page 3, 2017.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[26] H. Wang, X. Liang, H. Zhang, D.-Y. Yeung, and E. P. Xing. ZM-Net: Real-time zero-shot image manipulation network. *arXiv preprint arXiv:1703.07255*, 2017.

[27] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. *arXiv preprint arXiv:1711.07971*, 2017.

[28] X. Wang, G. Oxholm, D. Zhang, and Y.-F. Wang. Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. In *Proc. CVPR*, volume 2, page 7, 2017.

[29] H. Zhang and K. Dana. Multi-style generative network for real-time transfer. *arXiv preprint arXiv:1703.06953*, 2017.

[30] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.