Parameter-Free Style Projection for Arbitrary Style Transfer

Siyu Huang
Baidu Research
huangsiyu@baidu.com

Haoyi Xiong
Baidu Research
xionghaoyi@baidu.com

Tianyang Wang
Austin Peay State University
toseattle@siu.edu

Qingzhong Wang
City University of Hong Kong
qingzwang2-c@my.cityu.edu.hk

Zeyu Chen
Baidu Inc.
chenzeyu01@baidu.com

Jun Huan
Styling AI
lukehuan@shenshangtech.com

Dejing Dou
Baidu Research
doudejing@baidu.com

Figure 1: Stylizing Brad Pitt with Picasso's Self-Portrait using the state-of-the-art arbitrary image style transfer methods. Our Style Projection algorithm shows an appealing result.

ABSTRACT

Arbitrary image style transfer is a challenging task which aims to stylize a content image conditioned on an arbitrary style image. In this task the content-style feature transformation is a critical component for a proper fusion of features. Existing feature transformation algorithms often suffer from unstable learning, loss of content and style details, and non-natural stroke patterns. To mitigate these issues, this paper proposes a parameter-free algorithm, Style Projection, for fast yet effective content-style transformation. To leverage the proposed Style Projection component, this paper further presents a real-time feed-forward model for arbitrary style transfer, including a regularization for matching the content semantics between inputs and outputs. Extensive experiments have demonstrated the effectiveness and efficiency of the proposed method in terms of qualitative analysis, quantitative evaluation, and user study.

KEYWORDS

artistic style transfer, feature transformation

1 INTRODUCTION

Image style transfer task aims to properly fuse a style image (e.g., color and texture patterns) into an image while the content details (e.g., shapes and objects) are effectively preserved. Recently, Neural Style Transfer has demonstrated the effectiveness of deep neural networks in style transfer by optimizing a deep neural network model in an online or offline way [17]. The offline methods, typically a feed-forward network [10], are much faster in inference as they require no optimization process after the learning of models. Specifically, arbitrary style transfer is a very challenging sub-task of offline style transfer, which aims to synthesize artistic images conditioned on arbitrary styles.

In arbitrary style transfer, the content-style feature transformation plays a vital role and a series of content-style transformation algorithms have been proposed in the literature. The widely-used normalization-based methods including instance normalization (IN) [37], conditional IN (CIN) [8], and adaptive IN (AdaIN) [16] generally normalize the style of content features to style features. More recently, the whitening and coloring transformation (WCT) is employed to peel off and recover style information on content features. Although the whitening operation was designed to remove style information, it has side effect that content details are also removed unintentionally, leading to unpleasant synthesized results as shown in Fig. 1.

Inspired by the order statistics [5] that tells the order of random variables contains effective information, we introduce order statistics into content-style feature transformation. Specifically, we separate the content and style information into the order statistics and scalar values of features, respectively. Based on this, we propose a novel parameter-free content-style transformation algorithm, Style Projection, for real-time arbitrary image style transfer. In a simple manner, Style Projection reorders the style features according to the order of the content features, such that the correlation of feature values (shape, texture, etc.) is provided by the content features, while the scaling information (color patterns) is provided by the style features, thus enabling a reasonable content-style fusion.

We further present a real-time learning-based feed-forward model to leverage Style Projection for arbitrary style transfer. Our model

Figure 2: Arbitrary image style transfer examples synthesized by our real-time Style Projection method. The style images are shown in upper left black boxes, respectively.

works in an encoder-decoder fashion that stylizes a content image based on arbitrarily given style images. In addition to transferring style structures via Style Projection effectively, we observe that the semantics of a content image are also desired to be properly reconstructed in its corresponding synthesized image. Motivated by this, we propose to regularize the distribution distance of content images and synthesized images in feature space by minimizing the KL divergence, enabling a more distinct recovering of content details. Fig. 2 shows an example of our arbitrary image style transfer method. In experiments, we conduct extensive empirical studies including quantitative evaluation, qualitative analysis, ablation studies, and user study, to comprehensively validate the effectiveness and efficiency of our style transfer framework. The contributions of this paper are summarized as follows.

- We present a parameter-free method, namely Style Projection, for fast yet effective content-style feature transformation.
- We present a real-time feed-forward model for arbitrary style transfer, including a KL divergence loss for further matching the content semantics between input and output.
- We demonstrate the effectiveness and efficiency of our proposed methods through extensive empirical studies.

2 RELATED WORK

Image style transfer. Image style transfer [3, 8, 11, 12, 22–25, 27, 29] is a long term open and challenging problem in computer vision. The key aspect of successfully transferring style to a synthesized image is how to fuse content and style features in an appropriate manner. This makes the problem a texture/color pattern transfer task in nature. Pioneered by [10], deep learning based solutions have gained much interest. Gatys et al. [10] for the first time adopted a convolutional network to extract features for both content and style images and match the statistics in feature space. The style transfer process starts from a random noise input and iteratively updates this input until its features match the content and style features simultaneously. Such an optimization process costs a long time to converge and limits its applications.

Several later works [18, 23, 36] attempted to adopt a feed-forward neural network to replace the optimization process and achieved notable improvements on time efficiency, while the quality of synthesized images was also enhanced. For instance, Style-Swap [4] matches the statistics of content and style image patches and swaps the content patches with their closest-matching style patches. The Style Projection proposed in this paper falls into the category of fast and arbitrary style transfer. The normalization-based methods [8, 16, 37] and WCT-based methods [26] achieve high running time efficiency such that they are widely used in various style transformation applications [1, 2, 9, 19, 31, 38] in practice. Recently, Li et al. [24] proposed a light-weighted linear propagation method for image and video style transfer and obtained promising results.

Feature transformation. For arbitrary style transfer, content-style feature transformation algorithm is the critical component. Traditional approaches such as histogram matching [15], multi-resolution [6], and wavelet [33] transfer low-level statistics. However the semantics cannot be properly recovered in synthesized images. Recent advances focus on transformation algorithms that can be deployed with deep neural networks without making extra effort. WCT [26] is a typical such method which transforms content features using a whitening operation that makes the features uncorrelated, and then fuses content and style features by a coloring step which is actually the inverse of the whitening operation. Normalization-based methods, such as IN [37], CIN [9], and AdaIN [16, 19], conduct normalization on features for efficient content-style transformation. Since the normalization-based methods rely on the first-order statistics of content and style features, they are able to transfer the global style structures effectively whereas some style details may not be well captured.

In this paper, we present Style Projection to synthesize appealing stylized images in a real-time way by reordering the style features based on the order of content features. A related work is Deep Reshuffle [14] which learns a feature shuffling function under the supervision of an artifact patch-based loss. Deep Reshuffle is built upon a parametric and data-driven feature shuffling function such that it heavily relies on the distributions of training data. In addition, Deep Reshuffle involves an expensive optimization procedure in its image generation process. Our Style Projection works in a parameter-free manner, thus leading to a more efficient, robust, and generalized style transfer performance.

3 METHOD

3.1 Arbitrary Style Transfer

The goal of arbitrary image style transfer is to stylize a content image conditioned on an arbitrary style image. To address this task, we introduce a learning-based feed-forward style transfer model. Following the standard image style transfer practice in previous literature [16, 26], we embed content and style images into content and style features with an image encoder E. As discussed above, the critical component of our style transfer model is Style Projection algorithm, which is a parameter-free feature transformation approach for proper fusion of content and style features. Based on Style Projection, the content and style features are fused and then reconstructed as a new stylized image with an image decoder D. Despite the simplicity of the model, it is effective and there is

Algorithm 1 Parameter-free Style Projection.

Require: content feature map $x \in \mathbb{R}^{C \cdot H \cdot W}$
style feature map $y \in \mathbb{R}^{C \cdot H \cdot W}$
Ensure: stylized feature map $z \in \mathbb{R}^{C \cdot H \cdot W}$
1: $\vec{x} \in \mathbb{R}^{C \cdot V} \leftarrow$ reshape x;
$\vec{y} \in \mathbb{R}^{C \cdot V} \leftarrow$ reshape y;
2: index d_x, values $\vec{x}_r \leftarrow$ sort $\vec{x}$ along V;
index d_y, values $\vec{y}_r \leftarrow$ sort $\vec{y}$ along V;
3: $\vec{z}\,[:, i] \leftarrow \vec{y}_r\,[:, d_x\,[i]], i = 1, 2, \ldots, V$;
4: $z \in \mathbb{R}^{C \cdot H \cdot W} \leftarrow$ reshape $\vec{z}$.

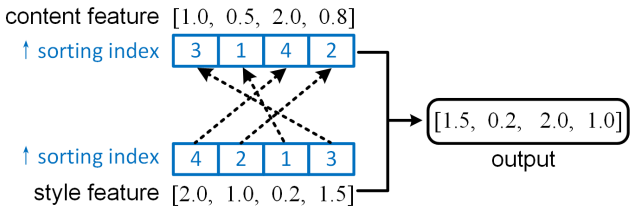


Figure 3: A simple example of Style Projection algorithm. The style vector is reordered according to the ranking index of the content vector. For instance, 0.8 is the second-smallest value in the content vector, and we thus replace it with the second-smallest value of the style vector, i.e., 1.0.

a leeway to incorporate useful tricks, for instance the semantic matching regularization of KL divergence, to further boost the style transfer performance. In the following sections, we first introduce Style Projection algorithm and then discuss the learning objectives of our style transfer model.

3.2 Style Projection Algorithm

In this work, we introduce Style Projection which is a simple, fast, yet effective algorithm for content-style fusion. Given a content feature map $x \in \mathbb{R}^{C \cdot H \cdot W}$ and a style feature map $y \in \mathbb{R}^{C \cdot H \cdot W}$, where C, H, W are channel, height, and width respectively, we firstly vectorize x and y across dimension H and W to obtain features $\vec{x} \in \mathbb{R}^{C \cdot V}$ and $\vec{y} \in \mathbb{R}^{C \cdot V}$, where $V = H \cdot W$. Then we compute rankings for elements in $\vec{x}$ and $\vec{y}$. Afterwards, we reorder $\vec{y}$ by aligning each element to its corresponding same ranked element in $\vec{x}$. This actually reorganizes the style feature $\vec{y}$ according to the sorting order of the content feature $\vec{x}$. Then we reshape the adjusted feature to get $z \in \mathbb{R}^{C \cdot H \cdot W}$, which will be treated as the input of a decoder D to generate a stylized image. The computing procedure of Style Projection is illustrated in Alg. 1. For clarification, a simple example of Style Projection is shown in Fig. 3.

Analysis on Style Projection. To better understand the effectiveness of Style Projection, we investigate the Gram matrix of images. The Gram matrix [10] can be used to evaluate texture synthesis algorithms by measuring the texture correlation between images. [14] theoretically proves that the reshuffling of style features does not alter the Gram matrices of style features, i.e., the style information is well preserved after style feature reshuffling. On the other hand, Style Projection preserves the structure of content images since the features are reordered according to the order of content features. Through Style Projection, the correlation of feature values (shape, texture, etc.) from content images and the scaling information (color patterns) from style images are well fused in a parameter-free manner.

Figure 4: An empirical study on feature shuffling mechanisms. "No Reshuffling" directly feeds style features to the decoder without any shuffling operation. "Random Reshuffling" feeds randomly shuffled style features (shuffled within each channel of feature maps) to the decoder.

To verify this claim, we conduct an empirical study on feature shuffling mechanisms and the results are shown in Fig. 4. We observe that among the three methods, only Style Projection is capable of fusing content and style properly. 'No reshuffling' of style features shows an exact reconstruction of the original style image. 'Random reshuffling' shows a repetition of random style patterns, demonstrating that the style patterns can be preserved after feature reshuffling. Only Style Projection, which reorders the style features based on content features, shows reasonable image stylization. This study reveals the effectiveness of Style Projection as it is able to preserve both content and style information during feature transformation.

3.3 The Learning of Style Transfer Model

We illustrate our style transfer model in Fig. 5. The learning objective of our style transfer model is composed of three parts, including the style loss, the content perceptual loss, and the content KL divergence loss. The style loss $\mathcal{L}_s$ is used to match the feature statistics between the style image s and the stylized image $\hat{c}$ as

$$\mathcal{L}_s = \sum_{i=1}^{N} \|\mu(E_i(s)) - \mu(E_i(\hat{c}))\|_2 + \sum_{i=1}^{N} \|\sigma(E_i(s)) - \sigma(E_i(\hat{c}))\|_2 \quad (1)$$

where μ and σ denotes the mean and standard deviation, and E_i is the intermediate output of the i-th layer of encoder E, and N is the number of encoder layers. The content perceptual loss $\mathcal{L}_p$ [18] is used to minimize the pixel-wise feature distance between the

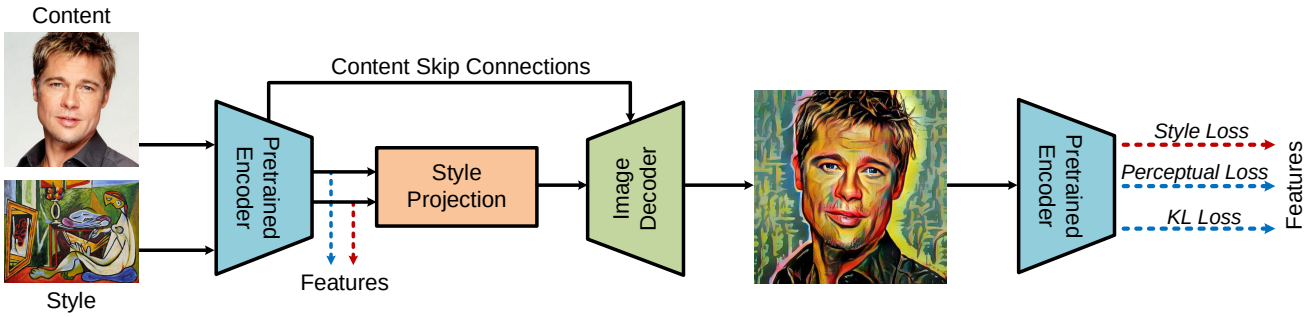


Figure 5: Our style transfer model. The inputs consist of a content image and a style image and their features are extracted by the same pre-trained image encoder. The content feature and the style feature are fused by our proposed Style Projection module and decoded by a learnable image decoder to generate the stylized image. The model is learned under the supervision of style loss, perceptual loss, and KL divergence loss, which are all computed in image feature space.

content image c and the stylized image $\hat{c}$ as

$$\mathcal{L}_p = \|E(c) - E(\hat{c})\|_2 \tag{2}$$

Semantic matching as distribution learning. Most image style transfer methods suffer from non-natural image clues in stylized results. We conjecture this is partially due to the missing of semantic information such as brightness. Inspired by the insights provided by generative models [13], we introduce a distribution matching objective, Kullback-Leibler (KL) divergence [21], into the style transfer framework to leverage more semantic information in content images, as

$$\mathcal{L}_{KL} = \mathcal{KL}\left[E(c) \,||\, E(\hat{c})\right] \tag{3}$$

where c is an input content image and $\hat{c}$ is the stylized image produced by decoder D. With the aid of KL divergence, we regularize D to generate images that contain more semantics of content images.

Overall, the complete learning objective of our model is formulated as

$$\mathcal{L} = \mathcal{L}_p + \lambda\mathcal{L}_s + \kappa\mathcal{L}_{KL} \tag{4}$$

where λ and κ are the loss weights of $\mathcal{L}_s$ and $\mathcal{L}_{KL}$, respectively. As Style Projection is a parameter-free method and the added KL divergence loss only brings negligible extra computing time, our style transfer approach is highly efficient.

4 EXPERIMENTS

4.1 Experimental Setup

Networks. We implement our style transfer model based on Pytorch framework [32]. The encoder network used in the model is the VGG-19 network [35] pre-trained on ImageNet [7], and its weights are not updated during training. The decoder network is composed of nine 'Padding-Conv-ReLU' blocks, except the last block that has no ReLU layer. Three up-sampling layers are adopted right after the 1-st, 5-th, and 7-th block to restore the input image dimension successively, where the nearest neighbor interpolation is employed for up-sampling. We do not use normalization layer in our decoder network since it will hurt the diversity of synthesized images, as demonstrated in [16].

Datasets. We adopt the training set of MS-COCO dataset [28] as the content images and that of WikiArt dataset [30] as the style images. In training, we resize all input images to the size of 512×512 and randomly crop each image to 256×256. All content and style images used for testing purpose are selected from the test set of the two datasets, and the test images are never observed by the model during training. Our encoder and decoder networks work in a fully-convolutional manner, and thus can be applied to images of arbitrary size in inferencec phase.

Learning details. We train our style transfer model for 160,000 iterations under the learning objective formulated in Eq. 4. We use an Adam optimizer [20] with an initial learning rate of 1e-4 and a learning rate decay of 5e-5. Unless otherwise specified, the loss weights λ and κ are set to 10 and 2.5, respectively. Note that the loss weights are employed to balance the style transfer and content semantics preservation. More empirical studies on the KL divergence weight κ are illustrated in the following ablation studies. We use a batch size of 8 for training, and we observe that a larger batch size can only bring a marginal performance improvement.

4.2 Comparison with State-of-the-Arts

Qualitative comparison. In Fig. 6, we show stylized images generated by different methods. The images generated by WCT have a low fidelity and we conjecture that this is due to the whitening operation which actually peels off some critical clues from the content images. AdaIN better reconstructs the content details, while there are still several 'non-natural strokes' which might be caused by only transferring the mean and standard deviation of the style features. For example, Brad Pitt's eye (see Fig. 1) and Lena's face are ill-synthesized. The images produced by the Style Swap method are too dark probably due to the loss of semantics which is caused by its way of replacing content features. Linear propagation fails to recover Lena's eyes and also fails to transfer the color of the style image into the synthesized result (e.g., Golden Gate). Our methods (last two columns) provide more pleasant results. Even using Style Projection independently, style information is properly transferred while main content is preserved. More importantly, Style Projection+Skip+KL method produces more appealing results where content details and semantics are better preserved.

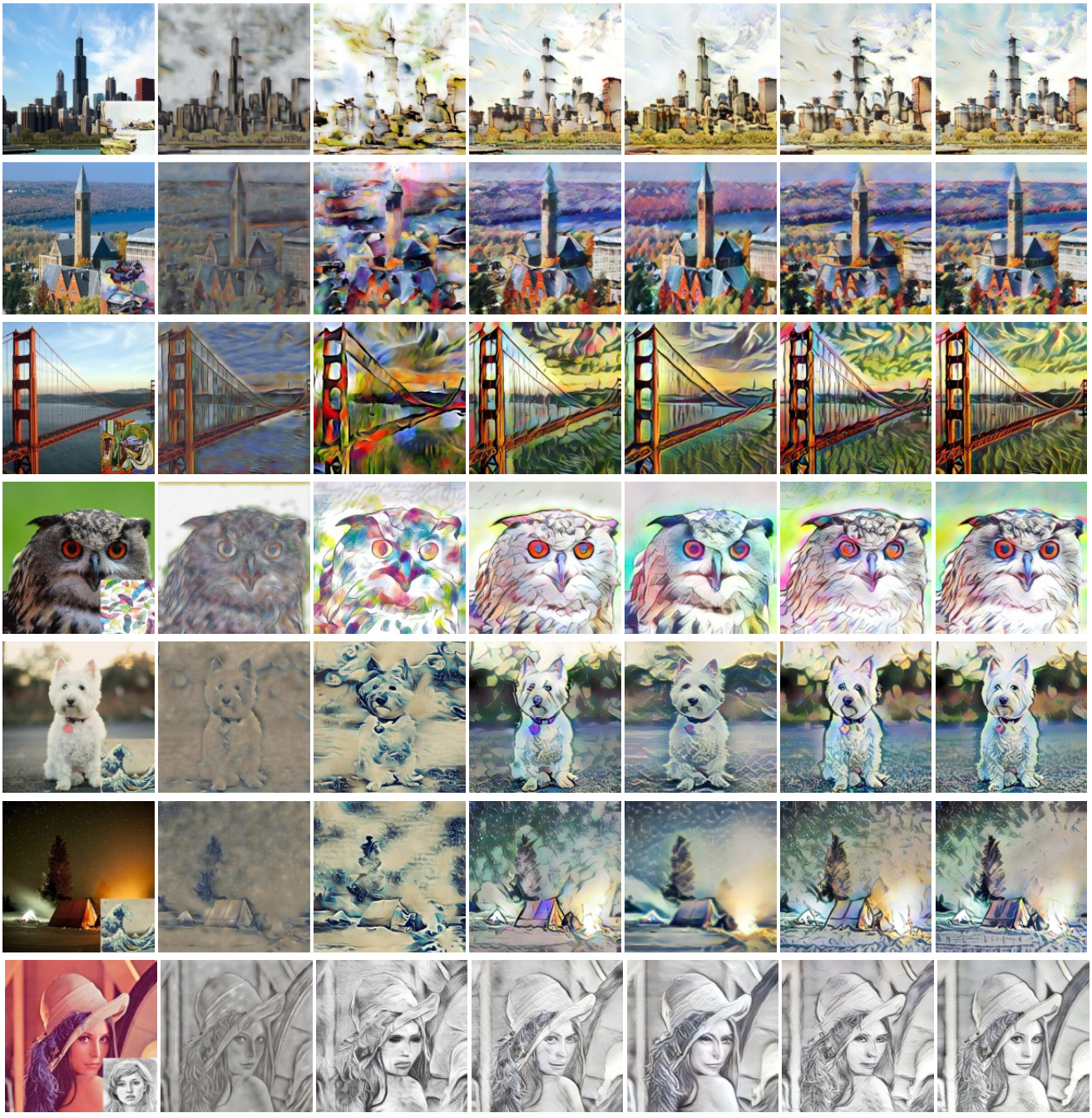


Figure 6: A qualitative comparison between the state-of-the-art content-style transformation modules. Our methods generally perform better in terms of preserving content details while properly transferring style information. Adding content skip connections and KL divergence loss to Style Projection results in more appealing images.

Quantitative evaluation. In Table 2, we quantitatively evaluate the state-of-the-art arbitrary image style transfer methods, including Style-Swap [4], WCT [26], AdaIN [16], Linear Propagation [24], Style Projection (ours), and Style Projection+skip connections+KL loss (ours). We investigate the content, style and total loss in inference. Our Style Projection shows competitive results compared with the state-of-the-art style transfer modules with respect to both style and content loss. The Style Projection+Skip+KL method

	Pairwise Comparison (%)			Method Score (%)			User Consistency (%)
	AdaIN vs. SP	AdaIN vs. Full	SP vs. Full	AdaIN	SP	Full	
Scene	36.36 / 63.64	28.79 / 71.21	40.15 / 59.85	32.58	51.89	65.53	68.94
Person	41.67 / 58.33	27.27 / 72.73	16.67 / 83.33	34.47	37.50	78.03	76.52
Total	39.02 / **60.98**	28.03 / **71.97**	28.41 / **71.59**	33.52	44.70	**71.78**	72.73

Table 1: User study on style transfer methods. "SP" denotes Style Projection and "Full" denotes Style Projection+Skip+KL."Pairwise Comparison" refers to users' selections when comparing any two of the three involved methods. "Method Score" is the percentage of times that users prefer the images synthesized by an individual method. "User Consistency" reflects the consistency among users. "Scene" and "Person" denote two categories of content images.

Method	Style	Content	Total
CNN [10]	0.90	2.51	3.41
Style-Swap [4]	4.87	2.56	7.43
WCT [26]	1.33	4.08	5.41
AdaIN [16]	0.39	2.56	2.95
Linear Propagation [24]	2.86	2.86	5.73
Style Projection	**0.38**	2.61	2.99
Style Projection + Skip	0.48	2.56	3.04
Style Projection + KL	**0.38**	2.25	2.63
Style Projection + Skip + KL	0.49	**2.09**	**2.58**

Table 2: Quantitative evaluation on images synthesized on testing set (the lower the better).

Method	Shuffling Function	Time/Image
Deep Reshuffle [14]	learnable	114 seconds
Style Projection (Ours)	parameter-free	0.068 seconds

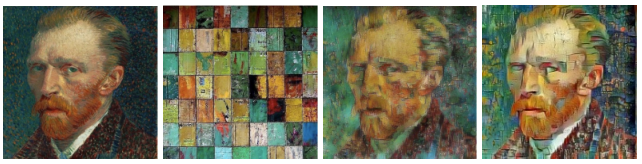


Figure 7: A comparison of shuffling-based feature transformation methods, including Deep Feature Reshuffle [14] and our proposed Style Projection.

presents more superior results than the other methods and similar style loss as AdaIN, indicating that the modules proposed in this paper, including Style Projection and KL divergence loss, work well in arbitrary style transfer tasks. We also observe that there is a trade-off between style loss and content loss. Generally, Style Projection+KL and Style Projection+Skip+KL reach a good balance in such a trade-off.

4.3 User Study

We further conduct an user study on the images synthesized by three different methods, including AdaIN, our Style Projection, and our full model (Style Projection+Skip+KL). We use 8 content images and each is associated with 3 different style images, and thus build 24 pairs of synthesized images. Each pair is generated by two of the three involved approaches, and each approach is sampled with an equal probability. The original content and style images are also provided to the users who have no style transfer knowledge. Within each pair, the two synthesized images are randomly placed. The users are asked to choose one synthesized image from each pair under three criteria: (1) whether the image is sharp and clean while its content is close to that of the content image, (2) whether the image has a similar style as the style image, (3) which image they prefer if they want to buy.

We have collected feed-backs from 33 individuals and the results are summarized in Table 1. For all the pairs of images that are produced by AdaIN and Style Projection, more users prefer images synthesized by Style Projection. Moreover, most users favor the images generated by our full model when it is compared with AdaIN. In addition, the produced images by the full model also obtain more votes than those synthesized by Style Projection only. The "method scores" and "user consistency" rates indicate that most users all favor the synthesized images of our methods. Interestingly, for the person images, more users prefer our full model (78.03%), which indicates that fidelity is deserved to be well maintained in the synthesized person images (see Fig. 1).

4.4 Study on Style Projection

Comparison between Deep Reshuffle and Style Projection. Both Deep Feature Reshuffle [14] and our Style Projection are reshuffling-based feature transformation methods. A comparison between DFS and Style Projection is illustrated in Fig. 7. The feature shuffling function in Deep Reshuffle is parametric and data-driven. Deep Reshuffle also includes an expensive optimization process in its framework, thus it is time-consuming in image generation. Our parameter-free Style Projection is able to give appealing stylized results with a high efficiency.

Trade-off between content and style in inference. We further investigate the way to control the level of stylization in Fig. 8. There is a trade-off between content preservation and style transfer by adding a parameter α to adjust the granularity of feature transformation. Specifically, the transformed feature z_α, which is the input of the decoder, is combined as follows for inference.

$$z_\alpha = \alpha z + (1 - \alpha)x \tag{5}$$

$\alpha = 0$ $\alpha = 0.33$ $\alpha = 0.67$ $\alpha = 1$ Style

Figure 8: The trade-off between content and style by tuning the style weight α of Style Projection in inference phase. Specifically, $\alpha = 0$ is equivalent to the reconstruction of the content image.

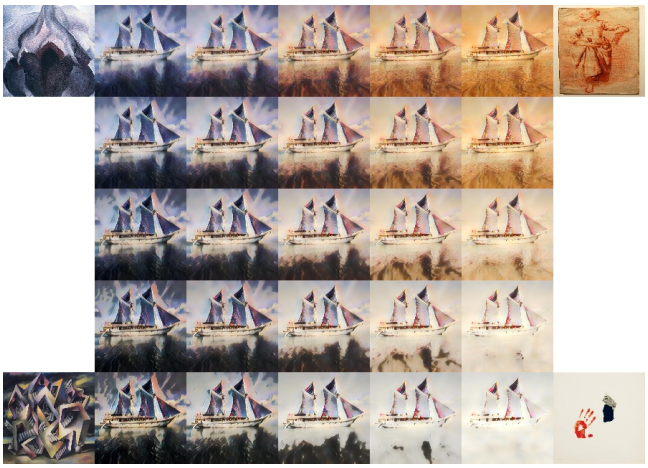


Figure 9: Style interpolation between four different styles via a weighted summation of stylized features.

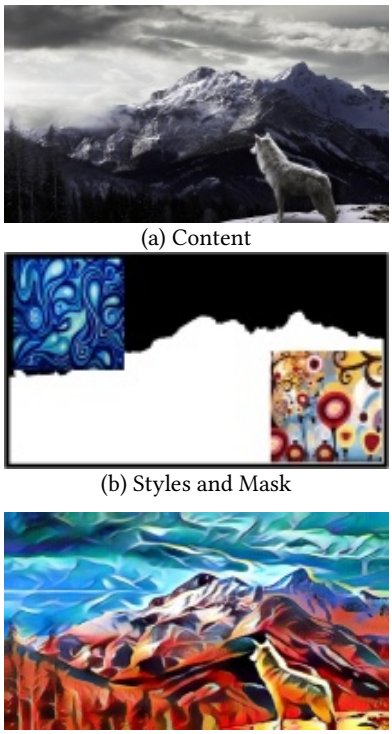
Fig. 8 shows an example of image stylization with the same model but different α in inference. All the images are clear in visual appearance and reasonable in stylization. The style is vastly transferred when α equals 1, whereas the content image will be reconstructed with no change if α is set to 0. The stylization level can be increased with the increment of α. It is important to note that α is not considered during training, and we fix $\alpha = 1$ when we compared our method with the other methods.

Style interpolation. For inference, our model is also able to stylize a content image with multiple style images. We obtain a combined feature for decoder by summing each weighted transformed feature as follows

$$z_{\text{interpolated}} = \sum_{m=1}^{M} w_m z_m \tag{6}$$

where z_m denotes the feature transformed from the m-th style feature, w_m the interpolation weight, and M the number of style images. Fig. 9 shows the style interpolation results of our method. The content is a sailboat and the four corners are style images. As expected, the style is changing gradually while the content is reasonably preserved in synthesized images.

Spatial control. Fig. 10 shows an example of spatial control where different parts of an image are stylized based on different styles. We adopt a fore/background mask to the content image, and stylize the background with the first style image, and the foreground



(a) Content

(b) Styles and Mask

(c) Ours

Figure 10: Spatial control with two different style images via a foreground/background mask.

with the second style image. In the artistic image, the style of the fore/background can be easily distinguished.

4.5 Study on Skip Connection and KL Loss

In this paper, we propose to adopt Style Projection, content skip connections, and KL loss in the style transfer framework. To better understand the role of each module, we make ablation studies on these modules and the results are shown in Fig. 11. We have the following observations:

(1) Without skip connections and KL loss, Style Projection module can decently transfer content details and style structure to a synthesized image (check SP).
(2) Using either skip connections or KL divergence contributes to content detail reconstruction (check SP, SP+KL, and SP+Skip).

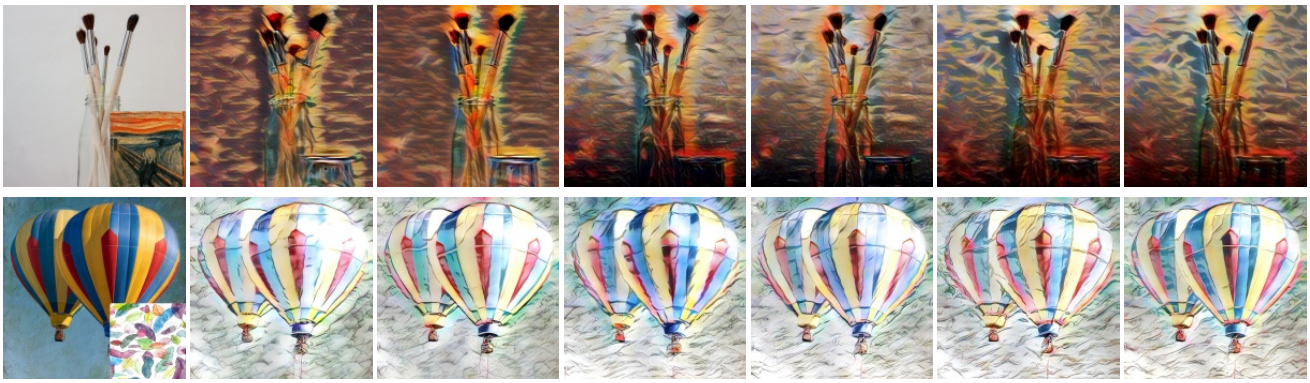


Figure 11: Ablation studies on content skip connections and KL divergence loss. AdaIN can also be benefited from content skip connections and KL divergence loss, showing a more distinct stylized image.

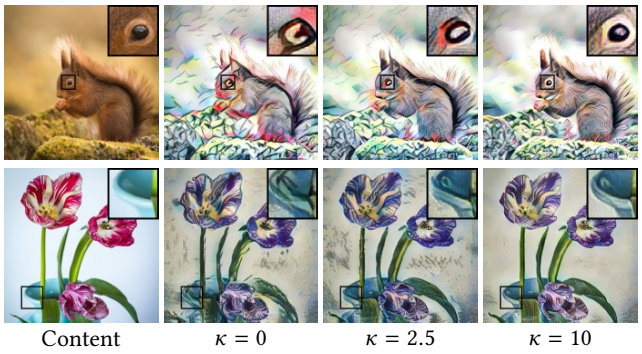


Figure 12: An empirical study on KL loss weight κ. It shows a better reconstruction of content details when κ increases. Please zoom in for the best viewing experience.

Method	256×256 pix	512×512 pix
CNN [10]	15.86	50.80
Style-Swap [4]	1.96	5.77
WCT [26]	0.689	0.997
AdaIN [16]	**0.036**	**0.064**
Avatar-Net [34]	0.248	0.356
Deep Reshuffle [14]	-	114
Ours	**0.031** / 0.004	**0.068** / 0.036
Ours+Skip+KL	**0.031** / 0.005	**0.069** / 0.036

Table 3: Time (seconds) for stylizing an image with style transfer methods. We show the performances of our methods with and without the I/O process (· / ·), respectively.

Moreover, using three modules simultaneously gives a more appealing result (check SP+Skip+KL).

(3) AdaIN [16] can also be benefited from content skip connections and KL loss (check AdaIN+Skip+KL). Style Projection is generally competitive or better than AdaIN with the same add-ons.

The KL loss weight κ can be tuned conveniently to control the level of content details preservation. We further investigate its role by conducting an empirical study in Fig. 12. A local region of an image is denoted by a black box and the zoomed-in result is placed on the top-right corner of the image. As shown in Fig. 12, with the increment of κ the content details in the stylized image are better preserved. When κ is set to a very large value (i.e., $\kappa = 10$), the content details can be well reconstructed to the content image, however the stylization effect will be sacrificed slightly.

4.6 Real-Time Style Transfer

We demonstrate that our proposed method achieves real-time inference in Table 3. The inference time is measured with a NVIDIA GTX 1080Ti GPU on the testing set, averaging over the time of producing 220 synthesized images. As can be seen, our method has a similar inference time expense to the other real-time approaches such as AdaIN. We observe that the main computing cost of our method comes from the encoder and decoder network. The introduction of Style Projection and KL divergence loss brings negligible extra computing cost. The inference time without I/O process tells that our style transfer model and its upgraded version Ours+Skip+KL are very efficient especially on low-resolution images (256×256).

5 CONCLUSION

In this paper, we have presented a real-time feed-forward model for arbitrary style transfer, including a parameter-free, fast, and universal content-style feature transformation module, Style Projection. We have also introduced the KL divergence loss into our style transfer model for a regularization of semantic consistency on content structures. We have conducted extensive experiments, including quantitative evaluation, qualitative analysis, and user study to validate the capacity and efficiency of our method on image style transfer tasks.

REFERENCES

[1] Jie An, Haoyi Xiong, Jiebo Luo, Jun Huan, and Jinwen Ma. 2019. Fast Universal Style Transfer for Artistic and Photorealistic Rendering. *arXiv preprint arXiv:1907.03118* (2019).

[2] Jie An, Haoyi Xiong, Jinwen Ma, Jiebo Luo, and Jun Huan. 2019. StyleNAS: An Empirical Study of Neural Architecture Search to Uncover Surprisingly Fast End-to-End Universal Style Transfer Networks. *arXiv preprint arXiv:1906.02470* (2019).

[3] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. 2017. Stylebank: An explicit representation for neural image style transfer. In *CVPR*. 1897–1906.

[4] Tian Qi Chen and Mark Schmidt. 2016. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337* (2016).

[5] Herbert Aron David and Haikady Navada Nagaraja. 2004. Order statistics. *Encyclopedia of Statistical Sciences* (2004).

[6] Jeremy S De Bonet. 1997. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Annual conference on computer graphics and interactive techniques*. 361–368.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*. 248–255.

[8] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. 2016. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629* (2016).

[9] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. 2017. A learned representation for artistic style. In *ICLR*.

[10] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *CVPR*. 2414–2423.

[11] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. 2017. Controlling perceptual factors in neural style transfer. In *CVPR*. 3985–3993.

[12] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. 2017. Exploring the structure of a real-time, arbitrary neural artistic stylization network. *arXiv preprint arXiv:1705.06830* (2017).

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*. 2672–2680.

[14] Shuyang Gu, Congliang Chen, Jing Liao, and Lu Yuan. 2018. Arbitrary Style Transfer with Deep Feature Reshuffle. In *CVPR*. 8222–8231.

[15] David J Heeger and James R Bergen. 1995. Pyramid-based texture analysis/synthesis. In *Annual conference on computer graphics and interactive techniques*. 229–238.

[16] Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *CVPR*. 1501–1510.

[17] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. 2019. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics* (2019).

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. 694–711.

[19] Tero Karras, Samuli Laine, and Timo Aila. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*. 4401–4410.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[22] Chuan Li and Michael Wand. 2016. Combining markov random fields and convolutional neural networks for image synthesis. In *CVPR*. 2479–2486.

[23] Chuan Li and Michael Wand. 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*. 702–716.

[24] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. 2019. Learning Linear Transformations for Fast Image and Video Style Transfer. In *CVPR*. 3809–3817.

[25] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. 2017. Diversified texture synthesis with feed-forward networks. In *CVPR*. 3920–3928.

[26] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. 2017. Universal style transfer via feature transforms. In *NIPS*. 386–396.

[27] Jing Liao, Yuan Yao, Lu Yuan, Gang Hua, and Sing Bing Kang. 2017. Visual attribute transfer through deep image analogy. *arXiv preprint arXiv:1705.01088* (2017).

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*. 740–755.

[29] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. 2017. Deep photo style transfer. In *CVPR*. 4990–4998.

[30] K Nichol. 2016. Painter by numbers, wikiart. https://www.kaggle.com/c/painter-by-numbers.

[31] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic Image Synthesis with Spatially-Adaptive Normalization. In *CVPR*. 2337–2346.

[32] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*.

[33] Javier Portilla and Eero P Simoncelli. 2000. A parametric texture model based on joint statistics of complex wavelet coefficients. *IJCV* 40, 1 (2000), 49–70.

[34] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. 2018. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *CVPR*. 8242–8250.

[35] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[36] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. 2016. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images.. In *ICML*, Vol. 1. 4.

[37] D. Ulyanov, A. Vedaldi, and V. Lempitsky. 2017. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*. 6924–6932.

[38] Guojun Yin, Bin Liu, Lu Sheng, Nenghai Yu, Xiaogang Wang, and Jing Shao. 2019. Semantics Disentangling for Text-to-Image Generation. In *CVPR*. 2327–2336.