EFANet: Exchangeable Feature Alignment Network for Arbitrary Style Transfer

Zhijie Wu,[1*] **Chunjin Song,**[1*] **Yang Zhou,**[1†] **Minglun Gong,**[2] **Hui Huang**[1†]
[1]Shenzhen University, [2]University of Guelph
{wzj.micker, songchunjin1990, zhouyangvcc, hhzhiyan}@gmail.com, minglun@uoguelph.ca

Abstract

Style transfer has been an important topic both in computer vision and graphics. Since the seminal work of Gatys et al. first demonstrates the power of stylization through optimization in the deep feature space, quite a few approaches have achieved real-time arbitrary style transfer with straightforward statistic matching techniques. In this work, our key observation is that only considering features in the input style image for the global deep feature statistic matching or local patch swap may not always ensure a satisfactory style transfer; see e.g., Figure 1. Instead, we propose a novel transfer framework, EFANet, that aims to jointly analyze and better align exchangeable features extracted from the content and style image pair. In this way, the style feature from the style image seeks for the best compatibility with the content information in the content image, leading to more structured stylization results. In addition, a new whitening loss is developed for purifying the computed content features and better fusion with styles in feature space. Qualitative and quantitative experiments demonstrate the advantages of our approach.

Introduction

A style transfer method takes a pair of images as input and synthesize an output image that preserves the content of the first image while mimicking the style of the second image. The study on this topic has drawn much attention in recent years due to its scientific and artistic values. Recently, the seminal work (Gatys, Ecker, and Bethge 2016) found that multi-level feature statistics extracted from a pre-trained CNN model can be used to separate content and style information, making it possible to combine content and style of arbitrary images. This method, however, depends on a slow iterative optimization, which limits its range of application.

Since then, many attempts have been made to accelerate the above approach through replacing the optimization process with a feed-forward neural networks (Dumoulin, Shlens, and Kudlur 2016; Johnson, Alahi, and Fei-Fei 2016; Li et al. 2017a; Ulyanov et al. 2016; Zhang and Dana 2017). While these methods can effectively speed up the stylization process, they are generally constrained to a predefined set of styles and cannot adapt to an arbitrary style specified by a single exemplar image.

Notable efforts (Chen and Schmidt 2016; Huang and Belongie 2017; Li et al. 2017b; Shen, Yan, and Zeng 2017; Sheng et al. 2018) have been devoted to solving this flexibility v.s. speed dilemma. A successful direction is to apply statistical transformation, which aligns feature statistics of the input content image to that of the style image (Huang and Belongie 2017; Li et al. 2017b; Sheng et al. 2018). However, as shown in Figure 1, the style images can be dramatically different from each other and from the content image, both in terms of semantic structures and style features. Performing style transfer through statistically matching different content images to the same set of features extracted from the style image often introduces unexpected or distorted patterns (Huang and Belongie 2017; Li et al. 2017b). Several methods (Sheng et al. 2018; Yao et al. 2019; Park and Lee 2019) conquer these disadvantages through patch swap with a multi-scale feature fusion, but may contain spatially distorting semantic structures when the local patterns from input images differ a lot.

To address the aforementioned problems, in this paper, we jointly consider both content and style images and extract a *common* style feature which is customized for this pair of images only. By maximizing the common information, our goal is to align the style features of both content and style images as much as possible. This follows the intuition that when the target style feature is compatible with the content image, a good style transfer can be probably guaranteed. Since the style feature of an image is computed from its own content information, which means they are compatible with each other within the same image, we argue that aligning the style features of the two input images could help to improve the final stylization; see the comparison of our method with & without common feature in Figure 1.

Intuitively, the common style feature we extracted bridges the gap between the input content and style images, making our method outperform existing methods in many challenging scenarios. We call the aligned style features upon the common feature as *exchangeable* style features. Experiments demonstrate that performing style transfer based

[*] Equal contribution. Order determined by coin toss.

[†] Corresponding authors.

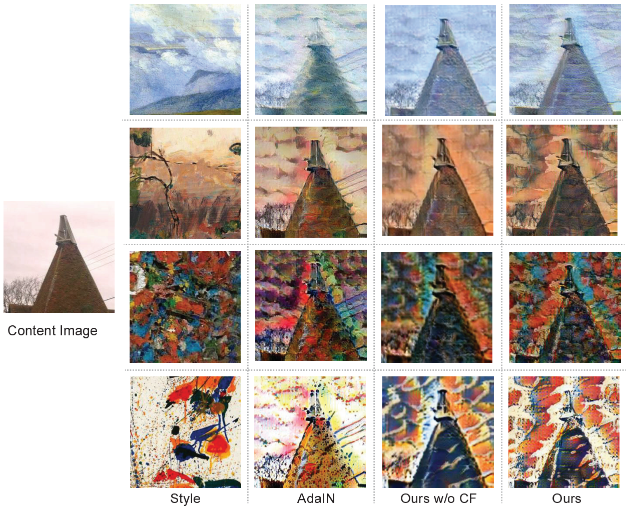


Figure 1: The existing method (AdaIN) ignores differences in style images while our approach jointly analyzes each content-style image pair and computes exchangeable style features. As a result, AdaIN and the baseline model without common features (4th column) only work well with a simple style (1st and 2nd row). When the target styles become more complex and the content-style images have different patterns/color distributions, AdaIN and the baseline model fail to capture the salient style patterns and suffer from insufficiently stylized results (color distribution and textures in 3rd & 4th row). In comparison, our model better adapts to pattern/color variation in the content image and map compatible patterns/colors in the style images accordingly.

on our exchangeable style features yields more structured results with better visual style patterns than existing approaches; see e.g., Figures 1 and 5.

To compute exchangeable style features from feature statistics of input images, a novel *Feature Exchange Block* is designed, which is inspired by the works on private-shared component analysis (Bousmalis et al. 2016; Cao et al. 2018). In addition, we propose a new *whitening loss* to facilitate the combination between content and style features by removing style patterns existed in content images. To summarize, the contributions of our work include:

- The importance of aligning style features for style transfer between two images is clearly demonstrated.
- A novel Feature Exchange Block as well as a constraint loss function are designed for the pair-wise analysis of learning common information in-between style features.
- A simple yet effective *whitening loss* is developed to encourage the fusion between content and style information by filtering style patterns in content images.
- The overall end-to-end style transfer framework can perform arbitrary style transfer in real-time and synthesize high-quality results with favored styles.

Related Work

Fast Abitrary Style Transfer Intuitively, style transfer aims at changing the style of an image while preserving its content. Recently, impressive style transfer is realized by Gatys et al. 2016 based on deep neural networks. Since then, many methods are proposed to train a single model that can transfer arbitrary styles. Here we only review the related works on arbitrary style transfer and refer the readers to (Jing et al. 2017) for a comprehensive survey.

Chen et al. 2016 realize the first fast neural method by matching and swapping local patches between the intermediate features of content and style images, which is thus called Style-Swap. Then Huang et al. 2017 propose an adaptive instance normalization (AdaIN) to explicitly match the mean and variance of each feature channel of the content image to those of the style image. Li et al. 2017b further apply whitening and coloring transform (WCT) to align the correlations between the extracted deep features. Sheng et al. 2018 develop Avatar-Net to combine local and holistic style pattern transformation, achieving better stylization regardless of the domain gap. Very recently, AAMS (Yao et al. 2019) tries to transfer the multi-stroke patterns by introducing self-attention mechanism. Meanwhile, SANet (Park and Lee 2019) promotes Avatar-Net by learning a similarity matrix and flexibly matching the semantically nearest style features onto the content features. And Li et al. 2019 speeds up WCT with a linear propagation module. To improve the generalization ability, Song et al. 2019 evaluate errors in the synthesized results and correct them accordingly in an iterative manner. The above methods, however, all achieve stylization by a straightforward statistic matching or local patch matching and ignore the gaps between input features, which may not be able to adapt to the unlimited variety.

In this paper, we still follow the holistic alignment with respect to feature correlations. The key difference is that before applying style features, we jointly analyze the similarities between the style features of content and style images. Thus these style features can be aligned accordingly, which enables the style features to match the content images more flexibly and improves the final compatibility level between target content and style features significantly.

Feature Disentanglement Learning disentangled representation aims at separating the learned internal representation into factors of data variations (Whitney 2016). It improves the re-usability and interpretation of the model, which is useful for e.g., domain adaptation (Bousmalis et al. 2016; Cao et al. 2018). Recently, several concurrent works (Lee et al. 2018; Huang et al. 2018; Gonzalez-Garcia, van de Weijer, and Bengio 2018; Ma et al. 2018) have been proposed for multi-modal image-to-image translation. They map the input images into one common feature space for content representation and two unique feature spaces for styles. Yi et al. 2018 design BranchGAN to achieve scale-disentanglement in image generation. Wu et al. 2019 advance 3D shape generation by disentangling geometry and structure info. For style transfer, some efforts (Zhang, Zhang, and Cai 2018; Zhang et al. 2018) are also made to

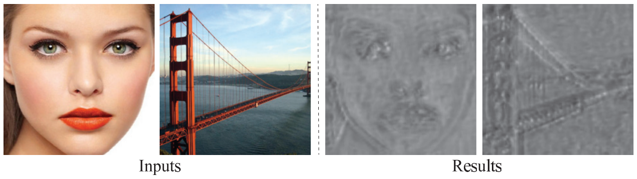


Figure 2: Images decoded from whitened features. The results on the right are rescaled for better visualization. The whitened features still keep spatial structures but various style patterns are removed.

separate a representation of one image into the content and style. Different from the mentioned methods, we perform feature disentanglement only on style features of the input image pair. A common component is extracted, and is used to compute the exchangeable style features for style transfer.

Developed Framework

Following (Gatys, Ecker, and Bethge 2016), we consider the deep feature extracted by the network pretrained on large dataset as the content representation for an image, and the feature correlation at a given layer as the style information. By fusing the content feature with a new target style feature, we can synthesize a stylized image.

The overall goal of our framework is to align the style features between the style and content image pair, such that the style feature from one image can better match the content of the other image, resulting in a better stylization. To achieve that, a key module of Feature Exchange block is proposed to jointly analyze the style features of the two input images. A common feature is disentangled to encode the shared components between the style features. Then guided by the common feature, the target style features are aligned to be more similar to each other and thus be more compatible with each other's contents.

Exchangeable Feature for Style Transfer

As illustrated in Figure 3(a), our framework mainly consists of three parts: one encoder, several EFANet modules ($\Omega(\cdot)$) and one decoder for generating the final images. We denote f_c^i and f_s^i, $i \in \{1, ..., L\}, L = 4$ as the feature maps outputted by the $relu\_i$ layer of the pre-trained VGG encoder, which correspond to content and style images (I_c and I_s) respectively. We equip multi-scale style adaption strategy to boost the stylization. Specifically, in the bottleneck of encoder-decoder architecture, starting from f_c^L and f_s^L, different EFANet modules are applied to progressively fuse the styles from input images into the corresponding decoded features in a coarse-to-fine manner as $f_{cs}^i = \Omega(\hat{f}_{cs}^i, f_s^i)$. The f_{cs}^i indicates a decoded stylized feature and $\hat{f}_{cs}^i = u(f_{cs}^{i+1})$, where $u(\cdot)$ is an upsampling operator and the superscript i denotes the i-th scale. Note that, initially we set $\hat{f}_{cs}^L = f_c^L$ and apply the superscript j to indicate the j-th style vector of a Gram matrix in the following paragraphs.

In Figure 3(b), given f_s and $\hat{f}_{cs}$ as inputs, we first compute two Gram matrices across the feature channels as the raw style representations and denote them as G_s and $G_{cs} \in R^{C \times C}$. The C indicates the channel number for f_s and $\hat{f}_{cs}$. In order to preserve more style details in output results and reduce computation burden, we process only a part of style information at a time and represent G_s and G_{cs} as two lists of style vectors, e.g. $G_s = \{g_s^1, g_s^2, ..., g_s^C\}$ and $G_{cs} = \{g_{cs}^1, g_{cs}^2, ..., g_{cs}^C\}$. Each style vector, g_s^j and g_{cs}^j, compactly encodes the mutual relationships between the j-th channel and the whole feature map. Then each corresponding style vector pair (g_s^j, g_{cs}^j) is processed using one Feature Exchange block. Accordingly a common feature g_{com}^j and two unique feature vectors for decoded information (as content) and style, g_{cu}^j and g_{su}^j, can be disentangled.

Guided by g_{com}^j, the style features are aligned in the following manner: we first concatenate g_{com}^j with the raw style vectors g_s^j and g_{cs}^j respectively. Then they are sent into fully connected layers individually, yielding the aligned style vectors $\tilde{g}_s^j$ and $\tilde{g}_{cs}^j$. We call them as exchangeable style features since each of them can be used easily to adapt its style to the target image. Then we stack the style vectors $\{\tilde{g}_s^j\}$ and $\{\tilde{g}_{cs}^j\}$ into two matrices, $\tilde{G}_s$ and $\tilde{G}_{cs}$, for later fusion as:

$$\tilde{G}_s = [\tilde{g}_s^1, \tilde{g}_s^2, ..., \tilde{g}_s^C], \quad \tilde{G}_{cs} = [\tilde{g}_{cs}^1, \tilde{g}_{cs}^2, ..., \tilde{g}_{cs}^C].$$

Inspired by the whitening operation of WCT (Li et al. 2017b), we also assume that better stylization results can be achieved when the channels of target content features are highly uncorralated before content-style fusion. Specifically, in our case the whitening operation can be regarded as a filter function, which removes style info from the content feature according to its own style vector. Hence after feature alignment, to better transfer a new style to an image, we first use the exchangeable style feature to purify its own content feature through a fusion as:

$$\tilde{f}_{cs} = \Psi_{whi}(\hat{f}_{cs}, \tilde{G}_{cs}) = \hat{f}_{cs} \cdot W_{whi} \cdot \tilde{G}_{cs},$$

where $\Psi_{whi}(\cdot)$ and W_{whi} indicates the fusion operation and a learnable matrix respectively (Zhang, Zhang, and Cai 2018; Zhang and Dana 2017). We also develop a *whitening loss* to encourage the removal of correlations between different channels; see Figure 2 as a validating example. The details of whitening loss are discussed in the Loss Function section below.

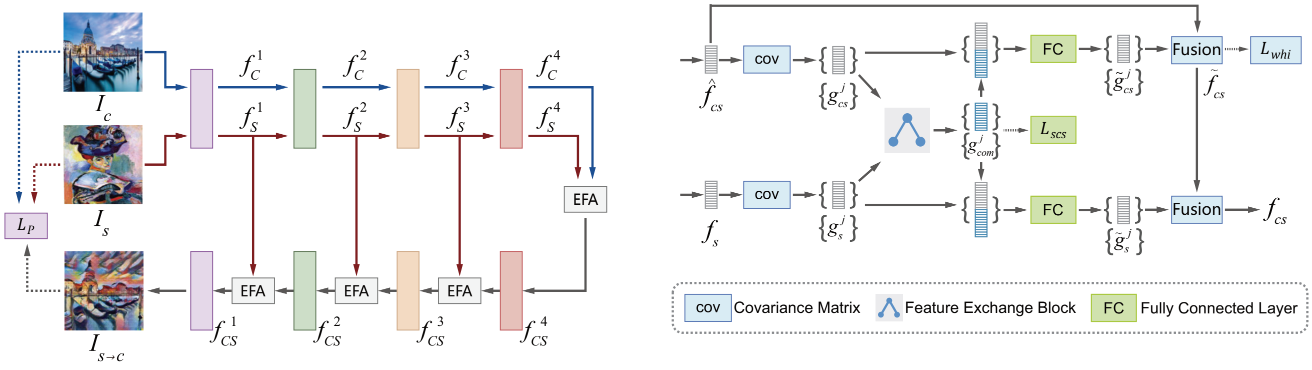
Finally, we exchange the aligned style vectors and fuse them with the purified content features as:

$$f_{cs} = \Psi_{fusion}(\tilde{f}_{cs}, \tilde{G}_s) = \tilde{f}_{cs} \cdot W_{fusion} \cdot \tilde{G}_s.$$

Then the f_{cs} will be propagated to receive style information at finer scales or decoded to output stylized images. The decoder is trained to learn the inversion from the fused feature map to image space, and hereby, style transfer is eventually achieved for both input images. Note that the resulting $I_{s \to c}$ denotes the stylization image that transfers style in I_s to I_c,

Feature Exchange Block

According to Bousmalis et al. (2016), explicitly modeling the unique information would help improve the extraction of the shared component. To adapt this idea for our exchangeable style features, a Feature Exchange block is proposed to



(a)

(b)

Figure 3: (a) Architecture overview. The input image pair I_c and I_s, goes through the pre-trained VGG encoder to extract feature maps $\{f_c^i\}$ and $\{f_s^i\}, i \in \{1, ..., L\}, L = 4$. Then, starting from f_c^L and f_s^L, different EFANet modules are applied to progressively fuse styles into corresponding decoded features for final stylized images. (b) The architecture of EFANet module. Given $\hat{f}_{cs}$ and f_s as inputs, we compute two Gram matrices as the raw styles and then represent them as two lists of feature vectors $\{g^j_{cs}\}$ and $\{g^j_s\}$. Each corresponding style vector pair (g^j_{cs} and g^j_s) is fed into the newly proposed Feature Exchange Block and a common feature vector g^j_{com} is extracted via the joint analysis. We concatenate g^j_{com} with g^j_{cs} and g^j_s respectively to learn two exchangeable style feature $\tilde{g}^j_{cs}$ and $\tilde{g}^j_s$. $\tilde{g}^j_{cs}$ is used for the content feature purification, which will be further fused with $\tilde{g}^j_s$, outputting f_{cs}. Finally f_{cs} will be either propagated for finer-scale information or decoded into stylized images $I_{s\to c}$.

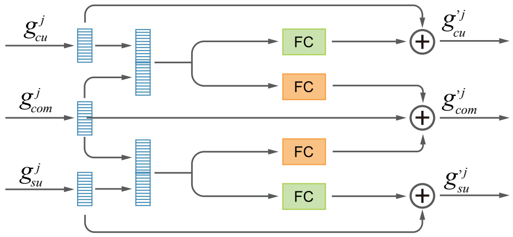


Figure 4: Architecture of a Feature Exchange Block. Here ⊕ denotes element-wise addition. Each block has three inputs, one common feature g^j_{com} and two unique features for content g^j_{cu} and style g^j_{su} images, respectively. Note that for the first block, g^j_{cu} and g^j_{su} are initialized with g^j_{cs} and g^j_s respectively, and g^j_{com} are initialized with their combination. Then the block allows common feature to interact with unique features and outputs refined results g'^j_{com}, g'^j_{cu}, and g'^j_{su}.

jointly analyze the style features of both input images and model their inter-relationships, based on which we explicitly update the common feature and two unique features for the disentanglement. Figure 4 illustrates the detailed architecture, where the unique features, g^j_{cu} and g^j_{su}, are first initialized with g^j_{cs} and g^j_s respectively and the g^j_{com} with their combination. Then they are updated by the learned residual features. Using residual learning is to facilitate gradient propagation during training and convey messages so that each input feature can be directly updated. This property allows us to chain any number of Feature Exchange blocks in a model, without breaking its initial behavior.

As shown in Figure 4, there are two shared fully-connected layers inside each block. To be specific, the disentangled features are updated as:

$$\begin{aligned} g'^j_{com} &= \Theta_{com}([g^j_{cu}, g^j_{com}]) + \Theta_{com}([g^j_{su}, g^j_{com}]) + g^j_{com}, \\ g'^j_{cu} &= \Theta_{uni}([g^j_{cu}, g^j_{com}]) + g^j_{cu}, \end{aligned}$$

where $\Theta_{com}(\cdot)$ and $\Theta_{uni}(\cdot)$ denote the fully-connected layers to output residuals for the common features and unique features respectively. $[\cdot, \cdot]$ indicates a concatenation operation. We can update g^j_{su} in a similar way.

By doing so, the feature exchange blocks enable g^j_{com} and g^j_{cu} (or g^j_{su}) to interact with each other by modelling their dependencies and thus to be refined to the optimal.

To make sure the feature exchange block conduct proper disentanglement, a constraint on the disentangled feature is added following Bousmalis et al. (2016). First, g^j_{com} should be orthogonal to both g^j_{cu} and g^j_{su} as much as possible. Meanwhile, it should let us be able to reconstruct g^j_s and g^j_{cs} based on the finally disentangled features. Therefore, a feature exchange loss can be defined as:

$$L^j_{ex} = g^j_{com} \cdot g^j_{cu} + g^j_{com} \cdot g^j_{su} + \|g^j_{cs} - \bar{g}^j_{cs}\|_1 + \|g^j_s - \bar{g}^j_s\|_1,$$

where $\bar{g}^j_{cs}$ is the reconstructed style vector by feeding the sum of g^j_{com} and g^j_{cu} into a fully connected layer. $\bar{g}^j_s$ is the reconstruction from g^j_{com} and g^j_{su}. Note that this fully connected layer for reconstruction is only valid in training stage, and L^j_{ex} is only computed with the final output of the feature exchange block. And we use only one feature exchange block in each EFANet module.

Finally, to maximize the common information, we also penalize the amount of unique features. Thus the final loss function for the common feature extraction is:

$$L_{com} = \sum_{j=1}^{C} L^j_{ex} + \lambda^{uni}(\|g^j_{cu}\|^2 + \|g^j_{su}\|^2),$$

where $\|\cdot\|$ denotes L_2 norm of a vector, and λ^{uni} is set to 0.0001 in all our experiments.

Loss Function for Training

As illustrated in Figure 3, three different types of losses are computed for each input image pair. The first one is perceptual loss (Johnson, Alahi, and Fei-Fei 2016), which is used to evaluate the stylized results. Following previous work (Huang and Belongie 2017; Sheng et al. 2018), we employ a VGG model (Simonyan and Zisserman 2014) pre-trained on ImageNet (Deng et al. 2009) to compute the perceptual content loss:

$$L_p^c = \|E(I_c) - E(I_{s\to c})\|_2 \, ,$$

and style loss:

$$L_p^s = \sum_{i=1}^{L} \left\|G^i(I_s) - G^i(I_{s\to c})\right\|_2,$$

where $E(\cdot)$ denotes the VGG-based encoder and $G^i(\cdot)$ represents a Gram matrix for features extracted at i-th scale in the encoder module. As mentioned before, we set $L = 4$.

The second one is the *whitening loss*, which is used to remove style information in target content images at training stages. According to Li et al. (2017b), after whitening operation, $\tilde{f}_{cs} \cdot (\tilde{f}_{cs})^{\mathrm{T}}$ should equal the identity matrix. Thus we define the *whitening loss* as:

$$L_{whi} = \|\tilde{f}_{cs} \cdot (\tilde{f}_{cs})^{\mathrm{T}} - I\|_2$$

where I denotes the identity matrix. By doing so, we can encourage feature map $\tilde{f}_{cs}$ to be as uncorrelated as possible.

The third one is the common feature loss, L_{com}, defined previously for a better feature disentanglement.

Note that, for both L_{whi} and L_{com}, we sum up the losses over all scales, e.g. $L_{whi} = \sum_{i=1}^{L} L_{whi}^i$ and $L_{com} = \sum_{i=1}^{L} L_{com}^i$. The superscript i here indicates losses computed at i-th scale, where $i \in \{1, .., L\}$. To summarize, the full objective function of our proposed network is:

$$L_{total} = \lambda^{pc} L_p^c + \lambda^{ps} L_p^s + \lambda^{whi} L_{whi} + \lambda^{com} L_{com},$$

where the four weighting parameters are respectively set as 1, 7, 0.1 and 5 through out the experiments.

Implementation Details

We implement our model with Tensorflow (Abadi et al. 2016). In general, our framework consists of an encoder, several EFANet modules and a decoder. Similar to prior work (Huang and Belongie 2017; Sheng et al. 2018), we use the VGG-19 model (Simonyan and Zisserman 2014) (up to relu4_1) pre-trained on ImageNet (Deng et al. 2009) to initialize the fixed encoder. For the decoder, after the fusion of style and content features, two residual blocks are used, followed by upsampling operations. Nearest-neighbor upscaling plus convolution strategy is used to reduce artifacts in the upsampling stage (Odena, Dumoulin, and Olah 2016). We choose Adam optimizer (Kingma and Ba 2014) with a batch size of 4 and a learning rate of 0.0001, and set the decay rates by default for 150000 iterations.

Place365 database (Zhou et al. 2014) and WiKiArt dataset (Nichol 2016) are used for content and style images respectively, following (Sanakoyeu et al. 2018). During training, we resize the smaller dimension of each image to 512 pixels with the original image ratio. Then we train our model with randomly sampled patches of size 256 × 256. Note that in the testing stage, both the content and style images can be of any size.

Experimental Results

Comparison with Existing Methods We compare our approach with six state-of-the-art methods for arbitrary style transfer: AdaIn (Huang and Belongie 2017), WCT (Li et al. 2017b), Avatar-Net (Sheng et al. 2018), AAMS (Yao et al. 2019), SANet (Park and Lee 2019) and Li et al. (Li et al. 2019). For the compared methods, publicly available codes with default configurations are used for a fair comparison.

Results of qualitative comparisons are shown in Figure 5. For the holistic statistic matching pipelines, AdaIN (Huang and Belongie 2017) can achieve arbitrary style transfer in real-time. However, it does not respect semantic information and sometimes generates less stylized results with color distribution different from the style image (see row 1 & 3). WCT (Li et al. 2017b) improves the stylization a lot but often introduces distorted patterns. As shown in rows 3 & 4, it sometimes produces messy and less-structured images. Li et al. 2019 proposes a linear propagation module and achieves the fastest transfer among all the compared methods. But it often gets stuck into the instylization issuses and can not adapt the compatible style patterns or color variations to results (row 1 & 3).

Then Avatar-Net (Sheng et al. 2018) improves over the holistic matching methods by adapting more style details to results with a feature decorating module, but it also blurs the semantic structures (rows 3) and sometimes distorts the salient style patterns (see rows 1 & 5). While AAMS (Yao et al. 2019) stylizes images with multi-stroke style patterns, similar to Avatar-Net, it still suffers from the structure distortion issues (row 3) and introduces unseen dot-wise artifacts (row 2 & 5). It also fails to capture the patterns presented in style image (row 5). In order to match the semantically nearest style features onto the content features, SANet (Park and Lee 2019) shares the similar spirits with Avatar-Net but employs a style attention module in a more flexible way. Thus it might still blur the content structures (row 3) and directly copy some semantic patterns in content images to stylization results (e.g. the eyes in row 1, 2 & 3). Due to the local patch matching, SANet also distorts the presented style patterns and fails to reserve the texture consistency (row 5).

In contrast, our approach achieves more favorable performance. The alignment on style features allows our model to better match the regions in content images with patterns in style images. The target style textures can be adaptively transferred to the content images, manifesting superior texture detail (last row) and richer color variation (2nd row). Compared to most methods, our approach can also generate more structured results while the style pattern, like brush strokes, is preserved well (3rd row).

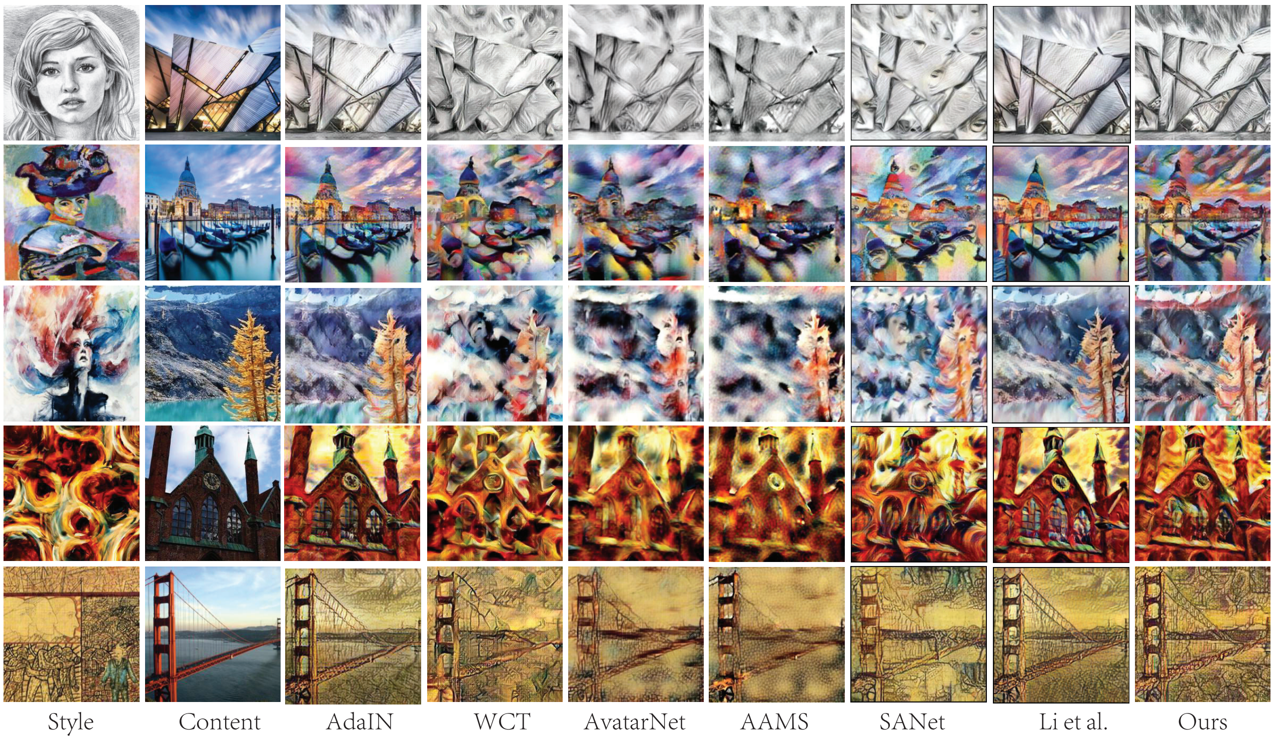


Figure 5: Comparison with results from different methods. Note that the proposed model generates images with better visual quality while the results of other baselines have various artifacts; see text for detailed discussions.

Table 1: Quantitative comparison over different models on perceptual (content & style) loss, preference score of user study and running time. Note that all the results are averaged over 100 test images except the preference score. The $Ours^*$ denotes a model equiped with single-scale strategy.

Loss	AdaIN	WCT	Avatar-Net	AAMS	SANet	Li et al.	Ours w/o CF	Ours*	Ours
Content (L_c)	14.4226	19.5318	16.8482	17.1321	23.3074	18.7288	16.3763	16.8600	16.5927
Style (L_s)	40.5989	27.1998	31.1532	34.7786	29.7760	37.3573	22.6713	24.9123	14.8582
Preference/%	0.110	0.155	0.150	0.137	0.140	0.108	-	-	0.200
Time/sec	0.0192	0.4268	0.9258	1.1938	0.0983	0.0071	0.0227	0.0208	0.0234

Assessing style transfer results could be subjective. We thus conduct two quantitative comparisons, which are reported in first 2 rows of Table 1. We first compares different methods in terms of perceptual loss. This evaluation metrics contain both content and style terms which have been used in previous approaches (Huang and Belongie 2017). It is worth noting that our approach does not minimize perceptual loss directly since it is only one of the three types of losses we use. Nevertheless, our model achieves the lowest perceptual loss among all feed-forward models, with style loss being the lowest and content loss only slightly higher than AdaIN. This indicates our approach favors fully stylized results over results with high content fidelity.

We then conduct a user study to evaluate the visual preference of the six methods. 30 content images and 30 style images are randomly selected from the test set and 900 stylization results are generated for each method. Then results of the same stylization are randomly chosen for a participant who is asked to vote for the method that achieves the best stylization. Each participant is asked to do 20 rounds of comparison. The stylized results from different methods are exhibited in a random order. Thus we collect 600 votes from 30 subjects. The average preference scores of different methods are reported in Column 4 of Table 1, which shows our method obtains the highest score.

Table 1 also lists the running time of our approach and various state-of-the-art baselines. All results are obtained with a 12G Titan V GPU and averaged over 100 256 × 256 test images. Generally speaking, existing patch based network approaches are known to be slower than the holistic matching methods. Among all the approaches, Li et al. achieves the fastest stylization with a linear propagation module. Our full model equiped with multi-scale strategy slightly increases the computation burden but are still comparable to AdaIN, thus achieving style transfer in real-time.

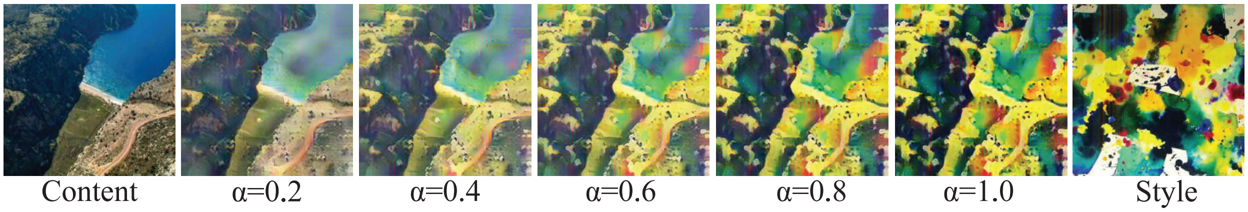


Figure 6: Balance between content and style. At testing stage, the degree of stylization can be controlled using parameter α.

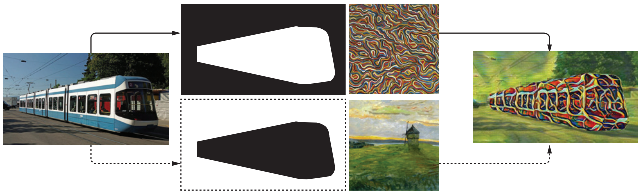


Figure 7: Application for spatial control. Left: content image. Middle: style images with masks to indicate target regions. Right: synthesized result.

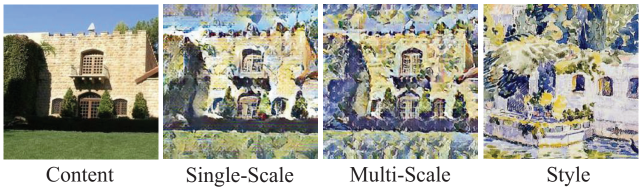


Figure 8: Ablation study on multi-scale strategy. By fusing the content and style in multi-scales, we can enrich the local and global style patterns for stylized images.

Ablation Study Here we respectively evaluate the impacts of common feature learning, the proposed *whitening loss* on content feature, and the multi-scale framework.

Common feature disentanglement during joint analysis plays a key role in our approach. Its importance can be evaluated by removing the Feature Exchange block and disabling the feature exchange loss, which prevents the network to learn exchangeable features. As shown in Figure 1, for the ablated model without common features, the color distribution and texture patterns in the result image no longer mimic the target style image. Visually, our full model yields a much more favorable result. We also compares the perceptual losses over 100 test images for both the baseline model (i.e. our model without common features) and our full model. As reported in Table 1, the style loss of our full model is significantly improved over the baseline, demonstrating the effectiveness of common features.

To verify the effect of *whitening operation* functioned on content features, we remove learnable matrices W_{whi} at all scales to see how the performance changes. As shown in Figure 9, without the purified operation and *whitening loss*, the baseline model blurs the overall contours with yellow blobs. In constrast, our full model better matches the target style to the content image and preserves the spatial structures & style pattern consistency, yielding more visually pleasing results. This proves that the proposed operation enables the content features to be more compatible with the target styles.

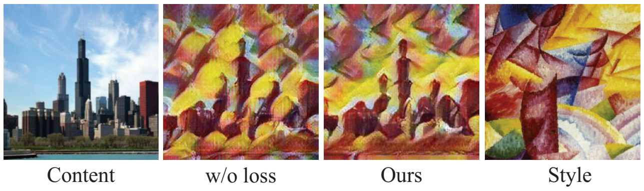


Figure 9: Ablation study on *whitening loss*. With the proposed loss, clearer content contours and better style pattern consistency are achieved.

The multi-scale strategy is evaluated by replacing the full model with an alternative model that only fuses content and style at $relu\_4$ layer while fixing the other parts. The comparison shown in Figure 8 demonstrates that the multi-scale strategy is more successful in capturing the salient style patterns, leading to better stylization results.

Applications We demonstrate the flexibility of our model using two applications. All these tasks are completed with the same trained model without any further fine-tuning.

Being able to adjust the degree of stylization is a useful feature. In our model, this can be achieved by blending between the stylized feature map f_{cs} and the VGG-based feature f_c before feeding to the decoder, which is:

$$F = (1 - \alpha) \cdot f_c + \alpha \cdot f_{cs}.$$

By definition, the network outputs the reconstructed image $I_{c \to c}$ when $\alpha = 0$, the fully stylized image $I_{s \to c}$ when $\alpha = 1$, and a smooth transition between the two when α is gradually changed from 0 to 1; see Figure 6.

In Figure 7, we present our model's ability for applying different styles to different image regions. Masks are used to specify the correspondences between different content image regions and the desired styles. Pair-wise exchangeable feature extraction only consider the masked regions when applying a given style, helping to achieve optimal stylization effect for individual regions.

Conclusions

In this paper, we have presented a novel framework, EFANet, for transferring an arbitrary style to a content image. By analyzing the common style feature from both inputs as a guider for alignment, exchangeable style features are extracted. Better stylization can be achieved for the content image by fusing its purified content feature with the aligned style feature from the style image. Experiments show that our method significantly improves the stylization performance over the prior state-of-the-art methods.

Acknowledgement

This work was supported in parts by NSFC (61861130365, 61761146002), GD Higher Education Innovation Key Program (2018KZDXM058), GD Science and Technology Program (2015A030312015), Shenzhen Innovation Program (KQJSCX20170727101233642), LHTD (20170003), and Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University.

References

Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, 265–283.

Bousmalis, K.; Trigeorgis, G.; Silberman, N.; Krishnan, D.; and Erhan, D. 2016. Domain separation networks. In *NIPS*.

Cao, J.; Katzir, O.; Jiang, P.; Lischinski, D.; Cohen-Or, D.; Tu, C.; and Li, Y. 2018. Dida: Disentangled synthesis for domain adaptation. *arXiv preprint arXiv:1805.08019*.

Chen, T. Q., and Schmidt, M. 2016. Fast patch-based style transfer of arbitrary style. *CoRR* abs/1612.04337.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Dumoulin, V.; Shlens, J.; and Kudlur, M. 2016. A learned representation for artistic style. *CoRR* abs/1610.07629.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2414–2423.

Gonzalez-Garcia, A.; van de Weijer, J.; and Bengio, Y. 2018. Image-to-image translation for cross-domain disentanglement. In *NIPS*.

Huang, X., and Belongie, S. J. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. *2017 IEEE International Conference on Computer Vision (ICCV)* 1510–1519.

Huang, X.; Liu, M.-Y.; Belongie, S. J.; and Kautz, J. 2018. Multimodal unsupervised image-to-image translation. *CoRR* abs/1804.04732.

Jing, Y.; Yang, Y.; Feng, Z.; Ye, J.; Yu, Y.; and Song, M. 2017. Neural style transfer: A review. *arXiv preprint arXiv:1705.04058*.

Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 694–711. Springer.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Lee, H.-Y.; Tseng, H.-Y.; Huang, J.-B.; Singh, M.; and Yang, M.-H. 2018. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 35–51.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M.-H. 2017a. Diversified texture synthesis with feed-forward networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition* 266–274.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M.-H. 2017b. Universal style transfer via feature transforms. In *NIPS*.

Li, X.; Liu, S.; Kautz, J.; and Yang, M.-H. 2019. Learning linear transformations for fast image and video style transfer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ma, L.; Jia, X.; Georgoulis, S.; Tuytelaars, T.; and Van Gool, L. 2018. Exemplar guided unsupervised image-to-image translation with semantic consistency. *arXiv preprint arXiv:1805.11145*.

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

Odena, A.; Dumoulin, V.; and Olah, C. 2016. Deconvolution and checkerboard artifacts. *Distill*.

Park, D. Y., and Lee, K. H. 2019. Arbitrary style transfer with style-attentional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sanakoyeu, A.; Kotovenko, D.; Lang, S.; and Ommer, B. 2018. A style-aware content loss for real-time hd style transfer. *CoRR* abs/1807.10201.

Shen, F.; Yan, S.; and Zeng, G. 2017. Meta networks for neural style transfer. *CoRR* abs/1709.04111.

Sheng, L.; Lin, Z.; Shao, J.; and Wang, X. 2018. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. 8242–8250.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *CoRR* abs/1409.1556.

Song, C.; Wu, Z.; Zhou, Y.; Gong, M.; and Huang, H. 2019. Etnet: Error transition network for arbitrary style transfer.

Ulyanov, D.; Lebedev, V.; Vedaldi, A.; and Lempitsky, V. S. 2016. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*.

Whitney, W. 2016. Disentangled representations in neural models. *arXiv preprint arXiv:1602.02383*.

Wu, Z.; Wang, X.; Lin, D.; Lischinski, D.; Cohen-Or, D.; and Huang, H. 2019. Sagnet: Structure-aware generative network for 3d-shape modeling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2019)* 38(4):91:1–91:14.

Yao, Y.; Ren, J.; Xie, X.; Liu, W.; Liu, Y.-J.; and Wang, J. 2019. Attention-aware multi-stroke style transfer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yi, Z.; Chen, Z.; Zhang, H.; Huang, X.; and Gong, M. 2018. Branched generative adversarial networks for multi-scale image manifold learning.

Zhang, H., and Dana, K. J. 2017. Multi-style generative network for real-time transfer. *CoRR* abs/1703.06953.

Zhang, R.; Tang, S.; Li, Y.; Guo, J.; Zhang, Y.; Li, J.; and Yan, S. 2018. Style separation and synthesis via generative adversarial networks. In *2018 ACM Multimedia Conference on Multimedia Conference*, 183–191. ACM.

Zhang, Y.; Zhang, Y.; and Cai, W. 2018. Separating style and content for generalized style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8447–8455.

Zhou, B.; Lapedriza, À.; Xiao, J.; Torralba, A.; and Oliva, A. 2014. Learning deep features for scene recognition using places database. In *NIPS*.