A Style-Aware Content Loss for Real-time HD Style Transfer

Artsiom Sanakoyeu[*], Dmytro Kotovenko[*], Sabine Lang, and Björn Ommer

Heidelberg Collaboratory for Image Processing,
IWR, Heidelberg University, Germany
`firstname.lastname@iwr.uni-heidelberg.de`

Abstract. Recently, style transfer has received a lot of attention. While much of this research has aimed at speeding up processing, the approaches are still lacking from a principled, art historical standpoint: a style is more than just a single image or an artist, but previous work is limited to only a single instance of a style or shows no benefit from more images. Moreover, previous work has relied on a direct comparison of art in the domain of RGB images or on CNNs pre-trained on ImageNet, which requires millions of labeled object bounding boxes and can introduce an extra bias, since it has been assembled without artistic consideration. To circumvent these issues, we propose a style-aware content loss, which is trained jointly with a deep encoder-decoder network for real-time, high-resolution stylization of images and videos. We propose a quantitative measure for evaluating the quality of a stylized image and also have art historians rank patches from our approach against those from previous work. These and our qualitative results ranging from small image patches to megapixel stylistic images and videos show that our approach better captures the subtle nature in which a style affects content.[1]

Keywords: Style transfer, generative network, deep learning

1 Introduction

A picture may be worth a thousand words, but at least it contains a lot of very diverse information. This not only comprises *what* is portrayed, e.g., composition

---

[*] Both authors contributed equally to this work.
[1] Project page: `https://compvis.github.io/adaptive-style-transfer`.

Fig. 1. Evaluating the fine details preserved by our approach. Can you guess which of the cut-outs are from Monet's artworks and which are generated? Solution is on p. 15.
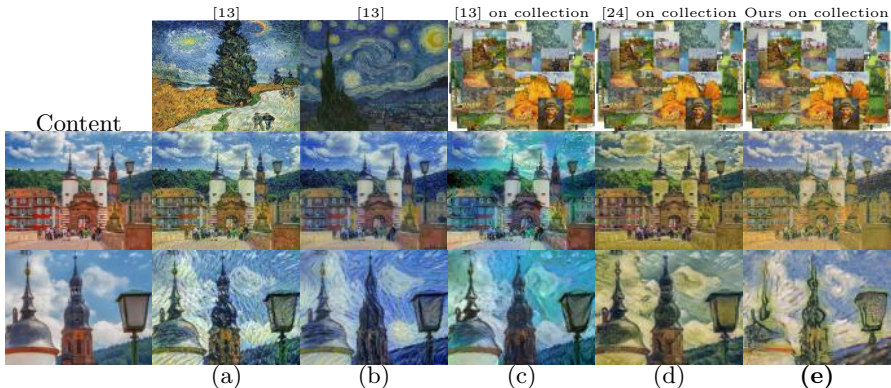
Fig. 2. Style transfer using different approaches on 1 and a collection of reference style images. (a) [13] using van Gogh's "Road with Cypress and Star" as reference style image; (b) [13] using van Gogh's "Starry night"; (c) [13] using the average Gram matrix computed across the collection of Vincent van Gogh's artworks; (d) [24] trained on the collection of van Gogh's artworks alternating target style images every SGD mini-batch; (e) our approach trained on the same collection of van Gogh's artworks. Stylizations *(a)* and *(b)* depend significantly on the particular style image, but using a collection of the style images *(c)*, *(d)* does not produce visually plausible results, due to oversmoothing over the numerous Gram matrices. In contrast, our approach *(e)* has learned how van Gogh is altering particular content in a specific manner (edges around objects also stylized, cf. bell tower)

of a scene and individual objects, but also *how* it is depicted, referring to the artistic style of a painting or filters applied to a photo. Especially when considering artistic images, it becomes evident that not only content but also style is a crucial part of the message an image communicates (just imagine van Gogh's Starry Night in the style of Pop Art). Here, we follow the common wording of our community and refer to 'content' as a synonym for 'subject matter' or 'sujet', preferably used in art history. A vision system then faces the challenge to decompose and separately represent the content and style of an image to enable a direct analysis based on each individually. The ultimate test for this ability is style transfer [13] – exchanging the style of an image while retaining its content.

In contrast to the seminal work of Gatys et al. [13], who have relied on powerful but slow iterative optimization, there has recently been a focus on feedforward generator networks [24,47,42,44,29,7,21]. The crucial representation in all these approaches has been based on a VGG16 or VGG19 network [41], pretrained on ImageNet [36]. However, a recent trend in deep learning has been to avoid supervised pre-training on a million images with tediously labeled object bounding boxes [46]. In the setting of style transfer this has the particular benefit of avoiding from the outset any bias introduced by ImageNet, which has been assembled without artistic consideration. Rather than utilizing a separate pre-trained VGG network to measure and optimize the quality of the stylistic output [13,24,47,42,44,29,7], we employ an encoder-decoder architecture with

adversarial discriminator, Fig. 3, to stylize the input content image and also use the encoder to measure the reconstruction loss. In essence the stylized output image is again run through the encoder and compared with the encoded input content image. Thus, we learn a style-specific content loss from scratch, which adapts to the specific way in which a particular style retains content and is more adaptive than a comparison in the domain of RGB images [52].

Most importantly, however, previous work has only been based on a *single* style image. This stands in stark contrast to art history which understands "style as an expression of a collective spirit" resulting in a "distinctive manner which permits the grouping of works into related categories" [10]. As a result, art history developed a scheme, which allows to identify groups of artworks based on shared qualities. Artistic style consists of a diverse range of elements, such as form, color, brushstroke, or use of light. Therefore, it is insufficient to only use a single artwork, because it might not represent the full scope of an artistic style. Today, freely available art datasets such as Wikiart [25] easily contain more than 100K images, thus providing numerous examples for various styles. Previous work [13,24,47,42,44,29,7] has represented style based on the Gram matrix, which captures highly image-specific style statistics, cf. Fig. 2. To combine several style images in [13,24,47,42,44,29,7] one needs to aggregate their Gram matrices. We have evaluated several aggregation strategies and averaging worked the best, Fig. 2(c). But, obviously, neither art history, nor statistics suggests aggregating Gram matrices. Additionally, we investigated alternating the target style images in every mini-batch while training [24], Fig. 2(d). However, all these methods cannot make proper use of several style images, because combining the Gram matrices of several images forfeits the details of style, cf. the analysis in Fig. 2. In contrast, our proposed approach allows to combine an arbitrary number of instances of a style during training.

We conduct extensive evaluations of the proposed style transfer approach; we quantitatively and qualitatively compare it against numerous baselines. Being able to generate high quality artistic works in high-resolution, our approach produces visually more detailed stylizations than the current state of the art style transfer approaches and yet shows real-time inference speed. The results are quantitatively validated by experts from art history and by adopting in this paper a *deception rate* metric based on a deep neural network for artist classification.

1.1   Related Work

In recent years, a lot of research efforts have been devoted to texture synthesis and style transfer problems. Earlier methods [18] are usually non-parametric and are build upon low-level image features. Inspired by Image Analogies [18], approaches [11,30,39,40] are based on finding dense correspondence between content and style image and often require image pairs to depict similar content. Therefore, these methods do not scale to the setting of arbitrary content images.

In contrast, Gatys et al. [13,12] proposed a more flexible iterative optimization approach based on a pre-trained VGG19 network [41]. This method pro-
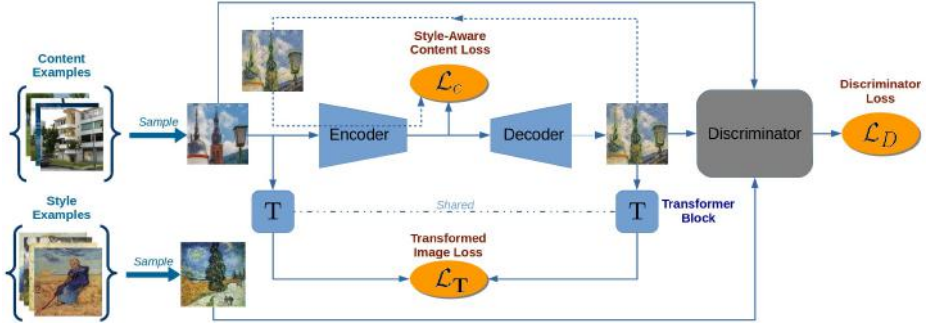
Fig. 3. Encoder-decoder network for style transfer based on style-aware content loss.

duces high quality results and works on arbitrary inputs, but is costly, since each optimization step requires a forward and backward pass through the VGG19 network. Subsequent methods [24,42,27] aimed to accelerate the optimization procedure [13] by approximating it with feed-forward convolutional neural networks. This way, only one forward pass through the network is required to generate a stylized image. Beyond that, a number of methods have been proposed to address different aspects of style transfer, including quality [14,49,4,47,23], diversity [28,44], photorealism [32], combining several styles in a single model [45,7,3] and generalizing to previously unseen styles [21,29,15,38]. However, all these methods rely on the fixed style representation which is captured by the features of a VGG [41] network pre-trained on ImageNet. Therefore they require a supervised pre-training on millions of labeled object bounding boxes and have a bias introduced by ImageNet, because it has been assembled without artistic consideration. Moreover, the image quality achieved by the costly optimization in [13] still remains an upper bound for the performance of recent methods. Other works like [48,5,34,9,1] learn how to discriminate different techniques, styles and contents in the latent space. Zhu et al. [52] learn a bidirectional mapping between a domain of content images and paintings using generative adversarial networks. Employing cycle consistency loss, they directly measure the distance between a backprojection of the stylized output and the content image in the RGB pixel space. Measuring distances in the RGB image domain is not just generally prone to be coarse, but, especially for abstract styles, a pixel-wise comparison of backwards mapped stylized images is not suited. Then, either content is preserved and the stylized image is not sufficiently abstract, e.g., not altering object boundaries, or the stylized image has a suitable degree of abstractness and so a pixel-based comparison with the content image must fail. Moreover, the more abstract the style is, the more potential backprojections into the content domain exist, because this mapping is underdetermined (think of the many possible content images for a single cubistic painting). In contrast, we spare the ill-posed backward mapping of styles and compare stylized and content images in the latent space which is trained jointly with the style transfer network. Since both content and stylized images are run through our encoder,
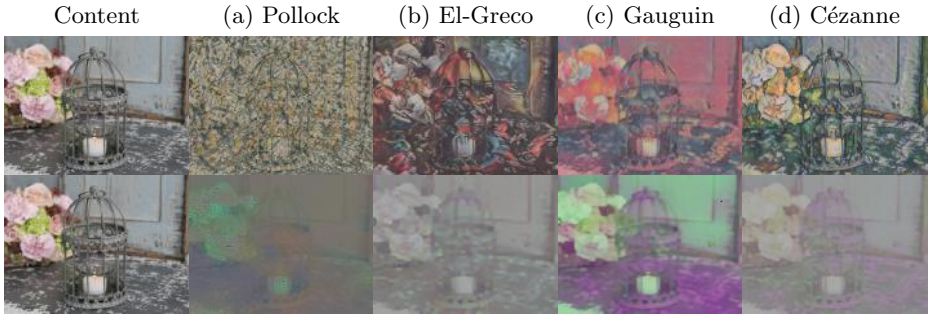
| Content | (a) Pollock | (b) El-Greco | (c) Gauguin | (d) Cézanne |

Fig. 4. 1st row - results of style transfer for different styles. 2nd row - sketchy content visualization reconstructed from the latent space $E(x)$ using method of [33]. (a) The encoder for Pollock does not preserve much content due to the abstract style; (b) only rough structure of the content is preserved (coarse patches) because of the distinct style of El Greco; (c) latent space highlights surfaces of the same color and that fine object details are ignored, since Gauguin was less interested in details, often painted plain surfaces and used vivid colors; (d) encodes the thick, wide brushstrokes Cézanne used, but preserves a larger palette of colors.

the latent space is trained to only pay attention to the commonalities, i.e., the content present in both. Another consequence of the cycle consistency loss is that it requires content and style images used for training to represent similar scenes [52], and thus training data preparation for [52] involves tedious manual filtering of samples, while our approach can be trained on arbitrary unpaired content and style images.

2   Approach

To enable a fast style transfer that instantly transfers a content image or even frames of a video according to a particular style, we need a feed-forward architecture [24] rather than the slow optimization-based approach of [13]. To this end, we adopt an encoder-decoder architecture that utilizes an encoder network E to map an input content image x onto a latent representation $z = E(x)$. A generative decoder G then plays the role of a painter and generates the stylized output image $y = G(z)$ from the sketchy content representation z. Stylization then only requires a single forward pass, thus working in real-time.

2.1   Training with a Style-Aware Content Loss

Previous approaches have been limited in that training worked only with a single style image [13,24,21,29,47,7,42] or that style images used for training had to be similar in content to the content images [52]. In contrast, given a single style image y_0 we include a set Y of related style images $y_j \in Y$, which are automatically selected (see Sec. 2.2) from a large art dataset (Wikiart). We do

not require the y_j to depict similar content as the set X of arbitrary content images $x_i \in X$, which we simply take from Places365 [51]. Compared to [52], we thus can utilize standard datasets for content and style and need no tedious manual selection of the x_i and y_j as described in Sect. 5.1 and 7.1 of [52].

To train E and G we employ a standard adversarial discriminator D [16] to distinguish the stylized output $G(E(x_i))$ from real examples $y_j \in Y$,

$$\mathcal{L}_D(E, G, D) = \mathbb{E}_{y \sim p_Y(y)} [\log D(y)] + \mathbb{E}_{x \sim p_X(x)} [\log (1 - D(G(E(x))))] \quad (1)$$

However, the crucial challenge is to decide which details to retain from the content image, something which is not captured by Eq. 1. Contrary to previous work, we want to directly enforce E to strip the latent space of all image details that the target style disregards. Therefore, the details that need to be retained or ignored in z depend on the style. For instance, Cubism would disregard texture, whereas Pointillism would retain low-frequency textures. Therefore, a pre-trained network or fixed similarity measure [13] for measuring the similarity in content between x_i and y_i is violating the art historical premise that the manner, in which content is preserved, depends on the style. Similar issues arise when measuring the distance after projecting the stylized image $G(E(x_i))$ back into the domain X of original images with a second pair of encoder and decoder $G_2(E_2(G(E(x_i))))$. The resulting loss proposed in [52],

$$\mathcal{L}_{cycleGAN} = \mathbb{E}_{x \sim p_X(x)} [\|x - G_2(E_2(G(E(x))))\|_1], \quad (2)$$

fails where styles become abstract, since the backward projection of abstract art to the original image is highly underdetermined.

Therefore, we propose a style-aware content loss that is being optimized, while the network learns to stylize images. Since encoder training is coupled with training of the decoder, which produces artistic images of the specific style, the latent vector z produced for the input image x can be viewed as its style-dependent sketchy content representation. This latent space representation is changing during training and hence *adapts* to the style. Thus, when measuring the similarity in content between input image x_i and the stylized image $y_i = G(E(x_i))$ in the latent space, we focus only on those details which are relevant for the style. Let the latent space have d dimensions, then we define a style-aware content loss as normalized squared Euclidean distance between $E(x_i)$ and $E(y_i)$:

$$\mathcal{L}_c(E, G) = \mathbb{E}_{x \sim p_X(x)} \left[\frac{1}{d} \|E(x) - E(G(E(x)))\|_2^2 \right] \quad (3)$$

To show the additional intuition behind the style-aware content loss we used the method [33] to reconstruct the content image from latent representations trained on different styles and illustrated it in Fig. 4. It can be seen that latent space encodes a sketchy, style-specific visual content, which is implicitly used by the loss function. For example, Pollock is famous for his abstract paintings,

so reconstruction (a) shows that the latent space ignores most of the object structure; Gauguin was less interested in details, painted a lot of plain surfaces and used vivid colors which is reflected in the reconstruction (c), where latent space highlights surfaces of the same color and fine object details are ignored.

Since we train our model for altering the artistic style without supervision and from scratch, we now introduce extra signal to initialize training and boost the learning of the primary latent space. The simplest thing to do is to use an autoencoder loss which computes the difference between x_i and y_i in the RGB space. However, this loss would impose a high penalty for any changes in image structure between input x_i and output y_i, because it relies only on low-level pixel information. But we aim to learn image stylization and want the encoder to discard certain details in the content depending on style. Hence the autoencoder loss will contradict with the purpose of the style-aware loss, where the style determines which details to retain and which to disregard. Therefore, we propose to measure the difference after applying a weak image transformation on x_i and y_i, which is learned while learning E and G. We inject in our model a transformer block $\mathbf{T}$ which is essentially a one-layer fully convolutional neural network taking an image as input and producing a transformed image of the same size. We apply T to images x_i and $y_i = G(E(x_i))$ before measuring the difference. We refer to this as *transformed image loss* and define it as

$$\mathcal{L}_{\mathbf{T}}(E,G) = \mathop{\mathbb{E}}_{x \sim p_X(x)} \left[\frac{1}{CHW} ||\mathbf{T}(x) - \mathbf{T}(G(E(x)))||_2^2 \right], \qquad (4)$$

where $C \times H \times W$ is the size of image x and for training T is initialized with uniform weights.

Fig. 3 illustrates the full pipeline of our approach. To summarize, the full objective of our model is:

$$\mathcal{L}(E,G,D) = \mathcal{L}_c(E,G) + \mathcal{L}_t(E,G) + \lambda\mathcal{L}_D(E,G,D), \qquad (5)$$

where λ controls the relative importance of adversarial loss. We solve the following optimization problem:

$$E,G = \arg \min_{E,G} \max_{D} \mathcal{L}(E,G,D). \qquad (6)$$

2.2 Style Image Grouping

In this section we explain an automatic approach for gathering a set of related style images. Given a single style image y_0 we strive to find a set Y of related style images $y_j \in Y$. Contrary to [52] we avoid tedious manual selection of style images and follow a fully automatic approach. To this end, we train a VGG16 [41] network C from scratch on the Wikiart [25] dataset to predict an artist given the artwork. The network is trained on the 624 largest (by number of works) artists from the Wikiart dataset. Note that our ultimate goal is stylization and numerous artists can share the same style, e.g., Impressionism, as well as a single artist can exhibit different styles, such as the different stylistic periods of Picasso.

However, we do *not* use any *style* labels. Artist classification in this case is the surrogate task for learning meaningful features in the artworks' domain, which allows to retrieve similar artworks to image y_0.

Let $\phi(y)$ be the activations of the fc6 layer of the VGG16 network C for input image y. To get a set of related style images to y_0 from the Wikiart dataset $\mathcal{Y}$ we retrieve all nearest neighbors of y_0 based on the cosine distance δ of the activations $\phi(\cdot)$, i.e.

$$Y = \{y \mid y \in \mathcal{Y}, \delta(\phi(y), \phi(y_0)) < t\}, \tag{7}$$

where $\delta(a,b) = 1 + \frac{\phi(a)\phi(b)}{||a||_2||b||_2}$ and t is the 10% quantile of all pairwise distances in the dataset $\mathcal{Y}$.

3   Experiments

To compare our style transfer approach with the state-of-the-art, we first perform extensive qualitative analysis, then we provide quantitative results based on the *deception score* and evaluations of experts from art history. Afterwards in Sect. 3.3 we ablate single components of our model and show their importance.

Implementation details: The basis for our style transfer model is an encoder-decoder architecture, cf. [24]. The encoder network contains 5 `conv` layers: 1×`conv-stride-1` and 4×`conv-stride-2`. The decoder network has 9 residual blocks [17], 4 upsampling blocks and 1×`conv-stride-1`. For upsampling blocks we used a sequence of nearest-neighbor upscaling and `conv-stride-1` instead of fractionally strided convolutions [31], which tend to produce heavier artifacts [35]. Discriminator is a fully convolutional network with 7×`conv-stride-2` layers. For a detailed network architecture description we refer to the supplementary material. We set $\lambda = 0.001$ in Eq. 5. During the training process we sample 768×768 content image patches from the training set of Places365 [51] and 768×768 style image patches from the Wikiart [25] dataset. We train for 300000 iterations with batch size 1, learning rate 0.0002 and Adam [26] optimizer. The learning rate is reduced by a factor of 10 after 200000 iterations.

Baselines: Since we aim to generate high-resolution stylizations, for comparison we run style transfer on our method and all baselines for input images of size 768×768, unless otherwise specified. We did not not exceed this resolution when comparing, because some other methods were reaching the GPU memory limit. We optimize Gatys et al. [13] for 500 iterations using L-BFGS. For Johnson et al. [24] we used the implementation of [8] and trained a separate network for every reference style image on the same content images from Places365 [51] as our method. For Huang et al. [21], Chen et al. [4] and Li et al. [29] implementations and pre-trained models provided by the authors were used. Zhu et al. [52] was trained on exactly the same content and style images as our approach using the source code provided by the authors. Methods [13,24,21,4,29] utilized only one example per style, as they cannot benefit from more (cf. the analysis in Fig. 2).
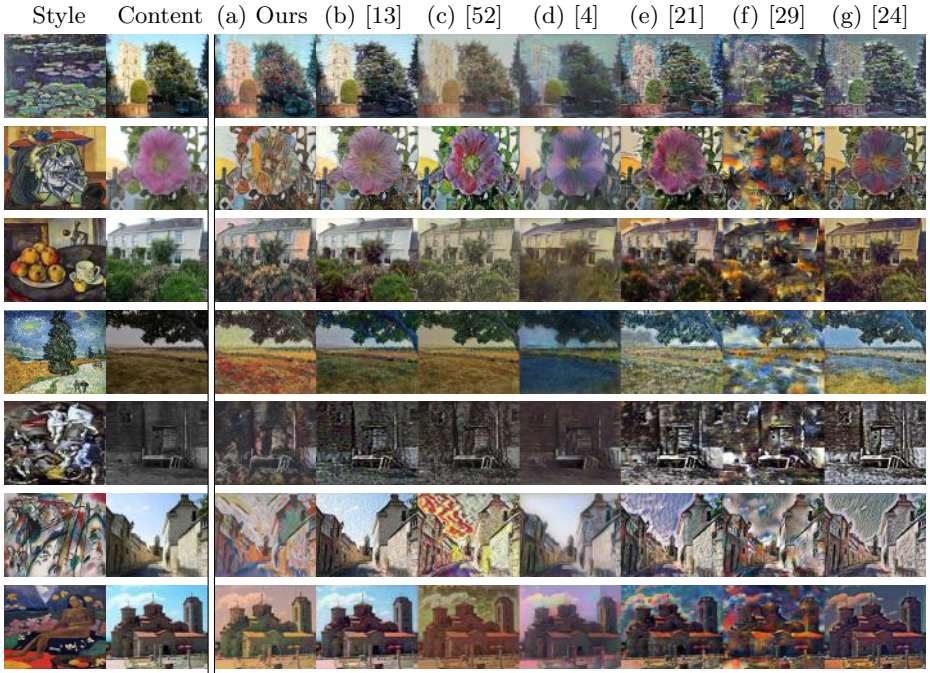
Style   Content (a) Ours (b) [13] (c) [52] (d) [4] (e) [21] (f) [29] (g) [24]

Fig. 5. Results from different style transfer methods. We compare methods on different styles and content images.

3.1   Qualitative Results

Full image stylization: In Fig. 5 we demonstrate the effectiveness of our approach for stylizing different contents with various styles. Chen et al. [4] work on the overlapping patches extracted from the content image, swapping the features of the original patch with the features of the most similar patch in the style image, and then averages the features in the overlapping regions, thus producing an over-smoothed image without fine details (Fig. 5(d)). [21] produces a lot of repetitive artifacts, especially visible on flat surfaces, cf. Fig. 5(e, rows 1, 4–6). Method [29] fails to understand the content of the image and applies different colors in the wrong locations (Fig. 5(f)). Methods [24,52] often fail to alter content image and their effect may be characterized as shifting the color histogram, e.g., Fig. 5(g, rows 3, 7; c, rows 1, 3–4). One reason for such failure cases of [52] is the loss in the RGB pixel space based on the difference between a backward mapping of the stylized output and the content image. Another reason for this is that we utilized the standard Places365 [51] dataset and did not hand-pick training content images, as is advised for [52]. Thus, artworks and content images used for training differed significantly in their content, which is the ultimate test for a stylization that truly alters the input and goes beyond a direct mapping between regions of content and style images. The optimization-based method [13] often works better than other baselines, but produces a lot of prominent ar-
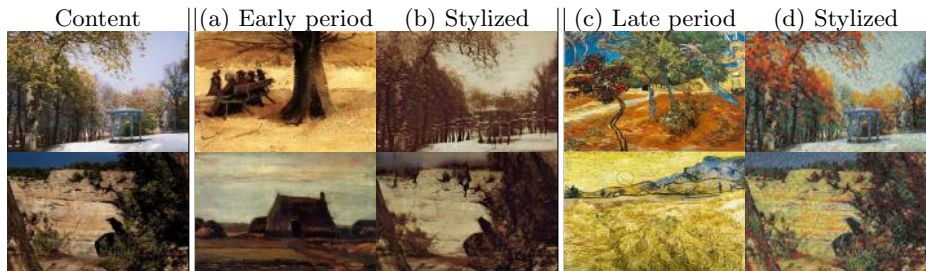
Content ||(a) Early period   (b) Stylized ||(c) Late period   (d) Stylized

Fig. 6. Artwork examples of the early artistic period of van Gogh (a) and his late period (c). Style transfer of the content image (1st column) onto the early period is presented in (b) and the late period in (d).

tifacts, leading to details of stylizations looking unnatural, cf. Fig. 5(b, rows 4, 5, 6). This is due to an explicit minimization of the loss directly on the pixel level. In contrast to this, our model can not only handle styles, which have salient, simple to spot characteristics, but also styles, such as El Greco's Mannerism, with less graspable stylistic characteristics, where other methods fail (Fig. 5, b–g, 5th row).

Fine-grained style details: In Fig. 7 we show zoomed in cut-outs from the stylized images. Interestingly, the stylizations of methods [13,4,21,29,20] do not change much across styles (compare Fig. 7(d, f–i, rows 1–3)). Zhu et al. [52] produce more diverse images for different styles, but obviously cannot alter the edges of the content (blades of grass are *clearly visible* on all the cutouts in Fig. 7(e)). Fig. 7(c) shows the stylized cutouts of our approach, which exhibit significant changes from one style to another. Another interesting example is the style of Pollock, Fig. 7(row 8), where the style-aware loss allows our model to properly alter content to the point of discarding it – as would be expected from a Pollock action painting. Our approach is able to generate high-resolution stylizations with a lot of style specific details and retains those content details which are necessary for the style.

Style transfer for different periods of van Gogh: We now investigate our ability to properly model fine differences in style *despite* using a group of style images. Therefore, we take two reference images Fig. 6(a) and (c) from van Gogh's early and late period, respectively, and acquire related style images for both from Wikiart. It can be clearly seen that the stylizations produced for either period Fig. 6(b, d) are fairly different and indeed depict the content in correspondence with the style of early (b) and late (d) periods of van Gogh. This highlights that collections of style images are properly used and do not lead to an averaging effect.

High-resolution image generation: Our approach allows us to produce high quality stylized images in high-resolution. Fig. 8 illustrates an example of the generated piece of art in the style of Berthe Morisot with resolution 1280×1280. The result exhibits a lot of fine details such as color transitions of the oil paint

Fig. 7. Details from stylized images produced for different styles for a fixed content image (a). (b) is our entire stylized image, (c) the zoomed in cut-out and (d)-(i) the same region for competitors. Note the variation across different styles along the *column* for our method compared to other approaches. This highlights the ability to adapt content (not just colors or textures) where demanded by a style. Fine grained artistic details with sharp boundaries are produced, while altering the original content edges.

and brushstrokes of different sizes. More HD images are in the supplementary material.

Fig. 8. High-resolution image (1280x1280 pix) generated by our approach. A lot of fine details and brushstrokes are visible. A style example is in the bottom left corner.

Fig. 9. Results of our approach applied to the HD video of Eadweard Muybridge "The horse in motion" (1878). Every frame was *independently* processed (no smoothing or post-processing) by our model in the style of Picasso. Video resolution is 1920×1280 pix. The full video is available on YouTube: `https://youtu.be/TtHJcL8Feu0`.

Real-time HD video stylization: We also apply our method to several videos. Our approach can stylize HD videos (1280×720) at 9 FPS. Fig. 9 shows stylized frames from a video. We did not use a temporal regularization to show that our method produces equally good results for consecutive frames with varying appearance w/o extra constraints. Stylized videos are in the supplementary material.

3.2   Quantitative Evaluation

Style transfer deception rate: While several metrics [37,2,19] have been proposed to evaluate the quality of image generation, until now no evaluation metric has been proposed for an automatic evaluation of style transfer results. To measure the quality of the stylized images, we introduce the *style transfer deception rate*. We use a VGG16 network trained from scratch to classify 624 artists on Wikiart. Style transfer deception rate is calculated as the fraction of generated images which were classified by the network as the artworks of an artist for which the stylization was produced. For fair comparison with other approaches, which used only one style image y_0 (hence only one artist), we restricted Y to only contain samples coming from the same artist as the query example y_0. We selected 18 different artists (i.e. styles). For every method we generated 5400 stylizations (18 styles, 300 per style). In Tab. 1 we report mean deception rate for 18 styles. Our method achieves 0.393 significantly outperforming the baselines. For comparison, mean accuracy of the network on hold-out real images of aforementioned 18 artists from Wikiart is 0.616.

Human art history experts perceptual studies: Three experts (with a PhD in art history with focus on modern and pre-modern paintings) have compared results of our method against recent work. Each expert was shown 1000 groups of images. Each group consists of stylizations which were generated by different methods based on the same content and style images. Experts were asked to choose one image which best and most realistically reflects the current style. The score is computed as the fraction of times a specific method was chosen as the best in the group. We calculate a mean expert score for each method using 18 different styles and report them in Tab. 1. Here, we see that the experts selected our method in around 50% of the cases.

Speed and memory: Tab. 2 shows the time and memory required for stylization of a single image of size 768×768 px for different methods. One can see that our approach and that of [24] and [52] have comparable speed and only very modest demands on GPU memory, compared to modern graphics cards.

Table 1. Mean deception rate and mean expert score for different methods. The higher the better.

Method	Deception rate	Expert score
Content images	0.002	-
AdaIn [21]	0.074	0.060
PatchBased [4]	0.040	0.132
Johnson et al. [24]	0.051	0.048
WCT [29]	0.035	0.044
CycleGan [52]	0.139	0.044
Gatys et al. [13]	0.147	0.178
Ours	**0.393**	**0.495**

Table 2. Average inference time and GPU memory consumption, measured on a Titan X Pascal, for different methods with batch size 1 and input image of 768×768 pix.

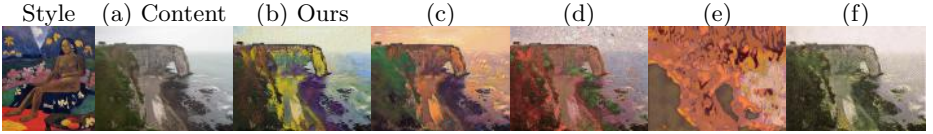| Method	Time	GPU memory
Gatys et al. [13]	200 sec	3887 MiB
CycleGan [52]	0.07 sec	1391 MiB
AdaIn [21]	0.16 sec	8872 MiB
PatchBased [4]	8.70 sec	4159 MiB
WCT [29]	5.22 sec	10720 MiB
Johnson et al. [24]	**0.06 sec**	**671 MiB**
Ours	0.07 sec	1043 MiB

Fig. 10. Different variations of our method for Gauguin stylization. See Sect. 3.3 for details. (a) Content image; (b) full model ($\mathcal{L}_c$, $\mathcal{L}_{rgb}$ and $\mathcal{L}_D$); (c) $\mathcal{L}_{rgb}$ and $\mathcal{L}_D$; (d) without transformer block; (e) only $\mathcal{L}_D$; (f) trained with all of Gauguin's artworks as style images. Please zoom in to compare.

Fig. 11. Encoder ablation studies: (a) stylization using our model; (b) stylization using pre-trained VGG16 encoder instead of E.

3.3   Ablation Studies

Effect of different losses: We study the effect of different components of our model in Fig. 10. Removing the style-aware content loss significantly degrades the results, (c). We observe that without the style-aware loss training becomes instable and often stalls. If we remove the transformed image loss, which we introduced for a proper initialization of our model that is trained from scratch, we notice mode collapse after 5000 iterations. Training directly with pixel-wise L2 distance causes a lot of artifacts (grey blobs and flaky structure), (d). Training only with a discriminator neither exhibits the variability in the painting nor in the content, (e). Therefore we conclude that both the style-aware content loss and the transformed image loss are critical for our approach.

Single vs collection of style images: Here, we investigate the importance of the style image grouping. First, we trained a model with only one style image of Gauguin, which led to mode collapse. Second, we trained with all of Gauguin's artworks as style images (without utilizing style grouping procedure). It produced unsatisfactory results, cf. Fig. 10(f), because style images comprised several distinct styles. Therefore we conclude that to learn a good style transfer model it is important to group style images according to their stylistic similarity.

Encoder ablation: To investigate the effect of our encoder E, we substitute it with VGG16 [41] encoder (up to `conv5_3`) pre-trained on ImageNet. The VGG encoder retains features that separate object classes (since it was trained discriminatively), as opposed to our encoder which is trained to retain style-specific content details. Hence, our encoder is not biased towards class-discriminative fea-

tures, but is style specific and trained from scratch. Fig. 11(a, b) show that our approach produces better results than with pre-trained VGG16 encoder.

4 Conclusion

This paper has addressed major conceptual issues in state-of-the-art approaches for style transfer. We overcome the limitation of only a single style image or the need for style and content training images to show similar content. Moreover, we exceed a mere pixel-wise comparison of stylistic images or models that are pre-trained on millions of ImageNet bounding boxes. The proposed style-aware content loss enables a real-time, high-resolution encoder-decoder based stylization of images and videos and significantly improves stylization by capturing how style affects content.

References

1. Bautista, M.A., Sanakoyeu, A., Tikhoncheva, E., Ommer, B.: Cliquecnn: Deep unsupervised exemplar learning. In: Advances in Neural Information Processing Systems. pp. 3846–3854 (2016)
2. Che, T., Li, Y., Jacob, A.P., Bengio, Y., Li, W.: Mode regularized generative adversarial networks. arXiv preprint arXiv:1612.02136 (2016)
3. Chen, D., Yuan, L., Liao, J., Yu, N., Hua, G.: Stylebank: An explicit representation for neural image style transfer. In: Proc. CVPR (2017)
4. Chen, T.Q., Schmidt, M.: Fast patch-based style transfer of arbitrary style. arXiv preprint arXiv:1612.04337 (2016)
5. Collomosse, J., Bui, T., Wilber, M.J., Fang, C., Jin, H.: Sketching with style: Visual search with sketches and aesthetic context. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
6. Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., Bharath, A.A.: Generative adversarial networks: An overview. IEEE Signal Processing Magazine **35**(1), 53–65 (2018)
7. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. Proc. of ICLR (2017)
8. Engstrom, L.: Fast style transfer. https://github.com/lengstrom/fast-style-transfer/ (2016), commit 55809f4e
9. Esser, P., Sutter, E., Ommer, B.: A variational u-net for conditional appearance and shape generation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2018)
10. Fernie, E.: Art History and its Methods: A critical anthology. Phaidon, London (1995), p. 361
11. Frigo, O., Sabater, N., Delon, J., Hellier, P.: Split and match: Example-based adaptive patch sampling for unsupervised style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 553–561 (2016)
12. Gatys, L.A., Ecker, A.S., Bethge, M.: Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. arXiv preprint arXiv:1505.07376 **12** (2015)
13. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on. pp. 2414–2423. IEEE (2016)
14. Gatys, L.A., Ecker, A.S., Bethge, M., Hertzmann, A., Shechtman, E.: Controlling perceptual factors in neural style transfer. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
15. Ghiasi, G., Lee, H., Kudlur, M., Dumoulin, V., Shlens, J.: Exploring the structure of a real-time, arbitrary neural artistic stylization network. arXiv preprint arXiv:1705.06830 (2017)
16. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
18. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image analogies. In: Proceedings of the 28th annual conference on Computer graphics and interactive techniques. pp. 327–340. ACM (2001)

19. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems. pp. 6629–6640 (2017)
20. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006). https://doi.org/10.1126/science.1127647, `http://science.sciencemag.org/content/313/5786/504`
21. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: ICCV (2017)
22. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Transactions on Graphics (TOG) **36**(4), 107 (2017)
23. Jing, Y., Liu, Y., Yang, Y., Feng, Z., Yu, Y., Song, M.: Stroke controllable fast style transfer with adaptive receptive fields. arXiv preprint arXiv:1802.07101 (2018)
24. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision. pp. 694–711. Springer (2016)
25. Karayev, S., Trentacoste, M., Han, H., Agarwala, A., Darrell, T., Hertzmann, A., Winnemoeller, H.: Recognizing image style. arXiv preprint arXiv:1311.3715 (2013)
26. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
27. Li, C., Wand, M.: Precomputed real-time texture synthesis with markovian generative adversarial networks. In: European Conference on Computer Vision. pp. 702–716. Springer (2016)
28. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Diversified texture synthesis with feed-forward networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2017)
29. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Advances in Neural Information Processing Systems. pp. 385–395 (2017)
30. Liao, J., Yao, Y., Yuan, L., Hua, G., Kang, S.B.: Visual attribute transfer through deep image analogy. arXiv preprint arXiv:1705.01088 (2017)
31. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
32. Luan, F., Paris, S., Shechtman, E., Bala, K.: Deep photo style transfer. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
33. Mahendran, A., Vedaldi, A.: Understanding deep image representations by inverting them. In: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2015)
34. Mao, H., Cheung, M., She, J.: Deepart: Learning joint representations of visual arts. In: Proceedings of the 2017 ACM on Multimedia Conference. pp. 1183–1191. ACM (2017)
35. Odena, A., Dumoulin, V., Olah, C.: Deconvolution and checkerboard artifacts. Distill (2016). https://doi.org/10.23915/distill.00003, `http://distill.pub/2016/deconv-checkerboard`
36. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y

37. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: Advances in Neural Information Processing Systems. pp. 2234–2242 (2016)
38. Shen, F., Yan, S., Zeng, G.: Meta networks for neural style transfer. arXiv preprint arXiv:1709.04111 (2017)
39. Shih, Y., Paris, S., Barnes, C., Freeman, W.T., Durand, F.: Style transfer for headshot portraits. ACM Transactions on Graphics (TOG) **33**(4), 148 (2014)
40. Shih, Y., Paris, S., Durand, F., Freeman, W.T.: Data-driven hallucination of different times of day from a single outdoor photo. ACM Transactions on Graphics (TOG) **32**(6), 200 (2013)
41. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
42. Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.S.: Texture networks: Feed-forward synthesis of textures and stylized images. In: ICML. pp. 1349–1357 (2016)
43. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
44. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In: Proc. CVPR (2017)
45. Wang, H., Liang, X., Zhang, H., Yeung, D.Y., Xing, E.P.: Zm-net: Real-time zero-shot image manipulation network. arXiv preprint arXiv:1703.07255 (2017)
46. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. arXiv preprint arXiv:1505.00687 (2015)
47. Wang, X., Oxholm, G., Zhang, D., Wang, Y.F.: Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017)
48. Wilber, M.J., Fang, C., Jin, H., Hertzmann, A., Collomosse, J., Belongie, S.: Bam! the behance artistic media dataset for recognition beyond photography. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017)
49. Wilmot, P., Risser, E., Barnes, C.: Stable and controllable neural texture synthesis and style transfer using histogram losses. arXiv preprint arXiv:1701.08893 (2017)
50. Xue, Y., Xu, T., Zhang, H., Long, R., Huang, X.: Segan: Adversarial network with multi-scale l_1 loss for medical image segmentation. arXiv preprint arXiv:1706.01805 (2017)
51. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: Advances in neural information processing systems. pp. 487–495 (2014)
52. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE International Conference on Computer Vision (2017)

A   Implementation Details

A.1   Network Architecture

We incarnate our approach using an encoder-decoder architecture with a discriminator. Below we follow the naming convention similar to the one used in [52].

Let:
* `cFs1-k` denote $F \times F$ Convolution-InstanceNorm-ReLU layer with k filters and stride 1;
* `cFs1-k-sigmoid` denote $F \times F$ Convolution-InstanceNorm-Sigmoid layer with k filters and stride 1;
* `cFs1-k-noact` denote $F \times F$ Convolution-InstanceNorm layer with k filters and stride 1;
* `dF-k` denote a $F \times F$ Convolution-InstanceNorm-ReLU layer with k filters and stride 2;
* `dF-k-LReLU` denote a $F \times F$ Convolution-InstanceNorm-LeakyReLU layer with k filters and stride 2;
* `Rk` denote a residual block that contains two 3×3 convolutional layers followed by InstanceNorm [43];
* `uk` denote an upscaling block that contains nearest-neighbor upscaling by a factor of 2 followed by Convolution-InstanceNorm-ReLU layer with k filters and stride 1.

All convolutional layers use reflection padding.

Encoder-decoder architecture   Encoder has InstanceNorm layer at the beginning and 5 convolutional layers: `InstanceNorm, c3s1-32, d3-32, d3-64, d3-128, d3-256`. Decoder contains 9 residual blocks, 4 upscaling blocks and 1 convolution 7×7 with sigmoid activation: `R256×9, u256, u128, u64, u32, c7s1-3-sigmoid`.

Transformer block T   Transformer block **T** contains a convolutional layer followed by a weight normalization layer with fixed norm $||W|| = 1$. Convolutional layer: 3 kernels of size 10×10, stride 1, uniformly initialized.

Discriminator architecture   Discriminator is implemented as a fully convolutional network and tries to classify if input image patches are real or fake. It contains 7 convolutional layers. We use Leaky ReLU activations with slope 0.2. Architecture is defined as: `d5-128-LReLU, d5-128-LReLU, d5-256-LReLU, d5-512-LReLU, d5-512-LReLU, d5-1024-LReLU, d5-1024-LReLU, c3s1-1-noact`.

We use 4 auxiliary classifiers (implemented as a convolutional layer with 1 filter each) to capture image details on different scales [22,50] and to alleviate artifacts: auxiliary classifiers were added after 1st, 2nd, 4th and 6th convolutional layers of the discriminator. We simply sum up all the losses from different scales.
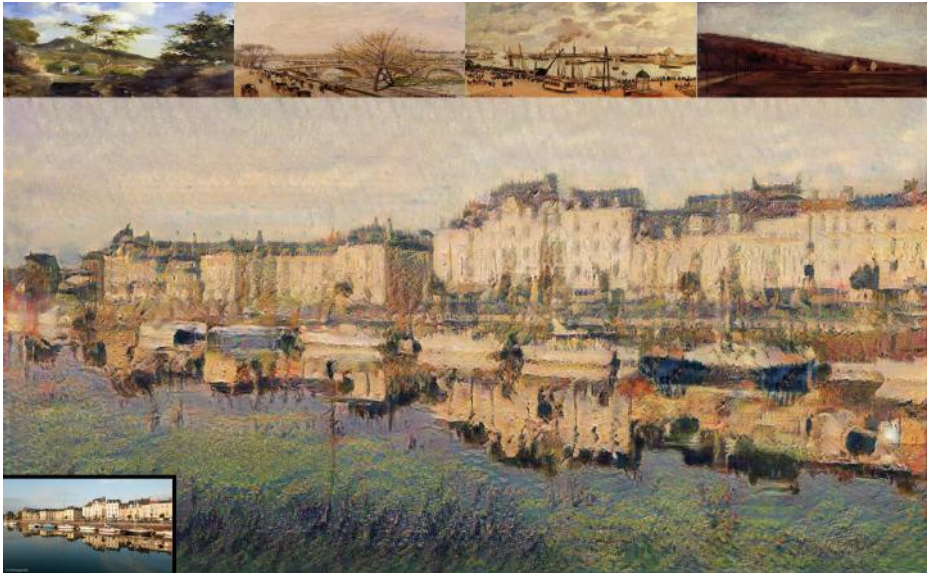
Fig. 12. High-resolution (2288×1280 pix) Pissarro stylization by our approach. The top row depicts 4 randomly chosen from set Y instances of style obtained by the style image grouping.

A.2   Style Image Grouping Details

As described in Sec. 2.2, the number of style images is not fixed and depends on the neighborhood size of the query image y_0 and how frequent the style is in the Wikiart [25] dataset. For different style examples in our experiments it varied from 55 to 1391 related images. We observe: the more style examples the better, as long as they are from the same style. Ablation studies (Sec. 3.3) show that using too few style examples (e.g. one) leads to mode collapse; at the same time using a lot of images, which are less related to each other (e.g. all images of an artist), produces unsatisfactory stylizations.

A.3   Training Details

All networks were trained from scratch on randomly cropped image patches of size 768×768 pix. λ in Eq. 5 was set to 0.001. We trained for 300000 iterations with batch size 1 and learning rate 2×10^{-4} using Adam [26] optimizer. We reduced the learning rate by a factor of 10 after 200000 iterations.

In each iteration we alternatively update encoder-decoder and discriminator. To balance out the discriminator and encoder-decoder training [6] we update solely the discriminator, if it has accuracy < 0.8, and update only encoder-decoder otherwise. The discriminator accuracy is calculated using exponential moving average during training.
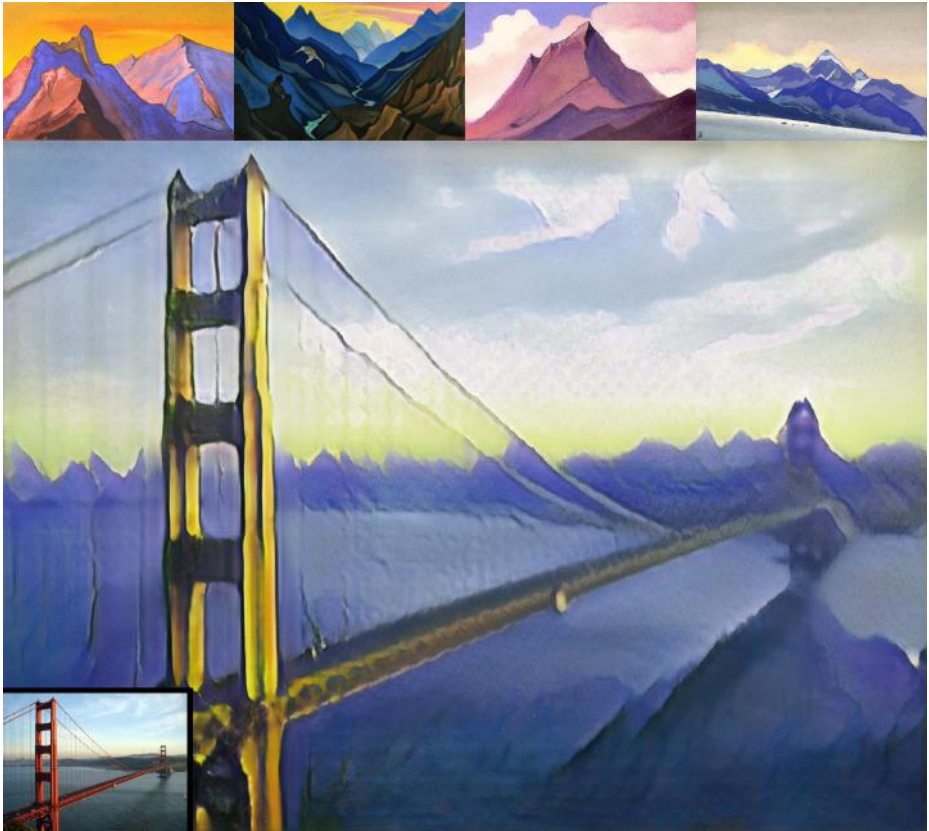
Fig. 13. High-resolution (1712×1280 pix) Roerich stylization by our approach. The top row depicts 4 randomly chosen from set Y instances of style obtained by the style image grouping.

B   Extra Qualitative Results

Altering the style of an existing artwork Our method is able to change the style of an existing artwork, rendering it in another style. It means, that our algorithm can also handle content images which are artistic, which, to our knowledge, was never shown in previous work before. We refer the reader to the results on the project page[1].

Real-time HD video stylization We stylized Eadweard Muybridge's "The horse in motion" video, 1878, in the style of Picasso. The stylized video resolution is 1920×1280 pix. Additionally, we apply our method to the video "Aix-en-Provence, France" by Jan van Bekkum in resolution 1920×1280 pix. We stitch together the original video and 3 stylizations in the style of Cézanne, Kandinsky

(a) Style   (b) Content   (c) Ours   (d)

Fig. 14. Investigation of the importance of an independent transformer block. (a) style image (van Gogh); (b) content image; (c) our stylization; (d) stylization with the loss $\mathcal{L}_{conv1}$ instead of $\mathcal{L}_{\mathbf{T}}$.

and Picasso produced by our approach. The videos are available on the project page[1].

More examples of high-resolution style transfer In Fig. 12, 13 we show extra high-resolution images stylized by our approach. Additional comparison of our method against baselines and more high-resolution visual results are available on the project page[1].

C   Extra ablation studies

Transformer block ablation To demonstrate the need in an independent transformer block, we replaced the transformed image loss $\mathcal{L}_{\mathbf{T}}$ with a loss $\mathcal{L}_{conv1}$ using conv1 features of the encoder,

$$\mathcal{L}_{conv1} = (E, G) = \mathop{\mathbb{E}}_{x \sim p_X(x)} \left[\frac{1}{d_{conv1}} ||\phi_{conv1}(x) - \phi_{conv1}(G(E(x)))||_2^2 \right], \qquad (8)$$

where $\phi_{conv1}(x)$ are the activations of the conv1 layer of the encoder E for input image x and d_{conv1} is the dimensianality of $\phi_{conv1}(x)$. This produced worse results (cf. Fig. 14(d)), since the loss $\mathcal{L}_{conv1}$ is then directly tied to the convolutional layer used for our style-aware content representation in the encoder. In contrast, the proposed transformer block can be learned independently from the early layers of the encoder.