Style Transfer by Rigid Alignment in Neural Net Feature Space

Gong Yaling

Abstract

Arbitrary style transfer is an important problem in computer vision that aims to transfer style patterns from an arbitrary style image to a given content image. However, current methods either rely on slow iterative optimization or fast pre-determined feature transformation, but at the cost of compromised visual quality of the styled image; especially, distorted content structure. In this work, we present an effective and efficient approach for arbitrary style transfer that seamlessly transfers style patterns as well as keep content structure intact in the styled image. We achieve this by aligning style features to content features using rigid alignment; thus modifying style features, unlike the existing methods that do the opposite. We demonstrate the effectiveness of the proposed approach by generating high-quality stylized images and compare the results with the current state-of-the-art techniques for arbitrary style transfer.

Keywords： Style Transfer.

1 Introduction

Given a pair of style and a target image, style transfer is a process of transferring the texture of the style image to the target image, keeping the structure of the target image unchanged. Most of the recent work in the neural style transfer is based on the implicit hypothesis is that working in deep neural network feature space can transfer texture and other high-level information from one image to another without altering the image structure much.

The goal of arbitrary style transformation is to be able to take style and content features as input and produce a stylized feature without affecting the quality of the reconstructed stylized image. However, current work in this area is failing in terms of the quality of the generated results. Most techniques over-distort the content or fail to balance low-level and overall stylistic patterns.

In this work, the authors address the above issues by modifying stylistic features rather than content features during the style transformation process. The specific assumption is to consider an image as a collection of points in feature space, where each point represents some spatial information, and if these point clouds are aligned with rigid alignment, it is possible to transform these points without introducing any distortion. By doing so, the problem of excessive content distortion is solved, since alignment does not manipulate content features.

2 Related works

Due to the wide variety of applications, the problem of style transfer has been studied for a long time in computer vision. Before seminal work by Gatys et al.[1], the problem of style transfer has been focused as non-photorealistic rendering (NPR) , and closely related to texture synthesis. Early approaches rely on finding low-level image correspondence and do not capture high-level semantic information well. As mentioned above, the use of CNN features in style transfer has improved the results significantly. We can divide the current Neural style transfer literature into four parts.

2.1 Slow optimization-based methods

Gatys et al.[1] introduced the first NST method for style transfer. The authors created artistic style transfer by matching multi-level feature statistics of content and style images extracted from a pre-trained image classification CNN (VGG[2]) using Gram matrix. Soon after this, other variations were introduced to achieve better style transfer, user controls like spatial control and color preserving or include semantic information. However, these methods require an iterative optimization over the image, which makes it impossible to apply in real-time.

2.2 Single style feed-forward networks

By approximating the iterative back-propagation procedure to a feed-forward neural network, trained either by the perceptual loss or Markovian generative adversarial loss. Although these approaches achieve style transfer in real-time, they require training a new model for every style. This makes them very difficult to use for multiple styles, as every single style requires hours of training.

2.3 Single network for multiple styles

An attempt is made to solve multiple styles by training a small number of parameters for each new style, while keeping the rest of the network unchanged. Conditional instance normalization[3] achieved it by training channel-wise statistics corresponding to each style. Stylebank[4] learned convolution filters for each style,[5] transferred styles by binary selection units and[6] trained a meta-network that generates a 14 layer network for each content and style image pair. On the other hand,[7] trained a weight matrix to combine style and content features. The major drawback is the model size that grows proportionally to the number of style images. Additionally, there is interference among different styles[8], which affects stylization quality.

2.4 Single network for arbitrary styles

Some recent works [[9],[10],[11],[12],[13]] have been focused on creating a single model for arbitrary style i.e., one model for any style. Gu et al.[13] rearrange style features patches with respect to content features patches. However, this requires solving an optimization problem to find the nearest neighbor, which is slow, thus not suitable for real-time use. Chen et al.[11] swaps the content feature patches with the closest style feature patch but fails if the domain gap between content and style is large. Sheng et al.[12] addresses this problem by first normalizing the features and then apply the patch swapping. Although this improves the stylization quality, it still produces content distortion and misses global style patterns. WCT[10] transfers multi-level style patterns by recursively applying whitening and coloring transformation (WCT) to a set of trained auto-encoders with different levels. However, similar to[12], WCT also produces content distortion; moreover, this introduces some unwanted patterns in the styled image[8]. Adaptive Instance normalization (AdaIN)[9] matches the channel-wise statistics (mean and variance) of content features to the style features, but this matching occurs only at one layer, which authors try to compensate by training a network on perpetual loss[14]. Although this does not introduce content distortion, it fails to capture style patterns.

3 Method

3.1 Overview

The common part of the existing arbitrary style transfer methods, that they all try to modify the content features during the style transfer process. This eventually creates content distortion. Different from existing methods, our approach manipulates the style features during style transfer. We achieve this in two steps. First, we apply channel-wise moment matching (mean and variance) between content and style features, just as AdaIN[9]. Second, we use rigid alignment (Procrustes analysis[4]) to align style features to content features. This alignment modifies the style features to adapt content structure, thus avoiding any content distortions while keeping its style information intact.

Generally Speaking style transfer as follows Let $z_c \in R^{C*H*W}$ is a feature extracted from a layer of a pre-trained CNN when the content image passes through the network. Here, H is the height, W is the width, and C is the number of channels of the feature z_c. Similarly, for style image $z_c \in R^{C*H*W}$ represents the corresponding features.

For any arbitrary style transfer method, we pass z_s and z_c to a transformation function T which outputs styled feature z_{cs} as described in eq. (1):

$$\mathbf{z}_{cs} = \mathcal{T}(\mathbf{z}_c, \mathbf{z}_s). \qquad (1)$$

Reconstruction of z_{cs} to image space gives the styled image. The difficult part is finding the transformation function T that is style-agnostic like [[12],[11],[10]], but unlike these, it captures local and global style information without distorting the content and does not need iterative optimization.

3.2 Moment Matching

Although AdaIN[9] is not style agnostic, it involves a transformation which is entirely style agnostic: channelwise moment matching. This involves matching channelwise mean and variance of content features to those of style features as follows:

$$\mathbf{z}_{c'} = \left(\frac{\mathbf{z}_c - \mathcal{F}_\mu(\mathbf{z}_c)}{\mathcal{F}_\sigma(\mathbf{z}_c)}\right)\mathcal{F}_\sigma(\mathbf{z}_s) + \mathcal{F}_\mu(\mathbf{z}_s). \qquad (2)$$

Here, $F_\mu(.)$ and $F_\sigma(.)$ is channel-wise mean and variance respectively. Although this channel-wise alignment produces unsatisfactory styled results, it is able to transfer local patterns of style image without distorting content structure.

3.3 Rigid Alignment

One simple way of alignment that prevents distortion is rigid alignment and (scaling). This involves shifting, scaling and finally rotation of the points that to be moved (styled features) with respect to the target points (content features after moment matching). For this we consider both features as point clouds of size C with each point is in R^{HW} space, i.e. z_c, $z_s \in R^{C\times HW}$. Now, we apply rigid transformation in following steps:

Step-I: Shifting. First, we need to shift both point clouds z_c and z_s to a common point in R^{HW} space. We center these point clouds to the origin as follows:

$$\begin{aligned}\bar{\mathbf{z}}_c &= \mathbf{z}_c - \boldsymbol{\mu}_c \\ \bar{\mathbf{z}}_s &= \mathbf{z}_s - \boldsymbol{\mu}_s.\end{aligned} \tag{3}$$

Here, μ_c and $\mu_s \in R^{HW}$ are the mean of the z_c and z_s point clouds respectively.

Step-II: Scaling. Both point clouds need to have the same scale before alignment. For this, we make each point cloud to have unit Frobenius norm:

$$\begin{aligned}\hat{\mathbf{z}}_c &= \frac{\bar{\mathbf{z}}_c}{\|\mathbf{z}_c\|_F} \\ \hat{\mathbf{z}}_s &= \frac{\bar{\mathbf{z}}_s}{\|\mathbf{z}_s\|_F}.\end{aligned} \tag{4}$$

Here, $||.||F$ represents Frobenius norm.

Step-III: Rotation. Next step involves rotation of z_s so that it can align perfectly with z_c. For this, we multiply z_s to a rotation matrix that can be created as follows:

$$\underset{\mathbf{Q}}{\arg\min} \|\hat{\mathbf{z}}_s\mathbf{Q} - \hat{\mathbf{z}}_c\|_2^2 \quad \text{s.t.} \quad \mathbf{Q} \text{ is orthogonal.} \tag{5}$$

Although this is an optimization problem, it can be solved as follows:

$$\|\hat{\mathbf{z}}_s\mathbf{Q} - \hat{\mathbf{z}}_c\|_2^2 = \operatorname{tr}\left(\hat{\mathbf{z}}_s^T\hat{\mathbf{z}}_s + \hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_c\right) - 2\operatorname{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right). \tag{6}$$

Since, $tr(z_s^Tz_s + z_c^Tz_c)$ term is independent of Q, so eq. (5) becomes:

$$\underset{\mathbf{Q}}{\arg\max} \operatorname{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right) \quad \text{s.t.} \quad \mathbf{Q} \text{ is orthogonal.} \tag{7}$$

Using singular value decomposition of $z_c^Tz_s = USV^T$ and cyclic property of trace we have:

$$\begin{aligned}\operatorname{tr}\left(\hat{\mathbf{z}}_c^T\hat{\mathbf{z}}_s\mathbf{Q}\right) &= \operatorname{tr}\left(\mathbf{USV}^T\mathbf{Q}\right) \\ &= \operatorname{tr}\left(\mathbf{SV}^T\mathbf{QU}\right) \\ &= \operatorname{tr}\left(\mathbf{SH}\right).\end{aligned} \tag{8}$$

Here, $H = V^TQU$ is an orthogonal matrix, as it is product of orthogonal matrices. Since, S is a diagonal matrix, so in order to maximize $tr(SH)$, the diagonal values of H need to equal to 1. Now, we have:

$$\begin{aligned}&\mathbf{H} = \mathbf{V}^T\mathbf{QU} = \mathbf{I} \\ &\text{or}, \qquad \mathbf{Q} = \mathbf{VU}^T.\end{aligned} \tag{9}$$

Step-IV: Alignment. After obtaining rotation matrix Q, we scale and shift style point cloud with respect to the original content features in the following way:

$$\mathbf{z}_{sc} = \|\mathbf{z}_c\|_F\hat{\mathbf{z}}_s\mathbf{Q} + \boldsymbol{\mu}_c \tag{10}$$

z_{sc} is the final styled feature. This alignment makes style features to adapt content structure while keeping its local and global patterns intact.

4 Implementation details

4.1 Comparing with released source codes

The reproduction code is based on the implementation of AdaIN[9], where the style transfer process is based on the use of moment matching. The reproduction work is mainly based on the addition of the rigid alignment method proposed by the authors to AdaIN[9].

4.2 Main contributions

1) The authors implement style transfer by using rigid alignment, which is different from traditional style transfer methods that rely on statistical matching of features.

2) A closed-form solution to the style transfer problem is provided.

3) The proposed method achieves a real-time style transfer effect without introducing content distortion.

5 Results and analysis

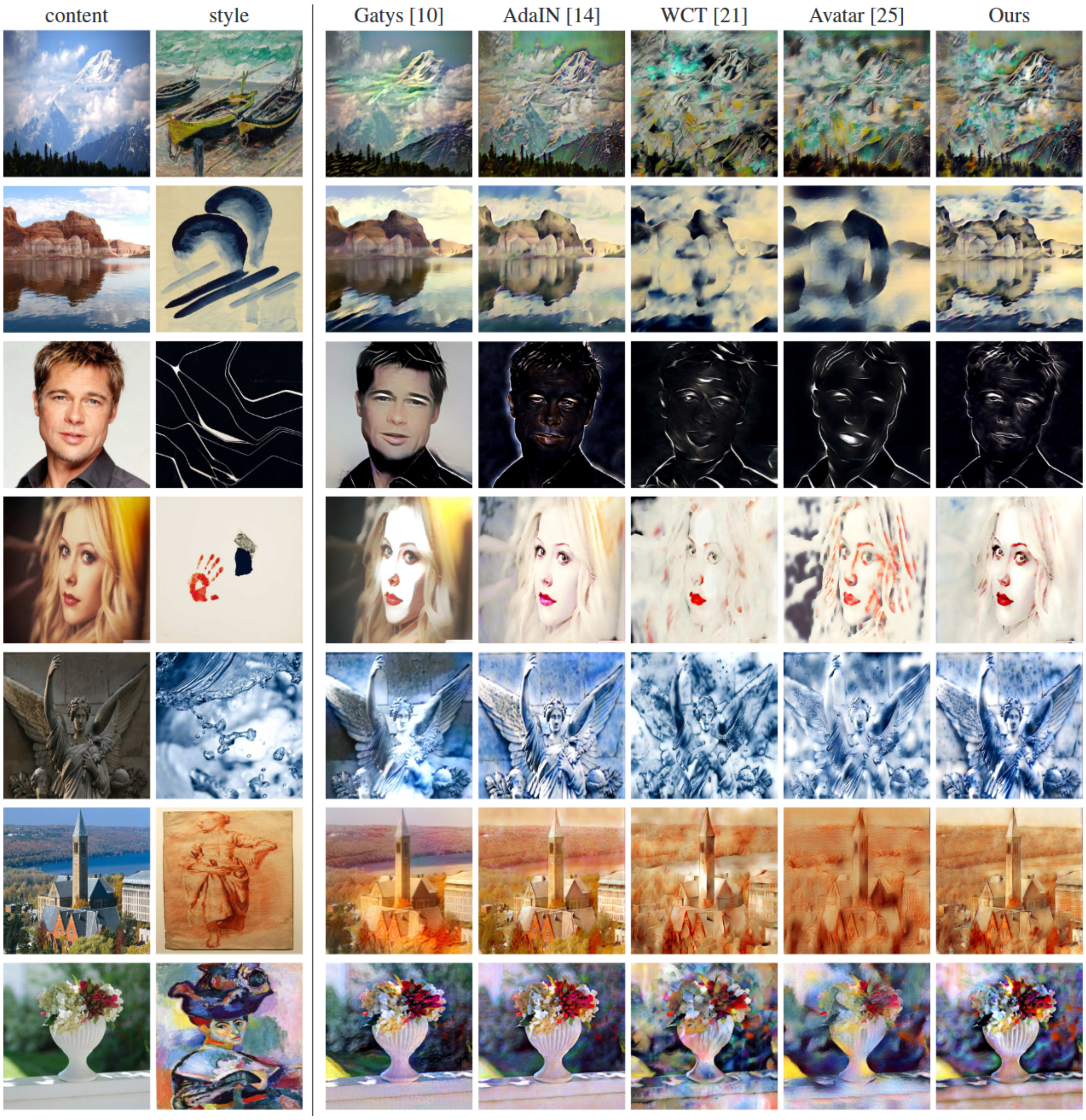

Figure 1: Experimental results

The experimental results are compared with two types of arbitrary style conversion methods. The first one is based on an iterative optimization approach[1] and the second one is a fast arbitrary style conversion approach [[10],[6],[9]]. The results of these stylizations are shown in Figure 1.

Although optimization-based approach[1] performs arbitrary style transfer, it requires slow optimization for this. Moreover, it suffers from getting stuck at a bad local minimum. This results in visually unsatisfied style transfer results, as shown in the third and fourth rows. AdaIN[9] addresses the issue of local minima along with efficiency but fails to capture the style patterns. For instance, in the third row, the styled image contains colors from the content, such as red color on the lips. Contrary to this, WCT[10] and Avatar-Net[6] perform very well in capturing the style patterns by matching second-order statistics and the latter one by normalized patch swapping. However, both methods fail to maintain the content structure in the stylized results. For instance, in the first row, WCT[10] completely destroys the content structure: mountains and clouds are indistinguishable. Similarly, in the second and fifth row, content image details are too distorted. Although AvatarNet[6] performs better than WCT[10] as in the first and fifth rows, it fails too in maintaining content information, as shown in the second and sixth rows. In the second row, the styled image does not even have any content information.

On the other hand, the proposed method not only captures style patterns similar to WCT[10] and Avatar-Net[6] but also maintains the content structure perfectly as shown in the first, second, and fifth row where the other two failed.

6 Conclusion and future work

In this work, the authors address the problem of content distortion by rigidly aligning stylistic features with content features without sacrificing the stylistic patterns in the stylized image. But the rigid alignment method provides a closed-form solution. This results in a loss of diversity in the final style transfer results. As a further direction, one can increase the diversity of the transfer by adding noise. An hourglass structure similar to Avatar-Net can also be trained to replace multiple autoencoders for multi-level style transfer to obtain better efficiency.

References

[1] GATYS L A, ECKER A S, BETHGE M. Image style transfer using convolutional neural networks[J]., 2016: 2414-2423.

[2] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition [J]. arXiv preprint arXiv:1409.1556, 2014.

[3] DUMOULIN V, SHLENS J, KUDLUR M. A learned representation for artistic style[J]. arXiv preprint arXiv:1610.07629, 2016.

[4] CHEN D, YUAN L, LIAO J, et al. Stylebank: An explicit representation for neural image style transfer [J]., 2017: 1897-1906.

[5] LI Y, FANG C, YANG J, et al. Diversified texture synthesis with feed-forward networks[J]., 2017: 3920-3928.

[6] SHEN F, YAN S, ZENG G. Neural style transfer via meta networks[J]., 2018: 8061-8069.

[7] ZHANG H, DANA K. Multi-style generative network for real-time transfer[J]., 2018.

[8] JING Y, YANG Y, FENG Z, et al. Neural style transfer: A review[J]. IEEE transactions on visualization and computer graphics, 2019, 26(11): 3365-3385.

[9] HUANG X, BELONGIE S. Arbitrary style transfer in real-time with adaptive instance normalization [J]., 2017: 1501-1510.

[10] LI Y, FANG C, YANG J, et al. Universal style transfer via feature transforms[J]. Advances in neural information processing systems, 2017, 30.

[11] CHEN T Q, SCHMIDT M. Fast patch-based style transfer of arbitrary style[J]. arXiv preprint arXiv:1612.04337, 2016.

[12] SHENG L, LIN Z, SHAO J, et al. Avatar-net: Multi-scale zero-shot style transfer by feature decoration [J]., 2018: 8242-8250.

[13] GU S, CHEN C, LIAO J, et al. Arbitrary style transfer with deep feature reshuffle[J]., 2018: 8222-8231.

[14] JOHNSON J, ALAHI A, FEI-FEI L. Perceptual losses for real-time style transfer and super-resolution [J]., 2016: 694-711.