

A Comparative Study of CNN- and Transformer-Based Visual Style Transfer

Hua-Peng Wei¹ (魏华鹏), Ying-Ying Deng² (邓盈盈), Fan Tang^{1,*} (唐帆), *Member, CCF*
Xing-Jia Pan³ (潘兴甲), and Wei-Ming Dong² (董未名), *Member, CCF, ACM, IEEE*

¹*School of Artificial Intelligence, Jilin University, Changchun 130012, China*

²*National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China*

³*Youtu Laboratory, Tencent Incorporated, Shanghai 200233, China*

E-mail: weihp20@mails.jlu.edu.cn; dengyingying2017@ia.ac.cn; tangfan@jlu.edu.cn; noahpan@tencent.com
weiming.dong@ia.ac.cn

Received January 5, 2022; accepted April 24, 2022.

Abstract Vision Transformer has shown impressive performance on the image classification tasks. Observing that most existing visual style transfer (VST) algorithms are based on the texture-biased convolution neural network (CNN), here raises the question of whether the shape-biased Vision Transformer can perform style transfer as CNN. In this work, we focus on comparing and analyzing the shape bias between CNN- and transformer-based models from the view of VST tasks. For comprehensive comparisons, we propose three kinds of transformer-based visual style transfer (Tr-VST) methods (Tr-NST for optimization-based VST, Tr-WCT for reconstruction-based VST and Tr-AdaIN for perceptual-based VST). By engaging three mainstream VST methods in the transformer pipeline, we show that transformer-based models pre-trained on ImageNet are not proper for style transfer methods. Due to the strong shape bias of the transformer-based models, these Tr-VST methods cannot render style patterns. We further analyze the shape bias by considering the influence of the learned parameters and the structure design. Results prove that with proper style supervision, the transformer can learn similar texture-biased features as CNN does. With the reduced shape bias in the transformer encoder, Tr-VST methods can generate higher-quality results compared with state-of-the-art VST methods.

Keywords transformer, convolution neural network, visual style transfer, comparative study

1 Introduction

Visual style transfer (VST) refers to the task which renders given visual media into the desired style while preserving its content structure. Since Gatys *et al.* [1] proposed the Gram matrix based style measurement, researchers have been focusing on performing style transfer with a convolutional neural network (CNN) in an end-to-end manner. Optimization-based or feed-forward approaches are two kinds of mainstream VST approaches focusing on directly stylizing input visual media without involving specific domain

knowledge of arts. Optimization-based methods [1, 2] achieve style transfer in an iterative optimization process. These methods can produce high-quality images but are computationally expensive due to the direct backward on image pixels. Feed-forward approaches [3–6] focus on arbitrary or universal style transfer using an encoder-decoder pipeline to generate the visual media with remarkable quality for different styles. Feed-forward methods can be further divided into two categories according to the training strategy, perceptual-based methods [3, 6] and reconstruction-based methods [4]. These VST methods are based on

Regular Paper

Special Section of CVM 2022

The work was supported by the National Key Research and Development Program of China under Grant No. 2020AAA0106200, the National Natural Science Foundation of China under Grant Nos. 62102162, 61832016, U20B2070, and 6210070958, the CASIA-Tencent Youtu Joint Research Project, and the Open Projects Program of the National Laboratory of Pattern Recognition.

*Corresponding Author

©Institute of Computing Technology, Chinese Academy of Sciences 2022

features extracted from CNN, which is pre-trained on image classification tasks.

Recently, Transformer^[7] has provided an alternative architecture for CNN in the research field of visual media computing. Researchers adopted the sequence-to-sequence architecture in the field of visual media computing. Represented by Vision Transformer (ViT)^[8], transformer-based models outperform CNN-based models in various CV tasks^[9–12]. The success of the transformer has heightened the need for clearing the properties of the transformer. Overall, researchers consider the difference between the ViT and the CNN backbone caused by the long-range dependencies and the local-receptive field. In-depth comparisons^[13,14] pointed out that CNN is texture-biased while Vision Transformer is shape-biased, as shown in Figs.1(a) and 1(c).

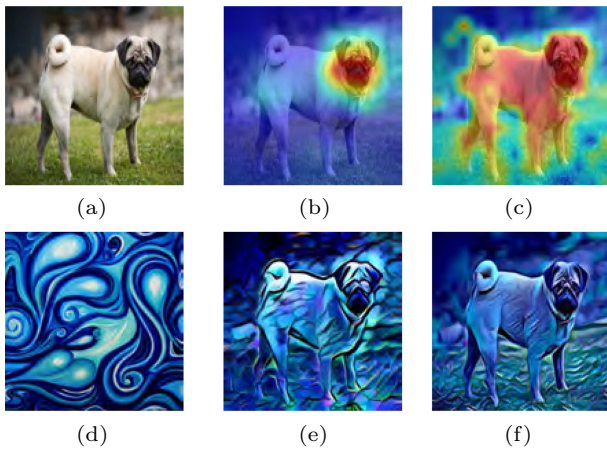


Fig. 1. Visual comparisons for CNN- and transformer-based structures. (a) Content. (b) VGG CAM^[15]. (c) ViT CAM. (d) Style. (e) AdaIN^[3]. (f) Tr-AdaIN.

Tuli *et al.*^[13] compared CNN and the transformer on the view of human vision similarity. By measuring and analyzing the error consistency for different networks, they found that in contrast to CNN, the transformer behaves with a higher shape bias and is largely closer to human vision. Their experiments showed that the transformer could maintain the accuracy and increase the shape bias when fine-tuned on augmented data, whereas CNN drops performance simultaneously. Naseer *et al.*^[14] studied the shape and texture bias of the transformer and CNN. They also pointed out that the strong shape bias leads the transformer to perform better than CNN and is comparable to humans on shape recognition. Besides, they found that the shape bias makes the transformer more robust towards domain shifts.

However, all these conclusions are made upon image classification tasks. Few studies have investigated comparisons between CNN and the transformer on generative tasks, such as style transfer. Shape and texture, corresponding to content and style, respectively, are considered to be the basic constituent elements in the VST process. From the class activation map (CAM, Figs.1(b) and 1(c)), we can observe that the pre-trained ViT prefers to make predictions based on the integral shape, whereas the VGG focuses on the local patch, showing that ViT has a more substantial shape bias than CNN. As shown in Figs.1(e) and 1(f), when using similar configurations for the transfer module, different types of backbones lead to different VST results.

To figure out the cause of shape bias in Vision Transformer, we attempt to engage the architecture into typical VST algorithms (NST^[1], AdaIN^[3], and WCT^[4]). These three VST algorithms represent the current mainstream style transfer pipelines: NST represents optimization-based style transfer, AdaIN represents perceptual-based style transfer, and WCT represents reconstruction-based style transfer. By visiting the following fundamental problem which is central to both VST and the research field of computational visual media, we open up the study: is the shape bias of Vision Transformer from the model parameters or the structure? Researchers widely hold the point that different structures lead to different results. Comparing the results of the transformer-based VST and CNN-based VST approaches, we find that the typical VST approaches do not work well on the transformer structure. However, by further controlling the training configuration of the Vision Transformer encoder, we prove that the same structure may generate different results when the model parameters are different. Overall, we conduct preliminary comparisons among CNN- and transformer-based VST approaches, which may throw new light upon the research field of deep structure based approaches. Our contributions are summarized as follows.

- We engage three typical VST approaches with the transformer by proposing Tr-NST, Tr-AdaIN, and Tr-WCT. Results show that pre-trained ViTs are invalid for performing mainstream style transfer methods due to the shape bias.
- We demonstrate that the shape bias can be reduced by training with proper supervision. ViTs can be capable to capture style strokes and patterns, which leads the models towards the texture bias instead of the shape bias.
- We discuss the influence of basic transformer

modules such as position encoding and upsampling way for transformer-based VST tasks and demonstrate that SPE^[7] and MLP (Multilayer Perceptron) are not proper for transformer-based VST.

2 Related Work

2.1 Neural Style Transfer

Style transfer^[16] is a long-standing task in the image process field. Gatys *et al.*^[1] opened up the neural style transfer area, introducing the style representation by computing the Gram matrix upon deep features extracted by convolution neural networks. Johnson *et al.*^[17] and Ulyanov *et al.*^[18] trained a single feed-forward network to transfer the specific style to arbitrary content and achieved a faster inference speed than directly optimizing pixels. But these models need to be trained from scratch to render a new style. Li *et al.*^[4] proposed the whitening and coloring transformation, namely WCT, for arbitrary style transfer. Huang and Belongie^[3] introduced the adaptive instance normalization, namely AdaIN, which performs normalization on content features with mean-variance statistics from style features. Most recently, An *et al.*^[19] proposed ArtFlow based on reversible neural flow, alleviating the content leak phenomenon. Although recent developments^[5, 6, 20–23] in the field of style transfer have made outstanding achievements towards the arbitrary style and high quality, these style transfer models are still almost based on deep features from CNN. In this paper, we propose to perform style transfer with Vision Transformer^[8].

2.2 Visual Transformer

Transformer^[7] is proposed for machine translation tasks and has achieved the SoTA performance in various NLP tasks. Transformer-based models can process long-range dependencies and aggregate the information from all the tokens by utilizing attention mechanisms. Inspired by the success of transformer-based models in the NLP field, there have been many attempts to adapt the transformer architecture to computer vision tasks. Chen *et al.*^[24] introduced a sequence transformer pre-trained to auto-regressively predict pixels. Proposed by Dosovitskiy *et al.*^[8], Vision Transformer (ViT) has achieved the SoTA performance on image classification tasks and outperforms CNN-based approaches. With a pure transformer architecture, ViT processes image patches as tokens instead of pixels.

In addition to the image classification task, the transformer has been used to solve various computer vision tasks^[25], such as object detection^[9], segmentation^[26], image processing^[27–29], image generation^[11, 30] and 3D computer vision^[12]. In this paper, we investigate the ability of Vision Transformer to represent style information and preserve the content structure by performing style transfer with Vision Transformer^[8].

2.3 Comparative Study of CNN and Visual Transformer

Analyses of CNN have been proposed a lot due to its high performance in visual tasks. Following the breakthrough of the transformer in the computer vision field, there are studies paying attention to the horizontal comparison between CNN and the visual transformer. Cordonnier *et al.*^[31] proved that self-attention layers could express any convolutional layer when given sufficient heads. Besides, they observed that fully-attentional models could act as CNN-based models where the kernel pattern is learned simultaneously as the filter. Tuli *et al.*^[13] explored that Vision Transformer has a higher shape bias than CNN for image classification tasks. They concluded that attention models focus on the important part of the image for the given task and these models can neglect the noisy background to make predictions. Naseer *et al.*^[14] got similar conclusions with Tuli. They analyzed the property of ViT and found that the transformer is more biased towards the shape than CNN. The shape-biased structure makes the transformer highly robust to severe occlusions, perturbations, and domain shifts for the image classification task. All of the above studies^[13, 14, 31] on comparing the transformer and CNN explore essential properties for the image classification tasks without considering the effect on generative studies. In this paper, we are mainly concerned with the properties of transformer for generative tasks by horizontally comparing and analyzing transformers and CNN.

3 Transformer-Based VST

3.1 Formulations

Given a content image $\mathbf{I}_c \in \mathbb{R}^{H \times W \times 3}$ and a style image $\mathbf{I}_s \in \mathbb{R}^{H \times W \times 3}$, where H stands for height and W for width, our goal is to render the style from $\mathbf{I}_s$ to $\mathbf{I}_c$ for generating stylized results $\mathbf{I}_{cs} \in \mathbb{R}^{H \times W \times 3}$. We split the input content image $\mathbf{I}_c$ and the style image $\mathbf{I}_s$ into patches, and then feed these patches into a linear

projection layer to obtain sequence tokens $\mathbf{E}_c$ and $\mathbf{E}_s$ with the shape of $L \times C$, where $L = \frac{H \times W}{m \times m}$ refers to the number of tokens, m refers to the patch size and C refers to the token's feature dimension. For any given image, we can obtain its corresponding sequence tokens $\mathbf{E}$ after the linear projection layer.

Before being fed into transformer encoder blocks, the sequence tokens can be added with an optional positional embedding. We adopt sinusoidal position embedding (SPE) [7] and learnable position embedding (LPE) [8] as alternative position encoding methods. Given the input position index pos , SPE can be calculated as:

$$SPE(pos, 2 \times idx) = \sin\left(\frac{pos}{10000^{2 \times idx/C}}\right),$$

$$SPE(pos, 2 \times idx + 1) = \cos\left(\frac{pos}{10000^{2 \times idx/C}}\right),$$

where pos is the token position in the sequence and idx is the index of the token dimension. LPE has the same shape as sequence tokens $\mathbf{E}$, which can also be added to the tokens to retain the positional information. Given a sequence of tokens $\mathbf{E}$ added with optional position embedding, each encoder block can be denoted in Fig.2(a). The multi-head self-attention can be calculated as:

$$MultiHeadAttention(\mathbf{Q}, \mathbf{K}, \mathbf{V})$$

$$= Concat(head_1(\mathbf{Q}, \mathbf{K}, \mathbf{V}), \dots, head_h(\mathbf{Q}, \mathbf{K}, \mathbf{V}))\mathbf{W}^o,$$

where $head_i = Attention(\mathbf{Q}\mathbf{W}_i^q, \mathbf{K}\mathbf{W}_i^k, \mathbf{V}\mathbf{W}_i^v)$, $i = \{1, \dots, h\}$, h refers to the number of attention heads, $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{H \times W \times 3}$ are input query, key and value embedding for attention input, respectively, $\mathbf{W}_i^q \in \mathbb{R}^{C \times C}$,

$\mathbf{W}_i^k \in \mathbb{R}^{C \times C}$, $\mathbf{W}_i^v \in \mathbb{R}^{C \times C}$ and $\mathbf{W}_i^o \in \mathbb{R}^{C \times C}$ are learnable parameters for projections, and $Attention(\cdot)$ refers to the attention operation introduced in [7]. The complete transformer encoder block can be formulated as:

$$\mathbf{X} = MultiHeadAttention(\mathbf{E}_i, \mathbf{E}_i, \mathbf{E}_i) + \mathbf{E}_i,$$

$$\mathbf{X}' = FFN(\mathbf{X}) + \mathbf{X},$$

where $FFN(\cdot)$ refers to the feed-forward network, $i = \{1, \dots, N_l\}$ is the layer index, N_l is the number of blocks, and $\mathbf{E}_i$ is the i -th layer's sequence token.

Similarly, each decoder block (as shown in Fig.3) can be formulated as:

$$\mathbf{Y} = MultiHeadAttention(\mathbf{E}_j, \mathbf{E}_j, \mathbf{E}_j) + \mathbf{E}_j,$$

$$\mathbf{Y}' = MultiHeadAttention(\mathbf{Y}, \mathbf{Z}, \mathbf{Z}) + \mathbf{Y},$$

$$\mathbf{Y}'' = FFN(\mathbf{Y}') + \mathbf{Y}',$$

where $\mathbf{Z}$ refers to the encoded sequence tokens from the transformer encoder when reconstructing images or Tr-AdaIN and Tr-WCT when performing style transfer, $j = 1, \dots, N_l$ is the layer index, and $\mathbf{E}_j$ is the j -th layer's sequence token. We apply the layer normalization in the way of Pre-LN [32].

3.2 Tr-NST

We follow Gatys *et al.* [1] to perform optimization-based style transfer by proposing Tr-NST. As shown in Fig.2(b), instead of extracting features from a pre-trained VGG [33], we utilize the transformer encoder features to compute the transformer-based perceptual loss. Generally, the feature sizes from different levels

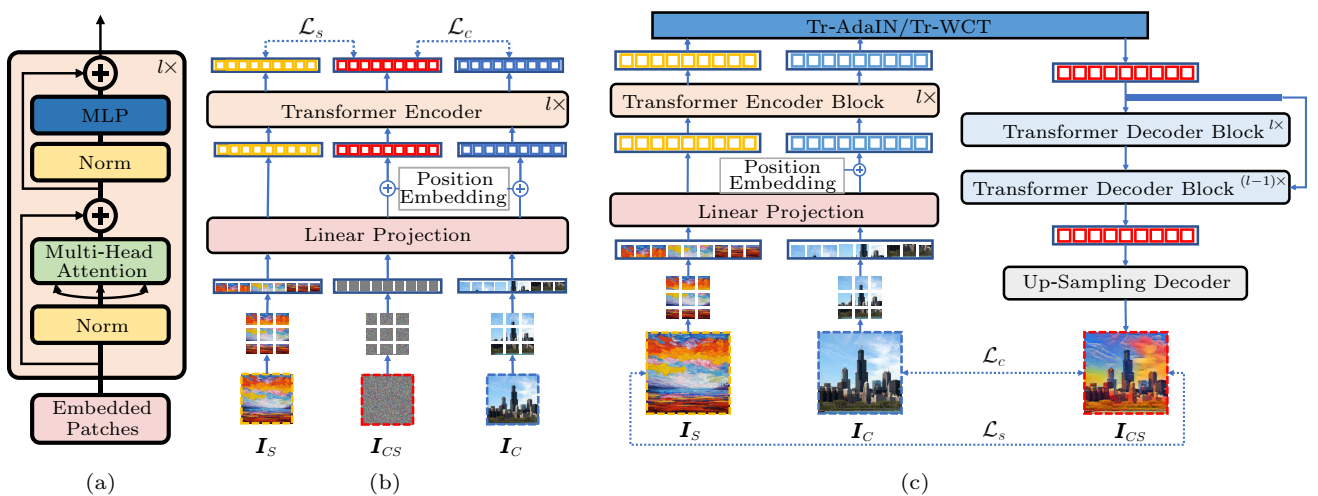


Fig.2. Model architectures for the three proposed transformer-based VST approaches. (a) Encoder block. (b) Tr-NST. (c) Tr-AdaIN/Tr-WCT. $l \times$ means l times, and $(l-1) \times$ means $(l-1)$ times.

are the same for the specific input. The content perceptual loss based on transformer features is defined as:

$$\mathcal{L}_c = \frac{1}{N_l \times CL} \sum_{i=0}^{N_l} \|re(\phi_i(\mathbf{E}_{cs})) - re(\phi_i(\mathbf{E}_c))\|_2,$$

where $re(\cdot)$ is used to reshape $\mathbf{E}$ into the shape of $C \times \frac{H}{m} \times \frac{W}{m}$ and $\phi_i(\cdot)$ denotes the features extracted from the i -th level transformer encoder block. Similarly, we define the style perceptual loss based on the transformer feature as:

$$\mathcal{L}_s = \frac{1}{N_l \times CL} \sum_{i=0}^{N_l} \|\mathbf{G}(\phi_i(\mathbf{I}_{cs})) - \mathbf{G}(\phi_i(\mathbf{I}_c))\|_2,$$

where $\mathbf{G}(\cdot) \in \mathbb{R}^{C \times C}$ denotes the feature's Gram matrix, which is defined as:

$$\mathbf{G}(\mathbf{E}) = \sum \mathbf{E}^T \mathbf{E}.$$

The total perceptual loss based on the transformer feature is defined as:

$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_s, \quad (1)$$

where λ is a weight for balancing two losses. We optimize an image $\mathbf{I}_{cs} \in \mathbb{R}^{H \times W \times 3}$ using $\mathcal{L}$ where $\mathcal{L}_c$ and $\mathcal{L}_s$ are computed with pairs $(\mathbf{I}_c, \mathbf{I}_{cs})$ and $(\mathbf{I}_s, \mathbf{I}_{cs})$, respectively. In our experiments, we use features extracted from the last five blocks in the ViT.

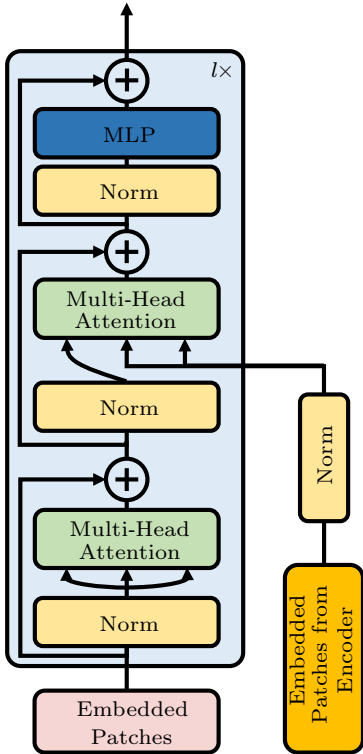


Fig.3. Transformer decoder block.

3.3 Tr-WCT

To perform arbitrary style transfer with the transformer, we add an additional transformer decoder for the style and content representation coupling and reconstruction. The decoder shares the same data shape as the encoder. The transformer decoder block has been introduced in Subsection 3.1 in detail. We propose Tr-WCT by engaging whitening and coloring transform (WCT) into the transformer encoder-decoder structure.

Following the single-level WCT [4], we also perform WCT in the feature space. Given the content sequence embedding $\mathbf{E}_c$ and the style embedding $\mathbf{E}_s$, we first normalize $\mathbf{E}_c$ to be zero-centered and then obtain the whitened content feature $\hat{\mathbf{E}}_c$ by:

$$\hat{\mathbf{E}}_c = \mathbf{E}_c \mathbf{D}_c^{-\frac{1}{2}} \mathbf{E}_c^T \mathbf{E}_c,$$

where $\mathbf{D}_c$ is a diagonal matrix with the eigenvalues of the covariance matrix $\mathbf{E}_c \mathbf{E}_c^T \in \mathbb{R}^{C \times C}$, and $\mathbf{E}_c$ is the corresponding orthogonal matrix of eigenvectors, satisfying $\mathbf{E}_c \mathbf{E}_c^T = \mathbf{E}_c \mathbf{D}_c \mathbf{E}_c^T$. After zero-centering $\mathbf{E}_s$, the colored feature $\hat{\mathbf{E}}_{cs}$ can be obtained by performing coloring transform, which is defined as:

$$\hat{\mathbf{E}}_{cs} = \mathbf{E}_s \mathbf{D}_s^{-\frac{1}{2}} \mathbf{E}_s^T \hat{\mathbf{E}}_c,$$

where $\mathbf{D}_s$ is a diagonal matrix with the eigenvalues of the covariance matrix $\mathbf{E}_s \mathbf{E}_s^T \in \mathbb{R}^{C \times C}$, and $\mathbf{E}_s$ is the corresponding orthogonal matrix of eigenvectors. Finally, $\hat{\mathbf{E}}_{cs}$ is re-centered by adding the mean vector $\mathbf{m}_s$ of the style:

$$\hat{\mathbf{E}}_{cs} = \hat{\mathbf{E}}_{cs} + \mathbf{m}_s.$$

Then $\hat{\mathbf{E}}_{cs}$ is fed into the transformer decoder to obtain the stylized image. When trained with a pre-trained ViT for reconstructing an input image, we use the pixel reconstruction loss [34]:

$$\mathcal{L}_{\text{rec}} = \|\mathbf{I}_o - \mathbf{I}_i\|_2^2,$$

where $\mathbf{I}_i$ and $\mathbf{I}_o$ are the input image and the reconstruction output, respectively. When retraining the whole auto-encoder with the strategy introduced in Subsection 4.3, we use the perceptual loss introduced in Subsection 3.2 to alleviate the shape bias.

3.4 Tr-AdaIN

Another mainstream arbitrary style transfer method is adaptive instance normalization (AdaIN) [3].

We propose Tr-AdaIN by replacing the Tr-WCT transform with AdaIN and retaining the decoder architecture. Given the content feature E_c and the style feature E_s , the output stylized feature E_{cs} from AdaIN is computed as $E_{cs} = \text{AdaIN}(E_c, E_s) = \sigma_{E_s}(\frac{E_c - \mu_{E_c}}{\sigma_{E_c}}) + \mu_{E_s}$, where σ_{E_c} , μ_{E_c} , σ_{E_s} and μ_{E_s} refer to the dimension-wise variance and mean of E_c and E_s , respectively. The detailed comparison against AdaIN and Tr-AdaIN is also illustrated in Fig.4.

For the model optimization, we use an additional pre-trained VGG [33] to support matching the mean and the variance of VGG features. The content loss is defined as $\mathcal{L}_c = \frac{1}{N_l} \sum_{i=0}^{N_l} \|(\Phi_i(E_{cs})) - (\Phi_i(E_c))\|_2$, where $\Phi(\cdot)$ denotes the output feature of the VGG relu layer.

The style loss is defined as:

$$\mathcal{L}_s = \frac{1}{N_l} \sum_{i=0}^{N_l} \|\mu(\Phi_i(I_{cs})) - \mu(\Phi_i(I_s))\|_2 + \sum_{i=0}^{N_l} \|\sigma(\Phi_i(I_{cs})) - \sigma(\Phi_i(I_s))\|_2,$$

where $\mu(\cdot)$ denotes the mean of features, and $\sigma(\cdot)$ denotes the variance of features.

$$\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_s. \quad (2)$$

In our experiments, we use the relu1_1, relu2_1, relu3_1, relu4_1 and relu5_1 layers of the pre-trained VGG-19.

4 Experiments

4.1 Setup

Datasets. We use MSCOCO [35] and WikiArt [36] as the content and style dataset respectively. During the training stage, each image is resized to 512×512 and then randomly cropped into 256×256 by default. The image can be any size during the test stage.

Implementation Details. We use two NVIDIA RTX 3090 cards^① to train our model using the Adam optimizer [37] with a learning rate at 5×10^{-4} . For Tr-AdaIN and Tr-WCT, the model is trained for 160 000 iterations. The warmup strategy is used during the first

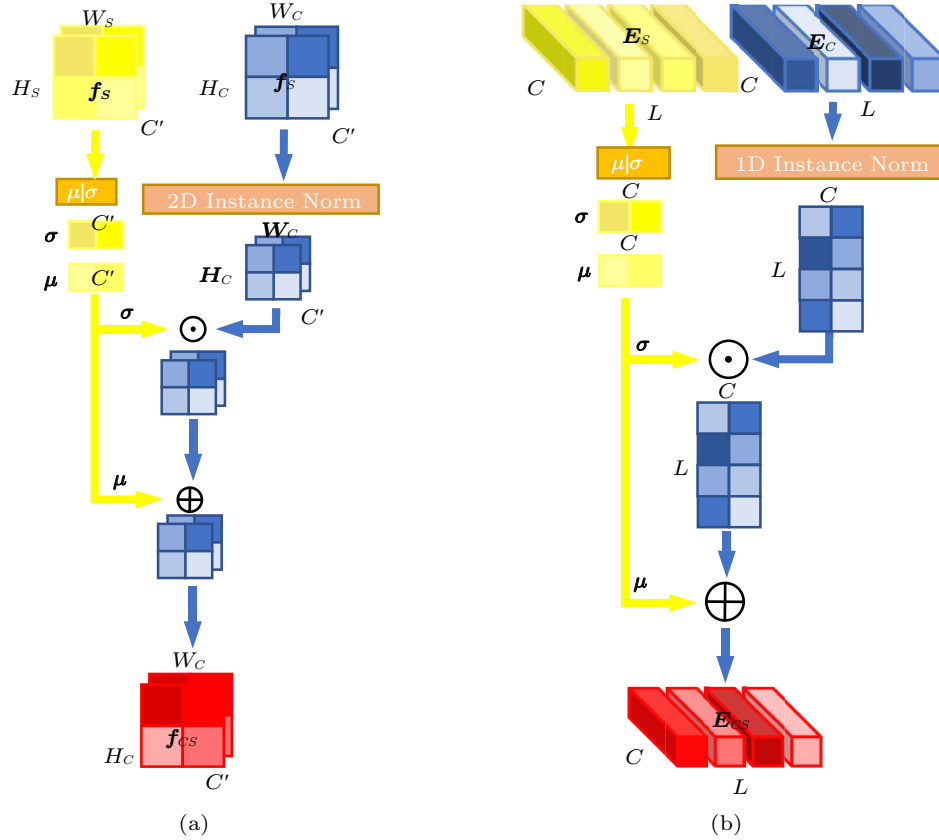


Fig.4. (a) AdaIN [3]. (b) Tr-AdaIN. For AdaIN, C' refers to the channel number of CNN features; for Tr-AdaIN, C refers to the feature dimension of tokens.

^①<https://www.nvidia.com/en-us/geforce/graphics-cards/30-series/rtx-3090-3090ti/>, Apr. 2022.

10 000 iterations. For Tr-NST, I_{cs} is trained for 4000 iterations. We use $\lambda = 1$ in (1) and (2) for perceptual loss; for the decoder training in Tr-WCT, we use a single-pixel reconstruction loss. We count the inference speed of each method, as shown in Table 1.

Table 1. Inference Speed (FPS) and FLOPs of Different Methods on Image Size 512×512

Method	Frames per Second (FPS)	FLOPs (G)
AdaIN	7.204	266.693
Tr-AdaIN	5.666	231.628
WCT	4.225	189.991
Tr-WCT	3.281	231.628
NST	10.269	63.339
Tr-NST	9.922	87.578

Note: For NST and Tr-NST, the FLOPs only count a single iteration; the inference speed describes the number of iterations per second. For other methods, the inference describes FPS and the FLOPs are computed for single pairs of content and style images.

Different from traditional VST approaches where the encoder is pre-trained on image classification tasks and not involved in the training process, we discuss the differences between CNN-based and transformer-based VST approaches by controlling the parameter training of the encoder. Given the transformer-based visual style transfer methods described in Section 3, we implement Tr-NST-p, Tr-AdaIN-p, and Tr-WCT-p using pre-trained ViT as the encoder. The training process only influences the decoder for Tr-AdaIN-p and Tr-WCT-p. We further implement Tr-NST, Tr-AdaIN, and Tr-WCT by randomly initializing and training the encoder. The encoders of Tr-AdaIN and Tr-WCT are optimized based on the losses calculated on the pre-trained VGG. After training, we use the encoder of Tr-AdaIN for Tr-NST. In general, the parameters of the encoder in Tr-VST and Tr-VST-p are affected by a pre-trained Vision Transformer and a pre-trained CNN, respectively. Since researchers have pointed out that the self-attention module can mimic a convolution layer [31],

we try to figure out whether the influence of the transformer’s structure could be reduced with a properly set training configuration.

4.2 Are CNN-Based VST Approaches Proper for Transformer?

In this subsection, we discuss the influence of adopting the transformer structure for VST tasks. As shown in Fig. 5, the CNN-based methods (NST, AdaIN, and WCT) can generate stylized images with high quality, but the methods adopted for the transformer (Tr-NST-p, Tr-AdaIN-p, and Tr-WCT-p) fail to some extent. Proposed by Gatys et al. [1], CNN-based NST is regarded as the benchmark in this field. However, Tr-NST-p fails to render any style elements from the style image, disturbs pixels and produces boundaries between patches. In the sense of style transfer, Tr-NST-p behaves no effect at all. AdaIN [3] provides a simple but effective way to match the style factors to the content image. Although Tr-AdaIN-p can also transfer the styles, the results contain boundary artifacts and unreasonable patterns.

WCT vs Tr-WCT-p. As a learning-free transformation, WCT can be performed with a reconstruction auto-encoder, generating high-quality stylized images in a coarse-to-fine manner. In this paper, we only consider single-level stylization. Compared with WCT, Tr-WCT-p generates unexpected results, only retaining the incomplete outlines from the content image and little style pattern from the style image. Such results motivate us to go through the intermediate results of Tr-WCT-p.

To test and verify the conclusion, we reconstruct images with whitened features, following the original WCT. The whiten results generated with Tr-WCT-p and VGG are shown in Fig. 6. When performed on VGG features, the whitening transform removes style information by removing the correlation among chan-

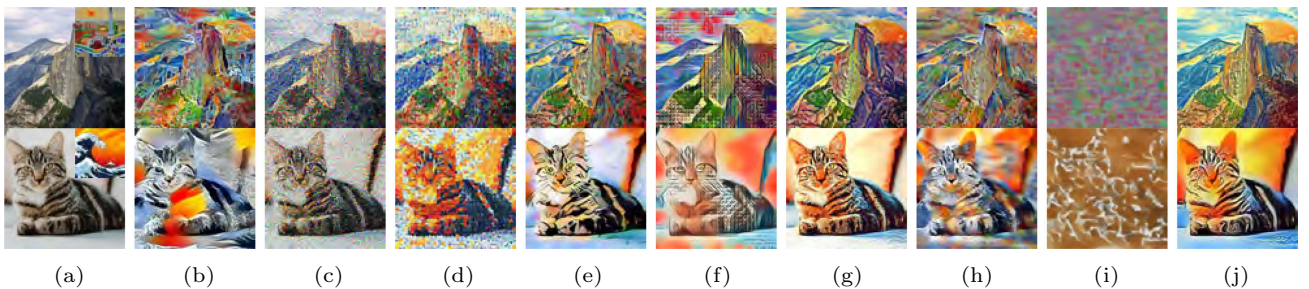


Fig. 5. Visual style transfer results using different methods. (a) Content and style. (b) NST [1]. (c) Tr-NST-p. (d) Tr-NST. (e) AdaIN [3]. (f) Tr-AdaIN-p. (g) Tr-AdaIN. (h) WCT [4]. (i) Tr-WCT-p. (j) Tr-WCT.

nels. But this operation fails to disentangle the style and the content on transformer features which do not provide sufficient style information. The stylized images from CNN can render the style from the reference image and save the content simultaneously, whereas the transformer-based WCT cannot even preserve the detailed content, only maintaining the blurred outline.

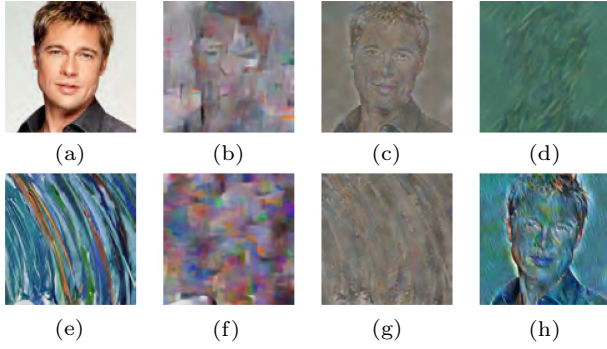


Fig.6. Stylized results and inverted features for WCT and Tr-WCT-p. (a) I_c . (b) Whiten I_c by Tr-WCT-p. (c) Whiten I_c by WCT. (d) I_{cs} by Tr-WCT-p. (e) I_s . (f) Whiten I_s by Tr-WCT-p. (g) Whiten I_s by WCT. (h) I_{cs} by WCT.

All the phenomena mentioned above show that the transformer pre-trained on ImageNet has a strong shape preference. We argue that the pre-trained ViT is not valid for extracting features to perform style transfer due to the strong bias from both parameters learned from the classification task and the structure design. On the one hand, the transformer can obtain long-distance dependence, i.e., the global receptive field with the help of multi-head attention mechanism. The multi-head attention mechanism is position-agnostic, which computes the attention score over all the sequence tokens, focusing on the relationships between tokens. These relationships between image patches are often reflected in the form of shape. Although Cordonnier *et al.* [30] proved that a self-attention could express any convolution, the classification task does not provide a valid guidance to the mimic, making the strong bias and the reconstruction error. On the other hand, all the pixels in the same block are treated as a token, losing both the spatial information and the texture detail. In Subsection 4.3, we will give a simple but efficient solution to alleviate this shape bias phenomenon.

4.3 Is the Shape Bias from Model Parameters or Model Structure?

A shape-biased model prefers to make predictions relying more on the shapes; on the contrary, a texture-biased model makes predictions towards the texture

more than the shape. Baker *et al.* [38] introduced that CNN has been shown to have stronger texture bias than shape bias in the image classification task. Tuli *et al.* [13] introduced that ViT pre-trained on the ImageNet dataset has a stronger shape bias than traditional CNN (like ResNet [39]). The ViT family and ResNet are trained on both the ImageNet-21K and ILSVRC-2012 [40] datasets. When testing on the stylized ImageNet [41], ResNet shows poorer performance, whereas ViT can maintain part of the accuracy. Naseer *et al.* [14] found that ViTs with higher shape bias perform not only better than ResNet but also more robustly. Following DeiT [42], Naseer *et al.* [14] obtained a ViT with stronger shape bias with the shape distillation training strategy.

For VST tasks, we tend to seek a texture-biased model rather than a shape-biased one. A model with a strong texture bias can capture various style elements like patterns and strokes, thereby representing style factors. For transformer-based VST approaches, we suppose that part of the shape bias comes from the parameters, while the model structure provides an additional part on this trend. We propose Tr-NST, Tr-AdaIN, and Tr-WCT by adding constraints to the transformer encoders by an additional pre-trained VGG.

We show the detailed qualitative examples in Fig.7. Tr-AdaIN succeeds to fuse the content feature E_c with the style feature E_s from the transformer encoder. The decoder in Tr-AdaIN maps the fused feature E_{cs} back to the image domain, generating the stylized image I_{cs} . Compared with the failed cases generated by Tr-AdaIN-p, Tr-AdaIN outperforms original AdaIN results based on pre-trained VGG. By performing Tr-WCT, we obtain the stylized images which also achieve the comparable quality with original WCT based on pre-trained VGG.

Tr-NST vs Tr-NST-p. As shown in Fig.5, results of Tr-NST are still not visually good compared with the original NST. Fig.8 shows another example. Tr-NST-p fails to extract required style information, and the optimization only adds noise to the content image. In contrast, Tr-NST can optimize the content image using the transformer encoder in Tr-AdaIN, rendering and coloring towards the stylized image. Note that boundary artifacts occur in Tr-NST and Tr-NST-p results. This observation can be attributed to the reason that all pixels within a certain patch are optimized in the same direction.

Comparison with Other Transformer-Based VST Work. Recently, several studies have also investigated

performing VST with the transformer^[20,23]. SANet^[20] introduces the attention mechanism as a style transfer module, utilizing normalized content and style features to calculate the attention map. StyleFormer^[23] improves the transformer-driven style composition ap-

proach which can learn content-consistent style composition for the style transfer task. SANet and StyleFormer achieve the SOTA stylization results, but these studies^[20,23] are still based on features extracted from a fixed VGG network^[33]. The Tr-VST methods are



Fig.7. Comparisons of visual style transfer results. (a) Style. (b) Content. (c) Tr-AdaIN. (d) StyleFormer^[23]. (e) ArtFlow^[19]. (f) MCCNet^[6]. (g) Linear^[21]. (h) SANet^[20]. (i) WCT^[4]. (j) AdaIN^[3]. The first row shows style images, and the second row shows content images. The rest rows are stylized results of different methods.



Fig.7. Comparisons of visual style transfer results. (Continued) (a) Style. (b) Content. (c) Tr-AdaIN. (d) StyleFormer [23]. (e) ArtFlow [19]. (f) MCCNet [6]. (g) Linear [21]. (h) SANet [20]. (i) WCT [4]. (j) AdaIN [3]. The first row shows style images, and the second row shows content images. The rest rows are stylized results of different methods.

designed to investigate the discrepancy in representing style patterns between the transformer and CNN. For Tr-VST, the transformer is used as a feature extractor instead of the style transfer module to be compared with VGG-based VST approaches.

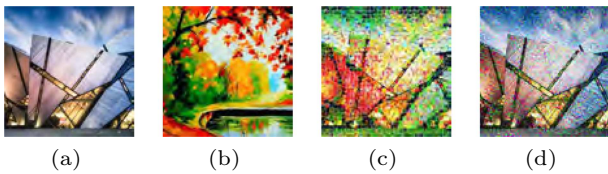


Fig.8. Visual comparisons against Tr-NST and Tr-NST-p. (a) I_c . (b) I_s . (c) Tr-NST. (d) Tr-NST-p.

4.4 Is Position Embedding Fitting for Style Transfer with Transformer?

The multi-head self-attention layer is in a position-agnostic manner and cannot naturally make use of the position in sequence tokens. Researchers introduced

position embedding to provide relative or absolute positional information. Vaswani *et al.* [7] introduced the sinusoidal positional encoding to inject positional information. Dosovitskiy *et al.* [8] added a 1D learnable embedding to the tokens to retain the sequence order.

We test SPE and LPE which are introduced in Subsection 3.1 on Tr-AdaIN. Besides, we add results generated by the model trained without position embedding. Results are shown in Fig.9. Notice that the resolution during the training stage is 256×256 ; thus other scales are unseen at training and only used for the inference stage. When performing LPE at other resolutions, we use the nearest-neighbor interpolation on PE to match the input shape. The model with LPE exhibits similar results to the model without any PE, but the one with SPE fails to maintain the consistent style rendering. SPE causes the model to learn the fixed relationship between style and position and leads to the increase of image brightness at other scales.

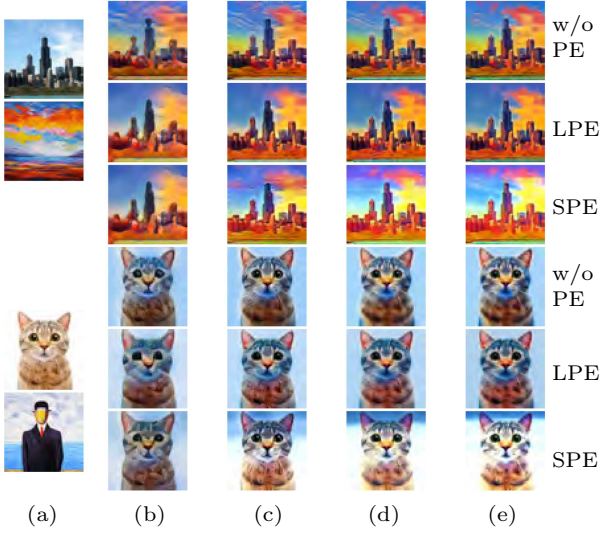


Fig.9. Visual results of different position embedding settings on Tr-AdaN. (a) Inputs. (b) 256^2 . (c) 512^2 . (d) 768^2 . (e) 1024^2 . w/o: without.

We visualize the position-wise cosine similarity of LPE and SPE on different resolutions, as shown in Fig. 10. When performing LPE on other resolutions, the nearest neighbor interpolation allows the model to perceive the relative position of the image at a larger scale, due to the consistent position embedding value at the same relative position, as shown in Fig. 10(a). When performing SPE at high resolutions, more patches are

involved compared with low-resolution training inputs. The position encoding of testing images is different from the low-resolution SPE, as shown in Fig. 10(b). The model fails to render style patterns with unseen position information, making different results at different resolutions.

4.5 Effect of Upsampling Methods for VST Tasks

An additional decode block is necessary to translate the sequence tokens back into images owing to the inconsistent shape. Recently transformer-based generative models design various decode blocks to invert the tokens into an image. TTSR^[10] chooses to stack transformers, and applies convolution layers to transform the concatenated features back into high-resolution images. IPT^[27] introduces multi-tail to deal with different tasks, and each tail uses convolution and pixel shuffle layers to up-sample the features.

We adopt CNN and MLP into our proposed transformer auto-encoder model as the upsampling block. The results are shown in Fig. 11. Observing that images generated by MLP appear severe checkboard artifacts, adjacent patches exhibit inconsistency in perception and produce distinct boundaries. MLP also fails to render complex textures, so that the same pattern appears in each patch. The same observation occurs

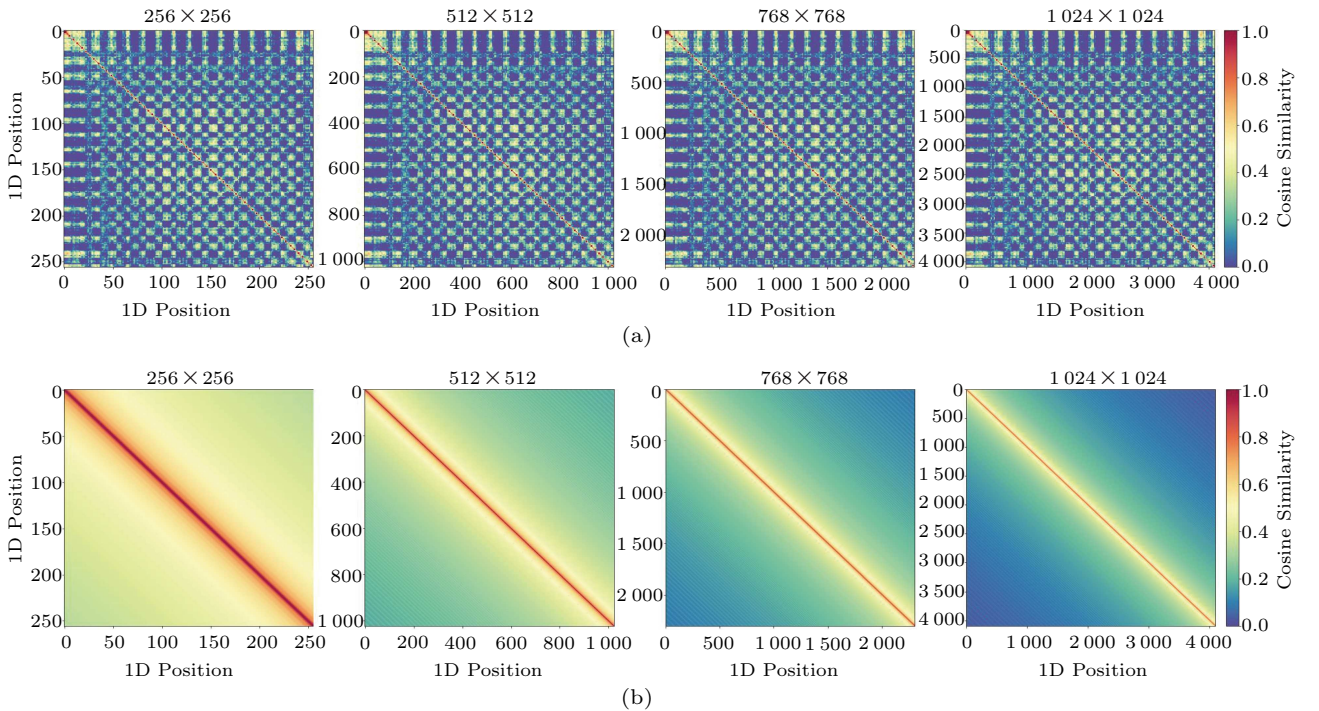


Fig.10. Visualization of position-wise cosine similarity of LPE and SPE on different inference resolutions. (a) LPE. (b) SPE.

in both reconstruction and stylization results, implying that directly using MLP is not an appropriate choice for inverting features into images.

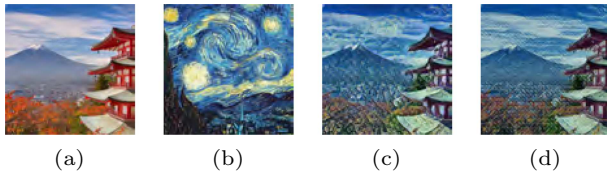


Fig.11. Visual comparisons against CNN and MLP as upsampling decoder. (a) I_c . (b) I_s . (c) Up-CNN. (d) Up-MLP.

5 Conclusions

In this paper, we compared CNN and the transformer on the view of style transfer and studied essential elements for designing transformer-based style transfer models. We adopted three typical visual style transfer algorithms (Tr-NST, Tr-AdaIN, Tr-WCT) into the transformer. We demonstrated that these original CNN-based VST methods do not work on a pre-trained ViT, due to its shape bias from both the learned parameters and the model structure design. By experimenting with setting model variations, we gave the solution for alleviating the strong shape and performing transformer-based style transfer by retraining our proposed model with additional perceptual loss. We also explored the impact of position embedding and upsampling methods on the stylization results. Tr-VST with LPE or without PE can generate similar results. But SPE is not valid for Tr-VST due to the learned relationship between the style factors with position information. Moreover, we demonstrated that combining CNN as the upsampling method is an appropriate choice for avoiding checkboard artifacts and repeated patterns. In the future, we will explore the impact of transformer design on shape bias and style transfer.

References

- [1] Gatys L A, Ecker A S, Bethge M. Image style transfer using convolutional neural networks. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp.2414-2423. DOI: [10.1109/CVPR.2016.265](https://doi.org/10.1109/CVPR.2016.265).
- [2] Kolkin N, Salavon J, Shakhnarovich G. Style transfer by relaxed optimal transport and self-similarity. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.10051-10060. DOI: [10.1109/CVPR.2019.01029](https://doi.org/10.1109/CVPR.2019.01029).
- [3] Huang X, Belongie S. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proc. the 2017 IEEE International Conference on Computer Vision*, Oct. 2017, pp.1501-1510. DOI: [10.1109/ICCV.2017.167](https://doi.org/10.1109/ICCV.2017.167).
- [4] Li Y, Fang C, Yang J, Wang Z, Lu X, Yang M H. Universal style transfer via feature transforms. In *Proc. the 31st International Conference on Neural Information Processing Systems*, December 2017, pp.385-395.
- [5] Deng Y, Tang F, Dong W, Sun W, Huang F, Xu C. Arbitrary style transfer via multi-adaptation network. In *Proc. the 28th ACM International Conference on Multimedia*, Oct. 2020, pp.2719-2727. DOI: [10.1145/3394171.3414015](https://doi.org/10.1145/3394171.3414015).
- [6] Deng Y, Tang F, Dong W, Huang H, Ma C, Xu C. Arbitrary video style transfer via multi-channel correlation. In *Proc. the 35th AAAI Conference on Artificial Intelligence*, February 2021, pp.1210-1217.
- [7] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L U, Polosukhin I. Attention is all you need. In *Proc. the 31st International Conference on Neural Information Processing Systems*, December 2017, pp.6000-6010.
- [8] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houshy N. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. the 9th International Conference on Learning Representations*, May 2021.
- [9] Carion N, Massa F, Synnaeve G, Usunier N, Kirillov A, Zagoruyko S. End-to-end object detection with transformers. In *Proc. the 16th European Conference on Computer Vision*, August 2020, pp.213-229. DOI: [10.1007/978-3-030-58452-8_13](https://doi.org/10.1007/978-3-030-58452-8_13).
- [10] Yang F, Yang H, Fu J, Lu H, Guo B. Learning texture transformer network for image super-resolution. In *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020, pp.5790-5799. DOI: [10.1109/CVPR42600.2020.00583](https://doi.org/10.1109/CVPR42600.2020.00583).
- [11] Lee K, Chang H, Jiang L, Zhang H, Tu Z, Liu C. ViTGAN: Training GANs with vision transformers. arXiv:2107.04589, 2021. <https://arxiv.org/abs/2107.04589>, Jan. 2022.
- [12] Guo M H, Cai J X, Liu Z N, Mu T J, Martin R R, Hu S M. PCT: Point cloud transformer. *Computational Visual Media*, June 2021, 7(2): 187-199. DOI: [10.1007/s41095-021-0229-5](https://doi.org/10.1007/s41095-021-0229-5).
- [13] Tuli S, Dasgupta I, Grant E, Griffiths T L. Are convolutional neural networks or transformers more like human vision? arXiv:2105.07197, 2021. <https://arxiv.org/abs/2105.07197>, Jan. 2022.
- [14] Naseer M, Ranasinghe K, Khan S, Hayat M, Khan F, Yang M H. Intriguing properties of vision transformers. In *Proc. the 35th Conference on Neural Information Processing Systems*, December 2021.
- [15] Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. Learning deep features for discriminative localization. In *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2016, pp.2921-2929. DOI: [10.1109/CVPR.2016.319](https://doi.org/10.1109/CVPR.2016.319).
- [16] Jing Y, Yang Y, Feng Z, Ye J, Yu Y, Song M. Neural style transfer: A review. *IEEE Trans. Visualization and Computer Graphics*, 2020, 26(11): 3365-3385. DOI: [10.1109/TVCG.2019.2921336](https://doi.org/10.1109/TVCG.2019.2921336).

- [17] Johnson J, Alahi A, Li F F. Perceptual losses for real-time style transfer and super-resolution. In *Proc. the 14th European Conference on Computer Vision*, Oct. 2016, pp.694-711. DOI: [10.1007/978-3-319-46475-6_43](https://doi.org/10.1007/978-3-319-46475-6_43).
- [18] Ulyanov D, Vedaldi A, Lempitsky V. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, July 2017, pp.4105-4113. DOI: [10.1109/CVPR.2017.437](https://doi.org/10.1109/CVPR.2017.437).
- [19] An J, Huang S, Song Y, Dou D, Liu W, Luo J. ArtFlow: Unbiased image style transfer via reversible neural flows. In *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2021, pp.862-871. DOI: [10.1109/CVPR46437.2021.00092](https://doi.org/10.1109/CVPR46437.2021.00092).
- [20] Park D Y, Lee K H. Arbitrary style transfer with style-attentional networks. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.5880-5888. DOI: [10.1109/CVPR.2019.00603](https://doi.org/10.1109/CVPR.2019.00603).
- [21] Li X, Liu S, Kautz J, Yang M H. Learning linear transformations for fast image and video style transfer. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019, pp.3809-3817. DOI: [10.1109/CVPR.2019.00393](https://doi.org/10.1109/CVPR.2019.00393).
- [22] Wang Z, Zhao L, Chen H, Qiu L, Mo Q, Lin S, Xing W, Lu D. Diversified arbitrary style transfer via deep feature perturbation. In *Proc. the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020, pp.7786-7795. DOI: [10.1109/CVPR42600.2020.00781](https://doi.org/10.1109/CVPR42600.2020.00781).
- [23] Wu X, Hu Z, Sheng L, Xu D. StyleFormer: Real-time arbitrary style transfer via parametric style composition. In *Proc. the 2021 IEEE/CVF International Conference on Computer Vision*, October 2021, pp.14618-14627. DOI: [10.1109/ICCV48922.2021.01435](https://doi.org/10.1109/ICCV48922.2021.01435).
- [24] Chen M, Radford A, Child R, Wu J, Jun H, Luan D, Sutskever I. Generative pretraining from pixels. In *Proc. the 37th International Conference on Machine Learning*, July 2020, pp.1691-1703.
- [25] Xu Y, Wei H, Lin M, Deng Y, Sheng K, Zhang M, Tang F, Dong W, Huang F, Xu C. Transformers in computational visual media: A survey. *Computational Visual Media*, 2022, 8(1): 33-62. DOI: [10.1007/s41095-021-0247-3](https://doi.org/10.1007/s41095-021-0247-3).
- [26] Wang Y, Xu Z, Wang X, Shen C, Cheng B, Shen H, Xia H. End-to-end video instance segmentation with transformers. In *Proc. the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2021, pp.8741-8750. DOI: [10.1109/CVPR46437.2021.00863](https://doi.org/10.1109/CVPR46437.2021.00863).
- [27] Chen H, Wang Y, Guo T, Xu C, Deng Y, Liu Z, Ma S, Xu C, Xu C, Gao W. Pre-trained image processing transformer. In *Proc. the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2021, pp.12299-12310. DOI: [10.1109/CVPR46437.2021.01212](https://doi.org/10.1109/CVPR46437.2021.01212).
- [28] Kumar M, Weissenborn D, Kalchbrenner N. Colorization transformer. In *Proc. the 9th International Conference on Learning Representations*, May 2021.
- [29] Liu S, Lin T, He D, Li F, Deng R, Li X, Ding E, Wang H. Paint transformer: Feed forward neural painting with stroke prediction. In *Proc. the 2021 IEEE/CVF International Conference on Computer Vision*, October 2021, pp.6598-6607. DOI: [10.1109/ICCV48922.2021.00653](https://doi.org/10.1109/ICCV48922.2021.00653).
- [30] Jiang Y, Chang S, Wang Z. TransGAN: Two pure transformers can make one strong GAN, and that can scale up. In *Proc. the 35th Conference on Neural Information Processing Systems*, Dec. 2021.
- [31] Cordonnier J B, Loukas A, Jaggi M. On the relationship between self-attention and convolutional layers. In *Proc. the 8th International Conference on Learning Representations*, April 2020.
- [32] Xiong R, Yang Y, He D, Zheng K, Zheng S, Xing C, Zhang H, Lan Y, Wang L, Liu T. On layer normalization in the transformer architecture. In *Proc. the 37th International Conference on Machine Learning*, July 2020, pp.10524-10533.
- [33] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In *Proc. the 3rd International Conference on Learning Representations*, May 2015.
- [34] Dosovitskiy A, Brox T. Generating images with perceptual similarity metrics based on deep networks. In *Proc. the 30th International Conference on Neural Information Processing Systems*, December 2016, pp.658-666.
- [35] Lin T Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick C L. Microsoft COCO: Common objects in context. In *Proc. the 13th European Conference on Computer Vision*, September 2014, pp.740-755. DOI: [10.1007/978-3-319-10602-1_48](https://doi.org/10.1007/978-3-319-10602-1_48).
- [36] Phillips F, Mackintosh B. Wiki Art Gallery, Inc.: A case for critical thinking. *Issues in Accounting Education*, 2011, 26(3): 593-608. DOI: [10.2308/iace-50038](https://doi.org/10.2308/iace-50038).
- [37] Kingma D P, Ba J. Adam: A method for stochastic optimization. In *Proc. the 3rd International Conference on Learning Representations*, May 2015.
- [38] Baker N, Lu H, Erlikhman G, Kellman P J. Deep convolutional networks do not classify based on global object shape. *PLoS Computational Biology*, 2018, 14(12): Article No. e1006613. DOI: [10.1371/journal.pcbi.1006613](https://doi.org/10.1371/journal.pcbi.1006613).
- [39] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, June 2016, pp.770-778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [40] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg A C, Li F F. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015, 115(3): 211-252. DOI: [10.1007/s11263-015-0816-y](https://doi.org/10.1007/s11263-015-0816-y).
- [41] Geirhos R, Rubisch P, Michaelis C, Bethge M, Wichmann F A, Brendel W. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proc. the 7th International Conference on Learning Representations*, May 2019.
- [42] Touvron H, Cord M, Douze M, Massa F, Sablayrolles A, Jégou H. Training data-efficient image transformers distillation through attention. In *Proc. the 38th International Conference on Machine Learning*, July 2021, pp.10347-10357.



Hua-Peng Wei is a Master student at the School of Artificial Intelligence, Jilin University, Changchun. He received his B.S. degree in computer science from Jilin University, Changchun, in 2020. His research interests include computational visual media and image processing.



include computational visual media and machine learning.

Ying-Ying Deng received her B.Sc. degree in automation from the University of Science and Technology Beijing, Beijing, in 2017. She is currently a Ph.D. candidate in the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing. Her research interests



computer graphics, computer vision, and machine learning.

Fan Tang is an assistant professor in the School of Artificial Intelligence, Jilin University, Changchun. He received his Ph.D. degree from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, in 2019. His research interests include



His research interests include computer vision and machine learning.

Xing-Jia Pan received his Ph.D. degree in pattern recognition and intelligent system from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, in 2020. He is currently a senior researcher at YouTu Laboratory, Tencent, Shanghai.



ACM and IEEE.

Wei-Ming Dong is a professor at Institute of Automation, Chinese Academy of Sciences, Beijing. He received his Ph.D. degree in computer science from the University of Lorraine, Nancy, in 2007. His research interests include image synthesis and image recognition. He is a member of CCF,