# Master: Meta Style Transformer for Controllable Zero-Shot and Few-Shot Artistic Style Transfer

Hao Tang[1*] Songhua Liu[2*] Tianwei Lin[3] Shaoli Huang[4]
Fu Li[3] Dongliang He[3] Xinchao Wang[2†]
[1]Center for Data Science, Peking University [2]National University of Singapore
[3]VIS, Baidu Inc. [4]Tencent AI Lab
tanghao@stu.pku.edu.cn songhua.liu@u.nus.edu
{lintianwei01,lifu,hedongliang01}@baidu.com shaolihuang@tencent.com xinchao@nus.edu.sg

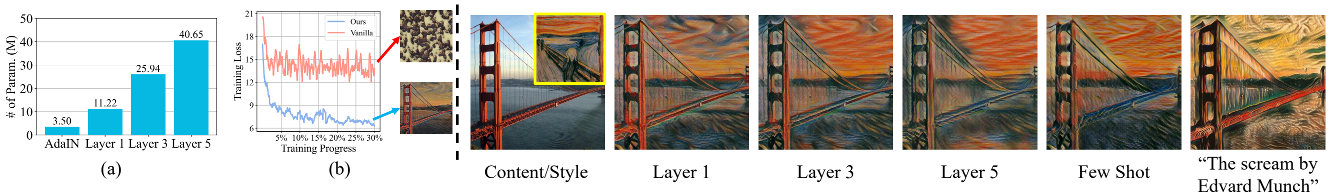


Figure 1. Left: With more Transformer layers, the number of parameters and training difficulty increase significantly. Right: Our method supports test-time controlling of stylization degree via tuning the number of adopted Transformer layers. It is also readily adaptable to few-shot style transfer, where stylization with only 1 layer can be further improved. Text-guided few-shot style transfer is achievable.

## Abstract

*Transformer-based models achieve favorable performance in artistic style transfer recently thanks to its global receptive field and powerful multi-head/layer attention operations. Nevertheless, the over-paramerized multi-layer structure increases parameters significantly and thus presents a heavy burden for training. Moreover, for the task of style transfer, vanilla Transformer that fuses content and style features by residual connections is prone to content-wise distortion. In this paper, we devise a novel Transformer model termed as* Master *specifically for style transfer. On the one hand, in the proposed model, different Transformer layers share a common group of parameters, which (1) reduces the total number of parameters, (2) leads to more robust training convergence, and (3) is readily to control the degree of stylization via tuning the number of stacked layers freely during inference. On the other hand, different from the vanilla version, we adopt a learnable scaling operation on content features before content-style feature interaction, which better preserves the original similarity between a pair of content features while ensuring the stylization quality. We also propose a novel meta learning scheme for the proposed model so that it can not only work in the typical setting of arbitrary style transfer, but also adaptable to the few-shot setting, by only fine-tuning the Transformer encoder layer in the few-shot stage for one specific style. Text-guided few-shot style transfer is firstly achieved with the proposed framework. Extensive experiments demonstrate the superiority of Master under both zero-shot and few-shot style transfer settings.*

## 1. Introduction

Artistic style transfer aims at applying style patterns like colors and textures of a reference image to a given content image while preserving the semantic structure of the content. In contrast to the pioneering optimization method [9] and early per-style-per-model methods like [17, 26], arbitrary style transfer methods [3, 13, 16, 21, 22, 24, 31] enable real time style transfer for any style image in the test time in a zero-shot manner. The flexibility has led to this *arbitrary-style-per-model* fashion to dominate style transfer research.

Recently, to enhance the representation of global information in arbitrary style transfer, Transformer [40] is introduced to this area [4], leveraging the global receptive field and powerful multi-head/layer structure, and achieves superior performance. Nevertheless, the over-parameterized multi-layer structure increases model parameters significantly. As shown in Fig. 1(a), there are 25.94M learnable parameters for a 3-layer Transformer structure in StyTr2 [4], v.s. 3.50M in AdaIN [13], a simple but effective baseline in arbitrary style transfer. Such a large model for standard Transformer inevitably presents a heavy burden for training. As shown in Fig. 1(b), when there are more than

*Equal contribution.
†Corresponding author.

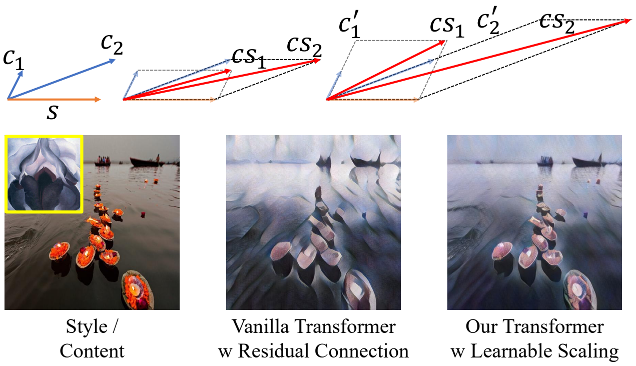


Figure 2. Residual connection in the vanilla Transformer tends to destroy the original similarity relationship on content structure in style transfer task. Our model is designed to address this problem with learnable scaling parameters. The top row shows a simple 2D visualization and the bottom one provides a qualitative example.

4 layers, vanilla Transformer even fails to get convergent in training, which limits the scalability of the Transformer model in style transfer.

Moreover, vanilla Transformer relies on residual connections [12] to stylize content features, which suffers from the content-distortion problem. We illustrate this effect with a 2D visualization in Fig. 2(top), where residual connections lead the transformation results of two content feature vectors to move towards the dominated style features and thereby tend to eliminate their original distinction. The visual effect is that the stylized images would be dominated by some strong style features, such as salient edges, with the original self-(dis)similarity of content structures destroyed, as the example shown in Fig. 2(bottom).

Focusing on these drawbacks, in this paper, we are dedicated to devising a novel Transformer architecture specifically for artistic style transfer. On the one hand, in the proposed model, different Transformer layers share a common group of parameters and a random number of stacked layers are adopted for each training iteration. Compared with the original version, sharing parameters across different layers reduces the total number of parameters significantly and leads to more convenient training convergence. As a byproduct, it is also readily for our model to control the degree of stylization via tuning the number of stacked layers freely in the inference time, as shown in Fig. 1(right). On the other hand, we equip Transformer with learnable scale parameters for content-style interactions instead of residual connections, which alleviates content distortion to a large degree and better preserves content structures while rendering vivid style patterns simultaneously, as shown in the 2D visualization and the qualitative example in Fig. 2.

Furthermore, beyond the typical zero-shot arbitrary style transfer, leveraging a meta learning scheme, our method is adaptable to the few-shot setting. By only fine-tuning the Transformer encoder layer in the few-shot stage, rapid adaptation to the model for a specific style within a limited number of updates is possible, where the stylization with only 1 layer can be further improved, as shown in Fig. 1. Beyond that, we first achieve text-guided few-shot style transfer with this framework, which largely alleviates the training burden of previous per-text-per-model solution. In this sense, we term the overall pipeline ***Meta Style Transformer*** (*Master*). Our contributions are summarized as follows:

- We propose a novel Transformer architecture specifically for artistic style transfer. It shares parameters between different layers, which not only helps training convergence, but also allows convenient control over the stylization effect.
- We identity the content distortion problem of residual connections in Transformer and propose learnable scale parameters as an option to alleviate the problem.
- We introduce a meta learning framework for adapting original training setting of zero-shot style transfer to the few-shot scenario, with which our Master achieves very good trade-off between flexibility and quality.
- Experiments show that our model achieves results better than those of arbitrary-style-per-model methods. Furthermore, under the few-shot setting, either conditioned on image or text, Master can even yield performance on par with that of per-style-per-model methods with significantly less training cost.

## 2. Related Works

**Arbitrary-Style-Per-Model Methods.** A lot of works achieve arbitrary style transfer via global feature transformation, *e.g.*, WCT [22], AdaIN [13], Linear style transfer [21], DIN [15], MCCNet [2], and MAST [14]. In general, they can achieve the most attractable style transfer speed but dismiss stylized effects for local details a lot. In order to add more consideration of local details to global transformation based methods, there are patch-similarity based solution [1, 10, 37] and attention based method [3, 24, 31, 44]. While paying attention to local details, it is still hard to transfer complex style patterns and prone to unsatisfactory distortions due to the simple swap and fusion strategies.

**Transformer in Style Transfer.** Transformer proposed in [40] is widely used in the natural language processing and has become a powerful baseline. Recently, Transformer receives better performance than CNN models in many vision tasks and are involved in various model zoos [6, 11, 18, 42, 43, 45]. Specifically, in the field of style transfer, [41] propose StyFormer, where the transformation is driven by the

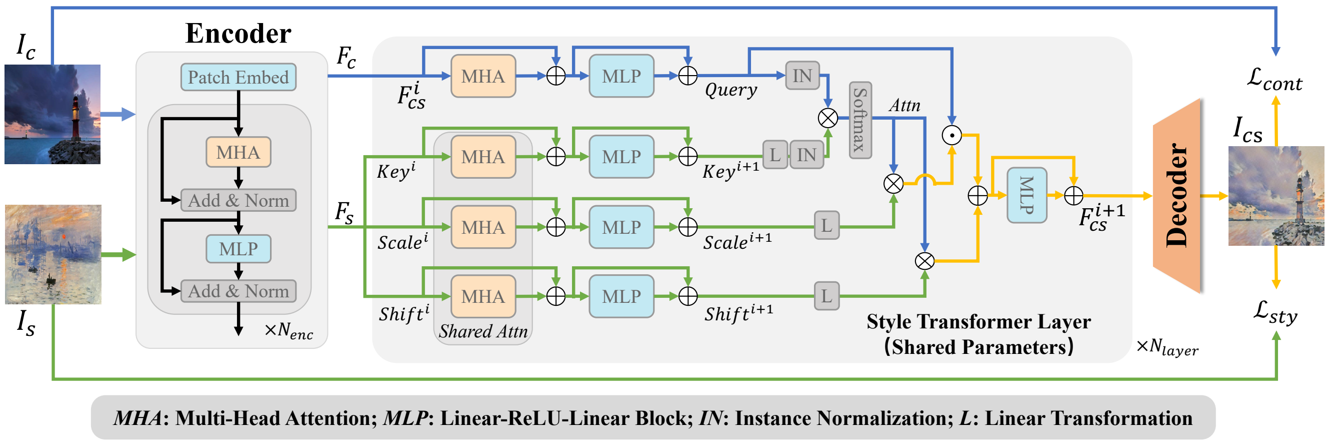


Figure 3. Overview of our model architecture.

cross-attention module in Transformer. [4] develop a Transformer model based on ViT [5]. However, the number of parameters and training difficulty would increase considerably with the increasing of Transformer layers. Also, following residual connections in original Transformer, it suffers from the distortion problem on content structures as shown in Fig. 2. Similar problems also exist in [27].

**Meta Learning in Style Transfer.** On this routine, [36] train a meta network to predict parameters of the generator model for one style reference image. [47] employ MAML algorithm [7] to find a style-free model for fast adaptation. Our method is different from theirs in three aspects: (1) our framework works for both few-shot and arbitrary style transfer thanks to the cross-attention in Transformer; (2) our meta learning algorithm is based on Reptile [30], a first-order meta learning algorithm without the necessity to operate gradients of higher levels, which is more efficient in training; and (3) only style encoder instead of the whole model needs to be updated during the few-shot learning stage, which makes it more convenient in practice.

**Text-Guided Synthesis.** The emergence of CLIP model [34] bridges text and image domain, which supports a series of works on text-guided synthesis [8, 32, 35]. In the field of style transfer, CLIPstyler [20] proposes a patch-wise CLIP loss and achieves text-guided style transfer. Nevertheless, as a per-text-per-model solution, it is inconvenient in practice to handle a large number of text inputs. Despite the recent dataset distillation scheme may alleviate this issue, existing approaches, however, largely focus on images as input [25, 28, 46]. In this paper, we first achieve text-guided few-shot style transfer with the proposed meta learning algorithm, which improves the flexibility of CLIPstyler significantly.

# 3. Methods

In this section, we give details of the proposed Meta Style Transformer (Master) for zero-shot and few-shot style transfer. We first introduce the network architecture of the proposed model, then illustrate how we train our model in a meta-learning fashion, and finally describe loss function.

## 3.1. Network Architecture

The proposed model comprises an encoder, a feature modification module, and a decoder, as demonstrated in Fig. 3. We employ the first 2 stages of Swin Transformer [29] as the encoder Enc to extract common image features for both content and style images. The decoder Dec follows the setting of [13] with 3 upsampling convolutional blocks. For feature modification, we propose a *Style Transformer* module for transferring complex style patterns, which will be introduced later. Taking a style image $I_s$ and a content image $I_c$ as input, we first divide them into $4\times 4$ patches and extract their corresponding feature maps $F_c$ and $F_s$ with the Swin encoder Enc:

$$\begin{aligned} F_c &= \mathrm{Enc}(I_c), \\ F_s &= \mathrm{Enc}(I_s), \end{aligned} \tag{1}$$

where the spatial scales for $F_c$ and $F_s$ are 1/8 of those for $I_c$ and $I_s$. Then, the embedded feature $F_{cs}$ is derived by:

$$F_{cs} = \mathrm{StyleTrans}(F_s, F_c), \tag{2}$$

where StyleTrans denotes the Style Transformer module, and $F_{cs}$ has the same shape as $F_c$. Finally, we can synthesize the stylized image $I_{cs}$ with the decoder as:

$$I_{cs} = \mathrm{Dec}(F_{cs}). \tag{3}$$

**Style Transformer.** Similar to previous Transformer-based style transfer models like StyTr2 [4], the Transformer encoder is used to encode style information while the Transformer decoder takes charge of content-style interaction. In our model, there is only one copy of parameters shared by all Transformer encoder and decoder layers. Also, different from standard Transformer where decoder layers would follow all encoder layers, encoder and decoder layers would be executed in an alternate fashion. The next layer would take the output of the current layer as input.

Our Transformer decoder layer is composed of self-attention, cross-attention, and non-linear blocks, for content encoding and content-style interaction. Specifically, content features are first processed with a self-attention step. Then, cross-attention is conducted by taking the content encoding as query and the style encoding as key and value, followed by a MLP for non-linear transformation.

Notably, in vanilla Transformer, features before and after the cross-attention step are fused by a residual connection, which is harmful for content structures as analyzed in Fig. 2. We thereby replace the residual connection with dynamic and learnable scaling and shifting steps, whose parameters are determined by the style encoder. In this way, the output of style encoder should consist of three parts: key for the following cross-attention $K_s$, scaling parameters $V_\sigma$, and shifting parameters $V_\mu$. The prediction of each part shares a same self-attention map to save memory but uses independent non-linear transformation. The process in the Transformer encoder layer can be formulated as:

$$\begin{aligned}
\mathrm{MHA}(Q,K,V) &= [head_1, ..., head_h]W^O, \\
head_i &= \mathrm{Att}(QW^{Q_i}, KW^{K_i}, VW^{V_i}), \\
\mathrm{Att}(Q,K,V) &= \mathrm{softmax}(\frac{QK^\top}{\sqrt{d_k}}V), \\
K'_s &= K_s + \mathrm{MHA}(K_s, K_s, K_s), \\
K_s &= K'_s + \mathrm{MLP}(K'_s), \\
V'_\sigma &= V_\sigma + \mathrm{MHA}(K_s, K_s, V_\sigma), \\
V_\sigma &= V'_\sigma + \mathrm{MLP}(V'_\sigma), \\
V'_\mu &= V_\mu + \mathrm{MHA}(K_s, K_s, V_\mu), \\
V_\mu &= V'_\mu + \mathrm{MLP}(V'_\mu),
\end{aligned} \tag{4}$$

where $K_s$, $V_\sigma$, and $V_\mu$ are initialized as the style feature $F_s$ before the first Transformer encoder layer. We do not incorporate normalization to encode style features since second-order statistics can largely represent style information.

Then, in the cross-attention of Transformer decoder, scaling and shifting parameters for each content feature point is aggregated from $V_\sigma$ and $V_\mu$ respectively according to the cross-attention map. The process in one Transformer decoder block can be written as:

$$\begin{aligned}
F'_{cs} &= F_{cs} + \mathrm{MHA}(F_{cs}, F_{cs}, F_{cs}), \\
\sigma &= \mathrm{MHA}(\mathrm{IN}(F'_{cs}), \mathrm{IN}(K_s), V_\sigma), \\
\mu &= \mathrm{MHA}(\mathrm{IN}(F'_{cs}), \mathrm{IN}(K_s), V_\mu), \\
F''_{cs} &= \sigma \odot F'_{cs} + \mu, \\
F_{cs} &= F''_{cs} + \mathrm{MLP}(F''_{cs}),
\end{aligned} \tag{5}$$

**Algorithm 1** Meta Training

**Require:** $\mathcal{D}_c$: content dataset; $\mathcal{D}_s$: style dataset; $\delta$: inner learning rate; $\eta$: outer learning rate; $k$: number of inner updates; $T$: maximal number of stacked layers;
**Ensure:** trained meta generator parameters $\theta$

initialize $\theta$ randomly
**for** iteration 1, 2, 3, $\cdots$ **do**
    sample a style image $I_s$ from $\mathcal{D}_s$
    $\omega \leftarrow \theta$
    **for** $k$ times **do**
        sample a batch of content image $I_c$ from $\mathcal{D}_c$
        sample the number of layers from $1 \sim T$
        forward propagation using Eq. 1-5
        compute inner loss $L$ using Eq. 8
        $\omega \leftarrow \omega - \delta \nabla L$
    **end for**
    $\theta \leftarrow \theta + \eta(\omega - \theta)$
**end for**

where IN denotes instance normalization [39] and $\odot$ represents element-wise multiplication. $F_{cs}$ is initialized as $F_c$ before the first Transformer decoder layer.

### 3.2. Training Pipeline

To achieve high-quality style transfer, we introduce a two-stage training strategy that comprises meta training and fast adaptation. The meta training stage is designed to learn a generic model initialization, while the fast adaptation adapts the network for a single style in a few iterations for few-shot style transfer. Note that zero-shot style transfer is a special case of the overall training configuration.

**Meta Training:** Inspired by *Reptile* [30], a first-order meta learning algorithm, rendering style patterns of a specific reference style image can be viewed as a task. We seek an optimal initialization for neural networks in this stage, so that the networks can be rapidly adapted for a new task in only a few shots. The main training procedure for this stage is shown in Algorithm 1. In each iteration, we sample 1 style and $k$ batches of contents to perform inner optimization to obtain "fast weights" $\omega$, which would later guide the update of "slow weights" $\theta$, to move a step in the direction of $\omega - \theta$. Notably, as there is only one group of parameters for Transformer encoder and decoder layers, we randomly choose a number as the number of stacked layers for the Style Transformer in each iteration.

**Fast Adaptation:** The trained model after the first stage would serve as an initialization and will be adapted for a single style. With the same objective, this stage behaves almost identically to the internal loop in Algorithm 1. The only difference is that only parameters of the Transformer encoder layer are necessary to be updated, since (1) the Transformer encoder layer is the main component to extract style patterns and has the most significant impact on stylization, and (2) it would save memory and speed up the adaptation.

**Zero-Shot Style Transfer:** "Zero-Shot" means that, once the meta training is done, there is no fast adaptation stage needed and the meta model itself can support arbitrary style transfer. To encourage the model to produce satisfactory results in the zero shot setup, we set the inner optimization time $k$ to 1, where the algorithm is reduced to the typical training paradigm of existing arbitrary style transfer methods. In this sense, Algorithm 1 provides a more general setting for style transfer in both zero-shot and few-shot cases.

**Text-Guided Style Transfer.** We perform text-guided style transfer based on our image style transfer model with slight modifications. Following the common practice, a pre-trained CLIP encoder is imported to extract text features. To be consistent with image input, we use a StyleGAN-like mapping network to convert text features into pseudo image features. In the meta training stage, since CLIP unifies the feature space, we use image instead of text as style input to avoid additional text dataset. In the fast adaptation stage, the mapping network along with the Transformer encoder is updated, considering there is still a gap between CLIP image feature and text feature.

### 3.3. Loss Function

The training objective in both meta-training and fast adaptation stages follows many works in arbitrary style transfer, which consists of content loss and style loss. Let $F^x$ represent features on `ReLU-x_1` layer of a pre-trained VGG19 network [38] for loss computation. The content loss is defined by the normalized perceptual loss [17]:

$$\mathcal{L}_{cont} = \sum_{x=2}^{5} \|\mathrm{IN}(F_c^x) - \mathrm{IN}(F_{cs}^x)\|_2, \tag{6}$$

while the style loss adopts the mean-variance loss [13]:

$$\mathcal{L}_{sty} = \sum_{x=2}^{5} (\|\mu(F_s^x) - \mu(F_{cs}^x)\|_2 + \|\sigma(F_s^x) - \sigma(F_{cs}^x)\|_2), \tag{7}$$

where $\mu$ and $\sigma$ calculate the channel-wise mean and standard deviation separately. The overall objective is given by the weighted summation of the two losses:

$$\mathcal{L} = \mathcal{L}_{cont} + \lambda \mathcal{L}_{sty}, \tag{8}$$

where $\lambda$ controls the balance between two terms.

In text-guided style transfer, the loss functions are the same as CLIPstyler, including global CLIP loss, directional CLIP loss and PatchCLIP loss.

## 4. Experiments

### 4.1. Implementation Details

We use MS-COCO [23] as our content dataset and WikiArt test set [33] as our style dataset. The content dataset contains roughly 80,000 images and the style dataset has about 20,000 images. The optimizer is Adam [19] with learning rates of both inner and outer loops set as 0.0001 and the batch size is 4. In training, we first resize the content and style image to $512 \times 512$ and then randomly crop to $256 \times 256$ resolution. During inference, our model can handle inputs of any size. The update times for inner optimization $k$ is set as 2 for few-shot case and 1 for zero-shot case. In training, the maximal number of stacked layers is 4. All the multi-head attention blocks are instantiated as shifted window attention in [29], with window size 8 and shift size 4. The model is trained on a Nvidia 3090 with 9k iterations in the meta training stage for convergence while only 100 steps for image input and 20 steps for text input in the fast adaptation for few-shot style transfer, which takes less than 1 minute. Hyper-parameter $\lambda$ is set as 10.

| Method | $\mathcal{L}_{cont}$ | $\mathcal{L}_{sim}$ | $\mathcal{L}_{sty}$ |
|---|---|---|---|
| AdaIN | $4.88_{\pm 0.70}$ | $0.53_{\pm 0.23}$ | $1.51_{\pm 0.78}$ |
| Linear | $3.93_{\pm 0.76}$ | $0.44_{\pm 0.17}$ | $1.97_{\pm 0.96}$ |
| AvatarNet | $5.76_{\pm 0.53}$ | $0.56_{\pm 0.19}$ | $3.19_{\pm 1.82}$ |
| SANet | $4.72_{\pm 0.61}$ | $0.50_{\pm 0.17}$ | $1.15_{\pm 0.58}$ |
| MANet | $4.93_{\pm 0.57}$ | $0.50_{\pm 0.17}$ | $1.41_{\pm 0.77}$ |
| MCCNet | $4.22_{\pm 0.69}$ | $0.47_{\pm 0.17}$ | $1.56_{\pm 0.84}$ |
| AdaAttN | $4.46_{\pm 0.70}$ | $0.43_{\pm 0.16}$ | $2.20_{\pm 1.19}$ |
| StyTr2 | $3.78_{\pm 0.99}$ | $0.48_{\pm 0.22}$ | $1.50_{\pm 0.69}$ |
| StyFormer | $4.94_{\pm 0.79}$ | $0.43_{\pm 0.15}$ | $2.20_{\pm 1.44}$ |
| MetaNet | $\mathbf{3.48}_{\pm 0.85}$ | $0.45_{\pm 0.19}$ | $2.47_{\pm 1.69}$ |
| MetaStyle* | $3.64_{\pm 1.12}$ | $0.42_{\pm 0.18}$ | $2.47_{\pm 1.06}$ |
| Johnson† | $4.60_{\pm 0.76}$ | $0.59_{\pm 0.20}$ | $1.02_{\pm 0.34}$ |
| Ours-Vanilla | $5.50_{\pm 0.63}$ | $0.59_{\pm 0.26}$ | $0.85_{\pm 0.38}$ |
| Ours-Norm | $4.70_{\pm 0.75}$ | $0.43_{\pm 0.13}$ | $0.93_{\pm 0.33}$ |
| Ours-ZS-L1 | $4.13_{\pm 0.68}$ | $0.41_{\pm 0.14}$ | $0.92_{\pm 0.40}$ |
| Ours-ZS-L3 | $4.20_{\pm 0.68}$ | $0.41_{\pm 0.13}$ | $0.81_{\pm 0.31}$ |
| Ours-FS* | $4.24_{\pm 0.82}$ | $\mathbf{0.38}_{\pm 0.12}$ | $\mathbf{0.79}_{\pm 0.25}$ |

Table 1. Quantitative comparisons. *ZS* and *FS* for our model denote zero-shot and few-shot modes. *Vanilla* denote replacing our architecture with original Transformer. *Norm* means adding layer normalization in the Transformer encoder layer. *L1*/*L3* means using 1/3 Transformer layers in the test time. $*$ and † denote few-shot and per-style-per-model methods.

### 4.2. Comparison with Prior Works

In this section, we compare results by our Master with 13 state-of-the-art style transfer methods, including 3 global transformation based methods (AdaIN [13], Linear style transfer [21], and MCCNet [2]), 1 patch swap based method (Avatar-Net [37]), 3 attention based methods (SANet [31], MANet [3], and AdaAttN [24]), 2 transformer based methods (StyTr2 [4] and StyFromer [41]), 2 meta learning based methods (MetaNet [36] and MetaStyle [47]), 1 per-style-per-model method by Johnson *et al.* [17] and 1 text-guided style transfer method (CLIPstyler).

**Quantitative Comparison.** We adopt content loss $\mathcal{L}_{cont}$ in Eq. 6 and style loss $\mathcal{L}_{sty}$ in Eq. 7 as evaluation metrics to reflect effects of learning by different methods. We also design a metric $\mathcal{L}_{sim}$ to reflect the preservation of the spatial-wise self cosine similarity of content structures:

$$D^x_{*,ij} = 1 - \frac{F^x_{*,i} \cdot F^x_{*,j}}{\|F^x_{*,i}\| \|F^x_{*,j}\|}$$
$$\mathcal{L}_{sim} = \sum_{x=3}^{4} \frac{1}{n_x^2} \sum_{i,j} \left| \frac{D^x_{c,ij}}{\sum_k D^x_{c,kj}} - \frac{D^x_{cs,ij}}{\sum_k D^x_{cs,kj}} \right|, \tag{9}$$

where $n_x$ is the number of spatial locations for the current feature map and the second foot script denotes the spatial index. Smaller $\mathcal{L}_{sim}$ means that the original spatial-wise relationship is better preserved during style transfer, *i.e.*, less content distortion. We use the test dataset in the code page of [13] for evaluation, with 11 content images and 20 style images to form 220 content-style pairs. We report the mean and standard deviation over the 220 cases in Tab. 1. Notably, the lowest style loss and content similarity loss are achieved by Master simultaneously compared with previous methods, even better than the per-style-per-model solution by Johnson *et al.* [17], and can be further reduced in the few-shot case with comparable content loss values, which demonstrates the **joint** advantages of our method in content preserving and style rendering. Meanwhile, the lower standard deviations demonstrate the robustness of our model.

**Qualitative Comparison.** Qualitative examples by different methods are shown in Fig. 4. Full comparisons with more methods can be found in the supplementary material.

We first give discussion on the global-transformation based methods. As shown in the 3rd and 4th rows, in AdaIN and Linear, features in all positions share the same transformation matrix, which fails to migrate detail textures in style images and has poor color saturation.

As for the attention-based methods, SANet in the 5th row introduces local focus to style images. However, due to the simple fusion of one-layer attention results and original content features, it brings textures and content distortion in many cases, *e.g.*, background areas of the 4th and 5th columns. AdaAttN in the 6th row dedicates to addressing the distortion problem in SANet, but it sacrifices the ability to render salient style patterns and textures.

We then compare our results with those by Transformer-based methods. StyFormer in the 7th row adopts a design of the cross-attention module in Transformer without exploring the effect of self-attention to enhance style rendering. Thus, the results appear under-stylized compared with those by our method. StyTr2 in the 8th row is a Transformer architecture with a pure self-attention mechanism, which makes the performance on the extraction and migration of local textures not satisfactory enough, *e.g.*, strokes of the 2nd column and waves of the 4th column. Moreover, since it follows the design of residual connections in


Figure 4. Comparison with previous state-of-the-art style transfer methods. Zoom in for better details.


Figure 5. Comparison with CLIPstyler in text-guided style transfer. Zoom in for better details.

vanilla Transformer, the issue of content distortion in Fig. 2 is still evident, as shown in the background areas of the 4th and 5th columns. Compared with these methods, Master renders more vivid global and local style patterns. Meanwhile, it well maintains content structures simultaneously with learnable scaling and shifting parameters. Furthermore, in the few-shot case, comparable performance can be achieved with only 1 Transformer layer.

MetaNet and MetaStyle also involve the concept of meta learning into their style transfer frameworks. On the one hand, MetaNet predicts parameters of a generator network given one style image via a meta network, which is an ambitious goal and makes the training more difficult. Thus, as shown in the 9th row, the stylized effects of their results still appear weak. On the other hand, MetaStyle achieves inferior performance in terms of local details as demonstrated in the 10th row, since only parameters of global normalization layers are learnable in the fast adaptation stage.

The single style transfer method by Johnson *et al.* in the 11th row tends to arrange patterns learned by the network arbitrarily, which may produce undesired and distorted effects. By contrast, the complex content-style dependence is constructed in our Transformer model and thus achieves more remarkable performance on transferring style patterns to proper positions in content images.

Some qualitative comparisons on text-guided style transfer with CLIPstyler are shown in Fig. 5, which uses per-text-per-model fashion. Through these results, we observe that despite significantly more training steps, CLIPstyler

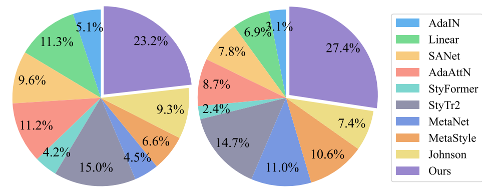


Figure 6. Results of user study. Left: Our zero-shot stylization results compared with SOTA methods. Right: Our few-shot stylization comparison results.

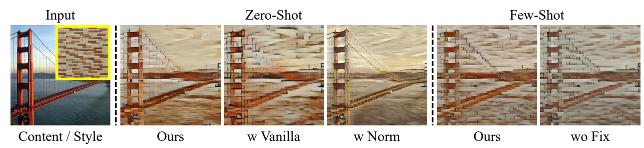


Figure 7. Ablation study on key designs in Master model.

still suffers from weak stylization. By contrast, our method generates more vivid results in only 20 adaptation steps, which demonstrates the advantages of the proposed Master architecture and the versatility of our training pipeline.

**User Study.** Following most style transfer works, we conduct user study and report user preference. We choose 20 content images and 20 style images to form 400 content-style pairs. We involve 100 people and randomly assign 20 stylized results from compared methods to each subject. Our method showcases zero-shot results with 3 Transformer layers in 10 tests and few-shot results in the remaining 10. For each pair, the order of results is randomized, and participants choose their favorite. With 2,000 votes in total, Fig. 6 demonstrates Master's superior style transfer quality.

### 4.3. Ablation Study

**Architecture.** We conduct ablation studies on key designs in Master to illustrate their effects on the stylization quality. As shown in Fig. 7, on the one hand, if vanilla Transformer model is used, without learnable scaling and shifting parameters, noisy textures can be introduced significantly, which distorts the original content structures and affects style transfer quality. On the other hand, if layer normalization operations in the standard Transformer are used in the style encoder, style patterns would become less saturated, since normalization removes second-order feature statistics which represents the style information to a large degree. Quantitative studies in Tab. 1 also support our analysis and come to the same conclusion.

**Meta Training and Fast Adaptation.** We provide training loss visualization in Fig. 8, to demonstrate the effectiveness of our meta training and fast adaptation algorithm. Our meta training stage learns an appropriate initial state, which enables fast decent and convergence of loss values in the fast adaptation stage. We show the same visualization in the per-style-per-model setting as a benchmark, using the same loss function and network architecture. As shown in Fig. 8,

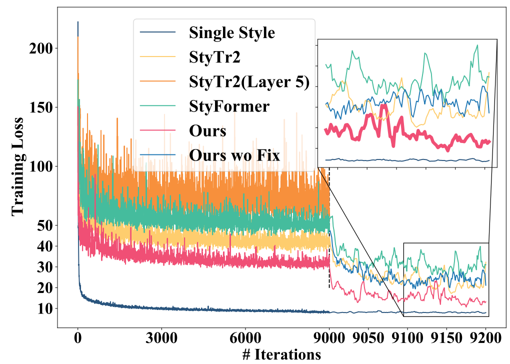


Figure 8. Training loss visualization during meta training and fast adaptation of three transformer based models: StyTr2, StyFormer, and our Master. The loss curve under per-style-per-model setting by our model is used as a reference. *Ours-wo-Fix* means that all the parameters need to be updated during the fast adaptation stage.

it requires roughly 3k iterations for the model to be convergent in the per-style-per-model case (Single Style), while the meta model (Ours) after 9k iterations can be adapted for any style image in only a few shots with competitive training results. Thus, the total number of iterations for our method is significantly smaller than per-style-per-model ones with the growth number of required styles.

By default, all the parameters except those in the style encoder module are fixed during the fast adaptation stage. We also experiment with training the whole model in this stage (Ours wo Fix in Fig. 8) and find that the training and convergence would become difficult and require more time and memory. Moreover, updating all the parameters may result in insufficient adaptation given a limited number of training steps. As demonstrated in Fig. 7, there is a gap on the global tone between the result and the style image.

**Base Model.** We evaluate StyFormer and StyTr2 as base models, presenting loss visualizations. Heavier Transformers raise training difficulty for earlier models, resulting in inferior convergence. When the number of Transformer layers is 5, StyTr2 even fails to converge. By contrast, our model adopts parameter-sharing across all Transformer layers, which results in an overall light-weight structure. Consequently, it enjoys better training effects in the meta training stage as shown in Fig. 8. Moreover, as shown in the zoom-in part, Master also enables overall lower and more stable training in fast adaptation. Qualitative examples by different base models can be found in the supplement.

**Controllable Style Transfer.** We compare with some widely-adopted approaches for the similar goal to control the degree of stylization, including (1) Recursion, which treats stylized results as input of content images in the next iteration, and (2) Linear Mixup, which conducts linear combinations between content features and stylized features before the decoder. Vanilla Transformer is adopted for these approaches. The major difference between Recursion and ours is that our method takes the iterative stylization into consideration in the training time so that the model would learn to transfer style patterns layer-by-layer. Thus, as shown in Fig. 9, Recursion merely uses more intensive colors with the increasing of repeat times. Also, for Linear Mixup, it is typically hard to generate reasonable style transfer results especially when the weight of contents is high. Compared with them, our method is capable of rendering more style patterns and increasing the artistic abstraction when more Transformer layers are stacked.

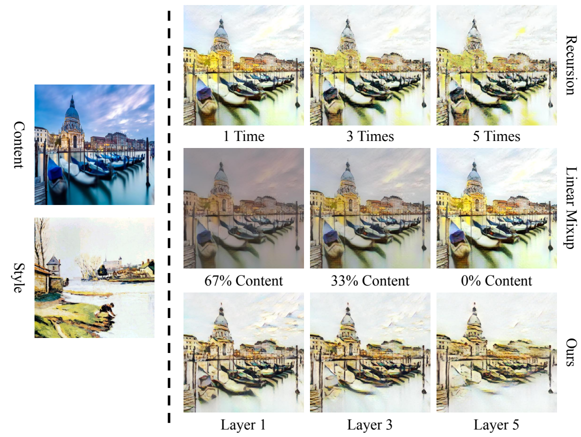


Figure 9. Comparisons with widely-adopted approaches to control the degree of stylization.

## 5. Conclusion

In this paper, we propose a novel Transformer model specifically for artistic style transfer, termed as Meta Style Transformer (Master). On one hand, parameters of different Transformer layers are shared, which reduces the total number of parameters significantly and thus enables easier convergence. It is also convenient for Master to control the degree of stylization via customizing the number of stacked layers in the inference time. On the other hand, different from standard Transformer, our model adopts dynamic and learnable scaling and shifting operations instead of original residual connections, which helps preserve similarity relationship in content structures while migrating remarkable style patterns. Specifically, Master is trained using a meta learning algorithm for few-shot style transfer. Only parameters of the Transformer encoder layer need to be updated in the few-shot stage, which benefits the fast and robust adaptation. Zero-shot arbitrary style transfer is a special case of the training configuration. Experiments suggest that Master outperforms previous state-of-the-art arbitrary style transfer methods on both content preserving and style rendering in the zero-shot case.

## Acknowledgment

This project is supported by the Singapore Ministry of Education Academic Research Fund Tier 1 (WBS: A-0009440-01-00).

## References


[1] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016. 2

[2] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu. Arbitrary video style transfer via multi-channel correlation. *arXiv preprint arXiv:2009.08003*, 2020. 2, 5

[3] Yingying Deng, Fan Tang, Weiming Dong, Wen Sun, Feiyue Huang, and Changsheng Xu. Arbitrary style transfer via multi-adaptation network. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2719–2727, 2020. 1, 2, 5

[4] Yingying Deng, Fan Tang, Xingjia Pan, Weiming Dong, Chongyang Ma, and Changsheng Xu. $stytr^2$: Unbiased image style transfer with transformers, 2021. 1, 3, 5

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. 3

[6] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. 3

[8] Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022. 3

[9] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[10] Shuyang Gu, Congliang Chen, Jing Liao, and Lu Yuan. Arbitrary style transfer with deep feature reshuffle. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8222–8231, 2018. 2

[11] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[13] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 1, 2, 3, 5, 6

[14] Jing Huo, Shiyin Jin, Wenbin Li, Jing Wu, Yu-Kun Lai, Yinghuan Shi, and Yang Gao. Manifold alignment for semantically aligned style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14861–14869, 2021. 2

[15] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4369–4376, 2020. 2

[16] Yongcheng Jing, Yining Mao, Yiding Yang, Yibing Zhan, Mingli Song, Xinchao Wang, and Dacheng Tao. Learning graph neural networks for image style transfer. In *ECCV*, 2022. 1

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 1, 5, 6

[18] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey, 2021. 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[20] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18062–18071, 2022. 3

[21] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast arbitrary style transfer. *arXiv preprint arXiv:1808.04537*, 2018. 1, 2, 5

[22] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *arXiv preprint arXiv:1705.08086*, 2017. 1, 2

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[24] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6649–6658, October 2021. 1, 2, 5

[25] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In *Conference on Neural Information Processing Systems*, 2022. 3

[26] Songhua Liu, Hao Wu, Shoutong Luo, and Zhengxing Sun. Stable video style transfer based on partial convolution with depth-aware supervision. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2445–2453, 2020. 1

[27] Songhua Liu, Jingwen Ye, Sucheng Ren, and Xinchao Wang. Dynast: Dynamic sparse transformer for exemplar-guided image generation. In *European Conference on Computer Vision*, 2022. 3

[28] Songhua Liu, Jingwen Ye, Runpeng Yu, and Xinchao Wang. Slimmable dataset condensation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 3

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 3, 5

[30] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms, 2018. 3, 4

[31] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5880–5888, 2019. 1, 2, 5

[32] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2085–2094, 2021. 3

[33] Fred Phillips and Brandy Mackintosh. Wiki art gallery, inc.: A case for critical thinking. *Issues in Accounting Education*, 26(3):593–608, 2011. 5

[34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 3

[35] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 3

[36] Falong Shen, Shuicheng Yan, and Gang Zeng. Meta networks for neural style transfer, 2017. 3, 5

[37] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8242–8250, 2018. 2, 5

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 5

[39] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 4

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 1, 2

[41] Xiaolei Wu, Zhihao Hu, Lu Sheng, and Dong Xu. Styleformer: Real-time arbitrary style transfer via parametric style composition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14618–14627, 2021. 2, 5

[42] Xingyi Yang, Jingwen Ye, and Xinchao Wang. Factorizing knowledge in neural networks. In *European Conference on Computer Vision*, 2022. 2

[43] Xingyi Yang, Daquan Zhou, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. In *Conference on Neural Information Processing Systems*, 2022. 2

[44] Yuan Yao, Jianqiang Ren, Xuansong Xie, Weidong Liu, Yong-Jin Liu, and Jun Wang. Attention-aware multi-stroke style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1467–1475, 2019. 2

[45] Jingwen Ye, Songhua Liu, and Xinchao Wang. Partial network cloning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2

[46] Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review. *arXiv preprint arXiv:2301.07014*, 2023. 3

[47] Chi Zhang, Yixin Zhu, and Song-Chun Zhu. Metastyle: Three-way trade-off among speed, flexibility, and quality in neural style transfer, 2019. 3, 5