Towards Ultra-Resolution Neural Style Transfer via Thumbnail Instance Normalization

Zhe Chen\textsuperscript{1}, Wenhai Wang\textsuperscript{2\textreturn}, Enze Xie\textsuperscript{3}, Tong Lu\textsuperscript{1\textreturn}, Ping Luo\textsuperscript{3}

\textsuperscript{1}State Key Lab for Novel Software Technology, Nanjing University
\textsuperscript{2}Shanghai Artificial Intelligence Laboratory \textsuperscript{3}The University of Hong Kong

corr@smail.nju.edu.cn, wangwenhai@pjlab.org.cn, xieenze@hku.hk, lutong@nju.edu.cn, pluo@cs.hku.hk

Abstract

We present an extremely simple \textbf{Ultra-Resolution Style Transfer} framework, termed URST, to flexibly process arbitrary high-resolution images (e.g., $10000 \times 10000$ pixels) style transfer for the first time. Most of the existing state-of-the-art methods would fall short due to massive memory cost and small stroke size when processing ultra-high resolution images. URST completely avoids the memory problem caused by ultra-high resolution images by (1) dividing the image into small patches and (2) performing patch-wise style transfer with a novel \textit{Thumbnail Instance Normalization} (TIN). Specifically, TIN can extract thumbnail features’ normalization statistics and apply them to small patches, ensuring the style consistency among different patches.

Overall, the URST framework has three merits compared to prior arts. (1) We divide input image into small patches and adopt TIN, successfully transferring image style with arbitrary high-resolution. (2) Experiments show that our URST surpasses existing SOTA methods on ultra-high resolution images benefiting from the effectiveness of the proposed stroke perceptual loss in enlarging the stroke size. (3) Our URST can be easily plugged into most existing style transfer methods and directly improve their performance even without training. Code is available at \url{https://git.io/URST}.

Introduction

With the development of deep learning, neural style transfer has achieved remarkable success \cite{Johnson, Alahi, Fei-Fei, Lu, Shen, Yan, Zeng, Sanakoyeu, Li, Zeng}. However, ultra-high resolution style transfer is rarely explored in these works. In natural scenes, ultra-high resolution images are often seen in large posters, photography works, and ultra-high definition (e.g., 8K) videos. There are two main challenges when stylizing ultra-high resolution images: (1) The massive memory cost of ultra-high resolution images may exceed the GPU memory capacity. (2) Small stroke size may cause unpleasant dense textures in ultra-high resolution results.

First, for the memory limitation, the existing methods mainly use lightweight network architecture \cite{Jing, An, Lu}, model pruning \cite{An, Wang}, and knowledge distillation \cite{Wang} to reduce memory cost. However, most of these methods are palliatives. As shown in Figure 1, with the growth of the input resolution, the memory cost of the distillation-based method \cite{Wang} increases sharply and finally runs out of the GPU memory (12GB in Titan XP). This phenomenon motivates us to design a more effective strategy for stylizing ultra-high resolution images.

The second problem is that the brush strokes in ultra-high resolution stylized results are relatively small. As shown in Figure 2(b), when given a high-resolution input, the model with small brush strokes would produce unpleasant dense textures. Enlarging the stroke size is a widely-used approach to address this problem. At present, the existing methods can be roughly divided into two categories. One is to train or inference with large style images \cite{Jing, Li, Zhang, Dana}. Another solution is to enlarge the receptive field of the style transfer network \cite{Jing, Wang}. However, most of these methods tend to take extra inference time and memory, are not suitable for ultra-high resolution style transfer.

To compensate the above limitations, this work proposes...
an Ultra-Resolution Style Transfer framework, termed URST. Different from previous methods (Jing et al. 2018, An et al. 2020, Wang et al. 2020, 2017), our method (1) takes small patches instead of a full image as input, which makes it possible to process arbitrary high-resolution images under limited memory resources. (2) We replace the original instance normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016) by the proposed thumbnail instance normalization (TIN), to ensure the style consistency among different patches. As shown in Figure 2(c), if we perform patch-wise style transfer with IN directly, style inconsistency among different patches would make them cannot be assembled into a pleasing image. (3) We propose a stroke perceptual loss as an auxiliary loss for neural style transfer, motivating style transfer networks to keep large brush strokes.

Overall, the proposed URST has three advantages:

1. Our framework can process arbitrary high-resolution images with limited memory. As shown in Figure 2 when stylizing an ultra-high resolution image of 10000×10000 pixels based on (Wang et al. 2020), our framework only requires 1.94GB memory, while the original method needs 30.25GB, 15 times larger. To our knowledge, it is the first unconstrained resolution style transfer method.

2. Our framework achieves high-quality style transfer of ultra-high resolution images. As shown in Figure 2(d), our method uses larger brush strokes to depict the scene, which is much better than the effects presented in Figure 2(b).

3. Our framework can be easily plugged into most existing style transfer methods. Even without training, our framework can also obtain high-resolution results.

Related Work

Neural Style Transfer. Inspired by the success of convolutional neural networks (CNNs), Gatys, Ecker, and Bethge (2016) first proposed a CNN-based style transfer algorithm, which opened up the new research field. To accelerate neural style transfer, Johnson, Alahi, and Fei-Fei (2016) and Ulyanov et al. (2016) attempted to train a feed-forward network to learn a specific artistic style. In recent years, to improve the efficiency of transferring new styles, researchers have proposed many multiple style transfer (Chen et al. 2017; Dumoulin, Shlens, and Kudlur 2017; Li et al. 2017a; Zhang and Dana 2018) and arbitrary style transfer (Gu et al. 2018; Huang and Belongie 2017; Deng et al. 2020; Lu et al. 2019; Sheng et al. 2018; Yao et al. 2019) methods. Nowadays, neural style transfer has achieved great progress, but due to massive memory cost and small stroke size, ultra-high resolution style transfer is still challenging.

High-Resolution Neural Style Transfer. GPU memory is the main factor that restricts high-resolution style transfer. (An et al. 2020) proposed ArtNet, a lightweight network pruned from GoogLeNet (Szegedy et al. 2015) for neural style transfer. Jing et al. (2020) developed a MobileNet-based lightweight network, significantly reducing the computation complexities compared with the original VGG encoder. (Wang et al. 2020) proposed a distillation-based method, which used the pre-trained VGG19 (Simonyan and Zisserman 2014) as the teacher and a small encoder as the student, successfully rendering high-resolution images up to 6000×6000 pixels on a single 12GB GPU. Although these methods reduce the memory consumption, they will still exhaust the GPU memory when processing ultra-high resolu-
Our URST includes an auxiliary loss termed stroke perceptual loss $L_{sp}$, to enlarge the stroke size. Thanks to the above key designs, we built an unconstrained resolution style transfer system for the first time.

**Overall Architecture**

The goal of the URST framework is to overcome the difficulties in GPU memory limit and small brush strokes when processing ultra-high resolution images (e.g., $10000 \times 10000$ pixels).

**Proposed Method**

**Overall Architecture**

An overview of the URST framework is depicted in Figure 3. Taking an ultra-high resolution content image $I_c$ as input, the pipeline of the URST can be divided into three stages: dividing, stylization, and assembling. (1) In the dividing stage, we first generate a thumbnail image $I_t$ for each content image, and then divide the content image $I_c$ into a sequence of small patches $\{I_p^{i} \mid i = 1, 2, ..., N\}$. (2) In the stylization stage, the thumbnail image $I_t$ is the first to be fed into the style transfer network, to collect the normalization statistics across the network. Then, these normalization statistics are applying to stylize the small patches, obtaining the stylized patches $\{\hat{I}_p^{i} \mid i = 1, 2, ..., N\}$. Here, our style transfer network is not specific. Most existing IN-based methods can be used as the style transfer network. (3) In the assembling stage, all stylized patches are assembled into an ultra-high resolution styled image $\hat{I}_c$.

Since the style transfer network in our framework can be different methods (e.g., AdaIN (Huang and Belongie 2017) and LinearWCT (Li et al. 2019)) whose loss functions are various, for convenience, we define the loss of the selected method as the original loss $L_o$. During training, we first optimize the network with the original loss calculated on the stylized thumbnail. In addition, URST introduces an auxiliary loss, termed stroke perceptual loss, to further improve the quality of ultra-high resolution style transfer. Its core idea is to penalize the perceptual differences in brush strokes between the stylized patch $\hat{I}_p$ and the upsampled patch $\hat{I}_{tp}$ that cropped from the stylized thumbnail $\hat{I}_t$. It should be noticed that the upsampled patch $\hat{I}_{tp}$ plays a role of the learning target. Therefore, the gradient flow of $\hat{I}_{tp}$ is detached.

**Patch-wise Style Transfer**

To process ultra-high resolution images, we propose patch-wise style transfer. Given an ultra-high resolution content image $I_c$, we use a $K \times K$ pixels sliding window with a stride of $S$ to divide the content image $I_c$ into multiple overlapping patches $\{I_p^i \mid i = 1, 2, ..., N\}$. Considering limited GPU memory resources, these patches will be fed to the net-
work in batches. After the loop of patch stylization, we obtain a sequence of stylized patches \( \{I_p^i\}_{i=1, 2, \ldots, N} \). Finally, we discard the overlapping regions on these stylized patches and assemble them into a complete image \( \hat{I}_c \).

Compared with previous methods (An et al. 2020; Wang et al. 2020) that use a full image as input, this patch-wise manner can flexibly process arbitrary high-resolution images, and also be easily plugged into most existing style transfer methods, such as AdaIN (Huang and Belongie 2017), WCT (Li et al. 2017b), and LinearWCT (Li et al. 2019). However, it is evident that the style of stylized patches is inconsistent (see Figure 2(c)), due to the individual normalization statistics calculated from each patch. Therefore, we propose the thumbnail instance normalization (TIN) to address this problem.

**ThumbINance Instance Normalization**

IN is a widely-used normalization layer in neural style transfer. Given an input tensor \( x \in \mathbb{R}^{N \times C \times H \times W} \), IN can be formulated as:

\[
IN(x) = \gamma \left( x - \mu(x) \right) / \sigma(x) + \beta,
\]

(1)

where \( \mu(x), \sigma(x) \in \mathbb{R}^{N \times C} \) are channel-wise statistics; \( \gamma, \beta \in \mathbb{R} \) are trainable affine parameters. However, we found that IN is not applicable to patch-wise style transfer, because stylized patches generated by the IN-based network are inconsistent in style. As demonstrated in Figure 4, we take a simple example to illustrate this problem. In Figure 4(a), we normalize the input as a whole. In Figure 4(b), we divide the input into four patches and normalize them individually. Comparing these two results reveals that the underlying cause of style inconsistency is the individual normalization statistics calculated from each patch.

Based on the above analysis, we propose a simple variant to IN, termed thumbnail instance normalization (TIN). Our TIN receives a patch \( x \in \mathbb{R}^{N \times C \times H \times W} \) and a thumbnail \( t \in \mathbb{R}^{N \times C \times H_t \times W_t} \) as input, and it can be formulated as:

\[
TIN(x, t) = \gamma \left( x - \mu(t) \right) / \sigma(t) + \beta.
\]

(2)

Different from IN, here \( \mu(t), \sigma(t) \in \mathbb{R}^{N \times C} \) are channel-wise mean and standard deviation of the thumbnail input \( t \). In this way, our TIN is able to ensure the style consistency among different patches, as shown in Figure 4(c).

Similarly, instance whitening (IW) (Pan et al. 2019) has the same problem, which is a standardization method based on second-order statistics (i.e., covariance matrix). It is also widely used in many neural style transfer methods (Li et al. 2019, 2017b; Wang et al. 2020; Yoo et al. 2019). Our TIN can be generalized to thumbnail instance whitening (TIW) with minor modifications. We will discuss this in the supplementary material.

**Stroke Perceptual Loss**

Based on the proposed TIN, we present an auxiliary loss for enlarging the stroke size, termed stroke perceptual loss:

\[
L_{sp}(\hat{I}_p, \hat{I}_{tp}) = \left\| \mathcal{F}_l(\hat{I}_p) - \mathcal{F}_l(\hat{I}_{tp}) \right\|^2,
\]

(3)

where \( \mathcal{F}_l \) is the output feature map of the layer \( l \) in the VGG network. \( \hat{I}_p \) is a stylized patch with small brush strokes, and \( \hat{I}_{tp} \) is a stylized patch that cropped and upsampled from the stylized thumbnail \( \hat{I}_c \), which has large brush strokes.

Thanks to the proposed TIN, the input pair \((\hat{I}_p, \hat{I}_{tp})\) has similar content and style, but the stroke size is different. Therefore, \( L_{sp} \) can mainly measure the perceptual differences in brush strokes. In other words, optimizing this loss is to encourage the style transfer network to generate brush strokes as large as that of the target \( \hat{I}_{tp} \).

**Total Loss**

As mentioned above, we define the loss function used in the selected method as the original loss \( L_o \). On this basis, we add the stroke perceptual loss \( L_{sp} \) as an auxiliary loss. Therefore, the total loss is expressed as:

\[
L = L_o + \lambda L_{sp},
\]

(4)

where \( \lambda \) is a weight to balance \( L_o \) and \( L_{sp} \). In our experiments, \( \lambda \) is set to 1.0 by default.
Experiments

Implementation Details

To verify the versatility of our URST, we apply it to 6 representative style transfer methods, including Johnson et al. (Johnson, Alahi, and Fei-Fei 2016), MSG-Net (Zhang and Dana 2018), AdaIN (Huang and Belongie 2017), WCT (Li et al. 2017b), LinearWCT (Li et al. 2019), and Wang et al. (Wang et al. 2020).

In the testing phase, we perform ultra-high resolution style transfer on photography works collected from pexels.com. We use a 1064×1064 pixels sliding window with a stride of 1000 to divide the input image, and the style image used in our framework is 1024×1024 pixels. Besides, we scale the shorter side of the input image to 1024 pixels, as the thumbnail.

During training, our stroke perceptual loss is computed at the relu4_1 layer of the VGG network. Following common practices (Chen and Schmidt 2016; Deng et al. 2020; Li et al. 2019), we use MS-COCO dataset (Lin et al. 2014) as content images and WikiArt dataset (Nichol 2016) as style images, both of which contain roughly 80,000 training samples. Following previous methods, we adopt a VGG19 (Simonyan and Zisserman 2014) pre-trained on ImageNet (Deng et al. 2009) as the loss network. All models are trained with a batch size of 8 on a Titan XP GPU, and other training settings are the same as the original settings in the selected style transfer methods (Huang and Belongie 2017; Li et al. 2019).

Ablation Study

Thumbnail Instance Normalization. As discussed, consistent normalization statistics are important for patch-wise style transfer. To verify this, we conduct experiments of patch-wise style transfer with IN and the proposed TIN, respectively. From Figure 5(b), we can observe that IN leads to the style inconsistency among different patches. Differently, our method avoids this problem by adopting TIN (see Figure 5(c)). In addition, we find that our results are as high-quality as the results demonstrated in Figure 5(a), while our memory consumption is less than 5GB, showing that TIN can approximate the IN statistics of the original ultra-high resolution image, enabling the low memory cost ultra-high resolution style transfer.

Moreover, we compare the stylized results generated by the model with our TIN and the model with random normalization statistics in Figure 7. Although using random normalization statistics can also keep the style consistency among different patches, it destroys the style information extracted from the style image, resulting in the unexpected styles as shown in Figure 7(c). In contrast, using TIN not only ensures the style consistency among different patches, but also maintains the information of the target style.
Figure 6: Ablation study of the proposed stroke perceptual loss $L_{sp}$. Comparison of these stylized images (3000×3000 pixels) indicates that our $L_{sp}$ can significantly enlarge the stroke size of the existing style transfer methods.

Figure 7: Thumbnail statistics vs. random statistics. This comparison demonstrates that using the normalization statistics of thumbnail features is the key to the success of patchwise style transfer.

Figure 8: Mean and standard deviation of feature maps of the VGG19 network under different thumbnail scales. It shows that with the growth of the thumbnail scale, the normalization statistics of feature maps tend to be stable.

**Thumbnail Size.** To further study the relationship between normalization statistics and thumbnail size, we resize an ultra-high resolution image (8192×8192 pixels) to the thumbnails of different scales, and calculate their normalization statistics in the style transfer network. Specifically, we first feed these thumbnails to the encoder (i.e., VGG19) of the style transfer network and obtain the output feature maps of relu1_1, relu2_1, relu3_1, and relu4_1. Then, we calculate the mean and standard deviation of these feature maps, and plot them in Figure 8. Note that when the thumbnail scale is equal to 8192×8192, the normalization statistics is the IN statistics. We see that with the growth of the thumbnail scale, the normalization statistics of feature maps tend to be stable. When the thumbnail scale is larger than 1024×1024 pixels, the TIN statistics are very close to IN statistics. This indicates that TIN can well approximate IN when the thumbnail scale is larger than 1024×1024. In addition, we also conduct a qualitative ablation study for the thumbnail size in the supplementary material, from which the same conclusion can be drawn. To balance speed and style transfer quality, we set the shorter side of the thumbnail to 1024 pixels by default.

**Stroke Perceptual Loss.** As shown in Figure 6 using the proposed stroke perceptual loss $L_{sp}$ as an auxiliary loss for
neural style transfer can significantly enlarge the stroke size of these existing methods. Compared with the baseline results (see Figure 6(b)(d)), with the guidance of $L_{sp}$, these models learned to use thicker lines and sparser textures to depict the scenery, which helps to improve the quality of ultra-high resolution style transfer (see Figure 6(c)(e)).

### Discussion

Different from previous methods (An et al. 2020; Wang et al. 2020), UREST is a versatile framework that can be easily plugged into most existing IN/IW-based methods. Moreover, with the growth of the input resolution, its GPU memory overhead hardly increases. Theoretically, UREST supports style transfer of arbitrary resolution images.

To further demonstrate the effectiveness of UREST, we evaluate it on an ultra-high resolution image of 12000\times8000 pixels (i.e., 96 megapixels), as shown in Figure 9. This result is produced based on AdaIN (Huang and Belongie 2017) and only costs 2.5GB of GPU memory. It also shows that our UREST has achieved superior performance in producing large brush strokes due to the effectiveness of the stroke perceptual loss. In conclusion, to our knowledge, this is the first time to build an unconstrained resolution style transfer system on a single 12GB GPU (Titan XP).

**Limitation**

One limitation of UREST is that it cannot be applied to the optimization-based methods, such as Gatys et al. (Gatys, Ecker, and Bethge 2016) and STROTSS (Kolkin, Salamon, and Shakhnarovich 2019), because these methods do not adopt IN (Ulyanov, Vedaldi, and Lempitsky 2016) or IW (Pan et al. 2019). But we think that “containing IN or IW” is a loose premise, since most existing style transfer methods can meet this prerequisite. In addition, we’d like to point out that these optimization-based methods are relatively slow, which always take more than 100 seconds to stylize an image of 1000\times1000 pixels, are not the best choice for ultra-high resolution style transfer.

**Conclusion**

In this work, we propose UREST, a simple yet effective framework for arbitrary high-resolution style transfer. We perform patch-wise style transfer to process ultra-high resolution input under limited GPU memory resources, and develop a thumbnail instance normalization (TIN) layer to ensure the style consistency among different patches. Moreover, to enlarge the brush strokes in ultra-high resolution stylized results, the stroke perceptual loss $L_{sp}$ is introduced as an auxiliary loss for neural style transfer. Extensive experiments show that our UREST surpasses existing SOTA methods on ultra-high resolution images and can be easily plugged into most existing IN/IW-based methods. Although we mainly study neural style transfer in this work, instance normalization is also widely used in other low-level vision tasks. Therefore, the application of our TIN on other tasks is worth exploring in the future.
References

An, J.; Li, T.; Huang, H.; Shen, L.; Wang, X.; Tang, Y.; Ma, J.; Liu, W.; and Luo, J. 2020. Real-time universal style transfer on high-resolution images via zero-channel pruning. arXiv preprint arXiv:2006.09029.

Chen, D.; Yuan, L.; Liao, J.; Yu, N.; and Hua, G. 2017. Stylebank: An explicit representation for neural image style transfer. In CVPR, 1897–1906.

Chen, T. Q.; and Schmidt, M. 2016. Fast patch-based style transfer of arbitrary style. arXiv preprint arXiv:1612.04337.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In CVPR, 248–255.

Deng, Y.; Tang, F.; Dong, W.; Sun, W.; Huang, F.; and Xu, C. 2020. Arbitrary style transfer via multi-adaptation network. In ACM MM, 2719–2727.

Dumoulin, V.; Shlens, J.; and Kudlur, M. 2017. A learned representation for artistic style. In ICLR.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In CVPR, 2414–2423.

Gatys, L. A.; Ecker, A. S.; Bethge, M.; Hertzmann, A.; and Shechtman, E. 2017. Controlling perceptual factors in neural style transfer. In CVPR, 3985–3993.

Gu, S.; Chen, C.; Liao, J.; and Yuan, L. 2018. Arbitrary style transfer with deep feature reshuffle. In CVPR, 8222–8231.

Huang, X.; and Belongie, S. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In ICCV, 1501–1510.

Jing, Y.; Liu, X.; Ding, Y.; Wang, X.; Ding, E.; Song, M.; and Wen, S. 2020. Dynamic instance normalization for arbitrary style transfer. In AAAI, 4369–4376.

Jing, Y.; Liu, Y.; Yang, Y.; Feng, Z.; Yu, Y.; Tao, D.; and Song, M. 2018. Stroke controllable fast style transfer with adaptive receptive fields. In ECCV, 238–254.

Jing, Y.; Yang, Y.; Feng, Z.; Ye, J.; Yu, Y.; and Song, M. 2019. Neural style transfer: A review. TVCG, 26(11): 3365–3385.

Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016. Perceptual losses for real-time style transfer and super-resolution. In ECCV, 694–711.

Kolkin, N.; Salavon, J.; and Shakhnarovich, G. 2019. Style transfer by relaxed optimal transport and self-similarity. In CVPR, 10051–10060.

Li, X.; Liu, S.; Kautz, J.; and Yang, M.-H. 2019. Learning linear transformations for fast image and video style transfer. In CVPR, 3809–3817.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M.-H. 2017a. Diversified texture synthesis with feed-forward networks. In CVPR, 3920–3928.

Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; and Yang, M.-H. 2017b. Universal style transfer via feature transforms. In NeurIPS, 386–396.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In ECCV, 740–755.

Lu, M.; Zhao, H.; Yao, A.; Chen, Y.; Xu, F.; and Zhang, L. 2019. A closed-form solution to universal style transfer. In ICCV, 5952–5961.

Lu, M.; Zhao, H.; Yao, A.; Xu, F.; Chen, Y.; and Zhang, L. 2017. Decoder network over lightweight reconstructed feature for fast semantic style transfer. In ICCV, 2469–2477.

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

Pan, X.; Zhan, X.; Shi, J.; Tang, X.; and Luo, P. 2019. Switchable whitening for deep representation learning. In ICCV, 1863–1871.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. NeurIPS, 32: 8026–8037.

Sanakoyeu, A.; Kotovenko, D.; Lang, S.; and Ommer, B. 2018. A style-aware content loss for real-time hd style transfer. In ECCV, 698–714.

Shen, F.; Yan, S.; and Zeng, G. 2018. Neural style transfer via meta networks. In CVPR, 8061–8069.

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

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In CVPR, 1–9.

Ulyanov, D.; Lebedev, V.; Vedaldi, A.; and Lempitsky, V. S. 2016. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. In ICML, 1349–1357.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2016. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022.

Wang, H.; Li, Y.; Wang, Y.; Hu, H.; and Yang, M.-H. 2020. Collaborative distillation for ultra-Resolution universal style transfer. In CVPR, 1860–1869.

Wang, X.; Oxholm, G.; Zhang, D.; and Wang, Y.-F. 2017. Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer. In CVPR, 5239–5247.

Yao, Y.; Ren, J.; Xie, X.; Liu, W.; Liu, Y.-J.; and Wang, J. 2019. Attention-aware multi-stroke style transfer. In CVPR, 1467–1475.

Yoo, J.; Uh, Y.; Chun, S.; Kang, B.; and Ha, J.-W. 2019. Photorealistic style transfer via wavelet transforms. In ICCV, 9036–9045.

Zhang, H.; and Dana, K. 2018. Multi-style generative network for real-time transfer. In ECCV Workshops.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In ICCV, 2223–2232.
Appendix

Thumbnail Instance Whitening

In recent years, instance whitening (IW) (Pan et al. 2019) has been a widely-used module in neural style transfer (Li et al. 2019, 2017b, Wang et al. 2020, Yoo et al. 2019). Instead of normalizing data using channel-wise mean and standard deviation like instance normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016), IW whitens the data using its covariance matrix, which gives rise to better stylization quality than IN. Given an input tensor \( x \in \mathbb{R}^{N \times C \times H \times W} \), IW can be formulated as:

\[
IW(x) = [\Sigma(x)]^{-\frac{1}{2}} (x - \mu(x)),
\]

where \( \Sigma(x) \in \mathbb{R}^{N \times C \times C} \) and \( \mu(x) \in \mathbb{R}^{N \times C} \) are covariance matrices and mean vectors.

However, IW has the same problem as IN when performing patch-wise style transfer. As discussed, the style of stylized patches is inconsistent due to the individual statistics calculated from each patch. To address this problem, we generalize our TIN to its IW version, namely thumbnail instance whitening (TIW). As same as TIN, our TIW receives a patch \( x \in \mathbb{R}^{N \times C \times H \times W} \) and a thumbnail \( t \in \mathbb{R}^{N \times C \times H_t \times W_t} \) as input, and it can be formulated as:

\[
TIW(x, t) = [\Sigma(t)]^{-\frac{1}{2}} (x - \mu(t)),
\]

where \( \Sigma(t) \in \mathbb{R}^{N \times C \times C} \) and \( \mu(t) \in \mathbb{R}^{N \times C} \) are covariance matrices and mean vectors of the thumbnail input \( t \). In this way, we avoid the style inconsistency among different patches for most existing style transfer methods.

Memory and Speed Analysis

GPU Memory Usage. In this section, we report the memory cost for a single content image of different resolutions. As shown in Table 1, most existing methods cannot process high-resolution images (e.g., 4000×4000 pixels) with limited memory. (Wang et al. 2020) is a recent distillation-based method designed for high-resolution style transfer, but it can only process up to 6000×6000 pixels. Unlike them, our URST can keep the memory cost below 5GB, and with the growth of the input resolution, our GPU memory overhead hardly increases. Theoretically, URST supports style transfer of arbitrary resolution images.

Time Cost. The time cost of URST is linear to the number of patches. For example, on a single Titan XP GPU, AdaIN (Huang and Belongie 2017) takes about 0.15 seconds to stylize a 1000×1000 image, thus “URST + AdaIN” takes about 15 seconds to process a 10000×10000 image.

More Applications

Image-to-Image Translation. To further verify the effectiveness of our method, we conduct an extra experiment on CycleGAN (Zhu et al. 2017), a representative image-to-image translation method. Specifically, we adopt its official code\(^1\) and pre-trained models, and replace all IN layers with our TIN layers. As shown in Figure 10, our TIN can also be successfully applied on CycleGAN.

Flexible User Control. Our URST is compatible with various user controls. Here, we take style weight control as an example, as shown in Figure 11.

Table 1: GPU memory comparison (in GB). All results are tested by PyTorch (Paszke et al. 2019) on a Titan XP GPU (12GB). “-” denotes out-of-memory.

| Res. | Johnson | MSG-Net | Adain | WCT | LinearWCT | Wang et al. |
|------|---------|---------|-------|-----|-----------|-------------|
|      | Orig.   | Ours    | Orig. | Ours| Orig. | Ours | Orig. | Ours |
| 1000² | 1.21 | 2.12 | 2.55 | 3.21 | 1.40 | 2.17 | 1.84 | 4.61 | 2.79 | 3.44 | 0.90 | 1.94 |
| 2000² | 4.31 | 2.12 | 7.15 | 3.21 | 4.62 | 2.17 | 4.10 | 4.61 | 6.85 | 3.44 | 1.80 | 1.94 |
| 3000² | 9.48 | 2.12 | 12.95 | 3.21 | 9.72 | 2.17 | 7.87 | 4.61 | 1.44 | 3.27 | 1.94 |
| 4000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 5.36 | 1.94 |
| 5000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 8.00 | 1.94 |
| 6000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 11.28 | 1.94 |
| 7000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 1.94 |
| 8000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 1.94 |
| 9000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 1.94 |
| 10000² | 12.12 | 3.21 | 12.31 | 3.21 | 3.21 | 2.17 | 1.46 | 4.61 | 3.44 | 1.94 |

Figure 10: Experiments on CycleGAN (monet-to-photo task). The experimental settings are the same as Figure 5.

Figure 11: Trade-off between the content and style image. These results are based on LinearWCT (Li et al. 2019).

More Experimental Results

Qualitative Ablation for Thumbnail Size. As presented in Figure 12, the thumbnail size increases, the result is gradually close to baseline’s style transfer performance, showing that TIN can approximate IN statistics of ultra-high-resolution images when the thumbnail size reaches 1024.

More Ultra-high Resolution Results. More ultra-high resolution results of 12000×8000 pixels (i.e., 96 megapixels) produced by our URST framework are presented in Figure 13 which are based on AdaIN (Huang and Belongie 2017) and LinearWCT (Li et al. 2019), respectively.

https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

1. \[https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix\]

More Applications

Image-to-Image Translation. To further verify the effectiveness of our method, we conduct an extra experiment on CycleGAN (Zhu et al. 2017), a representative image-to-image translation method. Specifically, we adopt its official code\(^{1}\) and pre-trained models, and replace all IN layers with our TIN layers. As shown in Figure 10, our TIN can also be successfully applied on CycleGAN.

Flexible User Control. Our URST is compatible with various user controls. Here, we take style weight control as an example, as shown in Figure 11.

More Experimental Results

Qualitative Ablation for Thumbnail Size. As presented in Figure 12, the thumbnail size increases, the result is gradually close to baseline’s style transfer performance, showing that TIN can approximate IN statistics of ultra-high-resolution images when the thumbnail size reaches 1024.

More Ultra-high Resolution Results. More ultra-high resolution results of 12000×8000 pixels (i.e., 96 megapixels) produced by our URST framework are presented in Figure 13 which are based on AdaIN (Huang and Belongie 2017) and LinearWCT (Li et al. 2019), respectively.

https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

More Applications

Image-to-Image Translation. To further verify the effectiveness of our method, we conduct an extra experiment on CycleGAN (Zhu et al. 2017), a representative image-to-image translation method. Specifically, we adopt its official code\(^{1}\) and pre-trained models, and replace all IN layers with our TIN layers. As shown in Figure 10, our TIN can also be successfully applied on CycleGAN.

Flexible User Control. Our URST is compatible with various user controls. Here, we take style weight control as an example, as shown in Figure 11.

More Experimental Results

Qualitative Ablation for Thumbnail Size. As presented in Figure 12, the thumbnail size increases, the result is gradually close to baseline’s style transfer performance, showing that TIN can approximate IN statistics of ultra-high-resolution images when the thumbnail size reaches 1024.

More Ultra-high Resolution Results. More ultra-high resolution results of 12000×8000 pixels (i.e., 96 megapixels) produced by our URST framework are presented in Figure 13 which are based on AdaIN (Huang and Belongie 2017) and LinearWCT (Li et al. 2019), respectively.

https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix
Figure 13: More ultra-high resolution stylized results (12000×8000 pixels) produced on a single 12GB GPU (Titan XP). On the upper left are the content image and style image. Six close-ups (660×330 pixels) are shown on the right side of the stylized result.