Erschienen in: ICASSP 2024 : IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Proceedings. Piscataway, NJ: IEEE, 2024, pp. 3800-3804
https://doi.org/10.1109/icassp48485.2024.10446950

LIGHTING IMAGE/VIDEO STYLE TRANSFER METHODS BY ITERATIVE CHANNEL PRUNING

Kexin Wu[†], Fan Tang[⋆], Ning Liu[◁], Oliver Deussen[‡], Thi-ngoc-hanh Le[*], Weiming Dong[⊖], Tong-yee Lee[*]

[†] Jilin University, [⋆] ICT-CAS, [◁] Midea Group, [‡] University of Konstanz
[*] National Cheng Kung University, [⊖] CASIA

ABSTRACT

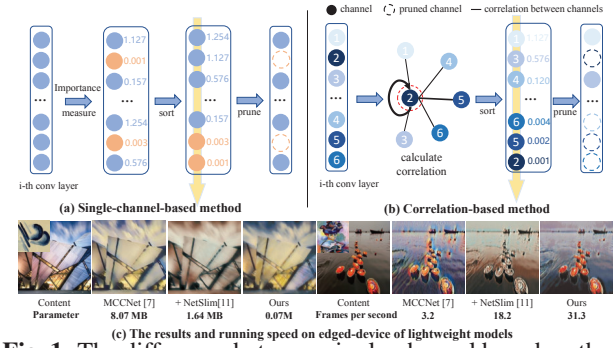
Deploying style transfer methods on resource-constrained devices is challenging, which limits their real-world applicability. To tackle this issue, we propose using pruning techniques to accelerate various visual style transfer methods. We argue that typical pruning methods may not be well-suited for style transfer methods and present an iterative correlation-based channel pruning (ICCP) strategy for encoder-transform-decoder-based image/video style transfer models. The correlation-based channel regularization preserves the feature distributions for content and style references, and the iterative pruning strategy prevents layer collapse when pruning on the encoder-decoder structure. Experiments demonstrate that the proposed ICCP can generate visual competitive results compared to SOTA style transfer methods and significantly reduces the number of parameters (at least 70K) and inference time. Model is available at `https://github.com/wukx-wukx/ICCP`.

Index Terms— visual style transfer, model pruning

1. INTRODUCTION

With the rapid development of deep learning, artificial intelligence generated content (AIGC) has become a popular research area. As a typical AIGC task, style transfer [1] is a creative computer graphics and multimedia application based on modern visual signal processing techniques. By rendering an image/video with artistic features guided by a style reference, visual style transfer (VST) applications enable content creators and users to generate restyled visual media.

Although significant progress has been made for image/video stylization tasks, most of these methods [2–5] focus on improving the visual effects by using high-performance cloud servers. The high memory usage and latency prevent these methods from being widely used on resource-constrained mobile devices. To create lightweight models, existing structured [6, 7] or unstructured [8, 9] pruning methods try to cut off the channels directly or remove less important connections, respectively.



(c) The results and running speed on edged-device of lightweight models

Fig. 1. The difference between single-channel based methods and our correlation-based method.

However, most pruning methods are designed for classification and object detection and are not suitable for VST tasks. As shown in Fig. 1, when pruning 80% parameters of MCCNet [4] by NetSlim [7], the outputs fail to maintain style consistency with the input style reference (shown on the top left of the content image). The reasons are two folds for such observation: 1. VST methods usually employ an encoder-decoder-based backbone, which is more complex than plain CNN networks; 2. For image classification, network pruning prioritizes the most discriminative channels, typically those that elicit high response outputs. VST tasks rely on the distribution of features across different channels to measure style information [4]. Pruning based on single-channel criteria could potentially compromise the preservation of valuable style information. Therefore, it is critical to take the nuances of the specific task at hand into account when deciding on the appropriate pruning criteria to ensure optimal performance.

In this study, we present a novel approach to generate lightweight VST models that are suitable for mobile devices. Our approach involves a correlation-based channel regularization process that evaluates the similarities among channels within a layer, as shown in Fig. 1(b). Unlike single-channel-based pruning methods, which consider the most discriminative channels, we prune channels that produce similar outputs based on the degree of their correlation. This correlation-based channel regularization term helps to increase the gap

This work was supported in part by the National Science Foundation of China 62102162, 61832106, U20B2070 and the National Science and Technology Council (under no 110-2221-E-006-135-MY3), Taiwan. (Corresponding author: Fan Tang.)

Konstanzer Online-Publikations-System (KOPS)
URL: http://nbn-resolving.de/urn:nbn:de:bsz:352-2-1m528lc3rvpg8

between the preserved and pruned channels and reduce the similarities of the preserved channels. To prevent layer collapse, where all channel outputs in a layer are pruned, we adopt an iterative pruning strategy that is more suitable to the encoder-transform-decoder VST model structure.

In summary, (1) we propose a correlation-based channel regularization by adding restraints on the correlations among channels, which can reduce redundant channels and keep channel diversity; (2) we introduce an iterative pruning strategy to keep the whole style transfer performance by gradually increasing the prune ratio and pruning encoder and decoder alternatively; (3) experiments on image and video stylization tasks prove the efficiency and generalization ability of the proposed method, which could be applied to different VST models in a plug-and-play way and reduce 99.13% parameters and save 92.78% inference time.

2. METHODS

2.1. Problem Definition

Given a content reference (image or video frame) I_c and a style reference I_s, arbitrary style transfer methods aim to generate a re-stylized image I_{cs}. The style transfer model $\mathbf{M}$ consists of an encoder $\mathbf{E}$, a transform module $\mathbf{T}$, and a decoder $\mathbf{D}$. By the encoder $\mathbf{E}$, we obtain content representations f_c of the content image and extra style embedding f_s. Then f_c and f_s are fused through $\mathbf{T}$ to obtain the stylized representation f_{cs} which maintains the structure of the content image while being rendered to the desired style. Finally, $\mathbf{D}$ decodes f_{cs} to I_{cs} with the help of style transfer losses. We aim to generate a lightweight model $\mathbf{M}'$ (with $\mathbf{E}'$-$\mathbf{T}'$-$\mathbf{D}'$) that allows us to generate similar visual effects to $\mathbf{M}$, but reduces the needed resources when computed on mobile devices.

2.2. Correlation-based Channel Regularization

Different from traditional single-channel-based pruning methods, to extend the difference between preserved and pruned channels while also reducing the preserved channels' similarities, we propose a correlation-based channel regularization term. In detail, when training the network, we have a convolutional layer l with output feature map $F^l \in \mathbb{R}^{B\times C\times H\times W}$, where B, C, H, W denote batch size, channel number, height and weight, respectively. For simplicity, we denote F^l as F. Then we obtain a channel index subset that satisfies the following conditions:

$$R = \{i|\gamma_i \geq t\}, \tag{1}$$

where γ_i uses the definition in article [7], denotes the value of the scale factor corresponding to the i_{th} channel of current convolutional layer. t is the scale factor threshold, whose value depends on the prune ratio r. And the size of the R is C'. So we have

$$F' = F_{:,R,:,:}, F' \in \mathbb{R}^{B\times C'\times H\times W}, \tag{2}$$

We convert F' to $F' \in \mathbb{R}^{B\times C'\times HW}$ and get its transpose $(F')^T \in \mathbb{R}^{B\times HW\times C'}$, then calculate the similarities among the C' channels

$$S = F'(F')^T, \tag{3}$$

where $S \in \mathbb{R}^{B\times C'\times C'}$, then add the values of all dimensions to receive a value $S' \in \mathbb{R}$. Adding all layers' similarity values, we get

$$S^L = \sum_{l=1}^{L} S', \tag{4}$$

then we add an L1 regularization on S^L to get correlation-based regularization term $L_c = |S^L|$. Therefore, the whole optimization object for image-stylized approaches is:

$$L_{total} = L_{stylized}(f(I_c, I_s), I_c, I_s) + \lambda \sum |\gamma| + \eta L_c, \tag{5}$$

where λ and η are weight factors, γ denotes the value of the scale factor which is defined by Liu *et al.* [7].

In the optimization process, the third term we proposed in Eq. (5) will get smaller, which means the similarities among the remaining channels are getting smaller. In other words, there will not be remaining channels focusing on a similar area, and thus this removes redundancy. The second term, which is adapted from Liu *et al.* [7], makes the differences between pruned and remaining channels greater. The larger difference makes it easier for our algorithm to identify "hard" channels, i.e., channels for which it is difficult to determine whether to be pruned or not.

2.3. Iterative Layer-wise Pruning Strategy

For the $\mathbf{E}$-$\mathbf{T}$-$\mathbf{D}$ based VST approaches, $\mathbf{E}$ is responsible for extracting features from style and content images, while $\mathbf{D}$ (symmetrical structure with $\mathbf{E}$) is used for transforming the features to stylized images. Since it is unreasonable to deal with the encoder and decoder independently (ignoring the symmetry between them), we deal with the encoder and decoder alternatively to exploit their symmetry. When slimming the models by using an extremely large pruning ratio, a typical train-pruning-fine-tuning pipeline will prune too many channels directly at once. To address this problem, we increase the prune ratio gradually each iteration by alternatively pruning the encoder and decoder. Furthermore, different from sorting all channels of all layers by their importance [7], we prune each layer separately to avoid affecting the correlation among channels of each layer.

The overall pruning process is shown in Algorithm 1. We set the number of iterations to $N_I = 2$ and control the number of channels pruned for each layer. At the limit, there will be a fixed number of channels remaining for each layer. Thus the total prune ratio r of the model is

$$r = r_1 + r_2 - r_1 r_2, \tag{6}$$

where r_1, r_2 are the prune ratio of different iterations.

For typical approaches, there will be one iteration, training, pruning, and fine-tuning $\mathbf{E}$ and $\mathbf{D}$ directly with a large prune ratio. Since our method has two iterations, it prunes

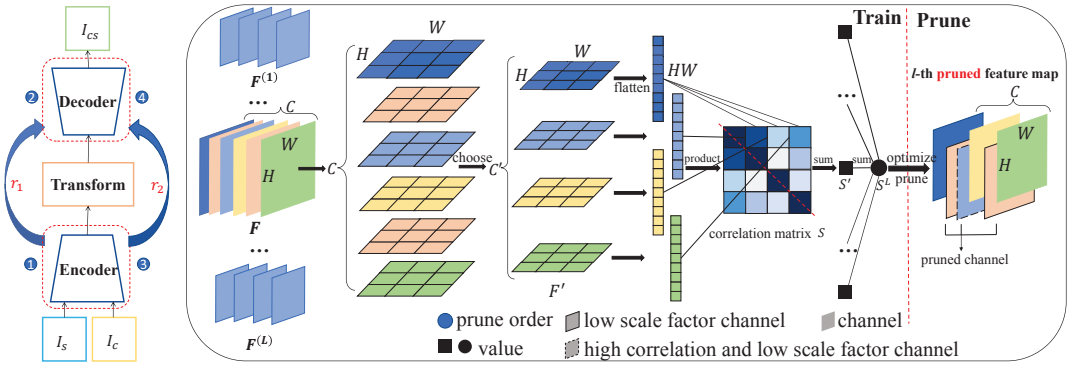


Fig. 2. ICCP pipeline and creating L_c. **Left:** Pipeline design. I_c and I_s are inputs and denote content image and style image, respectively. I_{cs} is the style transferred image. The circle with number and arrows represents the order of Iterative pruning, r_1, r_2 denote the prune ratio in different iterations. The red dotted rectangle marks the modules that will be pruned (Encoder and Decoder). **Right:** Correlation-based channel regularization. Taking the first iteration pruning Encoder as an example, the front of the red dotted line represents the training process, while the back represents the pruning process.

E and **D** alternatively and increases the prune ratio gradually until it reaches the desired value. And each time **E** or **D** is processed, the training, pruning, fine-tuning process is included. Through applying iterative pruning, we can take advantage of the symmetry of encoder and decoder structures, and this way avoid layer-collapse and performance drops.

Algorithm 1 Iterative layer-wise prune.

Input: Pre-trained model **M** {**E**-**T**-**D**}, training data $\{I_s, I_c\}$, number of iterations N_I, prune ratio $\{r_1,...,r_N\}$;
Output: The compressed model **M**′ {**E**′- **T**′- **D**′};

for $i \in \{1, ..., N_I\}$ **do**
  Train **E** with L_{total};
  Prune each layer of **E** with prune ratio r_i;
  E $\rightarrow$ **E**′;
  Fine-tune **E**′ and **T**′;
  Train **D** with L_{total};
  Prune each layer of **D** with prune ratio r_i;
  D $\rightarrow$ **D**′;
  Fine-tune **D**′ and **T**′;
  M′ $\rightarrow$ **M**
end for
return compressed model **M**′;

3. EXPERIMENTS

3.1. Datasets and Experimental Settings

Datasets. We use MS-COCO [10] and WikiArt [11] as content and style image datasets for network training. MS-COCO is a dataset with a total of 2.5 million labeled instances in 328k images, and the WikiArt dataset contains many artworks in different styles.

Training, Pruning and Fine-tuning Settings. We select MCCNet [4] as a basic visual stylization approach and choose NetSlim [7], AKECP [12], CHIP [13] and CPST [14] as contrast pruning methods to prune MCCNet. For our ICCP, when training, we set the weight factor λ=1e-4, η=1e-7, the learning rate to 1e-4, and the batch size to 8. The fine-tuning process has the same setting as training but removes our proposed correlation-based channel regularization term L_c. The prune ratios for two iterations r_1 and r_2 are set to 0.3 and 0.857, respectively, creating an overall prune ratio of 0.9. And the training and fine-tuning steps are all set to 40,000. For other pruning methods, we follow their settings but set the prune ratio $r = 0.9$. We implement all these pruning methods on two NVIDIA TITAN RTX and test the lightweight models on the Samsung Galaxy S10.

Table 1. Quantitative results for different pruning methods.

Method	FLOPs	Temporal Loss	Parameters		
			Encoder	Transform Module	Decoder
MCCNet [4]	209.32G	7.30×10^{-8}	3.51M	1.05M	3.51M
+NetSlim [7]	9.09G	6.70×10^{-8}	0.48M	1.05M	0.11M
+AKECP [12]	209.32G	7.16×10^{-8}	3.51M	1.05M	3.51M
+CHIP [13]	12.14G	6.49×10^{-8}	0.14M	1.05M	1.25M
+CPST [14]	7.30G	6.50×10^{-8}	0.21M	1.05M	0.10M
+Ours	**2.11G**	6.44×10^{-8}	**0.03M**	**0.01M**	**0.03M**

Table 2. Average run time (seconds) for our method and other methods with three input sizes on Samsung Galaxy S10.

Input size	MCCNet [4]	+NetSlim [7]	+AKECP [12]	+CHIP [13]	+CPST [14]	+ICCP (Ours)
128×128	0.310	0.055	0.298	0.057	0.061	**0.032**
255×255	0.770	0.134	0.758	0.141	0.131	**0.072**
512×512	2.757	0.478	2.829	0.516	0.546	**0.199**

3.2. Efficiency Analysis

In this study, we compare the parameters and FLOPs of MCCNet before pruning and after using NetSlim [7], AKECP [12], CHIP [13], CPST [14] and our ICCP. The comprehensive result is shown in Table 1. In total, ICCP achieves the lowest FLOPs and parameters. The encoder, decoder, and transform module parameters have been reduced by 99.15%, 99.15%,

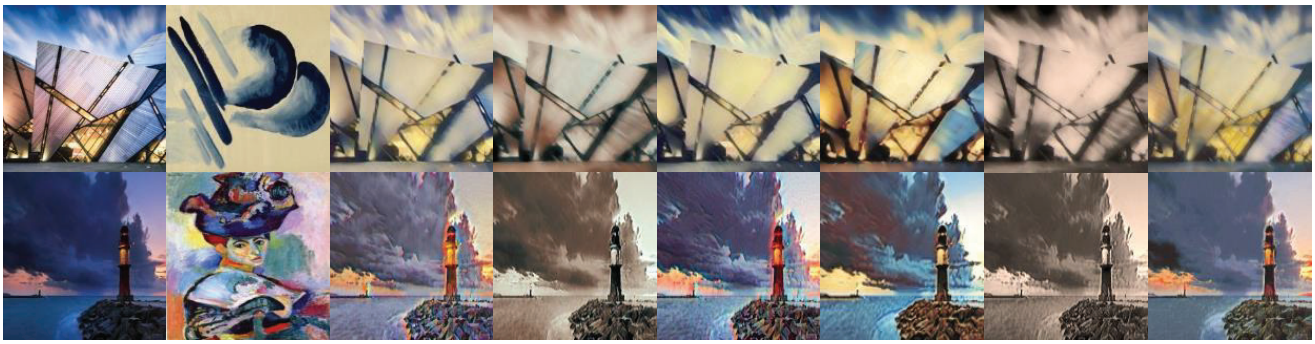


Fig. 3. Visual comparisons for different pruning methods. NetSlim and CPST fail to generate stylized results for layer collapsing during pruning; AKECP does not reduce the model size; CHIP produces unexpected color shifts compared with the results of MCCNet. Compared with these methods, our method could generate similar results with MCCNet while significantly reducing parameters.

and 99.05%, respectively. Likewise, FLOPs have been reduced by 98.99% when using ICCP. AKECP achieves compression by setting some parameters to zero, exhibiting similar parameters and FLOPs to MCCNet. CHIP and CPST remove the channels directly but reserve higher parameters and FLOPs than our ICCP.

Our ICCP achieves the fastest inference time ($0.032s$, $0.072s$, and $0.199s$) with all three input image sizes. For an input size of 512×512, our inference speed is improved by 92.78% compared to the original MCCNet [4]. Our proposed method enables the style transfer task in a real-time performance, potentially enlarging the scope of application towards different VST methods. To measure the coherence of the videos generated from different pruning methods, we calculate temporal loss following [15]. The result is shown in Table 1. We observe a small difference between these values, demonstrating that our method can generate relatively stable videos even if the number of parameters and memory requirement is reduced significantly.

3.3. Visual Quality Assessment

We compare stylized images and videos generated by different pruning methods, including MCCNet [4], MCCNet + NetSlim [7], MCCNet + AKECP [12], MCCNet + CHIP [13], MCCNet + CPST [14], and MCCNet + ours. It should be noted that CPST is a channel pruning method specially designed for style transfer. Fig. 3 shows that our ICCP can generate visually appealing results compared with other single/multi-channel-based pruning methods.

Furthermore, we compare the image style transfer result of our pruned model with several SOTA image stylization methods, including AdaIN [16], MCCNet [4], Artflow [17], AdaAttn [18]. AdaIN, MCCNet, and AdaAttn are encoder-transform-decoder-based CNN approaches while Artflow is based on the flow model. Our method yields competitive and comparable results to the aforementioned SOTA methods, as

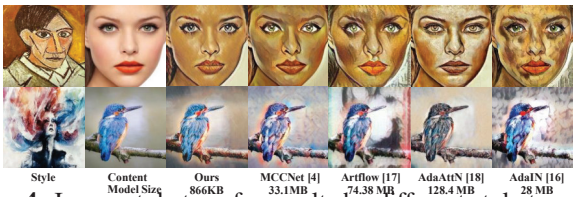


Fig. 4. Image style transfer results by different style transfer approaches. Our method could significantly reduce the model size while preserving the visual quality.

depicted in Fig. 4. Specifically, our ICCP effectively transfers the given styles for all the shown cases while accurately preserving the input content structure/patterns. In certain instances, our ICCP performs even better. For example, as exemplified in the 1st row, our ICCP can generate a more realistic, closer-to-reference style (vs. AdaAttN) without noise (vs. AdaIN and Artflow) on the girl's face. In the 2nd row, the generated image of MCCNet has a highlight around the bird's body, but not in our approach. The visual results of our ICCP are comparable to those of these image stylization methods, with much less memory requirement and computation time.

4. CONCLUSIONS

This paper proposes an iterative correlation-based channel pruning (ICCP) strategy for encoder-transform-decoder-based image/video style transfer methods, which can be deployed on resource-constrained devices with fewer parameters and faster inference speed. The proposed ICCP introduces a channel regularization that adds restraints on channel correlation and an iterative pruning strategy to avoid the layer collapse problem. Quantitative and qualitative measurements adequately prove the validity of our method.

Limitations. Although the proposed ICCP approach achieves a significant reduction in parameters, the training process still requires considerable time to compute correlations. In the future, we aim to explore methods to reduce the model training times, leading to better results.

5. REFERENCES

[1] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song, "Neural style transfer: A review," *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, vol. 26, no. 11, pp. 3365–3385, 2020.

[2] 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 (AAAI)*, 2020, pp. 4369–4376.

[3] Minxuan Lin, Fan Tang, Weiming Dong, Xiao Li, Changsheng Xu, and Chongyang Ma, "Distribution aligned multimodal and multi-domain image stylization," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 3, pp. 1–17, 2021.

[4] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu, "Arbitrary video style transfer via multi-channel correlation," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021, pp. 1210–1217.

[5] Quan Wang, Sheng Li, Xinpeng Zhang, and Guorui Feng, "Multi-granularity brushstrokes network for universal style transfer," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 18, no. 4, pp. 1–17, 2022.

[6] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann, "Towards optimal structured cnn pruning via generative adversarial learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2790–2799.

[7] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2736–2744.

[8] Zhenhua Liu, Jizheng Xu, Xiulian Peng, and Ruiqin Xiong, "Frequency-domain dynamic pruning for convolutional neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 1051–1061.

[9] Alireza Aghasi, Afshin Abdi, Nam Nguyen, and Justin Romberg, "Net-trim: Convex pruning of deep neural networks with performance guarantee," 2017, vol. 30.

[10] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 740–755.

[11] Fred Phillips and Brandy Mackintosh, "Wiki art gallery, inc.: A case for critical thinking," *Issues in Accounting Education*, vol. 26, no. 3, pp. 593–608, 2011.

[12] Haonan Zhang, Longjun Liu, Hengyi Zhou, Wenxuan Hou, Hongbin Sun, and Nanning Zheng, "Akecp: Adaptive knowledge extraction from feature maps for fast and efficient channel pruning," in *Proceedings of the ACM International Conference on Multimedia*, 2021, pp. 648–657.

[13] Yang Sui, Miao Yin, Yi Xie, Huy Phan, Saman Aliari Zonouz, and Bo Yuan, "Chip: Channel independence-based pruning for compact neural networks," 2021, vol. 34, pp. 24604–24616.

[14] Minseong Kim and Hyun-Chul Choi, "Compact image-style transfer: Channel pruning on the single training of a network," *Sensors*, vol. 22, no. 21, pp. 8427, 2022.

[15] Wenjing Wang, Jizheng Xu, Li Zhang, Yue Wang, and Jiaying Liu, "Consistent video style transfer via compound regularization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, vol. 34, pp. 12233–12240.

[16] 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 (ICCV)*, 2017, pp. 1501–1510.

[17] Jie An, Siyu Huang, Yibing Song, Dejing Dou, Wei Liu, and Jiebo Luo, "Artflow: Unbiased image style transfer via reversible neural flows," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 862–871.

[18] 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 International Conference on Computer Vision (ICCV)*, 2021, pp. 6649–6658.