Neural Preset for Color Style Transfer

(CVPR 2023)

Paper Authors:

Zhanghan Ke, Yuhao Liu, Lei Zhu, Nanxuan Zhao, Rynson W.H. Lau

Presentation author:

Umut Özyurt

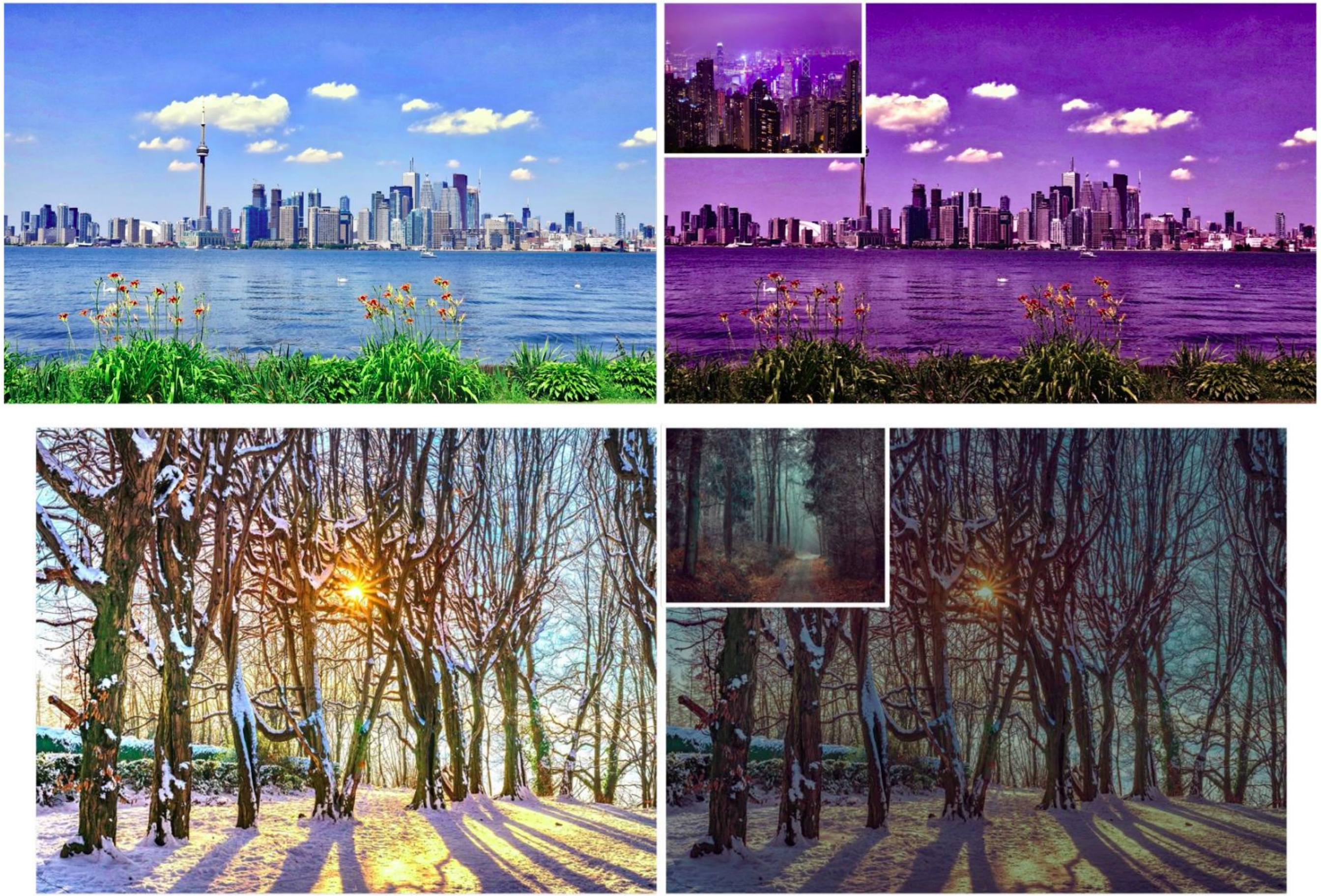
MIDDLE EAST TECHNICAL UNIVERSITY

This presentation was prepared for the CENG796 - Deep Generative Models course for the Spring 2024 term. It is part of the coursework assigned by our lecturer, Assoc. Prof. Ramazan Gokberk Cinbis.

Input Image (8K Resolution)

Ours (0.061s)

Output examples from the presented paper



Output examples from the presented paper

Figures (from the presented papers supplemental materials):
https://openaccess.thecvf.com/content/CVPR2023/supplemental/Ke_Neural_Preset_for_CVPR_2023_supplemental.pdf

METU

Overview of the presentation

- → Introduction
- → Related work
- → Method
- → Experiments
- → Main Contribution and Pros/Cons
- → Conclusion
- → Q & A

1- Introduction

Problem Definition

→ Transfer only colors from an image without altering the targets texture. Hence, it is NOT artistic style transfer.

→ Utilize self-supervised learning.

→ Make the inference time fast.

→ Use feasible GPU memory.

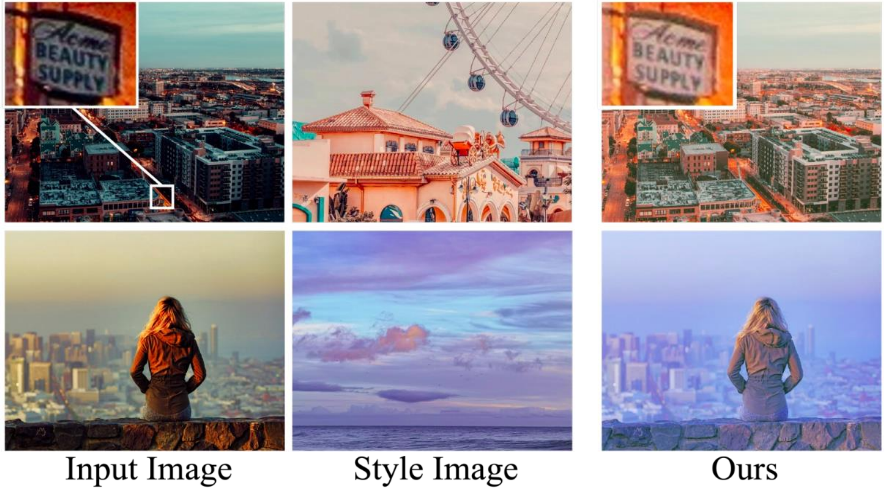


Figure 1. Example color style transfers to demonstrate how well the texture remained as it should.

Figure 1 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

Why Color Style Transfer?

- Producing different styles of images, just like using different customizable Instagram filters.

- Further utilizing the 2-stage transfer approach, switching styles are genuinely fast.

- Using the models on different tasks, such as low-light image enhancement , underwater image correction , image dehazing, and image harmonization.

Application Examples

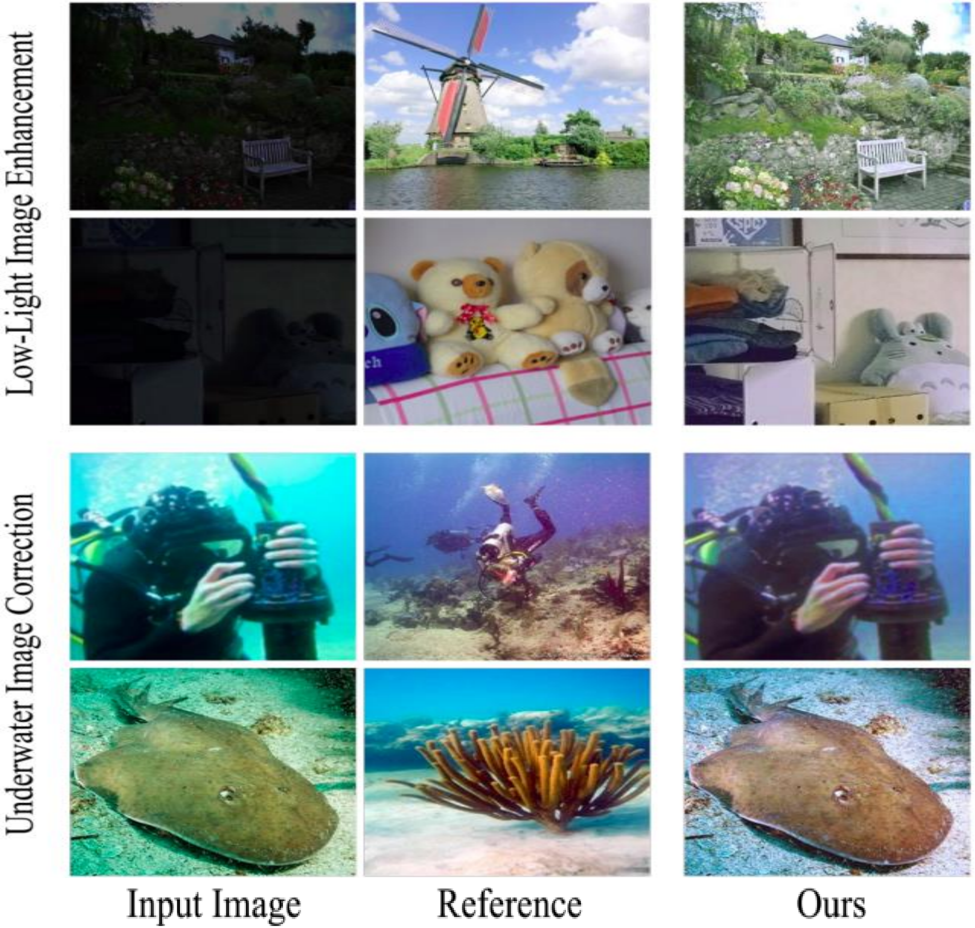


Figure 2. Applications of color style transfer model on Low-Light Image Enhancement and Underwater Image Correction Tasks

Figure 2 (from the presented papers supplemental materials): https://openaccess.thecvf.com/content/CVPR2023/supplemental/Ke_Neural_Preset_for_CVPR_2023_supplemental.pdf

Application Examples

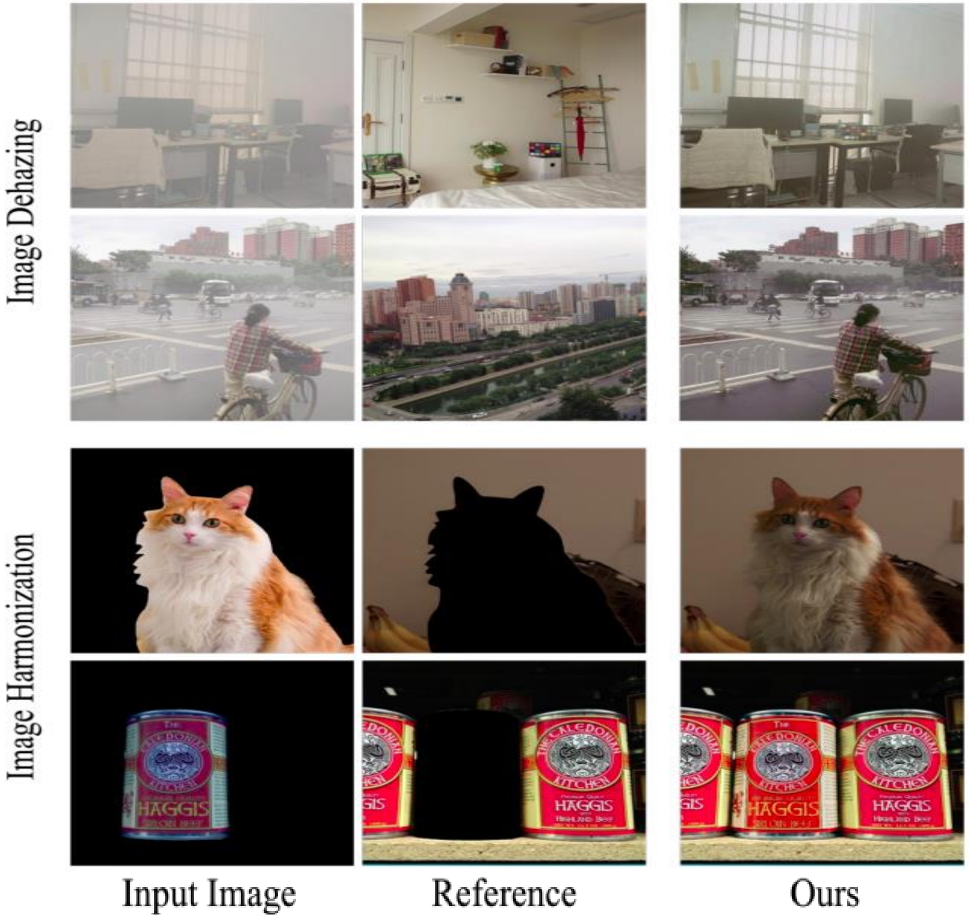


Figure 3. Applications of color style transfer model on Image Dehazing and Image Harmonization tasks.

METU

Figure 3 (from the presented papers supplemental materials): https://openaccess.thecvf.com/content/CVPR2023/supplemental/Ke_Neural_Preset_for_CVPR_2023_supplemental.pdf

2-
Related Work

Related Work

- Color Style Transfer
- Deterministic Color Mapping with CNNs.
- Self-Supervised Learning (SSL).

Related Work Discussion

- What is missing or should be improved so far?
- Visual artifacts
- High memory requirements, high inference times.
- Inefficient style switch for images and low scalability for high resolution.

3-
Method

Two-Stage Pipeline

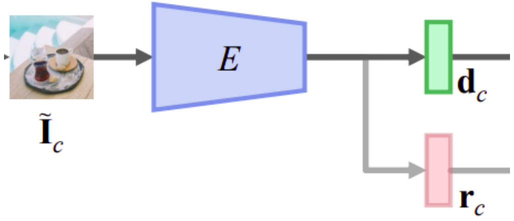


Figure 4. Proposed two-stage pipeline from the paper

Two-Stage Pipeline

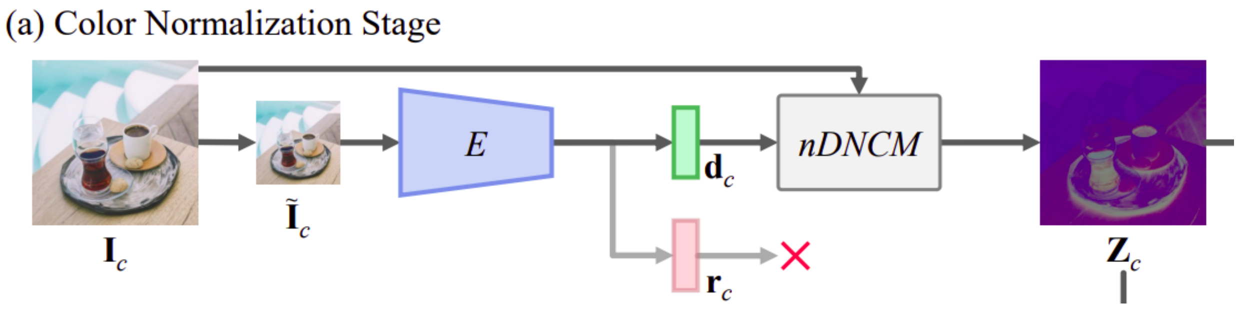


Figure 4. Proposed two-stage pipeline from the paper

Two-Stage Pipeline

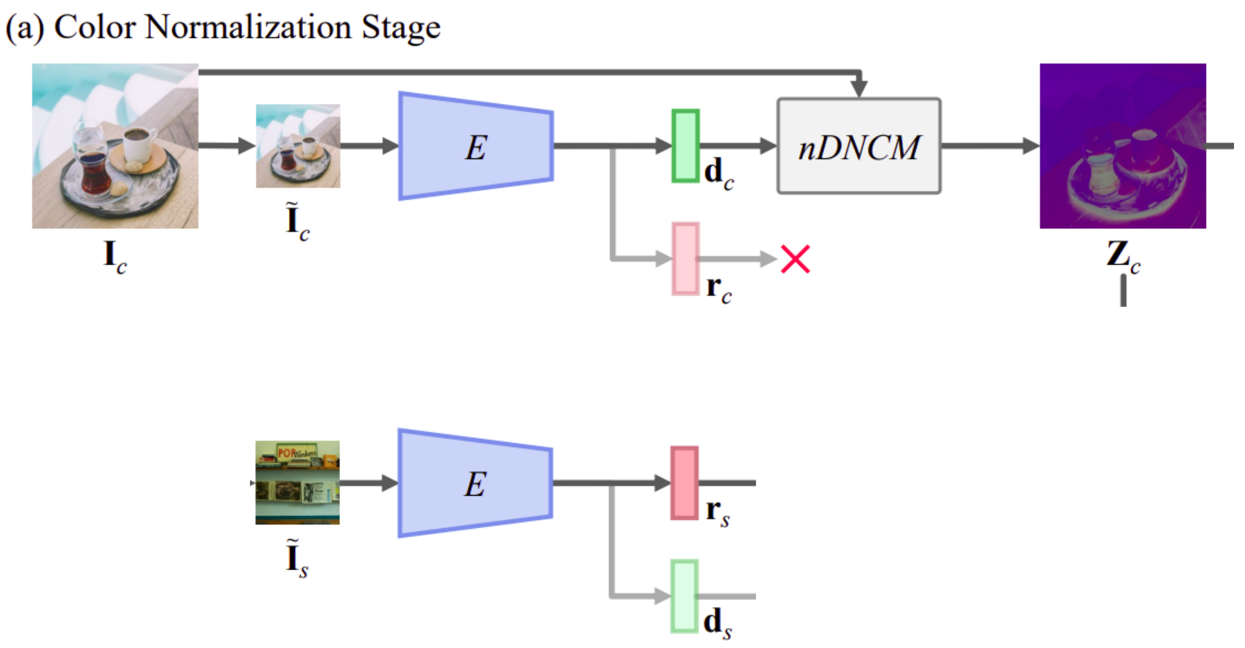


Figure 4. Proposed two-stage pipeline from the paper

Two-Stage Pipeline

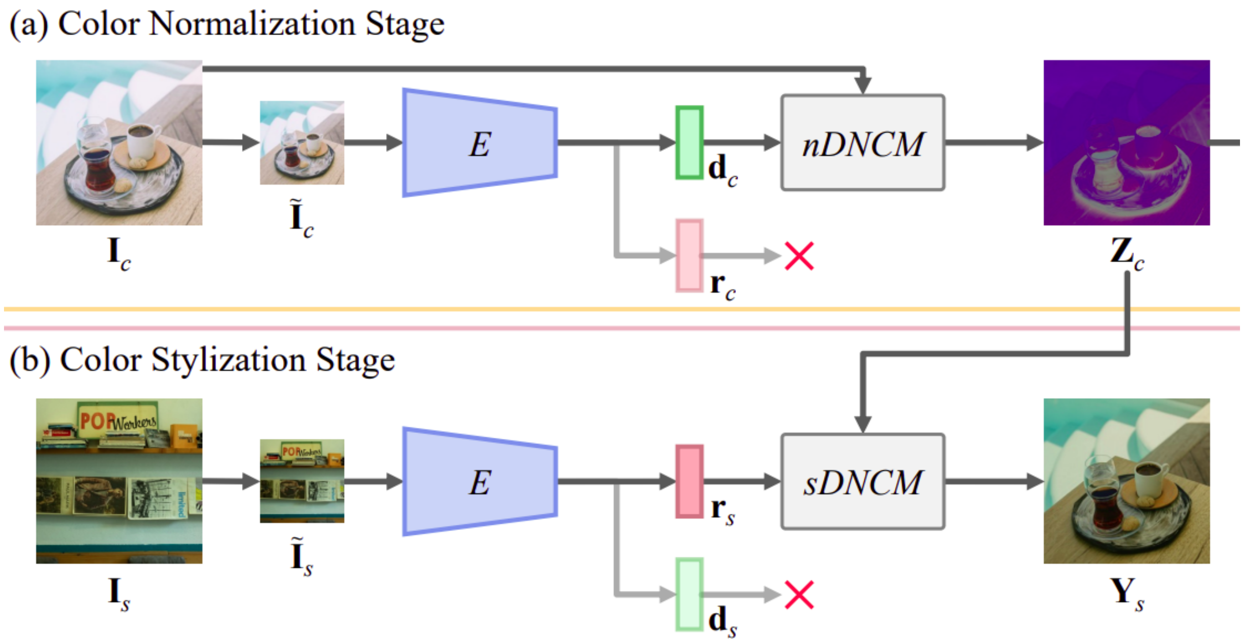


Figure 4. Proposed two-stage pipeline from the paper

Two-Stage Pipeline

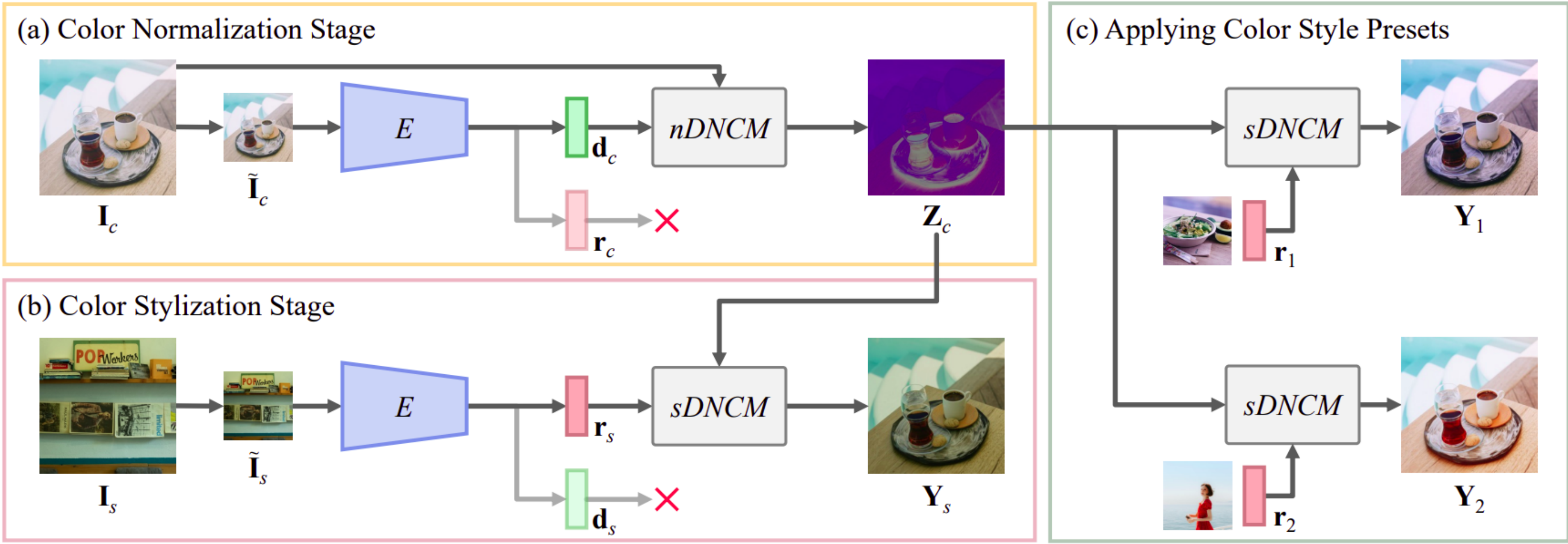


Figure 4. Proposed two-stage pipeline from the paper

Figure 4 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

DNCM Mapping

- What is DNCM?

- Deterministic Neural Color Mapping (DNCM) is proposed in the paper to model arbitrary <u>deterministic</u> color mappings

- It simply uses 3 matrix multiplications.

DNCM Mapping Visualization

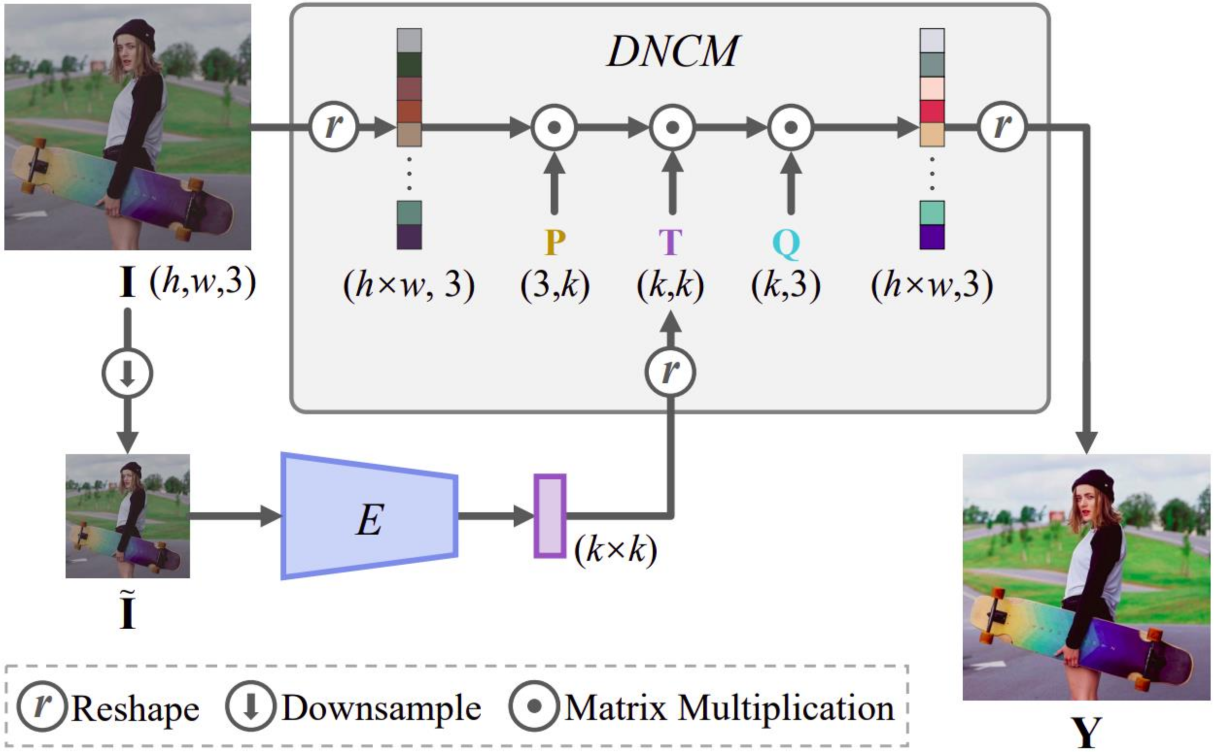


Figure 5. Proposed DNCM Module

Figure 5 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

Full Architecture

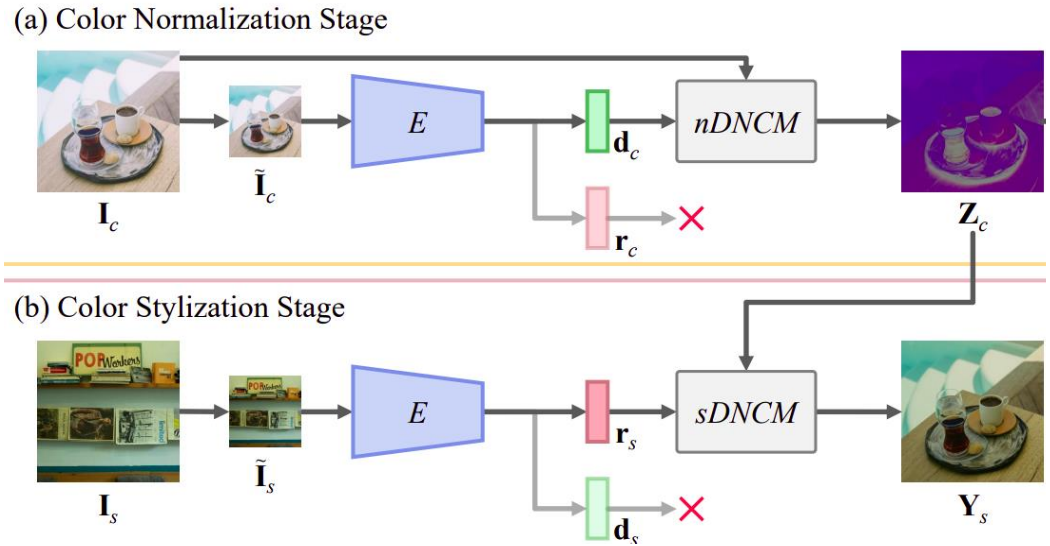


Figure 4. Proposed two-stage pipeline from the paper

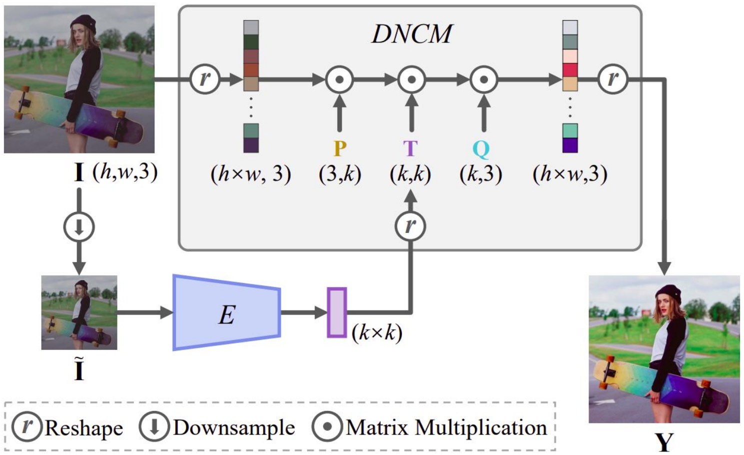


Figure 5. Proposed DNCM Module

Figure 4, 5 (from the presented paper):
https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

Full Architecture

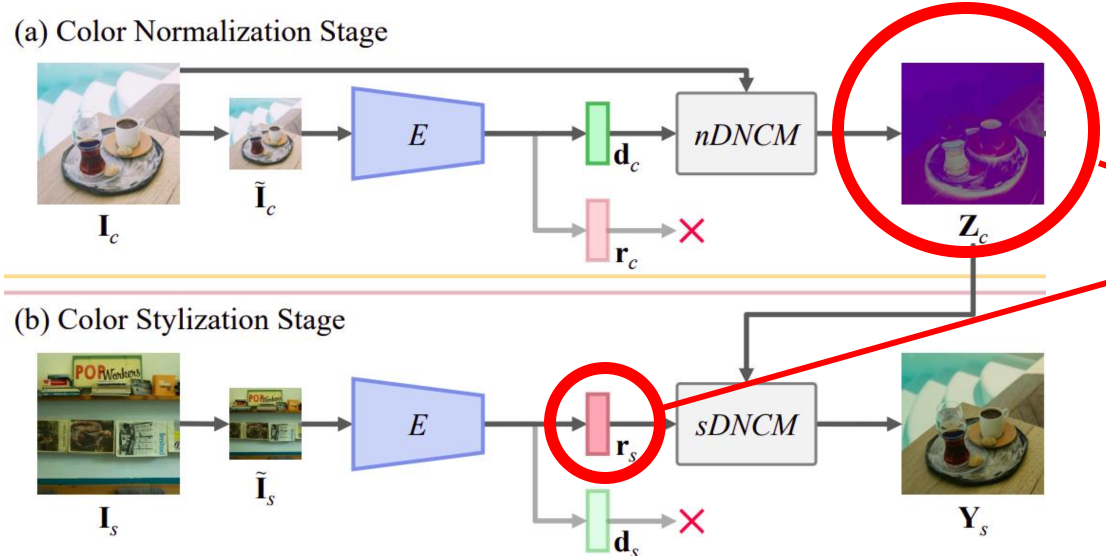


Figure 4. Proposed two-stage pipeline from the paper

Reusable outputs!

Combining any Z_c with any r_s is possible!

Figure 4, 5 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

How to Train the Models?

➔ What parameters do we have to train?

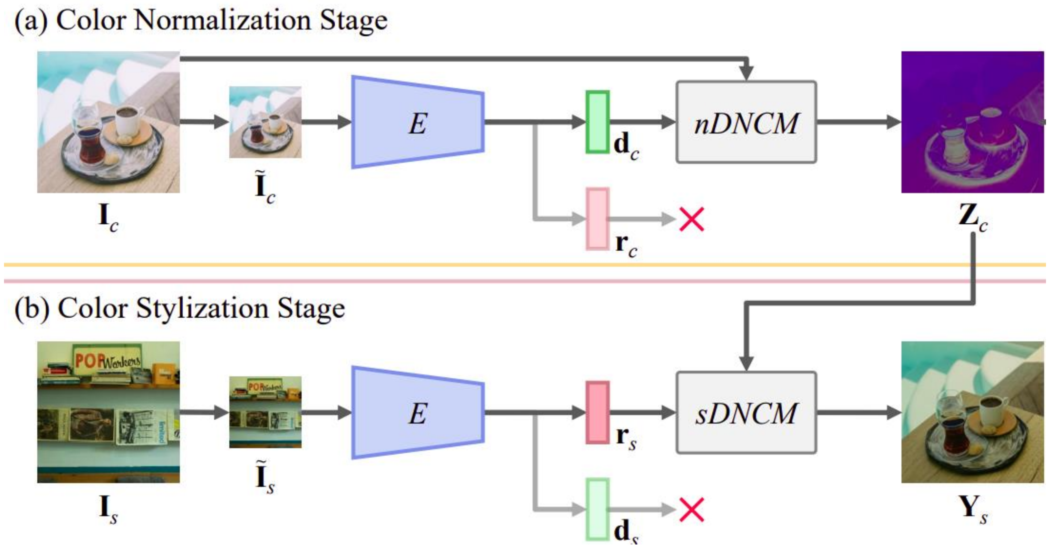


Figure 4. Proposed two-stage pipeline from the paper

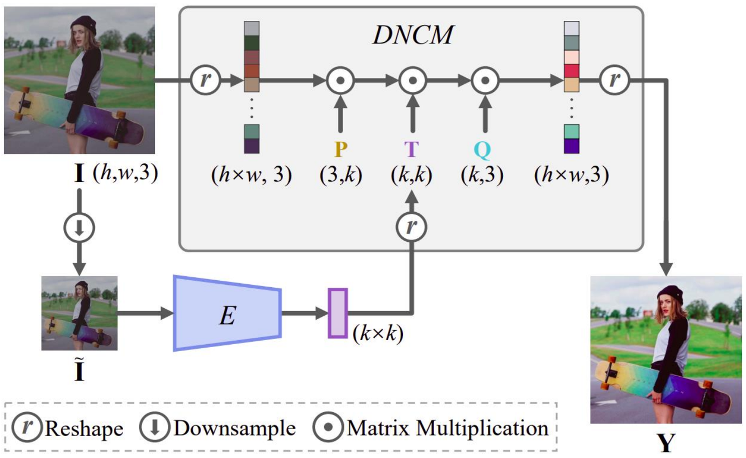
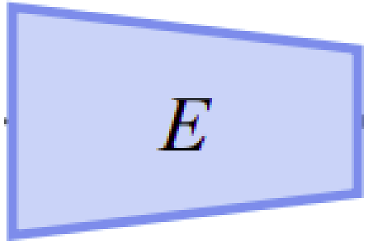


Figure 5. Proposed DNCM Module

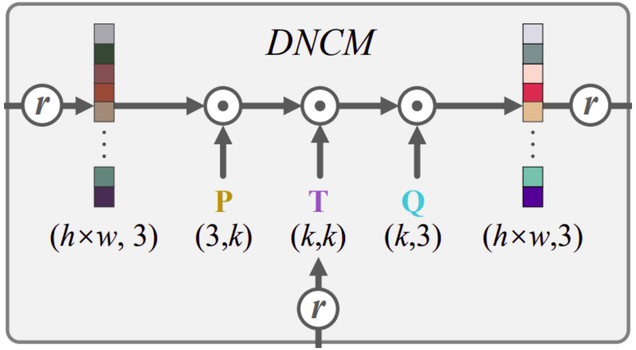
How to Train the Models?

- What parameters do we have to train?

- A unified encoder that gets a resized (256x256) image and outputs d_x and r_x to be used in the DNCM modules.



- Two P (3xk) and two Q (kx3) matrices for sDNCM and nDNCM.



Training Objective

$$\min \ \mathbb{E}_{\mathbf{I}_c,\mathbf{I}_s,\mathbf{G}_s\sim p_{\mathbf{I}}}\Big[\big|\mathbf{G}_s - S_2(S_1(\mathbf{I}_c),\mathbf{I}_s)\big|\Big],$$

I_c is the content image, I_s is the style image.
S_1 is the the color normalization, and S_2 is the stylization.

$$\min \ \mathbb{E}_{\mathbf{I}\sim p_{\mathbf{I}}}\Big[\sum_{i}^{n}\sum_{j}^{n}\big|\mathbf{I}_j - S_2(S_1(\mathbf{I}_i),\mathbf{I}_j)\big|\Big]$$

I_i is the content image, I_j is the style image.
I_i and I_j are obtained from the same image with random filters.
Hence, their "content" should be same.
S_1 is the the color normalization, and S_2 is the stylization.

Equations (from the presented papers supplemental materials):
https://openaccess.thecvf.com/content/CVPR2023/supplemental/Ke_Neural_Preset_for_CVPR_2023_supplemental.pdf

Training Objective

$$\min \ \mathbb{E}_{\mathbf{I}\sim p_{\mathbf{I}}}\left[\sum_{i}^{n}\sum_{j}^{n}\left|\mathbf{I}_j - S_2(S_1(\mathbf{I}_i), \mathbf{I}_j)\right|\right]$$

Given any fixed S_1, the optimal S_2^* should satisfy:

$$\mathbf{I}_j = S_2^*(S_1(\mathbf{I}_j), \mathbf{I}_j) = S_2^*(S_1(\mathbf{I}_i), \mathbf{I}_j).$$

WARNING!

Using end-to-end CNNs (like most of the autoencoders) for S_1 and S_2 leads to a serious problem for this objective function.

S_2 can easily learn to output the identity of its right-side input, ignoring the output of S_1, perfectly optimizing the objective with a trivial solution. So, $S_2(X, I_j) = I_j$ can be obtained with CNNs, which is not we want.

Equations (from the presented papers supplemental materials): https://openaccess.thecvf.com/content/CVPR2023/supplemental/Ke_Neural_Preset_for_CVPR_2023_supplemental.pdf

Self-Supervised Learning

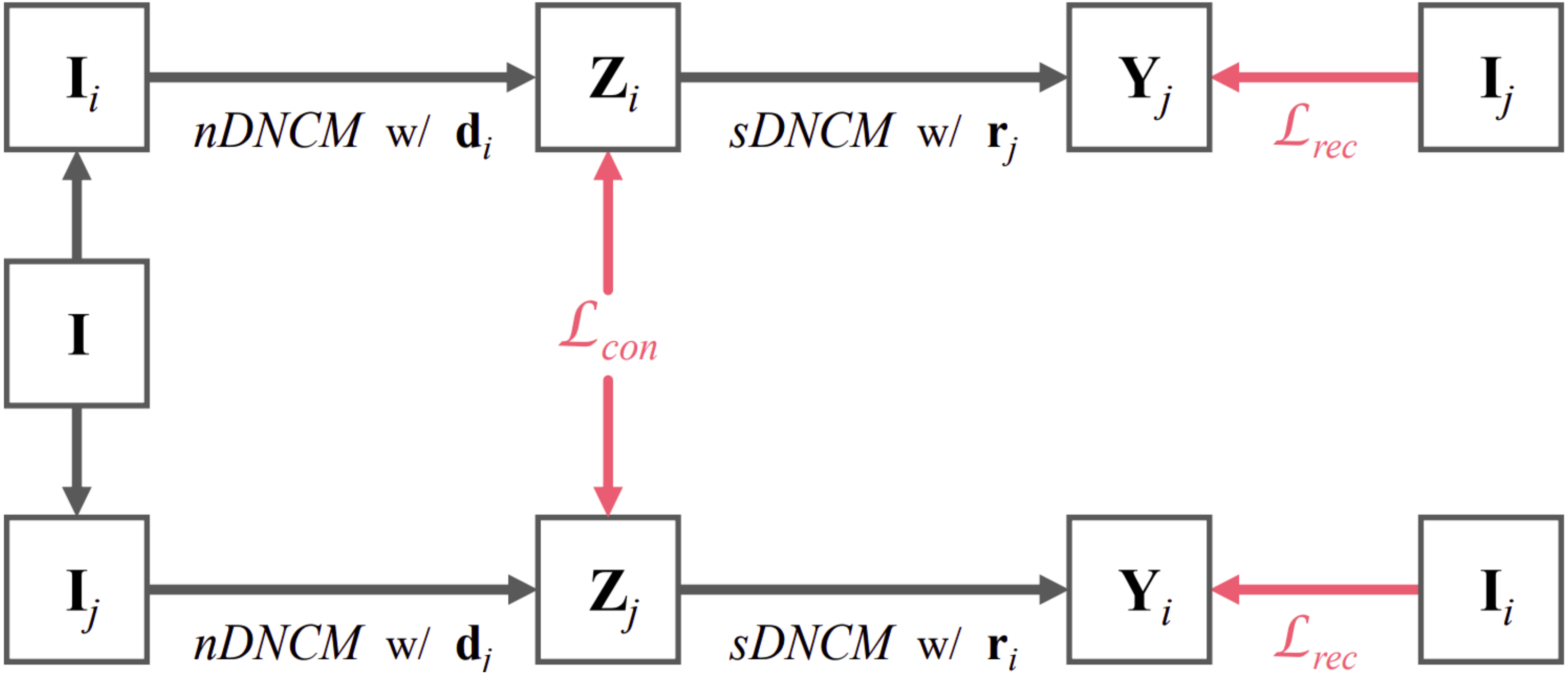


Figure 6. Self-Supervised Learning approach.

Figure 6 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

Self-Supervised Learning

- Obtain I_i and I_j from the same I by applying perturbations (random image filters).

- Get the d_i/r_i and d_j/r_j from the encoder outputs. Apply d_i and d_j to I_i and I_j , respectively, with nDNCM to get normalized color space outputs Z_i and Z_j.

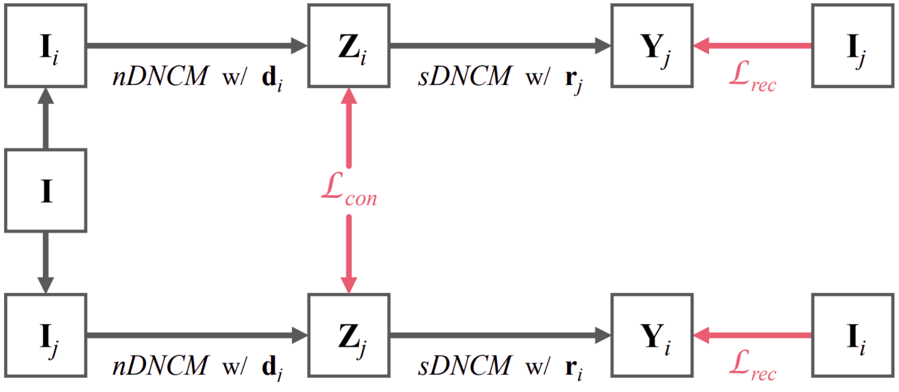


Figure 6. Self-Supervised Learning approach.

- Apply r_i and r_j to Z_j and Z_i, respectively, to get stylized images.

Training Objectives

We want Z_i and Z_j to be exactly the same.

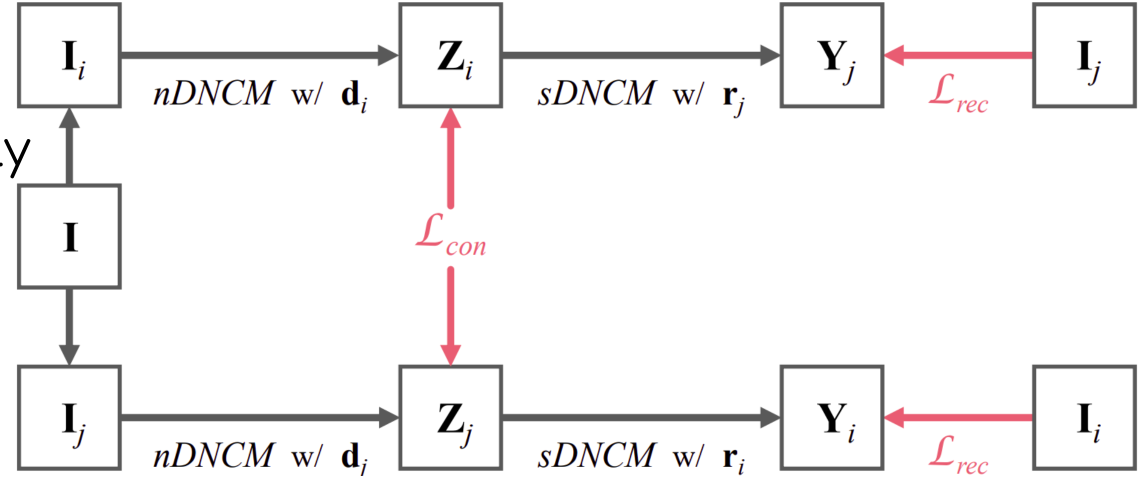


Figure 6. Self-Supervised Learning approach.

We want Y_i and I_j to be exactly the same.

We want Y_j and I_i to be exactly the sam e

$$\mathcal{L}_{con} = ||\mathbf{Z}_i - \mathbf{Z}_j||_2 = ||nDNCM(\mathbf{I}_i, \mathbf{d}_i) - nDNCM(\mathbf{I}_j, \mathbf{d}_j)||_2.$$

$$\mathcal{L}_{rec} = ||\mathbf{Y}_i - \mathbf{I}_i||_1 + ||\mathbf{Y}_j - \mathbf{I}_j||_1.$$

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{con},$$

Figure 6 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

METU

4- Experiments

Datasets and Implementation

→ MS COCO dataset (*) for training.

→ 5000 LUTs, and input color perturbations (**) for getting I_i and I_j.

→ EfficientNet-B0 is used with 256x256 input size as the encoder.

→ K is picked as 16.

→ Trained with Adam for 32 epochs.

* *Microsoft COCO: Common Objects in Context*
https://arxiv.org/pdf/1405.0312

** *Harmonizer: Learning to Perform White-Box Image and Video Harmonization*
https://arxiv.org/pdf/2207.01322

*** *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*
https://arxiv.org/pdf/1905.11946

Quantitive Evaluation

→ They observed prior metrics does not measure quality well.

→ For style similarity, they trained a discriminator on a labeled dataset (with 700+ style classes) to predict the style similarity.

→ For content similarity, they compute SSIM on image edges. Prior works used HED (*), but since it is often incorrect and predict only rough edges, they replaced it with LDC (**).

* Holistically-Nested Edge Detection
https://openaccess.thecvf.com/content_iccv_2015/papers/Xie_Holistically-Nested_Edge_Detection_ICCV_2015_paper.pdf

METU

** LDC: Lightweight Dense CNN for Edge Detection
https://www.researchgate.net/publication/361823660_LDC_Lightweight_Dense_CNN_for_Edge_Detection

Similarity Comparison

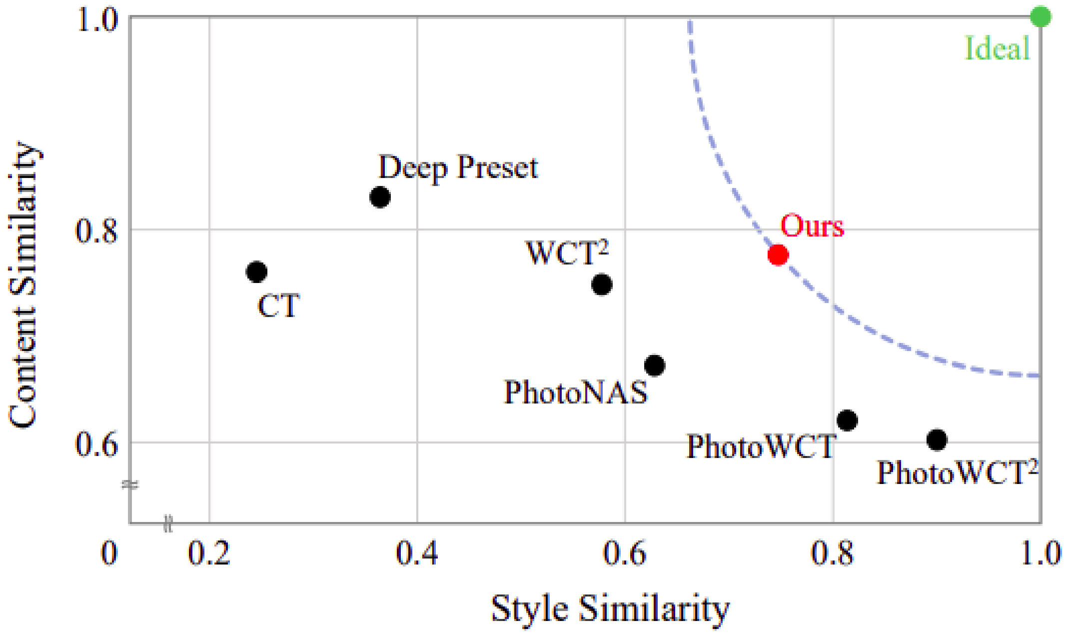


Figure 7. Similarity comparisons with prior methods

Figure 7 (from the presented paper):
https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

METU

Similarity Comparison

Method	CT [43]	PhotoWCT [34]	WCT^2 [54]	PhotoNAS [1]	$PhotoWCT^2$ [6]	Deep Preset [19]	Ours
Average Ranking ↓	4.97	5.75	2.67	3.39	4.11	5.30	**1.81**

Figure 8. Average Human Ranking (subjective) Comparisons

Figure 9. Qualitative Comparisons

Inference Time Comparison

Method	GPU Inference Time ↓ / Memory ↓				Model Size ↓
	FHD (1920 × 1080)	2K (2560 × 1440)	4K (3840 × 2160)	8K (7680 × 4320)	Number of Parameters
PhotoWCT [34]	0.599 s / 10.00 GB	1.002 s / 16.41 GB	OOM	OOM	8.35 M
WCT^2 [54]	0.557 s / 18.75 GB	OOM	OOM	OOM	10.12 M
PhotoNAS [1]	0.580 s / 15.60 GB	0.988 s / 23.87 GB	OOM	OOM	40.24 M
Deep Preset [19]	0.344 s / 8.81 GB	0.459 s / 13.21 GB	1.128 s / 22.68 GB	OOM	267.77 M
$PhotoWCT^2$ [6]	0.291 s / 14.09 GB	0.447 s / 19.75 GB	1.036 s / 23.79 GB	OOM	7.05 M
Ours	**0.013 s / 1.96 GB**	**0.016 s / 1.96 GB**	**0.019 s / 1.96 GB**	**0.061 s / 1.96 GB**	**5.15 M**

Figure 10. Inference Time, GPU memory and Model Size comparisons with RTX3090 GPU.

Ablation Studies

k	2	4	8	<u>16</u>	32
Style Similarity ↑	0.128	0.510	0.636	<u>0.746</u>	**0.769**
Content Similarity ↑	0.765	**0.823**	0.781	<u>0.771</u>	0.764

Figure 11. Effect of the "k" hyper-parameter on the similarity values.

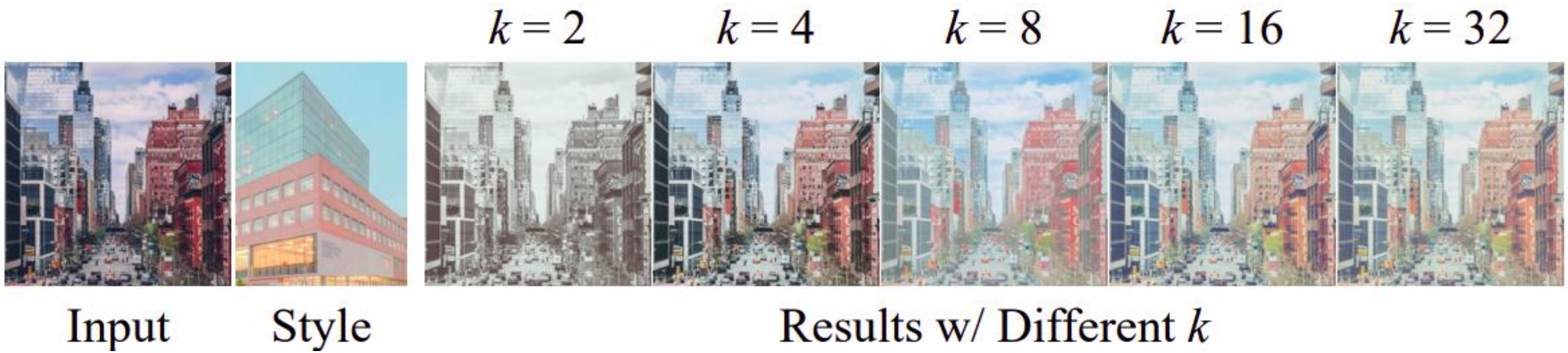


Figure 12. Effect of the "k" hyper-parameter visualized.

Ablation Studies

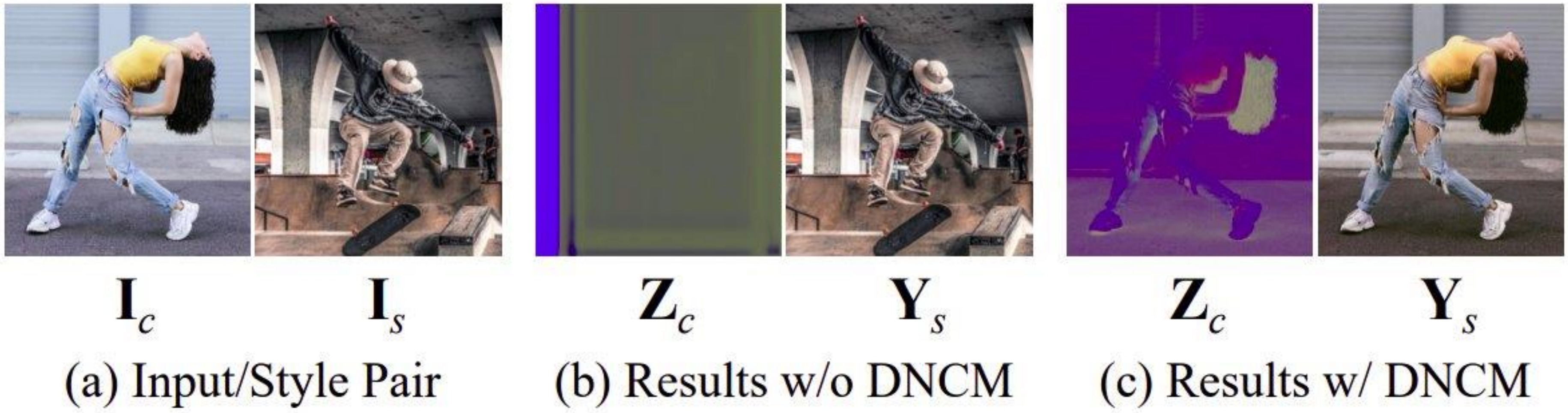


Figure 13. Effect of not using DNCM. Demonstration of the tendency to escaping to a trivial solution.

Figure 13 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

Ablation Studies

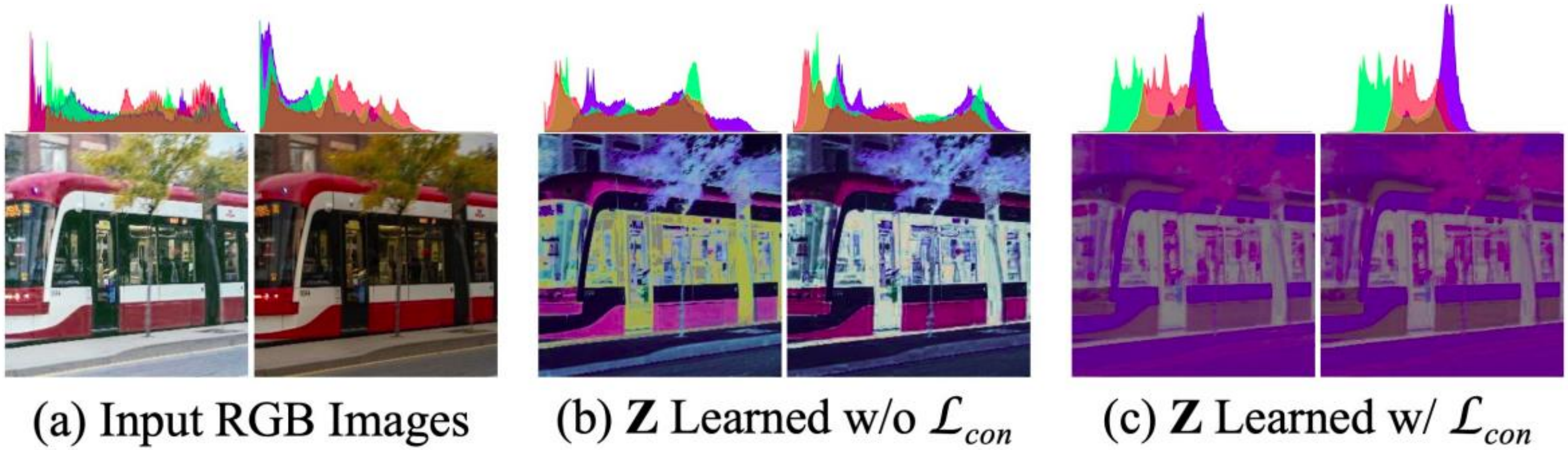


Figure 14. Effect of not using consistency loss

5- Main Contribution and Pros/cons

Main Contribution

➔ DNCM (Deterministic Neural Color Mapping) to avoid artifacts.

➔ Two-stage approach, enables caching and provides high speed.

➔ Self-supervised learning approach.

➔ Custom/improved and better quantitative similarity metrics.

Advantages

- The approach can scale efficiently with high resolution inputs.

- DNCM and two-stage approach significantly reduced the computational expense (Low GPU memory usage and approximately 28x speed up).

- Most of the major artifacts avoided with DNCM. It is crucial for keeping the delicate content such as text readability.

- Without fine-tuning, the model can be used in other tasks such as low-light enhancement.

Disadvantages

→ The approach is observed to increase JPEG artifacts.

Figure 15. Demonstration of increased JPEG artifacts

→ If some content colors does not exist on the style image, result may be unwanted.

Figure 16. Unsatisfactory outputs

→ Local-adaptive color style transfer is not supported.

Figure 17. Demonstration of the lack of local-adaptive mapping

Figure 15, 16, 17 (from the presented paper):
https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

6- Conclusion

Conclusion

→ Using a two-stage approach may tremendously speed up the inference time when switching styles.

→ The self-supervised approach can be utilized with DNCM approach, which also avoids some distortions.

→ No need for deep and heavy architectures for color style transfer.

Input Image (2K Resolution) Reference Style Image PhotoNAS (0.988s) PhotoWCT^2 (0.447s) Ours (0.016s)

Figure 18. Output comparison of the proposed method including the inference times

Figure 18 (from the presented paper): https://openaccess.thecvf.com/content/CVPR2023/papers/Ke_Neural_Preset_for_Color_Style_Transfer_CVPR_2023_paper.pdf

References (1 of 2)

Ke, Z., Liu, Y., Zhu, L., Zhao, N., & Lau, R. W. H. (2023, March 23). Neural Preset for Color Style Transfer. arXiv.org. https://arxiv.org/abs/2303.13511

Ke, Z., Sun, C., Zhu, L., Xu, K., & Lau, R. W. H. (2022, July 4). Harmonizer: Learning to Perform White-Box Image and Video Harmonization. arXiv.org. https://arxiv.org/abs/2207.01322

Lin, T. Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Dollár, P. (2014, May 1). Microsoft COCO: Common Objects in Context. arXiv.org. https://arxiv.org/abs/1405.0312

References (2 of 2)

Soria, X. G., Pomboza-Junez, G., & Sappa, N. D. (2022, January 1). LDC: Lightweight Dense CNN for Edge Detection. IEEE Access. https://doi.org/10.1109/access.2022.3186344

Tan, M., & Le, Q. (2019, May 28). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv.org. https://arxiv.org/abs/1905.11946

Xie, S., & Tu, Z. (2015, December 1). Holistically-Nested Edge Detection. https://doi.org/10.1109/iccv.2015.164

Attribution

Thank you for listening

Q & A