# PhotoWCT$^2$: Compact Autoencoder for Photorealistic Style Transfer Resulting from Blockwise Training and Skip Connections of High-Frequency Residuals

Tai-Yin Chiu
The University of Texas at Austin

Danna Gurari
University of Colorado Boulder

## Supplementary Materials

This document supplements Sections 3.1, 3.2, and 4.2 of the main paper. It includes the following:

- Details of ZCA transformation (supplements **Section 3.1**).
- Description of the bug in the post-processing code of PhotoWCT, our modification, and why fixing the bug greatly improves the speed of the post-processing used in PhotoWCT (supplements **Sections 3.1**).
- Additional details about our blockwise model architecture and our training strategy (supplements **Section 3.2**).
- Results of blockwise training in the reversed order (supplements **Section 3.2.2**).
- PhotoWCT with three cascaded autoencoders and its redesign (supplements **Section 3.2.2**).
- Additional details about our regularized style loss (supplements **Section 4.2**).
- Qualitative stylization results (supplements **Section 4.2**).

## Details of ZCA transformation

ZCA transformation [5] is the key to the realization of style transfer in WCT$^2$, PhotoWCT, PhotoNAS, and our PhotoWCT$^2$ by making the gram matrix of a content feature match that of a style feature. It takes as input a content feature of shape $H_c \times W_c \times C$ and a style feature of shape $H_s \times W_s \times C$ extracted from, say, the *reluN_1* layer of VGGNet, where $H_c$ ($H_s$) and $W_c$ ($W_s$) are the height and width of the content (style) feature, while $C$ is the channel length. We first reshape the content and style features to the shapes $C \times H_cW_c$ and $C \times H_sW_s$ and denote the reshaped features $\mathbf{F}_c$ and $\mathbf{F}_s$, respectively. Then we apply eigen-decomposition to the covariances $\frac{1}{H_cW_c}\bar{\mathbf{F}}_c\bar{\mathbf{F}}_c^{\mathrm{T}}$ and $\frac{1}{H_sW_s}\bar{\mathbf{F}}_s\bar{\mathbf{F}}_s^{\mathrm{T}}$:

$$\begin{aligned}\frac{1}{H_cW_c}\bar{\mathbf{F}}_c\bar{\mathbf{F}}_c^{\mathrm{T}} &= \mathbf{E}_c\mathbf{\Lambda}_c\mathbf{E}_c^{\mathrm{T}} \\ \frac{1}{H_sW_s}\bar{\mathbf{F}}_s\bar{\mathbf{F}}_s^{\mathrm{T}} &= \mathbf{E}_s\mathbf{\Lambda}_s\mathbf{E}_s^{\mathrm{T}},\end{aligned} \tag{1}$$

where $\bar{\mathbf{F}}_c$ and $\bar{\mathbf{F}}_s$ are the centralized content and style features:

$$\begin{aligned}\bar{\mathbf{F}}_c &= \mathbf{F}_c - \mathrm{mean}(\mathbf{F}_c) = \mathbf{F}_c - \sum_{i=1}^{H_cW_c}[\mathbf{F}_c]_{:,i} \\ \bar{\mathbf{F}}_s &= \mathbf{F}_s - \mathrm{mean}(\mathbf{F}_s) = \mathbf{F}_s - \sum_{i=1}^{H_sW_s}[\mathbf{F}_s]_{:,i}.\end{aligned} \tag{2}$$

By transforming the content feature $\mathbf{F}_c$ as in Equation 3, the gram matrix of the transformed feature $\mathbf{F}_{cs}$ will match that of the style feature.

$$\mathbf{F}_{cs} = (\mathbf{E}_s\mathbf{\Lambda}_s^{\frac{1}{2}}\mathbf{E}_s^{\mathrm{T}})(\mathbf{E}_c\mathbf{\Lambda}_c^{-\frac{1}{2}}\mathbf{E}_c^{\mathrm{T}})\bar{\mathbf{F}}_c + \mathrm{mean}(\mathbf{F}_s) \tag{3}$$

It can be shown that $\frac{1}{H_cW_c}\mathbf{F}_{cs}\mathbf{F}_{cs}^{\mathrm{T}}$ is equal to $\frac{1}{H_sW_s}\mathbf{F}_s\mathbf{F}_s^{\mathrm{T}}$ [2].

## Modification of PhotoWCT's post-processing code

The post-processing in the PhotoWCT code includes two parts: guided filtering and image smoothing, with the former code running much faster than the latter. Since smoothing is also one effect of guided filtering, the second post-processing step of image smoothing is not necessary. We suspect the image smoothing function was invoked to overcome a misuse of the `guidedFilter` function provided in the OpenCV package, as described below.

The `guidedFilter` function takes four arguments: a guide image, an image to filter, a filtering window radius, and an $\epsilon$ parameter to prevent overfitting. The PhotoWCT code follows the guided filter paper [4] to use $0.1^2$ as the value of $\epsilon$[1]. However, while the paper assumes image pixel

[1] In practice, values between $0.01^2$ to $0.1^2$ work well.

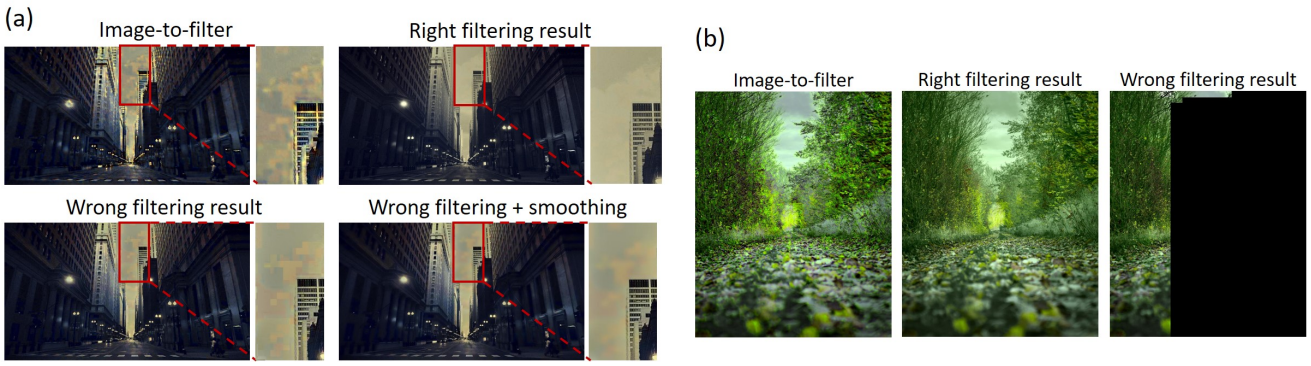


Figure 1: Two examples showing the improvement after fixing the bug in the PhotoWCT code. In (a), we observe that the result from the wrong invocation of the `guidedFilter` function results in a grid artifact that needs to be fixed with additional smoothing. (b) shows a worse case where wrongly invoking the function results in a failure.

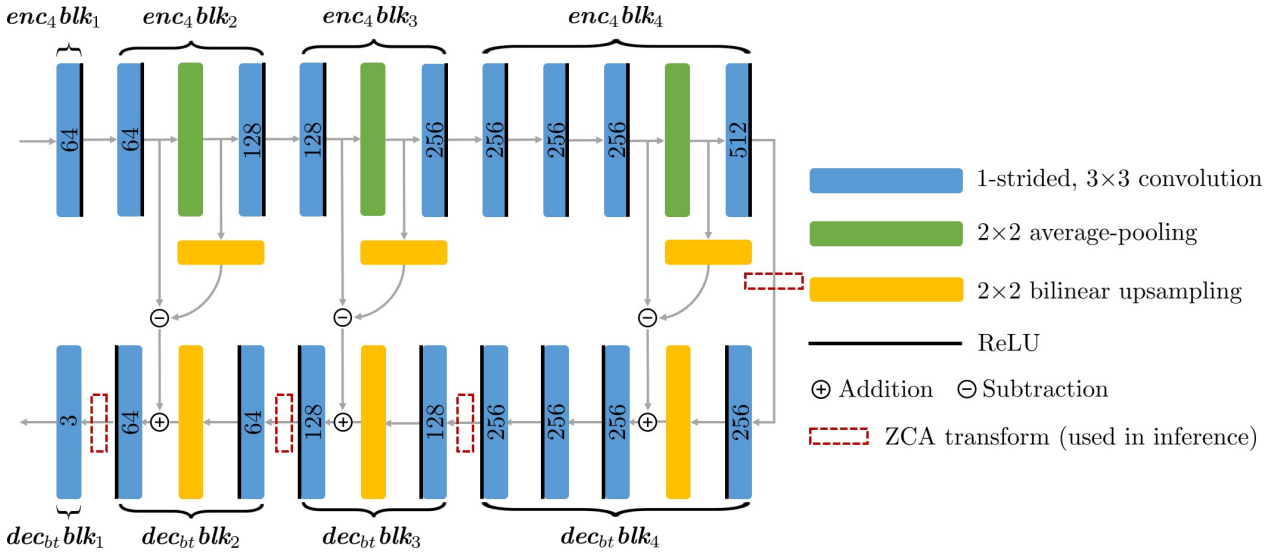


Figure 2: Detailed structure of our model's architecture that gets used during blockwise training. The number in each convolution layer is the number of channels at its output.

values are in the range of 0 and 1, `guidedFilter` takes images with pixel values ranged from 0 to 255 and so the value of $\epsilon$ should be scaled accordingly. In our modification, we set $\epsilon$ to be $(0.02 \times 255)^2$. As expected, with this value, the second post-processing step of image smoothing becomes unnecessary, and the removal of the second post-processing greatly improves the speed reported in previous works.

Figure 1 exemplifies the importance of fixing the code. Before fixing the bug, a filtering result might suffer from a grid artifact, where a region that is supposed to be smooth contains groups of pixels that are not well blended. This artifact is exemplified in Figure 1(a). To remove this artifact, the second post-processing step of smoothing is needed. Even still, this extra filtering step can fail to conceal the issues introduced by the bug, as exemplified in Figure 1(b).

We set the filtering window radius to be 100 for the speed test in Section 4.1. We set it to 50 for evaluation of the image quality and stylization strength in Section 4.2, since the test images used in the speed test are (much) larger than the images from the DPST dataset used in Section 4.2.

## Model-to-train and training details

A detailed diagram showing the structure of our model that gets used during blockwise training is shown in Figure 2, expanding upon its illustration in Figure 1b:4 and Figure 3a of the main paper. For training, we use the MS-COCO dataset [7]. Each image in the dataset is resized to 512×512 and randomly cropped to 256×256. We use a

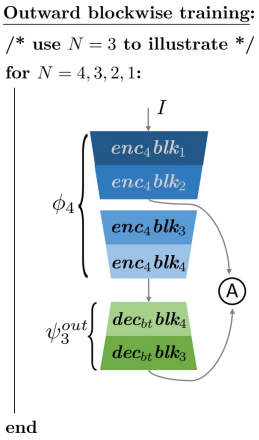


Figure 3: Outward blockwise training

batch size of eight images. We use the Adam optimizer with learning rate $1 \times 10^{-4}$ and without weight decay. In blockwise training, each $dec_{bt}blk_N$ block is trained for 20 epoch.

## Results of blockwise training in the reversed order

The blockwise training in the paper trains the decoder blocks $dec_{bt}blk_N$'s in the order from $N = 1$ to $N = 4$. That is, the training is inward from the outermost block $dec_{bt}blk_1$ to the innermost $dec_{bt}blk_4$. Here we show the results of outward training from the innermost to the outermost block as exemplified in Figure 3.

In outward blockwise training, the decoder blocks $dec_{bt}blk_N$'s are trained in the order from $N = 4$ to $N = 1$ by minimizing the function inversion loss $\mathcal{L}_N^{out}$:

$$\mathcal{L}_N^{out}(I) = ||\phi_{N-1}(I) - \psi_N^{out}(\phi_4(I))||_2^2 \qquad (4)$$

where $\phi_N$ and $\psi_N^{out}$ are the functions of the series $\{enc_4blk_1, \ldots, enc_4blk_N\}$ and $\{dec_{bt}blk_4, \ldots, dec_{bt}blk_N\}$, respectively. When training a decoder block, the previously trained blocks and the encoder are fixed. Note that when $N = 1$, $\mathcal{L}_1^{out}$ minimizes the reconstruction loss. In this document, we denote our models trained outwardly and inwardly as $\text{AEC}_{bt}^{out}$ and $\text{AEC}_{bt}^{in}$ (PhotoWCT$^2$ (BT)), respectively.

Next we compare the inward and the outward blockwise trainings using the same metrics in the main paper. Table 1(c) shows the mean metric scores for image quality and stylization strength across 100 stylized images resulting from each method. Note that the values of $\bar{\mathcal{L}}_{s,m}$ reported in Table 1 here are different from those reported in the Table 1 in the main paper, since here style losses of $\text{AEC}_{bt}^{out}$, PhotoWCT3, and $\text{AEC}_{bt,3}^{in}$ (see next section) are included in the normalization for the computation of $\bar{\mathcal{L}}_{s,m}$. We observe both $\text{AEC}_{bt}^{in}$ (i.e., PhotoWCT$^2$ (BT)) and $\text{AEC}_{bt}^{out}$ result in image quality comparable to that from the baselines and as strong stylization strength as PhotoWCT. Table 2 shows the loss values averaged across 500 images for feature and image reconstruction in the decoders trained by inward and outward blockwise trainings. We observe the inward training has a slightly better image and feature reconstruction ability than the outward training.

## Redesign of PhotoWCT to use three autoencoders

As discussed in the main paper, our method can generalize for use with different numbers of autoencoders. We illustrate this here by removing the fourth autoencoder $\text{AEC}_4$ in the PhotoWCT cascade. We denote the resulting 3-autoencoder PhotoWCT as PhotoWCT3.

Quantitative results comparing PhotoWCT and PhotoWCT3 are shown in Table 1. As expected, we observe a slight drop in stylization strength from PhotoWCT (-0.57) to PhotoWCT3 (-0.47), while PhotoWCT3 uses fewer parameters and runs at faster speeds.

We next apply our inward blockwise training method to redesign PhotoWCT3 into a single autoencoder, which is denoted $\text{AEC}_{bt,3}^{in}$ here. Results are shown in Table 1. Reinforcing our findings in the main paper, we observe that $\text{AEC}_{bt,3}^{in}$ achieves comparable stylization strength while yielding faster speeds than PhotoWCT3. For example, it takes 0.2 fewer seconds for QHD rendering for $\text{AEC}_{bt,3}^{in}$ than PhotoWCT3. When comparing $\text{AEC}_{bt,3}^{in}$ to PhotoWCT, while there is a more noticeable drop in stylization strength from the original PhotoWCT method (i.e., four autoencoders) to $\text{AEC}_{bt,3}^{in}$, $\text{AEC}_{bt,3}^{in}$ runs much faster than PhotoWCT. Take QHD rendering for instance. $\text{AEC}_{bt,3}^{in}$ spends only half the rendering time of PhotoWCT (0.64s vs. 1.24s).

Note that when redesigning PhotoWCT3 into a single autoencoder, we use inward rather than outward blockwise training. The reason is that inward blockwise training not only results in better image and feature reconstruction ability, but while we redesign PhotoWCT using inward blockwise training as in the main paper, we already accomplish the redesign of PhotoWCT3. That is, inward blockwise training trains the decoder blocks $dec_{bt}blk_1$, $dec_{bt}blk_2$, and $dec_{bt}blk_3$ in $\text{AEC}_{bt,3}^{in}$ first and then trains the final block $dec_{bt}blk_4$ used in $\text{AEC}_{bt}^{in}$. In contrast, due to the reversed training order, redesigning PhotoWCT3 with outward blockwise training requires re-training from scratch.

## Regularized style loss

We use the regularized style loss $\mathcal{L}_s$ in equation 5, which was introduced in DPST [8], for the evaluation of stylization strength. Intuitively, style can be thought of as a composition of ingredients such as color, lightness, and artistic effects, including image pattern and painting styles (oil

| Model | (a) Size | | (b) Speed performance | | | | | (c) Image quality & Stylization strength | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | # par | # layer | 1024×512 | HD 1280×720 | FHD 1920×1080 | QHD 2560×1440 | 4K 3840×2160 | BRIS (27.4) | NIQE (3.19) | NIMA-q (5.11) | NIMA-a (5.27) | $\bar{\mathcal{L}}_{s,m}$ |
| PhNAS | 40.24M | 35 | 0.23 | OOM | OOM | OOM | OOM | 33.0 | 3.24 | 4.75 | 4.92 | 1.42 |
| WCT$^2$ | 10.12M | **24** | 0.30 | 0.43 | 0.80 | OOM | OOM | **30.8** | 3.07 | **4.91** | 5.01 | 1.15 |
| PhWCT | 8.35M | 48 | 0.21+0.03 | 0.32+0.06 | 0.61+0.14 | 1.01+0.23 | OOM | 31.8 | **2.90** | 4.88 | 5.06 | **-0.57** |
| AEC$_{e2e}$ | **7.05M** | **24** | **0.18+0.03** | **0.24+0.06** | **0.39+0.14** | **0.59+0.23** | **1.22+0.54** | 31.7 | 2.91 | 4.90 | **5.10** | -0.52 |
| AEC$^{in}_{bt}$ | **7.05M** | **24** | **0.18+0.03** | **0.24+0.06** | **0.39+0.14** | **0.59+0.23** | **1.22+0.54** | 31.6 | **2.90** | 4.90 | **5.10** | -0.55 |
| AEC$^{out}_{bt}$ | **7.05M** | **24** | **0.18+0.03** | **0.24+0.06** | **0.39+0.14** | **0.59+0.23** | **1.22+0.54** | 31.8 | 2.92 | 4.88 | 5.06 | -0.54 |
| PhWCT3 | 1.34M | 24 | 0.13+0.03 | 0.19+0.06 | 0.36+0.14 | 0.60+0.23 | OOM | 32.1 | 2.98 | **4.88** | 5.04 | **-0.47** |
| AEC$^{in}_{bt,3}$ | **1.15M** | **14** | **0.09+0.03** | **0.13+0.06** | **0.24+0.14** | **0.40+0.23** | **0.77+0.54** | **31.6** | **2.92** | **4.88** | **5.06** | -0.42 |

Table 1: Extended Table 1 from the main paper. (Top 6 rows) Performance comparison between three previous methods PhotoNAS, WCT$^2$, and PhotoWCT and our models AEC$_{e2e}$, AEC$^{in}_{bt}$ and AEC$^{out}_{bt}$ (PhotoWCT$^2$ resulting from end-to-end training, inward and outward blockwise trainings). (Last 2 rows) Comparison of PhotoWCT3 (PhotoWCT reduced to three cascaded autoencoders) and AEC$^{in}_{bt,3}$ (redesign of PhotoWCT3 using inward blockwise training).

| Strategy | *relu3_1* | *relu2_1* | *relu1_1* | image |
|---|---|---|---|---|
| Outward | 0.037 | 0.028 | 0.009 | 0.0010 |
| Inward | **0.035** | **0.021** | **0.008** | **0.0006** |

Table 2: Loss values for feature and image reconstruction in the decoders resulting from inward and outward blockwise trainings. The inward training has a slightly better image and feature reconstruction ability than the outward training.

paintings, watercolor paintings, etc.). These ingredients are captured in Gatys et al.'s [3] formulation of style loss ($\sum \beta_l \mathcal{L}_{s,l}$). However, since artistic effects result in artifacts, they are undesired ingredients in photorealistic stylization. To avoid these artifacts, a regularization term $\mathcal{L}_{reg}$ is introduced to Gatys's formulation to remove the artistic effects. Mathematically, with $I_o$, $I_c$, $I_s$ being the stylized, content, and style images, the regularized loss is defined as follows:

$$\mathcal{L}_s(I_o; I_c, I_s) = \sum_{l=1}^{5} \beta_l \mathcal{L}_{s,l}(I_o, I_s) + \lambda \mathcal{L}_{reg}(I_o, I_c), \quad (5)$$

$$\mathcal{L}_{s,l}(I_o, I_s) = ||\frac{1}{H_{o,l}W_{o,l}}\mathbf{F}_{o,l}\mathbf{F}_{o,l}^{\mathrm{T}} - \frac{1}{H_{s,l}W_{s,l}}\mathbf{F}_{s,l}\mathbf{F}_{s,l}^{\mathrm{T}}||_2^2, \quad (6)$$

$$\mathcal{L}_{reg}(I_o, I_c) = \sum_{ch\in\{R,G,B\}} vec(I_{o,ch})^{\mathrm{T}} \mathcal{M}(I_{c,ch}) vec(I_{o,ch}). \quad (7)$$

In equation 6, $\mathbf{F}_{o,l}$ and $\mathbf{F}_{s,l}$ are the *relu'l'_1* features of $I_o$ and $I_s$ extracted from VGGNet. $(H_{o,l}, W_{o,l})$ and $(H_{s,l}, W_{s,l})$ are the (height, width) of $\mathbf{F}_{o,l}$ and $\mathbf{F}_{s,l}$. In equation 7, $vec(I_{o,ch})$ is the pixels of $I_o$ in the $ch$ channel vectorized into a column vector. $\mathcal{M}(I_{c,ch})$ is the Matting Laplacian matrix of the $ch$ channel of $I_c$. Following the official implementation of DPST, the weights $\beta_l$'s are $1/5$, while $\lambda$ is $10^2$.

To account for regularized style loss values falling in different ranges for different content-style pairs, we normalize the loss value $\mathcal{L}_{s,m,p}$ resulting from the photorealistic style transfer method $m$ and the content-style pair $p$ as follows ($m \in$ {PhotoNAS, WCT$^2$, PhotoWCT, AEC$_{e2e}$, AEC$^{in}_{bt}$} in the main paper, while $m \in$ {PhotoNAS, WCT$^2$, PhotoWCT, AEC$_{e2e}$, AEC$^{in}_{bt}$, AEC$^{out}_{bt}$, PhotoWCT3, AEC$^{in}_{bt,3}$} in this document):

$$\bar{\mathcal{L}}_{s,m,p} = \frac{\mathcal{L}_{s,m,p} - \mu_p}{\sigma_p} = \frac{\mathcal{L}_{s,m,p} - \frac{1}{|m|}\sum_m \mathcal{L}_{s,m,p}}{\sqrt{\frac{1}{|m-1|}\sum_m (\mathcal{L}_{s,m,p} - \mu_p)^2}}, \quad (8)$$

where $|m|$ is the number of considered methods (i.e., $m$ is five and eight in the main paper and this document, respectively.) As such, $\bar{\mathcal{L}}_{s,m,p}$ is distributed around 0. Moreover, given a content-style pair $p$, the relative order of $\bar{\mathcal{L}}_{s,m,p}$'s is preserved to match that of $\mathcal{L}_{s,m,p}$'s. Note that the reported loss value $\bar{\mathcal{L}}_{s,m}$ for the method $m$ is the mean across 100 normalized style losses $\{\bar{\mathcal{L}}_{s,m,1}, \ldots, \bar{\mathcal{L}}_{s,m,100}\}$ for 100 stylized images resulting from $m$.

## Stylization results

We show the stylization results for the DPST dataset in Figures 4 to 14. Each row in the figures contains the results of photorealistic style transfer methods from a pair of content and style images and the associated segmentation. In particular, a segment in a content image is rendered with the style of the corresponding segment denoted in the same color in the style image. Note that the results from PhotoNAS [1] do not use the segmentation labels, since the

PhotoNAS paper clearly states “the proposed algorithm allows transferring photo styles without any assist of region masks acquired by segmenting content and style inputs” and the official code does not support this feature, either.

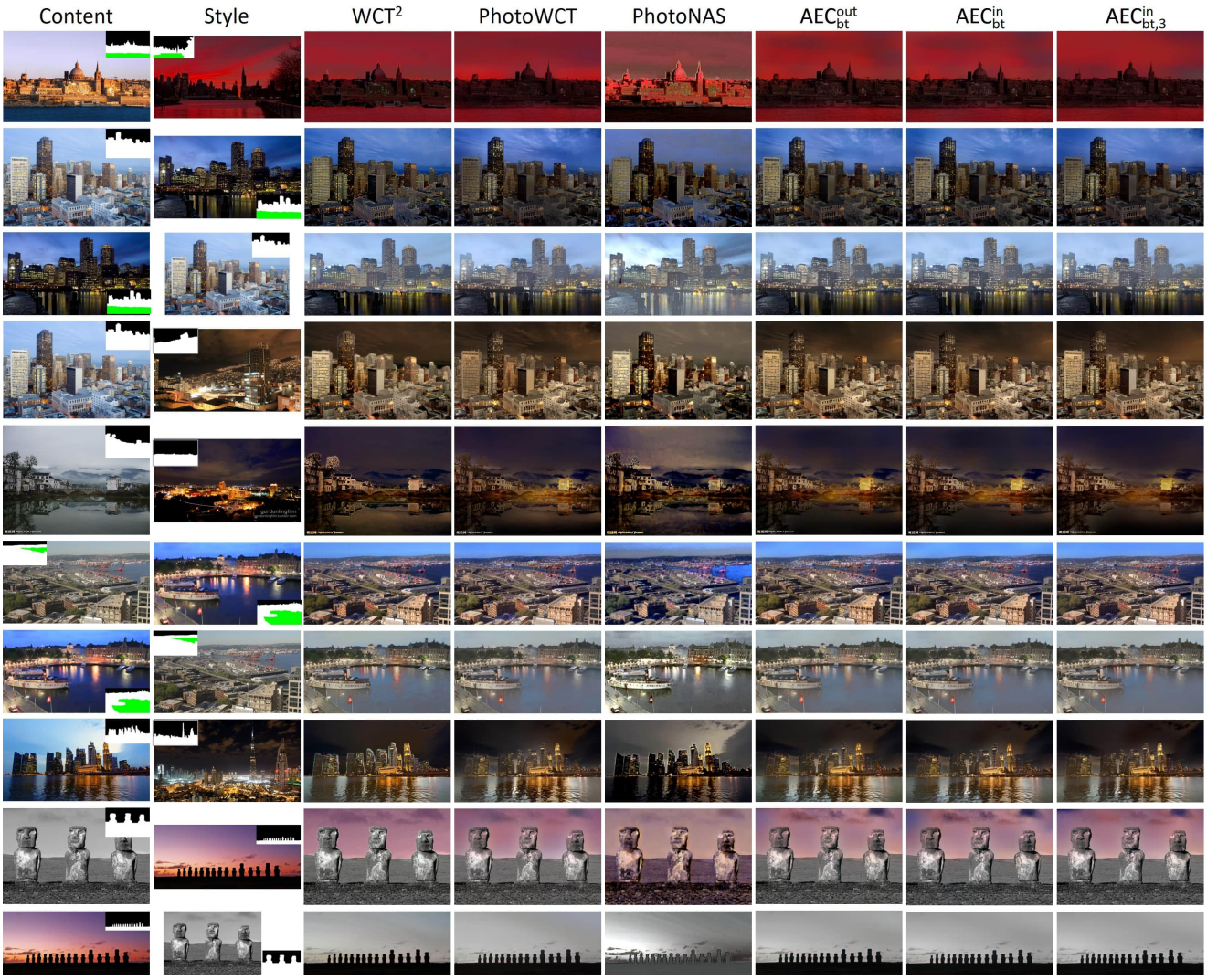


Figure 4: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 1/11)

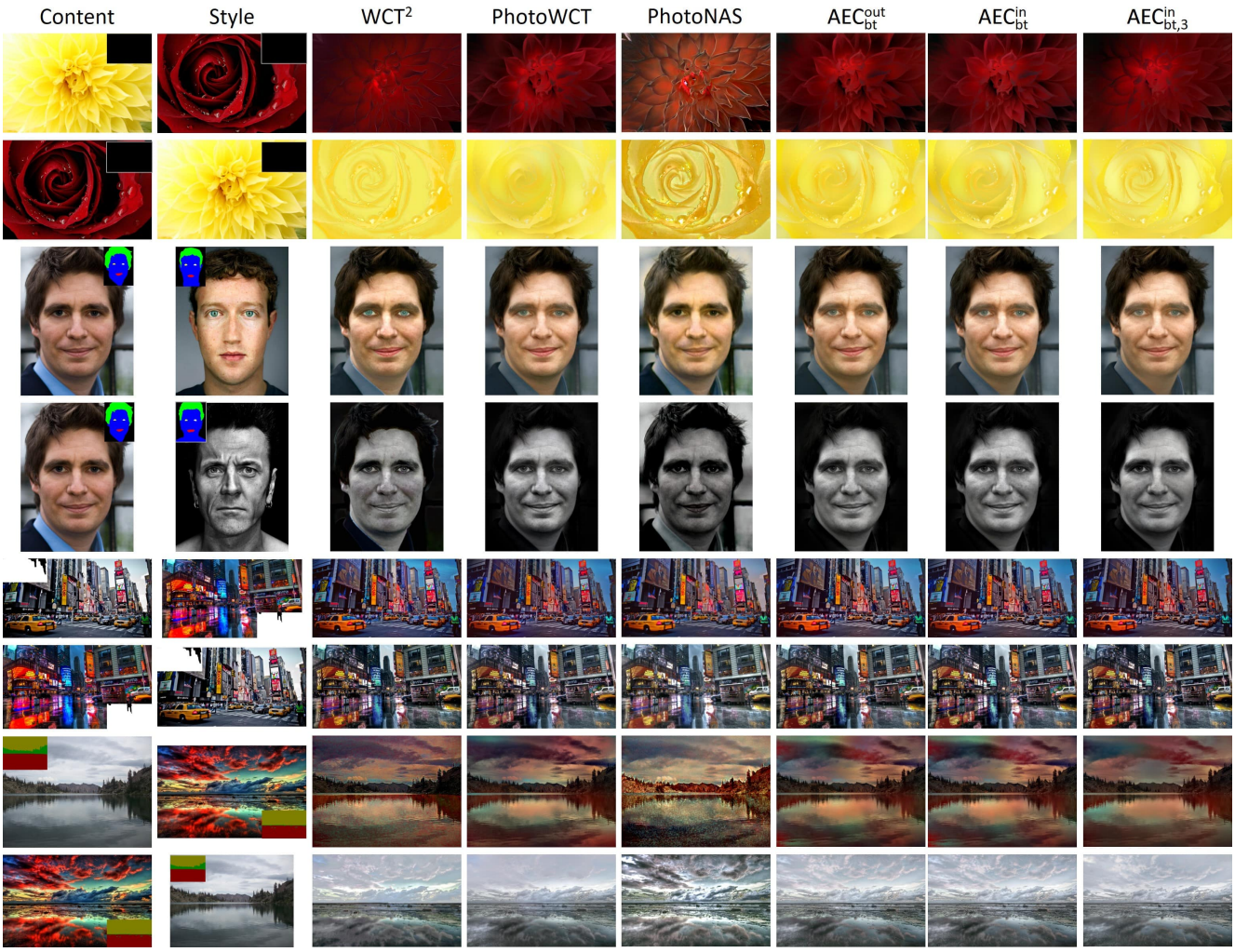


Figure 5: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 2/11)

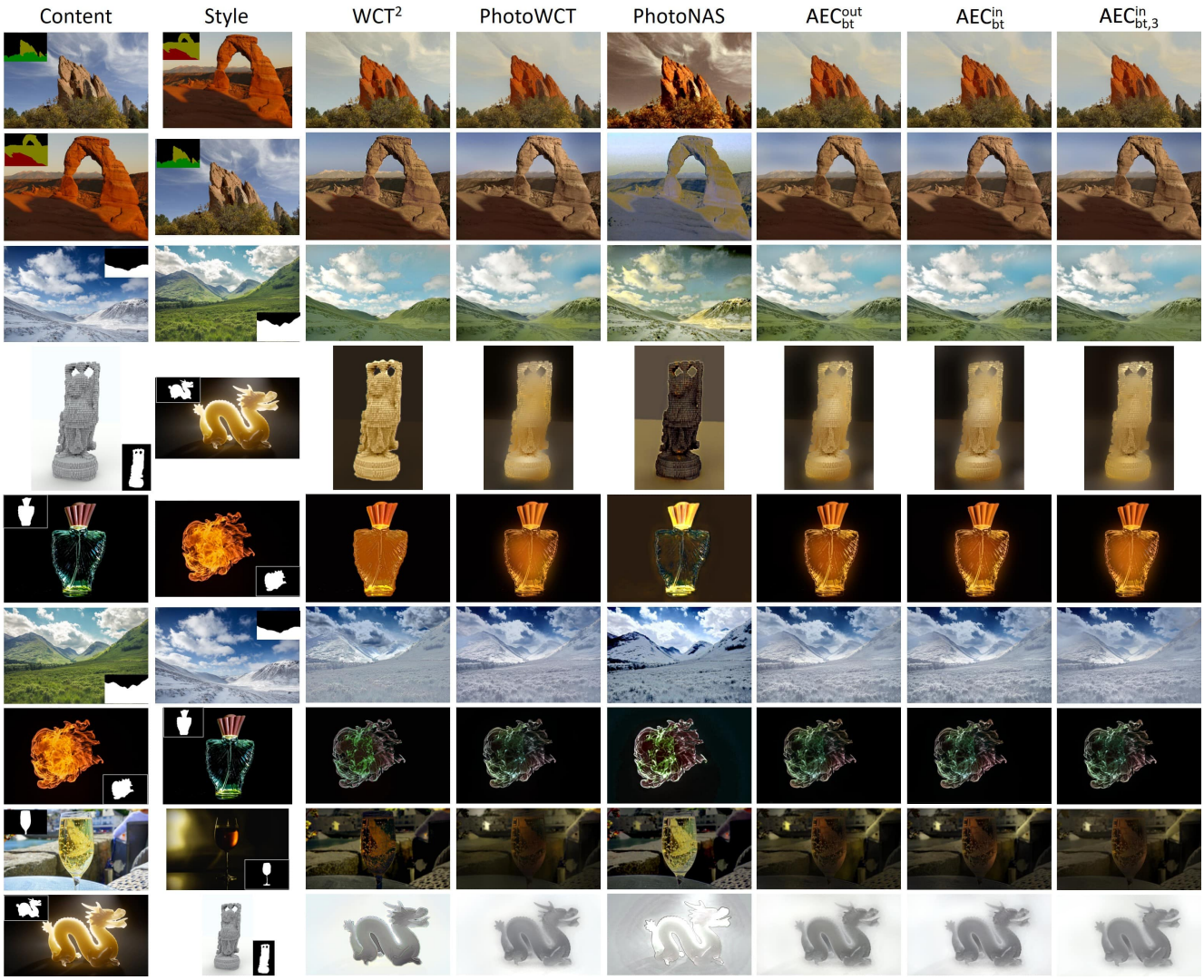


Figure 6: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 3/11)

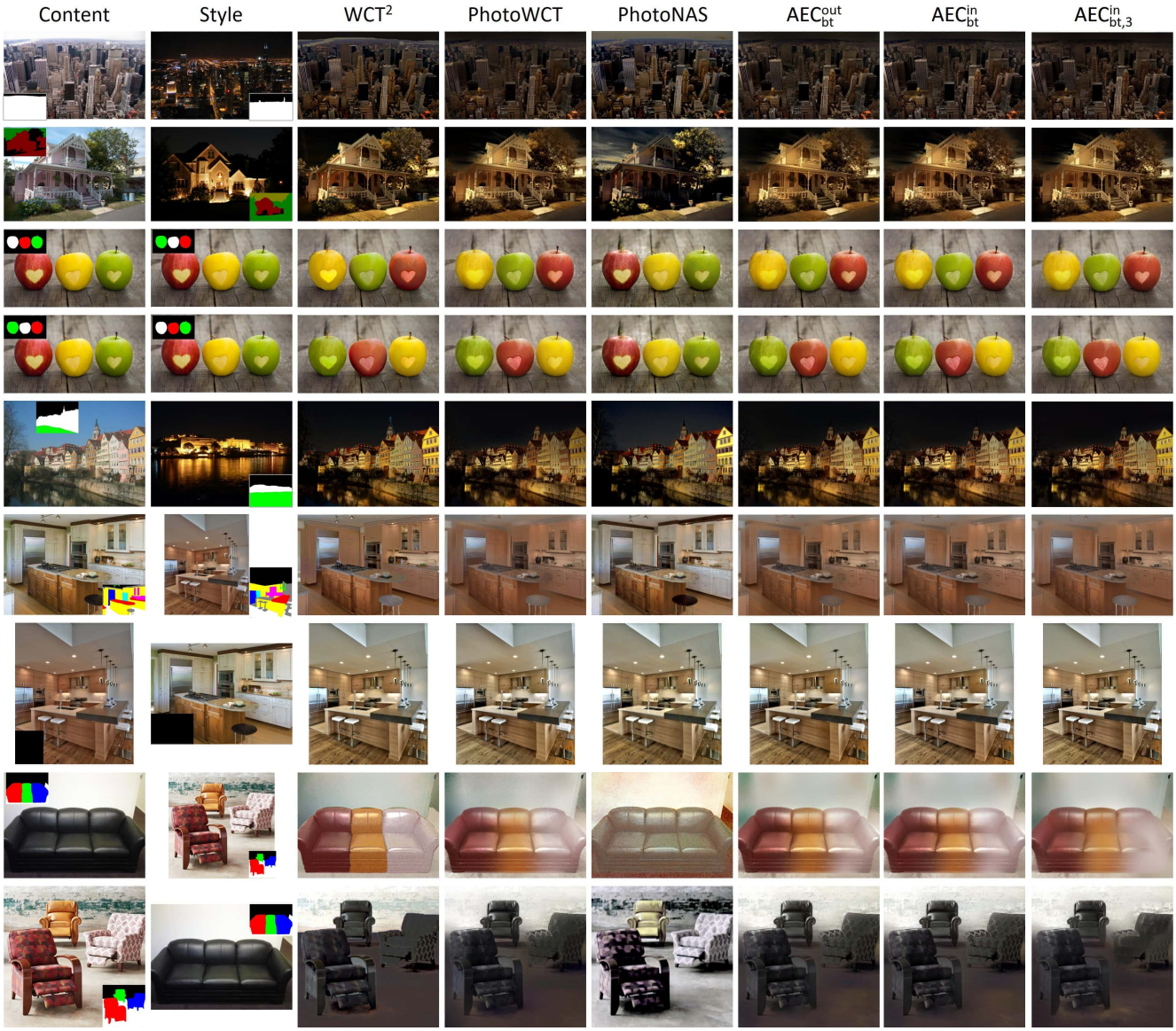


Figure 7: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 4/11)

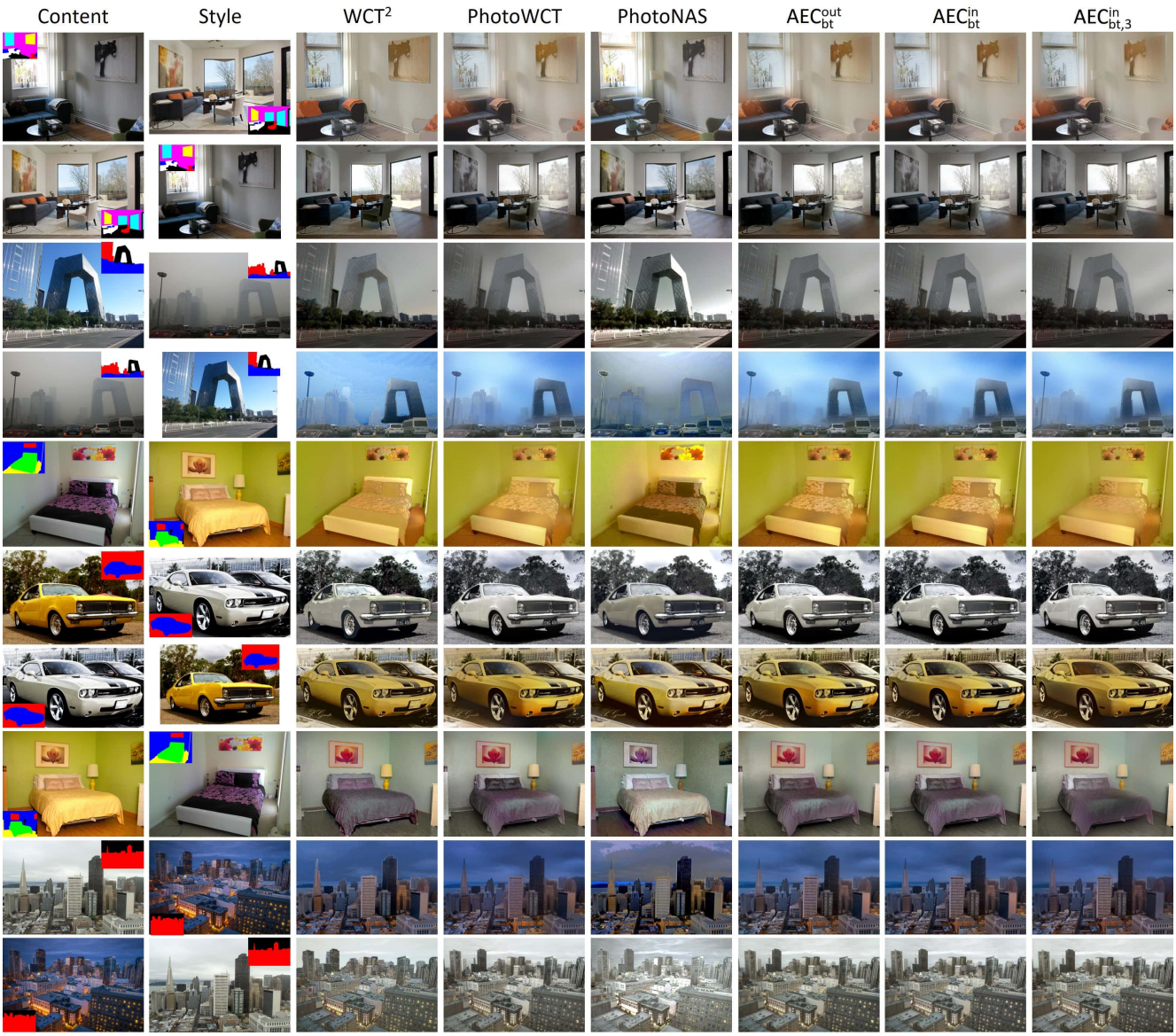


Figure 8: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 5/11)

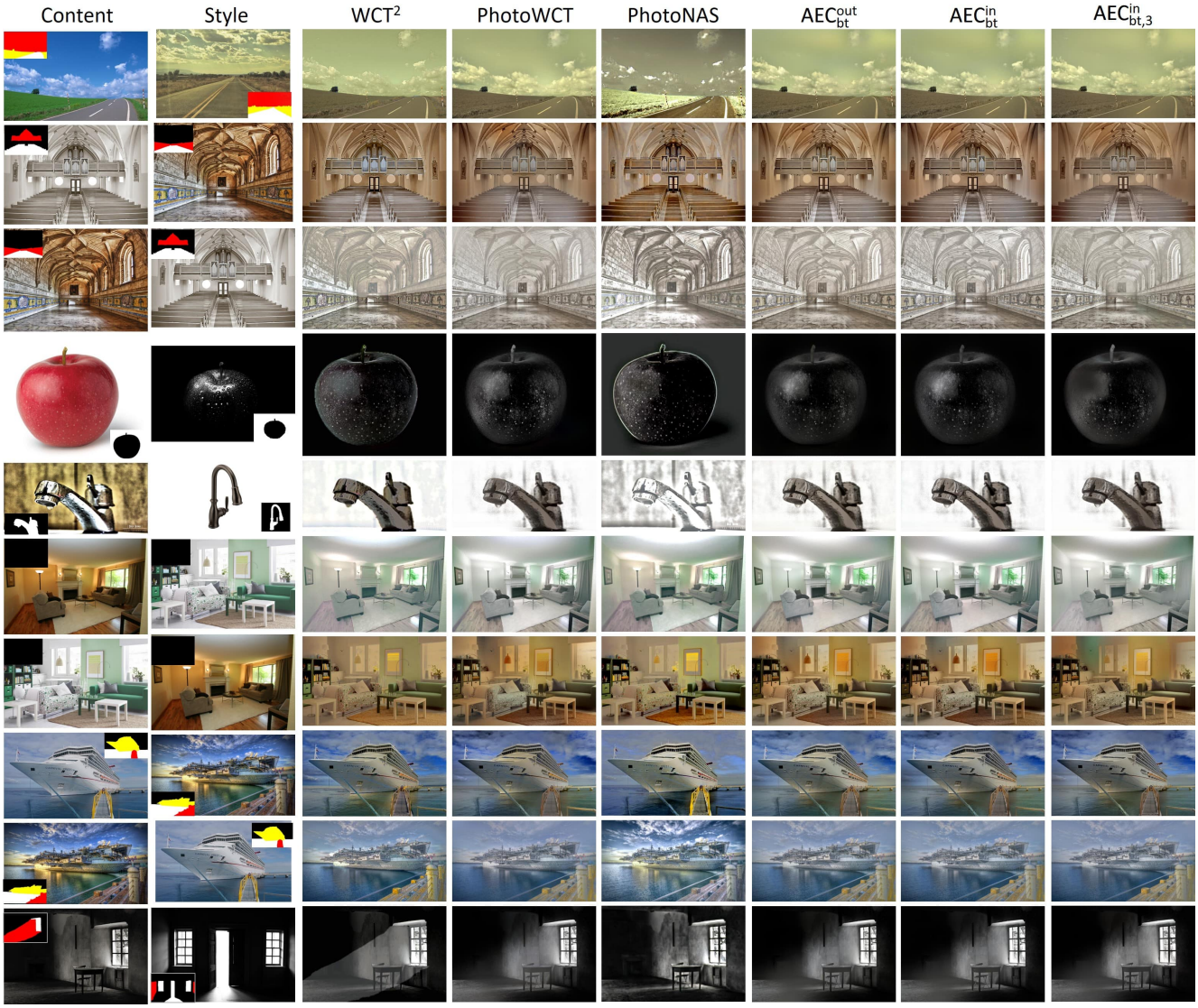


Figure 9: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 6/11)


Figure 10: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 7/11)

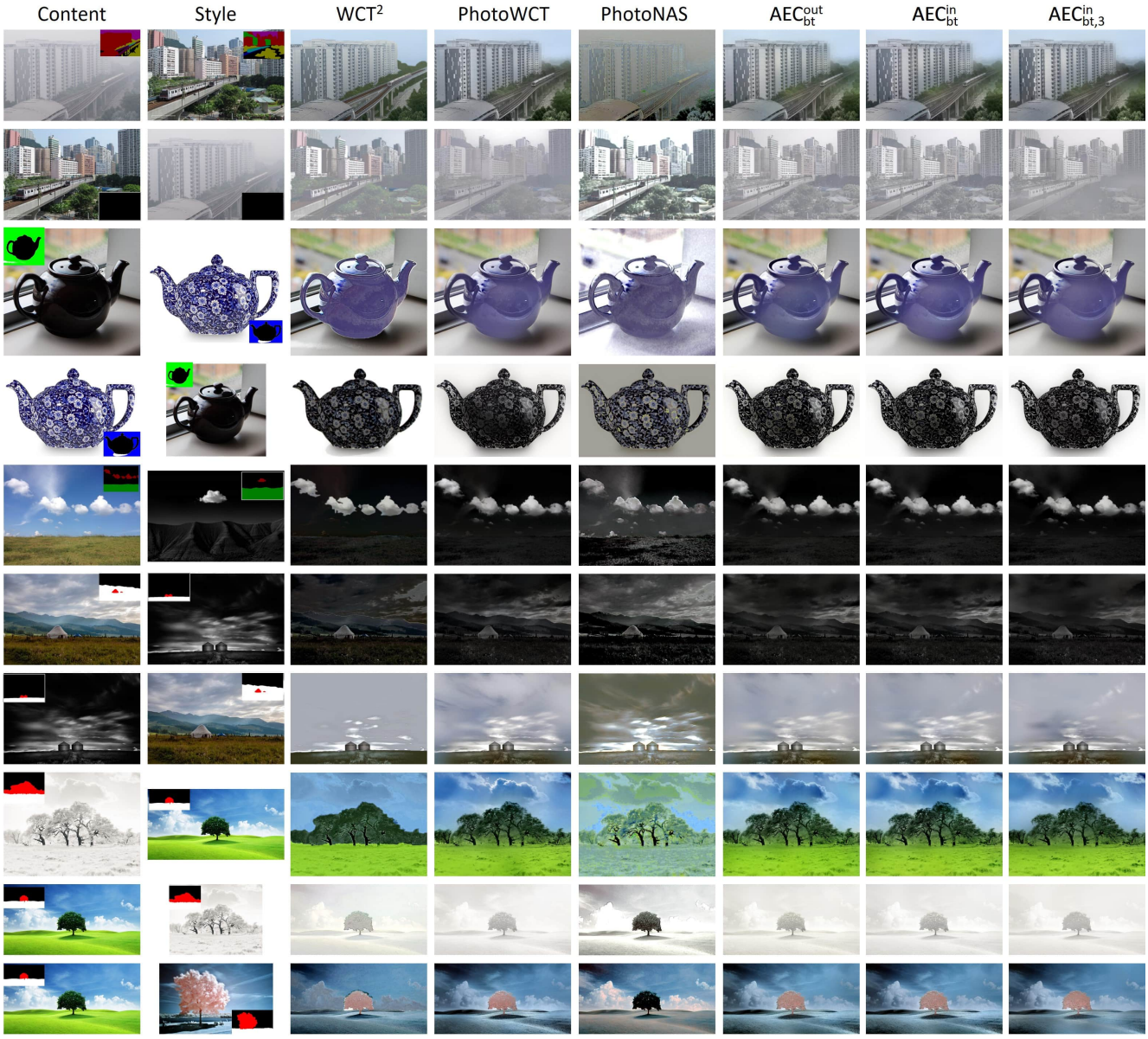


Figure 11: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 8/11)

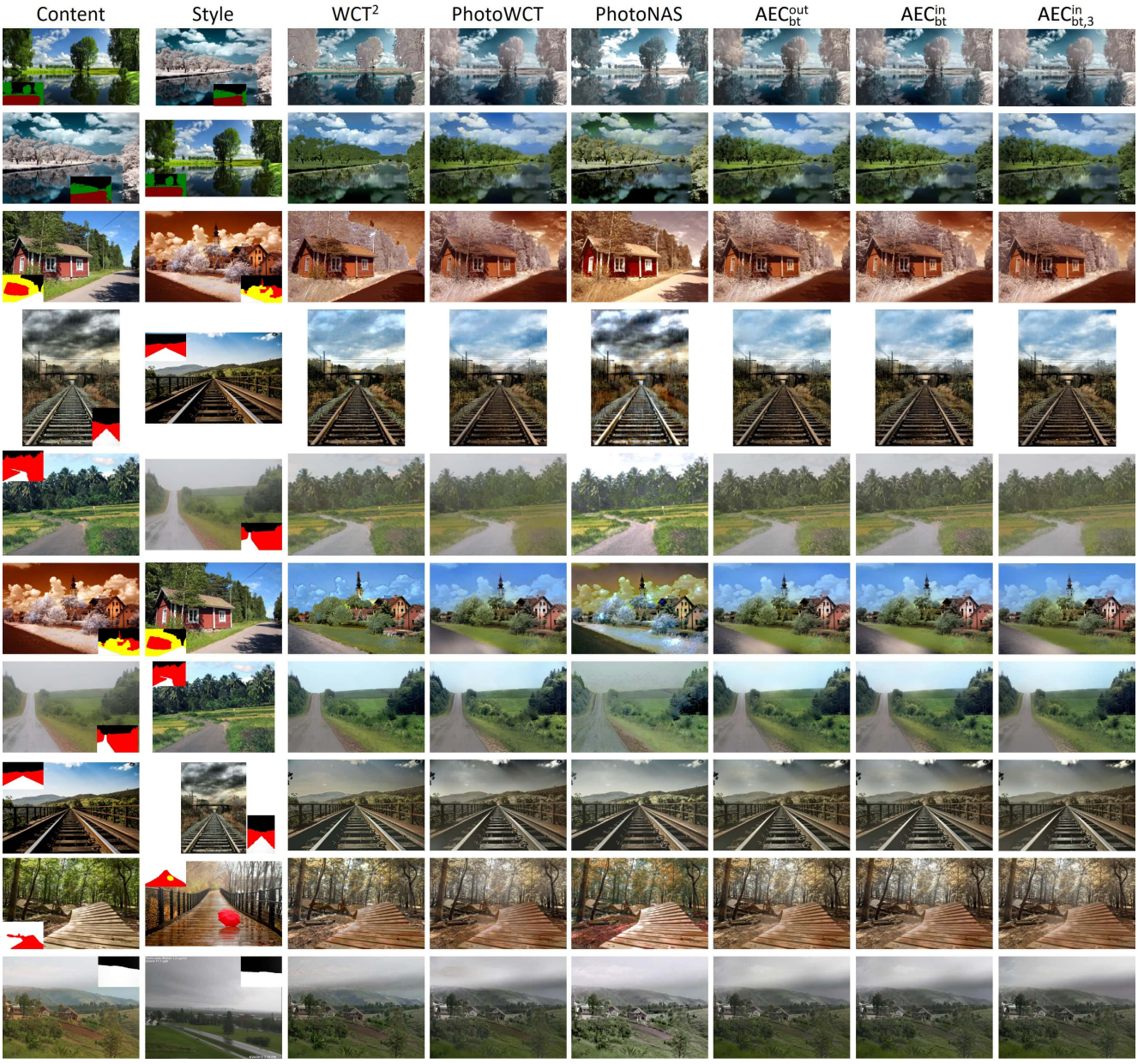


Figure 12: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 9/11)

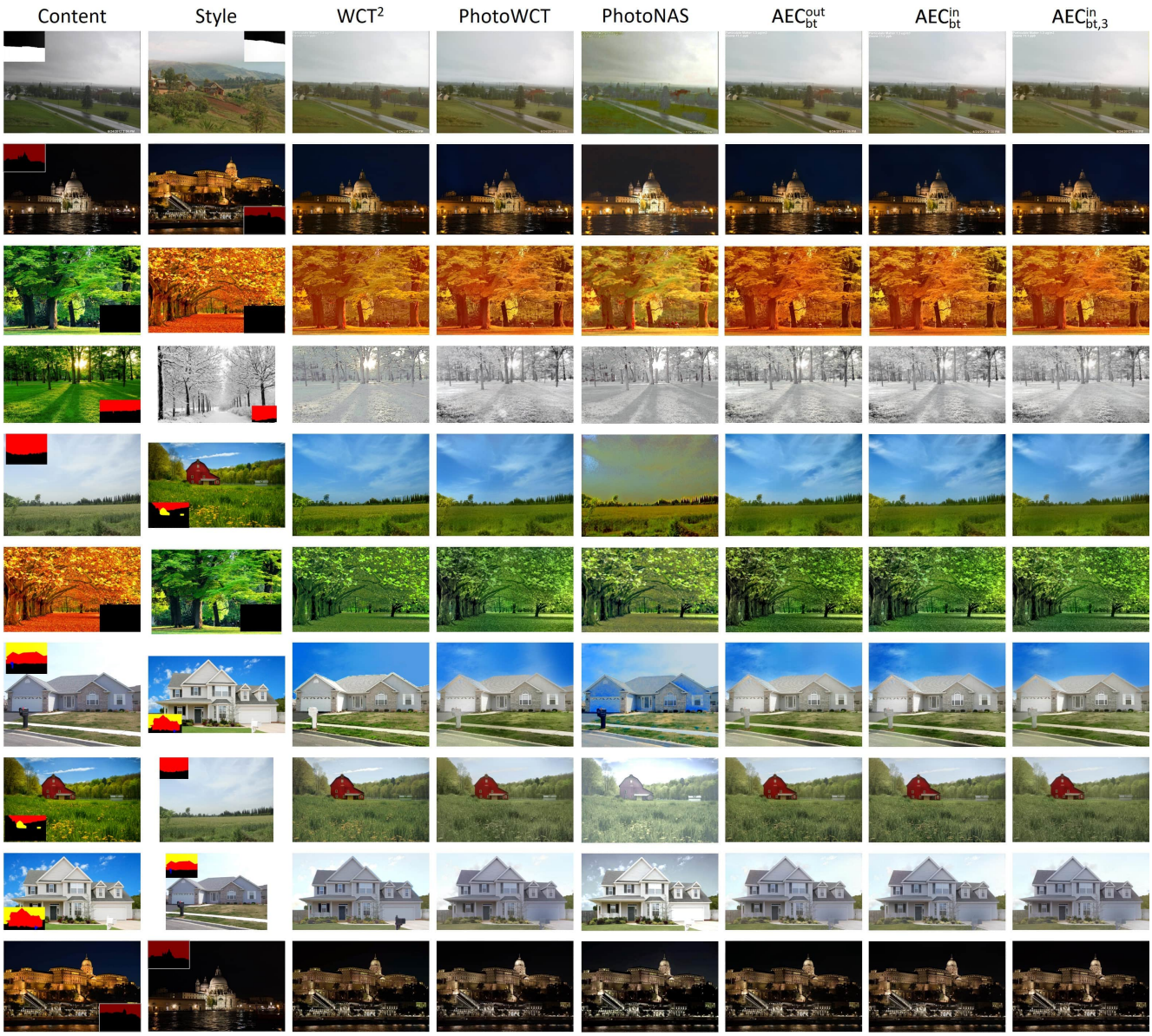


Figure 13: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 10/11)

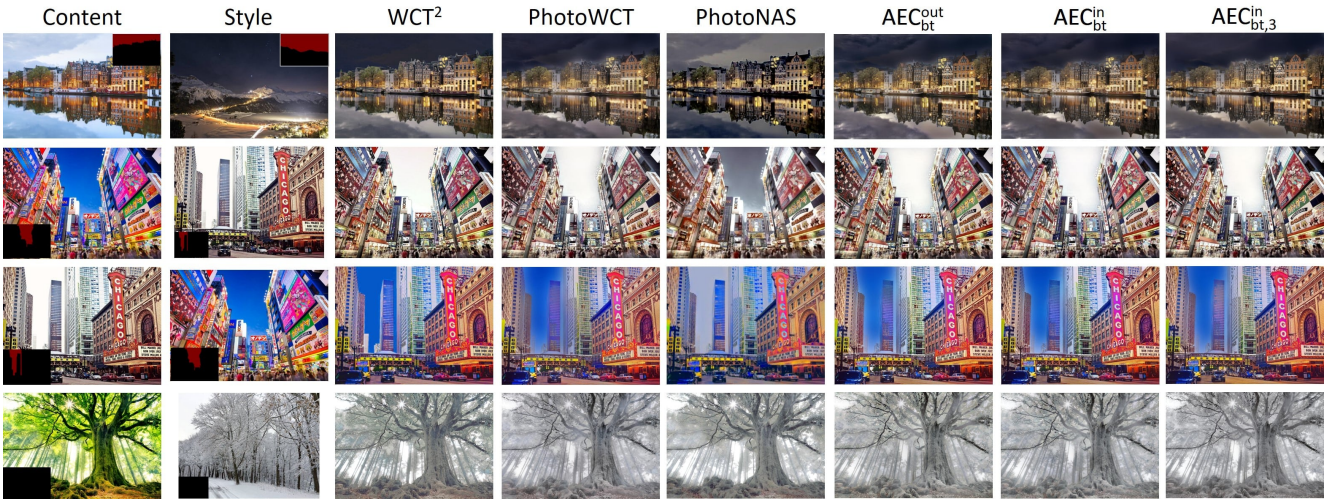


Figure 14: Results of stylization with segmentation for images in the DPST dataset. Three baselines are two state-of-the-arts PhotoWCT [6] and WCT$^2$ [9] and a more recent method PhotoNAS [1], while AEC$^{in}_{bt}$ (i.e., PhotoWCT$^2$) and AEC$^{out}_{bt}$ are the autoencoder in Figure 2 trained inward blockwisely and outward blockwisely, respectively, and AEC$^{in}_{bt,3}$ is AEC$^{in}_{bt}$ with enc$_4$blk$_4$ and dec$_{bt}$blk$_4$ removed. The results exemplify that our models achieve comparable stylization performance to the state-of-the-arts in a fraction of the time required by the existing methods. (Part 11/11)

## References


[1] Jie An, Haoyi Xiong, Jun Huan, and Jiebo Luo. Ultrafast photorealistic style transfer via neural architecture search. In *AAAI*, pages 10443–10450, 2020.

[2] Tai-Yin Chiu. Understanding generalized whitening and coloring transform for universal style transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4452–4460, 2019.

[3] 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*, pages 2414–2423, 2016.

[4] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *European conference on computer vision*, pages 1–14. Springer, 2010.

[5] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Advances in neural information processing systems*, pages 386–396, 2017.

[6] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 453–468, 2018.

[7] 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.

[8] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4990–4998, 2017.

[9] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9036–9045, 2019.