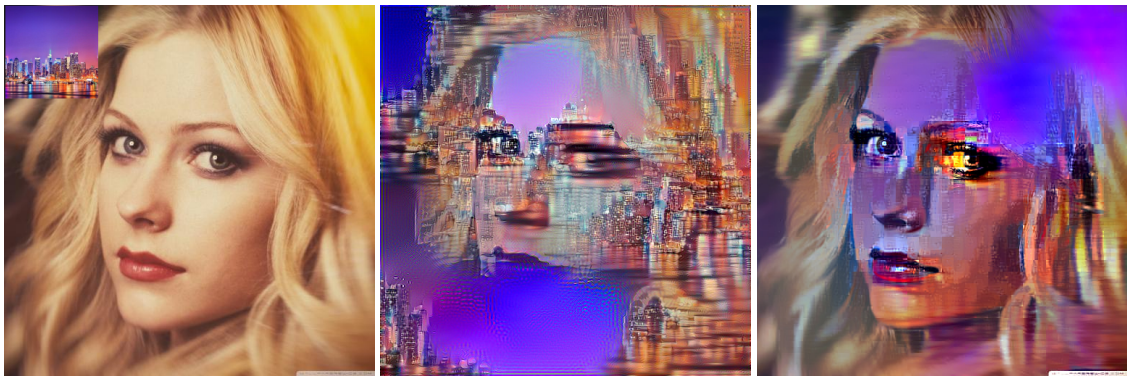
DeepObjStyle: Deep Object-based Photo Style Transfer (Supplementary Material)

Indra Deep Mastan and Shanmuganathan Raman
Indian Institute of Technology Gandhinagar
Gandhinagar, Gujarat, India
{indra.mastan, shanmuga}@iitgn.ac.in



Content & Style | Neural Style [1] | DPS [5]

WCT2 [8] | STROTSS [3] | DeepObjStyle

Figure 1: **Style transfer-1 (content-mismatch).** (a) The style image is shown at the top left corner of the content image. The style and content images contain different types of objects. (b) Neural style distorts the structure of the scene. (c) DPS [5] does not maintain the clarity of image features. (d) WCT2 [8] does not maintain the perceptual quality well and creates colored patches. (e) STROTSS [3] distributed image features better, but the output image does not preserve local level features details. (f) Our DeepObjStyle minimizes the effects of the content mismatch and preserve a better structure.

The outline of supplementary material is as follows. Sec. 1 provide more visual comparisons for the style transfer output and extended versions of figures from the manuscript. We provide more background of the DeepObjStyle in Sec. 2, Sec. 3, and Sec. 4. We provide implementation details in Sec. 5.

(a) Content image

(b) Style image

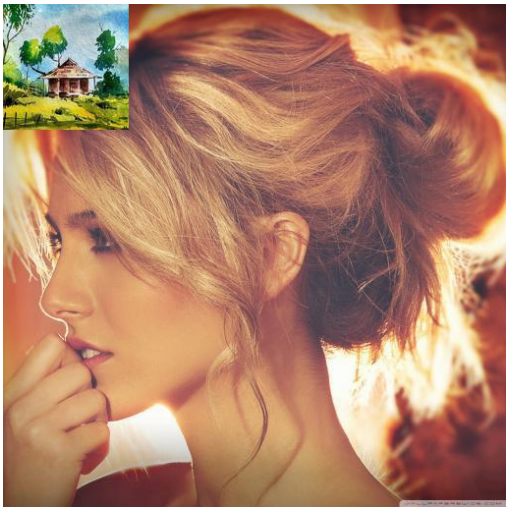
(c) DPS [5]

(d) WCT2 [8]

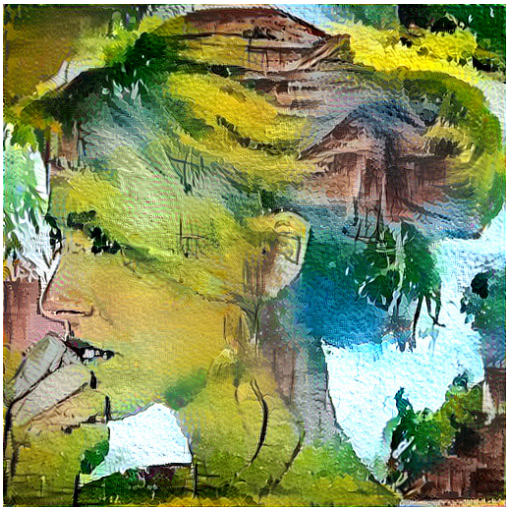
(e) STROTSS [3]

(f) DeepObjStyle (ours)

Figure 2: Style transfer-2 (wordcloud). The figure shows the extended version of Fig. 4 of the manuscript.



(a) Content & Style



(b) Neural Style [1]

(c) DPS [5]

(d) WCT2 [8]

(e) STROTSS [3]

(f) DeepObjStyle

Figure 3: Style transfer-3.

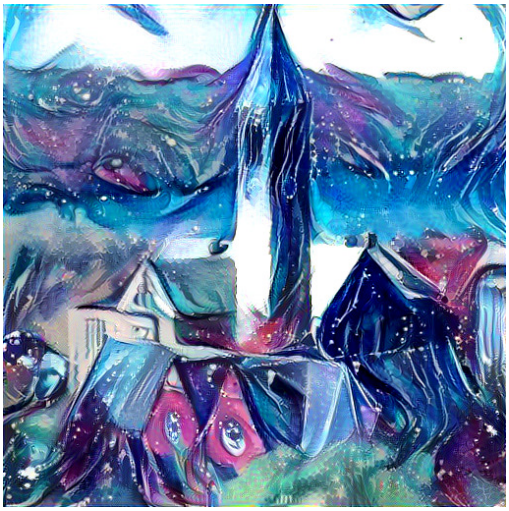
(a) Content & Style (b) Neural Style [1]

(c) DPS [5] (d) WCT2 [8]

(e) STROTSS [3] (f) DeepObjStyle

Figure 4: Style transfer-4.

(a) Content & Style



(b) Neural Style [1]

(c) DPS [5]

(d) WCT2 [8]

(e) STROTSS [3]

(f) DeepObjStyle

Figure 5: Style transfer-5.

1. Visual Comparision

We provide extended versions of the figures of the manuscript and more visual comparisons for the generated images. We have performed extensive experiments for many challenging scenarios of the style transfer. We observed that for some images, the style transfer output is visually good in the presence of content mismatch (Sec. 1.1). However, for some images, we found that when the mismatch of image features between input style and the content image is very high. Therefore, the style transfer output is not very clear. We call the above scenario as the extreme content mismatch (Sec. 1.3). The style transfer of the images containing a word cloud helps to investigates the distribution of image feature considering the semantics (Sec. 1.2). We discuss more about the results as follows.

1.1. Style Transfer (Content Mismatch)

Fig. 1 shows the style transfer when input style image and content images have a semantically different set of objects, *i.e.*, buildings in style image and face of a woman in the content image (extended version of Fig. 1 of the manuscript). We summarize the additional observation obtained from Fig. 1 as follows.

Neural style deforms the geometry of the objects. It is because the gram loss performs global style transfer and combines uncorrelated image features of the style and the content image to an object of the output image. DPS [5] output image in which the local level image features are not very clear due to mixing uncorrelated style features and content features. WCT2 [8] does not distribute the image features well and creates colored patches in the style transfer output. We have described in the manuscript, WCT2 uses the Microsoft COCO dataset to train the decoder. We believe that colored patches in WCT2 output might be because it is not generalizing well for the new style and the content images. STROTSS [3] performs style transfer by optimally transporting the image features on to the style transfer output with minimum distortions. However, the local level image feature details and the semantics of the scene are not very clear in the content mismatch example. DeepObjStyle (ours) provide a better distribution of image feature. It is because DeepObjStyle performs style transfer considering object context to minimize content mismatch.

We show more examples of content mismatch in Fig. 3, Fig. 4, and Fig. 5.

1.2. Style Transfer (Wordcloud Images)

Fig. 2 show the style transfer when input style and content images contain a wordcloud. It is challenging because the style features from the style image could spill over the word cloud in the output image. As we have described in the manuscript for Fig. 2, DPS [5] spill over the style features on the word-cloud, which makes the text unreadable. It could be because DPS distributes the style features without considering the context of the objects. For example, the features from the regions that have *low* contextual similarity in the content and the style images are mixed in the style transfer output. Therefore, resulting in a content mismatch. WCT2 [8] created colored patches in the style transfer output and reduced the image quality. It does not distribute the style features, and the content features well. STROTSS [3] optimally transport the style features on to the content image. However, the small font details are not preserved when performing style transfer. DeepObjStyle (ours) perform style transfer with a better distribution of the style and the content features to the output image. It transfers the image feature by considering the features with higher contextual similarity, and the features with the negative correlation are not used for the style transfer. Therefore, style features (blue color) are *not spilled* over the word-cloud. We show more examples of style transfer in the presence of word-cloud in Fig. 10, Fig. 11, and Fig. 12.

1.3. Style Transfer (Extreme Content Mismatch)

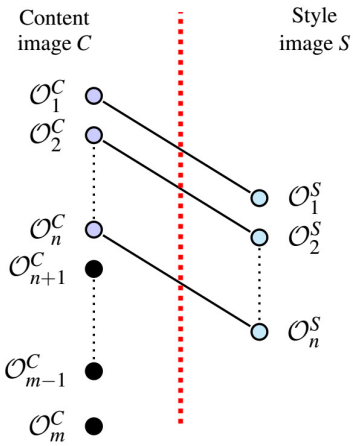
We believe that the extreme content mismatch comes when there is a high image features difference in the input style and content images. For example, style image with a dark color object and content image with a light color object. We have shown an example of extreme content mismatch in Fig. 13. It could be observed that DeepObjStyle preserves better object structure in the style transfer output.

2. Unmapped Objects

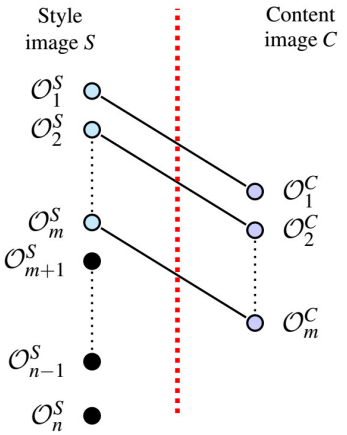
Fig. 6 shows an example of the object map and the unmapped objects that exists when performing style transfer. As we described in the manuscript, given a content image C and a style image S, the style transfer output O incorporates content features from C and style features from S. Let $\mathcal{O}^C = \{\mathcal{O}_i^C\}_{i=1}^m$ denotes the objects of the content image, $\mathcal{O}^S = \{\mathcal{O}_i^S\}_{i=1}^n$ denotes the objects of style image. Let $\mathcal{O}^O = \{\mathcal{O}_i^O\}_{i=1}^m$ denotes the objects of style transfer output image. The one-to-one mapping would allow the content object $\mathcal{O}_i^C$ and the style object $\mathcal{O}_i^S$ to provides the content and the style features to an object of output image $\mathcal{O}_i^O$. The black nodes in Fig. 6 shows the unmapped objects. The content mismatch in the style transfer output would occur when the unmapped objects are present in the style image or the content image. In the manuscript, we describe unmapped object loss for unmapped objects. The challenge is to use the unmapped objects while minimizing the content mismatch in the style transfer output.

3. More Background for DeepObjStyle

DeepObjStyle deep photo-style loss [5] for image features supervision for the mapped objects. It also uses con-



(a) STP-C. The objects of the content image are more than that of the objects of the style image (i.e., $m > n$).



(b) STP-S. The objects of the style image are more than that of the objects in the content image (i.e., $m < n$).

Figure 6: **Style Transfer Problems (STP)**. A pictorial representation of object mapping for STP-C and STP-S defined in Sec. 2 of manuscript.

textual loss [6] formulation for the following two things. Contextual content loss for the mapped objects and contextual style loss for the unmapped objects (Sec. 3 of the manuscript). Deep photo-style provides photorealism in the output and the contextual loss transfer image features between the contextually similar regions to minimize the content mismatch. Here we describe these loss functions as follows.

3.1. Contextual Loss (CL).

We use the contextual loss proposed by Mechrez *et al.* [6]. The main idea is first to extract context vectors from the image using VGG19, and then measure the similarity between images, based on the similarity between their context vectors. Therefore, CL could compare images even if they are not aligned as it ignores the comparison of the spatial positions in the images. Formally, consider an image x and a target y. Let the features extractor VGG19 pretrained network be denoted by ϕ. Let $X = \{\phi^l(x)_i\}_{i=1}^N$ and $Y = \{\phi^l(y)_j\}_{j=1}^N$ be the context vectors present at layer l. The contextual similarity is computed by finding for each feature $\phi^l(y)_j$, a feature $\phi^l(x)_i$ that is most similar to it and then sum for all $\phi^l(y)_j$.

$$CX(x,y) = CX(X,Y) = \frac{1}{N}\sum_j \max_i CX_{ij} \quad (1)$$

Here, CX_{ij} is the similarity between the context vectors $\phi^l(x)_i$ and $\phi^l(y)_j$ [6] . The contextual loss is $\mathcal{L}_{cl}(x,y,l) = -\log CX(X,Y)$. It minimizes dissimilarities between the contextual features computed from the source image x and the target image y.

3.2. Deep Photo Style Loss (DPS).

Here we give expresession for deep photo-style loss proposed by Luan et al. for completeness [5]. The deep photo-style loss $\mathcal{L}_{dps}$ includes the content loss $\mathcal{L}_C^l$ and the style loss $\mathcal{L}_{s+}^l$ with the photo-realism regularization $\mathcal{L}_m$. The expression for the loss $\mathcal{L}_{dps}$ is given in Eq. 2.

$$\mathcal{L}_{dps} = \alpha_{11}\sum_{l\in L}\mathcal{L}_C^l + \alpha_{12}\sum_{l\in L}\mathcal{L}_{s+}^l + \alpha_{13}\mathcal{L}_m \quad (2)$$

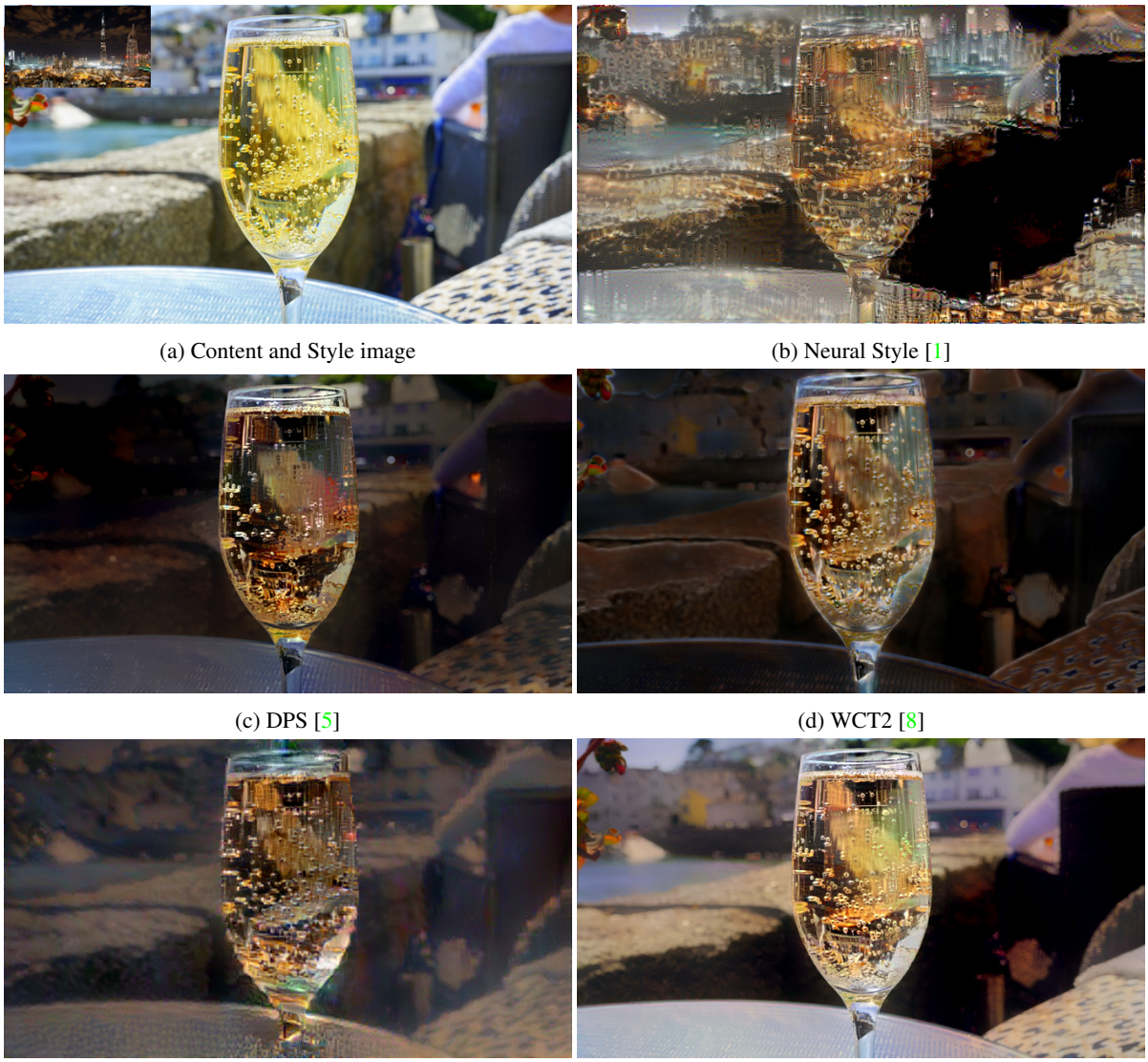
where, L is the total number of convolutional layers and l denotes the $l-$th convolutional layer. As we have described in the manuscript that the deep photo-style loss $\mathcal{L}_{dps}$ constrains image features of the style transfer output to be locally affine in colorspace to suppress distortions. The photo-realism regularization $\mathcal{L}_m$ term in the objective function provide photorealistic style transfer.

4. Deep Photo Style Transfer$^+$ (DPS$^+$)

This section describes an object-based approach that incorporates DPS [5] with unmapped object loss, named DPS$^+$. It is independent of the contextual loss. We used it to get a fair comparison for DPS [5] (see Fig. 3 of the manuscript). DPS$^+$ uses the loss $\mathcal{L}_{dps^+}$ defined in Eq. 3.

$$\mathcal{L}_{dps^+} = \gamma_1\mathcal{L}_{dps} + \gamma_2\,\mathcal{L}_{gl,S}^U \quad (3)$$

Here, $\mathcal{L}_{dps}$ is the mapped object loss and $\mathcal{L}_{gl,S}^U$ is the unmapped object loss. Mapped object loss $\mathcal{L}_{dps}$ is the deep photo-style loss defined in Eq. 2. The definition of the unmapped objects loss $\mathcal{L}_{gl,S}^U$ based on the style transfer problems (STP) is as follows.



(a) Content and Style image

(b) Neural Style [1]

(c) DPS [5]

(d) WCT2 [8]

(e) STROTSS [3]

(f) DeepObjStyle (ours)

Figure 7: Style transfer-6.

- STP-E: If there is an equal number of semantic objects in the style and the content image, then we do not have unmapped objects. As a result, $\mathcal{L}^{U}_{gl,S}$ does not exist and we get $\mathcal{L}_{dps^{+}} = \gamma_1 \mathcal{L}_{dps}$. Therefore, DPS^{+} is an extension of DPS to include unmapped objects.

- STP-C: The unmapped objects loss $\mathcal{L}^{U}_{gl,S}$ is computed between the unmapped objects in the output O and the objects of the style image S. We define $\mathcal{L}^{U}_{gl,S}$ to be similar to $\mathcal{L}^{l}_{gl,S}$ as given in Eq. 4 of the manuscript.

- STP-S: The unmapped objects loss $\mathcal{L}^{U}_{gl,S}$ is computed between the unmapped objects of the style image S and the output image O. We define $\mathcal{L}^{U}_{gl,S}$ to be equivalent to $\mathcal{L}^{l}_{gl,S}$ as shown in Eq. 6 of the manuscript.

In Fig. 3 of the manuscript, we show that style transfer output of DPS^{+}. However, a better distribution of image features is achieved using DeepObjStyle, which compares the semantics of the objects when performing the style transfer.

5. Implementation Details

Style transfer methods mostly use the feature extractor pre-trained VGG19 [7] to compute the loss function [5, 3]. It is because the style representation and content representations are captured at the different layers of the VGG19. We also took pre-trained *VGG19* is used as the feature extractor. We used `conv1_1`, `conv2_1`, `conv3_1`, `conv4_1` and

conv5_1 as the style representation. The layer conv4_2 is utilized as the content representation. We use the content image as input because we observed that the contextual loss does not perform well with the random noise as input. It is because feeding a random image as input to the feature extractor VGG19 might not output useful contextual features to compute the contextual loss.

Word-cloud images are created by pasting word-cloud on the style images and the content images. The input images for style transfer were taken from the project page of [4, 9]. The images from [4] contains the segmentation information. The images from [9] do not have the segmentation information, and we took the full image as one segment to observe how the different methods distribute image features.

We perform smoothing on the style transfer output similar to DPS [5]. It is done to ensures spatially consistent stylizations and remove structural artifacts. We get the edges from the content image and then add it to the style transfer output to enhance the edges. We use unsharp mask [2] to perform the enhancement of edges. The above simple step enhances the image structure very well.

References

[1] 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. 1, 3, 4, 5, 8, 10, 11, 12, 13

[2] Soroush Javadi. Unsharp mask. https://github.com/soroushj/python-opencv-numpy-example/blob/master/unsharpmask.py, 2017. 9

[3] Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. Style transfer by relaxed optimal transport and self-similarity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10051–10060, 2019. 1, 2, 3, 4, 5, 6, 8, 10, 11, 12, 13

[4] Yang Liu. deep-photo-style-transfer-tf. https://github.com/LouieYang/deep-photo-styletransfer-tf, 2017. 9

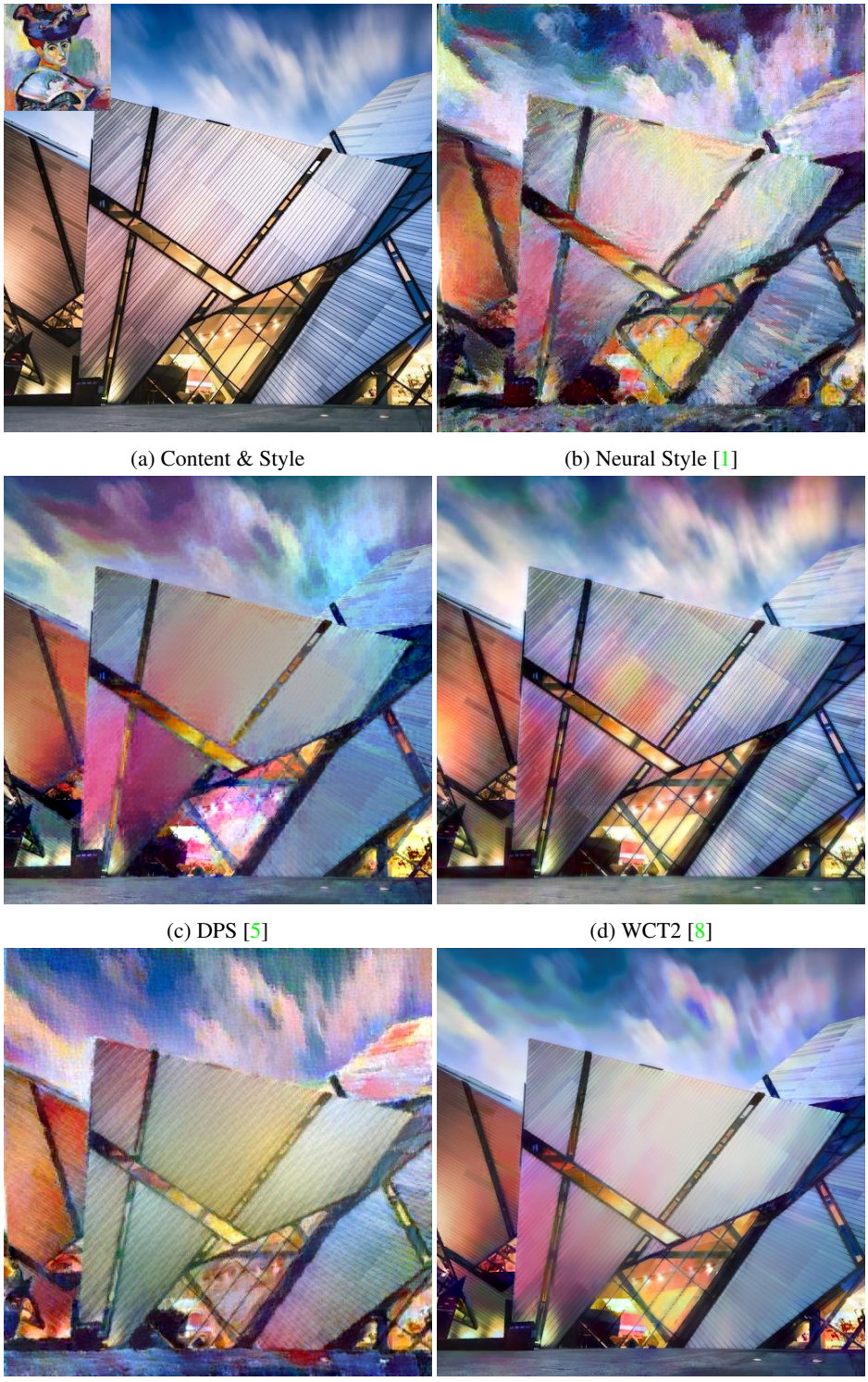
[5] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6997–7005, 2017. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13

[6] Roey Mechrez, Itamar Talmi, and Lihi Zelnik-Manor. The contextual loss for image transformation with non-aligned data. *European Conference on Computer Vision (ECCV)*, 2018. 7, 12, 13

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 8

[8] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *International Conference on Computer Vision (ICCV)*, 2019. 1, 2, 3, 4, 5, 6, 8, 10, 11, 12, 13

[9] yulunzhang. Mst. https://github.com/yulunzhang/MST/tree/master/data, 2019. 9



(a) Content & Style (b) Neural Style [1]

(c) DPS [5] (d) WCT2 [8]

(e) STROTSS [3] (f) DeepObjStyle

Figure 8: Style transfer-7.

(a) Content & Style (b) Neural Style [1]

(c) DPS [5] (d) WCT2 [8]

(e) STROTSS [3] (f) DeepObjStyle

Figure 9: Style transfer-8.

(a) Content

(b) Style

(c) Neural [1]

(d) DPS[5]

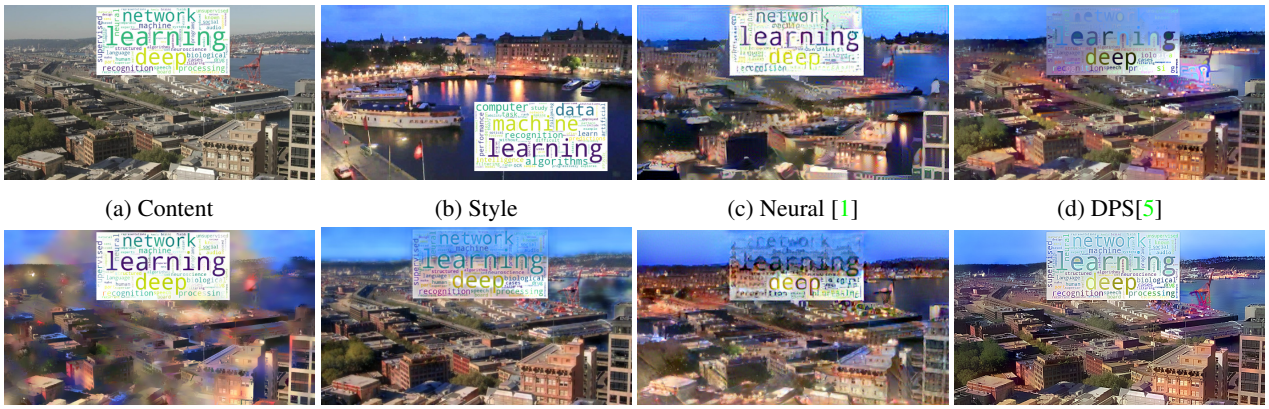
(e) CL[6]+$\mathcal{L}_m$[5]

(f) WCT2 [8]

(g) STROTSS [3]

(h) DeepObjStyle

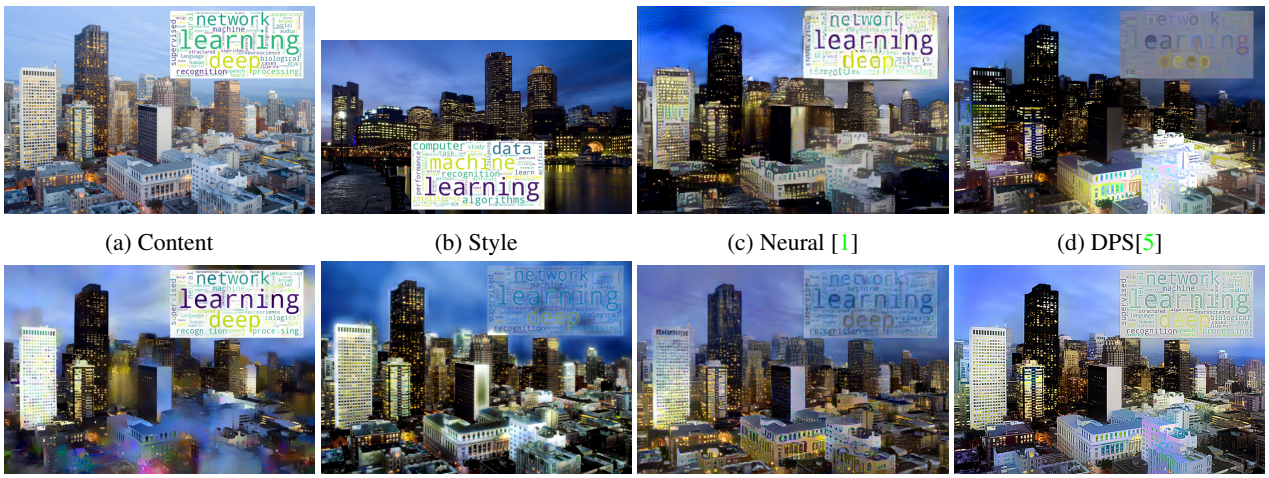
Figure 10: Style transfer-9 (word-cloud).



(a) Content (b) Style (c) Neural [1] (d) DPS[5]

(e) CL[6]+$\mathcal{L}_m$[5] (f) WCT2 [8] (g) STROTSS [3] (h) DeepObjStyle

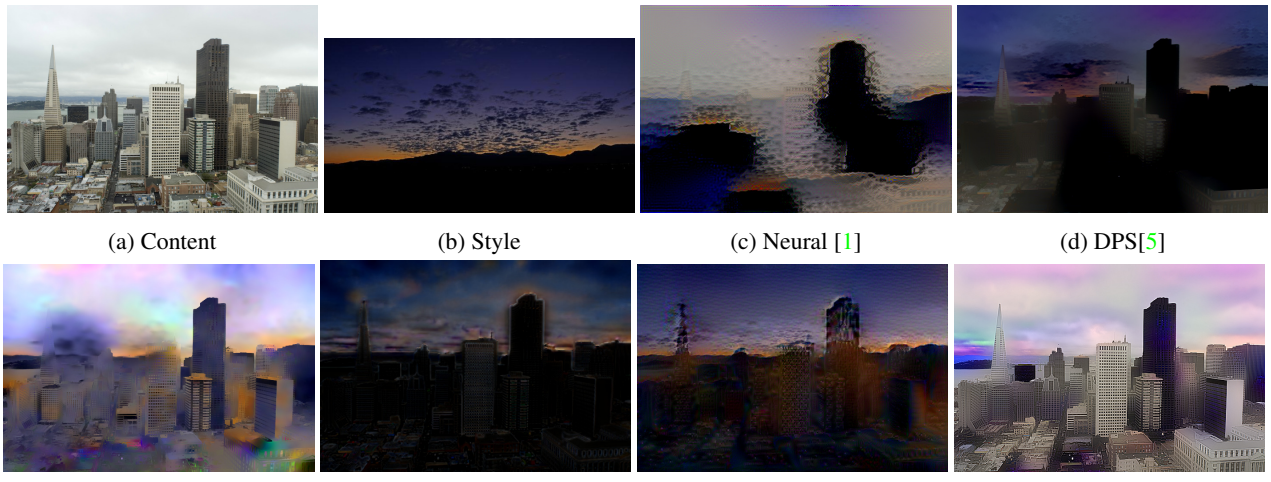
Figure 11: Style transfer-10 (word-cloud).



(a) Content (b) Style (c) Neural [1] (d) DPS[5]

(e) CL[6]+$\mathcal{L}_m$[5] (f) WCT2 [8] (g) STROTSS [3] (h) DeepObjStyle

Figure 12: Style transfer-11 (word-cloud).



(a) Content (b) Style (c) Neural [1] (d) DPS[5]

(e) CL[6]+$\mathcal{L}_m$[5] (f) WCT2 [8] (g) STROTSS [3] (h) DeepObjStyle

Figure 13: Style transfer-12 (extreme-mismatch).