Class-Based Styling: Real-time Localized Style Transfer with Semantic Segmentation

Lironne Kurzman
University of British Columbia
lironkurz@gmail.com

David Vazquez
Element AI
dvazquez@elementai.com

Issam Laradji
Element AI; University of British Columbia
issam.laradji@gmail.com

Abstract

We propose a Class-Based Styling method (CBS) that can map different styles for different object classes in real-time. CBS achieves real-time performance by carrying out two steps simultaneously. While a semantic segmentation method is used to obtain the mask of each object class in a video frame, a styling method is used to style that frame globally. Then an object class can be styled by combining the segmentation mask and the styled image. The user can also select multiple styles so that different object classes can have different styles in a single frame. For semantic segmentation, we leverage DABNet that achieves high accuracy, yet only has 0.76 million parameters and runs at 104 FPS. For the style transfer step, we use the popular real-time method proposed by Johnson et al. [7]. We evaluated CBS on a video of the CityScapes dataset and observed high-quality localized style transfer results for different object classes and real-time performance. The code is available at https://github.com/IssamLaradji/CBStyling.

1. Introduction

One important aspect of style transfer is that it allows us to create new, beautiful artistic works. Style transfer maps the style of an image onto a target image, while maintaining the content of that target image. Jing et al. [5] conducted a review of neural style transfer methods, and evaluated their different applications. While style transfer methods have existed for over a decade, only recently did new methods emerge that leverage the powerful representation of deep learning. Perhaps Gatys et al. [3] proposed the first such style method that proved to be seminal work. The reason is that it enabled high-quality style creation by simply combining a style image and a content image. Unfortunately, this method is too slow to be used in real-time.

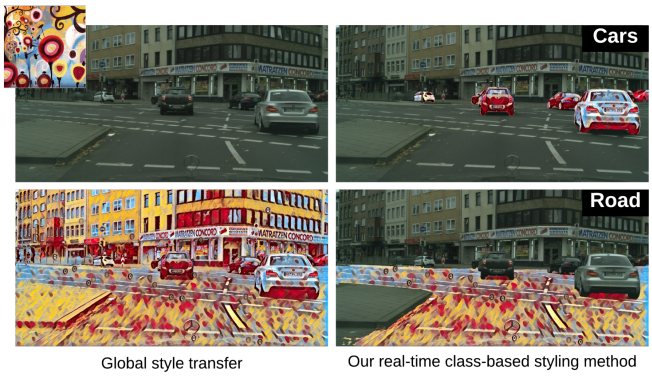

Figure 1: (Left) A global style transfer method from Johnson et al. [7] was applied to a CityScapes image. (Right) CBS (ours) was used to style *cars* and *road*.

Later, another seminal work emerged that can run in real-time [7]. It differs from the method proposed in Gatys et al. [3] by using a deep network that performs a single feed-forward at inference time, instead of optimizing for the style and content at inference time. Namely, the method by Johnson et al. [7] directly transforms an input image to a styled image, which makes it fast enough to run in real-time.

One limitation of Johnson et al. [7]'s method is that it transfers the style to the image globally. In fact, most methods have this limitation [6]. Castillo et al. [1] addressed this by proposing a localized style transfer method. It achieves this with the help of a Mask R-CNN [4] to obtain object instances of the image. Then, it uses Gatys et al. [3] method to style a specific object instance that is specified by the user. Further, Castillo et al. [1] proposed an MRF-based loss to smooth the boundary pixels of the stylized objects.

Unfortunately, their method cannot run in real-time. This is because Gatys et al. [3] method, Mask R-CNN, and the MRF-based loss all have slow inference time.

In this work, we propose CBS, a Class-Based Styling method that performs localized style transfer in real-time. Given an input image, it extracts masks for different object classes using a pretrained DABNet [8], a real-time semantic segmentation method (it runs at 104 fps for CityScapes [2]). In parallel, it uses Johnson et al. [7] real-time style transfer method to map a style on that input image.

CBS then applies the style only on those extracted masks. In this case, the user specifies which class the style should be applied to. For example, Figure 1 shows the style being applied to cars and road. Experimental results show that these two steps are fast enough to be real-time. Thus, we summarize our contributions as follows: (1) we propose a novel framework performing real-time localized style transfer, and (2) show that CBS achieves high-quality stylized images with real-time performance.

2. Related Work

Style Transfer It was Gatys et al. [3] who first introduced the use of deep convolutional neural networks (CNNs) in whole-image styling, by minimizing both the content and, style-loss simultaneously. Despite producing beautiful results it wasn't very applicable to real-time videos due to the relatively slow inference speed. When Johnson et al. [7] extended their work using a feed-forward network, inference became substantially quicker and video style transfer became possible.

Real-time Semantic Segmentation Real-time semantic segmentation is a trade-off between accuracy and inference speed. Many of the frameworks available for a single-image semantic segmentation would fare poorly in real-time due to their slow inference. Li et al. [8] proposed a Depth-wise Asymmetric Bottleneck module to address the speed of semantic segmentation. They extract local and contextual features jointly, improving the speed without compromising accuracy.

Localized Real-time Style Transfer Neural Style Transfer (NST) has been applied to specific regions of still images [1] through masking the objects of the target image, and modifying the original loss function to consider only the pixels of interest. Castillo et al. [1] have succeeded in creating images with partial styling as well. Their model classified pixels to foreground or background and applied the style to the pixels corresponding to objects selected by the user. They used Markov random fields (MRFs) to deal with the boundaries of the styled region in order smoothing out the outlier pixels as a way to anti-alias. However, applying their method does not work in real-time as they use Gatys et al. [3] method for the style transfer, which is slow.

[1] Found here: http://cs231n.stanford.edu/reports/2017/pdfs/416.pdf

3. Proposed Method

We present CBS, a Class-Based Styling method. that can perform style transfer to specific object classes in real-time. To style an object class c on an image I, CBS follows these steps.

First, using a fast segmentation network [8], CBS extracts a binary mask R_c for object class c. R_c has the same height and width as I with entries 1 for the object belonging to class c and 0 otherwise. Note that the segmentation network is pretrained on segmenting object class c. Simultaneously, CBS transforms I to a styled image T_S using a fast styling method (we used the method by Johnson et al. [7]).

Then, using the binary mask R_c, CBS extracts the background I_b from the unstyled image I and extracts the foreground T_f from the styled image T_S. Finally, CBS adds T_f and I_b to obtain a target image U that contains an unstyled background and a styled object class c (Figure 3).

Figure 2 provides an illustration of this pipeline. While this pipeline is similar to that proposed in Castillo et al. [1], theirs is slow due to the requirement of Gatys et al. [3] method, Mask R-CNN, and MRF blending. In contrast, our proposed framework can run in real-time with 16 fps. Below we explain CBS's components in detail.

3.1. Fast Styling Method

The goal is to generate a stylized image with the same content as input image I in the style of an image S in real-time. Thus, we use the fast styling method (FSM) proposed by Johnson et al. [7]. It trains a fully convolutional neural network (FCN) that learns to generate an image $T_S = M(I)$ with the content of the input image I and style S. As a result, no optimization is required at inference time. Rather, the FCN performs a single forward pass for an image I to get the stylized image. This leads to a high FPS rate when this method is applied on video. At training time, FSM optimizes the following loss function,

$$\mathcal{L}(I) = \underbrace{\frac{1}{C_2 H_2 H_2} ||F_2(M(I)) - F_2(I)||_2^2}_{\text{content loss}} + \underbrace{\sum_{l}^{L} \frac{1}{C_l} ||G(M(I), l) - G(S, l)||_F^2}_{\text{style loss}}$$

$$\underbrace{G_{i,j}(X, l)}_{\text{gram matrix}} = \frac{1}{H_l W_l} \sum_{h}^{H_l} \sum_{w}^{W_l} F_{l_{(h,w,i)}}(X) \cdot F_{l_{(h,w,j)}}(X), \quad (1)$$

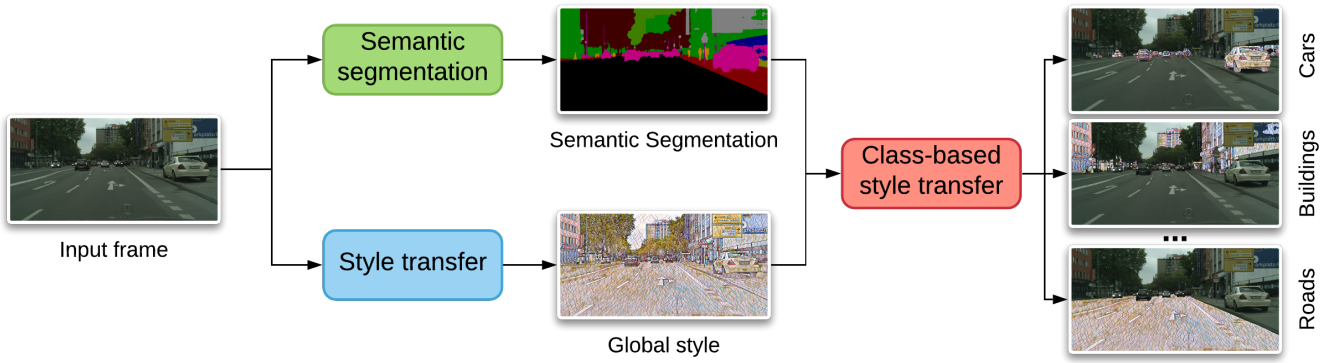

Figure 2: CBS pipeline. First, CBS takes a video frame as an input image and performs two operations in parallel: semantic segmentation and global style transfer. Then, the segmentation mask is combined with the styled image to style only the object classes of interest.

where $F_l(\cdot)$ is a feature extraction network (typically chosen as VGG16 [10]) that outputs a feature map at the l-th level (VGG16 outputs 4 levels of feature maps). $C_l \times H_l \times W_l$ is the dimension of the level l feature map.

The content loss encourages FSM to preserve the spatial structure without preserving the color, texture and exact shape (these 3 properties define the style of the image). It can be considered as a feature reconstruction loss defined as the L2 norm between the feature representations of the input image and the generated image.

The style loss is there to preserve the style given by the style image S which includes color and texture. It is computed as the uncentered covariance between the extracted features of an image, which captures the relationship between feature pairs. This is defined as the Gram matrix which evaluates which feature channels tend to activate together. Since this loss does not affect the spatial structure of the input image, this mainly encourages the input image to undergo a style change. On the other hand, the transformed input image keeps its spatial structure with the help of the content loss.

3.2. Fast Segmentation Network

The goal of the semantic segmentation component is to generate binary masks that represent the regions containing the object classes C of interest. We use DABNet [8], which is a recent semantic segmentation method that achieves real-time performance by combining depth-wise separable convolution and dilated convolution. As a result, it can extract dense features under a shallow network which allows many operations to run in parallel rather than sequentially. To train DABNet, we first use a labeled dataset with images X_n and their corresponding one-hot encoded ground truth Y_n. The ground truth has the object classes that we wish to style. Next, we train DABNet to output the right prediction mask $S(X_n)$ by minimizing the following cross-entropy loss,

$$\mathcal{L}_{ce} - \sum_{h,w} \sum_{c \in C} \log \left(S(X_n)^{(h,w,c)} \right). \tag{2}$$

Finally, the network outputs the segmentation regions for object class c as a binary matrix R_c. Note that our framework is amenable to other real-time semantic segmentation methods.

3.3. Styling the Object classes

The final step of CBS is to generate an image U that has stylized object classes with an unstylized background. For an object class c and style S, the foreground mask (obtained by DABNet) is extracted for the stylized image (obtained by FSM) and then added to the background image. This is accomplished with the following element-wise product ($*$) and addition,

$$U(c, S) = (R_c * T_S) + (\mathbf{1} - R_c) * I. \tag{3}$$

4. Experiments

We perform experiments to evaluate the efficacy of our method in two main aspects. Namely, the quality of the generated styles for different object classes; and the speed of CBS for styling video frames.

In Figure 3 we show qualitative examples for a variety of style and content images. We see that the colors and textures of the style images successfully transferred to the object classes. In addition, the extracted segmentation masks correctly capture the objects of interest.

In our next benchmark, we ran CBS on Tesla P100 GPUs. Note that CBS was implemented in Pytorch [9]. For the Cityscapes [2] 1024×2048 frames, CBS is able to process them at a speed of 16 FPS or 60.83 ms / image, making it feasible to run localized style transfer in real-time. Further, DABNet achieves 70.1% Mean IoU on the CityScapes

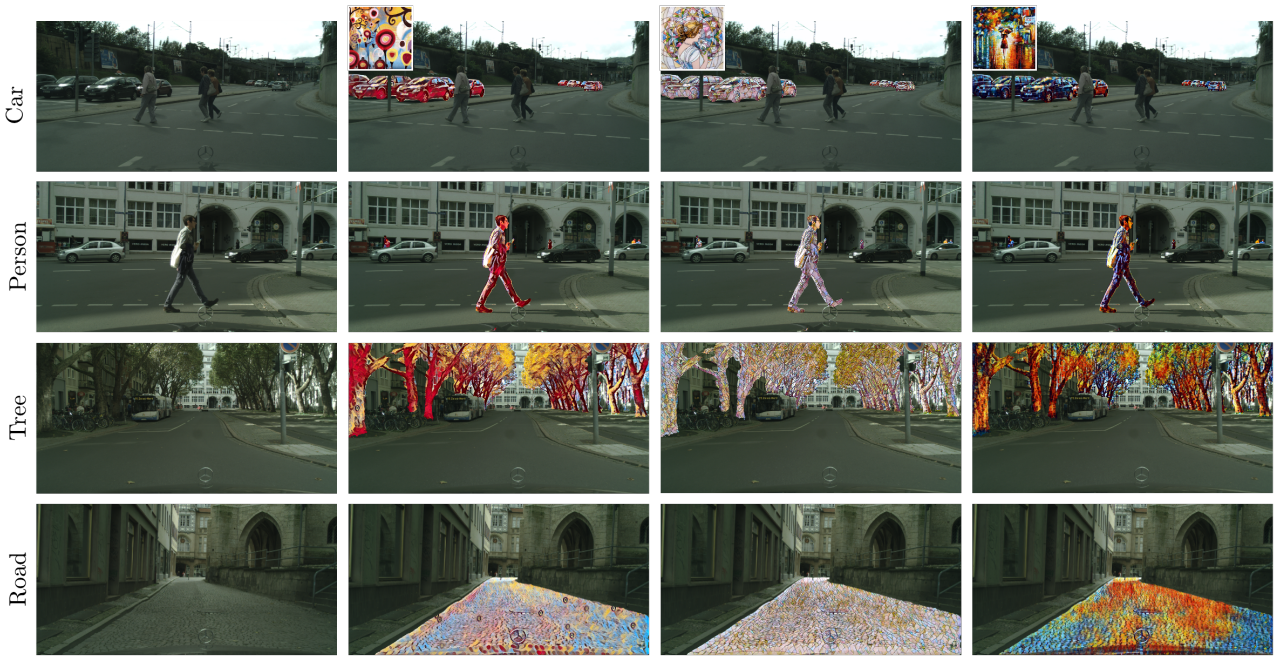

Figure 3: Qualitative results showing 3 different styles being applied to 4 different object classes.

test set allowing it to achieve quality segmentation, which, in turn, makes the localized style transfer look more appealing. In contrast, Castillo et al. [1] would take at least 15 seconds for a single image due to the Gatys et al. [3] method required at inference time. This makes it unsuitable for real-time style transfer. Thus, CBS can be used in more applications where artistic work requires real-time performance.

5. Conclusion

Our model achieves real-time localized style transfer by simultaneously segmenting and styling each frame. We anticipate that stylizing each class separately in real time can pave the way to more intricate videos, appealing works of arts, or in advertisements using product styling. For future work, we plan to extend this method for 3D scenes where localized style transfer can be applied to 3D objects.

References

[1] C. Castillo, S. De, X. Han, B. Singh, A. K. Yadav, and T. Goldstein. Son of zorn's lemma: Targeted style transfer using instance-aware semantic segmentation. In *ICASSP*, 2017.

[2] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. *CVPR*, 2016.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv:1508.06576*, 2015.

[4] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask r-cnn. *ICCV*, 2017.

[5] Y. Jing, Y. Yang, Z. Feng, J. Ye, and M. Song. Neural style transfer: A review. *T-VCG*, 2017.

[6] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song. Neural style transfer: A review. *arXiv:1705.04058*, 2017.

[7] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

[8] G. Li, I. Yun, J. Kim, and J. Kim. Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. *arXiv:1907.11357*, 2019.

[9] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.