

---

# Style transfer for rooms

## Neural Style Transfer/Image Reconstruction/Computer Vision

---

Warren Xia (waxia@stanford.edu) | Marie Chu (mariechu@stanford.edu)  
Department of Computer Science | Stanford University

### Abstract

It is costly to remodel a room and drafting designs for it can be time consuming. To better visualize a room before starting the work to remodel it, it would be helpful to apply well-known styles (modern, minimalistic, contemporary, etc.) to visualize the space and ensure it aligns with the owner's desires. We utilize an autoencoder architecture with block training, high frequency residual skip connections, and a bottleneck feature aggregation to achieve photorealistic style transfer of interior rooms.

Github repo: <https://github.com/mariemchu/PhotoNetWCT2.git>[3]  
WCT Baseline repo: <https://github.com/eridgd/WCT-TF>

## 1 Introduction

To give the owner of a home a better idea of how a room would look in their preferred style before remodeling, we aim to apply style transfer to interior rooms. Deep Neural Networks have been applied to object and face recognition. In 2015, Gatys et.al applied this to the realm of fine art [5]. They took natural images and stylized them with famous artworks by extracting the content representation and style representation of each image. In 2017 Li et al. introduced an autoencoder approach to the task [6]. We expanded these works to style transferring of rooms in a home. The input to our algorithm is an image of a room used as the content and another image of a room used as the style. Our final model uses an autoencoder approach, like that of Li et al. to output a generated image of the room with the new style.

## 2 Related Work

**Photorealistic Style Transfer with Autoencoders:** In this project we chose to use an auto encoder architecture. Early works with this method were first introduced by Li et al. in Universal Style Transfer via Feature Transforms. Although these methods had promising results, some produced poor stylization effects, required segmented images, or were too parameter heavy. We aim to address these issues with our proposed model that combines the approaches from An et al. [1] and Chiu and Gurari [2]. The approaches taken by each will be described in the methods section.

## 3 Dataset and Features

To train the autoencoder we will use the MSCOCO [4] dataset that was used in a few of the WCT papers. In the training set of this dataset there are 118,288 images, 5000 images in the validation set, 40670 and images in the test set. We also used the ADE20K [9] dataset for our initial semantic segmentation approach (described in experiments below) which has 25,574 training images and 2,000 images in the validation set. This dataset contains the semantic segmentation of each scene. However it also includes images of not only indoor rooms but also outdoor spaces, cities, factories, etc. After filtering out irrelevant images we were left with 6118 training images and 523 validation images. We used these images to produce final stylized results of different rooms. However, the segmentation of these images aren't perfect which we discuss in experiments below.



Figure 1: MSCOCO

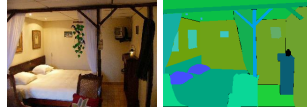


Figure 2: ADE20K

## 4 Method and literature review

Like Li et al. we used an autoencoder architecture to perform style transfer. We will discuss the method that our model performs style transfer through the previous papers it builds upon in this section.

Our final model is an autoencoder architecture using the baseline of WCT. The VGG-19 network is used as the feature extractor (encoder) and a symmetric decoder is trained on images from the MSCOCO dataset to invert the VGG-19 features and reconstruct the content image. The decoders are trained to minimize pixel reconstruction loss and feature loss:

$$L = \|I_o I_i\|_2^2 + \lambda \|\phi(I_o) - \phi(I_i)\|_2^2$$

After training, the encoder and decoder are fixed. To perform style transfer both the content and style images are fed into the encoder to extract the vectorized feature maps. WCT (whitening and coloring) transformations are then performed to make the featurized content match the covariance matrix of the featurized style. The whitening step maintains the global structure while the coloring step introduces the style from the style image. This is then decoded to reconstruct the image with the new style [6].

For our autoencoder we looked at related research in the field to guide our design process. Two other related models sought to improve upon the vanilla autoencoder by modifying the downsampling and upsampling layer. In PhotoWCT, it was replaced with pooling and unpooling to preserve spatial information, reduce artifacts and make the image more photorealistic. An additional post smoothing step was also introduced to remove artifacts using the original content image [7]. In WCT<sup>2</sup>, the pooling and unpooling layer in the VGG encoder and decoder was replaced with a wave corrected transfer that would perfectly reconstruct a signal without post processing steps. WCT<sup>2</sup> introduced skip connections as well as usage of segmented images to produce better results.[8].

An et al. then tackled the problem of the lack of style introduced in WCT<sup>2</sup>'s skip connections and the need for segmented images to produce photorealistic results. An autoencoder called PhotoNet was used with the pretrained VGG-19 as the encoder and a decoder that would reconstruct the image, similar to the variations of WCT from before. However, PhotoNet also introduces a bottleneck feature aggregation (BFA) module at the bottleneck which concatenates multi-scale features produced by different levels of the network. It does this by resizing features from *ReLU\_1\_1* to *ReLU\_4\_1* to the size of *ReLU\_5\_1* in the VGG encoder, and then concatenating them together at the bottleneck. Feature aggregation enables networks to integrate information from different fields-of-views, thus may enhance low-level detail preservation of stylization that happens in high-level features and lead to more details in the reconstructed image. Furthermore, An et al. replaced the skip connections in WCT<sup>2</sup> with Instance Normalized Skip Links (INSL) as WCT<sup>2</sup> generally lost its ability to produce stylized images since the short circuit could block the information stream flow into transfer module work at the bottleneck. INSL seems to alleviate the short circuit phenomenon and strengthen the detail preservation and distortion elimination abilities of photorealistic style transfer networks[1].

Chiu and Gurari introduced blockwise training to perform coarse-to-fine feature transformations in a single autoencoder instead of the cascade of four autoencoders used in PhotoWCT. The coarse-to-fine feature transformations was also an improvement from WCT<sup>2</sup> that had fine-to-course transformations which had weaker stylization strengths as the fine-tune details may have been overshadowed later by coarser big-picture modifications. Secondly, skip connections of high-frequency residuals were introduced in order to preserve image quality when applying the sequential coarse-to-fine feature transformations. This preserved the advantages of WCT<sup>2</sup>'s better image reconstruction with less parameters [2].

### 4.1 Our Method

Considering these approaches, we chose to make a model that combined the approaches in PhotoNas with that of PhotoWCT<sup>2</sup>. Specifically, we added bottleneck feature aggregation, BFA, from PhotoNas to PhotoWCT<sup>2</sup> to help further preserve details in the content image. We kept the high-frequency residuals from PhotoWCT<sup>2</sup> that allowed for better

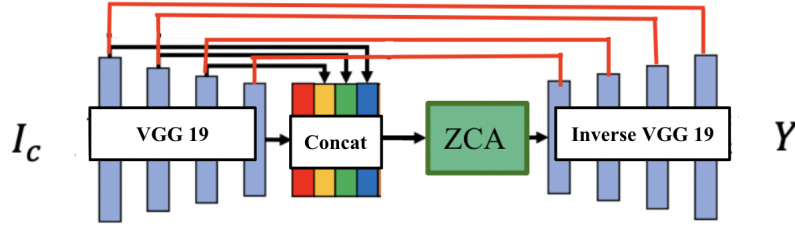


Figure 3: PhotoNetWCT<sup>2</sup>: Our autoencoder architecture with a BFA layer, ZCA transform (specific type of WCT transform) and high frequency residual skip connections

feature reconstruction and preserved the information with less parameters than WCT<sup>2</sup> and kept blockwise training which would also help minimize the feature reconstruction error. We ran our model against the baseline of WCT along with PhotoWCT<sup>2</sup> with different content and style images to compare our results to.

## 5 Experiments

### 5.1 NST with Multi-Label Semantic Segmentation Weights

Initially we had a completely different approach and followed Gatys' approach in "A Neural Algorithm of Artistic Style" by using layers of the existing CNN as representations of the image's style and content and then using gradient descent from a white noise image to optimize the loss function.

$$L_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha L_{content}(\vec{p}, \vec{x}) + L_{style}(\vec{a}, \vec{x})$$

Using VGG19 as our baseline model we applied Neural Style transfer to an interior room giving it a content and style image. The results showed distortions and noise in the image as this approach leads to a more "artistic" result and notably applied a very global style onto the image. We pivoted to trying to use pre segmented images however in this case not only were existing noise artifacts not fixed but some new artifacts were introduced as well. In some cases, the segmentation map was not precisely labeled leading to cases where errors in the segmentation map would cause parts of the image to be improperly styled. One example of these kinds of artifacts is the chandelier in **Figure 4**. Ultimately

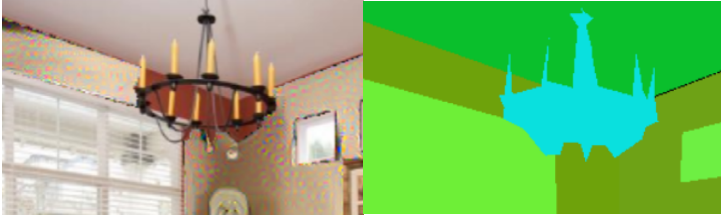


Figure 4: segmented style transfer    Figure 5: segmentation map

these downsides motivated us to pivoting to working on the autoencoder approach which would likely yield more photorealistic results and remove the potential problems of working with and needing segmentation maps.

### 5.2 Hyperparameters for the autoencoder method

The two main hyperparameters we tuned were the number of VGG layers we concatenated in our BFA, and the learning rate. We had three VGG layers we could concatenate so to test we ran three separate models appending different numbers of VGG layers and evaluated them against the image reconstruction loss and feature reconstruction loss terms defined in the original PhotoWCT<sup>2</sup> paper. We also tried generating images for these different models to compare outputs

Variation	Image Reconstruction Loss	Feature Reconstruction Loss	Total Loss
Concat Relu1-1,Relu2-1,Relu3-1	0.0012917289	0.16197614	0.1632679
Concat Relu1-1,Relu2-1	0.0013787546	0.18316413	0.1845428846
Concat Relu1-1	0.0018157327	0.18441695	0.1862326827

Overall all variations led to similar image outputs with only slight differences in lighting and shadows. We did find however that the image reconstruction loss and the feature reconstruction loss was the lowest for the variation that con-catted all three possible VGG layers. As a result we chose this variation as the one we trained the final model on.

After we determined which concatenation of layers led to the best results, we tuned the learning rate of the model by performing a log-scale random search to see which had the best loss convergence. We found that the learning rate used by the original PhotoWCT<sup>2</sup> produced good results and that other variations did not lead to significantly faster convergence. Ultimately we settled on a rate of 1e-4 to train our final model.

## 6 Results and Discussion

### 6.1 Image Outputs

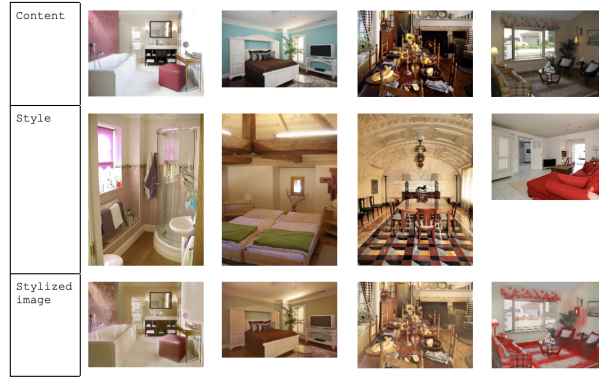


Figure 6: Results with three concatenated ReLu outputs in our BFA layer (see appendix for more results and larger images)

Compared to WCT, our results were significantly more realistic. As expected, the introduction of skip connections to the decoder from PhotoWCT<sup>2</sup>'s approach and BFA from PhotoNet gave the decoder significantly more information to work with over WCT and as a result, our method was able to reconstruct the image better and produce more photorealistic results. However compared to PhotoWCT<sup>2</sup>, the approach that just added blockwise training with high frequency skip connections, we found our approach was not significantly better. We suspect part of the reason why this was the case was because PhotoWCT<sup>2</sup> already had skip connections from the VGG encoder to the decoder. While it was theorized that feature aggregation might still be able to provide a unique benefit because it provided the information at the bottleneck rather than sending it directly to specific layers, it seems that in practice adding BFA didn't significantly increase the amount of information that the decoder had when stylizing the image as the skip connections already provided similar information. There were still some cases where our model was able to perform slightly better than PhotoWCT<sup>2</sup>. Notably as seen in the appendix image of the red living room, our approach is sometimes more conservative in altering parts of the image and actually correctly doesn't choose to stylize parts of the image that PhotoWCT<sup>2</sup> would. These cases are few and far between. Our method overall though did not perform any worse than PhotoWCT<sup>2</sup> in image output in the situations tested, showing that BFA is fully compatible with the blockwise training approach.

### 6.2 Metrics

Metric	PhotoWCT2	Ours
Image Reconstruction Loss	0.0006619	0.0012997
Feature Reconstruction Loss	0.0551108	0.1576392

We also compared our approach to PhotoWCT<sup>2</sup> with two metrics used in the original PhotoWCT<sup>2</sup> paper, image reconstruction loss and feature reconstruction loss. The reconstruction loss is defined as the pixelwise L2 distance





Figure 7: Results with baseline and related works (see appendix for larger images)

between an input image, and the reconstructed image after it is run through the encoder and reconstructed with the decoder,  $\|I - I_{rec}\|_2^2 / HWC$  where  $HWC$  are height width and number of channels. The feature reconstruction loss is the L2 loss between a feature  $F_n$  from a given layer  $reluN - 1$  to the corresponding decoder block feature  $F_{N,r}$ . This is expressed as  $\frac{\|F_n - F_{N,r}\|_2^2}{\|F_n\|_2^2}$ . Overall, our losses don't deviate too much from PhotoWCT<sup>2</sup> and at least empirically by looking at the model image outputs, there isn't as much of a difference. One reason that could explain the difference in losses is the training time. The PhotoWCT2 model didn't need to train additional parameters associated with our BFA layer and likely had more training time then we did for this report. It is fairly likely then that given more training time the difference in these metrics would shrink.

## 7 Conclusion and Future Work

Our paper demonstrates the compatibility of the BFA approach with blockwise training and our approach is partially successful at stylizing some components of rooms. Ultimately we have demonstrated that adding BFA alone to PhotoWCT<sup>2</sup> approach doesn't yield significant benefits, likely because the information given is redundant given the existence of skip connections. A major challenge for the project was the training time for the auto encoder architecture which we would've liked to train for more epochs in our experiments if given more computational resources. We also considered a GANs approach to this problem that we were unable to explore due to time constraints. However, this approach would've been more promising as at the core of it, we would like to be able to generate, add, remove, or change existing furniture into the new style the user proposes. With style transfer this isn't possible as the style we introduce cannot actually modify the structure of the underlying furniture it is being applied to.

## 8 Contributions

Warren Xia: running original baseline of Gatys NST for project proposal, running WCT baseline, conducting initial NST tests with segmented images, researching and debugging for final BFA approach, results analysis (presentation and report)

Marie Chu: research and write up for related works and method used. final implementation of BFA into blockwise trained autoencoder with high residual skip connections, experiments with different layers of features aggregated for BFA, slide deck creation for video (overview autoencoder method, and dataset), AWS instance setups, running results for WCT<sup>2</sup> and final results.

## 9 Appendix

### References

- [1] Jie An, Haoyi Xiong, Jun Huan, and Jiebo Luo. Ultrafast photorealistic style transfer via neural architecture search. 2020.
- [2] Tai-Yin Chiu and Danna Gurari. Photowct2: Compact autoencoder for photorealistic style transfer resulting from blockwise training and skip connections of high-frequency residuals. 2021.
- [3] Chui and Gurari. Photowct2.
- [4] Lin et al. Coco: common objects in context.
- [5] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. 2015.
- [6] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin lu, and Ming-Hsuan Yang. Universal style transfer via feature transformations. 2017.
- [7] Fujun Luan, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. 2018.
- [8] Jaesun Yoo, Youngjung Uh, Sandhyuk Chun, ByeongKyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. 2019.
- [9] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. Available at <https://groups.csail.mit.edu/vision/datasets/ADE20K/> (2017).

Content				
Style				
WCT (baseline)				
Photo WCT^2				
WCT^2 (w/ segmentation)				
Ours				

Figure 8: Larger image of results









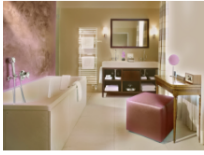
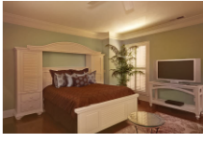






Figure 9: PhotoWCT<sup>2</sup> output



Figure 10: Our output

Content				
Style				
Stylized image				

Content	Style	Result			
					
					
					
					
					

Figure 11: more images with our approach