Learning with Image Style Transfer

Yiqi Yuan yy3754@nyu.edu
Courant
New York University
New York City, NY 10003, USA
Ziyi Huang zh1488@nyu.edu
Courant
New York University
New York City, NY 10003, USA

July 31, 2021

Abstract

This paper discusses the universal style transfer model by Gatys[GEB16], an image style transfer model using Convolutional Neural Networks. We first experiment with the model by using each single layer in the Convolutional Neural Network at a time to see the performance of each style or content layer. In order to optimize the model, we use the Learned Perceptual Image Patch Similarity method as a quantitative metric to evaluate the model results. We also replace the loss network by perceptual loss to improve the quality of the output.

1 Introduction

Style transfer is a computer vision technique to recompose the style feature of a source content image to another extracted from a source style image. In other words, the style transfer is to blend the source style image and source content image together such that the target image is transformed to look like the content image but "painted" in the style of the style image. To synthesize and preserve the features from both style and content image, it defines two distance functions, one that indicates the content difference between the content image and the target image, one that indicates the style difference between the style image and the target image. The goal of the transferring process is to minimize both the content distance and style distance.

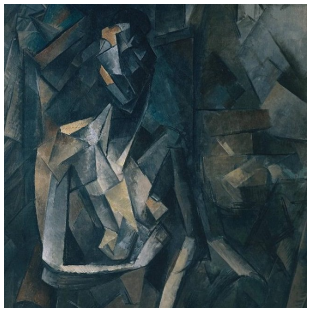
Our project is based on the universal style transfer model by Gatys[GEB16], and we try to make some improvements for that model (1) for those similar output images, there is no quantitative metric to evaluate them; (2) we think that the L2 loss is not the best choice in this model, because it is too sensitively affected by the intensity or gray scale changes; (3) because of the iterative approach, this model runs very slow.

2 Background/Related Work

For the universal style transfer model by Gatys[GEB16], there are three inputs: a source content image, a source style image, and an input image, which is initialized by a clone of the source content image. A normalized version of the 16 convolutional and five pooling layers of the 19-layer VGG layer is trained to perform object recognition and localisation. The given input image is encoded in each layer of this Convolutional Neural Network by the filter responses to that image to obtain the content representation. The content loss is the squared-error loss between two feature representations:

$$\mathcal{L}_{content}(\vec{\boldsymbol{p}}, \vec{\boldsymbol{x}}, l) = \frac{1}{2}\sum_{i,j}(F_{ij}^{l} - P_{ij}^{l})^2$$

where P^l is used to store the content representation in a layer l when the content image is passed through the network; the matrix $F^l \in R^{N_l \times M_l}$ is used to store the content representation when



(a) Style Image

(b) Content Image

Figure 1: (a) is Femme nue assise by Pablo Picasso, 1910. We use these two images as the source style image and source content image in the Experiments part.

a random white noise image is passed through the network. A layer with N_l distinct filters has N_l feature maps each of size M_l, where M_l is the height times the width of the feature map. To obtain the style representation of an input image, we use a feature space designed to capture texture information. These feature correlations between different filter responses are given by the Gram matrix $G^l \in R^{N_l \times N_l}$, where G^l_{ij} is the inner product between vectorized feature maps i and j in layer l: $G^l_{ij} = \sum_k (F^l_{ik} - F^l_{jk})$ In order to match the style representation to a given input image, it uses gradient descent from a white noise image to minimize the mean-squared distance between the entries of the Gram matrices from the original image and the Gram matrices of the output image. The element-wise-mean squared difference between G^l and A^l is used to compute the style loss: $E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - A^l_{ij})^2$. The total style loss is:

$$\mathcal{L}_{style}(\vec{\boldsymbol{a}}, \vec{\boldsymbol{x}}) = \sum_{l=1}^{L} w_l E_l$$

The total loss function is

$$\mathcal{L}_{total}(\vec{\boldsymbol{p}}, \vec{\boldsymbol{a}}, \vec{\boldsymbol{x}}) = \alpha \mathcal{L}_{content}(\vec{\boldsymbol{p}}, \vec{\boldsymbol{x}}) + \beta \mathcal{L}_{style}(\vec{\boldsymbol{a}}, \vec{\boldsymbol{x}})$$

The gradient with respect to the pixel value $\vec{\boldsymbol{x}}$ is $\frac{\partial \mathcal{L}_{total}}{\partial \vec{\boldsymbol{x}}}$. Change the ratio α/β, model can achieve a trade-off between the content matching and style matching.

We also reviewed another popular style transfer paper written by Johnson and Li. This paper introduces the Faster Style Transfer model by Johnson and Li[JAL16]. Faster Style Transfer model by Johnson and Li applies an Image Transformation Network, a deep residual Convolutional Neural Network, which is trained to solve the optimization problem proposed by Gatys. The loss network in this model is a pretrained VGG-16 on the ImageNet Dataset. We think the perceptual loss is a worthy idea, but there is a shortcoming that this model is limited by styles at test-time. If we would like to replace the style, the model needs to re-train the image transfer net, this process takes a long time. In sum, Gatys' iterative optimizer is limited by speed, Faster Style Transfer model optimizes the speed but limited by the styles. Consequently, we make more efforts to combine the essences from different models, and try to achieve the model optimization.

3 Experiments

Our experiments start with plotting style loss using each layer in the Convolutional Neural Network with five pooling layers at a time. We name these five pooling layers as conv1, conv2, conv3, conv4, conv5. First, we use the outputs of these five pooling layers for calculating the style loss. We set the content weight to zero, initialize the input image to white noise image, and generate the style images using each of these layers. The style images which are generated should correspond to the pastiches of the painting. The outputs are shown below:

For the output from earlier layers (conv1 and conv2), the patterns in the style image which require smaller receptive fields are prominent. In the later layers (conv3 and conv4) bigger patterns are more

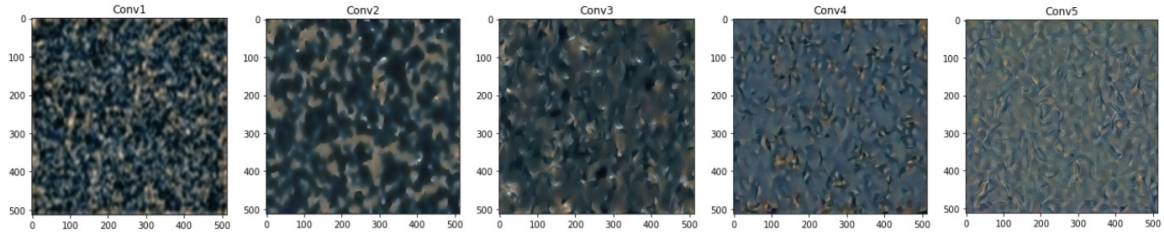


Figure 2

prominent. And the last layer (conv5) is blurry and has less feature of the original style image. The style experiments show that the conv3 layer will preserve more features of the original style image, so we pick the conv3 layer in the next experiments.

Afterwards, we experiment with plotting content loss, also using one layer in the Convolutional Neural Network with five pooling layers each time. We keep using the style layer conv3, and attempt five content layers one by one to compute content loss. We name these layer combinations as conv3_1, conv3_2, conv3_3, conv3_4, conv3_5. The input image is initialized to white noise image. The outputs are shown below:

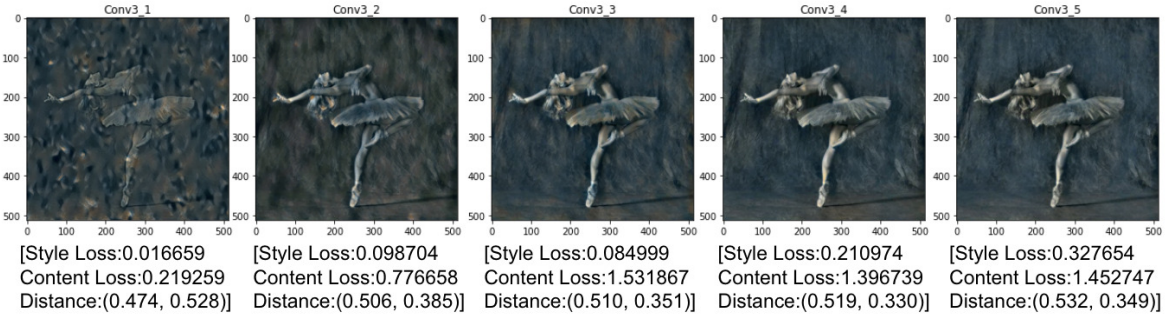


Figure 3

Clearly, the combination of conv3_1 and conv3_2 are badly observed by naked eye. Then we compare the content loss and the Distance among the three outputs: conv3_3, conv3_4 and conv3_5. The combination of style layer 3 and content layer 4 (conv3_4) generates the best performance because it generates the least style Distance value (The definition of Distance is discussed in section 4.1).

4 Optimization

4.1 Model Evaluation

To solve the no quantitative evaluation problem we mentioned above, Classic per-pixel measures, such as l2 Euclidean distance, commonly used for regression problems, or the related Peak Signal-to-Noise Ratio (PSNR), are insufficient for assessing structured outputs such as images, as they assume pixel-wise independence. All of the three architectures (SqueezeNet, AlexNet, and VGG) in Learned Perceptual Image Patch Similarity(LPIPS) method perform better than traditional perceptual distance metric, such as l_2, SSIM, and FSIM in the human perceptual judgment experiment[ZIE+18]. Therefore, LPIPS could be used to compare the difference between the output image and the source images. We use the VGG architecture in our model because the higher capacity network VGG always performs better than the lower capacity network SqueezeNet and AlexNet.

The PerceptualSimilarity based on LPIPS is illustrated in the figure that shows how the distance and distorted patches x, x_0 is obtained with network $\mathcal{F}$.

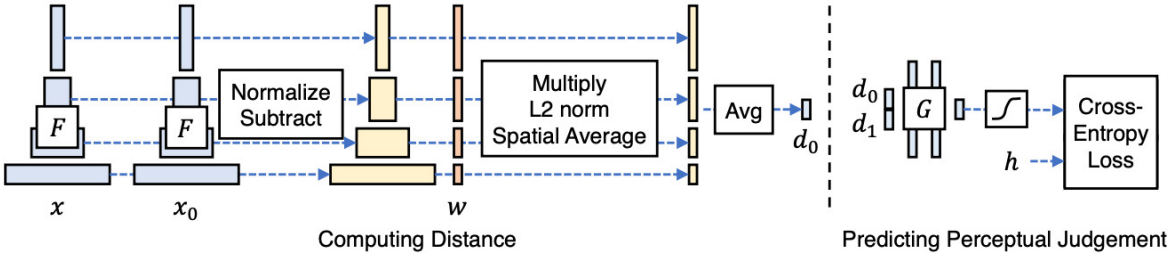


Figure 4

In the Figure 4 (left) and equation illustrate how distance is obtained between reference and distorted patches x, x_0 with network $\mathcal{F}$. The feature stack is extracted from L layers and unit-normalize in the channel dimension, which designate as $\hat{y}^l$, $\hat{y}^l_0 \in R^{H_l \times W_l \times C_l}$ for layer l. Then scale the activations channel-wise by vector $w^l \in R^{C_l}$ and computing the l_2 distance. Finally, average spatially and sum channel-wise. Note that using $w_l = 1 \forall l$ is equivalent to computing cosine distance[ZIE+18].

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||w_l \odot (\hat{y}^l_{hw} - \hat{y}^l_{0hw})||_2^2$$

The Distance is used to measure the difference between the output image and the source style, content images. The first variable in Distance indicates the style difference, the second variable indicates the content difference. Larger value means more difference.

4.2 Perceptual Loss

Perceptual loss function works by summing all the squared errors between all the pixels and taking the mean, which is in contrast to a per-pixel loss function that sums all the absolute errors between pixels[GEB16]. In the Gatys' model, the loss function is a per-pixel loss function. L2 loss is badly affected if the image shifted, even only be shifted by one pixel, but perceptual loss is not. Comparing with L2 loss, perceptual loss is insensitive by the intensity differences of the images, which is more similar to human visual perception[ZIE+18]. L2 loss and perceptual loss have similar sensitivity of contrast. In order to verify our viewpoints, we make two simple experiments.

4.2.1 Intensity

The first experiment aims to see how the intensity affects the L2 loss and perceptual loss. A white image, a black image and a gray image are applied, we calculate the L2 loss and perceptual loss between every two images. The results are shown as below:

	(White, Black)	(White, Gray)	(Gray, Black)
L2 Loss	4	1	1
Perceptual Loss	0.4538	0.4945	0.5390

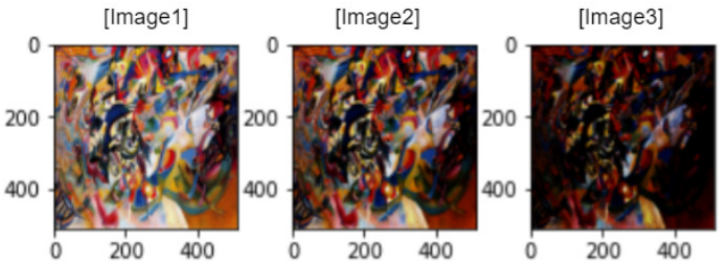
Figure 5: Intensity Result Table.

It is obvious that the L2 loss is affected by the intensity a lot, but perceptual loss is not susceptible.

4.2.2 Contrast

The second experiment aims to show the influence of contrast. We adjust the contrast of the input image to three levels, then also output the L2 loss and perceptual loss:

The L2 loss between Image1 and Image3 is about 4 times the L2 loss between Image1 and Image 2. Similarity, The perceptual loss between Image1 and Image3 is about 3.6 times the perceptual loss



	(Image1, Image2)	(Image1, Image3)
L2 Loss	0.0341	0.1367
Perceptual Loss	0.1251	0.4468

Figure 6: Contrast Result Table.

between Image1 and Image 2. In short, the contrast causes similar influences for L2 loss and perceptual loss.

The idea of using perceptual loss is from the Faster Style Transfer model by Johnson & Li[JAL16], but in order to avoid long time re-train image transfer net problem, we combine the idea of perceptual loss and Gatys' model. We keep the style loss part from Gatys' model, but replace the content loss network by Perceptual Loss network. LPIPS method is used to achieve the perceptual loss computation.

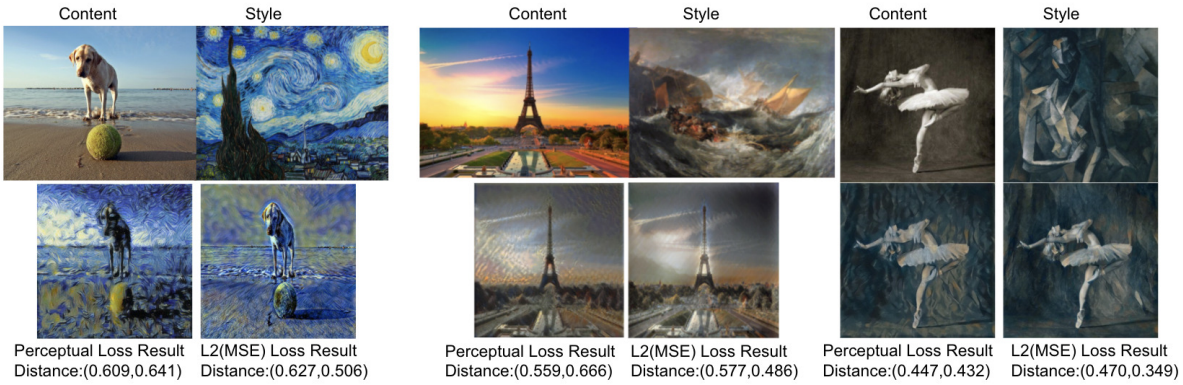


Figure 7: Comparison of the output images by Universal Style Transfer model by Gatys and by our optimized model

From these outputs values (Figure 7), We find that the style Distance value of the model using perceptual loss is always smaller than the style Distance value of the model using L2 loss. Through comparing these output images (Figure 7), we find that the L2 model's outputs have blurry background, but the Perceptual loss model have better performance in preserving style textures. It avoids the model rely more on the content image.

5 Conclusion

In this paper, we analyze the universal style transfer model by Gatys. On this basis, we make some improvements to the model. We apply the LPIPS method to resolve the model evaluation problem. The distance computed by LPIPS method could quantitatively indicate the difference in style and content between similar images. In addition, we modify the loss network for this model. The perceptual loss function is more commonly used as it often provides the results closer to human perception. Our model better preserves both style and content features. Besides, our model improves the speed but is not limited by styles.

In future work, we would consider applying the whitening and coloring transform method into our model to achieve the style photorealistic[LFY+17].

Code on GitHub see GitHub Link.

References

[GEB16] 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 (CVPR)*, June 2016.

[JAL16] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016.

[LFY+17] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *CoRR*, abs/1705.08086, 2017.

[ZIE+18] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.