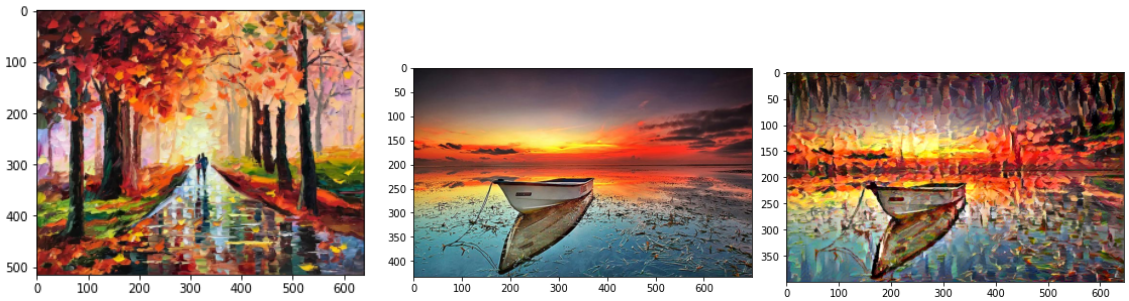
Understanding Neural Style Transfer

Khanh Vong

May 20, 2020

1 Introduction

Neural Style Transfer (NST) was first introduced in "A Neural Algorithm of Artistic Style" in 2015. Since then, researchers has develop better application of the findings to allows for NST to be applied to videos as well as real-time applications. To understand how NST has developed over the years, we must understand what NST originally was and how it was developed. This paper will discuss, the inner workings of NST when it was first introduced, and how it is able to combine the style, and the content of two different images into one so effectively. Figure 1 shows a style image, a content image, and the resulting generated image produced by style transfer.



(a) Style Image (b) Content Image (c) Generated Image

Figure 1: Neural Style Transfer In Action

2 Style And Content

2.1 Content

NST works by taking advantage of pretrained Convolutional Neural Networks (CNN) trained on large image dataset. The heirarchical structure of CNN, is that low-level features are detected on the lower layers, and higher-level features are detected on higher layers. On the higher levels, the CNN layer captures the *content* of the image. Thus, the content image can be fed to a high-level layer to extract the image's content. The bottom of Figure 2 shows what happens when we feed the content image from low to high layers of the CNN. Observe that the image replicates almost perfectly on the lower layers, perserving the visually exact pixel values; where the higher level layers preserves the higher level of abstraction while losing the pixel information. For the content, we are only interested in the higher level of the CNN. This will help preserves the content image's subject. In Figure 2 the content is the house.

2.2 Style

In order to capture the style of an image, we must find the correlation between multiple layers. Doing so we are able to extract the texture of an image. From the Figure 2 we can see at the top

of the figure is the different styles reproduced by the CNN on different layers. The different style output is a result of the number of layers to produce the output. For example, **a** is produced by the first CNN layer, **b** is produced by the first two layers, **c** is produced by the first three layers, and so on. From the output styles, we can see that by using more layers, we are able to produce stronger style of the style image.

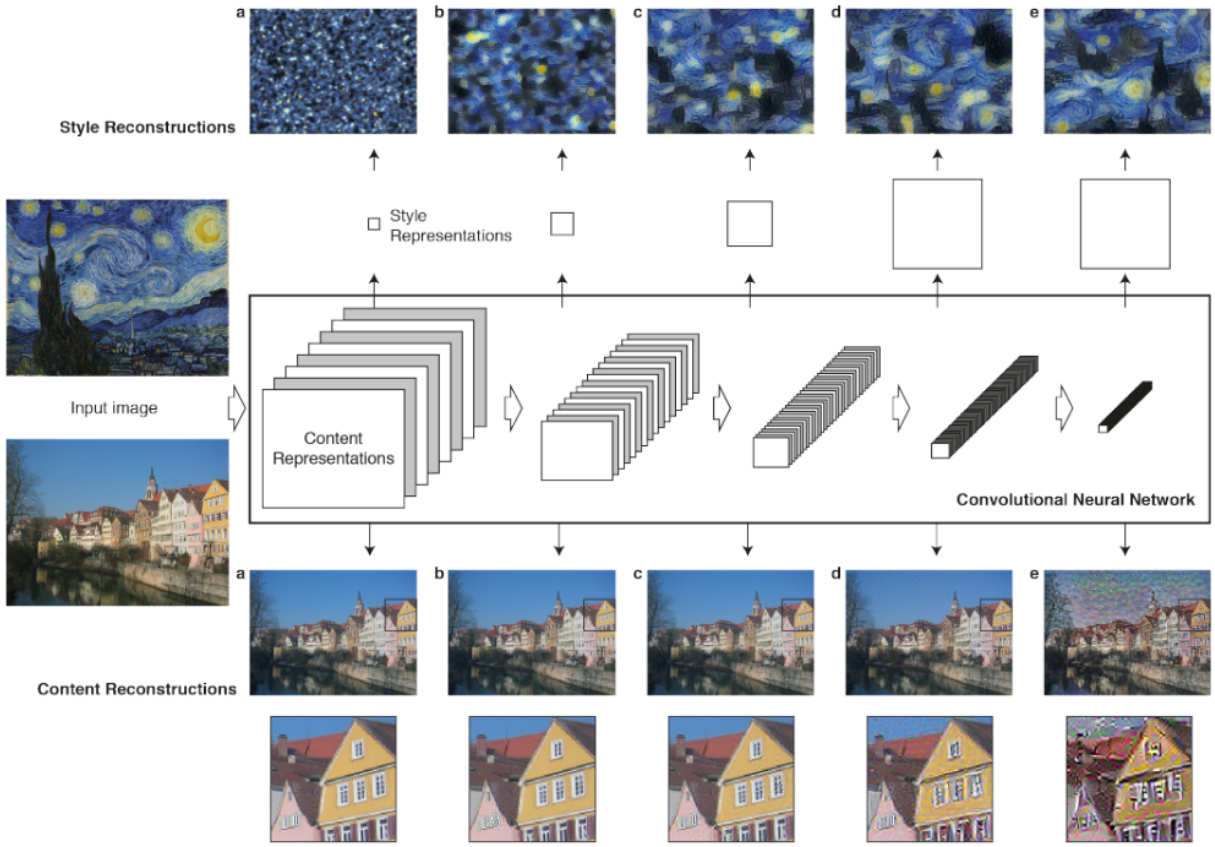


Figure 2: **Convolutional Neural Network** producing the style and the content from input images

3 Applying Loss Functions

For every learning problem there is a loss - which is a way for the model to improve during training. In this case we are considering three factors, content, style, and total variation. That is, we are attempting to preserve as much content, and style as possible in the output image. The final total variation loss is to ensure local coherence in the image.

3.1 Content Loss

We use the content loss to preserve the content of the image. The content defined in 2.1 preserves the higher level features at a higher layer. By feeding the base image and generated image through the higher CNN layer, and comparing the different activations, we are able to measure the content loss. In other words, we are able to measure whether the generated picture is preserving the content of the image.

3.2 Style Loss

Along with the content, our other goal is to preserve the style. However, unlike how we measure the the content loss, where we simply feed the images to a single layer, measuring the style loss requires

checking the differences in style multiple layers. This is because to incorporate the strongest style, we took a look at the style image on a single filter of every layers. Thus, we have to check the corresponding single filter of every layers for the generated images as well.

To get the style loss of the generated image from the style image, we will check each image's **gram matrix** - the *dot product* of a vector. More intuively, the dot product tell us how the vectors correlates. For our case we will find the dot product of our flatten image (a vector) and its transpose to produce a matrix to identify different correlations between filters. We will do this in all the layers used to determine the style. Our loss will check the correlations activations of the generated image with the style correlations by comparing their gram matrix.

3.2.1 Gram Matrix Operation

Suppose we have 2 filters z_0 and z_1, applied to vector $\vec{x}$ with 3 pixels:

$$F = \begin{array}{|c|c|c|c|} \hline & x_0 & x_1 & x_2 \\ \hline z_0 & z_0(x_0) & z_0(x_1) & z_0(x_2) \\ \hline z_1 & z_1(x_0) & z_1(x_1) & z_1(x_2) \\ \hline \end{array}$$

$$F_{ik} = Z_i(x_k)$$

Gram Matrix:

$$G_{ij} = \sum F_{ik} * F_{jk}$$

$$GramMatrix = \begin{array}{|c|c|l|l|} \hline & & \multicolumn{2}{c|}{\text{j}} \\ \hline & & 0 & 1 \\ \hline \text{i} & 0 & F_{00} * F_{00} + F_{01} * F_{10} + F_{02} * F_{02} & F_{00} * F_{10} + F_{01} * F_{11} + F_{02} * F_{12} \\ & 1 & F_{10} * F_{00} + F_{11} * F_{01} + F_{12} * F_{02} & F_{10} * F_{10} + F_{11} * F_{11} + F_{12} * F_{12} \\ \hline \end{array}$$

Higher values translate to higher corelation between filters. Intuitively, the highest values will lie diagonally down the matrix because a single filter correlates with itself perfectly.

3.3 Total Variation Loss

The final loss function we will add is the total variation loss. We are adding this loss in order to keep the local coherency of the image. This will also prevent pixilated looking images.

3.4 What We Are Minimizing

We will minimize all three losses. With the use of hyperparameters we can tweak the loss in according to out need. For example, increasing the hyperparameter for the style will generate an image that emphasizes the style over the content.

4 Applying Neural Style Transfer

Neural Style Transfer is done by incorporating a pretrained model (the original paper uses VGG19 trained on ImageNet), because pretrained models trained on many objects will effectively be able to extract the style and content of the images. To start creating images, we simply use gradient descent on the losses. Figure 3 shows the different results that were generated.

5 Conclusion

By observing neural style transfer we conclude that style, and content can be separated using CNN. This is an interesting discovery. After it was introduced in 2015, many people have improved upon it. One of discoveries include, faster NST, allowing for realtime video NST.

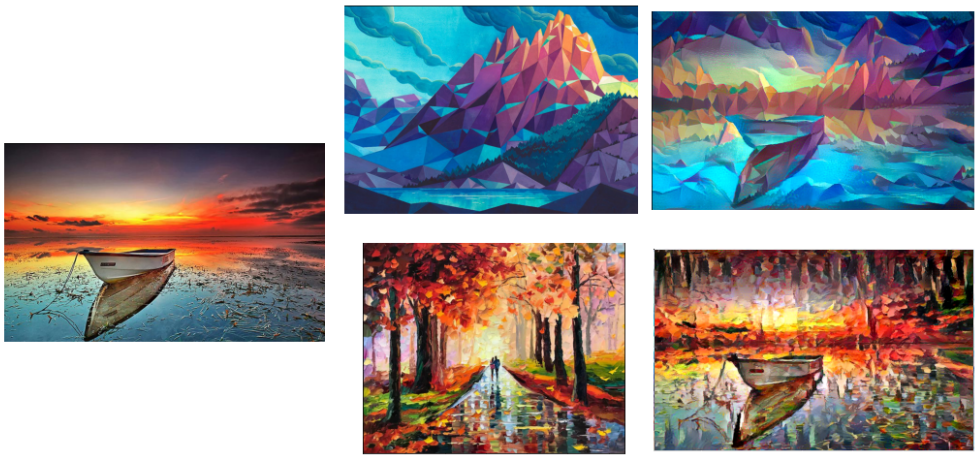


Figure 3: Generated Styles

6 References

- A Neural Algorithm of Artistic Style - Leon A. Gatys, Alexander S. Ecker, Matthias Bethge
- https://keras.io/examples/neural_style_transfer/
- https://towardsdatascience.com/neural-networks-intuitions-2-dot-product-gram-matrix-and-neural-style-transfer-5d39653e7916