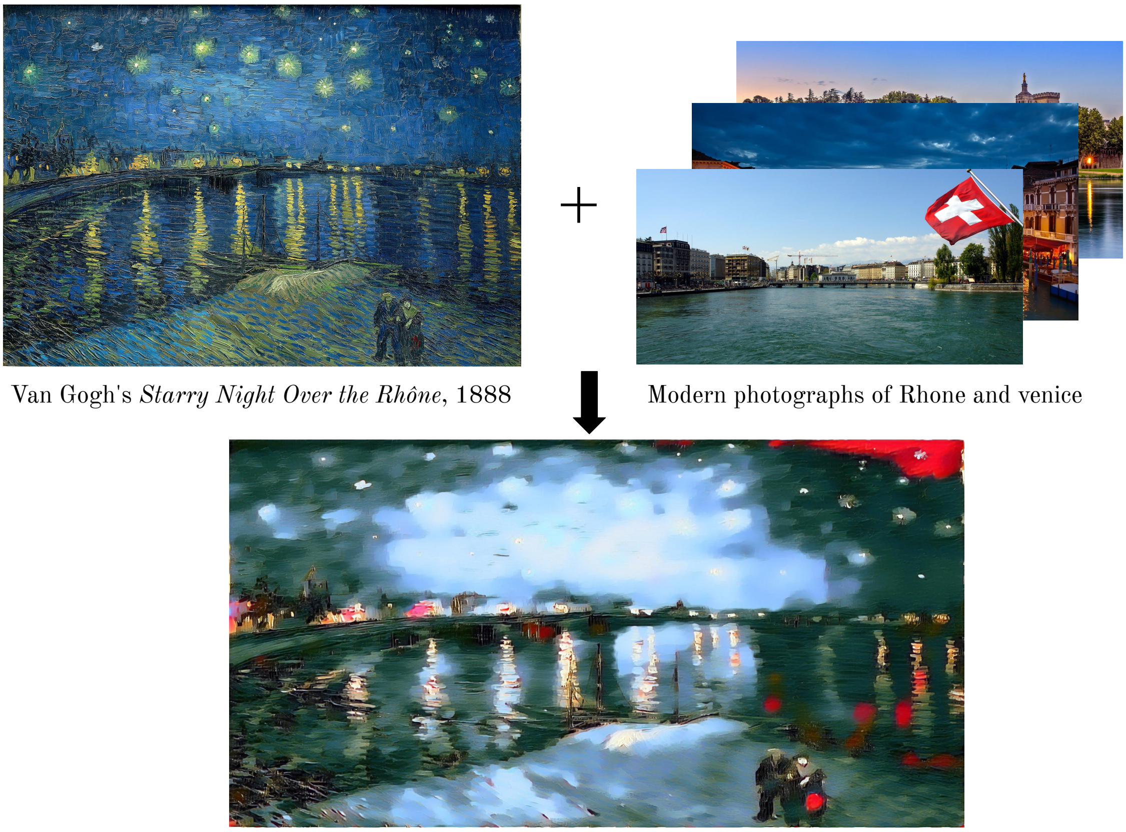
Zindi AI Art Submission

Matthew Baas — Zindi username: Baas

July 5, 2019



Van Gogh's *Starry Night Over the Rhône*, 1888

Modern photographs of Rhone and venice

Figure 1: Overview of method and results. An input stylized painting (top left) is combined with several modern photographs (top right) to produce the modern, less-stylized equivalent of the input painting (bottom).

1 Artist Statement

Most methods for making visual art with artificial neural networks involve style transfer, where usually a random input image is trained to minimize a *style loss* and a *content loss* in order for the image to eventually have the style of one image and the content of another.

I wanted to try and swap this usual order around and see if, given as input a stylistic painting (like the *Starry Night* painting in Fig 1) as a *content* image, can we remove the artist's unique style and leave the fundamental content of the painting behind? That is what I set out to do, and the main result is the image shown in the bottom of Fig 1. The full-resolution output image is shown on the last page and is also available at this link.

1.1 Method

Typically in neural style transfer, a pretrained neural network (VGG19 for this work) is frozen, and an input image is optimized with respect to both a *style loss* and a *content loss*. How these are formulated vary a lot depending on the project, but usually MSE losses, perceptual losses, and losses using gram matrices work well.

For my work, the order is flipped, and the stylistic painting is the content image, and the style image is frequently changed during optimization to one of several photographs of modern buildings near calm water, namely photographs of Rhone and Venice. The process I followed is shown in Fig 2.

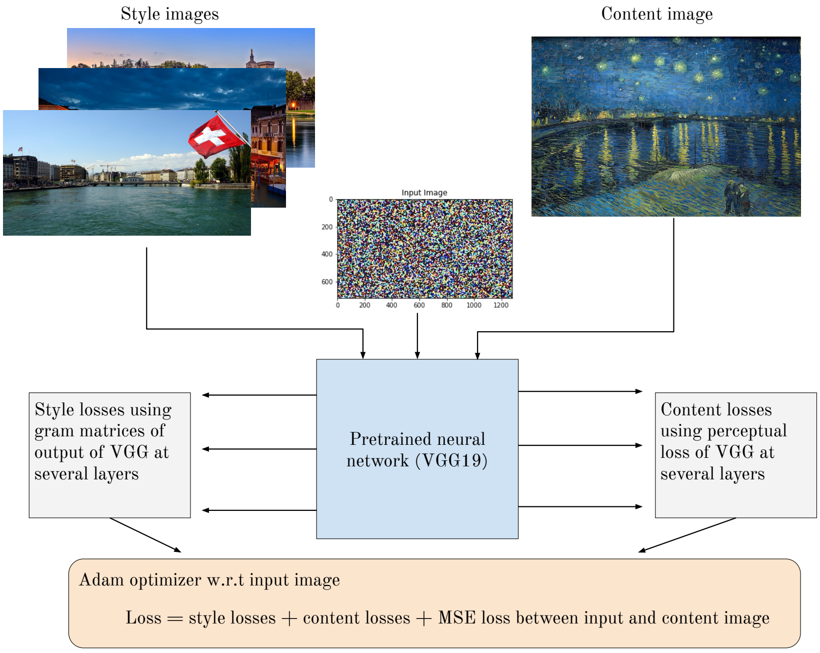

Style images
Content image
Input Image
0
200
400
600
0
200
400
600
800
1000
1200
Style losses using gram matrices of output of VGG at several layers
Pretrained neural network (VGG19)
Content losses using perceptual loss of VGG at several layers
Adam optimizer w.r.t input image
Loss = style losses + content losses + MSE loss between input and content image

Figure 2: Training method. The optimizer changes the pixel values of the input image using a loss function made from the style and content images.

Using this method, I tried to capture the "average style of modern photographs" and avoid the output image latching on to particular pixel styles in any of the individual style images (the photographs). The neural network I used was the PyTorch `VGG19` model pretrained on ImageNet. The style loss was the MSE loss between the gram matrices of the output of the network at the layers just before each pooling layer. Similarly the content loss was the MSE loss of the output of the network at the `Conv2D` layers before each pooling layer in `VGG19` (AKA perceptual/feature loss).

The loss also had a term that was just the MSE loss between the pixels of the input image and the content image, since I found this helped it latch on to the main features of the content image better :). I call this loss component the *pixel loss*. Pytorch was the main library I used in this project.

1.2 Getting things just right

Just doing the above and training for a while does not produce such great outputs. Most of them either slightly reduced how brush-stroke the original artwork looked (like in Fig 3), or they spiraled out of control and became somewhat hideous.

To make the output work and be aesthetically pleasing, each term of the loss function had a particular weight assigned to it. i.e the loss was the *weighted* sum of the style, content, and pixel losses from different layers. To make the training even more expressive, I changed the weighting of each term at different stages of training, which better allowed me to guide the formation of the output image.

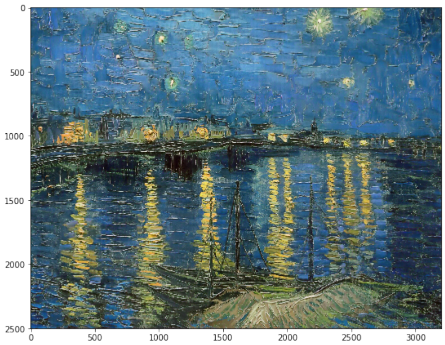


Figure 3: One of the failed results – it is too similar to the original, with the only big difference is the paint strokes have become more horizontal.

I started training with large weights on the pixel and content losses, and not too much on the style loss (so that the random noise image first latched on to the general features of the original Van Gogh image). Then after the image appeared to have the general fuzzy features of the original Van Gogh image, I decreased the pixel MSE loss substantially and slightly increased the style loss. Then after training for some more, I made the content loss weighting zero for all except the last `Conv2D` layer where the content and style loss are sampled from; and I further increased the style weighting for the middle layers of the VGG network.

After each little session of training, I would judge the output and see which layers of the network I should increase/decrease the content/style losses of for the next bit of training. In this way I incrementally formed the image toward the final output. Of particular interest is that the reflection of the brightened sky in the output is well represented in the water. The very hard brush strokes of the original painting are also greatly reduced in the output, as shown in Fig 4. Finally, **no editing or post-processing** was done to any of the images. The only input pre-processing done was the re-sizing of the Van Gogh and photographs to a common size.

2 Final output

The final artwork has a resolution 2048x1080 pixels, however it is best viewed slightly smaller. The full jpg is available also available at at this link.

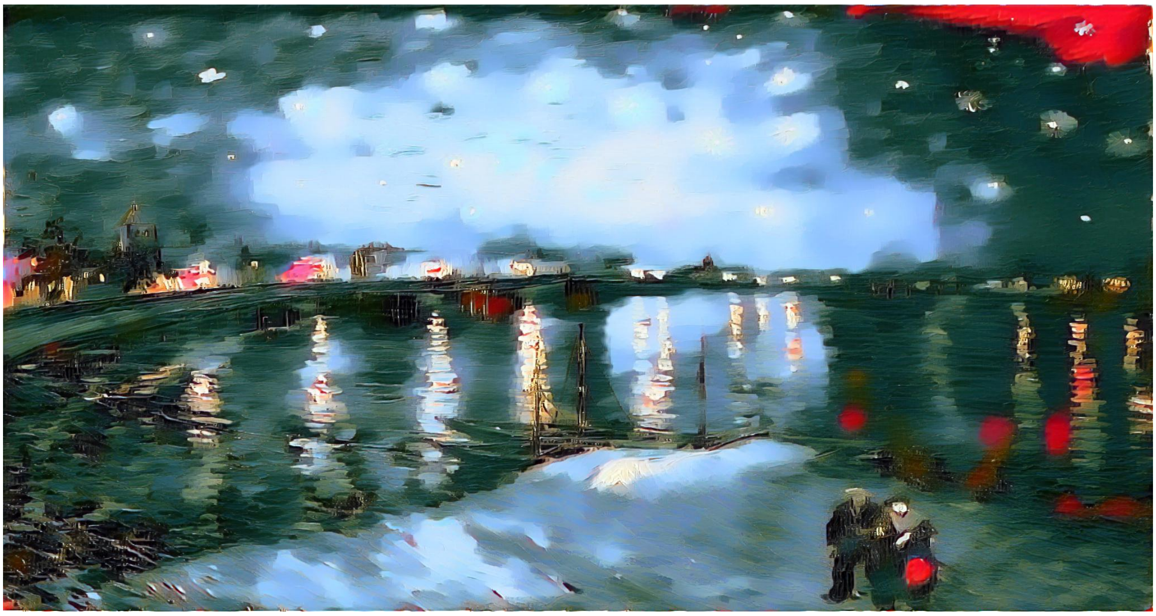


Figure 4: *Rhone modern* ∼ final output artwork.