MRP - Week 5

This week we will be looking at total variation loss and how it affects the results.

- The loss function allows us to control the amount of content and style we wish to inherit

$$L_{total}(S, C, G) = \alpha L_{content}(C, G) + \beta L_{style}(S, G)$$

A higher content/style ratio will yield an output image more representative of the original target image, while the opposite will yield an output image with stronger stylistic features.

- Once you have both content and style loss add them up and use any optimizer to perform gradient descent to change generated image such that it decreases its loss after each iteration

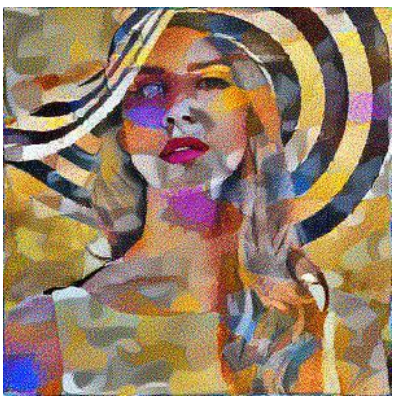
- The results generate with just L_content and L_style were unappealing and pixilated (see below)

Content Image

Style Images



Generated Images


- The generated images are ‘grainy’ due to the noise

- To reduce this, we can add an additional term called “total variation loss” to the loss function
- The function totals the sum of the absolute differences for neighboring pixel values.

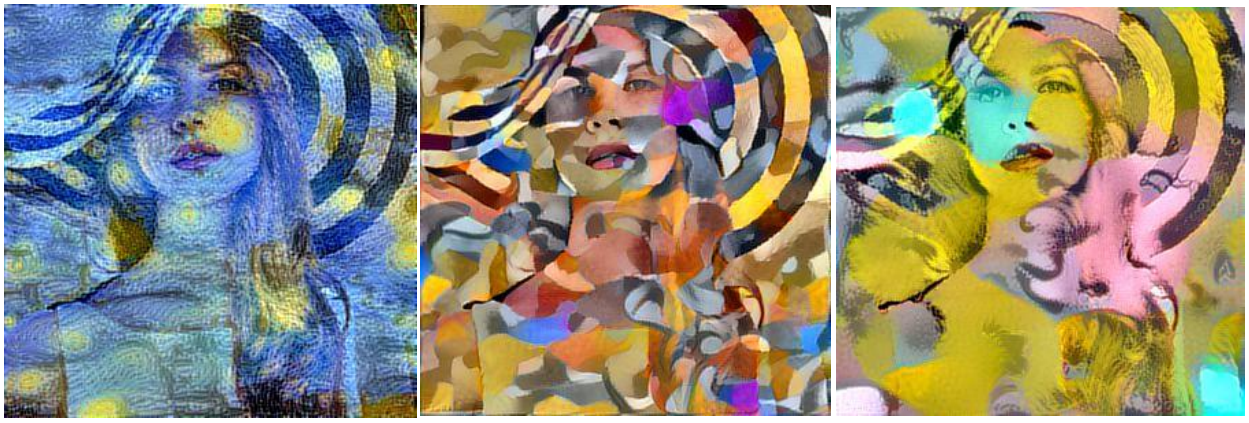
```
def total_variation_loss(x):
    a = backend.square(x[:, :height-1, :width-1, :] - x[:, 1:, :width-1, :])
    b = backend.square(x[:, :height-1, :width-1, :] - x[:, :height-1, 1:, :])
    return backend.sum(backend.pow(a + b, 1.25))

loss += total_variation_weight * total_variation_loss(combination_image)
```

- Leon Gaty’s original paper from 2015 did not use total variation loss (TVL)

- It was introduced in a paper by Mahendran and Vedaldi in 2015 with the aim to encourage image consistency and special continuity, minimizing pixilation and sharp feature formation

The Results:



Overall, I am happier with these results. There is a noticeable reduction in noise if you zoom in closer to look. In addition, the style adaptation is more prominent. In comparison to the previous results where only colors seemed to be inherited, we can see more texture coming into play.