EECS 330 Lab Project

# Neural Style Transfer

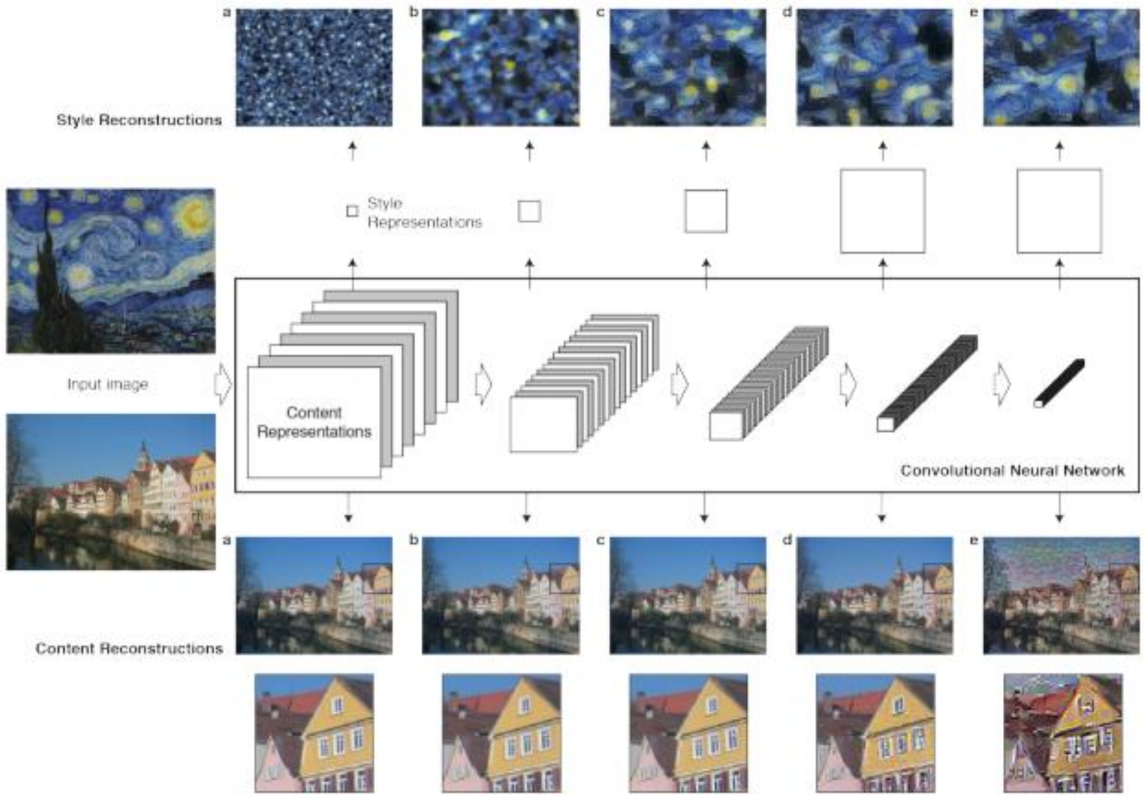
Vishisht Sharma

## Abstract

**In fine art, especially painting, humans have mastered the skill to create unique visual experiences through composing a complex interplay between the content and style of an image. Thus far the algorithmic basis of this process is unknown and there exists no artificial system with similar capabilities. However, in other key areas of visual perception such as object and face recognition near-human performance was recently demonstrated by a class of biologically inspired vision models called Deep Neural Networks.Here I implement an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images. Moreover, in light of the striking similarities between performance-optimised artificial neural networks and biological vision,our work offers a path forward to an algorithmic understanding of how humans create and perceive artistic imagery.**

## Introduction

Transferring the style from one image to another image is an interesting yet difficult problem. There have been many efforts to develop efficient methods for automatic style transfer [Hertzmann et al., 2001; Efros and Freeman, 2001; Efros and Leung, 1999; Shih et al., 2014; Kwatra et al., 2005]. Recently, Gatys et al. proposed a seminal work [Gatys et al., 2016]: It captures the style of artistic images and transfers it to other images using Convolutional Neural Networks (CNN). This work formulated the problem as finding an image that matching both the content and style statistics based on the neural activations of each layer in CNN.



When Convolutional Neural Networks are trained on object recognition, they develop a representation of the image that makes object information increasingly explicit along the processing hierarchy. Therefore, along the processing hierarchy of the network, the input image is transformed into representations that increasingly care about the actual content of the image compared to its detailed pixel values. We can directly visualise the information each layer contains about the input image by reconstructing the image only from the feature maps in that layer (Fig 1, content reconstructions, see appendix for details on how to reconstruct the image). Higher layers in the network capture the high-level content in terms of objects and their arrangement in the input image but do not constrain the exact pixel values of the reconstruction. (Fig 1, content reconstructions d,e). In contrast, reconstructions from the lower layers simply reproduce the exact pixel values of the original image (Fig 1, content reconstructions a,b,c). We therefore refer to the feature responses in higher layers of the network as the content representation

To obtain a representation of the style of an input image, we use a feature space originally designed to capture texture information. This feature space is built on top of the filter responses in each layer of the network. It consists of the correlations between the different filter responses over the spatial extent of the feature maps (see appendix for details). By including the feature correlations of multiple layers, we obtain a stationary, multi-scale representation of the input image, which captures its texture information but not the global arrangement.

The images are synthesised by finding an image that simultaneously matches the content representation of the photograph and the style representation of the respective piece of art (see appendix for details). While the global arrangement of the original photograph is preserved, the colours and local structures that compose the global scenery are provided by the artwork. Effectively, this renders the photograph in the style of the artwork, such that the appearance of the synthesised image resembles the work of art, even though it shows the same content as the photograph.

## Appendix

The results were generated using VGG-Network, a Convolutional Neural Network that rivals human performance on a common visual object recognition benchmark task .I used the feature space provided by the 16 convolutional and 5 pooling layers of the 19 layer VGGNetwork. I did not use any of the fully connected layers.

### Content Loss:

Our content loss definition is actually quite simple. We'll pass the network both the desired content image and our base input image. This will return the intermediate layer outputs (from the layers defined above) from our model. Then we simply take the euclidean distance between the two intermediate representations of those images.

More formally, content loss is a function that describes the distance of content from our input image x and our content image, p . Let C be a pre-trained deep convolutional neural network. Again, in this case we use VGG19. Let X be any image, then C(x) is the network fed by X. Let $F^l_{ij}(x) \in C(x)$ and $P^l_{ij}(x) \in C(x)$ describe the respective intermediate feature representation of the network with inputs x and p at layer l . Then we describe the content distance (loss) formally as:

$$L^l_{content}(p, x) = \sum_{i,j} (F^l_{ij}(x) - P^l_{ij}(p))^2$$

We perform backpropagation in the usual way such that we minimize this content loss. We thus change the initial image until it generates a similar response in a certain layer (defined in content_layer) as the original content image.

## Style Loss:

Computing style loss is a bit more involved, but follows the same principle, this time feeding our network the base input image and the style image. However, instead of comparing the raw intermediate outputs of the base input image and the style image, we instead compare the Gram matrices of the two outputs.

Mathematically, we describe the style loss of the base input image, x, and the style image, a, as the distance between the style representation (the gram matrices) of these images. We describe the style representation of an image as the correlation between different filter responses given by the Gram matrix $G^l$, where $G^l_{ij}$ is the inner product between the vectorized feature map i and j in layer l. We can see that $G^l_{ij}$ generated over the feature map for a given image represents the correlation between feature maps i and j.

To generate a style for our base input image, we perform gradient descent from the content image to transform it into an image that matches the style representation of the original image. We do so by minimizing the mean squared distance between the feature correlation map of the style image and the input image. The contribution of each layer to the total style loss is described by

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - A^l_{ij})^2$$

where $G^l_{ij}$ and $A^l_{ij}$ are the respective style representation in layer l of input image x and style image a. Nl describes the number of feature maps, each of size Ml=height∗width. Thus, the total style loss across each layer is

$$L_{style}(a, x) = \sum_{l \in L} w_l E_l$$

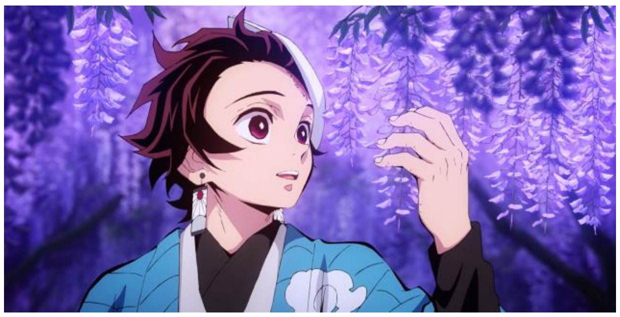
where we weigh the contribution of each layer's loss by some factor wl. In our case, we weight each layer equally:

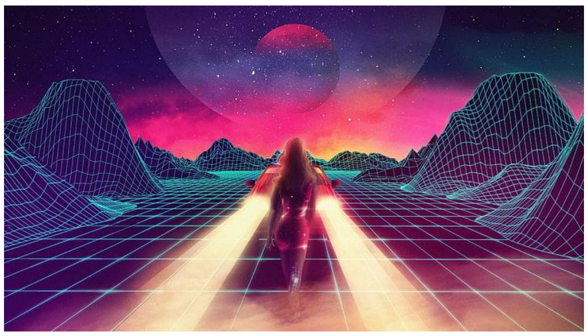
$$(w_l = \frac{1}{||L||})$$

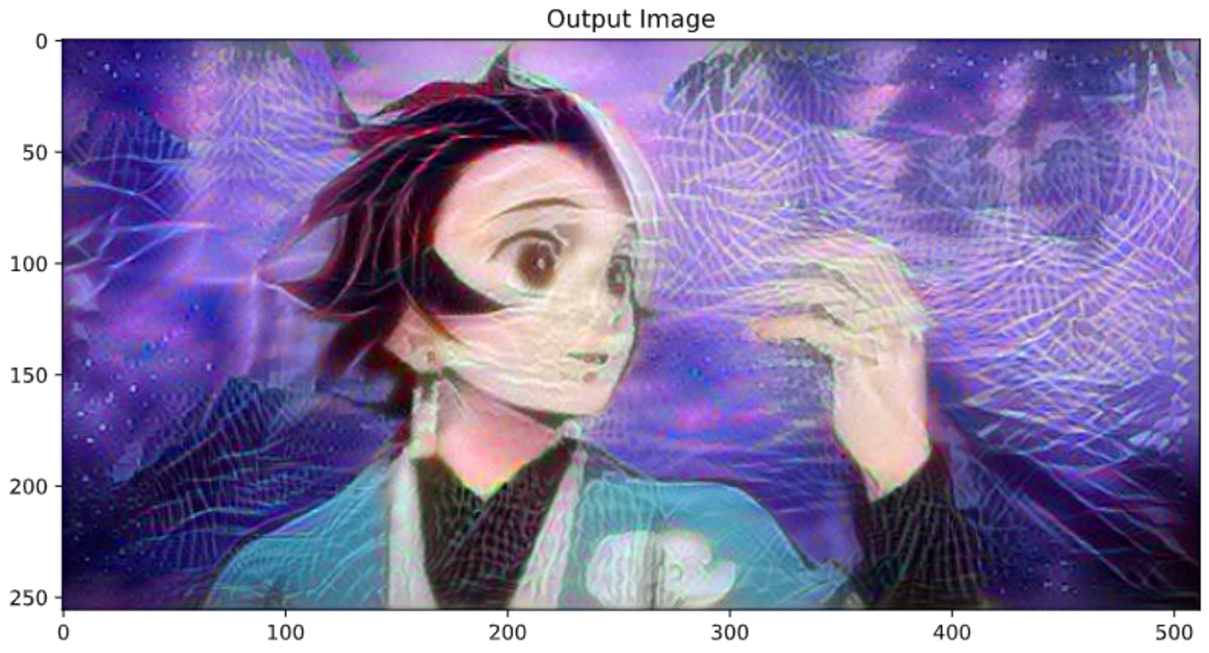
The loss function we minimise is

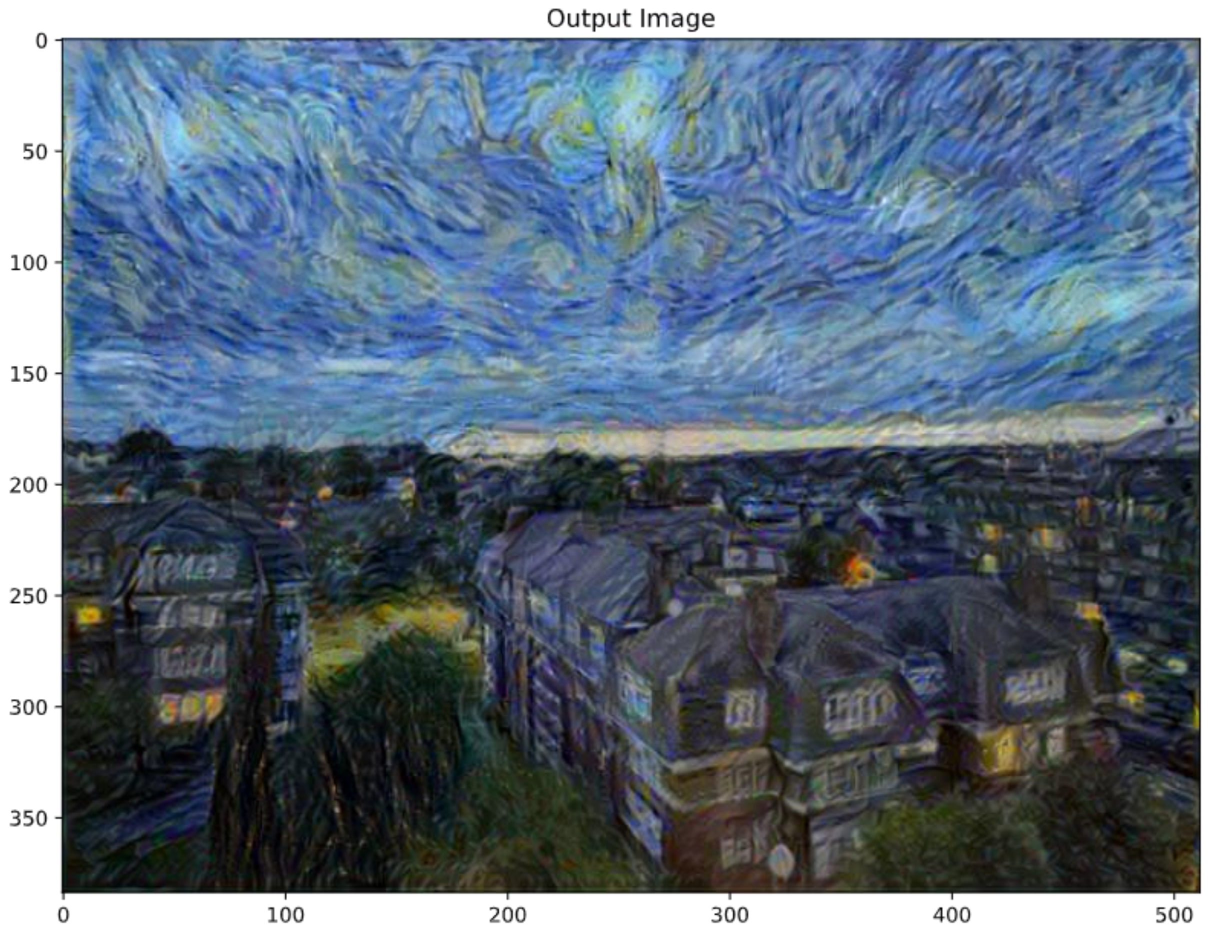
$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$

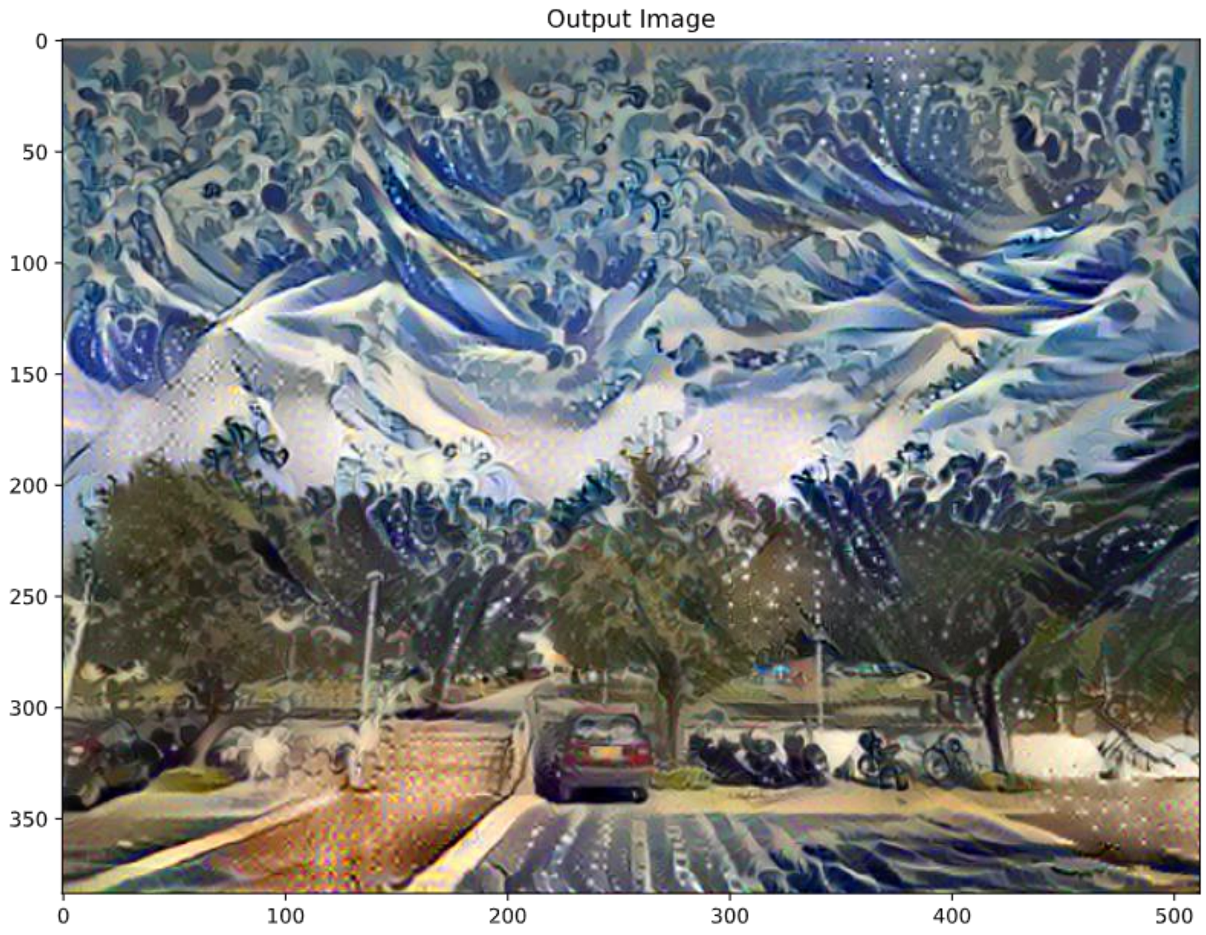
## Results

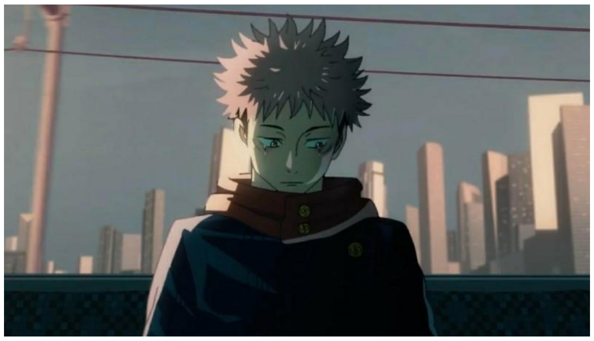


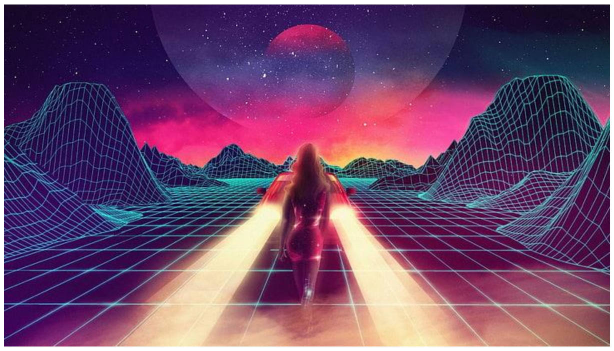


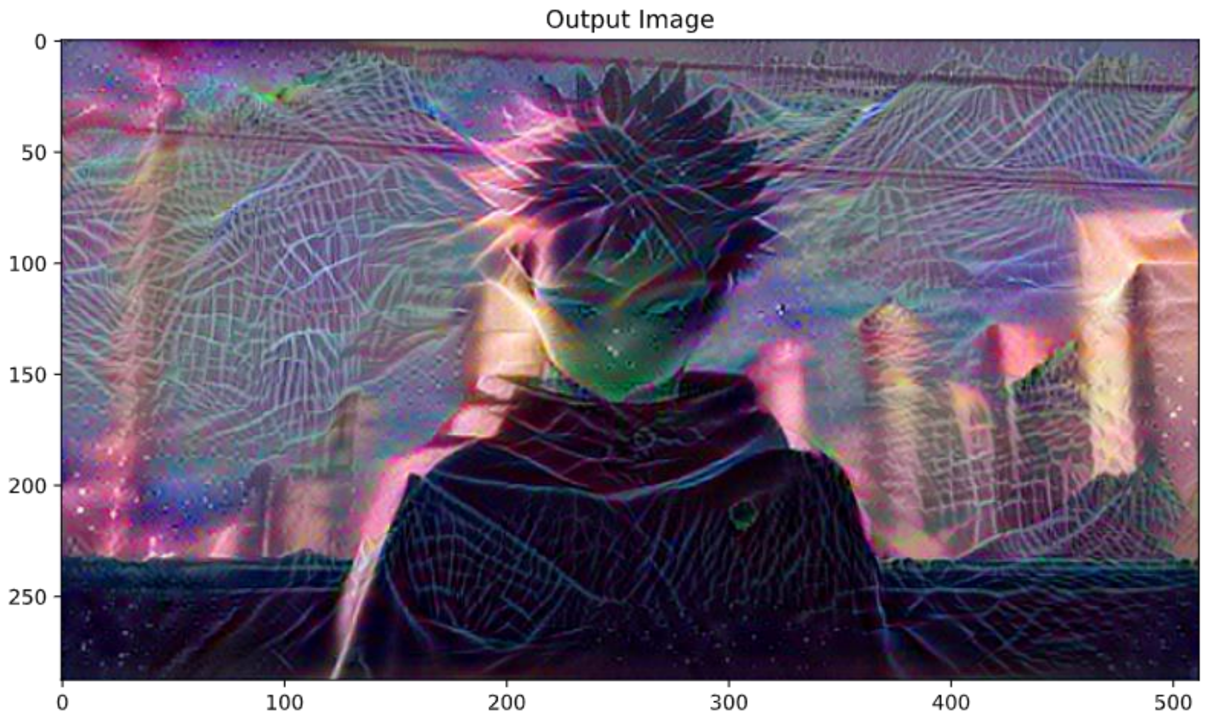



Output Image
0
50
100
150
200
250
300
350
0
100
200
300
400
500


Output Image
0
50
100
150
200
250
300
350
0
100
200
300
400
500






Output Image
0
50
100
150
200
250
0
100
200
300
400
500

**Code for this can be found on Github**

## References

arXiv:1508.06576 **[cs.CV]** A Neural Algorithm of Artistic Style Leon A. Gatys, Alexander S. Ecker, Matthias Bethge