A Neural Algorithm of Artistic Style

Abhyuday Puri (ap3758)[1] and Sai Sourabh Madur (sm4548)[2]

Abstract*— *Creating visually appealing fine art is a skill that has been mastered by humans over time. We lack an understanding of how the brain is able to do it, and therefore, there is no algorithm laid out for this task. In this paper, we implement a Deep Neural Network that creates artistic images of high perceptual quality by separating and combining style and content representations from images. This, in turn, gives us an intuition into the algorithmic understanding of how humans perceive artistic imagery.

I. INTRODUCTION

Convolutional Neural Networks are the most powerful tool when it comes to understanding and working with visual content. When trained for object recognition, they are able to extract increasingly explicit information about the image along the network. The output of each layer can be visualized (Fig 1) by reconstructing the image using only those feature maps. Higher layers in the network capture the objects and their arrangement in the input image, but don't constraint the pixel values. On the other hand, lower layers tend to reproduce the exact pixel values of the image. [3]

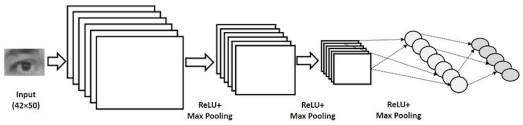

Fig. 1. The basic architecture of a CNN is represented here. At each layer, the various outputs give us information about some feature in the input image.

In order to fuse the content and style of 2 images, we need to be able to separate out these 2 representations from the input. This key finding of this paper is that this is possible using CNNs [2]. These can not be completely disentangled and when recreating an artistic image, one can smoothly regulate the emphasis on content or style. In the next section we discuss the algorithmic understanding of these so called content and style representations.

II. SUMMARY OF THE ORIGINAL PAPER

To capture the style of the image, the paper uses a feature space originally designed to capture texture information [4]. This is built upon the filter responses of each layer in the network. It consists of the correlation between the different activations. By including the activation correlation of multiple layers, we obtain a stationary, multi-scale representation of the input image, which captures its texture information but not the global arrangement [2].

A. Methodology of the Original Paper

The network used in the paper is the VGG-19 Network. The feature space used is that of the 16 convolutional layers and 5 pooling layers. None of the fully connected layers are used. We are now going to lay down the mathematics underlying the algorithm.

The basic idea of image reconstruction from the feature maps is to give the network 2 inputs, the original image $\vec{p}$, and a generated image $\vec{x}$. The generated image $\vec{x}$ is initialized randomly with AWGN, and the feature maps at the layer of reconstruction are compared. We want to optimize our generated image x such that the difference between the 2 feature maps is minimal. This lays the foundation of the algorithm.

A given input image $\vec{x}$ is encoded in each layer of the CNN by the filter responses to that image. Let the l layer have N_l distinct filters, and each filter has a feature map of size M_l, where M_l is the height times the width of the map. All the responses in layer l can be stored in a matrix $F^l \in \mathbb{R}^{N_l \times M_l}$ where F^l_{ij} is the activation of the i^{th} filter at position j in layer l. The squared-error loss is between the 2 feature representations is defined as:

$$L_{content}(\vec{p},\vec{x},l) = \frac{1}{2}\sum_{i,j}\left(F^l_{ij} - P^l_{ij}\right)^2 \tag{1}$$

The derivative of this loss with respect to the activations in layer l equals

$$\frac{\partial L_{content}}{\partial F^l_{ij}} = \begin{cases} \left(F^l - P^l\right)_{ij} & \text{if } F^l_{ij} > 0 \\ 0 & \text{if } F^l_{ij} < 0 \end{cases} \tag{2}$$

from which the back-propogation with respect to the image $\vec{x}$ can be computed. Thus, $\vec{x}$ is changed until it generates the same response in a certain layer of the CNN as the original image $\vec{p}$. Optimizing this gives us the *content component* of the image.

The style representation is built on top of the CNN responses, where the correlation between different filter responses is computed. These feature correlations are given by the Gram Matrix $G^l \in \mathbb{R}^{N_l \times N_l}$ where G^l_{ij} is the inner product between the vectorized feature map i and j in layer

[1]A. Puri is a MS student in the Electrical Engineering Department, Columbia University

[2]S. Madur is a MS student in the Electrical Engineering Department, Columbia University

l:

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk} \tag{3}$$

To generate a texture that matches the style of the image, we do the same thing as before. We pass a white noise image and improve it using gradient descent by minimizing the mean-squared distance between the entries of the Gram Matrix. So, let $\vec{a}$ and $\vec{x}$ be the original image and the image that is generated, and A^l and G^l their respective style representations in layer l. The contribution of that layer to the total loss is:

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

and the total loss is

$$L_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l \tag{5}$$

where w_l are weighting factors of the contribution of each layer to the total loss. The derivative of E_l with respect to the activations in layer l can be computed analytically:

$$\frac{\partial E_l}{\partial F^l_{ij}} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left(\left(F^l \right)^T \left(G^l - A^l \right) \right)_{ji} & \text{if } F^l_{ij} > 0 \\ 0 & \text{if } F^l_{ij} < 0 \end{cases} \tag{6}$$

The gradients of E_l with respect to the activations in lower layers can easily be back-propagated.

To generate images that mix the content of a photograph with the style of a painting we jointly minimize the distance of the white noise image from the content representation of the photograph in one layer of the network and the style representation of the painting in a number of layers of the CNN. So, let $\vec{p}$ be the photograph and $\vec{a}$ be the artwork. The loss function to be minimized is

$$L_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha L_{content}(\vec{p}, \vec{x}) + \beta L_{style}(\vec{a}, \vec{x}) \tag{7}$$

where α and β are the weighting factors for content and style reconstruction respectively. In the paper, they matched the content representation on layer 'conv4_2' and the style representations on layers 'conv1_1', 'conv2_1', 'conv3_1', 'conv4_1' and 'conv5_1' ($w_l = \frac{1}{5}$ in those layers and 0 in all other layers). The ratio α/β was either 1×10^{-3} or 1×10^{-4}.

B. Key Results of the Original Paper

The key finding of this paper is that the representations of content and style in the Convolutional Neural Network are separable. Both these representations can be manipulated independently to produce new, and perceptually meaningful images. Since they achieve a separation of content from style, it allows them to recast the content of any image in the style of any other image, without any constraints. It is indeed very fascinating that a system trained for object recognition, inherently learns image representations that allow it to separate content from style.

III. Methodology

This project follows a similar methodology as the paper. We used the VGG-19 network [5] trained for image classification to create an artistic imagery.

A. Objectives and Technical Challenges

There were 2 main objectives that were laid out when beginning this work:

- Replicating the paper and producing similar results
- Find a better suited setting of the hyper-parameters

The way the optimization is set up (to be able to work on any content image and any other style image), creating the artistic imagery is time consuming. No parameters are learned during the optimization, and only the feature space representation distance between the generated image and the style and content images is minimized in an iterative manner. This means that for every pair of style and content images, the entire process has to be repeated.

In addition to the time factor, since there is still no algorithmic understanding of how humans perceive and create artistic imagery, deciding the parameters for this minimization has no intuition behind it. A vast set of different parameter settings and cost functions need to be tried.

Finally, there exists no objective measure that can capture perceptive quality of an image. An image that might be appealing to one person doesn't necessarily imply that it will be appreciated in the same manner universally.

B. Problem Formulation and Design

The idea is to be able to identify and isolate the content from one image, and the style from another image, and fuse the 2 together to create an artistic imagery. The problem is formulated in the same way as above.

We use a VGG network that has been trained for object classification (link available in our git documentation). We define 2 reconstruction losses:

- Content Loss ($L_{content}$)
- Style Loss (L_{style})

The content loss is defined such that it aims to minimize the difference between the content image and the generated image by matching their pixel intensity values. The loss is defined as follows:

$$L_{content} = \frac{1}{2} \sum_{i,j=1}^{N} \left(F^l_{ij} - P^l_{ij} \right)^2 \tag{8}$$

Where the terms in this equation have the same meaning as above. P^l is the feature map of the content image at the l^{th} layer, and F^l is the feature map of the generated image in the l^{th} layer. The idea here is that if 2 images have similar content, their feature map should be identical as well, since they are being subjected to same set of operations when

being passed through the neural network.

For the style reconstruction error, we define it as:

$$L_{style} = \sum_{l} w_l \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - A_{ij}^l\right)^2 \tag{9}$$

where w_l is a hyper parameter. G^l and A^l are the Gram matrices of the feature maps of the generated and style image at the l^{th} respectively. Here, we are trying to minimize the gap between the 'co-variance' of the features of the images. This co-variance tends to contain the texture information of our images.

Once the replication has been done, we set up an exhaustive list of experiments to be able to identify the trade-off between different parameters (α, β) and choosing different layers in the network for the generation of the new image.

IV. Implementation

This section describes the Deep Neural Network architecture used by us, and details about the code.

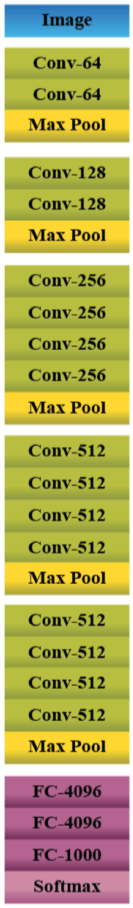

Fig. 2. The architecture of VGG-19 [6]

A. Deep Learning Network

Next, we provide the architecture of the VGG-19 network used. The dimension of the input image to the network is always ensured to be $224 \times 224 \times 3$. This network contains 16 convolutional layers and 3 fully connected layers. This means it has a total of 19 weight layers, hence the name VGG-19. The depth of each layer is shown in the Fig. 2. In each layer, the filer size being used is 3×3. For our task, we do not need the fully connected layers of this network. We take outputs only from the convolutional layers.

The network was trained on the ImageNet dataset for image classification [7]. We directly use this pre-trained network for our task. For our algorithm, you can use any readily available for content. For styles, we stick to popular images such as *The Starry Night* by Vincent Van Gough (1889), *Der Schrei* by Edvard Munch (1893) and *Composition VII* by Wassily Kandinsky (1913).

B. Software Design

This flow chart in Fig 3. represents the abstract view of the algorithm. It just gives us the basic understanding of how our data flows in the entire process. At the beginning, we feed into it the style image, the content image, and a randomly generated AWGN image, called the Generated image. The output of the network is then fed into our optimization algorithm that computes the reconstruction error for the generated image, and accordingly updates the values in it. This process is repeated until we achieve a satisfactory result. In our case, we ran the program for 1000 iterations, to achieve visually appealing results.

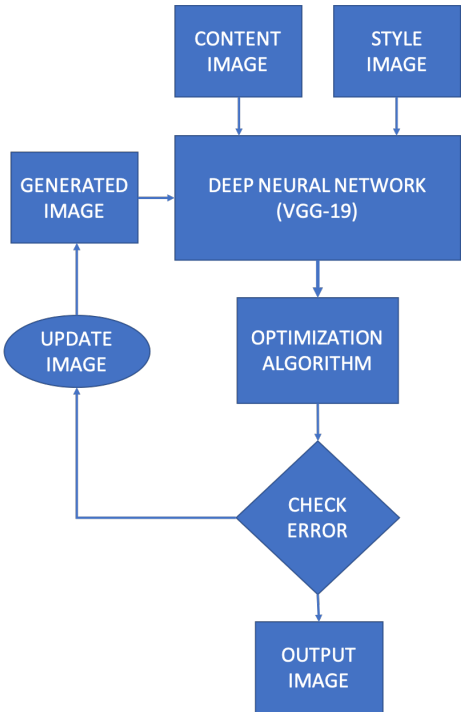

Fig. 3. The basic flow of data in the algorithm

We now look into the optimization algorithm details. The

loss function from equation (7) is used.

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

For $L_{content}$ we compute the reconstruction error for the content image and the generated image at the layer 'conv4_2.' In L_{style}, we use the gram matrix of the outputs from layers 'conv1_1', 'conv2_1','conv3_1', 'conv4_1' and 'conv5_1' to compute the style loss. w_l is chosen to be 0.2 for these layers, and 0 for the layers not being used. We always ensure that $\sum_l w_l = 1$.

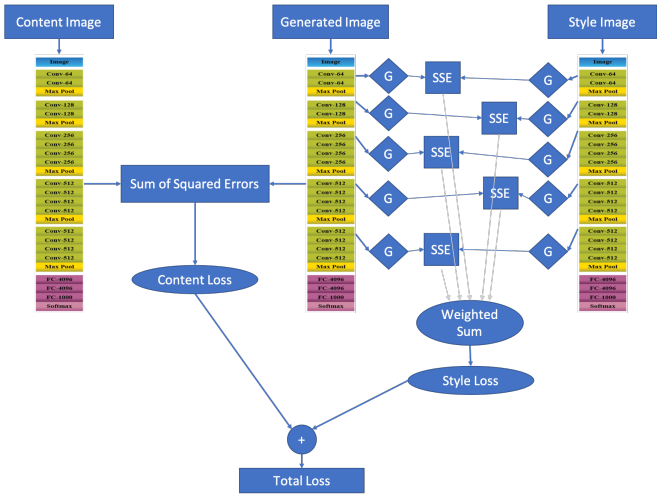

Fig. 4. The Cost Function Computation

In Fig 4, G is a transform that finds the Gram Matrix of the input given to it. The total loss is computed as the weighted sum of the content loss and the style loss. We use the Gradient descent method to optimize this problem. One must note that this ia a constrained problem, in a sense that the values in the generated image must always lie in the range $[0,255]$ (since it is an image). We used the TensorFlow AdamOptimizer for the same.

V. Results

A. Project Results

The images in Fig 5. and Fig 6. show the final output of our algorithm. As can be seen, images with good perceptive quality have been created. Next, we present a few more results that we derived which help in providing an algorithmic understanding of image perception.

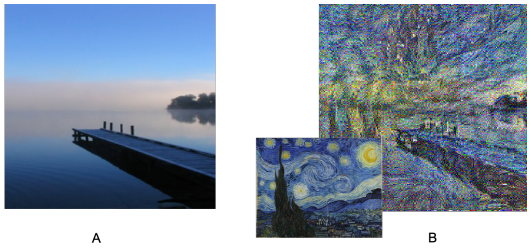

Fig. 5. **A** The original photograph. **B** *The Starry Night* by Van Gough.

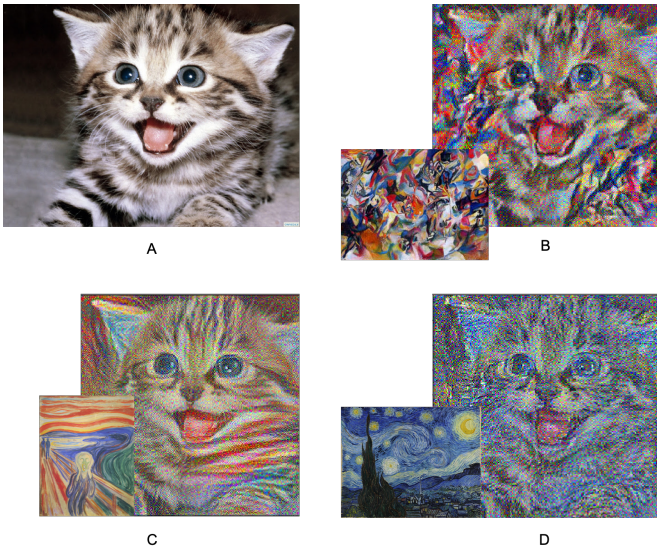

Fig. 6. Images that combine the content of a photograph with the style of several well-known artworks. **A** The original photograph. **B** *Composition VII* by Wassily Kandinsky (1913). **C** *Der Schrei* by Edvard Munch (1913). **D** *The Starry Night* by Vincent van Gough (1889).

In Fig 7., we see the result of varying the hyper-parameters, and also using different layers in our cost function for the style representation term. As α/β ratio is increased, we see higher amounts of the content image. This is what was expected as well, as incresing α means penalizing the optimization more for not re-generating the content well enough. The multiple rows in the image give results with a differnet number of layers used. The first row uses only the 'conv1_1' term in the L_{style} function. The second row uses 'conv1_1', 'conv2_1' and 'conv3_1'. Similarly, the last row uses 'conv1_1', 'conv2_1', 'conv3_1', 'conv4_1' and 'conv5_1'. Here we see that if we use lesser

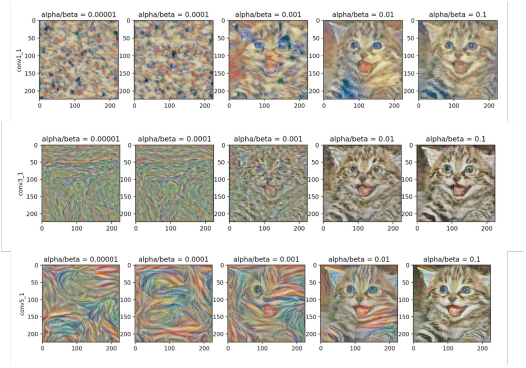

Fig. 7. Effect of varying the hyper-parameters

layers for the style representation, and especially, neglect the higher layers, there is not much style transferred to the image. It mainly reconstructs the content. Whereas, for an effective transfer of style, larger number of layers should be used.

Fig 8. shows us the content reconstruction from different layers of the network. As can be seen, the higher the layer

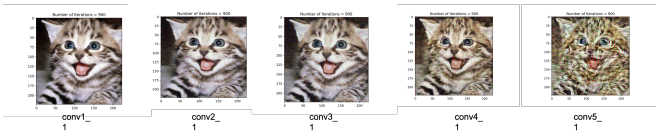

Fig. 8. Content Representation

is in the network, the more noise we get in the reconstructed image.

B. Comparison of Results

We were able to succesfully replicate the results of the paper. In addition to this, we were able to add more insights into the algorithmic understanding of this tasks.

C. Discussion of Insights Gained

The beauty of this algorithm is that it does not depend on the style or content image. We can chose the two of them independently, and it still performs well. This also lays the foundation for the algorthmic understanding of human perception of artistic imagery. We understand now that the style and content can be isolated from images, and this is a great step forward in understanding image perception.

VI. Conclusion

Style transfer is achieved by minimizing two loss terms. Minimizing the content loss at the shallow layers reproduces the image perfectly as shown in Fig 8 and minimizing at deeper layers results in an imperfect reproduction of the content image. Conversely , minimizing the style loss as we go deeper down the layers results in the reproduction of style content more closely with the style image.

The style transfer is highly sensitive on the content loss weight, alpha and the style loss weight beta as has been shown in Fig 7. Therefore, tuning alpha, beta, learning rate, optimization method(BFGS is reported to be better in many of the papers) and the type of pretrained model used will all play a key role in producing an aesthetically pleasing style transfer.

ACKNOWLEDGMENT

We would also like to thank the TA's and Prof. Zoran Kostic for their constant support throughout the course of this project.

Also, we would like to express our gratitude to Prof. Zoran for providing us with the cloud compute power which helped speed up our progress.

References

[1] https://bitbucket.org/ecbm4040/2018_assignment2_ap3758/src/master/Project-saved-model/

[2] Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). A Neural Algorithm of Artistic Style. ArXiv:1508.06576. URL: http://arxiv.org/abs/1508.06576

[3] Mahendran, A., & Vedaldi, A. (2014). Understanding Deep Image Representations by Inverting Them. ArXiv:1412.0035. URL: http://arxiv.org/abs/1412.0035

[4] Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). Texture Synthesis Using Convolutional Neural Networks. ArXiv:1505.07376 [Cs, q-Bio]. Retrieved from http://arxiv.org/abs/1505.07376

[5] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. ArXiv:1409.1556 [Cs]. Retrieved from http://arxiv.org/abs/1409.1556

[6] Mojumder, U. (n.d.). Vehicle Model Identification using Neural Network Approaches, 57.

[7] Deng, J. and Dong, W. and Socher, R. and Li, L.-J. and Li, K. and Fei-Fei, L., ImageNet: A Large-Scale Hierarchical Image Database, CVPR09, 2009