

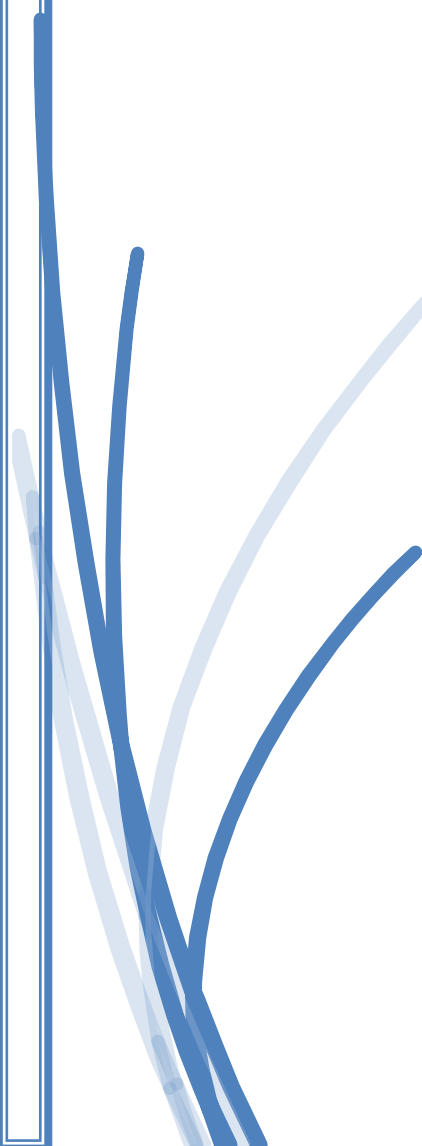


Neural Style Transfer

SHREYANSH SINGH

21119047

PRODUCTION AND INDUSTRIAL
ENGINEERING
4TH YEAR



Neural Style Transfer

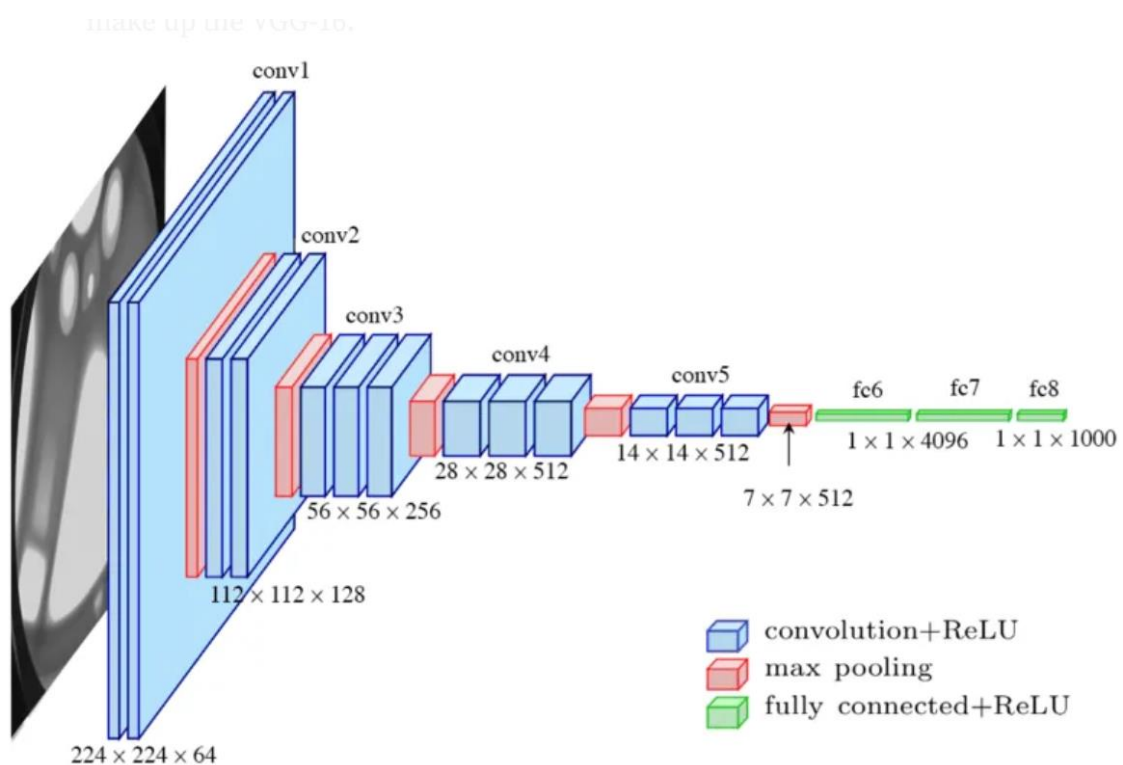
Introduction

Neural Style Transfer, a cutting-edge technique, allows us to recreate the style of renowned paintings, such as the Mona Lisa by Leonardo da Vinci and Starry Night by Vincent Van Gogh. This innovative project is based on the research paper 'A Neural Algorithm of Artistic Style' by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. The system uses neural representations to separate and merge the content and style of different images, creating new artistic images using a neural algorithm.

Approach

This project leverages the highly successful VGG 19 pre-trained convolutional neural network (CNN) model introduced by K. Simonyan and A. Zisserman in their paper 'Very Deep Convolutional Networks for Large-Scale Image Recognition.' VGG19, a model known for its simplicity and depth, achieved remarkable results in the 2014 ImageNet Challenge. It comprises 19 layers, 16 convolutional layers, three fully connected layers, and five max-pooling layers.

The architecture of the model is as follows:



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224×224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 conv1-256	conv3-256 conv3-256 conv3-256	conv3-256 conv3-256 conv3-256 conv3-256
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv1-512	conv3-512 conv3-512 conv3-512	conv3-512 conv3-512 conv3-512 conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

The VGG19 model encompasses 1,53,04,768 parameters, the internal variables the model utilises to make predictions. These parameters, including weights and biases, have been trained on the ImageNet dataset, a comprehensive collection of labelled images for deep-learning models. In the context of Neural Style Transfer, these parameters are instrumental in capturing and representing a broad spectrum of features in images, thereby making the VGG19 model a suitable choice for such tasks.

The weights of the VGG model are derived from the imagenet classification, given the scarcity of our data and computational resources. We are mindful of these constraints and have tailored our approach accordingly, ensuring the best possible results within our limitations. As a result, the model uses the pre-trained weights, and in the VGG model, Dense layers and softmax layers present in the image classification model are not included. This lets us focus on Neural Style Transfer's relevant features and representations.

The Visual Geometry Group (VGG) model, characterised by its architecture of 16 convolutional layers and five max-pooling layers, stands at the forefront of advancements in

image processing. This architecture facilitates the extraction of a broad spectrum of features from images, significantly reducing computational complexity and thus enhancing the efficiency of model training. Within this framework, the convolutional layers play a critical role in feature extraction, with the initial layers focusing on low-level feature extraction and the succeeding dense layers being proficient in high-level feature extraction, encompassing the identification of specific objects within the images. The primary objective of employing the VGG model is to harness its capabilities for generating images conforming to a predetermined stylistic format.

Our methodology for achieving this involves strategically leveraging texture details identifiable in the initial layers of each block within the VGG model, coupled with the establishment of meaningful correlations across different layers. This approach pivots on the reconstruction of the style of an input image through the creation of meticulously crafted style representations. These representations are derived from specific subsets of Convolutional Neural Network (CNN) layers, enabling the generation of images that seamlessly embody the style of a given image.

In tackling the challenge of blending context and style images, this approach facilitates independent manipulation of these elements to create a new image that effectively incorporates features from both sources. Despite the inherent challenge associated with replicating both the context and style images with precision, the ultimate aim is to produce visually compelling images by ensuring the alignment of style representations with the highest layers of the network.

This endeavour to merge the essence of context and style images into a single, unique image presents a promising avenue for exploration in image processing. Through meticulous experimentation and refinement of our approach, we aim to achieve and excel in creating images with aesthetic appeal and relevance. It is anticipated that the outcomes of this research will offer significant contributions to the fields of computer vision and image processing, providing new avenues for the creative manipulation of image aesthetics.

In the model, the input image x is encoded in each layer of the CNN through the filter responses. A layer contains several filters, each with a size of MI . The product of the height and width of the feature map is denoted as MI . The responses in layer l can be stored in a matrix F^l of size RNI_MI , where F^l_{ij} is the activation of the i th filter at position j in layer l . To visualise the image information encoded at different layers of the hierarchy (Fig 1, content reconstructions), gradient descent is performed on a white noise image to find another image that matches the feature responses of the original image. Let \tilde{p} and \tilde{x} be the original and generated images, and P^l and F^l their respective feature representations in layer l . The squared error loss between the two feature representations is then defined.

$$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} (F^l_{ij} - P^l_{ij})^2$$

The derivative of this loss concerning the activation in layer l equals

$$\frac{\partial \mathcal{L}_{content}}{\partial F_{ij}^l} = \begin{cases} (F^l - P^l)_{ij} & \text{if } F_{ij}^l > 0 \\ 0 & \text{if } F_{ij}^l < 0 . \end{cases}$$

The feature correlations can be expressed by the Gram matrix G_l consisting of vectors in layer l . The inner product G_{lij} represents the connection between feature map i and j .

This process involves minimising the mean-squared distance between the entries of the Gram matrix from the original image and the Gram matrix of the image to be created. Let \tilde{a} and \tilde{x} be the original and generated images, and A_l and G_l are their respective style representations in layer l . The contribution of that layer to the total loss is determined by the weighting factors of the contribution of each layer to the total loss (see below for specific values of w_l

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^L w_l E_l$$

in our results).

The derivative of E_l concerning the activations in a layer can be computed analytically:

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

The gradients of the error about the activations in lower layers of the network can be easily computed using standard error back-propagation. The five style reconstructions in Fig 1 were generated by matching the style representations on layer ‘conv1 1’ (a), ‘conv1 1’ and ‘conv2 1.’ (b), ‘conv1 1’, ‘conv2 1’ and ‘conv3 1’ (c), ‘conv1 1’, ‘conv2 1’, ‘conv3 1’ and ‘conv4 1’ (d), ‘conv1 1’, ‘conv2 1’, ‘conv3 1’, ‘conv4 1’ and ‘conv5 1’ (e).

To generate the images that mix the content of a photograph with the style of a painting (Fig 2), we jointly minimise the distance of a white noise image from the content representation of the picture in one layer of the network and the style representation of the painting in several layers of the CNN. So let \tilde{p} be the photograph and \tilde{a} be the artwork. The loss function we minimise is:

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

Please take note of the following information:

In the images depicted in Fig 2, we matched the content representation using layer ‘conv4 2’ and the style representations using layers ‘conv1 1’, ‘conv2 1’, ‘conv3 1’, ‘conv4 1’, and ‘conv5 1’ with the respective weightings of 1/5 in those layers and 0 in all other layers. The content reconstruction weighting factor's ratio to style reconstruction weighting was 1:10³ for Fig 2 B, C, D, or 1:10⁴ for Fig 2 E, F.

Fig 3 displays the results for different relative weightings of the content and style reconstruction loss (in the columns) and for matching the style representations only on layer ‘conv1 1’ (A), ‘conv1 1’ and ‘conv2 1’ (B), ‘conv1 1’, ‘conv2 1’ and ‘conv3 1’ (C), ‘conv1 1’, ‘conv2 1’, ‘conv3 1’ and ‘conv4 1’ (D), and ‘conv1 1’, ‘conv2 1’, ‘conv3 1’, ‘conv4 1’ and ‘conv5 1’ (E).

In the described model, we start by transforming two images into tensors. This serves as the foundation for creating a complex model to extract distinctive features from the context and style images. The model strategically extracts outputs from five distinct layers to ensure a nuanced texture in the final product. Alongside this, a singular layer's production is dedicated to capturing the essence of the context image.

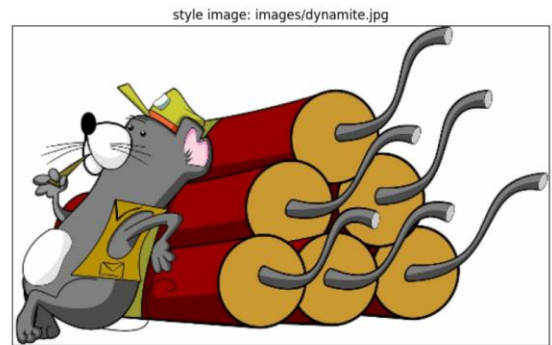
Further refining our approach for style transfer, the model leverages the outputs of the initial layers of each convolutional block—for instance, capturing the output from layers such as block_1_conv_1 and continuing in this manner across successive blocks. This methodical selection is intended to encapsulate the stylistic elements effectively. In contrast, for encapsulating the image's context or substantive content, the strategy shifts towards extracting the output from the very last layer of the VGG model's final block. It's crucial to note that this extraction process deliberately excludes the dense layers and the softmax function. This is achieved by setting the ‘include_top’ parameter to false, ensuring that our focus remains on the convolutional features rather than the classification layers typically used in deep learning models for image recognition tasks.

Once these foundational steps are completed, the model transitions into a critical phase, evaluating the loss associated with the style and context features. This evaluation is pivotal, as it measures how well the generated image represents the desired style and context compared to the original images.

Based on the outcomes of this loss evaluation, the final and perhaps most crucial step involves calculating the gradient descent. This iterative process minimises loss by making incremental adjustments to the generated image. Through this meticulous optimisation, the model gradually improves the visual quality and fidelity of the generated image, ensuring that it faithfully captures both the aesthetic style and the contextual essence of the provided images. This intricate process exemplifies the fusion of art and technology, achieving a harmonious balance between the two to create visually stunning synthesised images.

Result

the image generated by combining the features of both the context image and the style image is given below:



Discussion

In this model, I have demonstrated the capability to merge two images and create a composite image that incorporates the features of both the original context image and the style image. The generated image could be further enhanced by increasing the training epochs; I trained it for only 20 epochs in this example. We can use a pre-trained TensorHub module model to achieve better results. This approach will likely yield a much-improved image.