Style Transfer

Debashish (2021201034), Samyak (2019101013), Shreyash (2021201074)

Problem Description

Goal : Transfer the style while keeping the content same

The separation of image content from style is not necessarily a well defined problem.

This is mostly because it is not clear what exactly defines the style of an image. It might be the brush strokes in a painting, the colour map, certain dominant forms and shapes, but also the composition of a scene and the choice of the subject of the image.

What is Style Transfer?

Style Transfer is the technique of blending style from one image into another image keeping its content intact. The only change is the style configurations of the image to give an artistic touch to your image.

Neural Style Transfer deals with two sets of images: Content image and Style image.

This technique helps to recreate the content image in the style of the reference image. It uses Neural Networks to apply the artistic style from one image to another.

How does Style Transfer work?

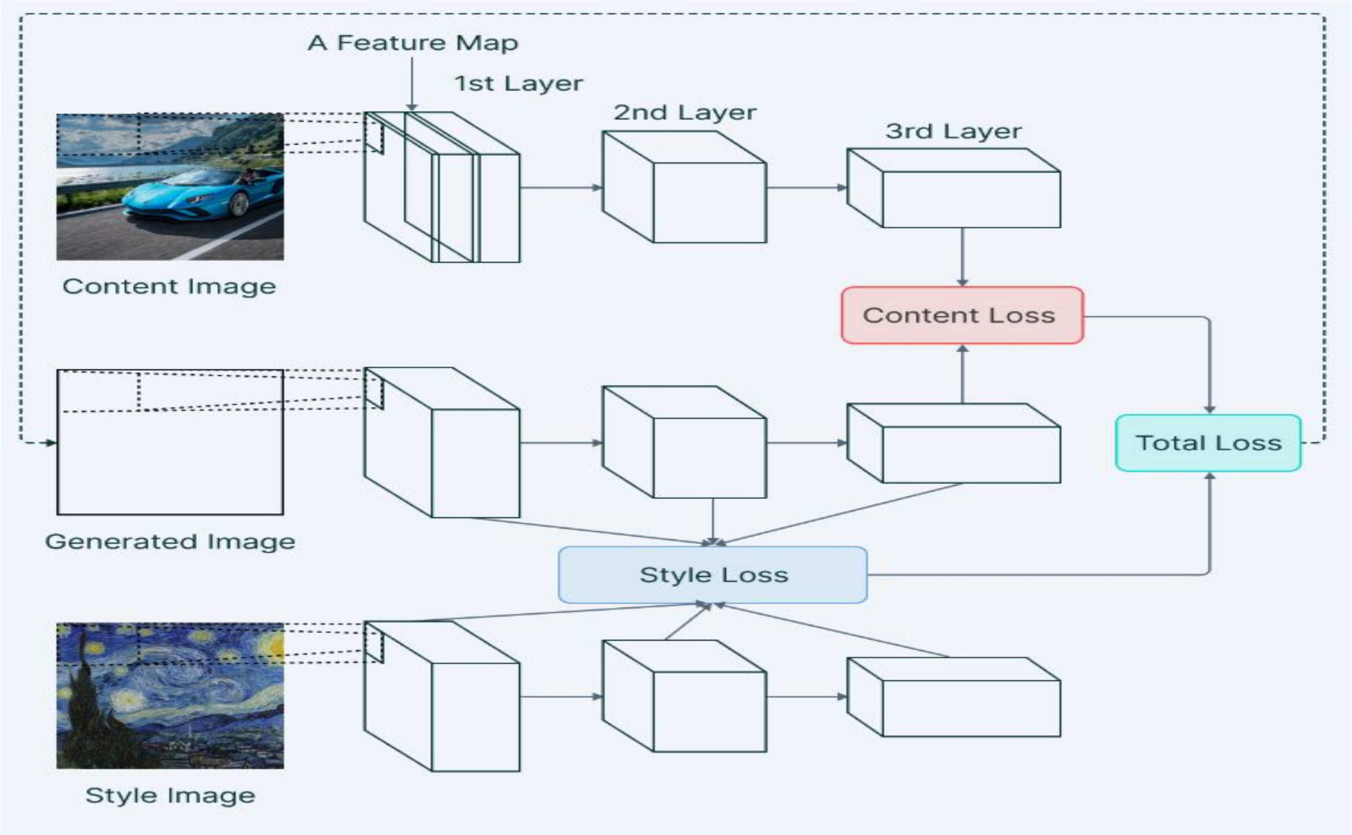
The aim of Style Transfer is to give the Deep Learning model the ability to differentiate between the style representations and content image.

It employs a pre-trained Convolutional Neural Network with added loss functions to transfer style from one image to another and synthesize a newly generated image with the features we want to add.

Style transfer works by activating the neurons in a particular way, such that the output image and the content image should match particularly in the content, whereas the style image and the desired output image should match in texture, and capture the same style characteristics in the activation maps.

These two objectives are combined in a single loss formula, where we can control how much we care about style reconstruction and content reconstruction.

Architecture



Content Representation

Higher layers of the model focus more on the features present in the image i.e. overall content of the image.

Content loss is calculated by Euclidean distance between the respective intermediate higher-level feature representation of input image (*x*) and content image (*p*) at layer l.

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F_{ij}^{l} - P_{ij}^{l}\right)^2$$

Style Representation

Style loss can't be computed like content loss because we cannot compare the intermediate features of the two images and get the style loss. We introduce "Gram Matrices".

Gram matrix measures the correlation between feature maps in a given layer. It measures the texture information rather than global arrangement. It is defined as the inner product between feature maps i and j in layer l.

Style loss E_l is defined as :

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

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l .$$

G^l_{ij} is the inner product between the vectorised feature maps i and j in layer l

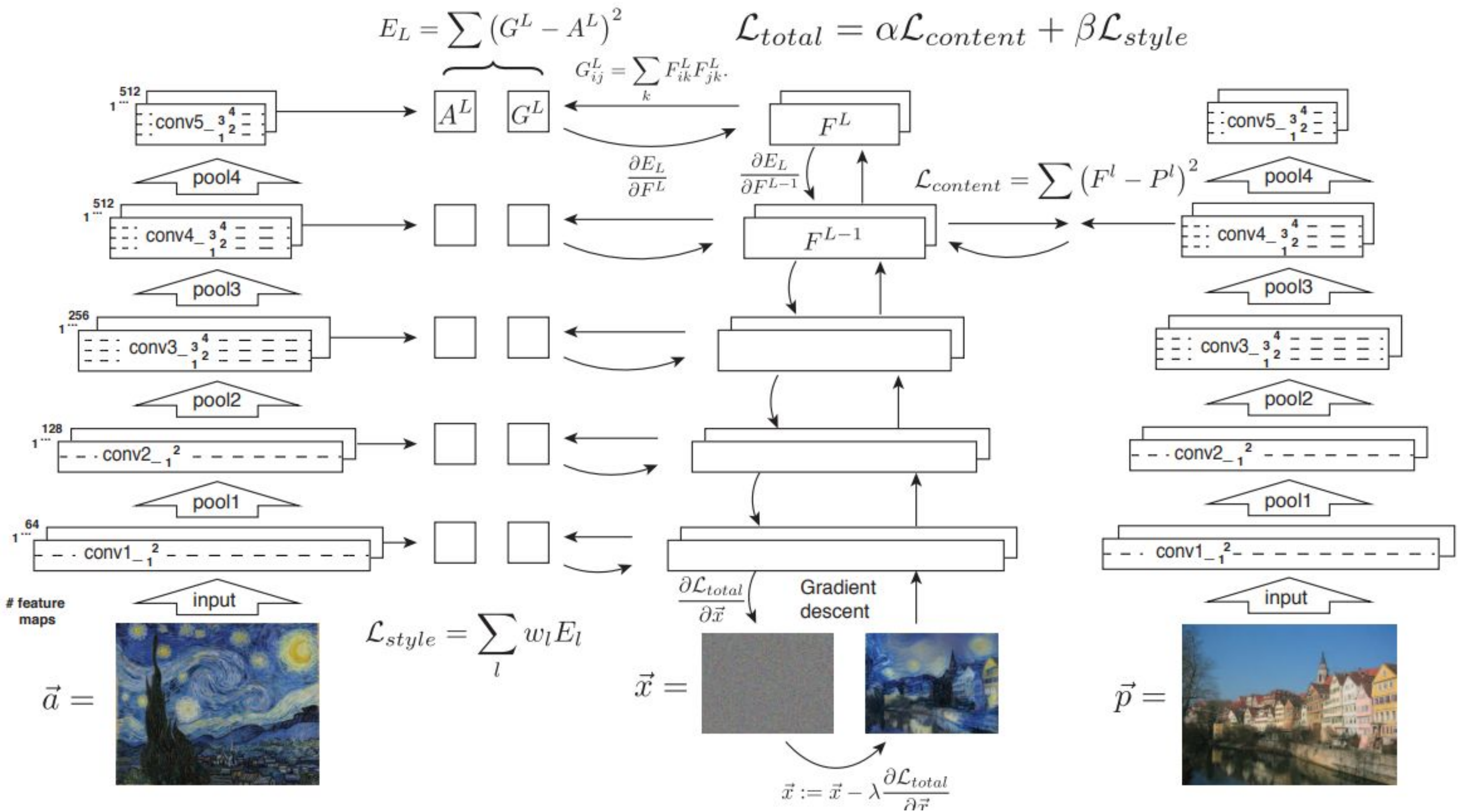
Style Loss

Total style loss is computed by weighing loss of each layer.

Derivative of style loss is computed at each layer L.

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

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



$E_L = \sum (G^L - A^L)^2$
$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$
$G^L_{ij} = \sum_k F^L_{ik} F^L_{jk}.$
A^L
G^L
F^L
F^{L-1}
$\frac{\partial E_L}{\partial F^L}$
$\frac{\partial E_L}{\partial F^{L-1}}$
$\mathcal{L}_{content} = \sum (F^l - P^l)^2$
conv5_1 conv5_2 conv5_3 conv5_4
pool4
conv4_1 conv4_2 conv4_3 conv4_4
pool3
conv3_1 conv3_2 conv3_3 conv3_4
pool2
conv2_1 conv2_2
pool1
conv1_1 conv1_2
input
512
256
128
64
feature maps
$\mathcal{L}_{style} = \sum_l w_l E_l$
$\frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$
Gradient descent
$\vec{a} =$
$\vec{x} =$
$\vec{p} =$
$\vec{x} := \vec{x} - \lambda \frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$

Overall Loss

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

where α and β are the weighting factors for content and style reconstruction, respectively.

Evaluation of Style Transfer

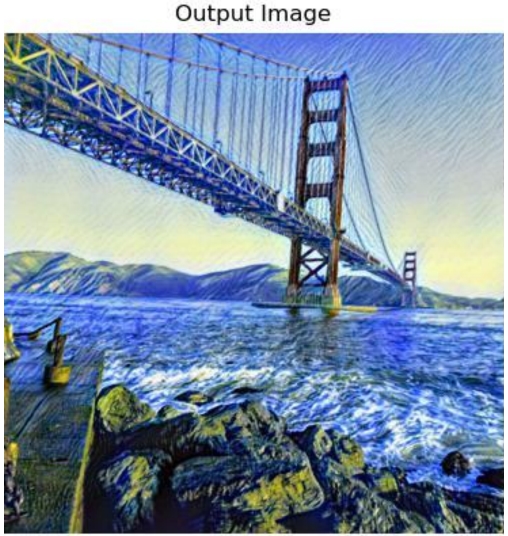
No evaluation metric which is mathematically precise.

We consider style transfer to be successful if the generated image 'looks like' the style image but shows the objects and scenery of the content image.

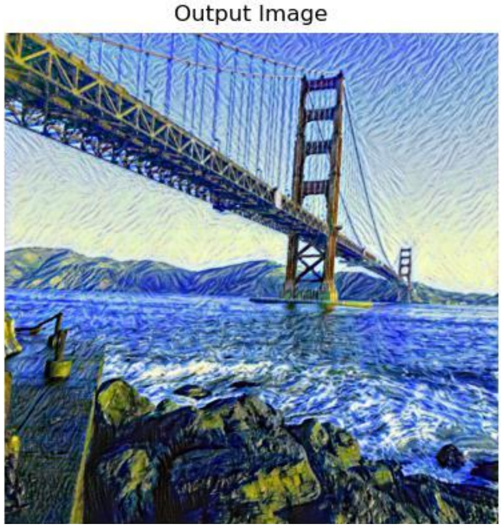
Experiments

- Try different optimisers like Adam, L-BFGS, SGD
- Try different ConvNets like VGG, ResNet, DenseNet
- Vary α / β ratio
- Pooling - Max, Avg
- Study the effect of different layers of the Convolutional Neural Network
- Change initialization of the images - noise, content image, style image
- Add batch normalization layers. We are referring Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization to make changes in the architecture

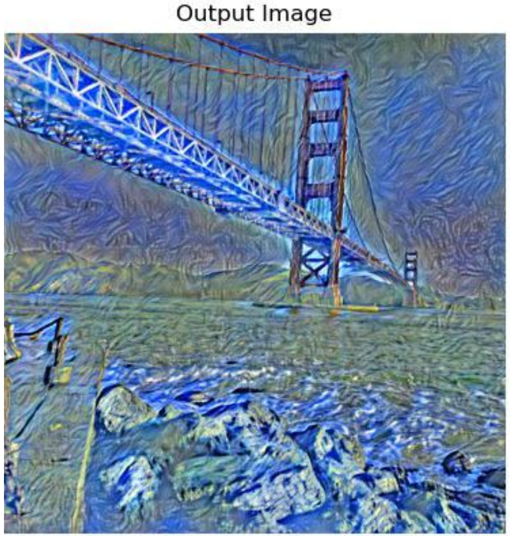
Results



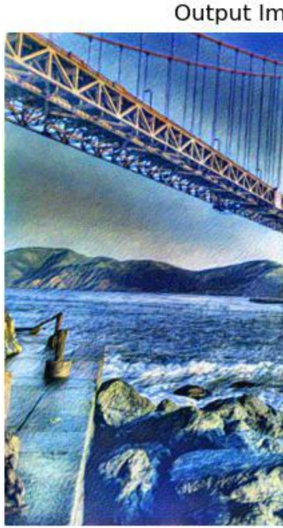
1/100000



1/1000000



1/10000



1/1000

Results

Output Image

1/100000

Output Image

1/1000000

Output Image

1/10000

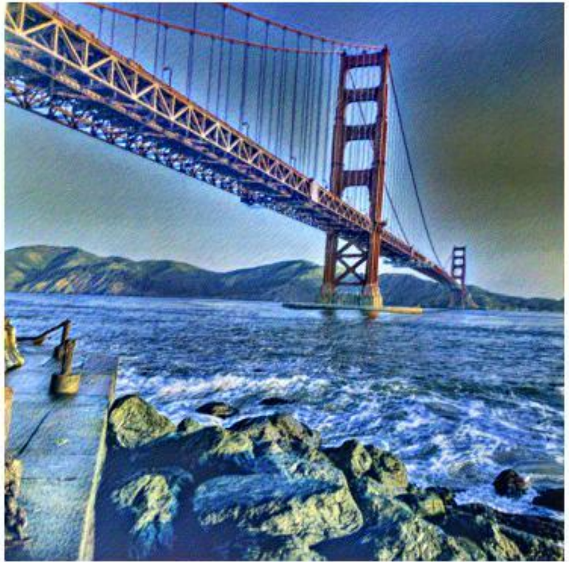
Output Image

1/1000

Results

Output Image

Output Image - a/b ratio - 0.001



Photorealistic vs Artistic Style Transfer

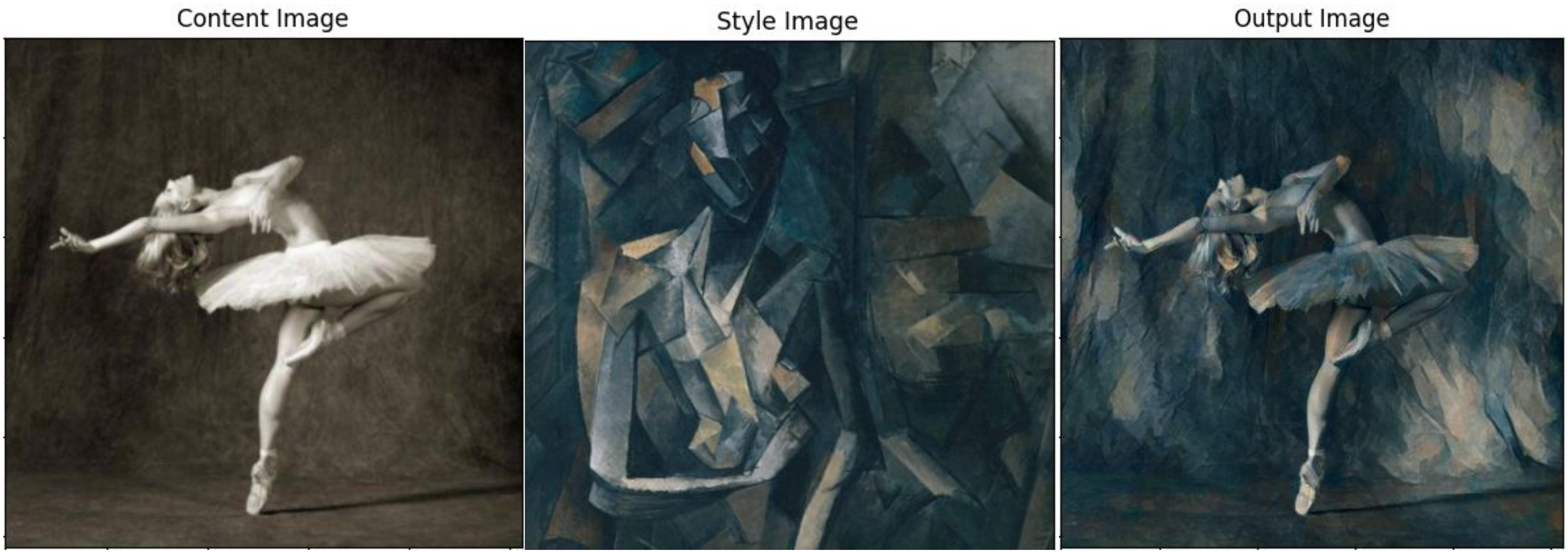
Artistic style transfer aims to create an artwork-like image, while photorealistic style transfer aims to create a photograph-like image.

In photorealistic, we prioritize realism over artistic expression and aesthetics. It involves using a GAN to generate the new image, which can be trained to minimize the difference between the generated image and a real photograph with a similar content and style.

Code Level Update

We have achieved the baseline as given in the paper. Github

Style Transfer Example



Results (Resnet vs Vgg19)

Resnet

vgg19

References

1. Image Style Transfer Using Convolutional Neural Networks
2. Neural Style Transfer: Everything You Need to Know
3. Pytorch