Neural Style Transfer

Group ID: 6

Group Members:	Yashvi Donga	16010420017
	Rohit Londhe	16010420054
	Rajvardhan Patil	16010420055

Guide: Dr. Sonali Patil

Knowledge Alone Liberates
ज्ञानादेव तु कैवल्यम्
Somaiya Vidyavihar
SOMAIYA
VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Somaiya
TRUST

Introduction

When it comes to tasks like object recognition and detection, the performance of deep neural networks has already surpassed that of humans. Using machine learning techniques to create art of higher quality is essential for achieving human-like capabilities and opens up new possibilities. Additionally, deep learning is currently being applied to the creation of art, thanks to the development of computer hardware and the proliferation of the technique.

Can we use artificial intelligence to support human creativity and discovery? Through this project we want to see how style transfer can be applied on an image using AI as a tool

Background work

- Neural Style Transfer is an algorithm can separate and recombine the image content and style of natural images. The algorithm produces a new images of high quality that combine the content image with the style image

Background

Sr no	Title	Publication Info	Model	Limitations	Key Takeaways (optional)
1	Neural map style transfer exploration with GANs	INTERNATIONAL JOURNAL OF CARTOGRAPHY2022, VOL. 8, NO. 1, 18–36	Pix2pix and cycleGANs	In some cases, the high density of visualinformation could generate confusion between cartographical objects for the models. training GANs is extremely computationally expensive: the generation of High Resolution images is only possible with very high end hardware and long training times.	
2	Image Style Transfer Using Convolutional Neural Networks	2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	CNN (VGG 19)	When synthesising an image that combines the content of one image with the style of another, there usually does not exist an image that perfectly matches both constraints at the same time. Both, the dimensionality of the optimisation problem as well as the number of units in the Convolutional Neural Network grow linearly with the number of pixels. Therefore the speed of the synthesis procedure depends heavily on image resolution. Another issue is that synthesised images are sometimes subject to some low-level noise.	For a specific pair of content and style images one can adjust the trade-off between content and style to create visually appealing images. The key finding of this paper is that the representations of content and style in the Convolutional Neural Network are separable. That is, we can manipulate both representations independently to produce new, perceptually meaningful images.
3	Neural Style Transfer Replication Project	Published by Princeton in 2018	VGG19	Same as last paper.	If we use maxpooling, which is the pooling layer used in VGG19 net, the loss function converges aroung 100 iterations. If we replace maxpooling with average pooling(suggested in the original paper) however, the algorithm will converge in around 15-30 iterations.
4	Neural Style Transfer: A Critical Review	IEEE Access (Volume: 9)	GANs and DCGANs	Hardware limitations are the primary cause of platform-related gaps. In the absence of a benchmark dataset and benchmark metrics, there exist data-related research gaps. Lastly, architecture-related gaps concern how model parameters change based on the dataset need.There is currently no benchmark dataset for Neural Style Transfer.	MS-COCO and Cityscape are the two datasets most frequently utilized in experiments.

Background

5	Generative Adversarial Networks	2014, arXiv:	GAN		Efficiency improvements: training could be accelerated greatly by divising better methods for coordinating G and D or determining better distributions to sample z from during training.
6	Image-to-Image Translation with Conditional Adversarial Networks	2018, arXiv	GAN		GAN can be used to generate photos from cityscape and facade labels. It can also be used to convert daytime photos to night time photos, and to generate pictures from traced edges. All these applications can be seen as variations of neural style transfer.
7	On convergence and stability of GANs	2017, arXiv:	GAN		This paper presents a way to avoid local minima called DRAGAN. DRAGAN enables faster training, achieves improved stability with fewer mode collapses, and leads to generator networks with better modeling performance across a variety of architectures and objective functions.
8	A Texture Analysis Method for Art Paintings	2003, Proceedings of the International Conference on Image Processing (ICIP)	-		The method is based on Gabor filters, which are used to extract texture features from art paintings. Gabor filters are designed to capture different aspects of texture, such as scale and orientation. The Gabor filter is defined by several parameters, including the scale, orientation, and spatial
9	Measuring Colorfulness in Natural Images	2014, IEEE Transactions on Image Processing	-		The method proposed in the paper computes a single score that takes into account the distribution of color information in the image. To do this, the method first splits the image into its red, green, and blue (RGB) channels. Then, it computes two chromatic channels, the rg and yb channels, which capture the differences in color information along the red-green and blue-yellow axes that the human visual system is most sensitive to. The paper notes that these two chromatic channels capture most of the color information in natural images and are

Problem Statement

In this project we aim to compare different ways to implement Neural Style Transfer. NST can be implemented using:

- CNN - Convolutional Neural Network
- GAN - Generative Adversarial Network

We also aim to find a way to measure the outputs of the 2 models. We aim to quantify the qualities like colour and texture of an image which in turn help compare the performance of the models.

Scope - Functional Requirements

- **User requirements:** Style and content image.
- **Perform Neural Style Transfer using CNN model**
  Performing neural style transfer using a Convolutional Neural Network model is an approach where system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images
- **Perform Neural Style Transfer using GAN model**
  Performing neural style transfer using a Generative Adversarial Network (GAN) model is a novel approach that has shown promising results in generating visually appealing images.
- **Compare the results of CNN and GAN model**
  Comparing the results of CNN and GAN models is an important step in evaluating the performance of neural style transfer techniques. While CNNs are widely used in style transfer, GANs have been shown to produce more realistic and diverse images.

Scope - Non - Functional Requirements

- **Performance:** Ensuring that the style transfer process is completed efficiently and within a reasonable amount of time, especially when handling large or high-resolution images.
- **Compatibility and Responsiveness:** Ensuring that different images of different types can be processed.

Datasets

- **Dataset: I'm something of a Painter Myself**
- The dataset contains 2 directories: monet_jpg, and photo_jpg.
- The monet directories contain Monet paintings. Use these images to train your model.
- The photo directories contain photos. Add Monet-style to these images and submit your generated jpeg images as a zip file.
- monet_jpg - 300 Monet paintings sized 256x256 in JPEG format
- photo_jpg - 7028 photos sized 256x256 in JPEG format

Technologies used

- Programming Language:
    - Data Visualization Libraries:
        - MatplotLib
        - IPython
    - Image Processing Libraries:
        - OpenCV
    - Python ML Libraries :
        - Tensorflow
        - Keras
        - NumPy
        - SciPy
- Python IDE:
    - Google Colab
    - Kaggle

Overview of Implementation

CNN - Convolutional Neural Network Model

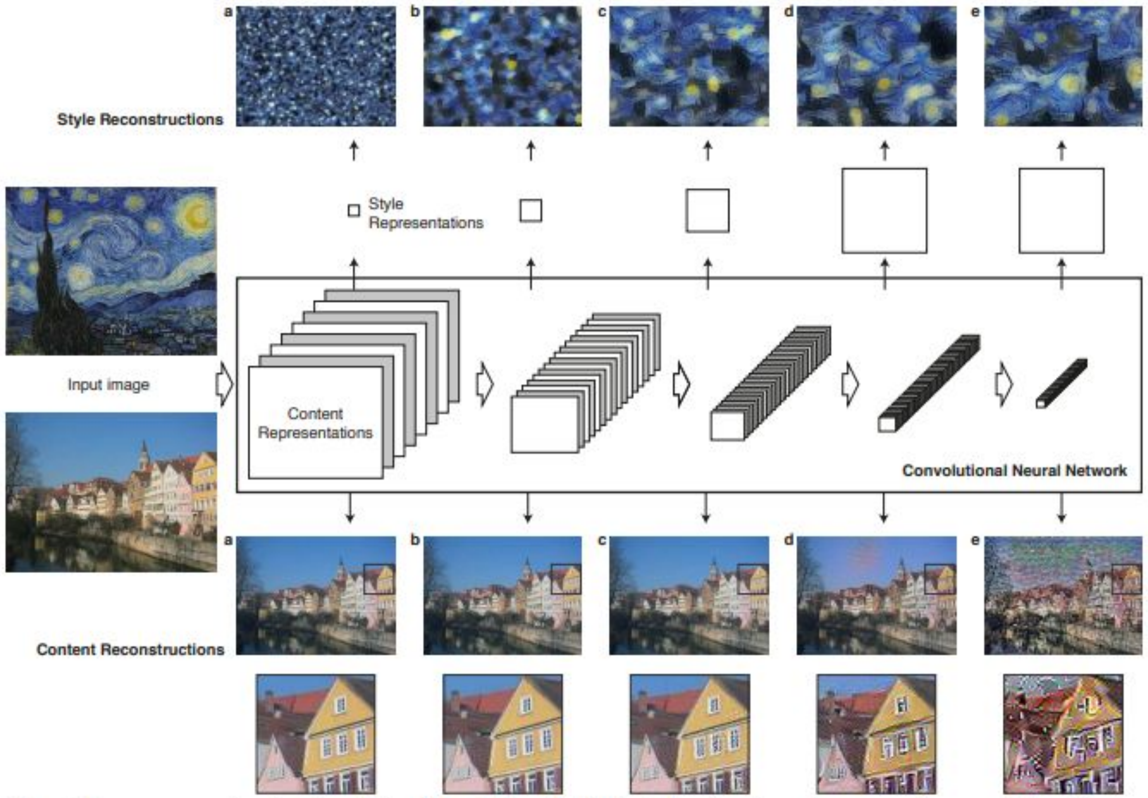
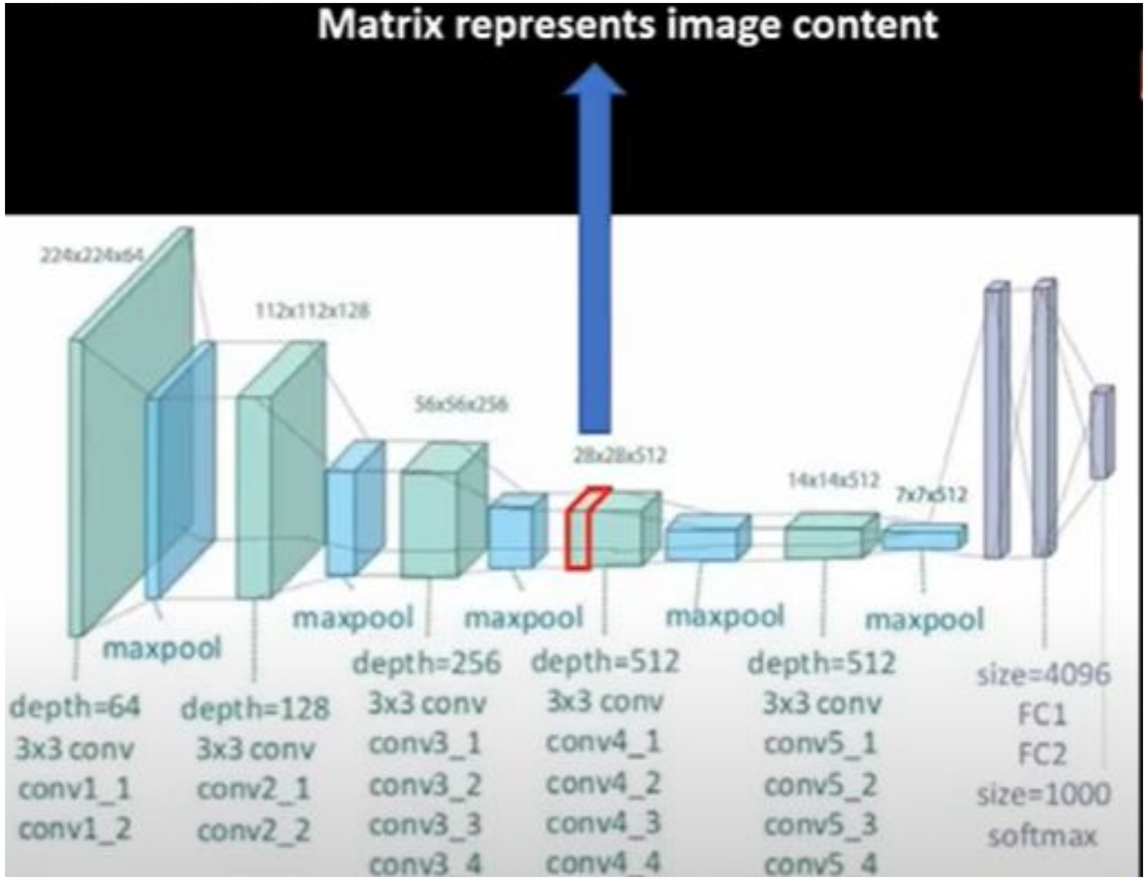


Figure 1. Image representations in a Convolutional Neural Network (CNN) [1]

Overview of Implementation

Content representation



Style representation

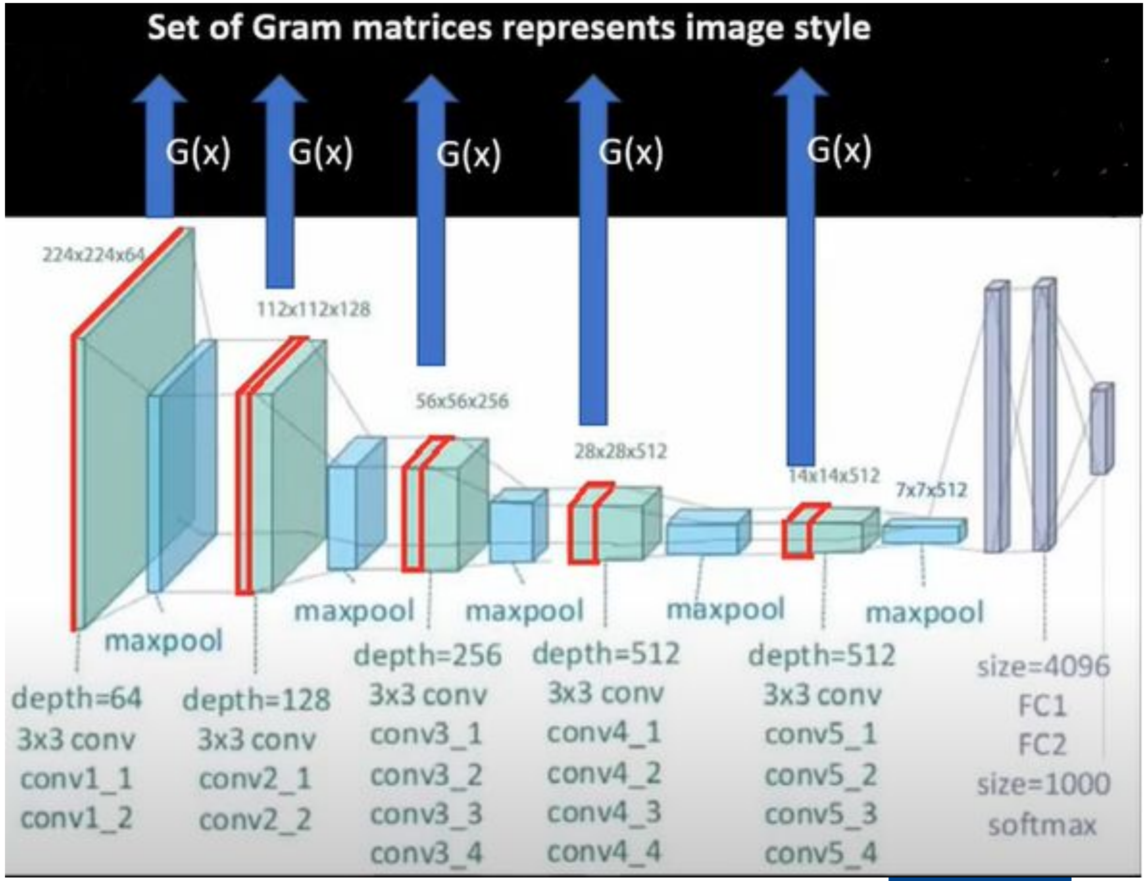


Figure II. VGG - 19

Overview of Implementation

- CNN - Convolutional Neural Network Model
  - For our implementation, we have used the VGG-19 network
  - To perform neural style transfer, the VGG-19 network is used to extract features from the content image and style image separately.
  - The feature maps from the content image and style image are then compared using a loss function that includes both the content loss and the style loss.
  - For optimization - Adam optimizer is used. During optimization, the input image is updated to minimize this loss function, which results in an image that combines the content of the content image with the style of the style image.

Overview of Implementation

- CNN - Convolutional Neural Network Model
  - The research paper[1] implementation was done using the LUA language. Whereas this implementation is in Python Language.
  - The research paper[1] using L-BFGS optimizer wheres as our implementation using Adam optimizer. Adam optimizer has been found to be effective for neural style transfer due to its ability to capture the content of an image more accurately, especially when the content image has a lot of details.

Overview of Implementation

- Loss Function
  - The style_loss function, which keeps the generated image close to the local textures of the style reference image
  - The content_loss function, which keeps the high-level representation of the generated image close to that of the base image
- Hyper parameters

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$$

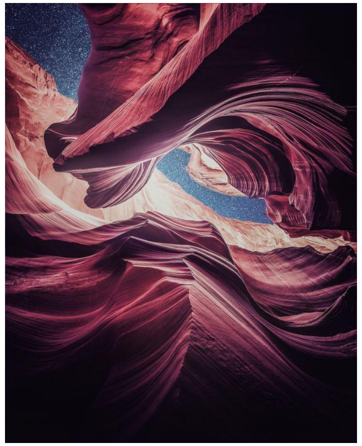
  - style_weight = 1e-2
  - content_weight = 1e4
- Adam Optimizer
  - learning_rate=0.02

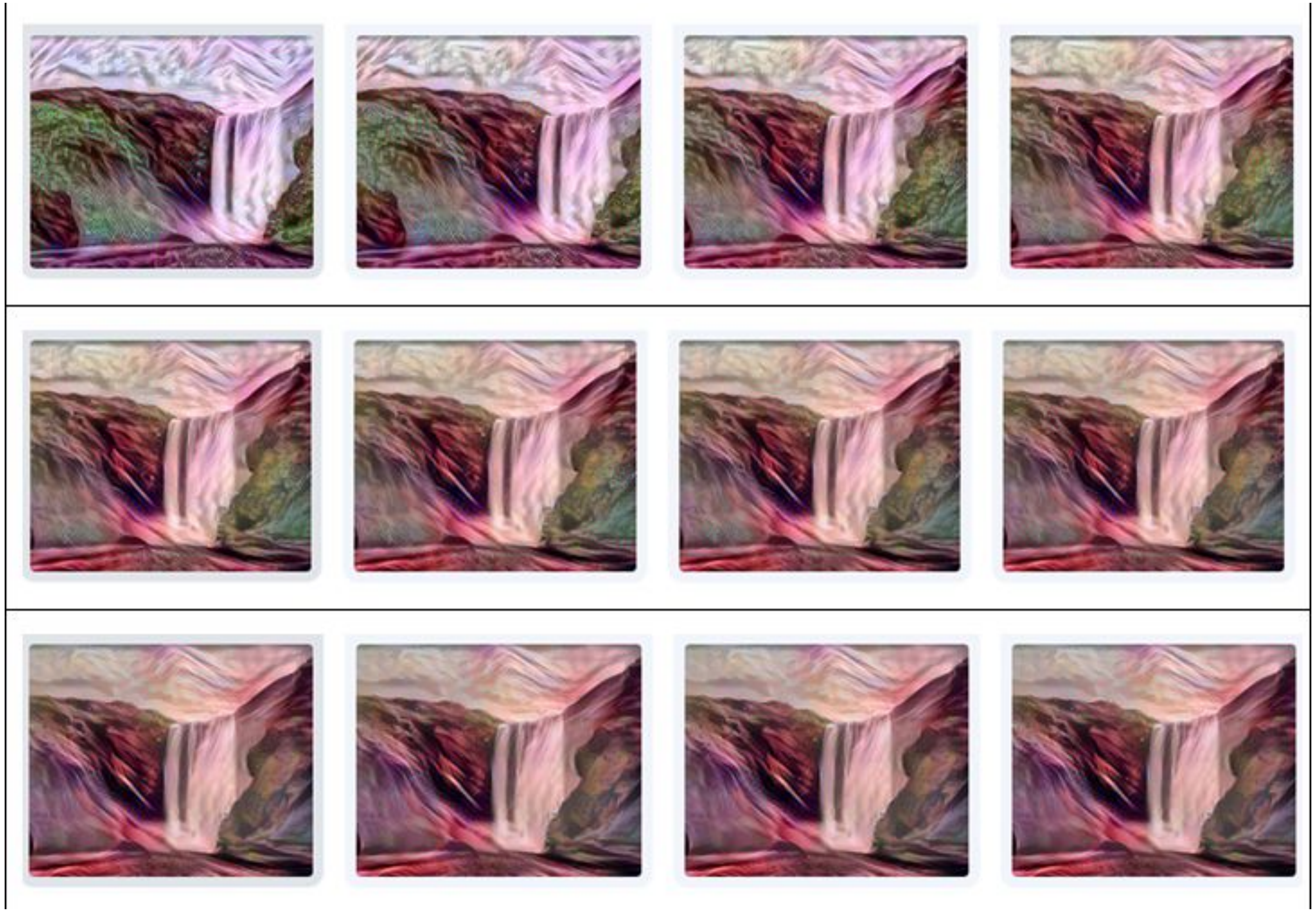
Overview of Implementation

- CNN - Convolutional Neural Network Model
    - Training Phase(Total Epochs = 1000, Image after every 50 epoch)

Content Image

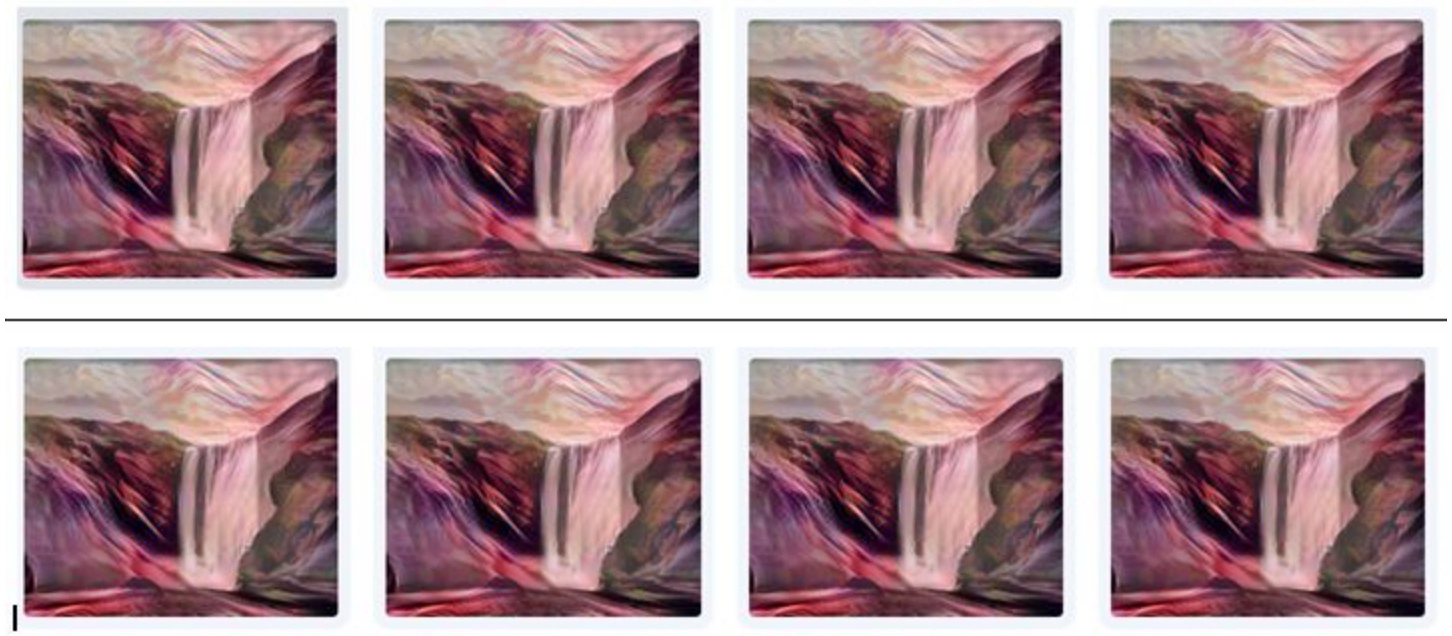
Style Image





Overview of Implementation

- CNN - Convolutional Neural Network Model
  - Training Phase(Total Epochs = 1000, Image after every 50 epoch)



Result Image:

Overview of Implementation

- GAN - Generative Adversarial Network.

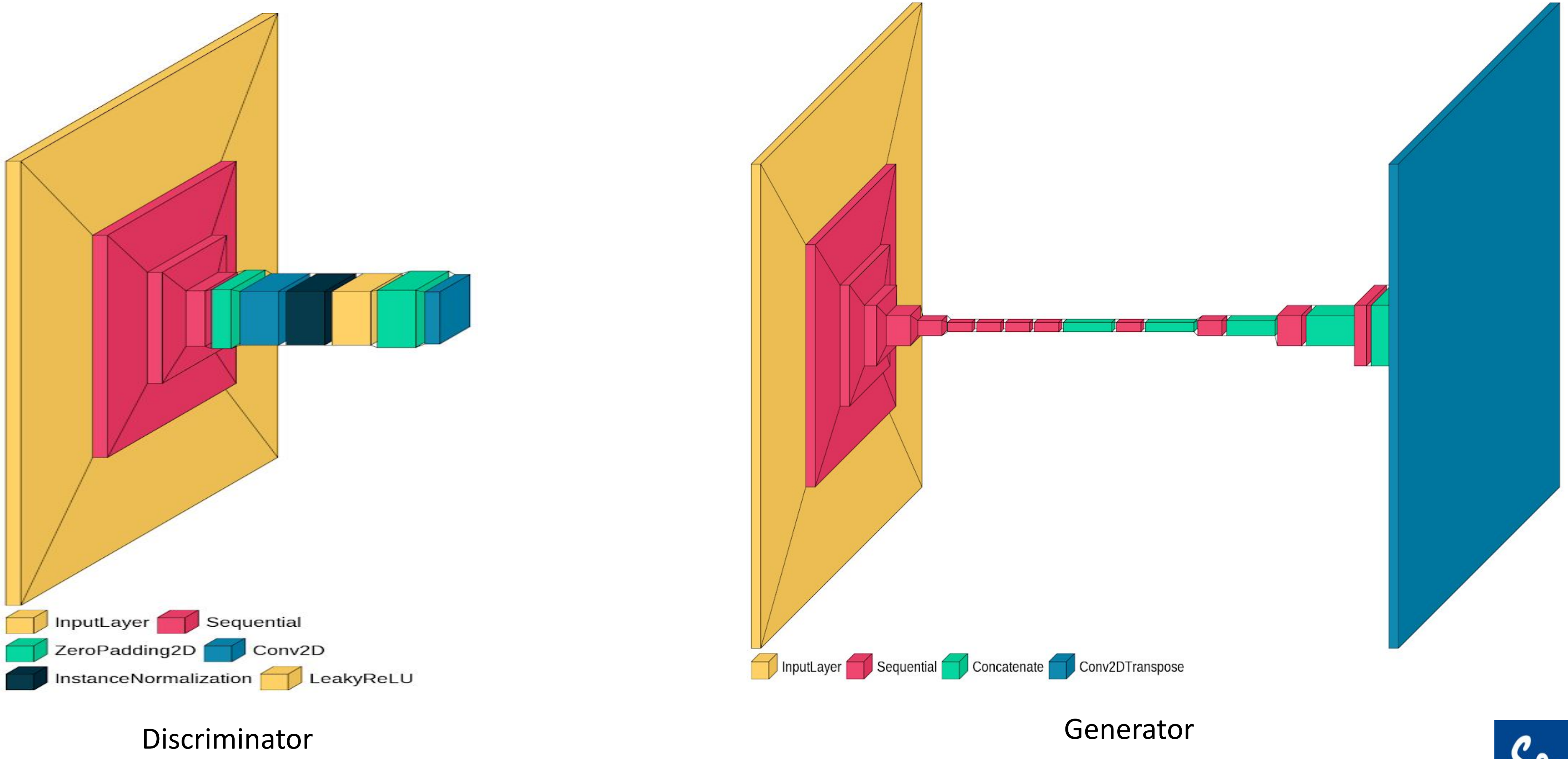


Figure 2. Working of a CycleGAN model [6]

Overview of Implementation

- GAN - Generative Adversarial Network
  - The CycleGAN model consists of two generators and two discriminators.
  - The generators, monet_generator and photo_generator, are responsible for transforming images from the source domain to the target domain and vice versa. The discriminators, monet_discriminator and photo_discriminator, are responsible for distinguishing between real images and fake/generated images in their respective domains.

Overview of Implementation

- GAN - Generative Adversarial Network
  - Generator:
    - 8 downsampling layer and 8 upsampling layers (stride length=2).
    - Skip connections generated.
    - Skip connections help overcome vanishing gradient problem.
    - input : [256, 256, 3]
    - output : [256, 256, 3]

Overview of Implementation

- GAN - Generative Adversarial Network
  - Discriminator:
    - Convolutional layers followed by an output layer.
    - ReLU activation
    - The output of this layer is a single scalar value that represents the probability of the input image being a real image.
    - input: [256 , 256, 3]
    - output: single value

Overview of Implementation

- GAN - Generative Adversarial Network
  - During training, the generators and discriminators are optimized alternately. The generators aim to fool the discriminators by generating realistic images, while the discriminators aim to correctly classify the images as real or fake.
  - Binary Cross entropy is used for discriminator and generator.
  - Difference between original and twice transformed image is the cycle consistency loss.

Color Score

- The code is designed to calculate the colorfulness score of an image.
- The colorfulness score measures how colorful an image is, based on the range of chromatic colors present in the image.
- The colorfulness score is calculated by computing the mean and standard deviation of the chromatic channels (rg and yb) and combining them into a single score in the paper.
- The rg chromatic channel measures the red-green difference, while the yb chromatic channel measures the blue-yellow difference.
- The higher final colorfulness scores indicating a more colorful image.

colorfulness = std_root + (0.3 * mean_root)

Color Score

```
# Split the image into its RGB channels
    (B, G, R) = cv2.split(image.astype("float"))

    # Compute rg and yb, the chromatic channels
    rg = np.absolute(R - G)
    yb = np.absolute(0.5 * (R + G) - B)

    # Compute the mean and standard deviation of both chromatic channels
    (rb_mean, rb_std) = (np.mean(rg), np.std(rg))
    (yb_mean, yb_std) = (np.mean(yb), np.std(yb))

    # Combine the mean and standard deviation into a single score
    std_root = np.sqrt((rb_std ** 2) + (yb_std ** 2))
    mean_root = np.sqrt((rb_mean ** 2) + (yb_mean ** 2))
    colorfulness = 10 - (std_root + (0.3 * mean_root))
```

Texture Score

- The code implements a texture scoring algorithm based on the Gabor filter bank.
- The Gabor filter bank is generated by varying the filter parameters such as orientation, scale, wavelength, and aspect ratio.
- The input image is filtered using the generated filter bank.
- The texture score is computed by taking the mean value of the filtered image for each filter and adding it to the overall score.
- The texture score represents the strength of texture in the input image across different orientations and scales.

Texture Score

```
# Calculate the Gabor filter bank
    filters = []
    ksize = 31
    for theta in np.arange(0, np.pi, np.pi / num_orientations):
        for sigma in np.logspace(0, np.log10(ksize), num_scales):
            for lamda in np.logspace(-1, 1, num_scales):
                for gamma in np.linspace(0.2, 1.0, num_scales):
                    kernel = cv2.getGaborKernel((ksize, ksize), sigma, theta, lamda, gamma,0,ktype=cv2.CV_32F)
                    kernel /= np.sum(kernel)
                    filters.append(kernel)

    # Apply the Gabor filters to the image and calculate the texture score
    for kernel in filters:
        filtered = cv2.filter2D(img, cv2.CV_32F, kernel)
        score += np.mean(filtered)
```

Result and Discussion

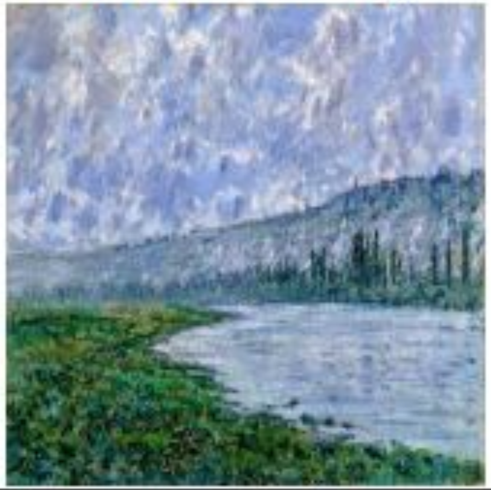
- CNN Model Result

Content Image	Style Image	Result Image	Scores
			Colourfulness score: -38.42 Texture score: 152.01
			Colourfulness score: -36.97 Texture score: 119.72
			Colourfulness score: -37.35 Texture score: 39.84

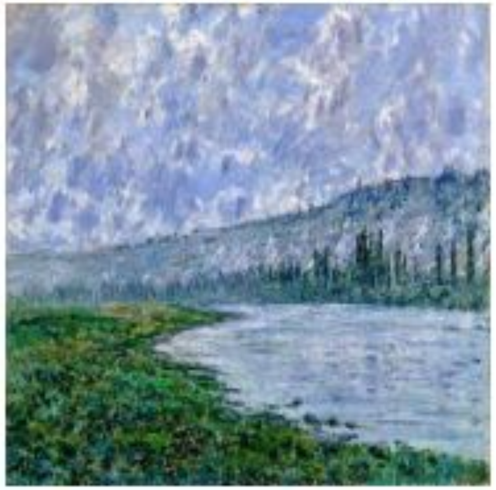
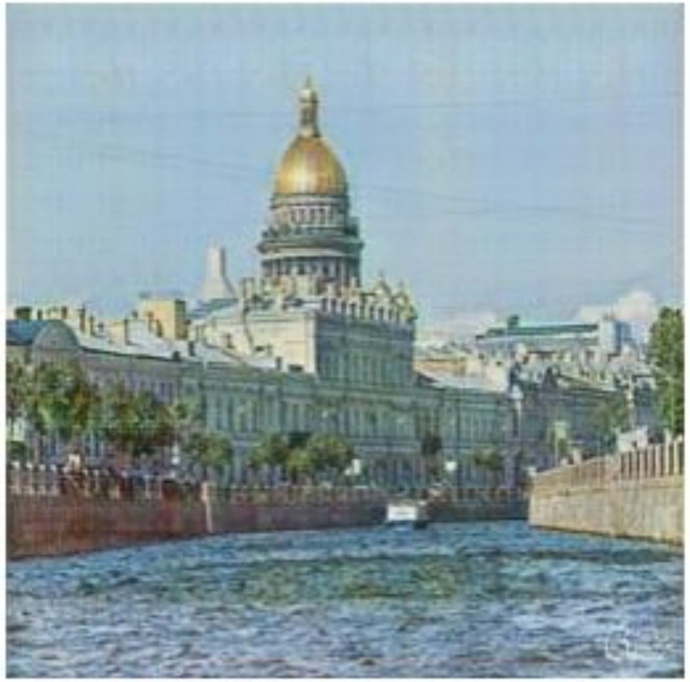
- CNN Model Result

Content Image	Style Image	Result Image	Scores
			Colourfulness score: -22.46 Texture score: 51.8
			Colourfulness score: -22.78 Texture score: 67.66
			Colourfulness score: -19.04 Texture score: 64.00

- GAN Model Result for Monet Paintings

Content Image	Monet Paintings	Result Image
	 	 Colourfulness score: -15.80 Texture score: 141.01

- GAN Model Result for Monet Paintings

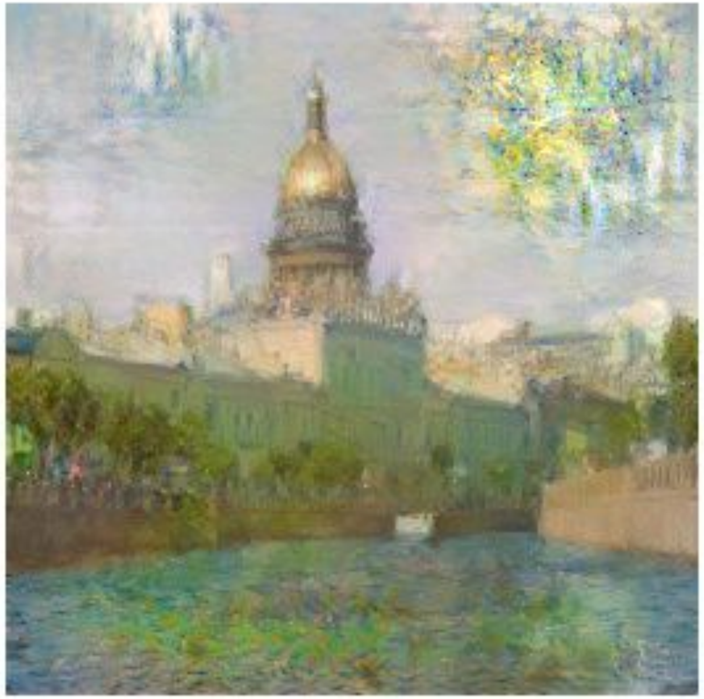
Content Image	Monet Paintings	Result Image
	 	 Colourfulness score: -13.20 Texture score: 78.56

- GAN Model Result for Monet Paintings

- Result Comparison using Colourfulness and Texture Scores

Content Image

Style Image

CNN Model Result	GAN Model Result
	
Texture score: 67.66	Texture score: 78.56
Colourfulness score: -22.78	Colorfulness score: -13.20

- Result Comparison using Colourfulness and Texture Scores

Content Image

Style Image

CNN Model Result	GAN Model Result
	
Texture score: 51.58	Texture score: 141.01
Colourfulness score: -22.46	Colorfulness score: -15.80

- Result Comparison using Colourfulness and Texture Scores

Content Image

Style Image

CNN Model Result	GAN Model Result
	
Texture score: 64.00	Texture score: 147.20
Colourfulness score: -19.04	Colorfulness score: -9.35

Conclusion

- Neural Style Transfer (NST) is a powerful technique that combines the content of one image with the style of another, resulting in visually appealing and unique images.
- While comparing the two implementation - CNN and GAN, we gained a lot insights into their respective strengths and weaknesses.
- Once of the biggest difference between the two implementation is that CNN only requires one style image, where as GAN requires multiple image of same style. Due to this GAN model tends to pick up more texture of the artwork than CNN.
- From the scoring measures we found, we saw that the that GAN result has a higher texture and colourfulness score than the CNN result.
- This project can serve as a starting point for exploring the potential of NST and for identifying the best approaches for generating high-quality stylized images.

Learnings

- Understanding and Implementing Research Paper
- Collaborating with Google Colab and Kaggle
- Implementing CNN model using a style image and a content image
- Implementing GAN model using dataset of Different painters
- Explored ways to measure the texture and color of an image

Implementation Schedule

Task	February				March				April			
	1st	2nd	3rd	4th	1st	2nd	3rd	4th	1st	2nd	3rd	4th
NST using CNN model												
NST using GAN model												
Developing an evaluation framework												

Thank You

References

[1] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, "Image Style Transfer Using Convolutional Neural Networks", IEEE, 2016.
[2] Sidonie Christophe, Samuel Mermet, Morgan Laurent & Guillaume Touya, "Neural map style transfer exploration with GANs", International Journal of Cartography, Jun 30, 2021
[3] https://keras.io/examples/generative/neural_style_transfer/
[4] A Texture Analysis Method for Art Paintings, Proceedings of the International Conference on Image Processing (ICIP), 2003 - Anirban Banerjee, Swagatam Das, Malay K. Kundu, and Dipak K. Basu.
[5] Measuring Colorfulness in Natural Images, IEEE Transactions on Image Processing in 2014 - K. Radhakrishnan and A. C. Bovik
[6] https://blog.jaysinha.me/train-your-first-cyclegan-for-image-to-image-translation/