Image Style Transfer with (Fast) Neural Style Transfer Model

Team Big Cow King
Feiran (Philip) Huang, Jingwen (Steven) Shi, Hongsheng Zhong

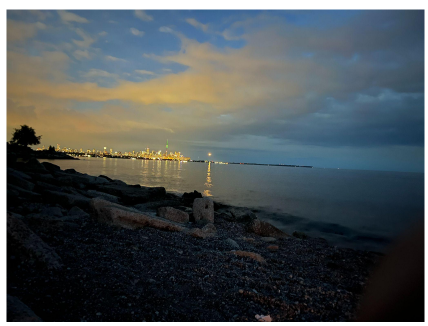
Machine Vision Problem

The aim of this project is to implement a Neural Style Transfer Model (NST) for transferring the art style of a chosen image to another image. This can make people in the future still be able to see the style of old artworks even if the original masterpiece has been lost or broken.

Dataset

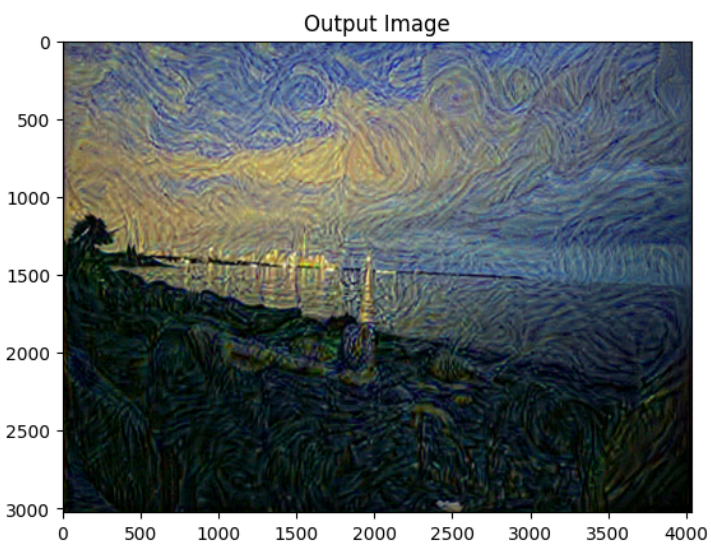
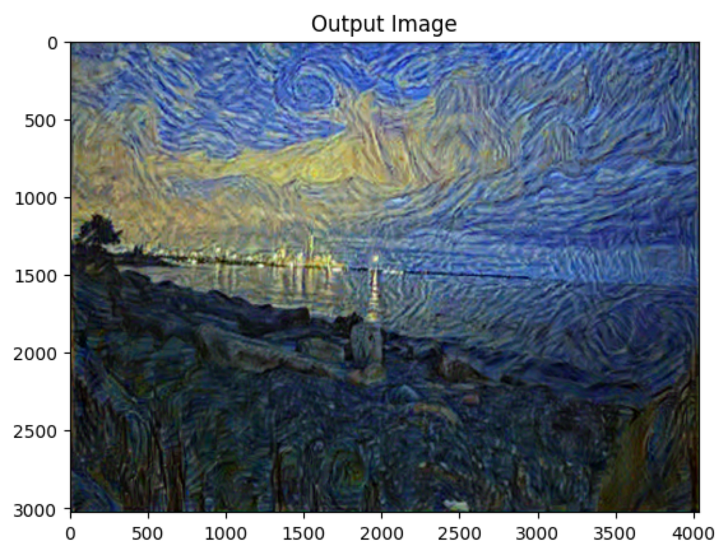
COCO 2017 Dataset: A comprehensive collection of real-life photos from the COCO (Common Objects in Context) dataset, containing various scenes, objects, and contexts captured in everyday life.
Best Art of All Time: Artworks encompassing various styles, genres, and historical periods, curated as a representative selection of the finest art across different cultures and movements.

Both datasets' images serve as content images and style images.

- **Example Input**:
    - Content Image: 
    - Style Image: 
- **Example Output**:

Implementation Decisions, Improvements, and Innovation Approach:

- Detaching Target
  - Content and style images are passed into the model during the generalization of the output image and have been detached from the code. This is because the content and style image should not be updated since we only want to update the generated image to minimize the loss. By calling the detach function, the content and style feather maps will be removed from the computation graph and remain unchanged for the rest of the computation.
- Improve Computation Effenficy
  - The original paper has chosen to insert the style/content layers after a set of convolution layers. However, we found that the image generated by inserting the loss layer after the ReLU activation function can accelerate the image-generating process much faster.
  - Since inserting after ReLU layers will converge much faster, it produces a better quality and visually more appealing image than convolutional layers under the same number of steps taken.

Conv (60 steps with 3s)	ReLU (60 steps with 3s)
	

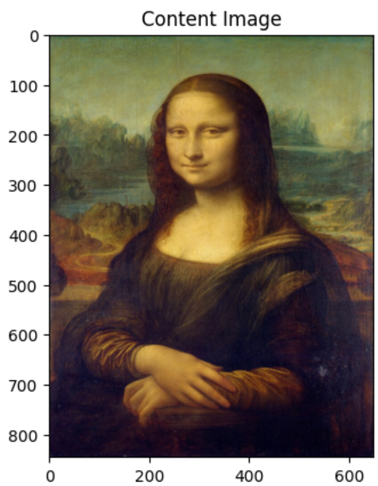
With VGG 19 Layer 1 - 5	With VGG 19 Layer 1,3,9,13,28
	

In neural style migration, focusing on basic image features such as edges and texture using layers 1-5 of a network such as VGG19 may overemphasize texture. By choosing a mixture of layers such as layers 1, 3, 9, 13, and 28, both simple and complex patterns can be obtained from the network. Ensure that the image retains the overall look and structure of the content image while capturing complex details from the style image.

- Using Gram Matrix over Covariance Matrix
  - In the covariance matrix, the mean of each feature is subtracted from the feature values which center the data with respect to 0. However, in image processing, these features may be related to pixel intensities (i.e. RGB values) and centers data around the mean will capture the intensity of one feature related to the presence of another.
  - The Gram matrix does not subtract from the mean which only focuses on the overall frequency of feature co-occurrences. Elements like brushstrokes and colour palettes are parts of the style that should not depend on the positioning of objects. Therefore, the ability that the Gram matrix can abstract away from these spatial relationships and only focus on the overall presence of the feature maps, is exactly what we need.

Unsolved Problems:

- Some styles can not be transferred to the content image. The reason behind that is not clear. This usually happens when:
    1. The colours or main objects of the two images are too different.
    2. There is not much obvious line or texture in the style image.

Content	Style	Output
		

Evaluation Results:

1. ReLU Converge Speed vs Conv Converge Speed:

	Conv		ReLU	
Step	Style Loss	Content Loss	Style Loss	Content Loss
10	2377.650879	43.962997	17.4932	3.700711
20	682.897217	51.401409	5.970345	3.185289
30	307.438232	49.848648	3.40082	2.763851
40	184.816498	49.311531	2.252596	2.464281
50	126.953033	46.515888	1.753494	2.267855
60	98.597458	44.977612	1.54337	2.128388
70	75.24659	42.910976	1.392532	2.032228
80	61.750702	41.522079	1.292523	1.952586
90	51.640327	40.076035	1.203516	1.90231
100	45.359356	38.968891	1.147376	1.844173