Style Transfer

Team Athenians

Pranav Kirsur: 2018101070
Avani Gupta: 2019121004
Pranav Tadimeti: 2018101055
Nivedita Rufus: 2019702002

Mentor TA: Gowri Lekshmy

Overview

Objectives

- Transfer any arbitrary visual styles to content images (Style Transfer)
- Allow user control on the amount of stylization.

Challenges

Preserving the actual content of the image, efficiency, quality of the output images

Universal Style Transfer via Feature Transforms

Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., & Yang, M. H. (2017).

arXiv preprint arXiv:1705.08086.

The paper Universal Style Transfer via Feature Transforms applies feature transforms like whitening and coloring which are further embedded to an image reconstruction network in order to perform style transfer on images.

Goal

Content

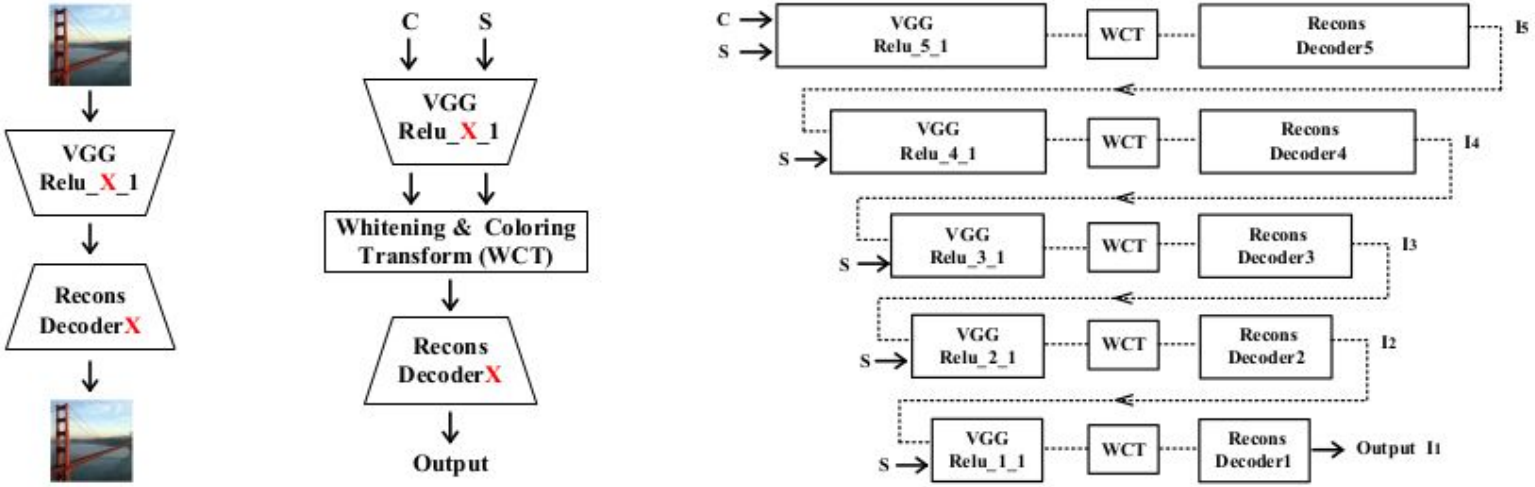
Style

Resultant image

Methodology

- The paper proposes to use **feature transforms**: whitening and coloring to directly match content feature statistics to those of a style image.
- The feature transforms are coupled with a pre-trained general encoder-decoder network, so that the transfer is done via feed-forward operations.
- Thus they do style transfer via an image reconstruction process coupled with feature transformations as above.
- The reconstruction part is responsible for inverting features back to the RGB space and the feature transformation matches the statistics of a content image to a style image.

Pipeline



(a) Reconstruction (b) Single-level stylization (c) Multi-level stylization

Pipeline (contd.)

- Pre-train five decoder networks Decoder (X=1,2,...,5) through image reconstruction to invert different levels of VGG features.
- With both VGG and Decoder X fixed, and given the content image C and style image S, perform the style transfer through whitening and coloring transforms.
- Extend single-level to multi-level stylization in order to match the statistics of the style at all levels.
- The result obtained by matching higher level statistics of the style is treated as the new content to continue to match lower-level information of the style.

Reconstruction Decoder

- VGG-19 used as an encoder.
- Decoder network is trained for inverting VGG features to the original image.
- Designed as being symmetrical to VGG-19 network (up to Relu_X_1 layer), with the nearest neighbor upsampling layer used for enlarging feature maps.
- Pixel reconstruction loss and feature loss are employed for reconstructing an input image.

$$L = \|I_{output} - I_{input}\|_2^2 + \lambda \|\Phi(I_{output}) - \Phi(I_{input})\|_2^2$$

Feature Transforms

Extract vectorized VGG features maps f_c and f_s from content image and style image respectively. Then we apply Whitening and coloring transforms as follows:

Whitening Transform

$$\hat{f}_c = E_c \, D_c^{-\frac{1}{2}} \, E_c^\top \, f_c$$

Where D_c is the diagonal matrix with the eigenvalues of the covariance matrix $f_c \, f_c^\top \in \Re^{C \times C}$,

and E_c is the corresponding orthogonal matrix of eigenvectors, satisfying $f_c \, f_c^\top = E_c D_c E_c^\top$

Coloring Transform

$$\hat{f_{cs}} = E_s \, D_s^{\frac{1}{2}} \, E_s^\top \, \hat{f}_c$$

Where D_s and E_s are calculated in same manner as above.

$$\hat{f_{cs}} = \hat{f_{cs}} + m_s$$

Multi-level coarse-to-fine stylization

- Features of deeper layers capture more complicated local structures.
- Features of initial layer carry more low-level information.
- Apply WCT at latter layer to obtain coarse stylized image and consider it as new content image to further adjust features in initial layers.

User Controls

- User can control scale, weight and spatial features of style:
  - Scale can be controlled by varying size of input style image.
  - Spatial control is provided via masks for specific regions and styles.
  - Weight is controlled by the style weight α in the feed-forward passes

Work done till mid - eval

- Single Level Stylisation
- Modules coded:
  - Dataloader
  - Reconstruction Decoder
  - WCT transforms

Work done after mid - eval

- Completed full multi-level pipeline
- Added user control:
  - Scale
  - Spatial control
  - Weight

Method of Experimentation

- Tried stylisation on various style and content images combinations.
- Kept style constant and varied content image.
- Kept content image constant and varied styles.
- Tried user control though scale, spatial control through masks, and different weights(values of alpha)

Single Level Stylization Results

Content Image

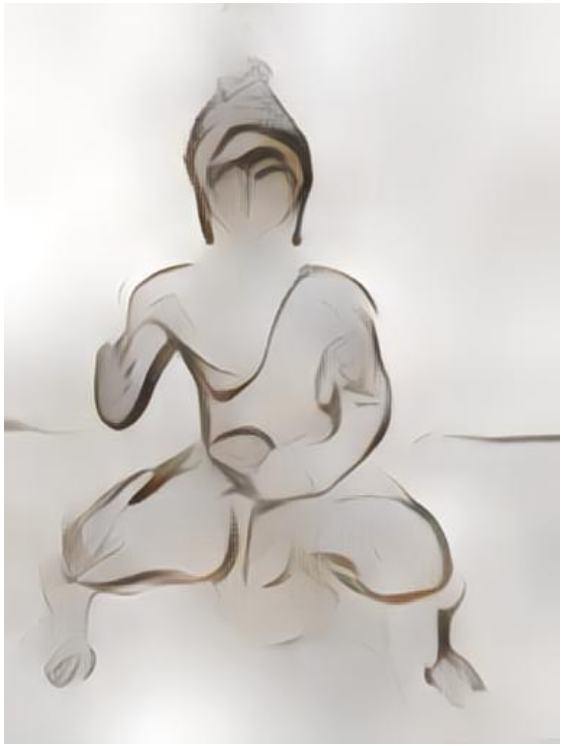
Style

Transformed image

Content Image

Style

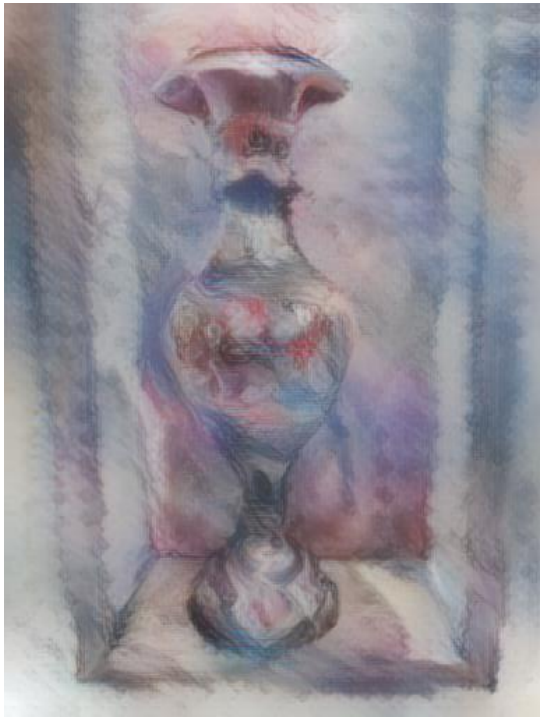
Transformed image



Content Image

Style

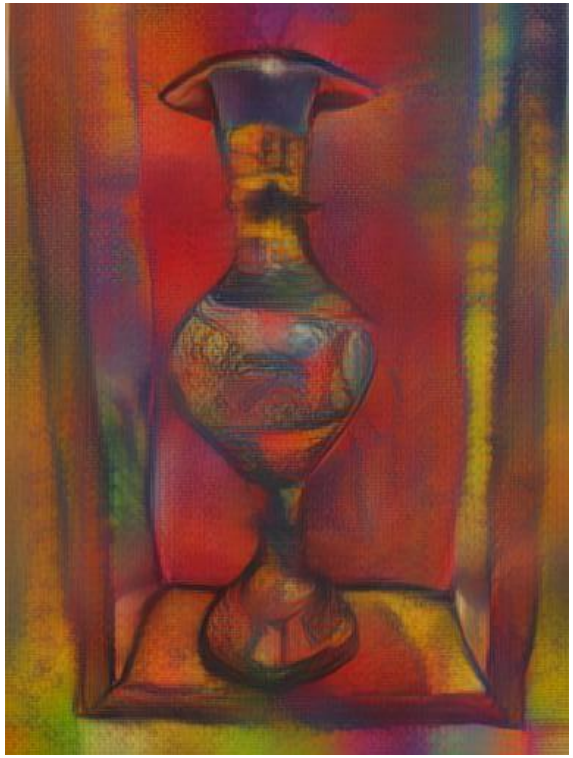
Transformed image



Content Image

Style

Transformed image



Multi Level Stylization Results

Content Image	Style	Transformed image
		

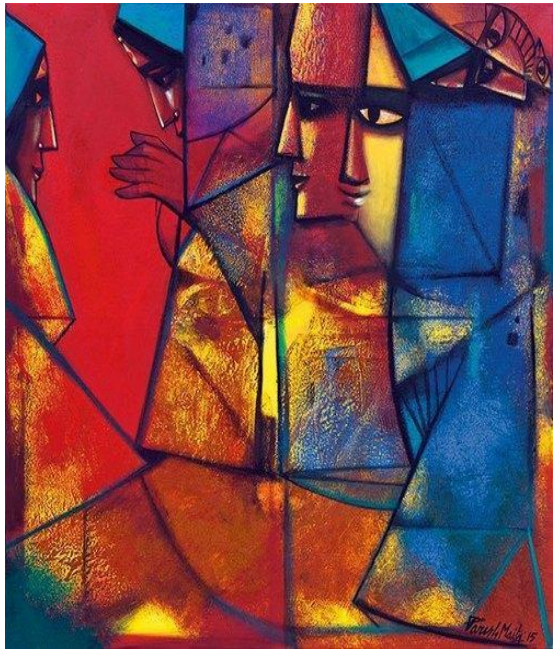
Content Image

Style

Transformed image

Content Image

Style



Transformed image

Content Image

Style

Transformed image

Content Image

Style

Transformed image

Comparing Single Level and Multi Level Stylization Results

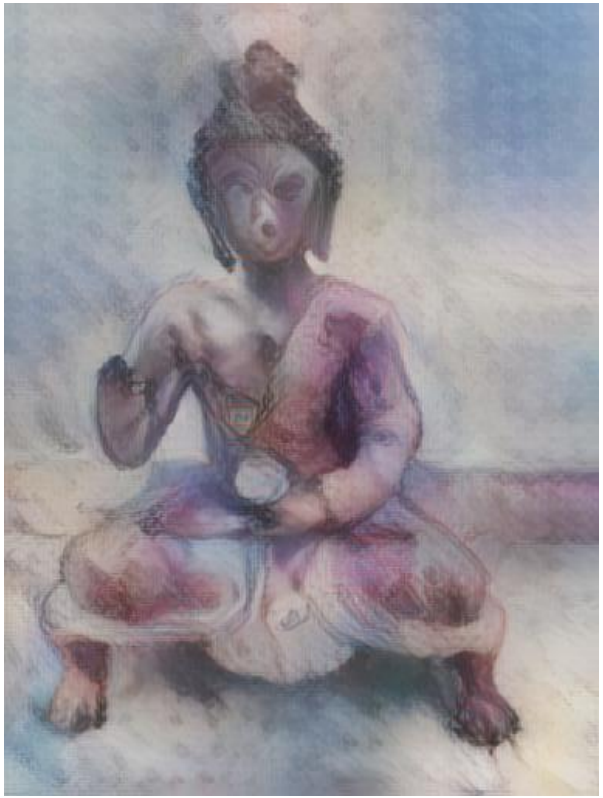
Single Level

Multi Level

Single Level

Multi Level

Single Level



Multi Level

Single Level

Multi Level

Experiment:

Vary the scale of style image: basically change size of style image

For Single Level stylization

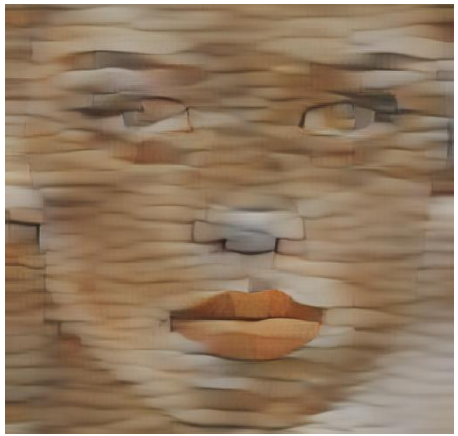
Content Image

Style

scale:
256

512

768



Content Image

Style

scale:
256

512

768

Content Image

Style

scale: 256

512

768

Content Image

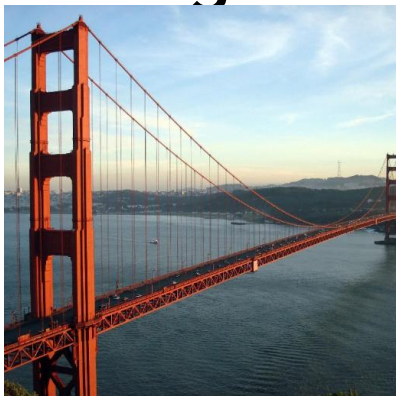
Style

scale: 256

512

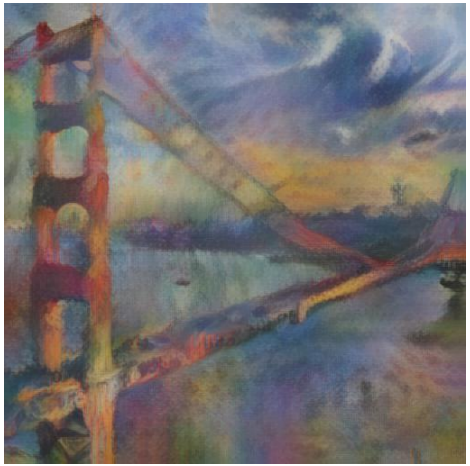
768

Content Image

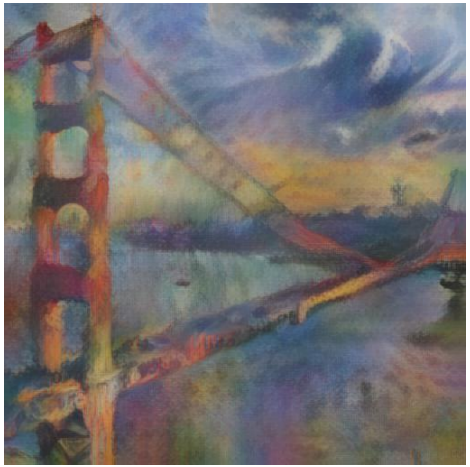


Style

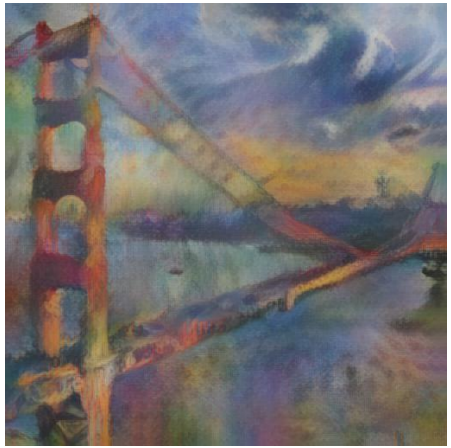
scale: 256



512



768



For Multi-Level stylization

Content Image

Style

scale: 256

512

768

Content Image

Style

scale: 256

512

768

Content Image

Style

scale: 256

512

768

Content Image

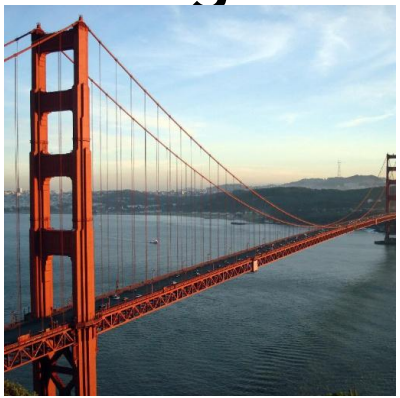
Style

scale: 256

512

768

Content Image



Style

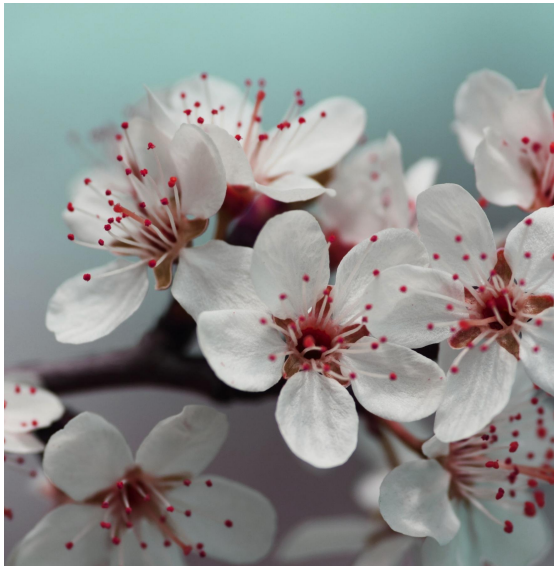
scale: 256

512

768

Spatial Control provided via masks

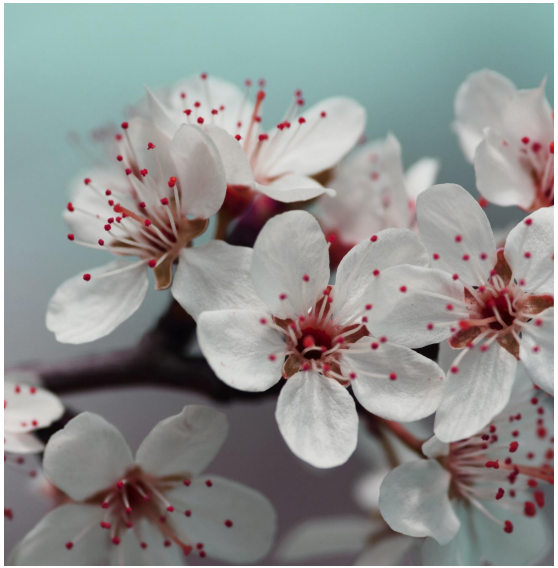
Content Image



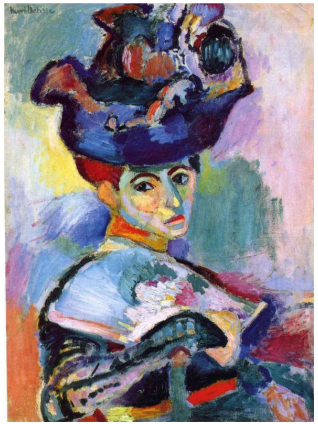
Style

Transformed image

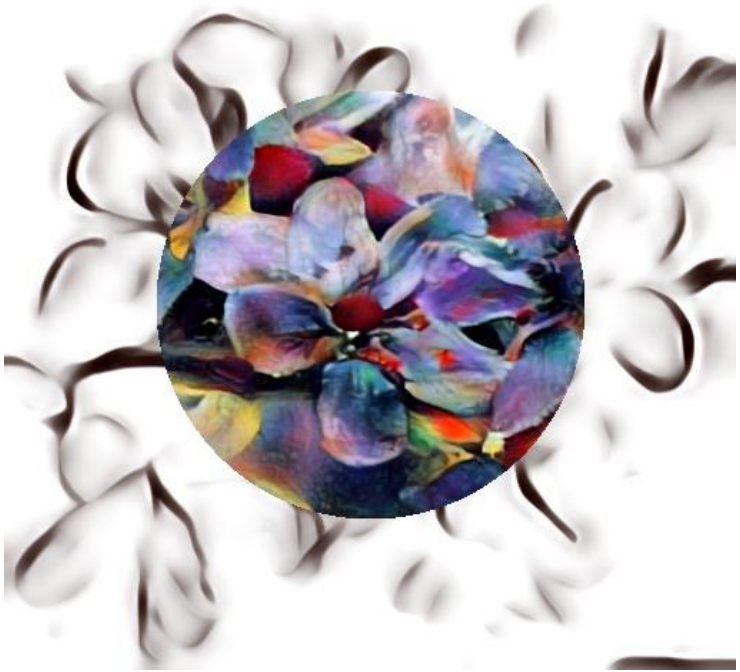
Content Image



Style



Transformed image



Changing the weight of style image (alpha)

Alpha = 1

Alpha = 0.6

Alpha = 0.4

Alpha = 1

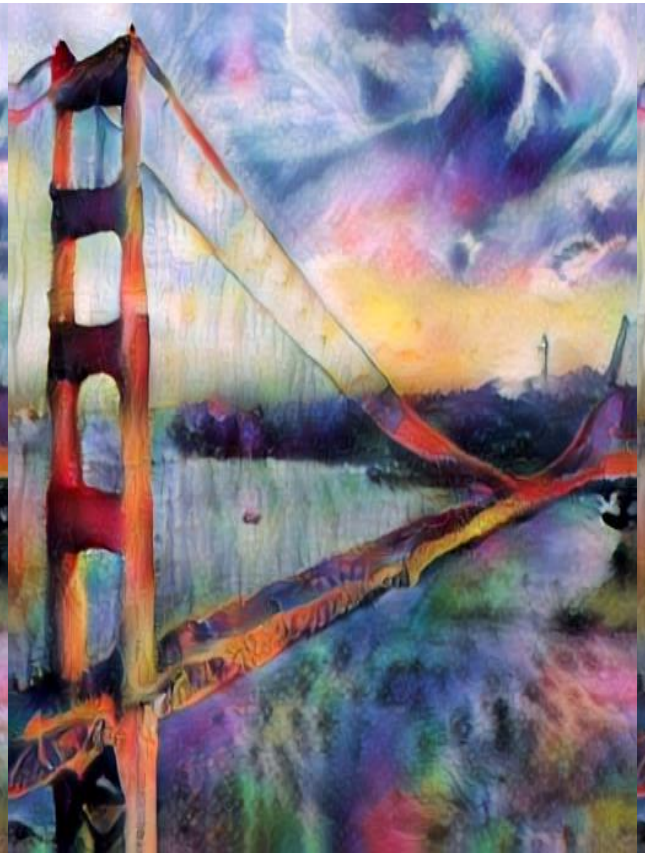
Alpha = 0.6

Alpha = 0.4

Alpha = 1

Alpha = 0.6

Alpha = 0.4



Alpha = 1

Alpha = 0.6

Alpha = 0.4

Frameworks and Libraries

- Python (Pytorch)
- OpenCV

Thank You!