Universal Style Transfer via Feature Transforms

Team - 27

Statistical Methods in Artificial Intelligence

The Team

Sreya Mittal - 201501058

Abhijith Nair - 201501169

Kulin Shah - 201501234

Tirth Maniar - 201501051

TA Mentor : Sairam Kolla

Professor : Dr. Vineet Gandhi

The Project

- Transfer any arbitrary visual styles to content images(Style Transfer).
- Can be used for texture synthesis.
- Allows user controls on the amount of stylization.
- Simple yet effective method.
    - Efficient.
    - Better results.
    - Does not require learning for each stylization.

Challenges

- Extract effective representations of style
- Preserve actual content
- Generalization to unseen styles
- Efficiency
- Quality of output

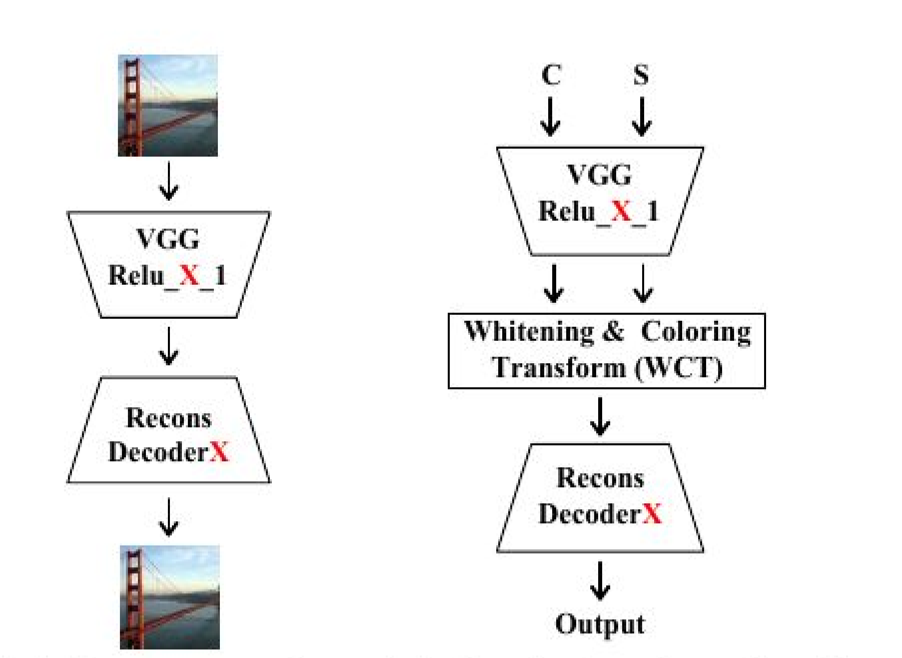
Goal

Content

Style

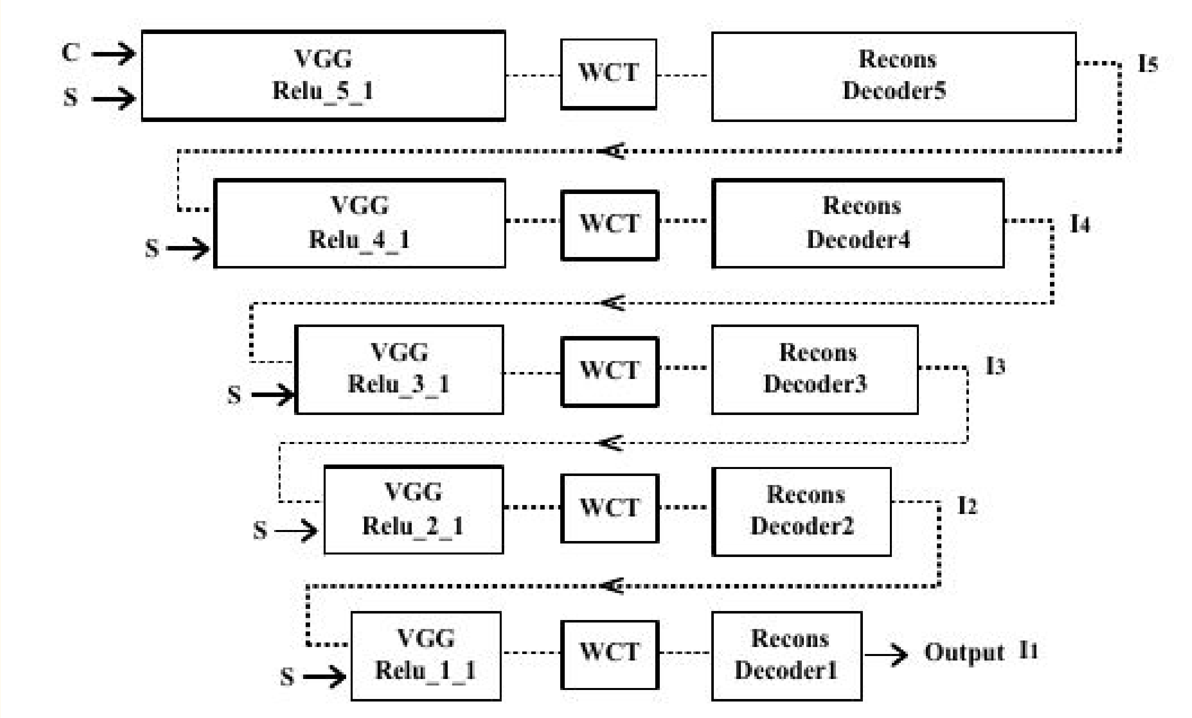
Result

Architecture



(a) Reconstruction (b) Single-level stylization

Architecture(contd.)



(c) Multi-level stylization

Algorithm

- Reconstruction Decoder
- Whitening Transform
- Coloring Transform
- Multi-level coarse-to-fine stylization

Reconstruction Decoder

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

where I_{input}, I_{output} are the input image and reconstruction output, and Φ is the VGG encoder that extracts the Relu_X_1 features.

- Minimisation of L yields auto-encoder for general image reconstruction.
- 5 different decoders trained for different VGG-19 layers.
- After training, decoder is fixed(will not be fine-tuned).

Whitening Transform

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

where D_c is a 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 $f_c\, f_c^{\top} = E_c D_c E_c^{\top}$.

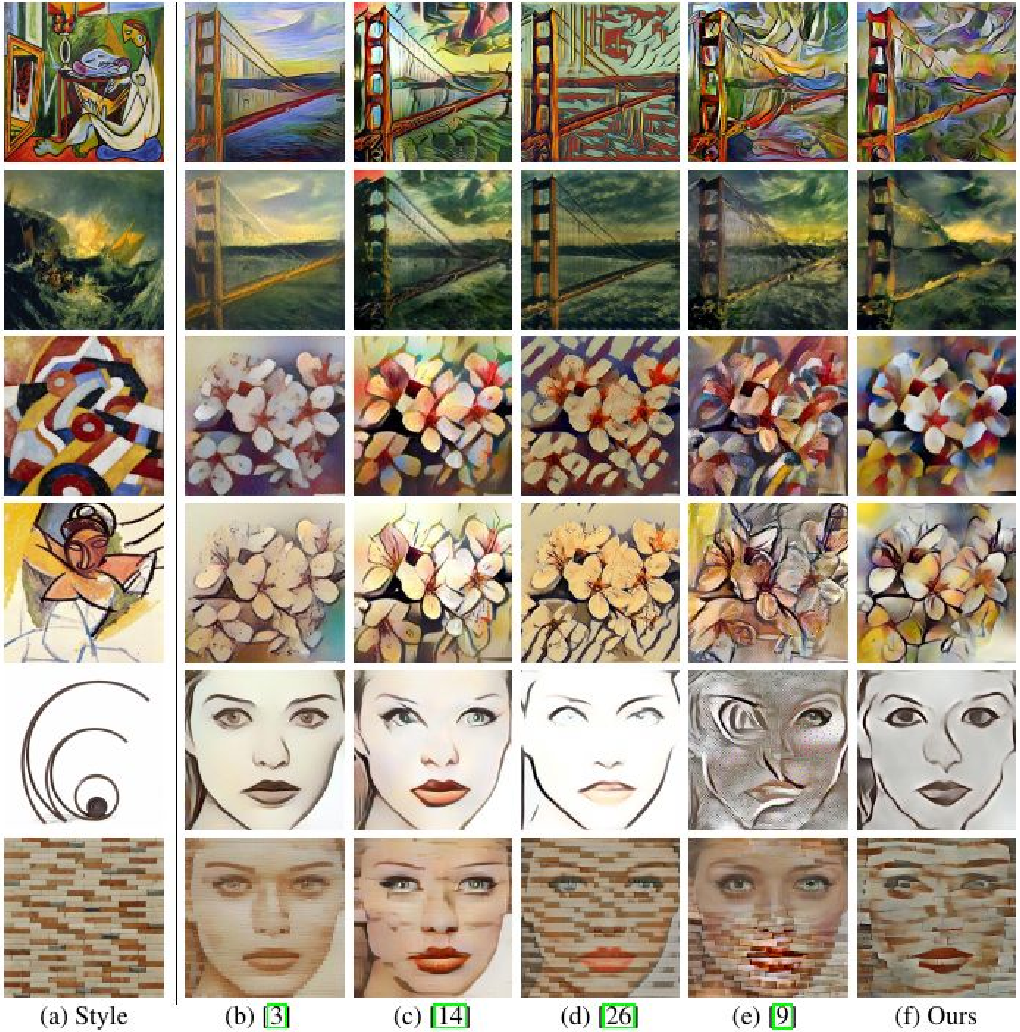
Coloring Transform

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

where D_s is a diagonal matrix with the eigenvalues of the covariance matrix $f_s \, f_s^{\top} \in \Re^{C \times C}$, and E_s is the corresponding orthogonal matrix of eigenvectors.

Multi-level coarse-to-fine stylization

- Higher layer features capture more complicated local structures
- Lower layer features carry more low-level information
- Apply WCT at higher layer to obtain coarse stylized image and consider it as new content image to further adjust features in lower



(a) Style
(b) [3]
(c) [14]
(d) [26]
(e) [9]
(f) Ours

Experimental Results

	Chen et al. [3]	Huang et al. [14]	TNet [26]	DeepArt [9]	Ours
Arbitrary	√	√	×	√	√
Efficiency	√	√	√	×	√
Learning-free	×	×	×	√	√

	Chen et al. [3]	Huang et al. [14]	TNet [26]	Gatys et al. [9]	Ours
$\log(L_s)$	7.4	7.0	6.8	6.7	6.3
Time/sec	2.1	0.20	0.18	21.2	1.5

Our Work

Pipeline of Project

- We did it using Keras from scratch.
- Modules
  - Reconstruction (Training)
  - Separate Encoder and Decoder
  - Make feature from encoder for style image and content image
  - WCT (Whitening and Coloring Transform)
  - Decode the feature to get stylized image

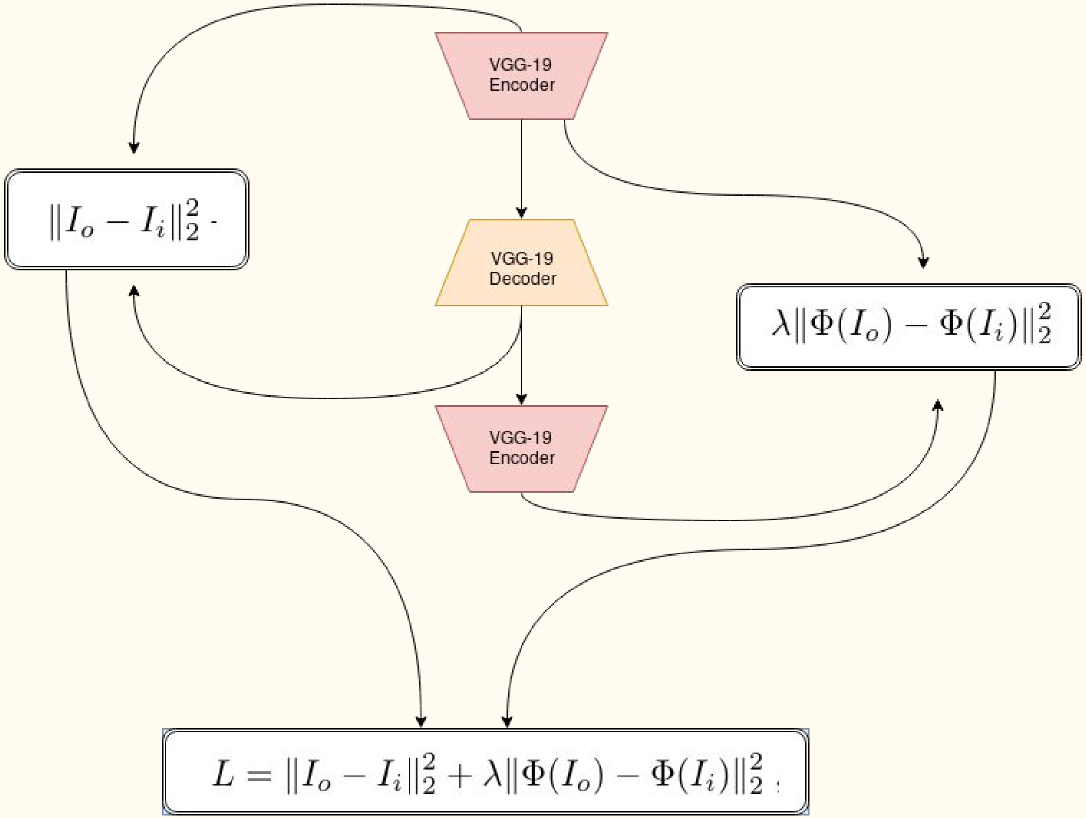
Challenges faced

- Loss function depends on feature of input image, input image, feature of output image and output image.

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

- Sequential model cannot be used. Requires functional model.

Reconstruction Architecture



Challenges Faced

- Training heavy
- Tried training VGG 19 decoder (around 100M parameters) on MS Coco
  - 20 small images takes 1 minute for 1 epoch
  - MS Coco contains 330K images
  - Computationally not feasible
- Trained small architecture on MNIST and CIFAR 10 dataset

Challenges faced

- Trained encoder and decoder together for reconstruction module
- Need to separate encoder and decoder after training for WCT
- Models were used as layers

Experiments

- Decoder Training
  - VGG-19 Relu layers
  - Microsoft COCO dataset
  - MNIST dataset
  - CIFAR-10 dataset
- Style Transfer
  - Describable Texture Dataset (DTD)
  - User Control

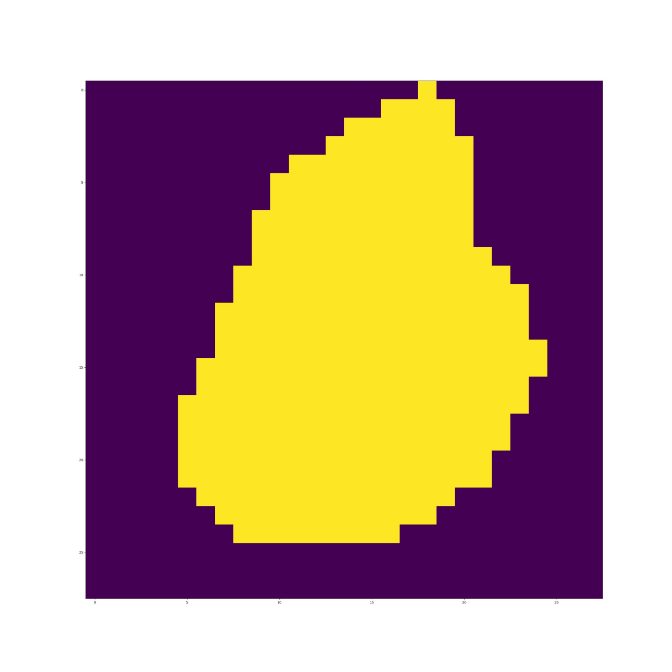
RESULTS

MNIST RESULTS

Content : 4
Style : 1

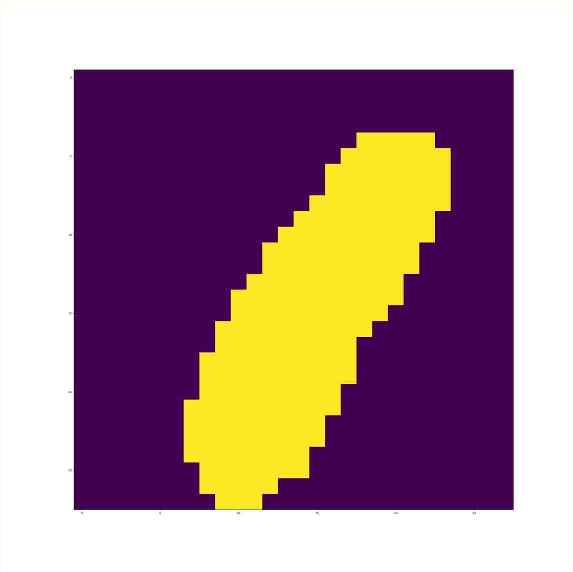
Content : 6
Style : 3

Content : 2
Style : 0



Content : 6
Style : 0

Content : 1
Style : 1



Content : 1
Style : 0

RESULTS FROM PRE-TRAINED MODEL

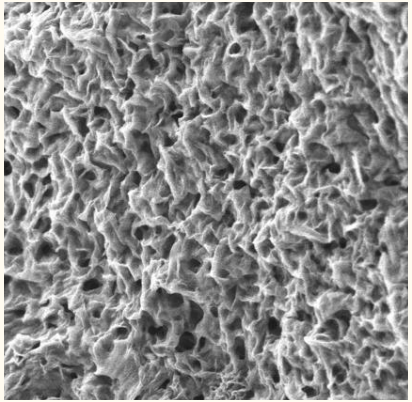
Underexposed image

Content

Style

Final Image

Overexposed image

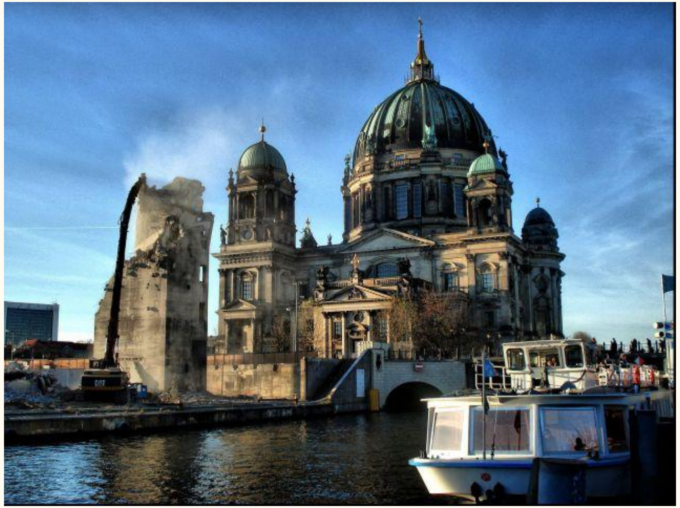


Content

Style

Final Image

Feature matching



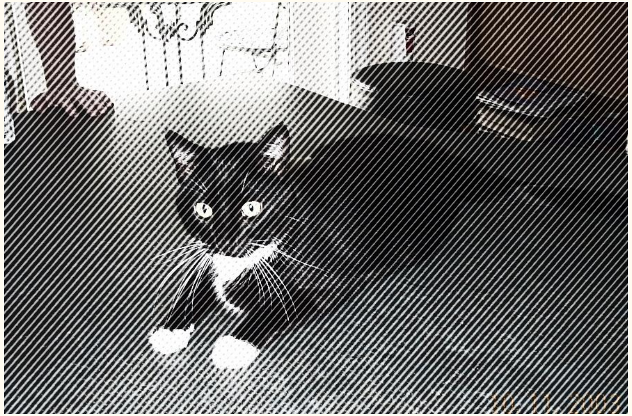
Content

Style

Final Image

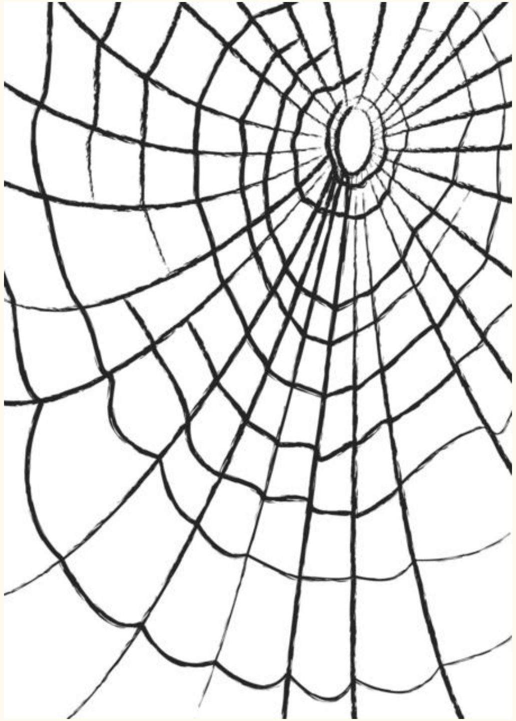
Ghosting

Flicker



Content

Style



Final Image

Content

Style

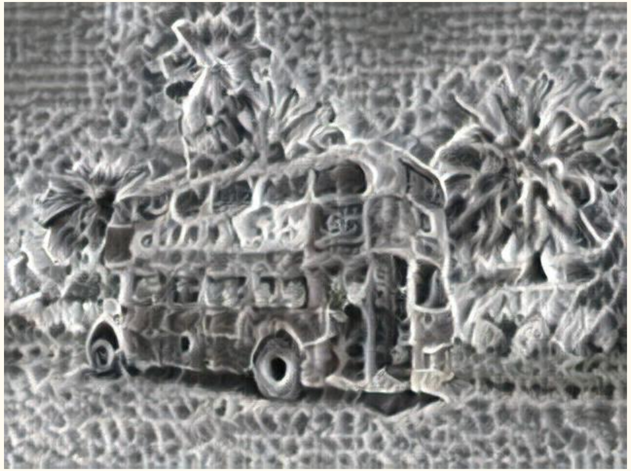
Final Image

Our photooooo……

User Control

Alpha = 0.2

Alpha = 0.5

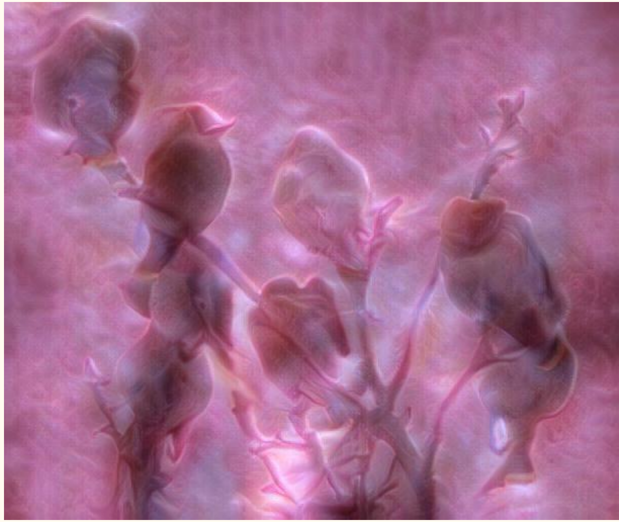


Alpha = 0.8

User Control

Alpha = 0.2

Alpha = 0.5



Alpha = 0.8

Colour dependence on style

Applications

- Style Transfer
- Texture synthesis
- Many other real life applications
  - Neural style transfer to design clothes
  - Displaying images as if it was drawn by an artist.

Tools

- Python
- Keras
- Deep learning library (Tensorflow, Pytorch...)

Few ideas worth trying

- Style transfer in videos
- Interpolation between styles
- Spatial control
- Comparison with other architecture (ResNet, GoogleNet, ...)

References

- Li Y, Fang C, Yang J, et al. Universal Style Transfer via Feature Transforms[J].
- Fashioning with Networks: Neural Style Transfer to Design Clothes. https://arxiv.org/pdf/1707.09899.pdf

THANK YOU!