Arbitrary Style Transfer

•••

Team Vizzun :

Abinash Maharana
Aditya Gupta
Mohsin M. Hafiz
Kartik Agarwal

GitHub Link: https://github.com/Computer-Vision-IIITH-2021/project-team-vizzun

Contents

Overview

Style transfer is an optimization technique used to take two images—a content image and a style reference image (such as an artwork by a famous painter)—and blend them together so the output image looks like the content image, but “painted” in the style of the style reference image.

&

Problems Faced

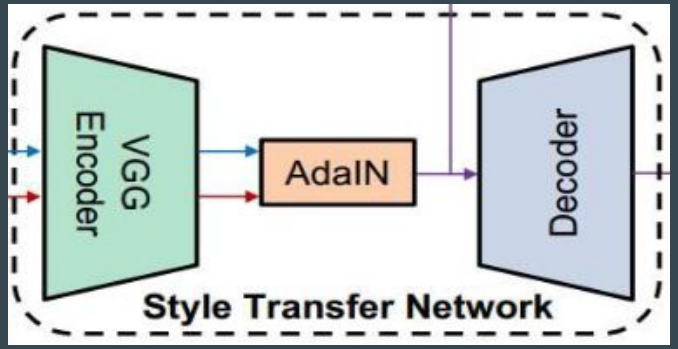
1. We faced a lot of difficulty in implementing a deep learning project because all of us were new to any deep learning library. We had to go through several tutorials to learn and then we could start the implementation. We faced issues like, not getting the output images in proper colors,etc.

2. The second problem we faced was the lack of resources. We had limited resources such as google colab having usage and time limits and thus we had to switch several accounts to get good results.

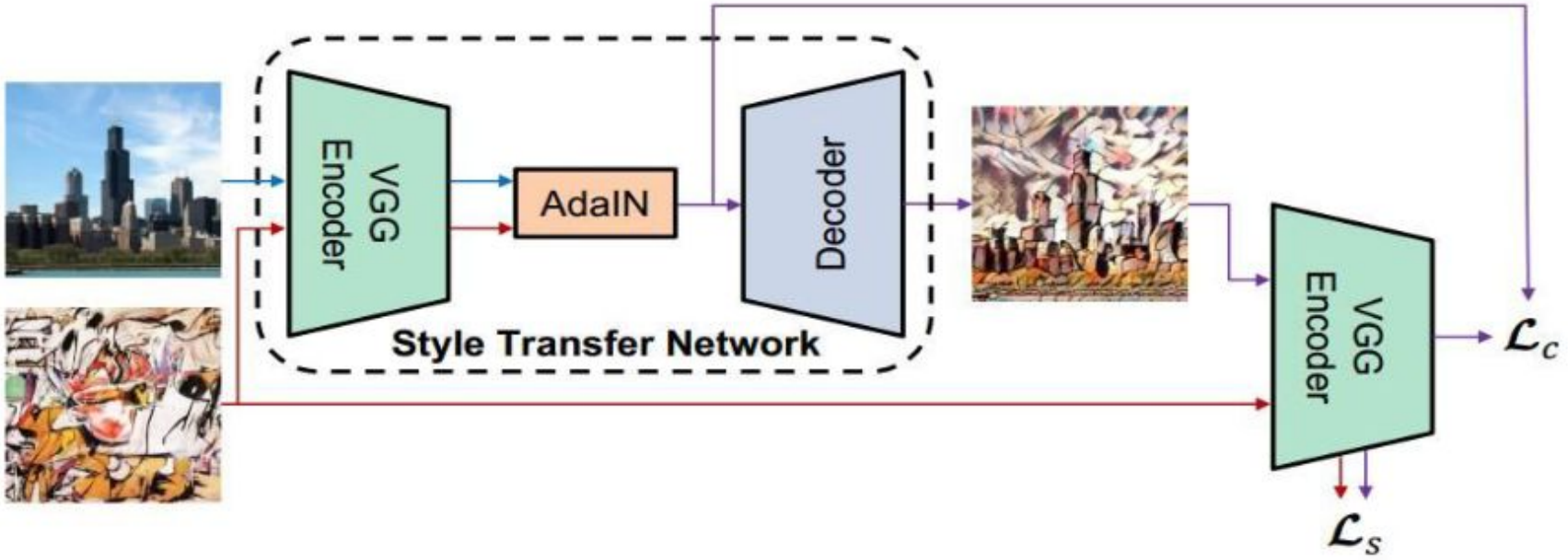
Components

Main components of our architecture:

1. Encoder
2. Adaptive Instance Normalization train layer
3. Decoder and other utilities.

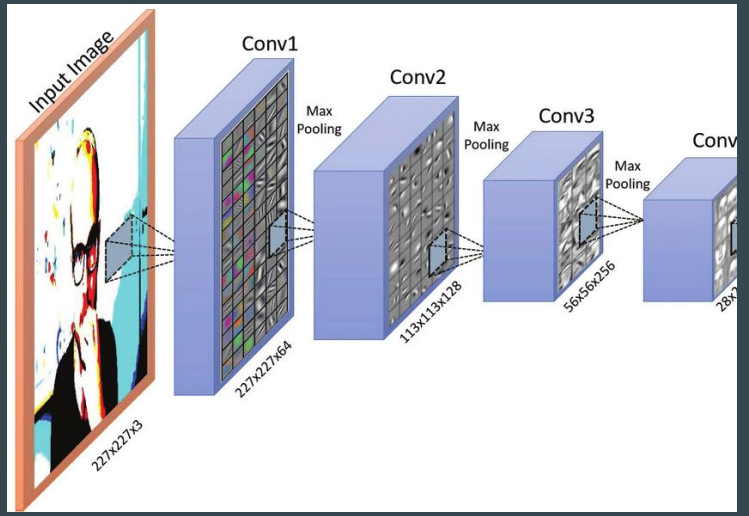


Our architecture is then trained on the MS-COCO dataset. After completion, our model is run to get output from our content and style images.


VGG Encoder
AdaIN
Decoder
Style Transfer Network
VGG Encoder
$\mathcal{L}_c$
$\mathcal{L}_s$

Encoder:

- Pre-trained VGG19 encoder.
- AdaIN layer works on the output from the encoder
- Training error calculated on encoded images



AdaIN:

- AdaIN, or Adaptive Instance Normalization is a simple extension to Instance Normalization, which itself was an improvement over Batch Normalization.
- Previous methods learned affine parameters from data in different ways.
- AdaIN aligns the channel-wise mean and variance of the input image to match that of the style image.
- “Unlike BN, CN or IN, AdaIN has no learnable affine parameters.” - this helps it achieve real-time style transfer very fast.

$$\mathrm{BN}(x) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta$$

$$\mathrm{IN}(x) = \gamma \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta$$

$$\mathrm{CIN}(x; s) = \gamma^s \left(\frac{x - \mu(x)}{\sigma(x)} \right) + \beta^s$$

AdaIN:

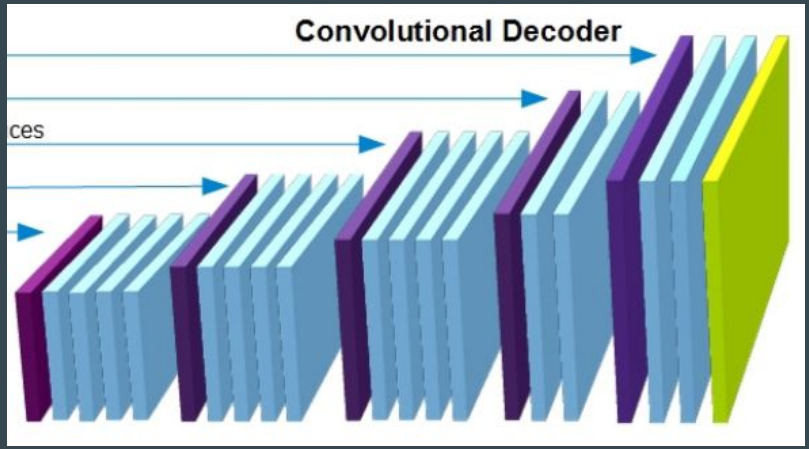
- Here we the change the feature statistics of the content image to that of the target image.
- We the multiply the variance of style image to our normalised content image and then add the mean of style image.
- This output is then send forward to decoder.

$$\text{AdaIN}(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y)$$

Decoder:

- Architecture used was reverse of encoder by replacing the Max Pooling with upsampling.

- In training, output is send back to encoder for error calculation and training of the network.



Results:

Content Image

Style Image

Output

Content Image

Style Image

Output

Content Image

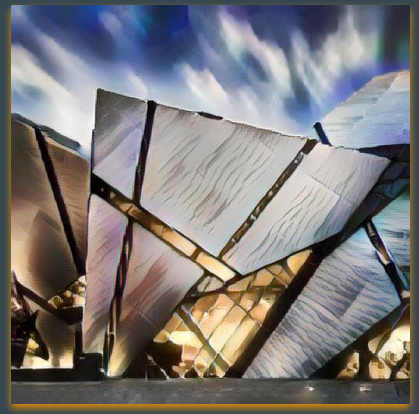
Style Image

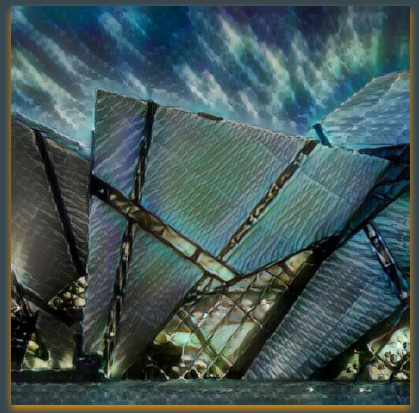
Output

Content Image

Style Image

Output





Content Image

Style Image

Output

Experiments

1) Content-Style trade-off

We can vary the degree of style transfer by varying Alpha during runtime.
As shown in the following slides, a smooth transition between content-similarity and style-similarity can be observed by changing α from 0 to 1.

2) Style Interpolation

A combination of styles can be given to the model with some weights (which sum up to 1) and a new style which will be an interpolation of the given styles will be applied on the content image

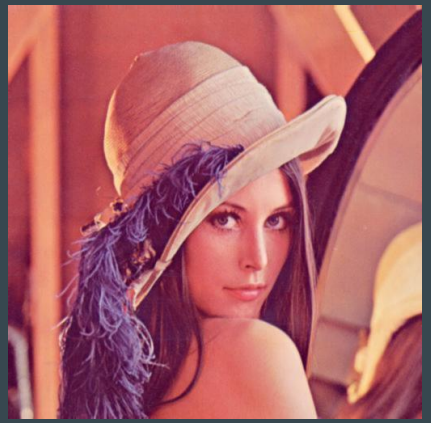
1) Varying Alpha results:

Content:

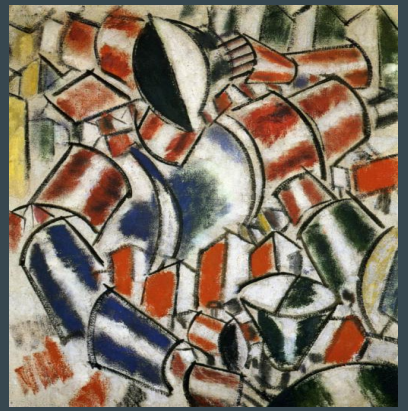
Style:

Alpha: 0.25 0.5 0.8 1

Content:

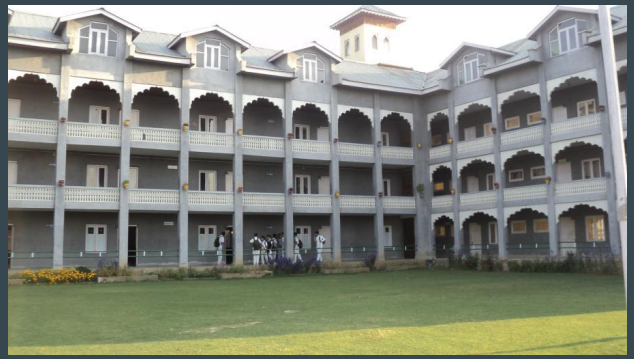


Style:

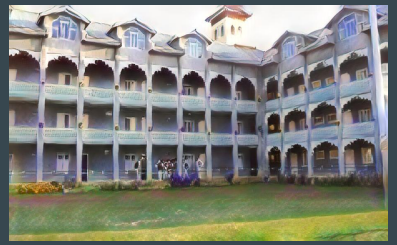


Alpha: 0.25 0.5 0.8 1

Content:



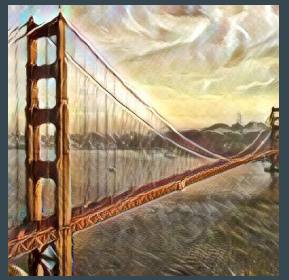
Style:

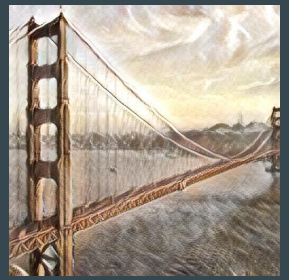


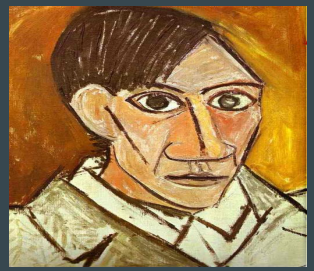
Alpha: 0.25 0.5 0.8 1

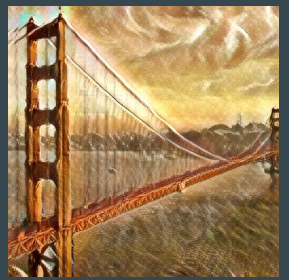
2) Style Interpolation results:

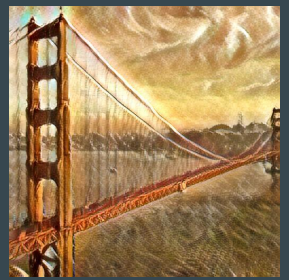
3 styles



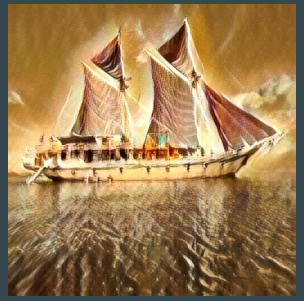
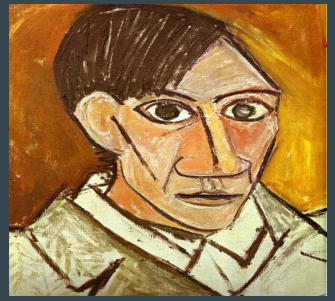
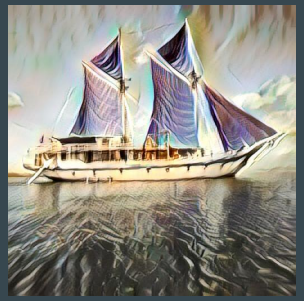
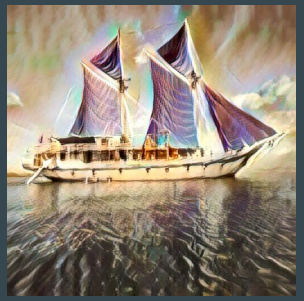
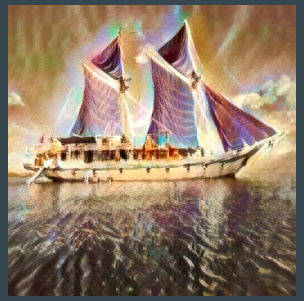




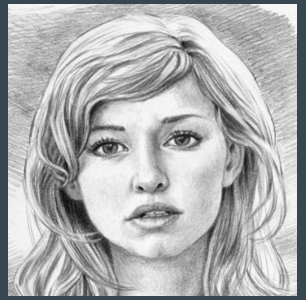


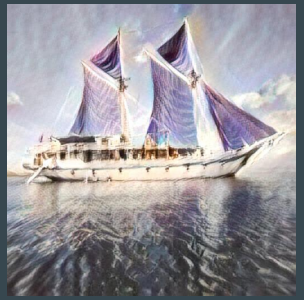




4 styles





References

- Huang, X., & Belongie, S.J. (2017). Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. 2017 IEEE International Conference on Computer Vision (ICCV), 1510-1519.

  https://arxiv.org/pdf/1703.06868.pdf

- Pytorch documentation.

  https://pytorch.org/docs/stable/index.html

- VGG19 pretrained model:

  https://iq.opengenus.org/vgg19-architecture/

Thank you.