Neural Style Transfer with TensorFlow

Anagh P
Roll No: 08
Reg.No: KTE22MCA-2008

Guided By
Prof. Athira Soman
Department of Computer Applications
Rajiv Gandhi Institute Of Technology, Kottayam

Contents

Introduction

- Neural Style Transfer, is a technique that merges the content of one image with the artistic style of another.
- ResNet50 and VGG16 are two popular CNN architectures widely used in computer vision tasks.
- Project aims to compare the performance of ResNet50 and VGG16 in Neural Style Transfer

Current state of art

- VGG16 and VGG19 are widely used in computer vision tasks.
- VGG16's 16-layer architecture is simpler than VGG19's 19 layers but offers comparable performance.

Motivation

- Neural Style Transfer (NST) mixes artistic styles with images seamlessly.
- Provides an avenue for artistic expression and experimentation.
- Understanding how different CNN architectures impact in efficiency and quality in style transfer.
- Comparing architectures helps find the most efficient model for real-time style transfer

Objectives

- Develop a VGG16 model for style transfer.
- Develop a ResNet50 model for style transfer.
- Implement a loss function that computes the total loss.
- Compare the images generated by the both models using the calculated total loss.

Literature survey

Table 1: Literature survey

Sl No.	Title	Author	Description	Outcome
1	A Neural Algorithm of Artistic Style	L. A. Gatys, A. S. Ecker, and M. Bethge, (2016)	• Introduced Neural Style Transfer • allowing separation of content and style	• High perceptual quality artistic images.
2	Generating Artistic Styles using Neural Style Transfer	Esha G, Nitisha P, Rahul Pal, (2021)	• Comparison of VGG16 and VGG19 in NST	• Both Models showed appealing results • VGG16 has lower loss values. .
3	Comparison of VGG16, VGG19 and ResNet50 architecture	S. Mascarenhas and M. Agarwal, (2021)	• Compares VGG16,VGG19,ResNet50	• ResNet50 is more accurate
4	Texture Synthesis using CNN	L. A. Gatys, A. S. Ecker, and M. Bethge, (2015)	• Introduces a system based on DNN • Creating artistic images	• Creates high-quality natural textures

Proposed Methodology

The proposed method includes several key stages:

- Image preprocessing
- Implement neural style transfer using ResNet50 & VGG16
- Feature extraction
  - Selection of these layers is just by trial and error

Table 2: Feature Extraction Layers

Layer Type	VGG16	ResNet50
Content Layer	block5_conv2	conv3_block4_out
Style Layer	block1_conv1 block2_conv1 block3_conv1 block4_conv1 block5_conv1	conv1_relu conv2_block1_out conv3_block1_1_conv conv4_block1_1_conv conv5_block1_1_relu
Total Loss	2782320.0	656.97

Proposed Methodology

- Loss function
  - Content loss:
    - It measures the difference between the content of the final image and the content image.
    - $L_{\text{cont}}(l,(p,x)) = \sum_i (F_l^{ij}(x) - P_l^{ij})^2$
  - Style loss:
    - It measures the difference between the style of the final image and the style image.
    - $\mathrm{E}_l = \frac{1}{4N_l^2 M_l^2} \sum_i (G_{ij}^l - A_{ij}^l)^2$
- Optimization
  - Gradient descent:
    - $L_{\text{total}}(\dot{p},\dot{a},\dot{x}) = \alpha L_{\text{content}}(\dot{p},\dot{x}) + \beta L_{\text{style}}(\dot{a},\dot{x})$
    - where α and β are the content and style reconstruction weighting factors.
- Fine tuning
  - The ideal learning rate and layers will be determined through systematic experimentation and validation.
- Optimization
- Comparison and analysis of both models

Architecture

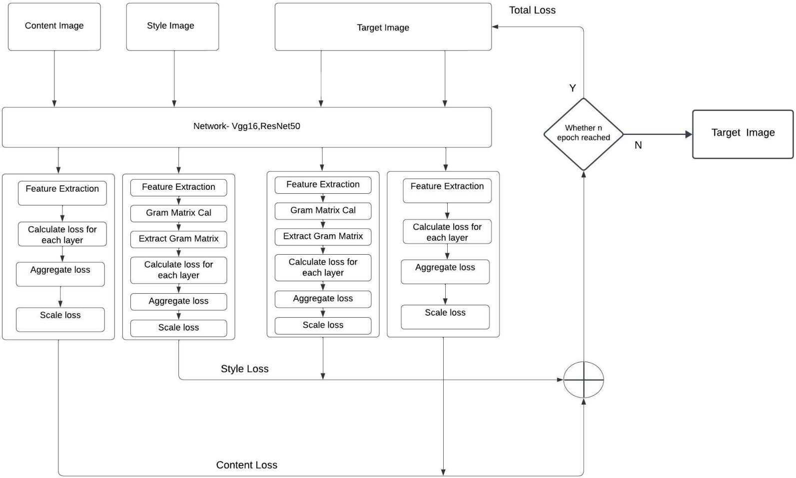

Figure 1: Proposed System Architecture

VGG16 Model Architecture

- **Layer Structure:** VGG16 consists of 16 layers, including 13 convolutional layers followed by 3 fully connected layers.
- **Filter Size and Pooling:** VGG16 uses 3x3 filters with max-pooling layers of 2x2 filters for downsampling.
- **Depth and Complexity:** VGG16's 13 convolutional layers enable it to learn intricate features at various levels of abstraction.
- **Parameter Count:** Despite its simplicity, VGG16 has around 138 million trainable parameters, facilitating the capture of intricate patterns in data.

ResNet50 Model Architecture

- **Layer Structure:** ResNet50 comprises 50 convolutional layers with residual blocks for feature extraction, followed by global average pooling and a fully connected layer for classification.
- **Filter Size and Pooling:** ResNet50 employs 3x3 convolutional filters with skip connections for pooling.
- **Depth and Complexity:** ResNet50's 50-layer architecture captures complex features with residual connections.
- **Parameter Count:** ResNet50 maintains efficiency with around 25 million parameters.

Materials and Methods - Tools

Platform and Environmental Setup

- The platform and enviornmental setup include windows 10 and Google Colab.

Software Tools

- T4 GPU
- Python
- TensorFlow

Hardware Tools

- RAM : 8GB
- CPU: AMD Ryzen 5 3550H CPU @ 2.10GHz

Result

- VGG16 perform better than ResNet50 on certain image combinations, while ResNet50 may excel on others.
- ResNet50's performance drops notably, while VGG16 consistently delivers decent results.
- VGG16 consistently demonstrated faster processing times compared to ResNet50 for neural style transfer.
- ResNet50 consistently achieved lower total loss values compared to VGG16.
- Identified that a learning rate of 15 for VGG16 and 20 for ResNet50 minimized total loss and produced b stylized images.

Result - VGG16

Figure 2: Input content image

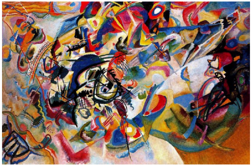

Figure 3: Input style image

Figure 4: Output stylized image

Result - ResNet50

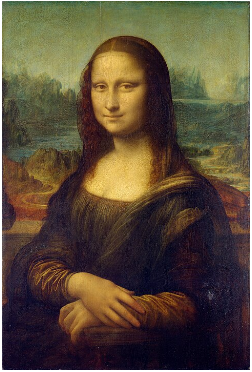

Figure 5: Input content image

Figure 6: Input style image

Figure 7: Output stylized image

Performance Analysis

VGG16 outperforms ResNet50

VGG16 Result

Figure 8: Input content image

Figure 9: Input style image

Figure 10: Output stylized image

Performance Analysis

ResNEt50 Result

Figure 11: Input content image

Figure 12: Input style image

Figure 13: Output stylized image

Performance Analysis

ResNet50 outperforms VGG16

ResNet50 Result

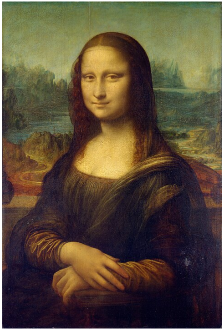

Figure 14: Input content image

Figure 15: Input style image

Figure 16: Output stylized image

Performance Analysis

VGG16 Result

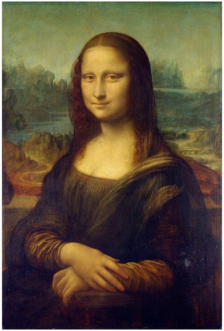

Figure 17: Input content image

Figure 18: Input style image

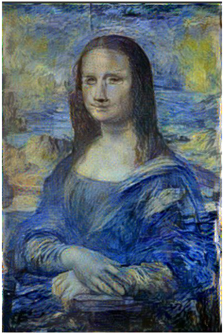

Figure 19: Output stylized image

Performance Analysis

Table 3: Comparison of Total Loss Values for Different Iterations

Iteration	VGG16 Loss	ResNet50 Loss
100	577010	50514.0
200	335490	10351.0
300	230670	4888.9
400	182800	2698.0
500	159210	1244.1
600	146810	665.73
700	139500	412.85
800	135910	294.32
900	131920	239.56
1000	130560	207.35

Performance Analysis

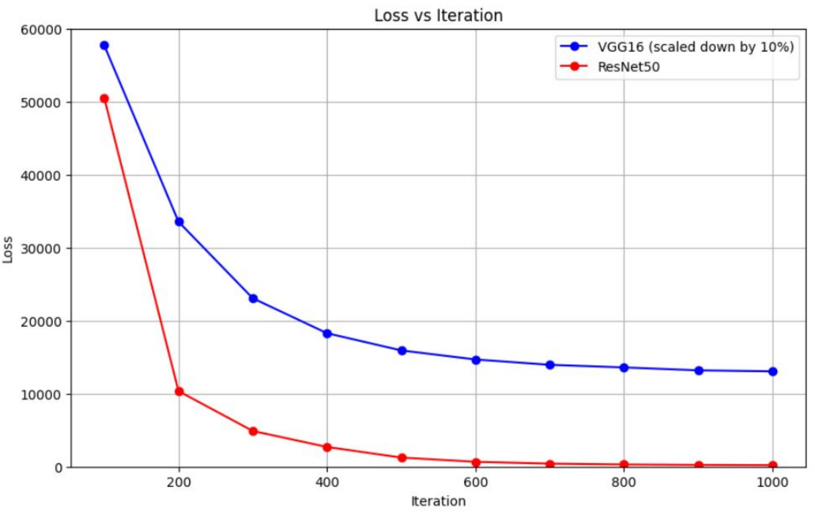

Figure 20: VGG16 vs ResNet50 Total loss function

Performance Analysis

Table 4: Comparison of Total time needed to complete each iteration in seconds

Iteration	VGG16	ResNet50
100	12.30	17.15
200	24.43	35.07
300	36.76	51.91
400	48.96	69.75
500	61.03	86.99
600	73.07	105.55
700	85.11	123.80
800	97.03	142.55
900	108.26	160.55
1000	120.52	179.43

Performance Analysis

Table 5: Total loss corresponding to different learning rates

Learning Rate	VGG16	ResNet50
1	186804.1	4204.2
10	131383.5	354.33
15	130880.1	254.19
20	134781.4	207.35
30	146925.6	150.57

Performance Analysis

Table 6: Comparison of VGG16's Feature extaction layers and their total loss

Layer Type	**1**	**2**	**3**
Content Layer	block5_conv2	block4_conv2	block3_conv2
Style Layer 1	block1_conv1	block1_conv1	block1_conv1
Style Layer 2	block2_conv1	block2_conv1	block2_conv1
Style Layer 3	block3_conv1	block3_conv1	block3_conv1
Style Layer 4	block4_conv1	block4_conv1	block4_conv1
Style Layer 5	block5_conv1	block5_conv1	block5_conv1
Total Loss	2782320.0	80605464.0	46028130.0

Performance analysis

Table 7: Comparison of ResNet50's feature extraction layers and their total loss

Layer Type	1	2	3
Content Layer	conv3_block4_out	conv3_block4_out	conv3_block4_out
Style Layer 1	conv1_relu	conv1_relu	conv1_relu
Style Layer 2	conv2_block1_out	conv2_block3_outt	conv2_block1_1_relu
Style Layer 3	conv3_block1_1_conv	conv3_block4_out	conv3_block1_1_relu
Style Layer 4	conv4_block1_1_conv	conv4_block6_out	conv4_block1_1_relu
Style Layer 5	conv5_block1_1_relu	conv5_block3_out	conv5_block1_1_relu
Total Loss	656.97.0	4832.46	508.0

Conclusion and Future Scope

- VGG16 generally performs better than ResNet50 in image generation, with ResNet50 occasionally producing poor-quality images.
- For future work, exploring advanced fine-tuning techniques and architectural modifications could optimize ResNet50's performance in neural style transfer.

Implementation Status and Plan

Table 8: Implementation Status and Plan

Task	**Status**
Dataset collection	Completed
Preprocessing	Completed
Implementation of VGG16 model	Completed
Implementation of ResNet50 model	Completed
Fine Tuning	Completed
Evaluation of models	Completed
Comparison of Vgg16 and ResNet50 models	Completed

Reference

[1] Leon Gatys, Alexander S Ecker, and Matthias Bethge. "Texture synthesis using convolutional neural networks". In: *Advances in neural information processing systems* 28 (2015).

[2] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2414–2423.

[3] Esha Ghorpade, Nitisha Pradhan, and Rahul Pal. "Generating Artistic Styles using Neural Style Transfer". In: *International Journal of Engineering Research & Technology (IJERT)* 10.06 (June 2021).

[4] Varun Gupta, Rajat Sadana, and Swastikaa Moudgil. "Image style transfer using convolutional neural networks based on transfer learning". In: *International journal of computational systems engineering* 5.1 (2019), pp. 53–60.

[5] Sheldon Mascarenhas and Mukul Agarwal. "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification". In: Nov. 2021, pp. 96–99. DOI: 10.1109/CENTCON52345.2021.9687944.

Git History

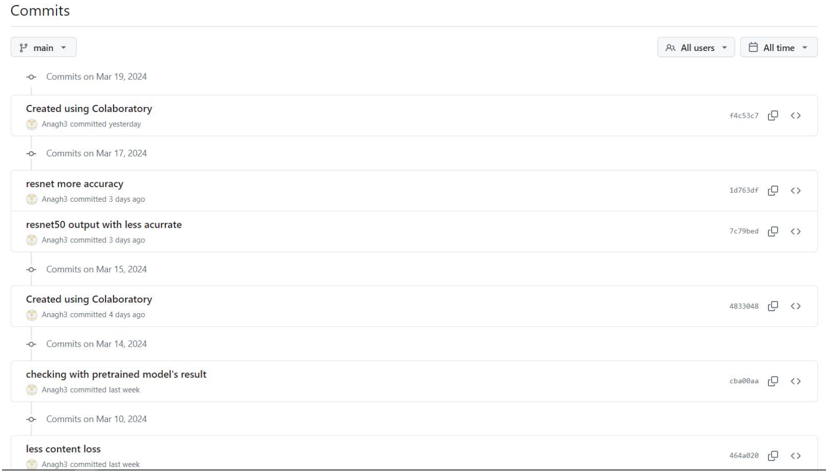

Figure 21: Git History

Thank you!