Style Transfer in Computer Vision

Miriam Xu

***Abstract*— This paper evaluates and compares the performance of three different approaches to mitigate visual artifacts in style transfer: multi-scale, patch-based, and regularization. Additionally, I offer an overview of the style transfer field, reviewing current methods and analyzing their limitations.**

***Index Terms*— content; style;**

I. Introduction

Style transfer involves modifying the visual style of an image while retaining its content. This process makes images adopt the artistic qualities of another image, allowing for creative and practical applications such as generating art, creating filters, and enhancing visual effects in media and entertainment.

Formally, style transfer algorithms aim to isolate and apply the ”style” features of one image to the ”content” structure of another, achieving an output that combines the two.

A. Concepts

The basic concepts in style transfer rely on the distinction between content and style. The content contains an image's primary structure or layout, usually composed of objects, shapes, and spatial relationships that define the recognizable parts of the scene. Style contains the texture, colors, patterns, and other visual characteristics that contribute to the image's overall aesthetic.

For example, take these two images and their style transfer output:

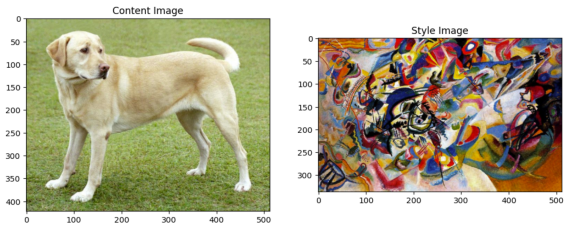

Fig. 1. Example of a content image and style image.

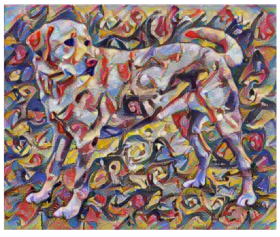

Fig. 2. Example output of style transfer.

The goal of style transfer is to generate an output image I_g that balances the content of a content image I_c with the style of a style image I_s.

This balance can be expressed as an optimization problem:

$$I_g = argminL(I, I_c, I_s)$$

where L is a loss function that measures deviations from both the content and style. The workflow of style transfer can be summarized as:

1) Decompose the input images into content and style components
2) Transform the style features of I_s to match the content structure of I_c
3) Synthesize the new image I_g.

While the specific definition of L varies by method, it typically includes a content term that penalizes deviations from the content of I_c, and a style term that penalizes deviations from the style of I_s. Some methods also incorporate regularization terms to ensure smoothness or naturalness in the generated image.

Style transfer is achieved by iteratively adjusting the pixels of a generated image until the content and style losses are minimized. This optimization process ensures that the generated image converges to one that simultaneously resembles the content image in structure and the style image in visual texture. Continuing the example from the previous figures, here is I_g formed with a shorter optimization:

Fig. 3. I_g with shorter optimization.

As you may observe, the content is overpowering the style. The functions used for loss minimization typically include content loss, style loss, perceptual loss, and regularization.

Content loss measures how well the output image I_g retains the structural or semantic feature of the content image I_c. This is often quantified using pixel-level similarity, such as mean squared error, or feature-based similarity.

Style loss measures the stylistic similarity between the output image I_g and the style image I_s. Depending on

the method, this may involve comparing textures, color distributions, or statistical properties such as Gram matrices.

Perceptual loss uses human-perceptual metrics from pretrained neural networks to ensure the output image is coherent.

Regularization in the form of smoothness constraints such as total variation loss can be added to prevent artifacts in I_g.

B. Notation and Syntax

Throughout this paper, we denote:
I_c : Content image.
I_s : Style image.
I_g : Generated image.

II. Existing Methods

Style transfer in computer vision has evolved significantly, mainly leveraging deep learning to blend content and style from images. Early methods were built on neural network-based approaches, such as Gatys et al.'s Neural Style Transfer (NST) [1].

Recent advancements have moved towards real-time style transfer using feedforward networks. Models like Johnson et al.'s Perceptual Loss Networks precompute style information during training, allowing for faster inference. Other works, like Huang and Belongie's Adaptive Instance Normalization (AdaIN), introduced mechanisms for more flexible and diverse style control by aligning content features to style statistics [2].

State-of-the-art methods include GAN-based approaches (e.g., StarGAN and CycleGAN) and diffusion models. GANs focus on generating high-quality and diverse styles while maintaining content fidelity [3].

A. Neural Style Transfer

Neural style transfer uses convolutional neural networks (CNNs) to optimize image representations for combining content and style. These methods calculated content and style losses based on feature activations in a pretrained CNN, such as VGG, and iteratively adjusted pixel values to create stylized outputs. Gram matrices are used to quantify the correlations between feature maps within a CNN layer [1].

Given a layer with N feature maps of size M, the Gram matrix G of a layer l is defined as:

$$G_i^l j = \sum_k F_i^l k F_j^l k$$

where F represents the feature map activations of layer l, and $G_i^l j$ encodes the degree of co-activation between feature maps i and j, capturing textural details indicative of style.

Content loss, $L_{content}$, denotes the difference between the content of the generated image and the content of the original image, typically as the mean squared error between feature maps:

$$L_{content} = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{H} \sum_{k=1}^{W} (F_{ijk}^l(I_g) - F_{ijk}^l(I_c))^2$$

where F_{ijk}^l represents the activation of the i-th feature map at spatial location (j, k).

Style Loss, L_{style}, quantifies the difference in style between the generated image and the style image S by comparing their Gram matrices G^l:

$$L_{style}(S, G) = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{i,j}^l(S) - G_{i,j}^l(G))^2$$

Where N_l and N_l denote the number of filters and spatial size at layer l.

The total loss function L combines these:

$$L = \alpha L_{content} + \beta L_{style}$$

where α and β are weighting factors that control the emphasis on content and style, respectively.

B. Feedforward Networks

Feedforward approaches like Johnson et al.'s method minimize the perceptual loss during training. The perceptual loss includes feature reconstruction loss and style reconstruction loss. Feature reconstruction loss is defined as $L_{feature}$:

$$L_{feature} = \sum_l ||\phi_l(G) - \phi_l(C)||_2^2$$

where ϕ_l are feature maps from a pretrained network. Style reconstruction loss is defines as L_{style}:

$$L_{style} = \sum_l ||G_{\phi_l}(G) - G_{\phi_l}(S)||_F^2$$

where G_{ϕ_l} is the Gram matrix at layer l [2].

C. Adaptive Instance Normalization (AdaIN)

AdaIN dynamically aligns the mean and variance of content features to those of the style features:

$$AdaIN(F_c, F_s) = \sigma(F_s)(\frac{F_c - \mu(F_c)}{\sigma(F_c)} + \mu(F_s)$$

where μ and σ are respectively the mean and standard deviation [2].

D. Generative Adversarial Networks (GANs)

GANs like CycleGAN use adversarial losses to map between content and style domains [3]. Adversarial Loss is defined as L_GAN:

$$L_GAN = \mathbb{E}[logD(Y)] + \mathbb{E}[log(i - D(G(X)))]$$

where D is the discriminator, and G is the generator. Cycle Consistency Loss is defined as L_{cycle}:

$$L_{cycle} = ||G(F(X)) - X||_1$$

III. Open Challenges

A. Current Limitations

Many open challenges are still present in style transfer; finding the right balance between I_s and I_c and creating a model that is both fast and high resolution, achieving high-quality results in real-time while preserving the content structure proves to be difficult. The issue I'd like to address is spatial consistency and artifacts.

Style transfer can introduce unnatural textures, distortions, or other artifacts, particularly when transferring highly detailed or intricate styles. Loss functions that penalize spatial inconsistency, along with patch-based methods that apply style in small, image-appropriate patches, have been explored. Temporal consistency techniques have also been introduced to avoid flickering in video style transfer. However, some artifacts continue to persist, especially around edges or high-contrast areas. Additionally, these methods are computationally intensive and do not always generalize well, especially when there are overlapping or fine details [1].

IV. Proposing Method

This research proposes to evaluate and compare three prominent techniques for reducing artifacts in style transfer: multi-scale style transfer, patch-based approach, and regularization in training.

A. Multi-scale Style Transfer

Multi-scale style transfer works by applying style transfer at multiple scales, progressively refining the output from coarse to fine detail. By stylizing the image at different resolutions and blending the results, the idea is that we can achieve a more cohesive look [1]. The general steps include:

1) Iteratively decrease the scale of the content image I_c to a set of five progressively smaller resolutions.
2) Starting from the lowest resolution, apply the style transfer at each scale.
3) Blend the output incrementally, using the result of the previous scale as a guide for the next.
4) Increase the scale of the final output to the original resolution.

B. Patch-based Approach

The patch-based approach treats the style transfer problem as a local matching task. Instead of applying the style globally across the entire image, they decompose both the content and style images into small patches. The style transfer process matches and applies style patterns patch by patch [4]. The general steps include:

1) Divide both I_c and I_s into small overlapping patches
2) Compare each patch in I_c to patches in I_s, finding the best match by comparing similarities between corresponding pixels.
3) Replace the content patches with the corresponding style patches.
4) Use a blending function to combine overlapping patches smoothly.

C. Regularization

Regularization techniques involve adding constraints to the loss function during the training of the style transfer model. These constraints help smooth the output image, reducing noise and preventing abrupt transitions that cause artifacts [5]. I will be applying total variation loss, which penalizes abrupt intensity changes between adjacent pixels:

$$L_{TV} = \sum_{i,j}((I_{i,j} - I^2_{i+1,j} + (I_{i,j} - I_{i,j+1})^2)$$

where I is the generated image, i is the row index of the pixel, and j is the column index of the pixel.

D. Comparison

Here is a comparison review of the three approaches gained from initial research.

Aspect	Multi-Scale	Patch-Based	Regularization
Artifact Reduction	Global and local patterns	Textures	Noise and Transitions
Content Preservation	High	Moderate	High
Style Consistency	High	High	Moderate
Computational Cost	High	High	Low to Moderate
Ease of Implementation	Moderate	Moderate to Difficult	Easy to Moderate

TABLE I
Comparison of Techniques for Artifact Reduction in Style Transfer

V. Concept to code

The program will use the datasets to evaluate the performance of each method across varying content and style, and compare outputs based on artifact presence and coherence. I will be building off of this code [] to implement the three methods. The source code uses neural style transfer using the VGG19 network architecture, a widely-used, pretrained image classification network. The program extracts the style and content, then runs a gradient descent using a Tensorflow Hub pretrained model. To run the code, simply open in CoLab and run all cells.

A. Future Steps

For testing, I will use the MS COCO (Microsoft Common Objects in Context) dataset as content images and a dataset of Van Gogh paintings for the style images.

Preprocessing must be implemented to:

1) Resize and normalize the images.
2) Extract 16x16 patches from images for patch-based approach.
3) Resize I_c for multi-scale approach.

For the multi-scale method, the style and content image will be resized through preprocessing. These images will then be iteratively passed through the preexisting style transfer model.

For the patch-based method, the style and content patches extracted from preprocessing will be compared to each other, matching the patches with the most pixel-level similarity through Euclidean distance. For two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, the Euclidean distance $d(P_1, P_2)$ is:

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Then replace each content patch with its corresponding style patch, using Guassian blending to prevent visible seams. The loss function will be adjusted to measure consistency between adjacent patches.

For regularization, I will extend the loss function to include total variation loss and gradient penalties. Weight adjustments will also be added for balancing regularization terms.

Output will be visualized through side-by-side comparisons of content, style, and output images. Additionally, I_c will be overlaid on I_g to better highlight artifacts.

REFERENCES

[1] M. B. Leon A. Gatys, Alexander S. Ecker, "Image style transfer using convolutional neural networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 9906, no. 2, pp. 694–711, 2016.

[2] Z. F. J. Y. Y. Y. . M. S. Yongcheng Jing, Yezhou Yang, "Neural style transfer: A review," *ArXiv*, no. 4, 2018.

[3] A. Helwan, "Recent advancements in gavs and style transfer," *Medium*, no. 5, 2023.

[4] T. Q. C. . M. Schmidt, "Fast patch-based style transfer of arbitrary style," *arXiv preprint arXiv:1612.04337*, no. 2, 2016.

[5] . L. F.-F. Justin Johnson, Alexandre Alahi, "Perceptual losses for real-time style transfer and super-resolution," *Proceedings of the European Conference on Computer Vision (ECCV)*, vol. 9906, no. 3, pp. 694–711, 2016.