Review of various Neural Style Transfer Methods: A Comparative Study

[1]Akash Goel, [2]Palak Singh, [3]Ragini Rani, [4]Kalash Jain

[1,2,3,4]Department of Computer Science, KIET Group of Institutions, Delhi-NCR, Ghaziabad, India

Abstract— NST, or Neural Style Transfer has revolutionized the field of image processing by allowing the amalgamation of artistic styles to photographs. First introduced by Gatys et al., NST relies on a slow and iterative optimization process. However, recent advances have introduced faster and more efficient approaches, such as Adaptive Instance Normalization (AdaIN) and Johnson's method. Gatys's method, which laid the foundation for NST, uses a CNN (Convolutional Neural Network) to extract information about the content of an image and artistic features or style of an image. Based on reducing the heterogeneity between features of content images and style images, this procedure although ultramodern, is tedious. By reconceptualize model normalization, AdaIN initiated a innovative technique for rapid amalgamation of content and style features from random images. It terminates the requirement of time-consuming optimization by focusing on real-time artistic shift with excellent mouldability. Johnson's technique involves a unique learning experience using sensory loss and previously learned interactions to demonstrate high-contrast tasks This allows for better preparation through the use of localized activities and in imagery on instantaneous analysis, resulting in a greater mixture of the two parts. This research studies the comprehensive insight into NST techniques and their evolution, highlighting potential approaches for image processing and the resolution method.

Keywords— Pioneering, Fusion, Artistic Imprint, Visual Stimuli, Iterative Optimization, Feature Statistics, Real-Time Alterations, Instance Normalization, Style Conversion, Network Functionalities, Style Adaptation

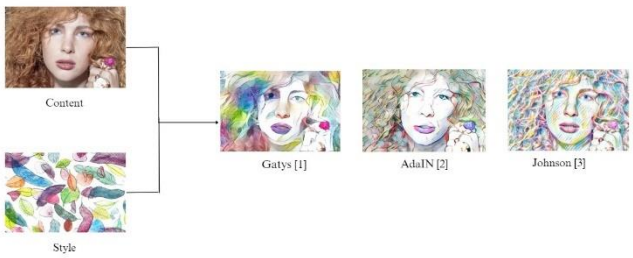
I. INTRODUCTION

Explained in the Gatys et al.'s "Neural algorithms of art form", Neural Style Transfer (NST) involves decomposition that integrates the artistic object's characteristics with another feature. Convolutional Neural Networks (CNNs), that combines artistic style with images and its mobility. Gatys and his colleagues have illustrated CNN's ability to extract both material and process information from visual sources through application. The CNN piece claims to represent the content and content of the image statistics for styles. Their approach, which resulted in custom images that frequently optimize for desired matches. CNN-featured distribution, got the field out of constraints and imposed by specific means or results from ground truth, it opened the door to the development of neural transfer. The launch of the NST has generated a lot of excitement ever since contributing to other efforts in business as well as in education seeking to innovate and expand fundamental algorithms. Moreover, it has been successfully used in a variety of industries and systems, consisting of as Prisma and Deep Forger. However, the most enormous drawback of this technique is the prolonged iterative optimization technique. In this paper, we're going to speak roughly some other crucial neural fashion shift method known as Adaptive Instance Normalization (AdaIN), a unique neural fashion transfer mechanism [2]. AdaIN combines the excellent features of two preceding strategies, supplying more desirable effectiveness and greater versatility of optimization-based totally methods. This indicates that AdaIN can be used for actual-time fashion changes, making it highly adaptable. AdaIN is a useful and powerful tool for creative style conversion on the grounds that, in evaluation to Gatys' preceding approach, it enables fast style modification and application without the want for complicated optimizations. AdaIN gives a singular viewpoint at the characteristic of example normalization (IN) in normalizing function facts that seize the fashion statistics of an photo. This technique involves editing the statistical traits of the enter content material to align with the traits of the input style, resulting in a combination of content and fashion. It's like seamlessly integrating the intrinsic features of one image with the melodious ingenious characteristics of any other.

This approach is remarkably faster than the previous method by Gatys et al., with no loss of flexibility in transferring to newstyles. It also provides runtime user control, making it a convenient and effective tool for neuromorphic transfer without the need to modify the training procedure but they were often limited to a single style.

Many problems are involved in converting images, such as sharpening a noisy image, turning a low-quality image into a high-quality image, or adding color to a grayscale image. In computer vision, tasks include recognizing objects in images or estimating object depth.

Another way to solve these image transformation problems is given by Johnson [3] and his colleagues which train a neural network to generate an output image by using the input image as a reference. However, instead of comparing each pixel of the predicted image with the actual image, which can be very detailed and slow, this method uses advanced features derived from a pretrained network to assess the resemblance between images. During training, this method, called perceptual loss, evaluates how closely the generated image resembles the real image more effectively than pixel-by-pixel comparison. When it's time to use the network trained on new images, it operates in real time, which is very convenient. Essentially, it combines the best of both worlds: accurate training using high-level features and real-time effectiveness.



In this, we compare style transfer outputs achieved with the methods of Gatys et al. [1], AdaIN [2], and Johnson [3]

I. RELATED WORK

1. Image processing and filtering - The process of creating artistic images involves simplifying and abstracting the original images. Therefore, it makes sense to explore and combine relevant image processing filters for the enhancement of a given photograph. For eg, in their work [4], Winnemoller and colleagues were the first to use bilateral filters and other than Gaussian filters to automatically generate active shapes image. Among the various image-based artistic rendering techniques, image filtering-based methods are normally simple to practice and effective. However, one limitation is their limited range of art styles.

2. Visual Texture Modelling - The field of visual texture modelling has long been central to structural synthesis [5]. Over time, two main approaches have emerged for visual texture modelling: parametric texture modelling involving a abstract statistics and non-parametric texture modelling using Markov Random Fields (MRF).

2.1 Parametric Texture modelling with summary statistics- Texture modelling can take the approach of gathering image statistics from a texture sample and use these abstract statistical properties to represent the texture. This concept was initially introduced by Julesz, who considered texture as pixel-based statistics of order N. Afterward, work in explored the use of set feedback, filtering for texture analysis, instead of direct pixel-based measurement. Based on this, Portilla and Simoncelli developed a texture model based on the response of a multi-scale directional filter and used a gradient descent method to enhance the synthesis outcome.

In a most latest parametric texture modelling method, as presented by Gatys et al. [6], was the first to leverage abstract statistics within the context of a Convolutional Neural Network (CNN). They introduced a new way to represent Gram-based to model textures, which involves examining the correlations between filter responses in different layers of a pre-trained classification network, such as the VGG network [7]. Specifically, this representation of Gram-based encodes the second-order statistics of the CNN filter response set.

2.2 Non-parametric Texture Modelling with MRFs - Another way to understand structural modelling is by using non-parametric resampling. Non-parametric methods like Markov Random Field (MRF) models suggest that style of each pixel in a texture image depends on nearby pixels. Based on this hypothesis, Efros and Leung introduced a method to synthesize individual pixels. This is achieved by exploring for similar areas in the source texture image and allocating pixels accordingly. Their work is one of the first nonparametric methods using MRF and it's like finding pieces in one picture and putting them together to make a new image.

Expanding on these advancements, Wei and Levoy took steps to enhance the efficiency of neighbourhood matching process by systematically using a fixed neighbourhood. This technique continues to align with the principles of non - parametric texture modelling using MRFs, highlighting the importance of pixel context in texture synthesis.

3. Image Reconstruction

In the field of computer vision, an important step involves filter an abstract interpretation from the input image. Image reconstruction, on the other hand, does this process in reverse; it aims to recreate the entire original image from an abstract representation. The goal here is to understand and extract the information contained in this abstract representation. Their main focus is on image reconstruction algorithms based on convolutional neural network (CNN) representations, which fall into two categories: Image-Optimisation-Based Online Image Reconstruction and Model-Optimisation-Based Offline Image Reconstruction [16].

3.1 Image-Optimisation-Based Online Image Reconstruction:

The CNN representation inversion method was originally introduced by Mahendran and Vedaldi [10]. When tasked with inverting a CNN representation, their algorithm goes through an iterative optimization process, typically starting with random noise. This process continues until the image matches the desired CNN image. Since it depends on gradient descent in image space, this method can be time consuming, especially for larger reconstructed images.

3.2 Model-Optimisation-Based Offline Image Reconstruction:

To tackle the efficiency challenges posed by [10], Dosovitskiy and Brox [8] proposed a different strategy. They suggest to pre - train a feed forward network, which offloads the computation to the training phase. During testing, the reverse process became easy with a simple network switch. This method substantially accelerates the image reconstruction procedure. In their later study [8], they improved the results by incorporating a Generative Adversarial Network (GAN) [9].

4. Neural Style Transfer was introduced by Gatys et al. [1], initially relied on slow optimization. Johnson et al. [3] accelerated the process using perceptual losses. Ulyanov et al. proposed a faster style transfer with a pre-trained style-specific feed-forward network. Later, they improved it by

introducing conditional instance normalization (CIN) [10]. Gatys et al. [11] extended CNN to handle arbitrary styles using an encoder. Cheng et al. proposed a patch-based style swapping, while Huang et al. proposed Adaptive Instance Normalization (AdaIN) [2]. Further extensions include whitening and coloring by Li et al., the decorated module follows the style of Sheng et al. and meta-networks for style transfer [3].

5. Feed-Forward Image Transformation

Various image processing methods, such as depth estimation, semantic segmentation and surface normal prediction, uses a fully-convolutional neural networks to generate detailed scene labels. They train these networks using the per-pixel classification or regression loss functions. Recent advances in surface and depth normal estimation using feed-forward convolutional networks, where training involves per-pixel regression or classification losses. Some technique surpass the per-pixel loss by incorporating techniques such as penalizing image gradients, recurrent CRF inference layers [13], or CRF loss layers to elevate the quality and overall consistency of the generated output.

5.1 Perceptual Optimization.

Recent research explores perceptual optimization, where images are generated from the optimization processes driven by high-level features extracted from convolutional networks. Objectives include maximizing class prediction scores [12] and dissecting individual features to gain a deeper understanding of network functionality. This technique also creates convincing "fooling" images with high confidence. Mahendran and Vedaldi has set the stage by introducing feature inversion, a process driven by minimizing the loss of feature reconstruction. This innovative approach reveals information stored on different network layers. In parallel, Dosovitskiy and Brox have made significant contributions by training a feed-forward network designed to invert convolutional features, providing a faster alternative to Mahendran and Vedaldi`s optimization process. However, it is important to highlight a key difference: while Dosovitskiy and Brox's network depends on per-pixel reconstruction loss, their network directly optimizes object reconstruction loss [10].

5.2 Style Transfer

Artistic style transfer introduced by Gatys et al. [1], a technique that combines image content and style through feature reconstruction and style reconstruction losses [10]. A same technique had been previously used for texture synthesis. This is computationally intensive. To mitigate this, they prepare a feed forward network to rapidly approximate style transfer solutions. At the same time, [14] also proposed a feedforward method for achieving speedy style transfer.

5.3 Image Super-Resolution.

Image super-resolution has been extensively explored with various techniques. Yang et al. [15] conducted a comprehensive evaluation of these methods, categorizing them into prediction-based, edge-based, statistical methods, patch-based and sparse dictionary approaches. Before the rise of convolutional neural networks, these methods included techniques like bilinear, bicubic, Lanczos, and more. Recent advancements include the work by [1], impressive results for enhancing the resolution of individual images. This is achieved through the utilization of a three-layer convolutional neural network combined with a per-pixel Euclidean loss, highlighting the effectiveness of this approach.

6. In the context of our project on Neural Style Transfer, recommender systems play an important role in recommending art styles that match the user's preferences and content. While our project focuses on image transformation using neural networks, the e-commerce recommendation system aims to improve product recommendations for users [18]. As part of our Neural Style Transfer project, this research highlights the power of deep learning techniques, especially CNNs, in solving complex recognition tasks. While Neural Transfer focuses on transforming visual content, this research shows the broader impact of deep learning in various applications, including handwritten character recognition for automation Postal [19].

II. METHODOLOGY

A. Gayts Method

The research presented in the main text uses VGG-Network, Certainly, they used a 19-layer convolutional neural network known for its outstanding performance in visual object recognition. In their method, they make use of all 16 convolutional layers and 5 pooling layers in this network, except for the fully connected layers. They use the publicly available VGG model within the caffe framework. To synthesize images, they chose average pooling instead of maximum pooling, which improves the gradient flow and leads to more pleasing results.

In a neural network, each layer acts as a complex set of filters, and these filters become more complex as you move through the network. As an input image ($\vec{a}$) is passed through the network, each layer processes it and generates what is known as a feature map. The quantity of feature maps in a layer corresponds to the count of distinct filters present within that layer. Each feature map is characterized by its size, which is determined by the multiplication of its height and width. An experiment is carried which involves a random noisy image to gain insights into the information gathered by each layer of the network. Gradient descent method utilizes in this experiment to adjust the image iteratively until it matches the response characteristics of a specific image of interest.

Throughout this process a comparison is made at each layer between the original image ($\vec{b}$) feature maps and generated image ($\vec{a}$) feature map. Then the squared error loss is calculated to evaluate the alignment of these feature representations. At each layer this loss serves as a metric for measuring the dissimilarity between the original image and

the generated image. The researchers are able to visualize and understand the nature of information encoded at various levels of the neural network by using this technique. Basically, this approach is valuable for reconstructing the content of an image that accurately captures the features represented in the network layer.

$$L_{content}(\vec{b}, \vec{a}, l) = 1/2\sum_{m,n}(F_{mn}^{l} - P_{mn}^{l})^{2}$$

The gradient of this loss with respect to the activations in layer l is equivalent.

$$\frac{\partial L_{content}}{\partial F_{m,n}^{l}} = \begin{cases} (F^{l} - P^{l})_{mn} & if\ F_{mn}^{l} > 0 \\ 0 & if\ F_{mn}^{l} < 0 \end{cases}$$

from there the gradient for image $\vec{a}$ can be calculated using back propagation of the standard error. Through the process of modifying the initial random image ($\vec{a}$), it is transformed until it generates a matching response in a specified layer of the CNN, mirroring that of the original image ($\vec{b}$).

At every layer within the neural network, the style representation is constructed using the CNN's response. This type of representation calculates the correlation between different filter responses, with the spatial extent of the input image taken into account through expectations. The correlations between these features are signified by the Gram matrix G^{l}, which has dimensions $N^{l}\times N^{l}$. In this matrix, each G_{mn}^{l} value indicates the scalar product of vectorized feature maps m and n in layer l.

$$G_{mn}^{l} = \sum_{k} F_{mk}^{l}\text{-}F_{nk}^{l}$$

To create textures that simulate the image's style, the gradient descent method is employed. Starting with the white noise image, adjusting it several times to replicate the original image's stylistic appearance. This adaptation requires minimizing the mean squared difference between the elements of the Gram matrix for the original image and the Gram matrix for the generated image. Therefore, assuming that $\vec{b}$ stands for the original image and $\vec{a}$ represents the generated image and B^{l} and G^{l} denote their corresponding representations in layer l. The contribution of each layer to the overall loss is then evaluated by-

$$S_{l} = \frac{1}{4N_{l}^{2}M_{l}^{2}}\sum_{m,n}(G_{mn}^{l} - B_{mn}^{l})^{2}$$

And the complete loss is

$$L_{style}(\vec{b}, \vec{a}) = \sum_{l=0}^{L} v_{l}\, S_{l}$$

The total loss contribution of each layer is determined by the coefficient v_{l}, and the specific values used will be detailed in the following results section. The analytical computation of derivative of S_{l} concerning the activations in layer l can be performed.

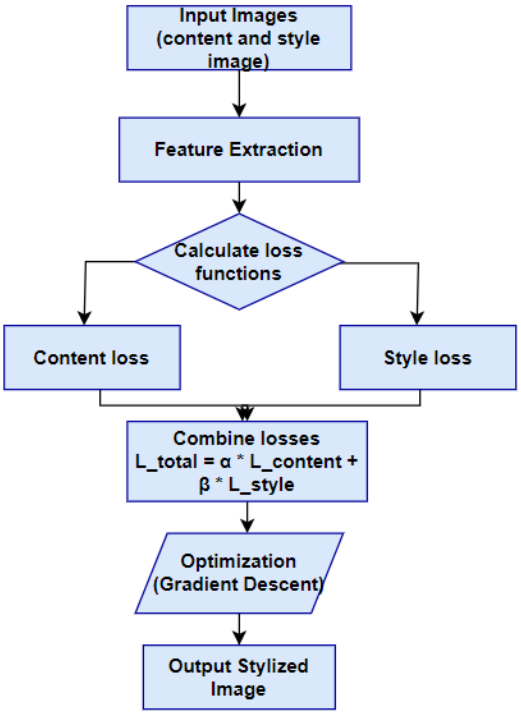
$$\partial S_{l}/\partial F_{mn}^{l} = \begin{cases} \frac{1}{N_{l}^{2}M_{l}^{2}}\left((F^{l})^{T}(G^{l} - B^{l})\right)_{ji} & if\ F_{mn}^{l} > 0 \\ 0 & if\ F_{mn}^{l} < 0 \end{cases}$$

The gradient of S^{l} associated with the activations easily calculated in the network's lower layers using the standard error backpropagation technique.

The objective is to generate images that combine a photograph's content with the style of a painting by reducing the dissimilarity between a white noise image and the manner in which the photograph's content is represented in a specific layer of the network. Additionally, here the aim is to minimize the differences between drawing style representations across multiple CNN layers, so if we have $\vec{o}$ as a photo and $\vec{q}$ as a work of art, then our objective is to minimize the loss function which is:

$$L_{total}(\vec{o}, \vec{q}, \vec{z}) = \alpha L_{content}(\vec{o}, \vec{z}) + \beta L_{style}(\vec{q}, \vec{z})$$

Here, α and β represent the weighting coefficients for the recreation of content and style, respectively [1].



Flowchart of Gayts Method

B. ADaIn Method

The style transfer network does something quite interesting It accepts two images as input: one for the content image and another for the style image. It then creates an output image that merges the content from one with the artistic style from the other. To do this, the AdaIN method offers a simple approach. It uses an encoder-decoder structure, like a translator, to achieve this magic. The encoder decodes the

input images and captures their essence, while the decoder takes this encoded information and mixes it to produce the final image with the desired style.

The content material photograph (c) and fashion picture (s) skip via the encoding community, generating function maps. These function maps are then forwarded to the AdaIN layer wherein the combined characteristic map is computed. This combined characteristic map is then surpassed to the decoder network, which is initialized randomly, and the decoder feature because the generator for the photograph converted thru neural fashion switch.

$$t = AdaIn(f_c, f_s)$$

$$T = g(t)$$

In AdaIn layer, the style function map (fs) and the content material function map (fc) play vital function inside the introduction of a combined characteristic map (t). This blended feature map is then used by the deciphering community which is represented by the function g, to generate the final stylized photograph.

Encoder

The encoder is an integral part of the pre-trained VGG19 For example. This model was initially trained on the ImageNet dataset and this section was later removed for convenience process of stylized image creation.
In AdaIN layer, features of both the content and style images are processed. The following equation is used to define this layer:

AdaIn formula

$$AdaIn(p, q) = \sigma(q)\frac{p-\mu(p)}{\sigma(p)} + \mu(q)$$

In this equation, " σ " signifies the standard deviation, and " μ " is used to represent the mean of the relevant variable.Notably, the mean and variance of the content feature map (*fc*) are adjusted to align with the mean and variance of the style feature maps (*fs*).

It's worth emphasizing that the AdaIN layer, as introduced by the authors, does not incorporate any additional parameters beyond mean and variance. This layer is not equipped with trainable parameters. This is why it's implemented as a Python function rather than being integrated as a Keras layer. The function processes both style and content feature maps, determines the mean and standard deviation of the image, and then generates an adaptive feature map normalized by the instance.

Decoder

The authors clearly state that the decoder network should mirror the architecture of the encoder network. To achieve this, they inverted the encoder configuration symmetrically. They incorporated Layers of UpSampling2D to improve the feature's map of spatial resolution.

It is important to emphasize that the authors recommend avoiding the use of any type of normalization layer within the decoder network. In fact, they demonstrate that including batch or version normalization has a negative impact on overall network performance.

Loss Functions

In constructing the loss functions which is utilized in the neural style transfer model, we apply the method proposed by the author[2]. They recommend utilizing the pre-trained VGG-19 model for computing the network loss function. It's worth emphasizing that this loss function only applies when training a decoder network.

The total loss (*Lt*) is a composite of two elements: content loss (*Lc*) and style loss (*Ls*). The lambda parameter (λ) is utilized to manage the extent of style transfer.

$$L_t = L_c + \lambda L_s$$

Content loss

The content loss is computed as the Euclidean distance between the features of the content image and the features of the following neural-style transferred image.

$$L_c = ||f(g(t)) - t||_2$$

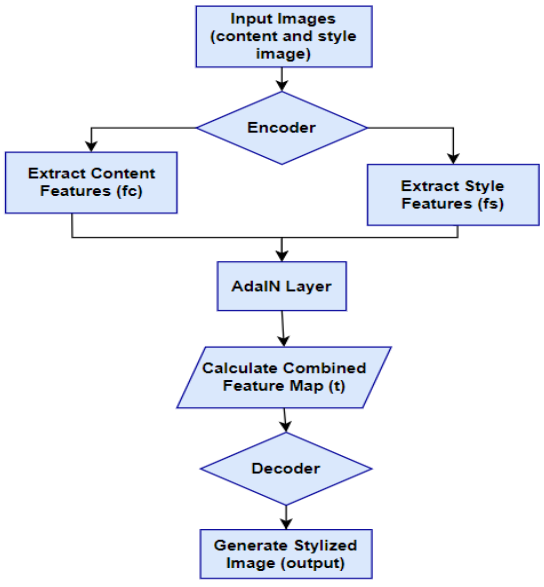
When content is lost, the authors recommend using the AdaIN *t* layers output as the content target, instead of employing the original image's features as the target. This adjustment is introduced to accelerate the convergence process (ϕ_i).

Style loss

In contrast to the more commonly employed Gram Matrix, the authors introduce a novel approach for computing style loss.This involves calculating the disparity in statistical properties, particularly mean and variance, offering a more conceptually straightforward metric. This is expressed as follows:

$$L_s = \sum_{i=1}^{L} ||\mu(\phi_i(g(t))) - \mu(\phi_i(s))||_2 + \sum_{i=1}^{L} ||\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))||_2$$

In this equation, "ϕ" represents the VGG-19 layers used to calculate the loss [2].



Flowchart of AdaIN method

C. Johnson Method

The system comprises of two main features which are: an image transformation network denoted as fv and a loss network represented as ϕ that is used to denote several loss function l_1,…., l_n.Weight v; is parameter of deep residual convolutional network which is image transformation network, it convert input images denoted as "p"and transform them into output image $\hat{q}$ through mapping $\hat{q} = fv(p)$. Each loss function computes a scalar value $l_i(\hat{q}, q_i)$ that measures the dissimilarity between the output image $\hat{q}$ and the target image q_i. These loss functions are important for training the image transformation network, which uses gradient descent to minimize a weighted combination of the various loss functions:

$$V^*=arg \min_{v} E_{p,\{q_i\}} [\sum_{i=1} \lambda_i \, l_i(\text{fv(p)}, q_i)]$$

To overcome the limitations associated with per-pixel loss and improve the ability to capture perceptual and semantic distinctions between images, we adopt a unique approach. These methods are based on the idea that convolutional neural networks, initially trained for image classification, have already internalized the perceptual and semantic details that we seek to evaluate with our loss functions. Therefore, we employ a pre-trained ϕ network to classify images as a constant loss network to determine our loss functions.
We employ a loss network, denoted as ϕ to determine the object reconstruction loss $\ell_{feat}^{\phi}$and the style reconstruction loss ℓ_{style}^{ϕ} to calculate dissimilarity between the style and content images. With each input image represented as p, we set our sights on content target q_s and a style target $\hat{q}$. To change the style, the target of the content q_c, align with the input image p and the output image $\hat{q}$, endeavors to fuse the content of p=q_c with the desired style represented as q_s. The network is trained according to these style objectives. In the context of ultra-high resolution, where input p is a low-resolution input, content target q_c is a high-resolution ground-truth image, and style reconstruction loss does not play a prominent role in the training process of the network according to the super-resolution coefficient.

Image Transformation Networks

This use a design that bypasses traditional pooling layers, opting for split and split structures.
For super resolution, we use residual blocks and specific convolution layers based on the up sampling factor "f", rather than relying on bicubic interpolation. This approach is computationally efficient and increases the effective size of receptive field.

Perceptual Loss Functions

We establish two separate perceptual loss functions that allow us to evaluate high-level perceptual and semantic disparities between images. These loss functions are derived from a pre-trained classification network known as ϕ, which itself takes the form of a deep convolutional neural network. In our experimental setups, we systematically utilize a 16-layer VGG network pre-trained on ImageNet to serve as the ϕ loss network.

Feature Reconstruction Loss

Instead of emphasizing a precise pixel-to-pixel match between the output image $\hat{q}$=$fv(p)$ and the target image q. We achieve this by examining the activation of the jth layer in network ϕ during processing of image p, denoted by ϕ_j. If j represents a convolutional layer, then ϕ_j (p) will be a feature map of size $C_j \times H_j \times V_j$.

$$\ell_{feat}^{\phi,j} (\hat{q},q)= \frac{1}{C_j H_j V_j} || \phi_j(\hat{q}) - \phi_j (q)||_2^2$$

Style Reconstruction Loss

The loss of feature reconstruction loss functions act as a penalty on the output image $\hat{q}$ f it diverges from the target q in terms of content. We utilize the activations $\phi_j(p)$ at the jth layer of the network ϕ for the input p. These activations materialize as a feature map of size $C_j \times H_j \times V_j$. Now, the Gram matrix G_j^{ϕ}(p) to be the $C_j \times C_j$ matrix constructed by calculating the relationship between these features are given by

$$G_j^{\phi}(\text{p})\text{c, c}' = \frac{1}{C_j H_j V_j} \sum_{h=1}^{H_j} \sum_{w=1}^{V_j} \phi_j(p)_{h,v,c} \, \phi_j(p)_{h,v,c}'$$

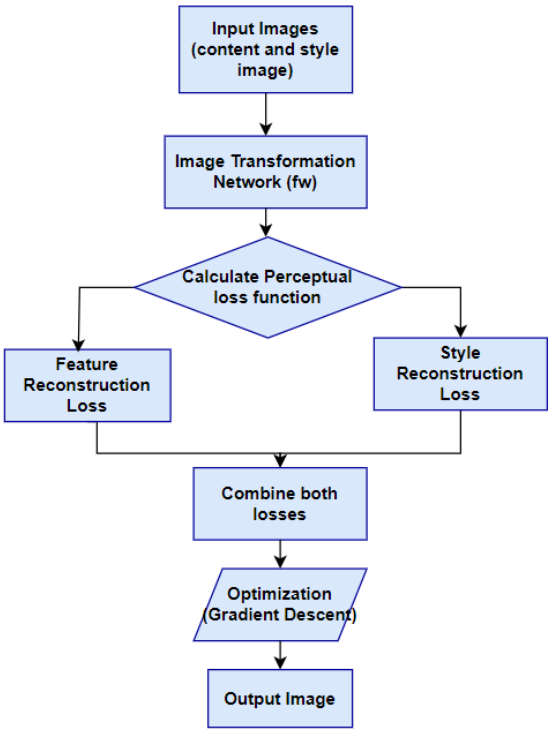
The Gram matrix G_j^{ϕ}(p) is a representation of how features in $\phi_j(p)$ co-activate. This can be seen as a measure of the uncentered covariance between these features, with treating each grid location as a distinct data point. To compute G_j^{ϕ}(p) efficiently, reshape $\phi_j(p)$ into a matrix ψ and calculate it as $\psi\psi^T$, which is then normalized by $C_j \times H_j V_j$.

$$G_j^{\phi}(\text{p}) = \psi\psi^T / \, C_j H_j V_j.$$

The style reconstruction loss can be calculated using the squared Frobenius norm, which represents the difference between the Gram matrix of the generated image and the target image.

$$\ell_{style}^{\phi,j} (\hat{q},q) = || \, G_j^{\phi}(\hat{q}) - G_j^{\phi}(\text{q}) \, ||_F^2.$$

Pixel loss, which quantifies the dissimilarity between the output image $\hat{q}$ and the target q, is determined by calculating the squared Euclidean distance normalized by their size C×H×V, defined as $\ell_{pixel}(\hat{q}, q) = ||\hat{q}, q||_2^2 /$ CHV. It's applicable when a ground-truth target q is available for the network to match [3].



Flowchart of Johnson's method

COMPARISON AND ANALYSIS

Approach	Efficiency	Real-time Performance	Quality
Gatys	Moderate (50-70%)	no	High (80-90%)
AdaIN	High (70-90%)	yes	Moderate (70-80%)
Johnson	High (70-90%)	yes	Moderate (60-70%)

Comparative Analysis of Neural Style Transfer Approaches

In our study, we conducted a comprehensive comparative analysis of three important neural transfer methods, namely Gatys neural transfer, AdaIN (real-time arbitrary transfer), and Justin Johnson's Fast Neural Style Transfer. This analysis focuses on three important aspects: efficiency, real-time performance, and quality of the type conversion results. Neural style transfer from Gatys, known for quality art style transfer, has relatively lower efficiency due to its optimization-based approach, which often requires significant computational resources. While it delivers exceptional artistic results, Gayts method may not be suitable for the application that requires real time style transfer. On the other hand, AdaIN method known for its efficiency, it gives the perfect solution for real-time type applications. This method provides a perfect balance of speed and quality and also ensures satisfactory results of a wide range of practical applications. Johnson method is highly effective because it provides a harmonious compromise between speed and quality. Whereas Johnson's results may not reach the same artistic quality levels as Gatys' method but they still offer a good quality result that would be reasonable for various tasks. When we have to choose between AdaIN and Johnson then the factors like artistic quality, efficiency, and real-time performance should be used for the decision-making process.

CONCLUSION

In this research paper we have discuss the three methods of neural style transfer that involves transforming images by combining artistic styles with photographs. In this firstly we explore the spearheading work of Gatys [1] and colleagues, who presented the concept of combining aesthetic styles with photos utilizing convolutional neural networks. Their method is robust in nature but this method can be slow and iterative, hindering the efficiency of the style transfer.

Secondly, we discuss the Adaptive Instance Normalization (AdaIn) [2] method, this method overcome the limitations of Gatys method. AdaIn method balances the flexibility and speed by utilizing instance normalization to effectively blend content and style. One of the key advantages of AdaIn method is that it allows real-time style switching with various styles, giving users control during runtime without the need to modify the training process. This innovative approach has transformed the way style transfer is conducted. It offers more efficient and flexible solution.
And the third notable method discussed in this paper is the Johnson [3] method, which utilizes perceptual loss functions and loss networks to measure perceptual and semantic disparities in images. Johnson method basically enhances the quality of image transformation and super-resolution tasks by using
pre-trained image classification networks. This helps in creating more accurate and high-quality image transformations, ultimately improving the overall efficiency of the process. The paper also investigates picture change systems and also focus on their effectiveness and design principles

ACKNOWLEDGMENT

This paper introduces an advance approach of neural style transfer, firstly it discusses the Gatys method of neural style transfer using CNN. To overcome the limitation of Gatys we executed AdaIn method for real-time style transfer. AdaIn method offers a seamless integration of styles onto images.

Additionally, this paper also discusses the Johnson's perceptual loss methodology which combine both speed and precision in the style transfer process. In this, the related areas of style transfer such as image reconstruction, image processing and texture modelling are discussed. Moreover, we express our gratitude towards the spearheading work of Gatys, AdaIn and Johnson, whose commitments have essentially progressed the field of neural style transfer and visual processing.

REFERENCES

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks” in IEEE Conference on Computer Vision and Pattern Recognition, 2016

[2] X. Huang and S. Belongie, “Arbitrary style transfer in real-time with adaptive instance normalization” in IEEE International Conference on Computer Vision (ICCV), 2017

[3] J. Johnson, A. Alahi, and L. Fei-Fei, “Perceptual losses for real-time style transfer and super-resolution,” in European Conference on Computer Vision, 2016

[4] L.-Y. Wei and M. Levoy, “Fast texture synthesis using tree structured vector quantization,” in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000

[5] Michael Elad and Peyman Milanfar, “Style transfer via texture synthesis”, in IEEE Transactions on Image Processing, 2017.

[6] Leon A. Gatys, Alexander S. Ecker and Matthias Bethge, “Texture synthesis using convolutional neural networks,” in part of Advances in Neural Information Processing Systems 28, 2015

[7] Karen Simonyan and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition” published in International Conference on Learning Representation, 2014.

[8] Alexey Dosovitskiy; Thomas Brox, “Inverting visual representations with convolutional networks,” in IEEE Conference on Computer Vision and Pattern Recognition, 2016

[9] Alexey Dosovitskiy and Thomas Brox, “Generating images with perceptual similarity metrics based on deep networks,” in Advances in Neural Information, 2016

[10] Aravindh Mahendran and Andrea Vedaldi, “Understanding deep image representations by inverting them” in CVPR, 2014

[11] Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., Lipson, H “Understanding neural networks through deep visualization” in ICML Deep Learning Workshop, 2015

[12] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, “A neural algorithm of artistic style” in Computer Vision and Pattern Recognition, 2015

[13] Fayao Liu, Chunhua Shen, Guosheng Lin, “Deep convolutional neural fields for depth estimation from a single image” in CVPR, 2015

[14] Chuan Li & Michael Wand, “Precomputed real-time texture synthesis with markovian generative adversarial networks” in European Conference on Computer Vision, 2016

[15] Chih-Yuan Yang, Chao Ma & Ming-Hsuan Yang, “Single-image super-resolution: a benchmark” in European Conference on Computer Vision, 2014

[16] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu and Mingli Song, “Neural Style Transfer: A Review”, published in IEEE Transactions on Visualization and Computer Graphics 26(11):3365-3385, 2020

[17] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, Philip Torr “Deeply supervised salient object detection with short connections”, in IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[18] Harsh Khatter, Shifa Arif, Utsav Singh, Sarthak Mathur and Satvik Jain, “Product Recommendation System for E-Commerce using Collaborative Filtering and Textual Clustering”, in Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021

[19] Sandhya Sharma, Sheifali Gupta, Deepali Gupta, Sapna Juneja, Gaurav Singal, Gaurav Dhiman, and Sandeep Kautish, “Recognition of Gurmukhi Handwritten City Names Using Deep Learning and Cloud Computing” in Hindawi Limited, 2022