IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 26, 2021, accepted September 11, 2021, date of publication September 15, 2021,
date of current version September 30, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3112996

Neural Style Transfer: A Critical Review

AKHIL SINGH[1], VAIBHAV JAISWAL[1], GAURAV JOSHI[1], ADITH SANJEEVE[1], SHILPA GITE[1,2], AND KETAN KOTECHA[2,3]

[1]Department of Computer Science and Information Technology, Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune 412115, India
[2]Symbiosis Centre of Applied AI (SCAAI), Symbiosis International (Deemed) University, Pune 412115, India
[3]Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune 412115, India

Corresponding authors: Shilpa Gite (shilpa.gite@sitpune.edu.in) and Ketan Kotecha (head@scaai.siu.edu.in)

This work was supported by Symbiosis International (Deemed University), Pune, India.

ABSTRACT Neural Style Transfer (NST) is a class of software algorithms that allows us to transform scenes, change/edit the environment of a media with the help of a Neural Network. NST finds use in image and video editing software allowing image stylization based on a general model, unlike traditional methods. This made NST a trending topic in the entertainment industry as professional editors/media producers create media faster and offer the general public recreational use. In this paper, the current progress in Neural Style Transfer with all related aspects such as still images and videos is presented critically. The authors looked at the different architectures used and compared their advantages and limitations. Multiple literature reviews focus on either the Neural Style Transfer (of images) or cover Generative Adversarial Networks (GANs) that generate video. As per the authors' knowledge, this is the only research article that looks at image and video style transfer, particularly mobile devices with high potential usage. This article also reviewed the challenges faced in applyingvideo neural style transfer in real-time on mobile devices and presents research gaps with future research directions. NST, a fascinating deep learning application, has considerable research and application potential in the coming years.

INDEX TERMS Style transfer, video style transfer, mobile, convolutional neural networks, generative adversarial networks.

I. INTRODUCTION

Since its conception, videos have been considered a popular multimedia tool for various functions like Education, entertainment, communication, etc. Videos have become more and more popular as the effort needed to make them keeps dropping thanks to advancements in Cameras and, more particularly, mobile cameras. Today, an average user uses mobile devices to capture videos rather than expensive dedicated setups [1]. On the other hand, entertainment producers use dedicated hardware and editing tools to create picturesque scenes with the help of Compute Generated Imagery (CGI) software like [2] and [3].

There are multiple resources, approaches, improvements, and implementations since the first Generative Adversarial Network was presented by Goodfellow *et al.* [7]. As of now, NST is extremely popular and widely used to edit images to create a host of effects (E.g., Prisma App) (Gatys *et al.* [13]) (Liu *et al.* [18]). Recently developments have been observed to use NST for video style transfer (Ruder *et al.* [31]), (Huang *et al.* [32]). This has significant applications like entertainment to directly transform the scene or parts, usually taking hours of manual work and supervision. It can also be used for recreational purposes fusing with Augmented Reality to create a virtual world modeled after the real one (Dudzik *et al.* [33]).

Generative Adversarial Networks (GANs) are often used to produce or synthesize data since conception (Goodfellow *et al.* [7]). This makes GANs a potential candidate for generating Images/Videos given a set of inputs that control its structure and texture. The paper focuses on Generative Adversarial Networks (GANs) developments and summarizes the advancements made to date (up to April 2021). It also describes basic techniques currently being used to transform videos and then move onto the NST-based techniques. To understand the developments and have comparisons, all categorized papers are reviewed into four parts, as shown in fig. 1. Each part has differentobjectives and key takeaways, such as advantages, limitations, research gaps, and future scope.

The papers selected for review were found using Scopus and Web of Science databases with the search terms such

The associate editor coordinating the review of this manuscript and approving it for publication was Longxiang Gao.

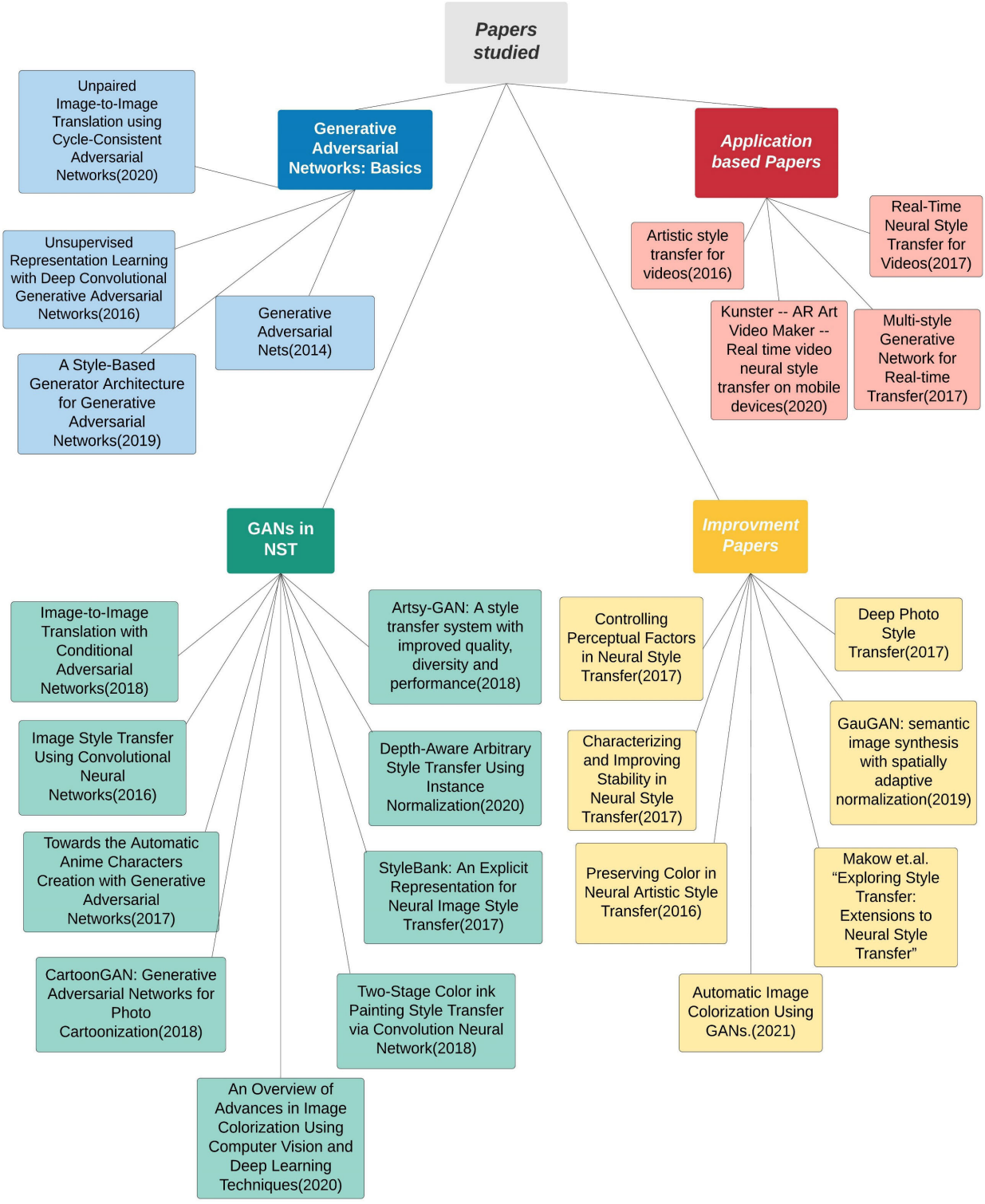

FIGURE 1. An overview and categorization of the papers studied in this review.

as ‘‘video neural style transfer,’’ ‘‘real-time video neural style transfer,’’ ‘‘generative adversarial networks,’’ ‘‘video neural style transfer on mobile devices,’’ ‘‘video style transfer improvement.’’ Shortlisted papers with code implementations publicly available (on GitHub or similar services) and based on the quality of the videos they generate (as shown in their demonstrations/Readme).

There is currently no benchmark dataset for Neural Style Transfer. MS-COCO and Cityscape are the two datasets most frequently utilized in the experiments within the

TABLE 1. A short summary of review papers and key contributions.

Reference	Objectives and Topics	Paper Theme	Year
[4]	The paper discusses a comprehensive overview of the current progress, a taxonomy of current algorithms in NST. It discusses several evaluation methods of comparison, a discussion of various applications of NST and open problems for future research.	Detailed Reviews of NST papers up to March 2018	2019
[5]	The paper presents a short survey of major techniques of doing neural style transfer on images, and very briefly examines one way of extending neural style transfer to videos.	Short survey of Neural Style Transfer on Images and Videos	2018
[6]	The paper presents a short survey of the current progress of NST from two aspects: the image optimization-based method and model-optimization-based method. It compares different types of the NST algorithms, applications of some proposals for future research.	Brief survey of Model-Optimization-Based Neural Style Transfer Method	2020

papersreviewed. These datasets are primarily used for object detection and recognition. Still, they may also be used as content photos for training various models since each dataset was around 330K and 25K in size, respectively. The style dataset was either scraped from online sources such as Danbooru, Safebooru, and Videvo.net or was created according to the requirements of the problem statement, which were in the range of 1k to 2k images.

While reviewing the literature, a few research gaps such as platform-related, dataset-related, and architecture-related deficiencies were identified. Hardware limitations are the primary cause of platform-related gaps. In the absence of a benchmark dataset and benchmark metrics, there exist data-related research gaps. Lastly, architecture-related gaps concern how model parameters change based on the dataset need. These gaps are further discussed in detail in section VIII.

As presented in table 1, there are a total of 3 review papers available in the NST domain. Out of those 3, only paper [4] can be considered as a comprehensive review paper. The novelty of our paper lies in terms of the latest papers review till Aug 2021, stating all related facets of NST. There are four major sections to the paper. The first part of the paper covers the basics of GANs, their types, and how they work; the second part covers the contemporary architecture of GANs with NST and how they work; the third part of the paper covers the improvements that can be made to GANs while applying NST to it, such as deep photo style transfer; and the fourth part is about how we can use NST along with GAN architecture on a real-time basis.

Highlights of this literature review are listed below:

- Qualitative analysis of the latest GAN architectures models, along with their advantages and limitations, is discussed.
- A summary and in-depth analysis of neural style transfer for both images and video are given, emphasizing mobile devices.
- Most relevant research papers on the Neural Style Transfer were explicitly identified focused on real-time NST, which narrows down the research in video style transfer.
- Research gaps and future research directions are also discussed in a detailed study of the challenges in applying forvideo neural style transfer in real-time on mobile devices.

Fig. 1 shows the papers reviewed in this research study and their categorization as per paper flow.

II. GENERATIVE ADVERSARIAL NETWORKS OVERVIEW

The first part deals with papers that define the basics of most GANs. These papers are essentially the backbone, as most other articles follow their pathway by improving upon or making amends to them. GANs generate data based on previously learned patterns and regularities as the model finds these patterns. Deep learning suits generative models as they can effectively recognize patterns in input data.

A. GENERATIVE ADVERSARIAL NETWORKS

[7] explores the framework, which was new around then for making generative models in a loosely organized cycle, wherein training two models: a generative model G which gets the details, and a discriminative model D that calculates the likelihood that an image comes from training examples instead of G. The arrangement technique of G would be to raise the likelihood of D creation a mistake. This arrangement resembles a more modest than usual max two-player game. Next to optional limits G and D, a response occurs, with G recovering the arrangement of data course and D up to 0.5 everywhere. For the situation where G and D are represented by multilayer perceptron, they always set up the fundamental structure with backpropagation. The technique used here is to get the most extreme probability of doling out the correct mark to both preparing models and tests from G and at the same time preparing G to limit log(1-D(G(z))).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}\left[\log\left(1 - D(G(\boldsymbol{z}))\right)\right] \quad (1)$$

B. STYLE BASED GENERATOR ADVERSARIAL NETWORKS

Generator improvement has seen less attention and improvement compared to Discriminator. To enhance the picture

TABLE 2. FIDS in FFHQ for networks trained for different percentages of training examples by allowing mixing regularization. (Karras *et al.* [8]).

Mixing Regularization	Number of latents during testing			
	1	2	3	5
E 0%	4.42	8.22	12.88	17.41
50%	4.41	6.10	8.71	11.61
90%	**4.40**	**5.11**	6.88	9.03
100%	4.83	5.17	**6.63**	**8.40**

quality produced by the Generator, [8] introduced a Style transfer literature-based generator. In this model, the Generator is trained with the Progressive GAN setup of Karras as a baseline. The following are details of the model:

1. Traditionally the Generator is provided with a latentcode through the input of the first layer of the feed-forward network. At the same time, in the new approach, it is omitted altogether and is started with a learned constant.
2. Provided a latent code z in the non-linear mapping network and latent input space Z, f: Z → W first generates w ∈ W
3. After the mapping is done, learned affine transformation specializes w to styles : $y = (ys, yb)$ which operates after each convolution layer of the generative network and controls the normalization of the generative network G.
4. The normalization technique used is adaptive instance normalization (AdaIN), the (2) for the same is:

$$\text{AdaI N}\left(\mathbf{x}_i, \mathbf{y}\right) = \mathbf{y}_{s,i}\frac{\mathbf{x}_i - \mu\left(\mathbf{x}_i\right)}{\sigma\left(\mathbf{x}_i\right)} + \mathbf{y}_{b,i} \tag{2}$$

   Over here to get the AdaIN between $\mathbf{x}_i$ and $\mathbf{y}$,firstly finding the distance between $\mathbf{x}_i$ and the mean of $\mathbf{x}_i$ ($\mu\left(\mathbf{x}_i\right)$) further dividing it by standard deviation of $\mathbf{x}_i$ ($\sigma\left(\mathbf{x}_i\right)$), then to scale, the value is multipliedit by $\mathbf{y}_{s,i}$ and bias it by $\mathbf{y}_{b,i}$.
5. The Generator is then given direct noise input, which allows it to generate stochastically. The noise input is uncorrelated noise input generated via a single channel of images. Dedicated noise input is given into each layer of the synthesis layer.
6. Using the learned pre-feature scaling factor, the noise image is first transmitted on all feature maps, and then the corresponding convolution layer output is applied.

The above changes in the Generator lead to the following observation and improvement:

- 20% improvement in FID over traditional Generator.
- This makes it possible by modifying the styles by scale to track image synthesis. Then the display of the mapping network and geometric transformation that preserves content and style images to produce new images from the trained distribution and generative network based on a series of styles to create examples. This resulted locally in each style effect, meaning only some portion of the image would be influenced by changing a small part of the Style.

The use of regularization mixing, which is a given number of images, is generated during training using two random

TABLE 3. In FFHQ for different generator architectures, separability scores and perceptual route lengths (lower is better).(Karras *et al.* [8]).

Method	Path Length		Separability
	Full	End	
Traditional generator	412.0	415.3	10.78
Style-based generator	446.2	376.6	3.61
Add noise inputs	200.5	160.6	3.54
+Mixing 50%	231.5	182.1	3.51
+Mixing 90%	234.0	195.9	3.79

TABLE 4. The effect of a mapping network in FFHQ. Karras *et al.* [8]).

Method	FID	Path Length		Separability
		Full	End	
Traditional 0	5.25	412.0	415.3	10.78
Traditional 8	4.87	896.3	902.0	170.29
Traditional 8	4.87	324.5	212.2	6.52
Style-based 0	5.06	283.5	285.5	9.88
Style-based 1	4.60	219.9	209.4	6.81
Style-based 2	4.43	**217.8**	199.9	6.25
Style-based 8	**4.40**	234.0	**195.9**	**3.79**

latent codes. Precisely, w1, w2 controls the Style of two different codes, z1, z2 across the mapping network so that w1 is applied before and w2 after the crossover point. This approach prevents the network from assuming that the object style is correlated.

- After each convolution, the architecture adds per-pixel noise, resulting in noise only affecting the stochastic aspects leaving intact the function and aspects at a high level.
- Global effects such as illumination, etc., were seen to be coherently regulated, whereas the noise was applied to each pixel separately, ideally only suitable for stochastic variation. When the network is monitoring, i.e., the Discriminatorpenalizes the pose with the noise, leading to inconsistency in space. This way, without clear instructions, the network can learn to use global and local networks properly. The perceptual path length is lowered by using the style-based Generator, as seen below in the picture:
- It is shown that increasing the mapping network's depth enhances both image quality and separability.

C. DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS

Unsupervised learning using Convolutional Neural networks (CNN) has seen less attention than supervised learning and its adoption in computer vision applications. To bridge the gap, [9] introduced Deep Convolutional GANs (DCGANs).

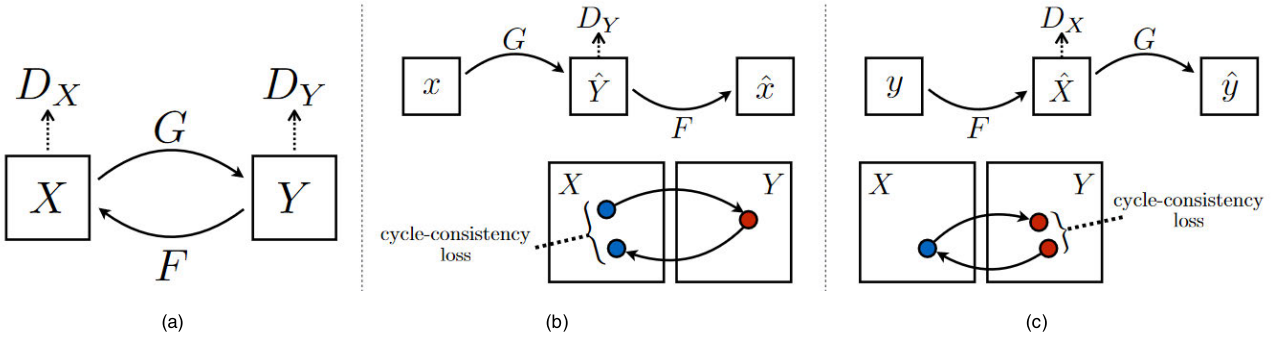

FIGURE 2. (a) CycleGAN model schematic showing the two generators and discriminators along with the image domains. (b) The transforms used to compute forward cyclical consistency loss. (c) The transforms used to compute backward cyclical consistency loss.

In this method, discriminators are trained for the image classification task, and generators have vector calculations that allow more control over generated images.

Following guidelines are proposed to create stable convolutional GANs:

1. Use strided convolution in Discriminator and fractional-strided convolution in Generator, enabling the model to tune upsampling and downsampling itself.
2. Batch normalization usage is done in the generators and the discriminators. This prevents the generators from mode collapse.
3. Remove dense hidden layers. Connect the highest Conv. features to input and output of both parts, which showed promising results. For Discriminator, flatten and feed the last Conv. Layer into sigmoid output.
4. For Generator, use ReLU activation and Tanh (only in the final layer).
5. In the Discriminator, make use of LeakyReLU in layers.

These architectural changes result in regular training and a model capable of handling high resolutions.

Testing on the CIFAR-10 and Street View House Numbers dataset (SVHN) dataset confirmed the impressive performance of DCGANS. However, it still falls short of Exemplar CNNs [10]. Another point to improve upon is that even with fewer feature maps in the Discriminator, it has a more extensive feature vector size, increasing training size at higher resolutions.

D. CYCLE CONSISTENT ADVERSARIAL NETWORKS

Unlike the Deep Convolution GANs, CycleGANs allow image translation on unpaired data. [11] achieve this the concept of ''Cyclical Consistency'' meaning that if two Generators, ''G'' and ''F,'' are trained to be inverses of each other than virtually, $F(G(X)) \approx X$. [11] introduce a second generator that takes the outputs of the first one to and tries to produce the actual input image. By training two GANs whose, generators perform inverses of each other, [11] decouples the translation's style and structural aspects (one model handles the style transfer while the other enforces structure). A key takeaway is that these models do not need paired data to train due to this structure. The loss function is thus modified to:

$$\begin{aligned} L(G, DY) &= E_{y\sim p(y)}\left[(DY(y)-1)^2\right] \\ &\quad + E_{x\sim p(x)}\left[DY(G(x))^2\right] \end{aligned} \tag{3}$$

$$\begin{aligned} L(F, DX) &= E_{x\sim p(x)}\left[(DX(x)-1)^2\right] \\ &\quad + E_{y\sim p(y)}\left[DX(F(Y))^2\right] \end{aligned} \tag{4}$$

$$\begin{aligned} L_{cyc}(G, F) &= E_{x\sim p(x)}\left[\| F(G(x)) - x \|\right] \\ &\quad + E_{y\sim p(y)}\left[\| G(F(y)) - y \|\right] \end{aligned} \tag{5}$$

$$L = L(G, DY) + L(F, DX) + \lambda * L_{cyc}(G, F) \tag{6}$$

where X and Y are two image domains, G is a generator transforming an image from domain X to Y. F is a generatori transforming an image from domain Y to X. DY is the discriminator concerning G (identifies real/generated images in Y domain). DX is the discriminator concerning F (identifies real/generated images in X domain). G(x) is the image generated by G on an input image x such that $x \in X$ and F(y) is the image generated by F on an input image y such that $y \in Y$. Thus, Equations (3) and (4) compute the Adversarial losses for the two GANs. In contrast, Equation (5) computes the cyclical consistency loss by comparing input images x and y to their remapped/generated versions, F(G(x)) and G(F(x)), respectively. Equation (6) describes the total loss of CycleGAN combining the adversarial and cyclical losses. The transformations are diagrammed by [11] in Figure 2.

The generators use a Resnet based architecture and a few encoder-decoder layers, while the discriminators use a PatchGAN architecture to focus on local structural details. The results show that CycleGANs perform exceptionally well on all test metrics barring the Pix2Pix model. The model's limitation is that it fails whenever an image sampled from a different distribution is input.

1) OBSERVATIONS

In summary, [7] defines a basic GAN with its objective function and training procedure. However, unconditional (which cannot get precise results) and uncontrollable (controlling

FIGURE 3. Output of CycleGANs. Left-most images are inputs, middle images are corresponding style transfer from the first Generator G. Right-most images are the reconstruction of inputs by second generator F. (Zhu *et al.* [11]).

the individual features used for generation). [8] builds upon this by modifying the Generator to allow control over disentangled features. [9] improves the architecture by introducing a Deep Convolutional Neural Network. [11] focuses on Unpaired Image Style Transfer presented with the help of ''cyclical consistency.''

III. GENERATIVE ADVERSARIAL NETWORKS IN NEURAL STYLE TRANSFER

Now architectures prevalent and in use concerning Neural Style Transfer (NST) are discussed. These papers look at proposing a new architecture and employing new methods. NST first appeared in Gatys *et al.* [13]. The approach takes a content image and applies the textures of the Style given. NST then gained momentum as many works followed, increasing the quality of images generated or generating them faster than Gatys *et al.* [13]. These efficiencies and/or effect improvements paved the way for faster image editing (E.g., Adobe Image stylization) or recreational use (E.g., Prisma App).

A. CONDITIONAL ADVERSARIAL NETWORKS FOR STYLE TRANSFER

Conditional GANs (cGANs) introduce image-to-image translation and a loss function to allow the models' training. It removes the usage of hand-engineered loss functions or mapping functions. [12] aims to create a common framework that predicts a particular set of pixels based on another given set of pixels. Instead of treating the output space as ''unconditional'' from the input image, cGANs use a structured loss function, considering the structural differences between input and generated images. Optimizing this loss function allows the generated images to be structurally related or ''conditioned'' as per the input image. The Generator has an

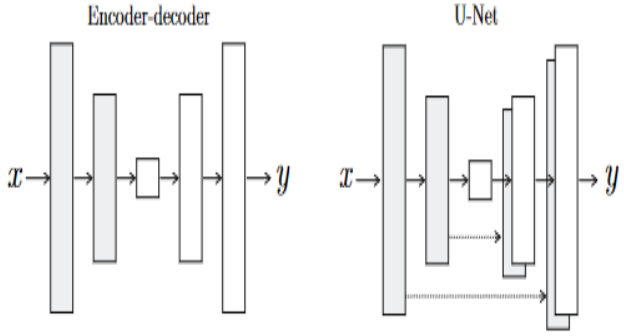

FIGURE 4. Encoder-decoder vs U-Net architecture. (Isola *et al.* [12].

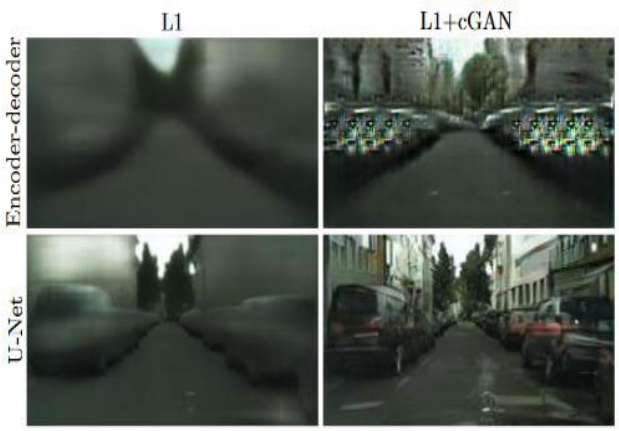

FIGURE 5. Introducing U-Net allows higher quality of generated images. (Isola *et al.* [12].

architecture based on U-Net, whereasthe Discriminator has a PatchGAN based architecture. The PatchGAN architecture is shown to be useful as it penalizes local structural differences. The effect of locality or ''patch size'' is also studied. The loss function is given as:

$$L_{cGAN} = E_{x,y}\left(log\left(D\left(x, y\right)\right)\right) + E_{x,z}\left(log\left(1 - D\left(x, G\left(x, z\right)\right)\right)\right) \quad (7)$$

where G and D are the Generator and Discriminator networks, x and y are content and style images, respectively, and z is a random noise vector that gets learned to produce the mapping G: {x, z} → y. The Discriminator is now fed ''x'' or input image as an input. In addition, an L1 distance term is added to make the generated images closer to ground truth and avoid blurred images:

$$L_{L1}(G) = E_{x,y,z}[||y - G(x, z)||] \quad (8)$$

Thus, the final objective is given as:

$$L_t = L_{cGAN}(G, D) + \lambda.L_{L1}(G) \quad (9)$$

where G and D are the Generator and Discriminator networks, L_{cGAN} is the conditional loss given in (7) and $L_{L1}(G)$ is the L1 loss of the generator as given in (8). The total loss is L_t

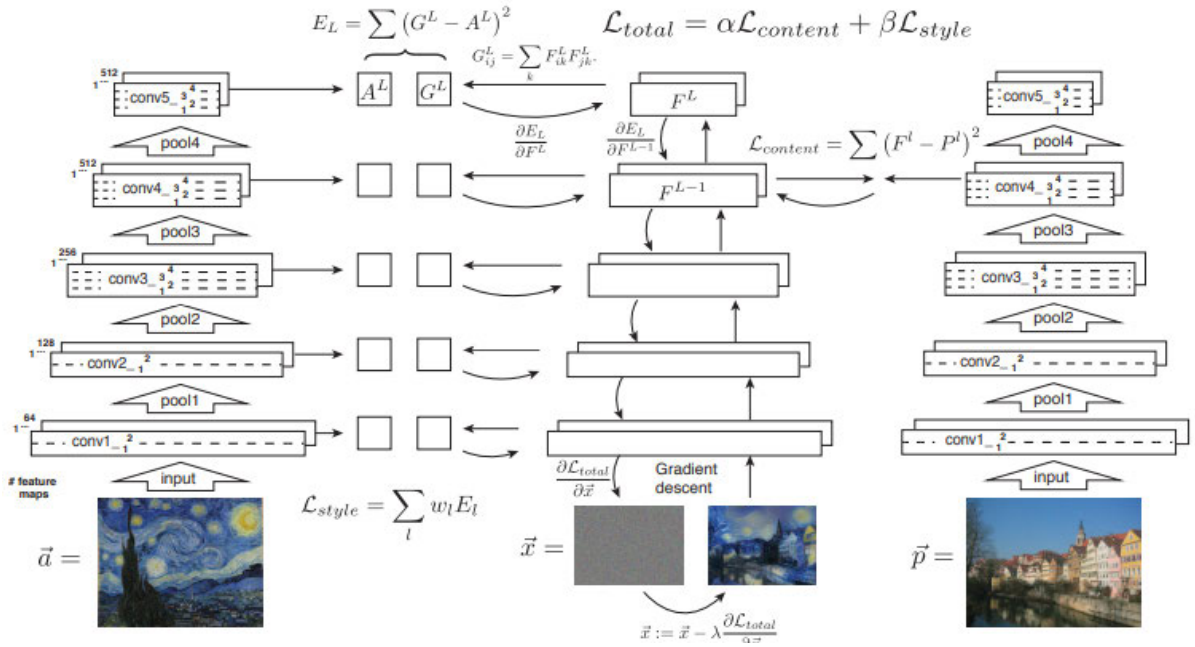

FIGURE 6. Style transfer algorithm. (Gatys *et al.* [13]).

and λ is a weight used to alter the importance of L1 loss in the total loss.

Noise is provided in dropouts and not as inputs as the models ignored the latter. In addition, the U-Net architecture introduces skip connections, which allow low-level details to be transferred easily between the input and output images. Meanwhile, PatchGANs, as discriminators, focuses on localized information more. Another significant advantage is that PatchGANs can work with a smaller subset of pixels at a time, decreasing the number of parameters, computation, and time required for discriminator predictions.

It is seen that having a small patch size causes loss of spatial features (structure of image) with useful spectral features (colorful images). As the transition towards a larger patch size, a balance of spatial and spectral features producing a crisp image. However, Increasing the patch size beyond this ''balance point'' causes a lower quality image to be generated.

Another plus for PatchGAN is that the Discriminator can be applied to large images. The places where the model fails to be good at are:

1. Sparse input images (Images with shallow structural details)
2. Unusual Inputs (Input which is not like training data).

B. IMAGE STYLE TRANSFER USING CNN

It is difficult to render an image's semantic content differently since it lacks representations that explicitly provide semantic information. To solve the limitation of using only low-level image characteristics of the target image, [11] presents an Artistic Style neural algorithm that can isolate and recombine the content of images (style texture) and generate the images using those Styles. The image representations used here are derived from the optimized Convolution Neural Network for object recognition, explicitly providing high-level image details. Overall, the approach combines CNN-based parametric texture models to invert their representations of the image.

The method used is:

- The standardized version of the 19-layer VGG network includes 16 convolutional and five pooling layers.
- By scaling weights, the network was normalized such that the mean activation of each convolutional filter over images and positions was equivalent to one.
- Image synthesis was done by using average pooling as it was seen that it provided a better result.
- For Content representation:
  - One can perform gradient descent to display image data on several levels of a white noise picture to locate another image that fits the feature responses of the content image.
    Let $\vec{p}$- original image,
    $\vec{x}$- generated image,
    P l- feature representation of $\vec{p}$ in layer l,
    F l- feature representation of $\vec{p}$ in layer l.

The squared-error loss between the two feature representations is defined as:

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F^l_{ij} - P^l_{ij}\right)^2 \tag{10}$$

The derivative of this loss w.r.t activations in layer l equals ∂L content

$$\frac{\partial \mathcal{L}_{\text{content}}}{\partial F^l_{ij}} = \begin{cases} \left(F^l - P^l\right)_{ij} & \text{if } F^l_{ij} > 0 \\ 0 & \text{if } F^l_{ij} < 0 \end{cases} \tag{11}$$

- Style representation:
  - To acquire a style portrayal of an input picture, feature space was utilized to capture textural data. The feature space can be based on top of the filter response of the model layer, which comprises of connection between various reactions, where exceptional cases are taken over the spatial degree of feature maps.
  - Feature correlation, given by: Gram matrix $G^l \in \mathcal{R}^{N_l \times N_i}$, where G^l_{ij} is the inner product between the vectorized feature maps i and j in layer l:

$$G^l_{ij} = \sum\nolimits_k F^l_{ik} F^l_{jk} \tag{12}$$

  - The total loss function seen was:

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum\nolimits_{i,j} \left(G^l_{ij} - A^l_{ij}\right)^2 \tag{13}$$

  - Total style loss:

$$\mathcal{L}_{\text{style}}\left(\vec{a}, \vec{x}\right) = \sum\nolimits_{l=0}^{L} w_l E_l \tag{14}$$

  - The derivative of E w.r.t. the activation functions in layers l can be computed analytically:

$$\frac{\partial E_l}{\partial F^l_{ij}} = \begin{cases} \frac{1}{N_l^2 M_i^2} \left(\left(F^l\right)^{\mathrm{T}} \left(G^l - A^l\right)\right)_{ji} \\ \qquad \text{if } F^l_{ij} > 0 \\ 0 \\ \qquad \text{if } F'_{ij} < 0 \end{cases} \tag{15}$$

- Style transfer:
  - the loss function jointly minimized distance between feature representations of the white noise of two images (content and Style):

$$\mathcal{L}_{\text{total}}\left(\vec{p}, \vec{a}, \vec{x}\right) = \alpha \mathcal{L}_{\text{content}}\left(\vec{p}, \vec{x}\right) + \beta \mathcal{L}_{\text{style}}\left(\vec{a}, \vec{x}\right) \tag{16}$$

The content image was resized to style image always before computing its feature representations to keep them for comparable sizes.

Results seen for the suggested image style transfer are:

- Both the content and the image style type are easily separable in CNN, and to produce new meaningful visuals;then the changes can be represented individually.
- Due to the many layers in the image synthesis, which layers fit the content and style representations were shown.
- The picture is cleaner if the matching is done up to higher layers initializing noise before initialization of gradient descent leads to the generation of arbitrary numbers of new images.
- The algorithm provides photo-realistic style transfer; an example can be seen in Fig. 7.

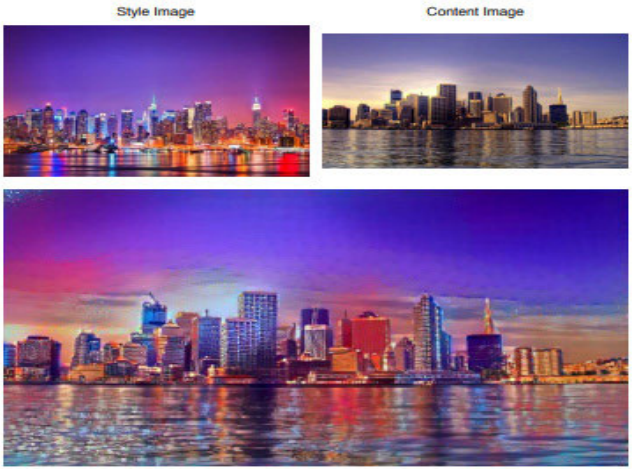

FIGURE 7. Photorealistic style transfer. (Gatys *et al.* [13]).

C. TOWARDS THE AUTOMATIC ANIME CHARACTERS CREATION

The most common problem in generating faces is that they get distorted on some features and get blurred. [14] addresses this problem in both data and model aspects. [14] provides three contributions for generating anime faces:

1. GAN model based on DRAGAN architecture.
2. A suitable clean anime facial dataset comprising of high-quality images which are collected from Getchu (Japanese game selling website)
3. An approach to train GAN from untagged images.

Tags are assigned to the dataset using Illustration2Vec (a CNN-based Tag estimation tool). This tool can detect and tag 512 different types of attributes. After tags are set, 34 tags are selected, which are suitable for the task at hand. In this way, any untagged dataset can be processed and prepared, which vastly opens data collection sources.

Model architecture is based on DRAGAN proposed by Kodaliet al. [15]. DRAGAN has the least computation cost than other GAN variants and is much faster to train. Generator architecture is shown in Fig. 8, which is based on a modified version of SRResNet [16]. It consists of 16 Residual Blocks and three feature upscaling blocks. The discriminator architecture is depicted in Fig. 9. It has 11 Residual Blocks and a dense layer that acts as an attribute classifier. The proposed model was compared with a standard DRAGAN model with DCGAN Generator based on Fréchet inception Distance (FID) scores. Table 5 shows, the proposed model has lower average FID scores identifying it as a better model.

Figure 10 shows the samples generated from the model. These samples are transparent, have sharp images, and have good diversity.

The only drawback of this model is that it cannot handle super-resolution. It is observed that the high-resolution images generated using this model have undesirable artifacts, which made the results messy.

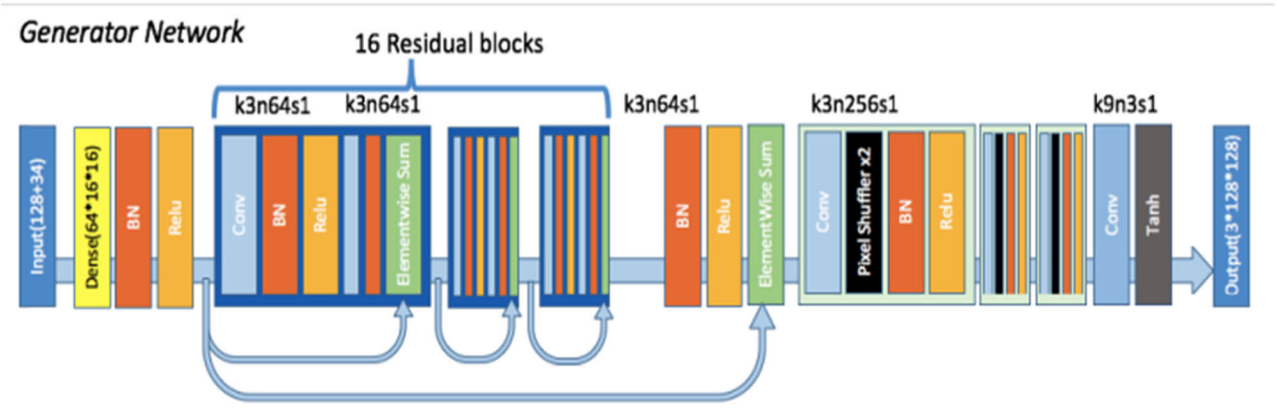

FIGURE 8. Generator architecture. (Jin *et al.* [14]).

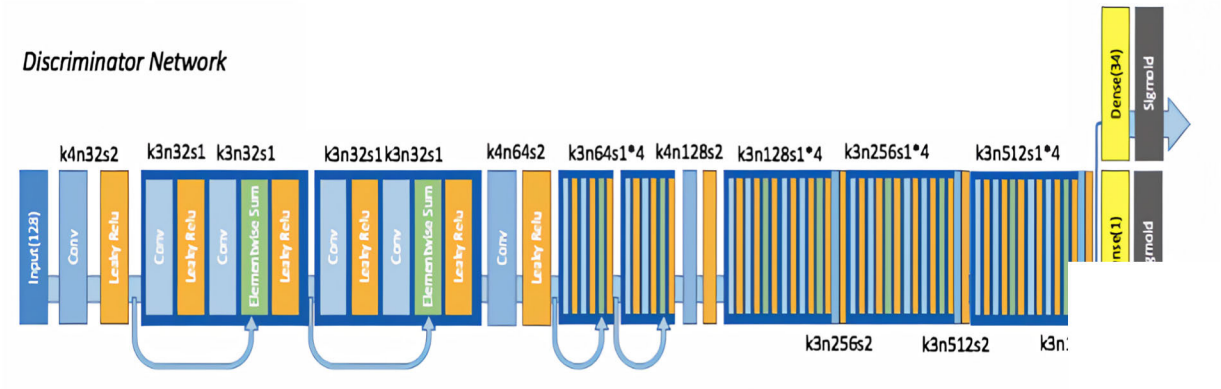

FIGURE 9. Discriminator architecture. (Jin *et al.* [14]).

FIGURE 10. Generated samples. (Jin *et al.* [14]).

TABLE 5. FID of proposed model and baseline model. (Jin *et al.* [14]).

Model	Average FID	MaxFID-MinFID
DCGAN Generator+DRAGAN	5974.96	85.63
Our Model	4607.56	122.96

D. CartoonGAN: GENERATIVE ADVERSARIAL NETWORKS FOR PHOTO CARTOONIZATION

[17] proposes a solution to convert real-world scenery images into cartoon-style images. The unique characteristic, smooth shading, and textures of cartoon-style images prove significant challenges to existing methods based on texture-based loss functions. [17] proposes CartoonGAN, a new GAN framework that can take unpaired images for training to tackle this problem. CartoonGAN architecture is shown in Fig. 11 Generator G, which comprises one flat Conv. Block proceeded by two down-Conv. Blocks are meant to perform compression as well as encoding of an input image. The content and manifold part are made up of eight residual blocks. Finally, two up-convolution blocks and a convolution layer create the cartoon-style output images. Discriminator D consists of flat layers preceded by two strided Conv. blocks to reduce resolution and encodefeatures. The final layers are made up of a feature construction block with convolution layers to obtain a classification.

The overall loss has two parts: adversarial loss and content loss described as

$$\mathrm{L}\left(\mathrm{G}, \mathrm{D}\right) = \mathcal{L}_{\mathrm{adv}}\left(\mathrm{G}, \mathrm{D}\right) + \omega \mathcal{L}_{\mathrm{con}}\left(\mathrm{G}, \mathrm{D}\right) \tag{17}$$

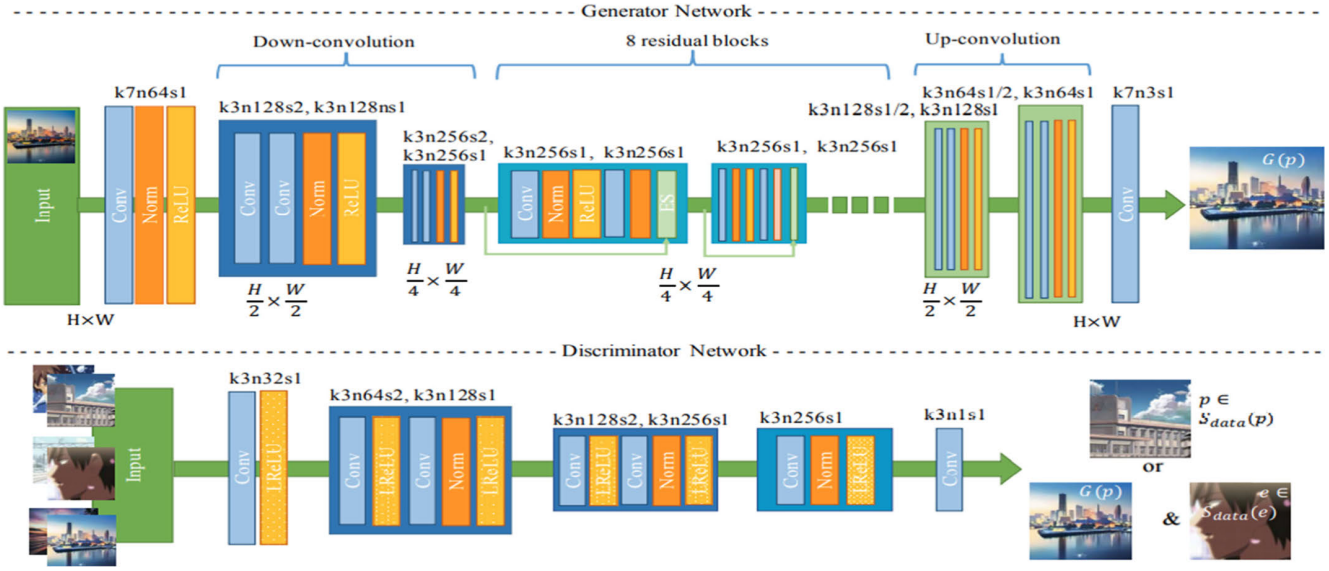

FIGURE 11. Proposed generator discriminator architecture. (Chen *et al.* [17].

Here ω is the weight by which canbe limited to the content retention amount from the input.

Adversarial loss L_{adv} (G, D) is an edge-promoting loss defined as:

$$\mathcal{L}_{adv}(G, D) = \mathbb{E}_{c_i \sim S_{\text{data}}(c)}\left[\log D(c_i)\right] + \mathbb{E}_{e_j \sim S_{\text{data}}(e)}\left[\log\left(1 - D\left(e_j\right)\right)\right] + \mathbb{E}_{p_k \sim S_{\text{data}}(p)}\left[\log\left(1 - D\left(G\left(p_k\right)\right)\right)\right] \quad (18)$$

Here, the Generator G outputs a generated image $G(p_k)$ for each photo p_k in the photo manifold p. e_j is a cartoon image without precise edges and c_i is the corresponding actual image. $D(c_i)$, $D\left(e_j\right)$, $D(G(p_k))$ are the probabilities of the discriminator D assigning correct labels to the actual image, cartoon image without clear edge, and generated image, respectively.

Content Loss L_{con} (G, D), which has a feature map in a pre-trained VGG network defined by:

$$\mathcal{L}_{\text{con}}(G, D) = \mathbb{E}_{p_i \sim S_{\text{data}}(p)}\left[||VGG_l(G_i(p_i)) - VGG_l(p_i)||_1\right] \quad (19)$$

Here, l refers to the feature maps of specific VGG layer.

Along with the model, an initialization phase is proposed to improve the GAN model's convergence. In this phase, the Generator is trained only with semantic content loss (19) and can reconstruct only the input images' content. The training data is unpaired, consisting of real-world and cartoon images that are all resized to 256 × 256. There are 5402 real-world training images. Cartoon images comprise of Makoto Shinkai (4,573), Mamoru (4,212), Miyazaki Hayao (3,617), and Paprika (2,302) style images.

As Fig. 12 shows, outputs from CartoonGAN are compared with NST [11] and CycleGAN [11] outputs trained on the same dataset. The Figure demonstrates the inability of NST and CycleGAN to handle cartoon style well. NST using only style imagery cannot control theStyle thoroughly because the local regions are styled differently. This leads to inconsistent artifacts. Similarly, results from CycleGAN are also unable to understand and depict the cartoon style appropriately. The absence of Identity loss renders it unable to preserve input image content. Even with identity loss, the results are unsatisfactory. The results clearly show that CartoonGAN effectively transforms real-world scenery images into cartoon-style efficiency and high quality. It efficiently performs much better than other top stylization methods.

E. ARTSY–GAN A STYLE TRANSFER SYSTEM

[18] introduces a novel method for GAN-based style transfer termed Artsy-GAN. The problemwith current approaches, such as using CycleGAN, is the slow training of these models due to their complexity. Another disadvantage is the source of randomness, which is limited to input images. [18] proposes three ways to tackle these problems:

1. Using perceptual loss instead of reconstructing to improve training speed and quality.
2. Using chroma sub-sampling to process the images improves inference/prediction speed and makes the model compact by reducing size.
3. Improving the diversity in generated output by appending noise to the Generator's input and pairing it with the loss function would force it to develop a variety of details for the same image.

Fig. 13 shows the model architecture of the Generator. The inputs are a 3-channel color image (RGB) with noise added to each channel. The Generator has three branches, each of which receives the same input but produces different output image channels that are converted back into RBG by a model

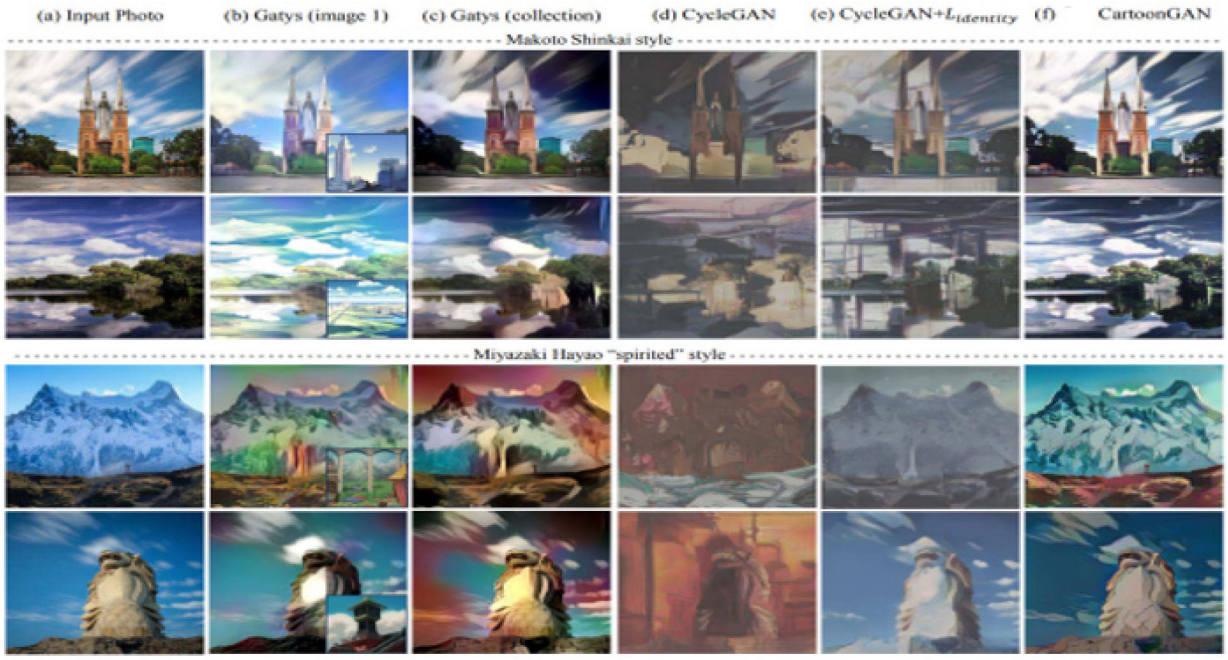

FIGURE 12. Generated output comparison of CartoonGAN, CycleGAN and NST. (Chen *et al.* [17]).

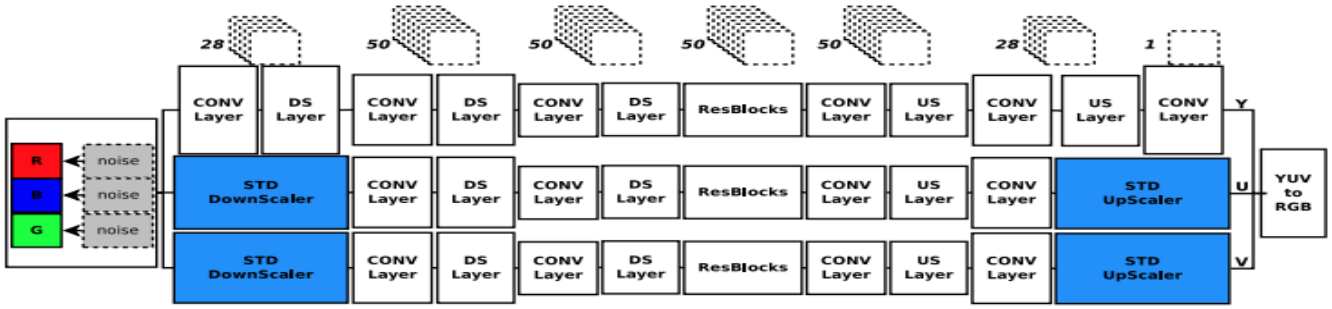

FIGURE 13. Architecture of generator. (Liu *et al.* [18]).

at the end of the network. The discriminator architecture is the same as CycleGAN using 70×70 PatchGANs [11], [17], [18].

The objective loss function is made up of three types of losses and is defined as

$$\mathcal{L}_{\text{FULL}} = \mathcal{L}_{\text{GAN}} + \alpha\mathcal{L}_{\text{PERCEPTUAL}} + \beta\mathcal{L}_{\text{DIVERSITY}} \quad (20)$$

where α and β control the significance of losses.

Here, the loss functions are:

1. An adversarial loss L_{GAN} for equalizing distribution of domains. It is defined as

$$\mathcal{L}_{\text{GAN}} = \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[\left(D\left(G\left(x,z\right)\right) - L_{\text{real}}\right)^2\right] \quad (21)$$

where, L_{real} is the table of actual data, z is a noise tensor, $G\left(x,z\right)$ is a produced image from generator G,and D is the discriminator.

2. Diversity loss $\text{L}_{\text{DIVERSITY}}$ to improve diversity in generate/output images, which are defined as

$$\text{L}_{\text{DIVERSITY}} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{mean}_{j\neq 1}^{-}||g(z_i)-g(z_j)|| + E + E} \quad (22)$$

where, N is the number of input noises as well as several outputs.

3. A perceptual loss $\text{L}_{\text{PERCEPTUAL}}$ to overcome the unconstrained problem by keeping the object and content in the output and can be described as:

$$\text{L}_{\text{PERCEPTUAL}} = \frac{1}{C_j H_j w_j}\left[||\phi_j(x) - \phi_j(G(x,z))^2\right] \quad (23)$$

where, $\phi_j\left(x\right)$ is the output of the j-th layer of feature encoder network ϕ for image x. If the j-th layer is a

FIGURE 14. Results compared with Cycle-GAN. (Liu *et al.* [18]).

TABLE 6. Comparison of FID of Artsy-GAN and CycleGAN. (Liu *et al.* [18]).

	Van Gogh	Monet	Ukiyoe	Cezanne
CycleGAN	180.37	125.82	206.05	162.83
Artsy-GAN	**168.03**	**125.65**	**198.32**	**160.36**

TABLE 7. Processing time comparison in GPU Tesla M40. (Liu *et al.* [18]).

Resolution	CycleGAN	Ours	Speed Up
640x480	0.037 +/- 0.008	0.034s +/- 0.003	9.33%
1024x768	0.081 +/- 0.031	0.036s +/- 0.004	55.60%
1280x720	0.106s +/- 0.032	0.037s +/- 0.007	64.85%
1280x960	0.125s +/- 0.054	0.040s +/- 0.007	67.91%
1280x1024	0.145s +/- 0.047	0.042s +/- 0.047	71.23%
1960x1080	0.176s +/- 0.093	0.044s +/- 0.008	74.96%

convolutional later then, $\phi_j(x)$ will be a feature map of $C_j * H_j * w_j$

Comparison of Artsy-GAN is made with CycleGAN based on FID, processing time, and diversity in generated images. Table 6 shows that Artsy-GAN has lower FID scores across all the styles for which both models are trained.

Table 7 Shows that Artsy-GAN is 9.33% faster than Cycle-GAN at the minimum resolution(640 × 480) taken and up to 74.96 % faster at the highest resolution (1960 × 1080). As the resolution increases, the difference in processing times increases, proving that Artsy-GAN is much faster and well-suited for higher resolution images.

Fig. 14 shows that CycleGAN output images are very similar for the same input image with shallow diversity even after adding noise to input images. Whereas Artsy-GAN output images vary significantly, confirming its high diversity. Finally, the proposed Artsy-GAN is a better, faster, and more diverse method for style transfer, which easily outperforms other SOTA methods depicted by the result. The perceptual loss proposed can also be used for different stylings, such as oil paintings with vibrant textures.

F. DEPTH AWARE STYLE TRANSFER

After the style transfers have been rendered using a different picture style, the depth of the content picture has not been reproduced. It is seen that those traditional methods like additional regularization in the optimization of the loss function, etc., are either ineffective in computing or require a different trained neural style network. AdaInapproach of Huang *et al.* [32] enables effective arbitrary style transfer to the content image. The depth map of the content image cannot be replicated. [20] proposed an extension to the AdaIn method to preserve the depth map by applying variable stylization strength. The comparison showed in the image Fig. 15.

The technique is the depth-aware AdaIN (DA-AdaIN), which works with varied strength: closer areas are less stylized, whereasdistant regions representing a background have a more stylistic feature. Based on the following styling, AdaIn applies the Style evenly to the content image:

$$\hat{I} = g\left(AdaIN\left(f\left(I_c\right), f\left(I_s\right)\right)\right) \tag{24}$$

where,

- I_c - Content Image
- I_s - Style Image
- f(·) is an encoder

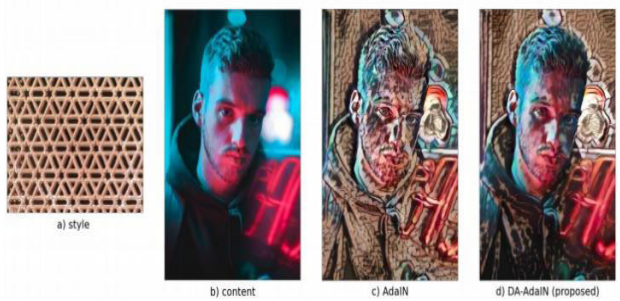

FIGURE 15. Comparison between proposed DA-AdaIn to AdaIn methods. (Kitov *et al.* [20]).

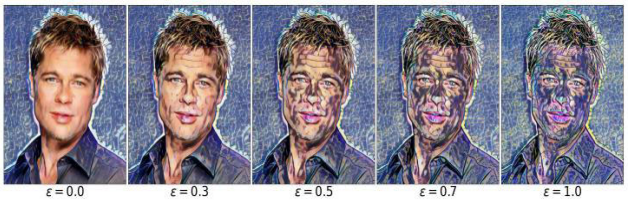

FIGURE 16. Result of style transfer based on depth contrast parameter β, $\varepsilon = 0$. (Kitov *et al.* [20]).

- g(·) Is a decoder trained for appropriate stylization with the encoder.
- AdaIN(x, y) is a variant of instance normalization.

The extension proposed is:

- AddStyle using varied strengths, based on your camera proximity, in various areas of the content image.
- Closer places must be preserved in the forefront so that less stylized; remote areas are considered more stylistic backgrounds. The hyperparameter $\alpha = [0, 1]$ can be regulated by the following formula in a standard stylizer strength check:

$$\hat{I} = g\left(\alpha f\left(I_c\right) + (1-\alpha)\,\mathrm{AdaIN}\left(f\left(I_c\right), f\left(I_s\right)\right)\right) \tag{25}$$

- Since $f\left(I_c\right)$ is the actual unaltered content encoder representation, whereas AdaI N $\left(f\left(I_c\right), f\left(I_s\right)\right)$. Is a completely styled encoder representation. To manage spatially varying strength, the modified formula can be used

$$\hat{I} = g\left(P \odot f\left(I_c\right) + (1-P) \odot \mathrm{AdaIN}\left(f\left(I_c\right), f\left(I_s\right)\right)\right) \tag{26}$$

- where $P \in \mathbb{R}^{H_e \times W_c}$ is stylization strength map shows repeated element multiplication for each channel for each spatial position in the content encoder representation:

$$\{P \odot F\}_{cij} = P_{ij} F_{cij} \tag{27}$$

- The algorithm has two hyperparameters:
  - $\beta > 0$ controls the prominence of the proximity map around its mean value.

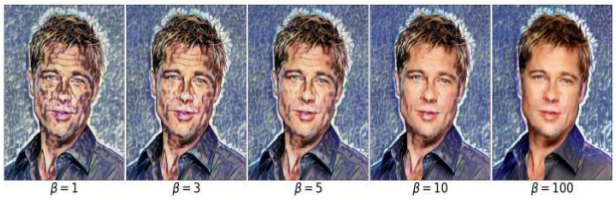

FIGURE 17. Result of style transfer based on proximity offset parameter ε, $\beta = 20$. (Kitov *et al.* [20].

  - $\varepsilon \in [0, 1]$ controls minimal offset of the image regions from the camera.

Image result based on different hyperparameters values:

G. StyleBank: AN EXPLICIT REPRESENTATION FOR NEURAL IMAGE STYLE TRANSFER

StyleBank is made of many convolutional filter banks, each of which explicitly reflects one Style and transmits the Style of neural images. To convert a picture to a particular style, the appropriate filter bank is used on top of the intermediate feature embedding generated by a single auto-encoder. The StyleBank and the auto-encoder are concurrently learned, with the auto-encoder encoding no style information due to the flexibility provided by the explicit filter bank representation. Additionally, it supports incremental learning to add a new image style by learning a new filter bank while keeping the auto-encoder unchanged. The explicit style representation and the adaptable network design enable us to combine styles at the picture and area levels.

To investigate an explicit representation for Style, [21] revisit traditional texton (referred to as the essential element of texture) mapping methods, in which mapping a texton to the target location is equivalent to convolution between a texton and a Delta function (indicating sampling positions) in the image space.

In response, [21] offersStyleBank, a collection of different convolution filter banks, each reflecting a distinct style. The matching filter bank is convolved with the intermediate feature embedding generated by a single autoencoder, which decomposes the original picture into several feature maps to convert a picture to a specific style.

In comparison to previously published neural style transfer networks, the proposed neural style transfer network is novel in the following ways:

- This method offers an explicit representation of styles using this way. After learning, the network canisolate styles from content.
- This technique enables region-based style transfer due to the explicit style representation. This is not possible with existing neural style transfer networks, but it is possible with classical texture transfer.
- This method enables concurrent training of many styles with a single auto-encoder and progressive learning of a new style without modifying the auto-encoder.

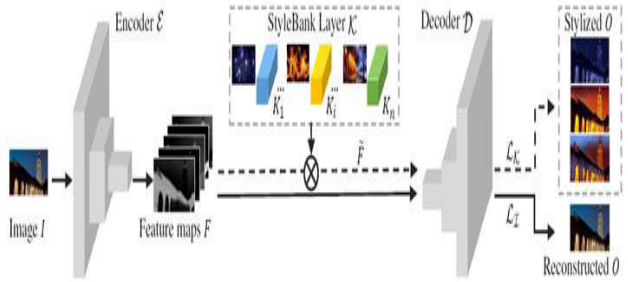

FIGURE 18. The network model. (Chen *et al.* [17]).

[21] construct a feed-forward network based on a simple image autoencoder (Figure 18), which converts the input picture (i.e., the content image) to the feature space via the encoder subnetwork.

H. TWO-STAGE COLOR INK PAINTING STYLE TRANSFER VIA CNN

[22] proposes the best approach to move bloom pictures to paint ink painting. Not quite the same as a common neural style move technique, the report presents a way that imitates the creation of shading ink painting. It can be viewed as two specific steps – edge marking and picture colorization. Rather than utilizing edge identification calculations, the line drawing is taken to help and adventure the CNN-based neural style move technique to get line drawing. Concerning picture colorization, the GAN-based neural style move strategy is utilized.

The framework comprises two segments: a line extraction model and an image colorization model. The line extraction model changes blossom photograph content into a line drawing through the planning x1 = f1(content). The image colorization model colorizes the line drawing x1 to give output y through the planning y = g (·). In this methodology, f1(content) is anticipated to be the planning as follows. At that point, an estimated shading portrayal could likewise be acquired from content ≈ f1 − 1(x1). Thus, both line and estimated shading portrayal in substance picture is developed through the planning f1(content). With matched information, contingent GAN, prepared in a directed way, may incorporate substance picture with client determined Style. Hence, adapted photos from generator fool discriminator, yet meet the necessities for shading ink painting tone.

1) AS REFERENCED BEFORE, THERE ARE TWO PRIMARY MODELS

First, the Line Extraction model includes an image colorization model that removes most lines of blossoms and leaves in substance pictures. The Line Extraction model is utilized to characterize loss capacities by estimating contrasts in substance and tastefulness between highlights extricated from pictures. In the training stage, the flower image is taken by the image colorization model, and output picture x1 is created in a like manner. The line extraction model is fixed during the preparing stage, and output highlights are utilized in picture loss capacities.

$$L = \lambda cLc(F(x_{content}), F(x1)) + \lambda sLs(G(xs), G(x1)) \quad (28)$$

In equation 28, Lc(·) is the Euclidean distance between content portrayals of substance pictures and adapted pictures. Ls(·) is the squared Frobenius standard of the contrast between the Gram lattices of style picture and adapted picture. F and G are the element change capacities. Secondly, Image Colorization Network, which further has Conditional GAN and DualGAN, is used to experiment and check which one gives the better output. In this way, when the Generator and Discriminator are adapted to additional data, it studies a strict model. As shown in Fig. 19, line drawing is taken over by both modeling and line extraction models. As the line drawing can be noticed, the Discriminator can observe how the Generator transforms the information line to a suitable photo. In this manner, the Discriminator will, in general, be more solid to separate the created photos from the significant.

In DualGAN, an unaided learning system figures out how to decipher pictures from area X to those in space Y and figure out how to upset the errand. During this case, as appeared in Fig. 21, two picture sets from 2 areas, explicitly, line drawing set (space X) and shading ink painting set (area Y), are taken care of into two gatherings of GAN. Generator GA initially changes line drawing x1 from space X into adapted composition picture y. Y is turned at that moment into a regenerated line image x1. In the meantime, GB generators convert the shading style of ink painting in an adapted line image x1 to a recreated shading ink color. L1 distance is obtained to live the remaking mistake, adding to the GAN target. Hence, generators figure out how to get pictures with perceptual authenticity.

2) OBSERVATIONS

A summary of contributions is presented in Table 8. One peculiar limitation seen is that the models tend to fail at higher resolution images.

IV. ADVANCEMENT PAPERS

This set of papers present advancements to current architectures. These advancementsallow different types of control to the Style Transfer by improving Color control, Stability, Spatial Control, and other vital aspects which enhance the quality of generated images.

A. PERCEPTUAL FACTOR CONTROL IN NEURAL STYLE TRANSFER

[23] presents an extension to the existing methods by proposing spatial, color, and scale control over a generated image's features. By breaking down the perceptual factors into these features, more appealing images can be generated that avoid common pitfalls. Finally, [23] shows a method to incorporate this control into already existing processes. The identification of perceptual factors is the key to producing higher-quality

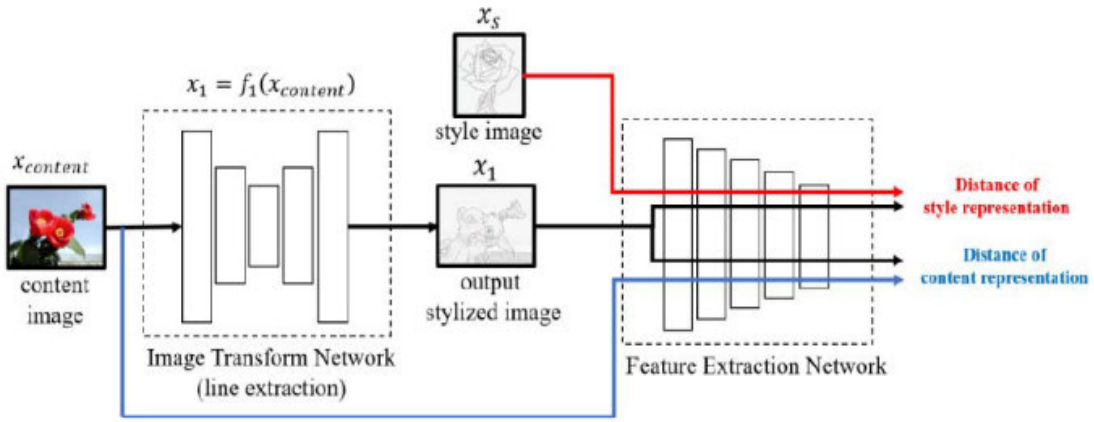

FIGURE 19. Line extraction model architecture. (Zheng and Zhang [22]).

TABLE 8. A short summary of architecture-based papers and their key contributions.

Ref. No.	Paper	Contributions
[12]	Isola et al. 2018	Differentiates and penalizes generated images based on output types
[13]	Gatys et al. 2016	Separates the style and content parts of the process (like [11]) and breaks down the total loss into two parts, each addressing a different issue.
[14]	Jin et al. 2017	Provides a way to generate facial features with less distortion.
[17]	Chen et al. 2018	Uses a novel initialization phase that helps in faster convergence and loss functions pertinent to cartoonish image generation.
[18]	Liu et al. 2018	Achieves faster convergence using perceptual loss, chroma subsampling and noise addition.
[20]	Kitov et al. 2020	Addresses depth preservation by adding variable stylization strength.
[21]	Chen et al. 2017	Allows simultaneous training of multiple styles using the concept of "Style Banks"
[22]	Zheng et al. 2018	Breaks down stylization into two stages: marking the edges and actual colorization.

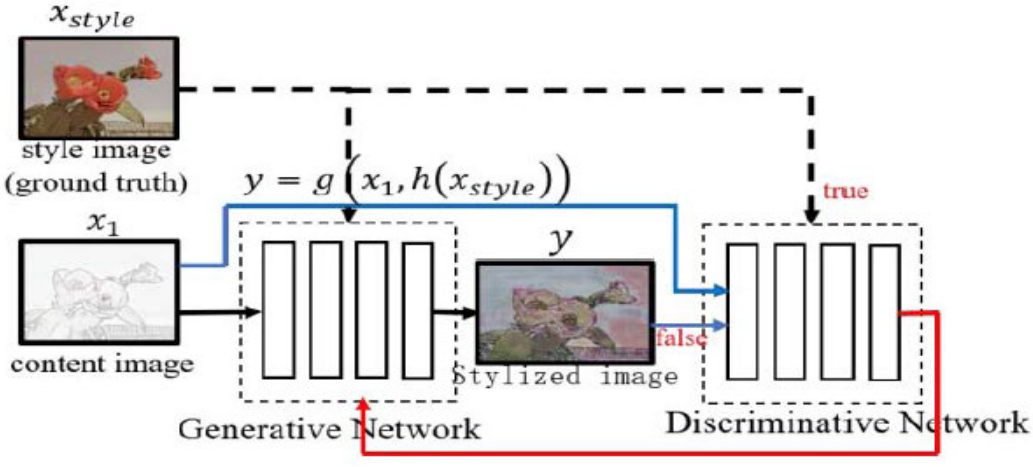

FIGURE 20. The organization model of image colorization model. (Zheng and Zhang [22]).

images. Spatial control implies controlling which region of the style image is applied to each region of the content image. This helps as different regions have different styling, and mapping them incorrectly can cause visual artifacts. The first method to do this uses Guidance-based Gram Matrices, where each image is provided with a spatial guidance channel indicating which region should be applied to what Style. This involves computing a Spatially Guided Feature Map for R regions and L layers as:

$$F_l^r(x)_{[:,i]} = T_1^r \circ F_1(x)_{[:,i]} \tag{29}$$

where $\circ$ denotes element-wise multiplication. The Guided Gram Matrix can then be defined as:

$$G_l^r(x) = F_l^r(x)^T F_l^r(x) \tag{30}$$

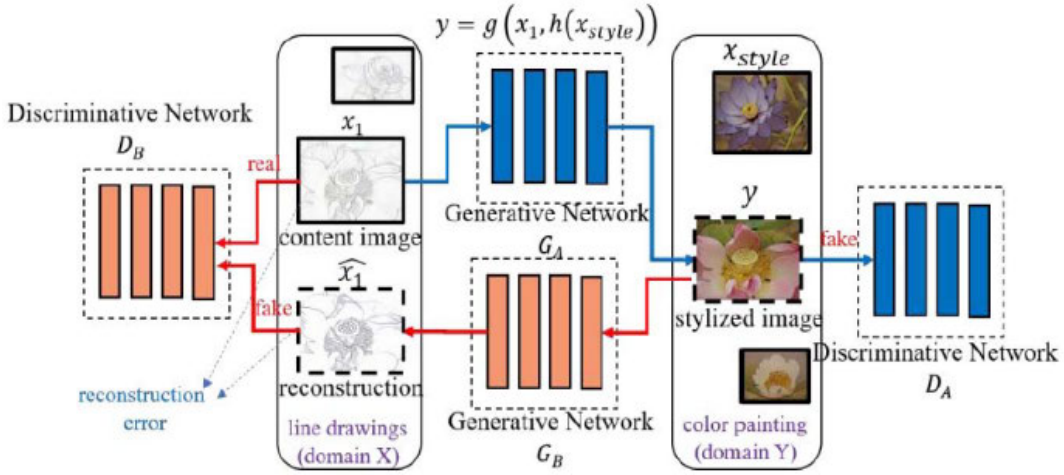

FIGURE 21. Model design of the complete network. (Zheng and Zhang [22]).

Furthermore, the contribution to the loss function is given as:

$$E_l = \frac{1}{4N_\ell^2} \sum_{r=1}^{R} \sum_{ij} \lambda_r \left(\mathbf{G}_\ell^r\left(\hat{\mathbf{x}}\right) - \mathbf{G}_\ell^r\left(\mathbf{x}_S\right)\right)_{ij}^2 \quad (31)$$

where N_l is the number of feature maps in layer "l," $\mathbf{G}_\ell^r\left(\hat{\mathbf{x}}\right)$ and $\mathbf{G}_\ell^r\left(\mathbf{x}_S\right)$ are the guided gram matrices generated as per Equations (29) and (30) for the generated image $\hat{\mathrm{x}}$ and the input style image x_S. λ_r is the weighting factor that controls the stylization strength in the corresponding region r.

An alternative approach focuses on stacking the guidance matrices with the feature maps directly. This is more efficient than the previous approach but comes at the cost of texture quality, as noted. The second factor addressed in [23] is Color control, independent of geometric shapes or textures. Color control is beneficial in situations where the model needs to contain the image's color is essential (E.g., Photo-realistic Style Transfers). [23] present two approaches to deal with this:

1. Luminance-only Transfer:Style Transfer is only performed on the luminance channel. This is done by extracting style and content Luminance channels and producing output luminance channels that are then combined with the original content colors to create the generated image.
2. Color Histogram Matching:In this method, the style image's colors are transformed such that their mean and covariance match with the content image's mean and covariance using a linear transform.

Each of them has its pros and cons. For instance, Luminance-only transfer preserves the content colors, but this comes at the expense of losing dependencies between luminance and colors. The color-matching might maintain this, but it depends on the transform, which can be rather tricky to find. Scale control allows us to pick separate styles at different scales. The image'sStyle is the spread of image texture in an arbitrary area [23] propose creating fresh pictures ofStyle from two separate photos combining a fine and a coarse-scale picture. This is handy when it comes to Style Transfer on high-resolution images. Given a high-resolution content and style image, the output is achieved by downsampling to the desired resolution. This output is upsampled and used as the initialization for original images. This technique requires fewer iterations for optimization and filters low-level noise as well. The method can be iterated to generate very high-resolution images.

As seen in Fig. 22, the method works well to get a high-resolution image like the one that does not use it. However, the "CTF" model requires fewer iterations and is seen to have less noise.

B. STABILITY IMPROVEMENTS IN NEURAL STYLE TRANSFER

The latest image style transfer methods can be grouped into two groups. The first one is the optimization approach that solves a particular optimization problem for the generated image. These results are outstanding but take some time to develop each picture. Second is Feed-forward approaches that provide solutions to these problems and are usable for real-time synthesis but tend to give unstable readings. [24] introduces a new method for stabilizing feed-forward style transfer methods for video stylization using a recurrent network trained using temporal consistency loss. In this method, the network tries to minimize the summation of three losses. The combined loss is defined as

$$\mathrm{L}\left(\mathrm{W}, \mathrm{c}_{1:\mathrm{T}}, \mathrm{s}\right) = \sum_{t=1}^{T} \left(\lambda_c L_C\left(p_t, c_t\right) + \lambda_s L_s\left(p_t, s\right) + \lambda_t L_t\left(p_{t-1}, p_t\right)\right) \quad (32)$$

Here λ_c, λ_s, and λ_t areused to assign importance to loss term.

The three losses are as follows:

1. Content style loss $\mathrm{L_c}$ which is defined as

$$\mathrm{L_c}\left(\mathrm{p}, \mathrm{c}\right) = \sum_{j \in C} \frac{1}{c_j H_j W_j} \left\|\phi_j(p) - \phi_j(c)\right\|_2^2 \quad (33)$$

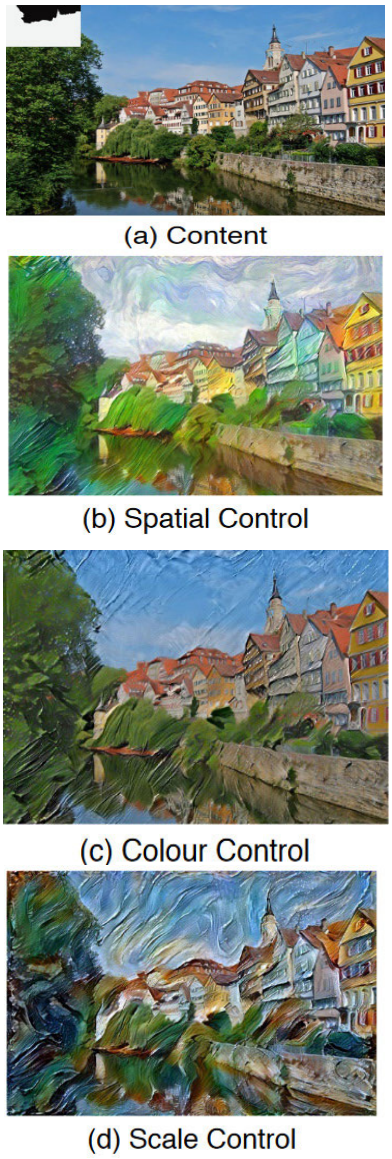

FIGURE 22. (a) The content image. (b) Spatial control that differentiates in sky and ground textures. (c) Color control that tries to preserve the original colors of the content image. (d) Two Styles are used on fine and coarse scale to stylize the image. (Gatys *et al.* [23]).

Here, $\phi_j(x)$ is the j-th layer activation network activation of the shape $C_j * H_j * W_j$ for image x.

2. Style reconstruction loss L_{sis} defined as

$$L_s(p, s) = \sum_{j \in s} \frac{1}{C_j \cdot H_J \cdot W_j} \left\| G(\phi_j(p)) - G(\phi_j(s)) \right\|_F^2 \tag{34}$$

Here, $G\left(\phi_j(x)\right)$ is a $C_j * C_i$ gram matrix for layer j activations

3. Temporal consistency loss L_t defined as

$$L_t(p_{t-1}, p_t) = \frac{1}{HW} ||m_t \Theta_{\rho_{t-1} - m_t \Theta \tilde{p}_t}||_F^2 \tag{35}$$

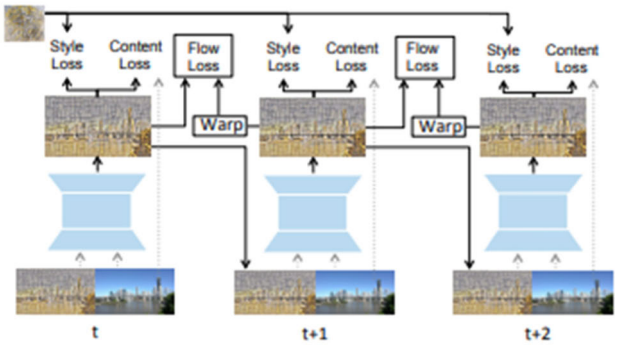

FIGURE 23. System overview. (Gupta *et al.* [24]).

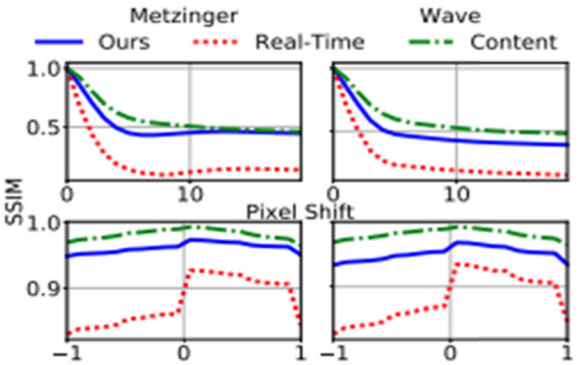

FIGURE 24. Image sharpness based on SSIN. (Gupta *et al.* [24]).

Here, $m(h, w) \in [0, 1]$ is 0 in the region of occlusion and motion boundaries, $\odot$ indicates element-wise multiplication, and H, W isthe height and width of the input frame. Style and content losses motivate high-level feature mapping of the content image with features in Style. Temporal consistency loss prevents drastic variations in the output between time steps. Content image and a previous frame are fed as input to the network. At each step, the output of the network is passed as input in the next step.

As Fig. 23 shows, it is a recurrent convolutional network where each style transfer network is a deep Conv. Network with two spatial downsampling blocks, followed by several residual blocks. The final layers are nearest-neighbor upsampling blocks.

Fig. 24 shows the results for translation and blurring distortions of images. An image patch is taken and distorted then SSIM is computed between both the original and distorted patch. Both are then stylized, and SSIM is calculated for the styled original and styled distorted patch. The proposed method is compared with the Real-Time baseline model on all styles. The results prove that this method is significantly more robust at controlling distortions.

Table 9 shows the results of the comparison done based on speed. This method matches the Real-Time baseline in terms of speed and is three times faster than the Optimbaseline [24].

Fig. 25 shows a pair-wise comparison of stylized frame output. PSNR/SSIM values are shown for each example pair.

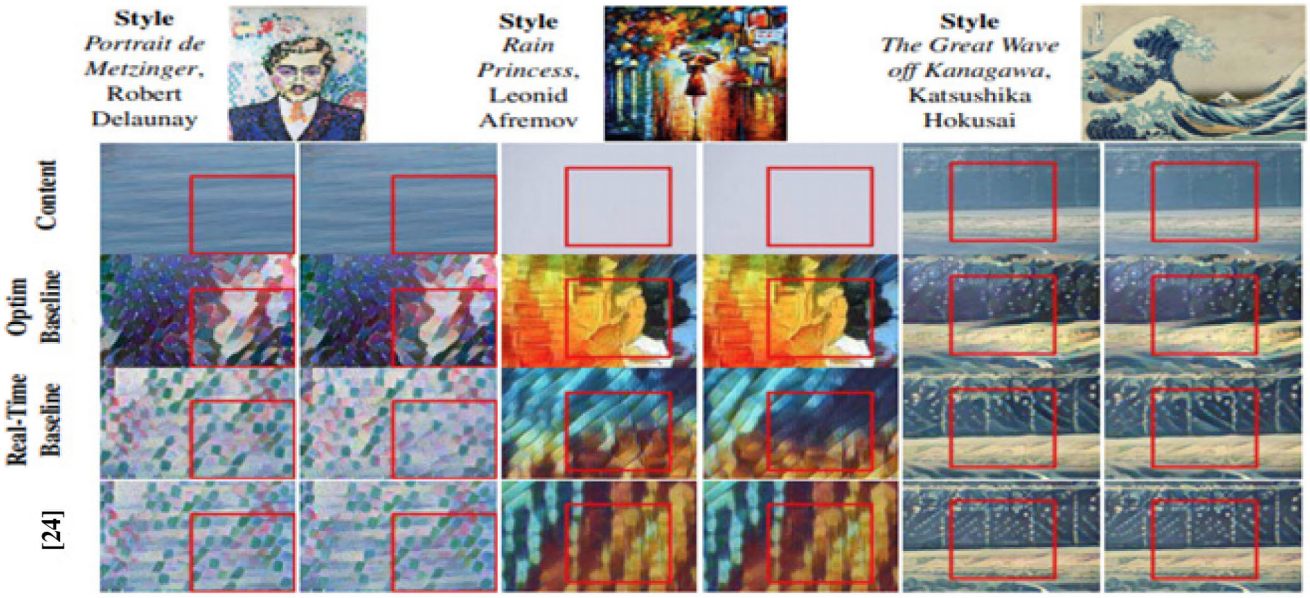

FIGURE 25. Pair-wise stylized image SSIN comparison. (Gupta *et al.* [24]).

TABLE 9. (Gupta *et al.* [24]).

Image-Size	Real-Time Baseline	Optim Baseline	Ours	Speedup
256 x 256	0.024	22.14	0.024	922x
512 x 512	0.044	59.64	0.044	1355x
1024 x 1024	0.141	199.6	0.141	1415x

FIGURE 26. On the output image (c), the undesirable style image color overlay is evident. (Gatys *et al.* [13]).

This method produces similar frames as OptimBaseline [24]. Still, on comparing with real-time baseline, the frames made are better and temporally consistent for unstable styles like Rain Princess and Metzinger.

There are two problems with this method:

1. Occasionally, as a result, one object can block others, which is undesirable.
2. Show-door artifacts appear in the generated image.

C. PRESERVING COLOR IN NEURAL ARTISTIC STYLE TRANSFER

Though there have been many papers on style transfer, there has been some shortcoming: the algorithms transfer the colors of the original painting to the output painting, which can alter the appearance in undesirable ways. [25] describes a simple linear method for retaining colors after style transfer, extending to the neural artistic style transfer algorithm. One of the problems seen, as said before, is that the yield after style transfer, however, replicates the Style of brushstrokes, mathematical shapes, and painterly structures displayed in the style picture. Nevertheless, it likewise duplicates the color distribution of the style picture undesirably.

Two different methods for preserving colors of the content image seen are color histogram and luminance only transfer.

1. Color histogram matching:
    1. Consider S- style image and C- input image. Style image's colors are transformed to coordinate the input image's colors, producing S'- a new style image that replaces S as an input to the NST algorithm. One choice that is to be made is the color transfer procedure.
    2. Each pixel is transformed as:

$$x_{S'} \leftarrow Ax_S + b \tag{36}$$

    where, A: 3 × 3 matrix B: 3D vector $\mathbf{x}_i = (R, G, B)^T$
    3. This transformation is chosen in such a way that the mean and covariance of the RGB value in the new image style (S') matches the content image (C), i.e., $\mu_{S'} = \mu_C$ and $\Sigma_{S'} = \Sigma_C$
    4. The values on A and b from equation (36) based on the condition mentioned about (C) are:

$$b = \mu_C - A\mu_S$$
$$A\Sigma_S A^T = \Sigma_C \tag{37}$$

    5. There are many different solutions for A which satisfies these constraints

1. The first variant is using the Cholesky decompositions:

$$A_{chol} = L_C L_S^{-1}$$

where, $\Sigma = LL^T$ is Cholesky decomposition of Σ.
2. Formulation of 3D color matching is the second variant, which is: $A_{IA} = \Sigma_C^{1/2} \Sigma_S^{-1/2}$
3. It is seen that transfer of color histogram before style transfer gives better outcomes, which is neural style move is figured from the first data sources S and C. Afterwards, the yield T is color-coordinated to C, creating another yield T′.
4. The algorithm also reduced competition between the reconstruction of the content image and the simultaneous matching of the texture details from the image style.
5. Luminance-only transfer:

- Visual perception is much more susceptible to changes in luminance than to color.
- Luminance channels Ls and Lc are initially derived from the style and content images., NST algorithm is applied to them and yield luminance image Lt.
- Using YIQ color space, I, and Q filters - the input picture's color information merged with Lt to generate the resulting image.
- The significant mismatch between the style luminosity histograms and the material images should be balanced before the Style is transferred.
  each style image's luminance pixel is updated:

$$L_{S'} = \frac{\sigma_C}{\sigma_S} (L_S - \mu_S) + \mu_C \tag{38}$$

where μ_S and μ_C is the mean luminance σ_S and σ_C is the standard deviation.

D. DEEP PHOTO STYLE TRANSFER

A profound learning way to deal with photographic style transfer manages an outsized kind of picture that reliably moves with the given Style. One of the commitments is to dispose of the works of art-like impacts by preventing spatial data losses and obliging the exchange activity in the shading area. Another critical commitment might be an answer for the test presented by the distinction in content between the given and reference pictures, ending in unwanted exchanges between random substances. The calculation utilized here takes two pictures: an input picture, commonly a stock photo, and an adapted and corrected reference picture, the reference style picture. The proposed approach might be a photorealism regularization term inside the target work during the improvement, compelling the reproduced picture to be spoken to by locally relative shading changes of the contribution to stop twists.

1) PHOTOREALISM REGULARIZATION

[26] describes how toregularize this optimization approach to maintain the structure of the original image and generate photo-realistic results. The idea is to express this limitation on

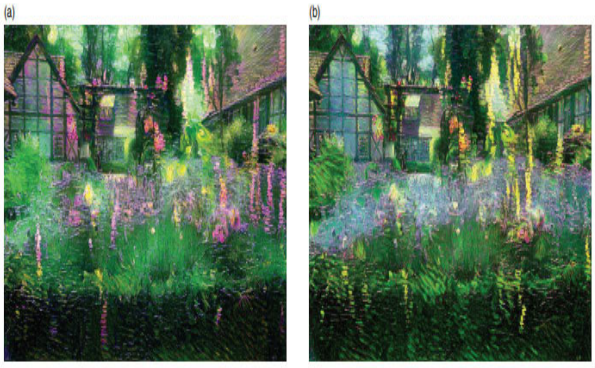

FIGURE 27. Result cholesky and image analogies color transfer. (Gatys *et al.* [13]).

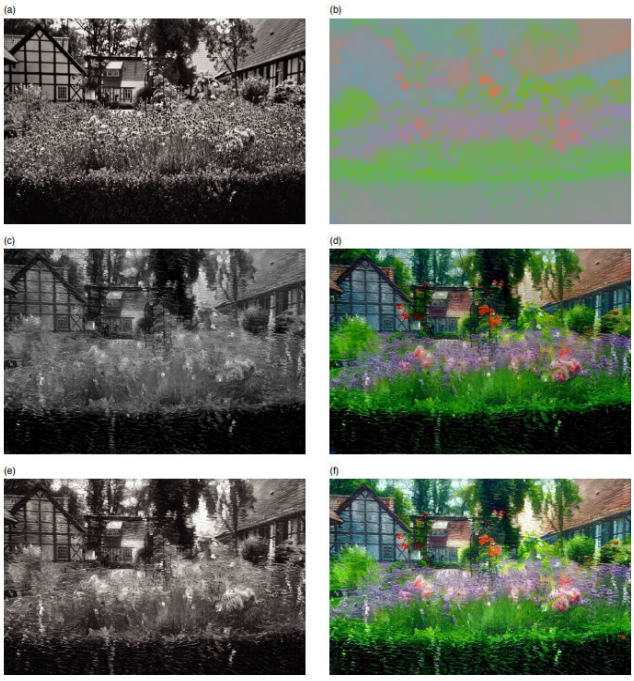

FIGURE 28. Working of luminance-based style transfer with color histogram.(Gatys *et al.* [13]).

the transformation performed to the input image rather than on the output image directly. The topic of characterizing the space of photo-realistic photos remains unresolved. [26] did not need to solve it; instead, utilized the fact that the input was already photo-realistic. The goal is to protect images from losing this attribute during the transfer by including a provision that penalizes image distortions. The answer is to find an image transform locally affine in color space, a function that translates the input RGB values onto their output counterparts for each output patch.

$$\mathcal{L}_{TOTAL} = \sum_{l=1}^{L} \alpha_l \mathcal{L}_C^l + \Gamma \sum_{l=1}^{L} \beta_l \mathcal{L}_{s+}^l + \lambda \mathcal{L}_m \tag{39}$$

L is the no. of convolutional layers, and l is the l^{th} Conv. layer of the network. Weight Γ controls style loss. Weights α_l and β_l are layer preference parameters. Weight λ is used

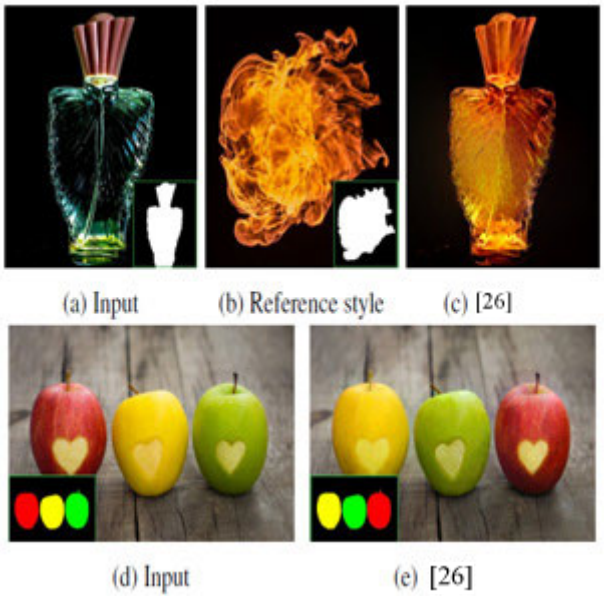

FIGURE 29. Manual division empowers assorted errands, for example, moving a fireball (b) to a scent bottle (a) to create a fire-enlightened look (c) or exchanging the surface between different apples (d, e). (Luan *et al.* [26]).

to control photorealism regularization. $\mathcal{L}_C^l$, $\mathcal{L}_{s+}^l$ and $\mathcal{L}_{s+}^l$ are content, augmented Style, and photorealism regularization, respectively. Fig. 29, Shows how clients can monitor the exchange outcomes by only offering semantic masks. This utilization case allows masterful applications and makes it possible to handle unusual cases for which semantic naming is not supported, e.g., direct fireball scent holders.

2) AUGMENTED STYLE LOSS WITH SEMANTIC SEGMENTATION

The style term is restricted by calculating the matrix on the whole picture. Because a Gram matrix defines its constituent vectors up to an isometry, it implicitly stores the precise distribution of brain responses, limiting its capacity to adjust to changes in semantic context and causing ''spillovers.'' The masks are added extra channels to the input picture and enhance the neural style method by concatenating the segmentation channels and updating the style loss. [26] also learned that the segmentation does not need to be pixel precise because the regularization finally restricts the output.

Fig. 30 shows instances of disappointment because of mismatching. These can be fixed utilizing manual segmentation.

E. GauGAN: SEMANTIC IMAGE SYNTHESIS WITH SPATIALLY ADAPTIVE NORMALIZATION

Conditional picture synthesis implies the task of creating photo-realistic pictures molding on some input data. [27] is about a particular restrictive picture blend changing over a semantic division veil to a photo-realistic picture. This structure has a broad scope of uses, for example, content generation and picture altering.

[27], which is worked by stacking convolutional, standardization, and nonlinearity layers, is the ideal situation,

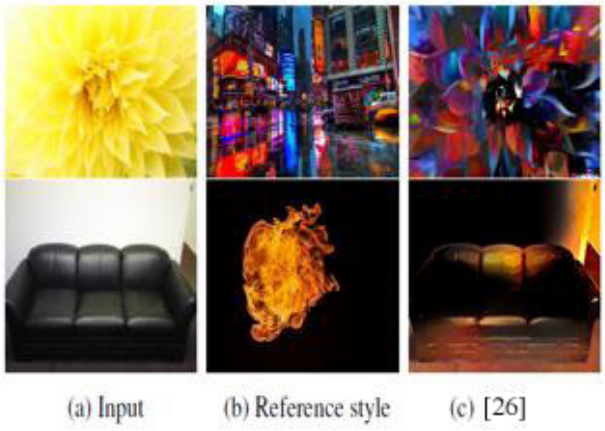

FIGURE 30. Failures are caused by mismatching. (Luan *et al.* [26]).

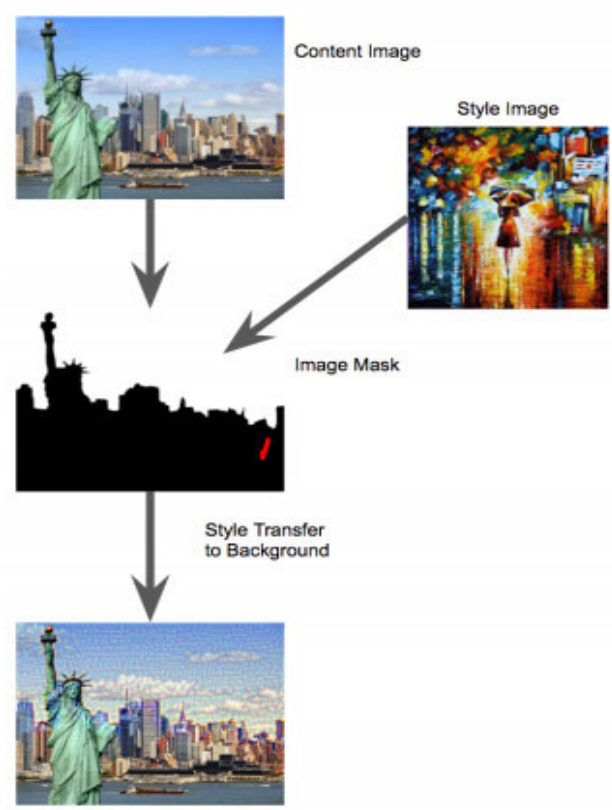

FIGURE 31. Flow of segmented Style transfer. (Makow *et al.* [28]).

defective because their normalization layers tend to ''wash away'' information in information semantic covers. To address the issue, the creator has proposed spatiallyadaptable standardization. This restrictive standardization layer directs the inceptions using semantic input formats through a spatially flexible, learned change and can reasonably multiply the semantic information all through the networks.

1) SPADE GENERATOR

There is no convincing motivation to deal with the division guide to the Generator's top layer with SPADE since the informed regulation boundaries have encoded enough

TABLE 10. Remarks on part 3.

Contribution	Ref. No.	Paper	Strength	Weakness
Spatial and Color Control	[23], [25], [28], [29]	Gatys et al. 2017, Gatys et al. 2016, Makow et al 2017, Dhir et al 2021	Spatial Control can reduce visual artifacts. Color Control contain the image's color.	Spatial Control requires computing spatially guided Gram Matrices which can be computationally expensive. Color Control comes at the expense of losing the dependencies like luminance and colors.
Photorealism Regularization	[26]	Luan et al. 2017	To eliminate the painting-like effects, prevent spatial distortion and limit the transfer operation to color space alone. A solution to the problem with content differences between input and reference pictures, which might result in unwanted transfers between unrelated contents.	There may be some scene elements more (or less) represented in the input than in the reference image.
Spatially Adaptive Normalization	[27]	Park et al. 2019	Spatially Adaptive Normalization can better preserve semantic information against common normalization layers.	Removing any loss function for the improvement, results in degradation of the generated pictures.
Temporal Consistency	[24]	Gupta et al. 2017	The network tries to minimize the summation of Control style loss, Style reconstruction loss and Temporal Consistency Loss, which results in more robust at controlling distortions.	Occasionally in results, one object can block others, which is undesirable. Show-door artifacts appear in the generated image.

information about the imprint design. This way, discard the Generator's encoder, which is consistently used in late plans that smooth out achieves a more lightweight network. Equivalently to existing class-contingent generators, the new Generator can acknowledge a subjective vector as info, engaging a fundamental and standard way for the multi-modular blend. Curiously, the division shroud in the SPADE Generator is dealt with through spatially flexible equilibrium without standardization. Only networks from the past layer are standardized. From this time forward, the SPADE generator can all the more probable protect semantic information. It acknowledges the benefit of standardization without losing semantic information.

2) MULTI-MODAL SYNTHESIS

Using a self-assertive vector as the Generator's contribution, the design gives an essential technique to the multi-modular union. To be explicit, one can add an encoder that quantifies a picture into an irregular vector, which the Generator then deals with. The encoder and generator structure a variationalautoencoder, in which the encoder endeavors to get the Style of the image. In contrast, the Generator solidifies the encoded Style and the division veil information by methods for SPADE to change the primary picture. Moreover, the encoder fills in as a style direction network at test time to get the Style of target pictures.

In the first place, [27] considers two kinds of tasksto the Generator: self-assertive commotion or down inspected division maps. Second, fluctuating the sort of limit-free standardization layers before applying the tweak limits. Next, move the convolutional piece size following up on the name guide, and find that part size of 1×1 harms execution, likely because it blocks utilizing the name's setting. Ultimately, adjusting the restriction of the generator network by changing the number of convolutional channels.

F. EXPLORING STYLE TRANSFER

In recent times NST algorithms have improved significantly on tasks such as image segmentation, replicating the content image into different images using styles. In [28], several new extensions and improvements to the original neural style transfer were seen, such as altering the original loss function to achieve multiple style transfers while preserving the color and semantically segmented style transfer. Gaty's approach includes a pre-trained feed-forward network that performs a forward pass ''image transformation'' on the input image before inputting it to style transformations, which can be done on real-time video applications.

Method:

- The baseline taken was fast neural style transfer, consisting of two components: picture transformation network Fw and loss function φ.
- The overall combined loss function is the final objective is given as:

$$W^* = \arg\min_W E_{x,\{y_i\}} \left[\sum_{i=1} \lambda_i l_i \left(f_W \left(x \right), y_i \right) \right] \quad (40)$$

where W- weights X- image to be transformed Y_i- Style image

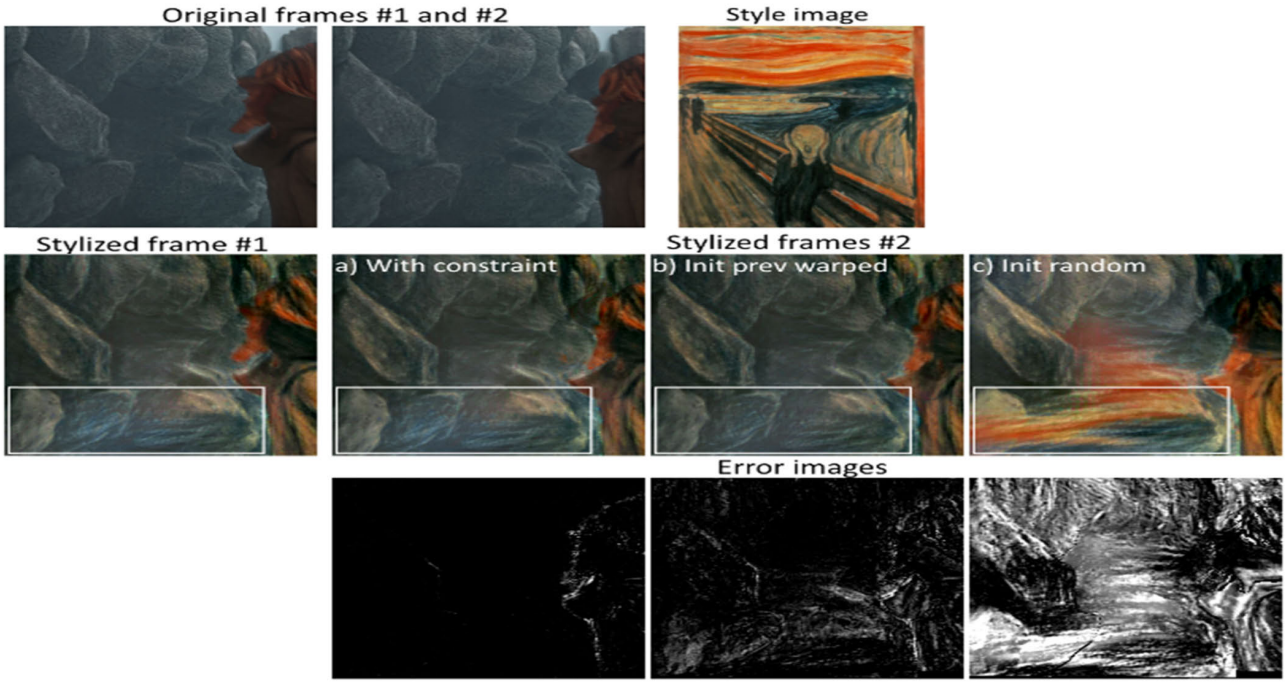

FIGURE 32. A scene from the test Sintel dataset, the style image used, and the outputs obtained from various methods. The highlighted regions are the ones with prominent differences. The error images show the temporal inconsistence which is prominent in the third approach.(Ruder *et al.* [31]).

- Image Transformation Network: Color images used are $3 \times 256 \times 256$ in shape.
  - Downscaling: done by convolutional layer with stride 2
  - Upscaling: done by convolutional layer with stride 1/2.
  - This method provides computational benefits of operating lower-dimensional spaces.
- Perceptual Losses:
  - Feature Reconstruction loss: pixels of the output image $\hat{y}$ have feature presentations similar to the loss network φ computes.

$$I_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \left\| \phi_j(\hat{y}) - \phi_j(y) \right\|_2^2 \quad (41)$$

  - Style Reconstruction loss: penalizes style differences such as colors and textures
    - Firstly, the Gram matrix is defined:

$$G_j^{\phi}(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'} \quad (42)$$

    - Style loss is the squared normalization of the Frobenius standard for the difference between gram output matrices of the generated and actual target image.:

$$l_{\text{style}}^{\phi,j}(\hat{y}, y) = \left\| G_j^{\phi}(\hat{y}) - G_j^{\phi}(y) \right\|_F^2 \quad (43)$$

    - It minimizes the style reconstruction loss results in generating an image that preserves stylistic features over not spatial characteristics of the target.
- Simple Loss function:
  - Pixel loss: the normalized distance between the output $\hat{y}$ and target

$$l_{\text{pixel}} = \frac{1}{CHW}(\hat{y}, y) = \| \hat{y} - y \|_2^2 \quad (44)$$

  - Total Variation Regularization: this is used for maintaining spatial smoothness.
  - Multiple Style Transfer: An extension of vanilla neural style transfer allows multiple style images to be transferred to a single content image.
  - Requires a smile modification to the style loss function:

$$l_{\text{multi}} = \sum_{i=1}^{n} w_i l_{\text{style}}^{\phi,J_i}(\hat{y}, y_i) \quad (45)$$

  - This allows the flexible choice of the style layers and weights independently for each style image.
  - Allows us to generate images that blend the styles of multiple images readily.
  - Trained on Adam optimizer
  - When forced to blend multiple styles, it leads to a more extensive style loss than a single style image.
  - Color Preserving Style Transfer: used the luminous only transfer, which works very well and takes a simple transformation after a typical style transfer algorithm.

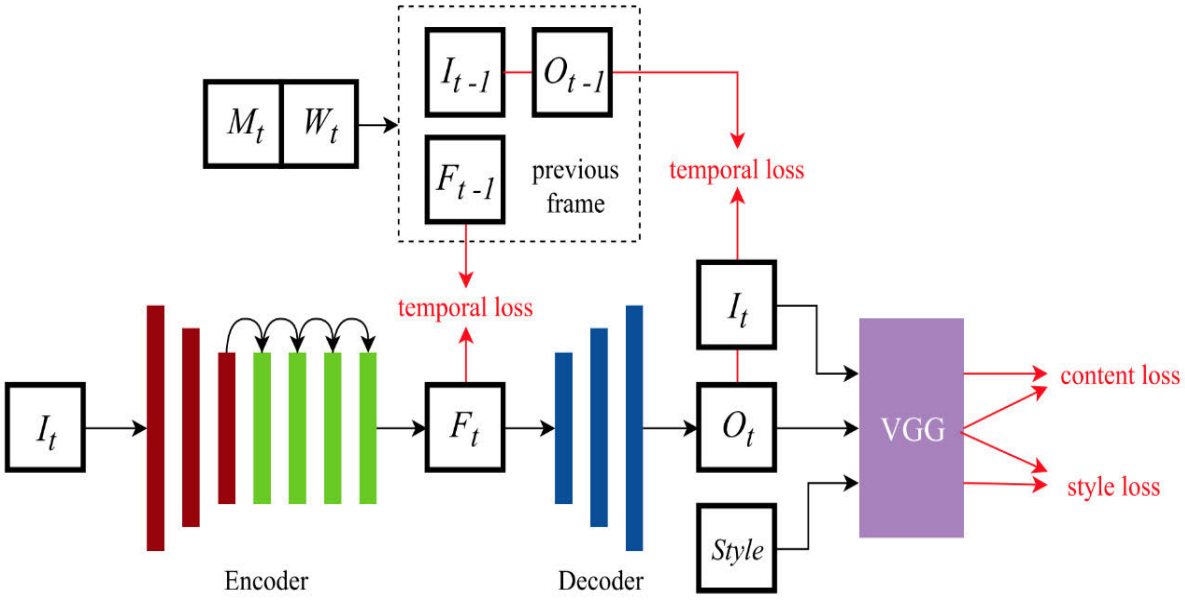

FIGURE 33. The ReCoNet architecture [33] for Video neural Style Transfer (presented by Gao *et al.*). It, Ft and Ot are input image, encoded feature map, and generated image at time "t". A Frame is made up of these three objects. The previous frame is compared with the current frame to compute "temporal loss" which results in better dependencies between two consecutive frames. (Ruder *et al.* [31]).

- Semantically Segmented Style transfer: clustering parts of input images together that belong to the same object class. It first generates a mask for the input of shape H x W for each pixel location to apply gradient descent and where to not. All the above extensions make it possible to be an effect to achieve real-time video processing applications.

G. AUTOMATIC IMAGE COLORIZATION USING GANs

[29] talks about how GANs can automate the image's colorization process without changing the picture's configuration. [29] have used Conditional GAN to achieve the result. The architecture approach used here was on fully connected networks; [29] used layers of convolutions in Generator. [29] have also used the technique similar to expanding encoder networks and compressing decoder networks to reduce the memory's dependency on training. The Generator takes in the greyscale image and then downsample it. It is compressed after it goes through here, and these operations are repeated four times resulting in a matrix. In the expansion phase, it gets upsampled. Batch normalization and the use of leaky ReLU help in better training and performance of GAN. The Discriminator starts with greyscale images and the predicted image to form the color image. The unique activation functions used to stabilize the last layers of Generator and Discriminator are the tanh activation function and Sigmoid activation function. Another unique method used here is Adam's Optimizer for learning rate and Strided convolutions, resulting in upgrading the training performance depending on the invariances' convolution layers. Convergence failure was experienced on various occasions, settling by changing optimizers, expanding learning rates, changing kernel rates, and presenting batch normalization.

1) OBSERVATIONS

This part focuses mainly on achieving better quality images from GANs by improving color accuracy. [23], [25] and [28] parallelly propose Spatial and Color Control, which allows the use of multiple styles and preserves content image color for generating more photo-realistic images. By constraining transform and adding a custom energy term, [26] provides a versatile model that handles various input images. [27] introduces "spatially adaptive normalization" that assists in synthesizing photo-realistic images. A key feature provided by [23] is Scale Control, which allows us to mix coarse and fine attributes of two different styles. This method helps with training on high-resolution images and is highly scalable in that regard. [24] is solely focused on video Neural Style Transfer and introduces temporal consistency in-between frames to allow dependency between adjacent frames.

V. APPLICATION-BASED PAPERS

This section looks at the approaches, challenges, and limitations in Neural Style Transfer for Videos on Mobile phones. A few architectures are proposed based on their performance on mobile devices.

A. ARTISTIC STYLE TRANSFER FOR VIDEOS

[31] presents the application of image style transfer to a complete video. A few additions are made regarding initializations and loss functions to suit the video input allowing stable stylized videos even with a high degree of motion. In addition, it processes each frame individually and adds temporal constraints that penalize deviation among point trajectories. [31] also, propose two more extensions:

Long-term motion estimates allow consistency over a more considerable period in regions with occlusion.

TABLE 11. Different methods tested on multiple sequences with their temporal consistency errors. (Ruder *et al.* [31]).

Methods	alley_2	ambush_5	ambush_6	bandage_2	market_6
DeepFlow	**0.00061**	**0.0062**	**0.012**	0.00084	0.0035
EpicFlow	0.00073	0.0068	0.014	**0.00080**	**0.0032**
Initprev warped	0.0016	0.0063	**0.012**	0.0015	0.0049
Initprev	0.010	0.018	0.028	0.0041	0.014
Init random	0.019	0.027	0.037	0.018	0.023

A multi-pass algorithm is used to reduce the artifacts at the image boundaries. The algorithm considers forward and backward optical flow resulting in a better-quality video. [31] propose using the previous frame to initialize the optimizer for the current frame. This allows similar parts of the frame to be rendered, whereas the changed parts are rebuilt. However, the technique has flaws when used on videos as moving objects are not initialized properly. To address this, [25] consider the optical flow by warping the previous output:

$$x'_{(i+1)} = \omega_i^{i+1} x^{(i)} \tag{46}$$

where $\omega_1^{(i+1)}$ warps the input stylized frame $x^{(i)}$ using the optical flow information derived from content frames $g^{(i)}$ and $g^{(i+1)}$. [31] use DeepFlow and EpicFlow optical flow estimation algorithms to do so. The next addition is the use of temporal consistency losses to penalize adjacent frame inconsistencies. To do so, they detect the disoccluded regions by comparing the forward and backward flows. The temporal loss then penalizes deviation between the generated

Image and the compatible optical flow parts of the warped image. This is done with the help of a feature map "a" that specifies per-pixel weightage depending on disocclusion and motion boundaries.

$$\mathcal{L}_{\text{temporal}}(\boldsymbol{x}, \boldsymbol{\omega}, \boldsymbol{a}) = \frac{1}{D} \sum_{k=1}^{D} a_k \cdot (x_k - \omega_k)^2 \tag{47}$$

Thus, the short-term loss function is given as:

$$\begin{aligned} &\mathcal{L}_{\text{shortterm}}\left(\boldsymbol{g}^{(i)}, \boldsymbol{c}, \boldsymbol{x}^{(i)}\right) \\ &\quad = \alpha \mathcal{L}_{\text{content}}\left(\boldsymbol{g}^{(i)}, \boldsymbol{x}^{(i)}\right) + \beta \mathcal{L}_{\text{style}}\left(\boldsymbol{c}, \boldsymbol{x}^{(i)}\right) \\ &\qquad + \gamma \mathcal{L}_{\text{temporal}}\left(\boldsymbol{x}^{(i)}, \omega_{i-1}^{i}\left(\boldsymbol{x}^{(i-1)}\right), \boldsymbol{a}^{(i-1,i)}\right) \end{aligned} \tag{48}$$

This is further extended to achieve longer-term consistency by incorporating the data for multiple previous frames rather than just one frame:

$$\begin{aligned} &\mathcal{L}_{\text{longterm}}\left(\boldsymbol{g}^{(i)}, \boldsymbol{c}, \boldsymbol{x}^{(i)}\right) \\ &\quad = \alpha \mathcal{L}_{\text{content}}\left(\boldsymbol{g}^{(i)}, \boldsymbol{x}^{(i)}\right) + \beta \mathcal{L}_{\text{style}}\left(\boldsymbol{c}, \boldsymbol{x}^{(i)}\right) \\ &\qquad + \gamma \sum\nolimits_{j \in J: i-j \geq 1} \mathcal{L}_{\text{temporal}}\left(\boldsymbol{x}^{(i)}, \omega_{i-j}^{i}\left(\boldsymbol{x}^{(i-j)}\right), \boldsymbol{a}_{\text{long}}^{(i-j,i)}\right) \end{aligned} \tag{49}$$

The weights $\boldsymbol{a}_{long}^{(i-j,i)}$ are computed as follows:

$$\boldsymbol{a}_{long}^{(i-j,i)} = max\left(\boldsymbol{a}^{(i-j,i)} - \sum\nolimits_{k \in J: i-k > i-j} \boldsymbol{a}^{(i-k,i)}, \boldsymbol{0}\right) \tag{50}$$

This means investigating past frames till consistent correspondence is obtained. The advantage of this is that each pixel is associated with the nearest frame, and as the optical flow computed over temporally closer images has a lesser error. Thus, it results in better videos. [31] handle the problem of strong motion using a multi-pass algorithm. The video is processed bi-directionally in multiple passes. By alternating the direction of optical flow, firmer consistency is achieved. Initially, every input is processed independently based on random initializations. The inputs are then mixed with the warped non-disoccluded parts of previous frames on which the optimization algorithm is run for some iterations. Next, the forward and backward passes are alternated. The frame initializations for forwarding and backward passes are given as:

$$\boldsymbol{x}^{(i)(j)} = \begin{cases} \boldsymbol{x}^{(i)(j-1)} & \text{if } i = 1 \\ \delta \boldsymbol{a}^{(i-1,i)} \circ \omega_{i-1}^{i}\left(\boldsymbol{x}^{(i-1)(j)}\right) & \\ + \left(\bar{\delta}\boldsymbol{1} + \delta \mathrm{a}^{(i-1,i)}\right) \circ \boldsymbol{x}^{(i)(j-1)} & \text{else.} \end{cases} \tag{51}$$

$$\boldsymbol{x}'^{(i)(j)} = \begin{cases} \boldsymbol{x}^{(i)(j-1)} & \text{if } i = N_{\text{frames}} \\ \delta \boldsymbol{a}^{(i+1,i)} \circ \omega_{i+1}^{i}\left(\boldsymbol{x}^{(i+1)(j)}\right) & \\ + \left(\bar{\delta}\boldsymbol{1} + \delta \mathrm{a}^{(i+1,i)}\right) \circ \boldsymbol{x}^{(i)(j-1)} & \text{else} \end{cases} \tag{52}$$

The optical flow computation implementation takes roughly 3 minutes per frame at a resolution of 1024×436, which is done with the help of parallel flow computation on the CPU. At the same time, style transfer occurs on the GPU. The short-term consistency results on the Sintel datasets are presented in Table 11, where multiple approaches' errors are compared across different videos.

The long-term consistency results are more qualitative. They are thus presented in the form of supplementary videos.

B. REAL-TIME NEURAL STYLE TRANSFER FOR VIDEO

The work looks at the possibility of making video-style transfers using a feed-forward network. Differentiated and direct

applying a current picture style move procedure to accounts, the proposed method uses the readied association to yield fleetingly consistent adjusted weights, which are significant. In distinction to the previous video style move methods, which rely upon progression on the fly, the technique referenced disagreement ongoing while at the same time creating severe visual outcomes.

The adapting network acknowledges one edge as information and produces its adjusted yield. The loss network, preprepared on the ImageNet order task, first focuses on the features of the revised yield outlines and registers the losses used to set up the adapting network. During the arrangement cycle, the adapting network and loss network are connected. The loss network's spatial loss is utilized to establish the adjustable network. With satisfactory setting up, the adapting network, tolerating one single casing as information, has encoded the worldly cognizance picked up from a video dataset and would in this manner have the option to make transiently unsurprising adjusted video outlines.

1) STYLIZING NETWORK

The adapting network speaks to changing a singular video edge to an adapted one. After three convolutional blocks, the component map's objective is diminished to a fourth of the information. By then, five lingering blocks are in this manner followed, provoking a brisk blend. Stood out from the current feed-forward association for picture style move, a tremendous favorable position of the network is used for fewer channels to reduce the model size, which winds up gathering a distinct loss in the stylization quality.

2) LOSS NETWORK

The sturdy and essential elements of the primary model, the adapted edge, and the style image for establishing the network adapter should be segregated for space and global loss calculations. VGG-19 is employed in this article as the loss network showing acceptable image content and style images. Two kinds of losses can be found in the model: Spatial Loss and Temporal Loss.

C. REAL-TIME VIDEO-NEURAL STYLE TRANSFER ON MOBILE DEVICES

[33] presents a solution to two problems of video style transfer:

1. The difficulty of usage by non-experts.
2. Hardware Limitations

They present an app that can perform neural style transfer to videos at over 25FPS. They also discuss performance concerning iOS-based devices where they test an iPhone 6s and iPhone 11 Pro. Limitations for Android devices are also discussed. The solution includes:

1. A real-time application of NST on mobile devices
2. Existing solutions to temporal coherence.

The traditional approach of applying a convolution-based image generator per frame causes ''temporal inconsistency'' or unrelated frames causing flicker artifacts. [19] tries to solve this problem; however, their model has time-consuming computations.

[33] use Gao *et al.*'s lightweight forward feed network. There are white bubbles seen in the images. However, these are caused due to instance normalization and can be removed using filter response normalization. However, no implementations exist for mobile devices. Other issues include faded colors. The model is trained in two stages,

First on Style and content losses and then on a regularization term.

$$L(t) = \gamma L_{content} + \rho L_{style} + \tau L_{tv} \tag{53}$$

Second on achieving temporal consistency

$$L(t-1,t) = \sum_{i \in t-1,t} \left(\gamma \mathcal{L}_{\text{content}}(i) + \rho \mathcal{L}_{\text{style}}(i) + \tau \mathcal{L}_{tv}(i)\right) + \lambda_f \mathcal{L}_{\text{temp},f}(t-1,t) + \lambda_o \mathcal{L}_{\text{temp},o}(t-1,t) \tag{54}$$

$L_{temp,f,}$ and $L_{temp,o}$ are features and output-based temporal losses presented in Gao's paper.

The main idea is to use the optical flux between adjacent frames. The models do not need this information, effectively making it faster since dense optical flow estimation is computationally expensive. On the other hand, introducing Temporal Coherence weakens the style transfer. Speaking of android vs. iPhone implementations, Apple had better support since 2018's A12 chip and CoreML library, allowing the use of dedicated NPUs effectively. However, conversions between PyTorch to TensorFlow result in additional layers causing a 30-40% FPS drop.

Furthermore, many Libraries are yet to provide full mobile GPU operation support. Thus, due to the lack of standardization, Android implementations are rare. [32] also, compare two iPhones (6s and 11Pro) with different model sizes, resolutions and chart their FPS:

Fig. 35 shows that the model mentioned above can output around 13 FPS at 480p on an iPhone 11Pro with half a million parameters. This indicates that Video *NST* on mobile devices still needs many improvements. Then the coarse-to-fine stylization presented in [24] can probably be applied to increase the resolution of the generated images.

D. MULTI-STYLE GENERATIVE NETWORK FOR REAL-TIME TRANSFER

[34] finds it challenging, with dimensionally integrated modeling, to obtain comprehensive styles in this study. A novel MSG-NET technique is presented, allowing brush dimension control in real-time. [34] believes that detailed form modeling with dimensional style integration in [34] is difficult to achieve. The method shown is a modern MSG-NET approach that achieves real-time control of the size of the brush. The image's resizing style adjusts the brush's relative size based on the changing input images. A more acceptable representation of image style requires a 2D method.

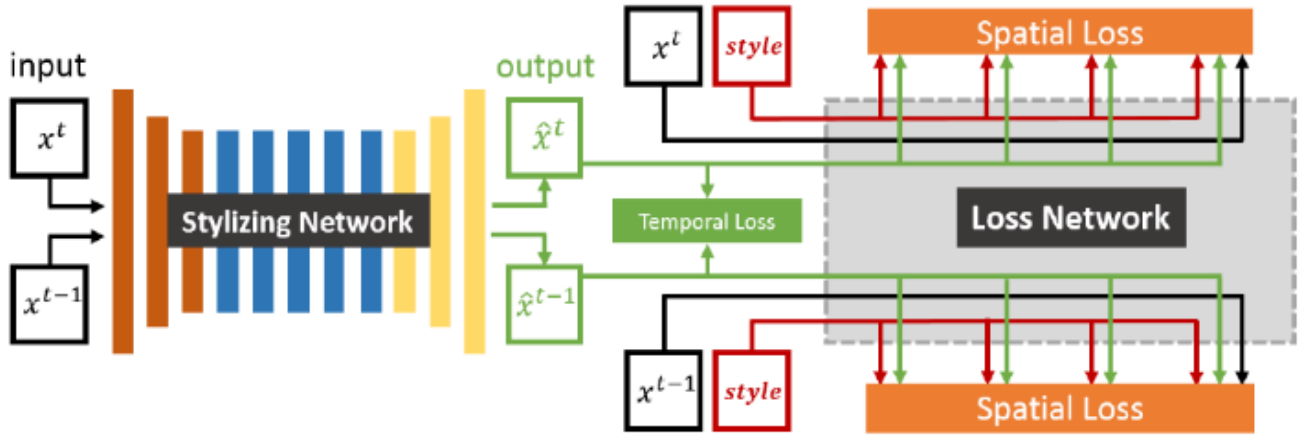

FIGURE 34. A chart of the proposed model. It includes two segments: an adapting and a loss network. Dark, green, and red square shapes address an info outline, a yield outline, and a given style picture independently. (Huang *et al.* [32]).

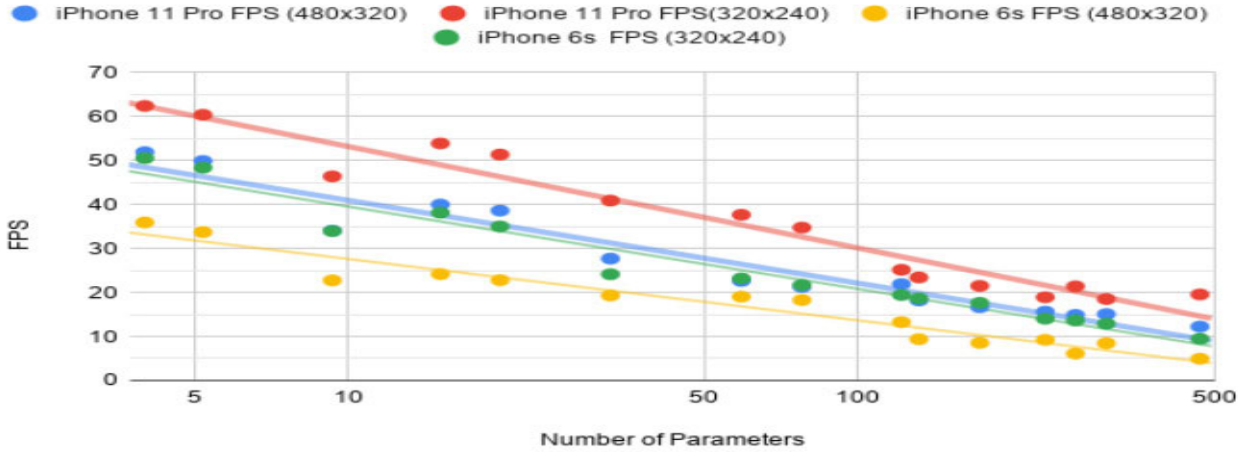

FIGURE 35. Performance achieved per configuration in terms of Frames Per Second (FPS) v/s Number of parameters in the model charted for two mobile devices at two resolution levels. (Dudzik et. al. 2020).

The model is based on the following works:

- Relation to Pyramid Matching: Early method was developed using texture synthesis using multi-style image pyramids. White noise image manipulation could lead to realistic image synthesis, so that fayre statics were inspired.
  - This method uses a similar feed-forward network, but it takes advantage of the benefits of deep learning networks without putting computational costs into the training process.
- Relation to Fusion Layer: The computed Comatch Layer uses both content and Style as input, hence a separate style from content.
- Content and Style Representation:
  - The image texture or Style can be represented as the distribution of the features by use of Gram Matrix

$$\mathcal{G}\left(\mathcal{F}^i(x)\right) = \sum_{h=1}^{H_i}\sum_{w=1}^{W_i} \mathcal{F}^i_{h,w}(x)\mathcal{F}^i_{h,w}(x)^T \quad (55)$$

  - The Gram Matrix is ordered less and describes the feature distributions
- CoMatchLayer: Explicitly matches statistics of second-order features based on the Style given.
  - $\hat{\mathcal{Y}}^i$ is a solution that holds the semantic information of the content image and matches the texture from the style image:

$$\hat{\mathcal{Y}}^i \& = \underset{\mathcal{Y}^i}{\operatorname{argmin}} \left\{ \left\| \hat{\mathcal{Y}} - \mathcal{F}(x_c) \right\|_F^2 + \alpha \left\| \mathcal{G}(\mathcal{Y}^i) - \mathcal{G}\left(\mathcal{F}^i(x_s)\right) \right\|_F^2 \right\} \quad (56)$$

  - To equalize the contribution target's Style and content, the α parameter is used. α is a parameter that allows a change of weightage for style loss.
  - An iterative technique allows the difficulty mentioned above to be minimized. However, in real-time, it is not practicable to achieve or distinguish the model.

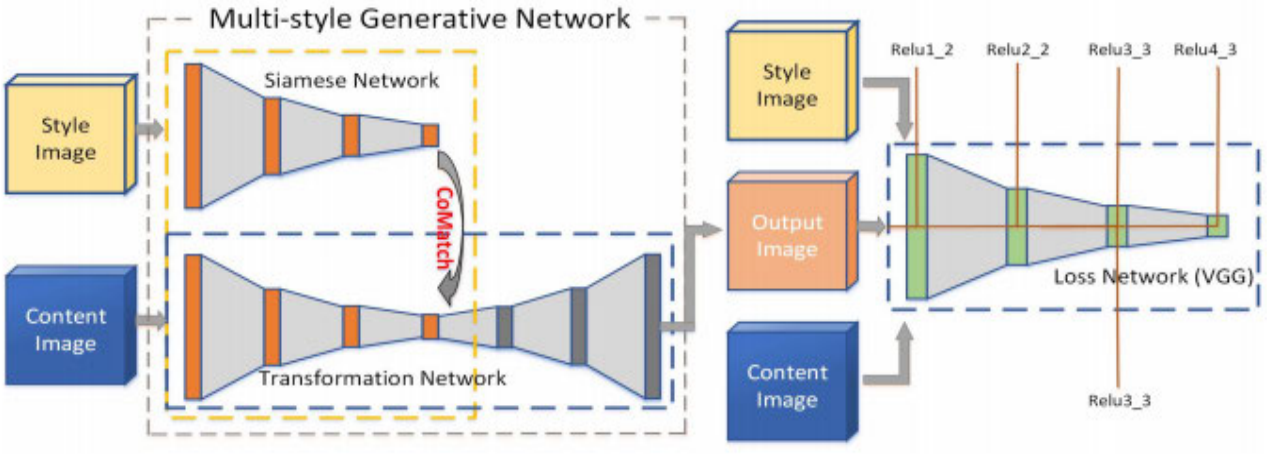

FIGURE 36. An overview of MSG-Net. (Zhang *et al.* [34]).

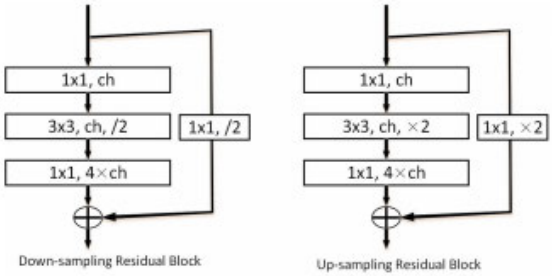

FIGURE 37. Extended architecture. (Zhang *et al.* [34]).

FIGURE 38. Spatial control result. (Zhang *et al.* [34]).

- Target style features map is tuned using the following approximation:

$$\hat{\mathcal{Y}}^i = \Phi^{-1}\left[\Phi\left(\mathcal{F}^i\left(x_c\right)\right)^T W \mathcal{G}\left(\mathcal{F}^i\left(x_s\right)\right)\right]^T \quad (57)$$

- The layer can be differentiated and introduced into the current Generative network and can learn directly without supervision from the loss function.

- Multi-style Generative Network (MSG-Net): This method introduces matching the feature statistics explicitly during runtime.
  - Siamese network and encoder of transformation network share their weights, which picks up the static features from the Style and gives out Gram Matrices.
  - Matches the features of style image at multiple levels with the content image using CoMatch.
  - Upsampled convolution: upsampling with convolution layer of stride 2. Compared to stride Convolution fractionally, the calculation complexity and parameters are precisely four times for this approach. This way, the network does not sample objects.
    Upsampled Residual block: Original architecture is extended with an upsampling version of fractionally strided convolution as soon in image Fig. 37.
  - Brush Stroke Size Control: The network was conditioned to learn different brush stroke sizes with different picture type sizes. Users can choose the brush stroke size after training.
  - The employment of weighted layers of ReLU and the normalizing process improves the picture quality created and resists the adaptation of picture contrast.
  - Minimizing the Loss by:

$$\hat{W}_{\mathcal{G}} = \underset{W_c}{\operatorname{argmin}} E_{x_e,x_s}\left\{\lambda_c \left\|\mathcal{F}^c\left(G(x_c,x_s)\right) - \mathcal{F}^c(x_c)\right\|_F^2 + \lambda_s \sum_{i=1}^{K}\left\|\mathcal{G}\left(\mathcal{F}^i(G(x_c,x_s))\right) - \mathcal{G}\left(\mathcal{F}^i(x_s)\right)\right\|_F^2 + \lambda_{TV}\ell_{TV}\left(G\left(x_c,x_s\right)\right)\right\} \quad (58)$$

The speed and size of models are crucial for mobile apps and cloud services. These are shown in Table 12.

- MSG-Net is shown faster due to an endless encoder in place of a pretrained VGG Network.
- Model Scalability: It is noted that there is no loss in quality as the number of styles rises on a real-time basis.
- Fig. 38 shows the spatial control using this model.

1) OBSERVATIONS

[19] Moreover, [32] extends on [24] and adds Optical Flow estimation based on multiple frames to improve temporal

TABLE 12. Comparison between different model's architecture based on model-size and speed. (Zhang *et al.* [34]).

	Model-size	Speed (256)	Speed (512)
Gatys et al.	N/A	0.07	0.02
Johnson et al.	6.7MB	91.7	26.3
Dumoulin et al.	6.8MB	88.3	24.7
Chen et al.	574MB	5.84	0.31
Huang et al.	28.1MB	37.0	10.2
MSG-Net-100	9.6MB	92.7	29.2
MSG-Net-1k	40.3MB	47.2	14.3

consistency. [32] introduces a CoMatch Layer that maps second-order feature statistics with target styles. [33] focuses on implementation on mobile devices and compares the performance of video style transfer [33] models of varying size and input images for two devices. It is observed that achieving reasonable frame rates with high resolutions is difficult, given the lack of GPU usage on mobile devices.

VI. NST EVALUATION METRICS

Evaluation metrics for NST could be challenging because of the variety in GANs models. However, accuracy, Fréchet Inception Distance (FID), Intersection-over-Union (IoU), time, perceptual path lengths, and warping error are the most often utilized metrics for the models constructed in the publications evaluated [36].

- The accuracy was used to measure the relative depth of the predicted images. It was also used to predict feature maps, where the higher the accuracy, the more accurate the feature maps indicated.
- The Fréchet Inception Distance (FID) approximates the real and fake feature distributions with two Gaussian distributions. They then compute the Fréchet distance (Wasserstein-2 distance) between two Gaussian distributions and use the findings to determine the model's quality.
- Few papers use the Intersection-over-Union (IoU) metric to determine the accuracy of segmentation and detection in object classification and localization.
- The perceptual path length quantifies the difference between consecutive images (VGG16 embeddings). It determines if the image changes along the shortest perceptual path in the latent space where fake images are introduced.
- The warping error is the difference between the warped and real subsequent frames. The warping error value is a good metric for determining the smoothness of video since it is an efficient technique to monitor video stability with many frames.

VII. POSSIBLE FUTURE APPLICATIONS OF NST

Apart from various exciting image transformation use cases, NST can be extended in a few more application areas such as:

- Movies: NST can change the scenes captured in movies using representational objects instead of green screens and tedious editing [37], [38].
- Online Education: Using different style banks, the same model can be used for other applications, such as creating animated versions of real-life stories in Education.
- Gaming: It can also be used in Mixed Reality (MR) games wherein the real world seen from the MR headset will change based on the style used for the game [38], [39].
- Fashion industry: NST can find applications in the fashion industry where designers and consumers can use it to overlay items while designing or trying them [40], [41].

Approaches like [42] provide a good starting point for real-time video style transfer and can be improved to work on mobiles efficiently. Observed with user privacy being in the headlines every day, Federated learning can also provide safer, more private data access by localizing training to specific devices. Some recent approaches include [43], [43]. Studies like [36] compare the evaluation metrics commonly used. Having more architectures that train on unpaired data is another interesting sub-domain to venture into. A good approach that performs style transfer on unpaired data is [44]. Although [44] works for high-resolution unpaired images and not videos, it can be considered a good entry point for high-resolution video style transfer. [45] uses Vision Transformers for image style transfer. There can be many more such fascinating use cases for NST shortly based on the user requirements.

VIII. RESEARCH GAPS

The research gaps observed in this literature review are summed up in Fig. 39 and can be grouped into three basic categories, namely architecture-related, platform-related, and dataset related

- Platform-related:
  a. Native Mobile NST: Implementing real-time video neural style transfer directly on mobiles. Most applications implement style transfer on mobiles via a Client-server approach. This is primarily due to mobiles having relatively new software and low-power hardware.
  b. Use of Federated Learning: Federated learning is another gap observed while looking at Mobile NST. It is a recent idea and has been used to overcome low power device limitations.
- Dataset Related:
  a. Lack of benchmark datasets: As discussed previously, multiple papers mix and match datasets by re-purposing them from different domains. While this has the pros of swapping and replacing datasets in training, the need for a benchmark dataset can be seen for evaluation purposes. A benchmark dataset could make testing, evaluating, and understanding the model's performance standardized. Another point observed is that some articles create their datasets and apply different transforms to data, which can distort the image's structure, leading to the generation of artifacts.
  b. Lack of a good benchmark metric: It is observed and discussed above that many papers turn to Amazon

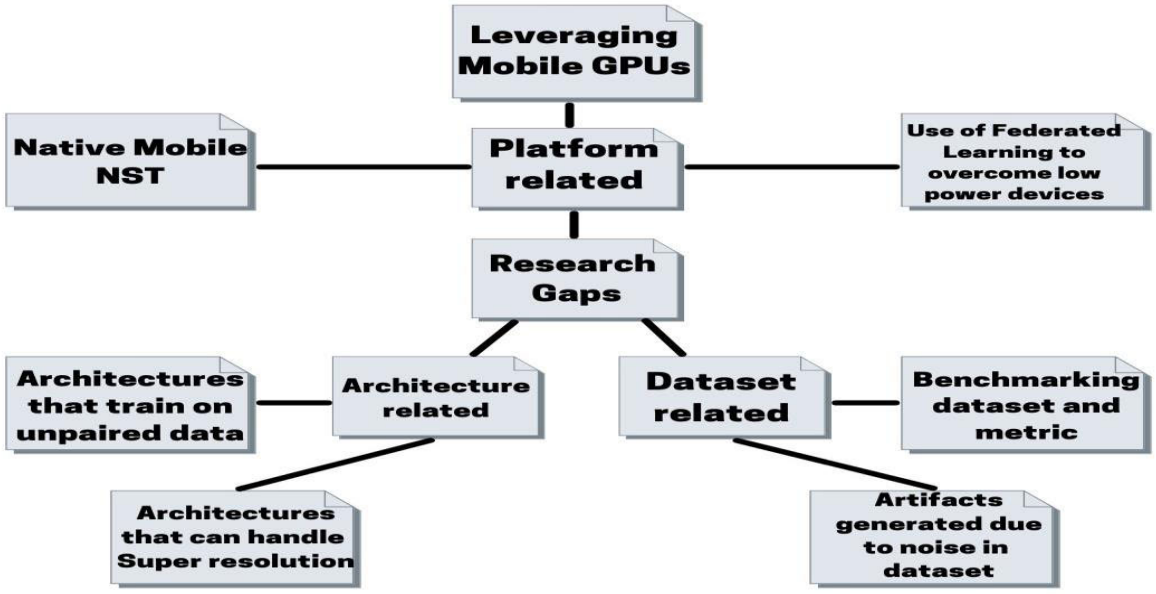

FIGURE 39. Research gaps.

M-Turks (a service that offers manual labor) to inspect the quality of the images generated. Photorealism is usually inspected manually and thus could be a place to add a metric. However, this can be difficult as photorealism is subjective and might change depending on context. In addition, as discussed previously, whereas there are metrics such as Intersection over Union or Accuracy, they rely on ''comparing'' two similar images. This can be particularly challenging to use as one needs some ''ground truth'' to compare to, and paired samples can be tricky to obtain.

- Model Architectures: It is seen that many of the models cannot handle super-resolution very well. The scalability of models in terms of the resolution of generated images is thus another concern. Apart from this, most data available or pieced together is usually unpaired, meaning the content and style images do not have the same structural composition.

IX. CONCLUSION AND FUTURE SCOPE

NST, one of the exhilarating AI applications adopted for artistic use of photos and videos, has started capturing the attention of GANs researchers in the last few years. These papers consisted of a comprehensive study of GANs and Video NST, divided into four parts. Initially, the working of GANs has been explained and its recent development on the different types of models for NST on mobile devices like CartoonGAN, Artsy-GANs, etc. The unpaired images can be used for training GANs using CycleGANs. Furthermore, adding ''temporal losses'' allows consistency between adjacently generated frames as seen over multiple architectures. Then the GANs improvement papers, explaining how Spatial, Color, and Scale control can allow better image generation. Lastly, how NST can be applied over mobile devices in real-time using GANs has been explained.

However, real-time NST on mobile devices with a reasonable frame rate is still relatively difficult to achieve. As time progresses, low power devices and devices with a smaller footprint will perform and handle large-scale computation better. This will be an exciting avenue to investigate, considering NST can be used in Augmented Reality. Non-iterative video NST is a good topic for future research since it can considerably reduce the time required to process videos. Since NST has vast potential, its research would see growing exponentially in coming years.

REFERENCES

[1] *The Smartphone vs. the Camera Industry*. Accessed: Apr. 20, 2021. [Online]. Available: https://photographylife.com/smartphone-vs-camera-industry/amp

[2] *Adobe Premiere Pro*. Accessed: Apr. 20, 2021. [Online]. Available: https://www.adobe.com/in/products/premiere/movie-and-film-editing.html

[3] *DaVinci Resolve*. Accessed: Apr. 20, 2021. [Online]. Available: https://www.blackmagicdesign.com/in/products/davinciresolve

[4] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, ''Neural style transfer: A review,'' *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 11, pp. 3365–3385, Nov. 2020.

[5] H. Li, *A Literature Review of Neural Style Transfer*. Princeton NJ, USA: Princeton Univ. Technical Report, 2019.

[6] J. Li, Q. Wang, H. Chen, J. An, and S. Li, ''A review on neural style transfer,'' *J. Phys., Conf. Ser.*, vol. 1651, Nov. 2020, Art. no. 012156.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, ''Generative adversarial networks,'' 2014, *arXiv:1406.2661*. [Online]. Available: http://arxiv.org/abs/1406.2661

[8] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," 2018, *arXiv:1812.04948*. [Online]. Available: http://arxiv.org/abs/1812.04948
[9] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434
[10] A. Dosovitskiy, P. Fischer, J. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2014.
[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2017, *arXiv:1703.10593*. [Online]. Available: http://arxiv.org/abs/1703.10593
[12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," 2016, *arXiv:1611.07004*. [Online]. Available: http://arxiv.org/abs/1611.07004
[13] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2414–2423.
[14] Y. Jin, J. Zhang, M. Li, Y. Tian, H. Zhu, and Z. Fang, "Towards the automatic anime characters creation with generative adversarial networks," 2017, *arXiv:1708.05509*. [Online]. Available: http://arxiv.org/abs/1708.05509
[15] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of GANs," 2017, *arXiv:1705.07215*. [Online]. Available: http://arxiv.org/abs/1705.07215
[16] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," 2016, *arXiv:1609.04802*. [Online]. Available: http://arxiv.org/abs/1609.04802
[17] Y. Chen, Y.-K. Lai, and Y.-J. Liu, "CartoonGAN: Generative adversarial networks for photo cartoonization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9465–9474.
[18] H. Liu, P. N. Michelini, and D. Zhu, "Artsy-GAN: A style transfer system with improved quality, diversity and performance," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 79–84.
[19] C. Li and M. Wang, "Precomputed real-time texture synthesis with Markovian generative adversarial network," in *Proc. ECCV*, 2016, pp. 702–716.
[20] V. Kitov, K. Kozlovtsev, and M. Mishustina, "Depth-aware arbitrary style transfer using instance normalization," 2019, *arXiv:1906.01123*. [Online]. Available: http://arxiv.org/abs/1906.01123
[21] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "StyleBank: An explicit representation for neural image style transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1897–1906.
[22] C. Zheng and Y. Zhang, "Two-stage color ink painting style transfer via convolution neural network," in *Proc. 15th Int. Symp. Pervas. Syst., Algorithms Netw. (I-SPAN)*, Oct. 2018, pp. 193–200.
[23] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman, "Controlling perceptual factors in neural style transfer," 2016, *arXiv:1611.07865*. [Online]. Available: http://arxiv.org/abs/1611.07865
[24] A. Gupta, J. Johnson, A. Alahi, and L. Fei-Fei, "Characterizing and improving stability in neural style transfer," 2017, *arXiv:1705.02092*. [Online]. Available: http://arxiv.org/abs/1705.02092
[25] L. A. Gatys, M. Bethge, A. Hertzmann, and E. Shechtman, "Preserving color in neural artistic style transfer," 2016, *arXiv:1606.05897*. [Online]. Available: http://arxiv.org/abs/1606.05897
[26] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," 2017, *arXiv:1703.07511*. [Online]. Available: http://arxiv.org/abs/1703.07511
[27] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "GauGAN: Semantic image synthesis with spatially adaptive normalization," in *Proc. ACM SIGGRAPH Real-Time Live*, Jul. 2019, p. 1, doi: 10.1145/3306305.3332370.
[28] N. Makow and P. Hernandez, "Exploring style transfer: Extensions to neural style transfer," Stanford Univ., Stanford, CA, USA, Tech. Rep. 2017-428, 2017.
[29] R. Dhir, M. Ashok, S. Gite, and K. Kotecha, "Automatic image colorization using GANs," in *Soft Computing and its Engineering Applications* (Communications in Computer and Information Science), vol. 1374, K. K. Patel, D. Garg, A. Patel, P. Lingras, Eds. Singapore: Springer, 2020, pp. 15–26, doi: 10.1007/978-981-16-0708-0_2.
[30] R. Dhir, M. Ashok, and S. Gite, "An overview of advances in image colorization using computer vision and deep learning techniques," *Rev. Comput. Eng. Res.*, vol. 7, no. 2, pp. 86–95, 2020. [Online]. Available: http://www.conscientiabeam.com/journal/76/abstract/6190, doi: 10.18488/journal.76.2020.72.86.95.
[31] M. Ruder, A. Dosovitskiy, and T. Brox, "Artistic style transfer for videos," 2016, *arXiv:1604.08610*. [Online]. Available: http://arxiv.org/abs/1604.08610
[32] H. Huang, H. Wang, W. Luo, L. Ma, W. Jiang, X. Zhu, Z. Li, and W. Liu, "Real-time neural style transfer for videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 783–791.
[33] W. Dudzik and D. Kosowski, "Kunster—AR art video maker—Real time video neural style transfer on mobile devices," 2020, *arXiv:2005.03415*. [Online]. Available: http://arxiv.org/abs/2005.03415
[34] H. Zhang and K. Dana, "Multi-style generative network for real-time transfer," 2017, *arXiv:1703.06953*. [Online]. Available: http://arxiv.org/abs/1703.06953
[35] D. Chen, J. Liao, L. Yuan, N. Yu, and G. Hua, "Coherent online video style transfer," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1105–1114.
[36] Q. Xu, G. Huang, Y. Yuan, C. Guo, Y. Sun, F. Wu, and K. Weinberger, "An empirical study on evaluation metrics of generative adversarial networks," 2018, *arXiv:1806.07755*. [Online]. Available: http://arxiv.org/abs/1806.07755
[37] B. Joshi, K. Stewart, and D. Shapiro, "Bringing impressionism to life with neural style transfer in come swim," 2017, *arXiv:1701.04928*. [Online]. Available: http://arxiv.org/abs/1701.04928
[38] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stereoscopic neural style transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6654–6663.
[39] Z. Hao, A. Mallya, S. Belongie, and M.-Y. Liu, "GANcraft: Unsupervised 3D neural rendering of minecraft worlds," 2021, *arXiv:2104.07659*. [Online]. Available: http://arxiv.org/abs/2104.07659
[40] W. Jiang, S. Liu, C. Gao, J. Cao, R. He, J. Feng, and S. Yan, "PSGAN: Pose and expression robust spatial-aware GAN for customizable makeup transfer," 2019, *arXiv:1909.06956*. [Online]. Available: http://arxiv.org/abs/1909.06956
[41] T. Nguyen, A. Tran, and M. Hoai, "Lipstick ain't enough: Beyond color matching for in-the-wild makeup transfer," 2021, *arXiv:2104.01867*. [Online]. Available: http://arxiv.org/abs/2104.01867
[42] O. Texler, D. Futschik, M. Kučera, O. Jamrička, Š. Sochorová, M. Chai, S. Tulyakov, and D. Sýkora, "Interactive video stylization using few-shot patch-based training," *ACM Trans. Graph.*, vol. 39, no. 4, p. 73, 2020.
[43] J. Song and J. Chul Ye, "Federated CycleGAN for privacy-preserving image-to-image translation," 2021, *arXiv:2106.09246*. [Online]. Available: http://arxiv.org/abs/2106.09246
[44] A. Li, C. Wu, Y. Chen, and B. Ni, "MVStylizer: An efficient edge-assisted video photo-realistic style transfer system for mobile phones," in *Proc. 21st Int. Symp. Theory, Algorithmic Found., Protocol Design Mobile Netw. Mobile Comput.* New York, NY, USA: Association for Computing Machinery, 2020, pp. 31–40, doi: 10.1145/3397166.3409140.
[45] A. Junginger, M. Hanselmann, T. Strauss, S. Boblest, J. Buchner, and H. Ulmer, "Unpaired high-resolution and scalable style transfer using generative adversarial networks," 2018, *arXiv:1810.05724*. [Online]. Available: http://arxiv.org/abs/1810.05724
[46] Y. Deng, F. Tang, X. Pan, W. Dong, C. Ma, and C. Xu, "StyTr^2: Unbiased image style transfer with transformers," 2021, *arXiv:2105.14576*. [Online]. Available: https://arxiv.org/abs/2105.14576
[47] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman, "Controlling perceptual factors in neural style transfer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3985–3993.
[48] *Computer Vision—ECCV 2018 Workshops*. New York, NY, USA: Springer, 2019.
[49] *Computer Vision—ECCV 2016*. New York, NY, USA: Springer, 2016.
[50] C. Zhou, Z. Gu, Y. Gao, and J. Wang, "An improved style transfer algorithm using feedforward neural network for real-time image conversion," *Sustainability*, vol. 11, no. 20, p. 5673, Oct. 2019.
[51] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Feb. 2, 2020, doi: 10.1109/TPAMI.2020.2970919.
[52] C. Gao, D. Gu, F. Zhang, and Y. Yu, "ReCoNet: Real-time coherent video style transfer network," 2018, *arXiv:1807.01197*. [Online]. Available: http://arxiv.org/abs/1807.01197
[53] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Visualizing and understanding generative adversarial networks extended abstract," 2019, *arXiv:1901.09887*. [Online]. Available: http://arxiv.org/abs/1901.09887

AKHIL SINGH is currently pursuing the B.Tech. degree in computer science with the Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune. His research interests include machine learning, deep learning, generative adversarial networks, and explainable artificial intelligence.

VAIBHAV JAISWAL is currently pursuing the B.Tech. degree in computer science with the Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune. His research interests include deep learning and generative adversarial networks.

GAURAV JOSHI is currently pursuing the B.Tech. degree in computer science with the Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune. His research interests include deep learning, generative adversarial networks, mixed reality, and game development.

ADITH SANJEEVE is currently pursuing the B.Tech. degree in computer science with the Symbiosis Institute of Technology, Symbiosis International (Deemed) University, Pune. His research interests include machine learning, deep learning, generative adversarial networks, and computer vision.

SHILPA GITE received the Ph.D. degree in deep learning for assistive driving in semi-autonomous vehicles from Symbiosis International (Deemed) University, Pune, India, in 2019. She is currently working as an Associate Professor with the Computer Science Department, Symbiosis Institute of Technology, Pune. She is also working as an Associate Faculty at the Symbiosis Centre of Applied AI (SCAAI). She is also guiding Ph.D. students in biomedical imaging, self-driving cars, and natural language processing areas. She has around 13 years of teaching experience. She has published more than 60 research articles in international journals and 25 Scopus indexed international conferences. Her research interests include deep learning, machine learning, medical imaging, and computer vision. She was a recipient of the Best Paper Award at 11th IEMERA Conference held virtually at Imperial College, London, in October 2020.

KETAN KOTECHA received the M.Tech. and Ph.D. degrees from IIT Bombay.

He is currently holding the positions as the Head of the Symbiosis Centre for Applied AI (SCAAI), the Director of the Symbiosis Institute of Technology, the CEO of the Symbiosis Centre for Entrepreneurship and Innovation (SCEI), and the Dean of the Faculty of Engineering, Symbiosis International (Deemed) University. He has expertise and experience in cutting-edge research and projects in AI and deep learning for the last 25 years. He has published more than 100 widely in a number of excellent peer-reviewed journals on various topics ranging from cutting-edge AI, education policies, teaching-learning practices, and AI for all. He has published three patents and delivered key note speeches at various national and international forums, including at the Machine Intelligence Laboratory, USA, IIT Bombay under World Bank Project, and the International Indian Science Festival organized by the Department of Science Technology, Government of India, and many more.

Dr. Kotecha was a recipient of the two SPARC projects worth INR 166 Lakhs from MHRD Government of India in AI, in collaboration with Arizona State University, USA, and the University of Queensland Australia. He was also the recipient of numerous prestigious awards like Erasmus+ Faculty Mobility Grant to Poland, DUO-India Professors Fellowship for research in responsible AI, in collaboration with Brunel University, U.K., LEAP Grant at Cambridge University, U.K., UKIERI Grant with Aston University, U.K., and a Grant from Royal Academy of Engineering, and U.K. under Newton Bhabha Fund. He is currently an Academic Editor and an Associate Editor of IEEE Access journal.