Applied Intelligence (2025) 55:306
https://doi.org/10.1007/s10489-024-06114-5

CaVIT: An integrated method for image style transfer using parallel CNN and vision transformer

ZaiFang Zhang[1] · ShunLu Lu[1] · Qing Guo[1] · Nan Gao[2] · YuXiao Yang[2]

Accepted: 23 November 2024

Abstract
This study focuses on image style transfer, aiming to generate images with the desired style while preserving the underlying content structure. Existing models face challenges in accurately representing both content and style features. To address this, an integrated method for image style transfer is proposed, utilizing a parallel CNN and Vision Transformer (CaVIT). It combines a Convolutional Neural Network (CNN) with a Vision Transformer (VIT) to achieve enhanced performance. Our method utilizes VGG-19 with residual blocks to encode style features for enhanced refinement. Additionally, the PA-Trans Encoder Layer is introduced, inspired by the Transformer Encoder Layer, to efficiently encode content features while preserving the complete content structure. The fused features are then decoded into stylized images using a CNN decoder. Qualitative and quantitative evaluations demonstrate that our proposed method outperforms existing models, delivering high-quality results.

Index Terms Image style transfer · Feature encoding · VIT · VGG-19

1 Introduction

The growing popularity of art has been fueled by the rapid advancement of social media and online technology. In the past, art creation was predominantly limited to professional painters, making it challenging for the general public to achieve a personalized painting style that resonated with their preferences. However, the emergence of image style transfer techniques and the rapid development of deep learning have democratized art, eliminating the notion of them being a privilege. Recent advancements in deep learning, such as its application in detecting and counting people in crowded environments using convolutional neural networks, have shown remarkable accuracy and robustness in complex visual tasks [1, 2]. This further supports the effectiveness of deep learning in image style transfer tasks. Image style transfer has become a focal point in the field of computer vision, representing a prominent area of research [3]. The task aims to generate a target image that combines style features from a source style image with content features from a source content image.

The most common traditional transfer methods were texture synthesis [4], which suffered from slow inference speeds and limited ability to transfer shallow features. The advent of neural networks has greatly facilitated the development of the following models, progressing from Per Style Per Model (PSPM) [5, 6], to Multiple Style Per Model (MSPM) [7–11], and eventually to Arbitrary Style Per Model (ASPM). However, previous neural network-based approaches often focused on extracting local features and were susceptible to content leakage. ArtFlow [12] effectively addressed this issue by utilizing a flow model. Nevertheless, in doing so, it sacrificed the generation of style features in the resulting images, preserving only simple attributes such as color while largely disregarding brushstrokes and artistic style.

✉ ZaiFang Zhang
zaifangzhang@shu.edu.cn

ShunLu Lu
15206289147@163.com

Qing Guo
gq211@shu.edu.cn

Nan Gao
nangao@shu.edu.cn

YuXiao Yang
yuxiao@shu.edu.cn

[1] School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China

[2] College of Sciences, Shanghai University, Shanghai 200444, China

Published online: 13 January 2025

Although the Transformer Encoder and Decoder structure [13] have effectively circumvented both of these issues, their model still encountered difficulties in preserving the content structure when applied to specific painting styles, such as the untrained painting style found in the dataset. Additionally, the inference time of the model was excessively long, hindering its practical applicability and scalability.

This paper introduces a novel encoder-decoder network architecture that combines CNN and VIT. The intermediate layer of VGG-19, known for its aptitude for extracting suitable style features [14], is employed for style feature encoding. Additionally, three residual blocks are added to refine the style features. Building upon the insights from Stytr2's superior characterization of content features [13], this paper utilizes its positional encoding for adapting the Transformer encoder scheme. The sequential structure of Multi-Headed Self-Attention (MSA) and Feedforward Neural Network (FFN) is altered to a parallel arrangement to alleviate computational overhead. Pretreatment of Query and Key (Q&K) in MSA is performed to prevent over-fitting and mitigate convergence difficulties in the loss function. Furthermore, the activation function RELU is replaced with GELU [15] to achieve appropriate attention weights. The fused features are then fed into adaptive instance normalization (AdaIN) [16] for simultaneous feature fusion, enabling arbitrary style transfer. Finally, the stylized image is decoded from the fused features. An identity loss term is introduced to enhance the feature extraction capability of encoders and the feature sampling ability of the decoder.

In summary, the main contributions are given as follows:

- A serial-combined network structure named CaVIT is introduced, employing two distinct encoders operating in parallel to extract style and content features. This architecture innovatively merges these features, utilizing a CNN decoder to synthesize stylized images that retain both the artistic style and the integrity of the original content. This method effectively addresses the challenge commonly encountered in style transfer tasks, where maintaining both style and content fidelity is problematic.
- This study introduces a novel Transformer Encoder known as the PA-Trans Encoder. The MSA and FFN modules simultaneously process the same input, aiming to preserve more complete content structure and enhance computational efficiency. To prevent degradation, Layer Normalization is applied to the input fed into the MSA or FFN modules. Furthermore, the Encoder Module utilizes GELU as the activation function for improved convergence.
- Residual blocks are incorporated to the VGG-19 Encoder module to refine style features. Furthermore, the loss computation involves the use of identity loss, which serves a dual purpose—preserving content information and improving the encoder's proficiency in feature extraction. These modifications significantly elevate the quality and visual appeal of the stylized images produced.

2 Related works

2.1 Image style transfer

Non-realistic rendering methods, which are based on brush strokes, were originally used [17–19] in this image style transfer area. Gatys et al. [14], as pioneers of deep learning in image style transfer, introduced a neural style transfer method which separated the style and content of an image. The Gram Matrix was calculated using VGG-19 [20] feature space to effectively capture statistical characteristics of input style images. Subsequently, the generated stylized images underwent iterative error backpropagation to optimize their overall appearance.

However, the global features of whole images were counted using a Gram matrix, which was not sensitive to the local distribution of the features. To address this limitation, Risser et al. [21] introduced histogram loss, which extracted the image distribution information and then combined it with the Gram matrix to constrain the style better. Building upon this work, Li et al. [22] proposed representing style through the mean and variance of inter-channel features, simplifying the computational steps while achieving more desirable style results. Zhang et al. [23] observed that existing loss calculation methods failed to adequately capture style information due to their reliance on second-order statistics. As a solution, they presented a novel approach that utilized mutual information from Contrast learning to compute style loss, leveraging a specially designed network structure to directly learn style relationships and distributions. While this method demonstrated improved style effects, it exhibited incomplete content integration, with numerous unidentified spots indicating a heavy dependence on specific styles. To ensure that the generated style map encompasses both style and content features, a method proposed by Li et al. [22] is adopted to calculate the content and style loss. Additionally, the identity loss technique developed by SANet [24] is employed to further constrain the features.

The process of image style transfer has evolved from PSPM to MSPM, ultimately culminating in ASPM, aimed at enhancing the efficiency of style transfer. Currently, ASPM has garnered widespread popularity due to its network structure, encoder-decoder framework, and the advantageous ability to convert graphs-to-graphs. Huang et al. [16] pioneered the implementation of ASPM through AdaIN, aligning statistical information on content and style features while

adapting fusion in accordance with randomly selected input style and content features. However, this approach resulted in the generation of stylized images with numerous artifacts and cracks. Subsequently, Jing et al. [25] developed dynamic instance normalization (DIN) as an alternative to AdaIN. Unlike AdaIN, which utilizes pre-trained weights and biases, DIN trained the network's weights and biases to obtain the mean and variance, leading to stylized images with fewer artifacts. Apart from these, Svoboda et al. [26] utilized a customized graph convolution layer to create a hidden space with a combination of global and local features and employed metric learning to separate content and style features, thus avoiding cracks. Another approach was proposed by Li et al. [27] who introduced a statistically based feedforward network capable of rapidly generating stylized images through linear feedforward computation. Despite various models enhancing the flexibility and feature extraction capabilities of the feedforward network, most of them still rely on the VGG encoder and decoder. However, these CNN-based encoder-decoder structures suffer from similar drawbacks, including incomplete capture of global features, a limited perceptual field, and a susceptibility to falling into local optima, which can ultimately result in the production of subpar content.

As a result, attention mechanisms have been introduced in some models. Park et al. [24] spatially rearranged the style features to make them semantically relevant to the content features. They developed a new attention module to directly fuse the obtained features, although it generated coarser images. Yao et al. [28] introduced a self-attention mechanism as a complement to the framework of the auto-encoder in order to extract salient features from the content image. They utilized these features harmoniously to fuse multi-stroke patterns into different spatial regions of the output image, but it led to blurrier generated images. Revisit attention mechanism in arbitrary neural style transfer (AdaAttN), proposed by Liu et al. [29] performed attention mechanism normalization at each pixel point. Its content features were more complete compared to the extraction of style features, and the generated images sometimes carried the style features of the input content images. The various phenomena described above highlight a significant problem caused by the network structure of CNN: content leakage. Although flow models were suggested to address this issue, their construction could be challenging, and effective transfer of style features may not be achieved.

Furthermore, there exist various image style transfer methods based on GAN [30], which incorporated an additional discriminator constraint compared to neural networks. The general public initially used CGAN [31] for image translation, which shared similarities with the style transfer process. The CycleGAN proposed by Zhu et al. [32] is the pioneer of GAN in the field of image style transfer. They introduced the cycle consistency commonly used in machine translation to provide constraints for the generator, playing an equivalent role to content loss in style transfer. Dmytro Kotovenko et al. [33] incorporated content variation blocks to constrain the content features while DRB-GAN [34] developed dynamic residual blocks that connect the style encoding network and the style transfer network, reducing the disparity between individual style transfer and set style transfer in a single model. However, these models based on GAN are not widely considered due to their generation uncertainty and the complexity of network structure.

To ensure stability in the style transfer process, we have also referred to relevant studies from other fields. These studies focus on optimizing system stability and managing computational complexity. For instance, the stability and robustness of complex genetic regulatory networks have been analyzed using stochastic Markov models with time delays [35], which inspired us to incorporate a parallel structure to reduce computational complexity and accelerate model training. Similarly, the input-to-state stability theory has been applied to stochastic neural networks with Markovian parameters to derive stability conditions [36]. This research motivated us to simplify the computational complexity of our model and enhance stability through effective parameter control. These studies provided valuable insights for designing a model with enhanced stability and reduced computational burden, ultimately enabling a more stable and efficient style transfer process in our research.

2.2 Transformer encoder for feature extraction

The advancement of the Transformer model has witnessed substantial progress in the field of Natural Language Processing (NLP).Then the application of Transformer in vision (VIT) was first proposed by Dosovitskiy et al. [37] which showed excellent results in the field of image classification.

Transformer was gradually applied to the fields of object detection and image segmentation, such as Deformable-DETR [38] and SegFormer [39] etc. Deng et al. [13] first applied it to image style transfer and obtained fairly good generation results. The fundamental concept of the approach involved encoding features by MSA and FFN modules, complemented by a residual connection. However, attention to global information increased the computational burden inevitably. Additionally, Transformer Encoder required the Transformer decoder. Wu et al. [40] had also applied Transformer modules; their approach developed a Transformer-driven Style Composition to produce affine coefficients that can be used to control style transfer.

StyTr2 [13] has developed encoding and decoding mechanisms with Transformer Encoder-Decoder in VIT [37]. They introduced an optional encoding technique called Content-Aware Positional Encoding (CAPE) to control multi-scale

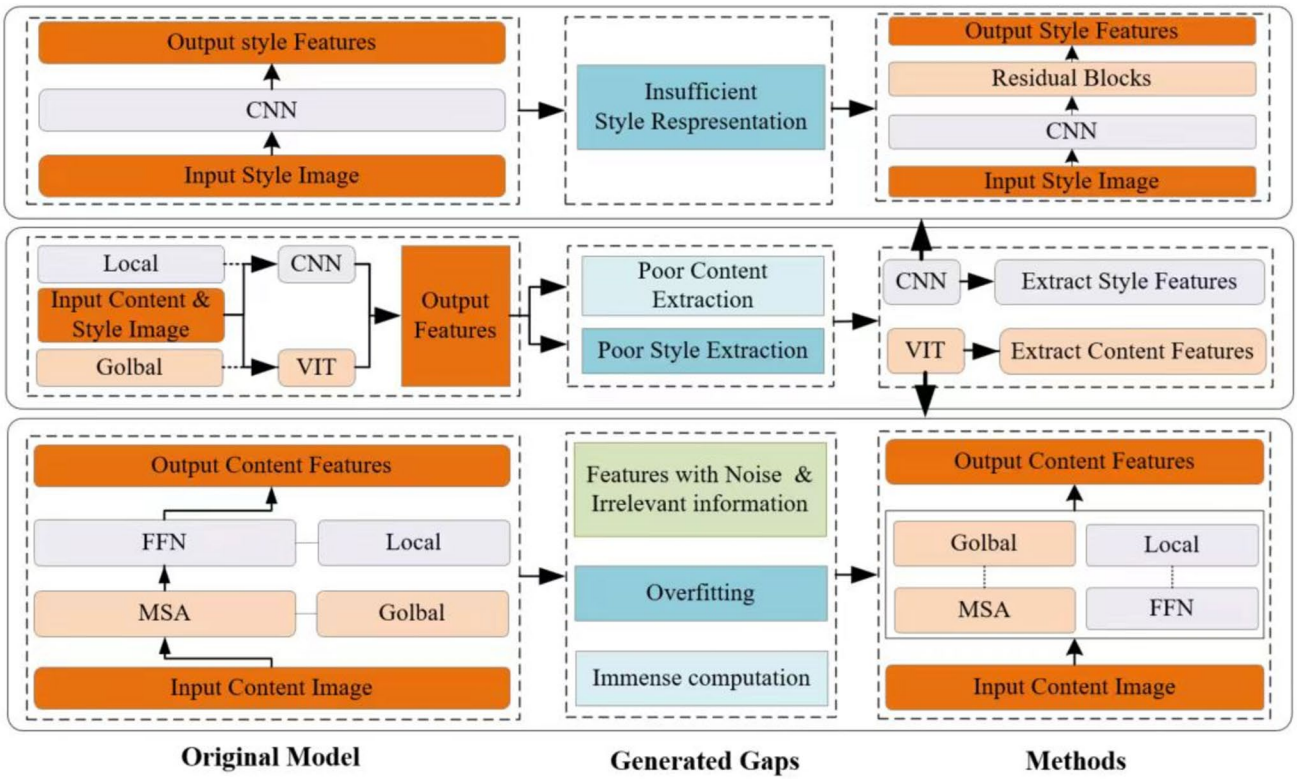


Fig. 1 Generated gaps and methods compared to original models

differences, resulting in smaller losses in both the content and style domains. However, when transferring certain styles, the content structure still remains fragmented. Our model simplifies the encoding of content features using an adapted Transformer Encoder, referred to as the PA-Trans Encoder, along with a CNN Decoder instead of a Transformer Decoder. In conclusion, our model not only achieves comparable results to the previous method in StyTr2 but also significantly improves computational efficiency.

3 Methods

The core objective of image style transfer is to generate a stylized image that follows the guidance of input style images while preserving the content structure of the input content images. However, previous models, whether utilizing VGG-19 or Transformer, had the potential to degrade certain features during the process of features' extraction. Figure 1 provides an illustration of the identified defects of common models and proposes assumptions as solutions. One of the primary challenges faced by existing models is the inadequate content extraction and style representation, which hinders the achievement of a balanced outcome between the two. As a consequence, generated images frequently exhibit unidentified noise or content distortion.

This study introduces two distinct encoders for separately extracting style and content features from input images, leading to significant advancements for each encoder individually. The VGG-19 network is utilized in this study to extract style features. The uniform kernel size design of VGG-19 significantly reduces the model's trainable parameters while effectively extracting features. An increase in the number of layers enhances the network's capability to extract deep features. Typically, scholars use this network to extract both content and style features from images. By removing the three fully connected layers at the end, the feature map is directly output, facilitating the separation of style and content. The specific network architecture is depicted in Fig. 2.

3.1 A network architecture

A brand-new joint parallel encoding mechanism is proposed to overcome the defects of incomplete content retention and style transfer. After the referred style image (I_S) is input into the VGG-19 pre-trained encoder, these extracted features are consigned to three residual blocks (details will be shown in Section C) for more refined features. According to Stytr2 [13], the referred content image (I_C) is embedded into patches. These patches are expressed as $\varepsilon_C = \{\varepsilon_{C_1}, \varepsilon_{C_2}, \cdots, \varepsilon_{C_n}\}$. CAPE replaces original

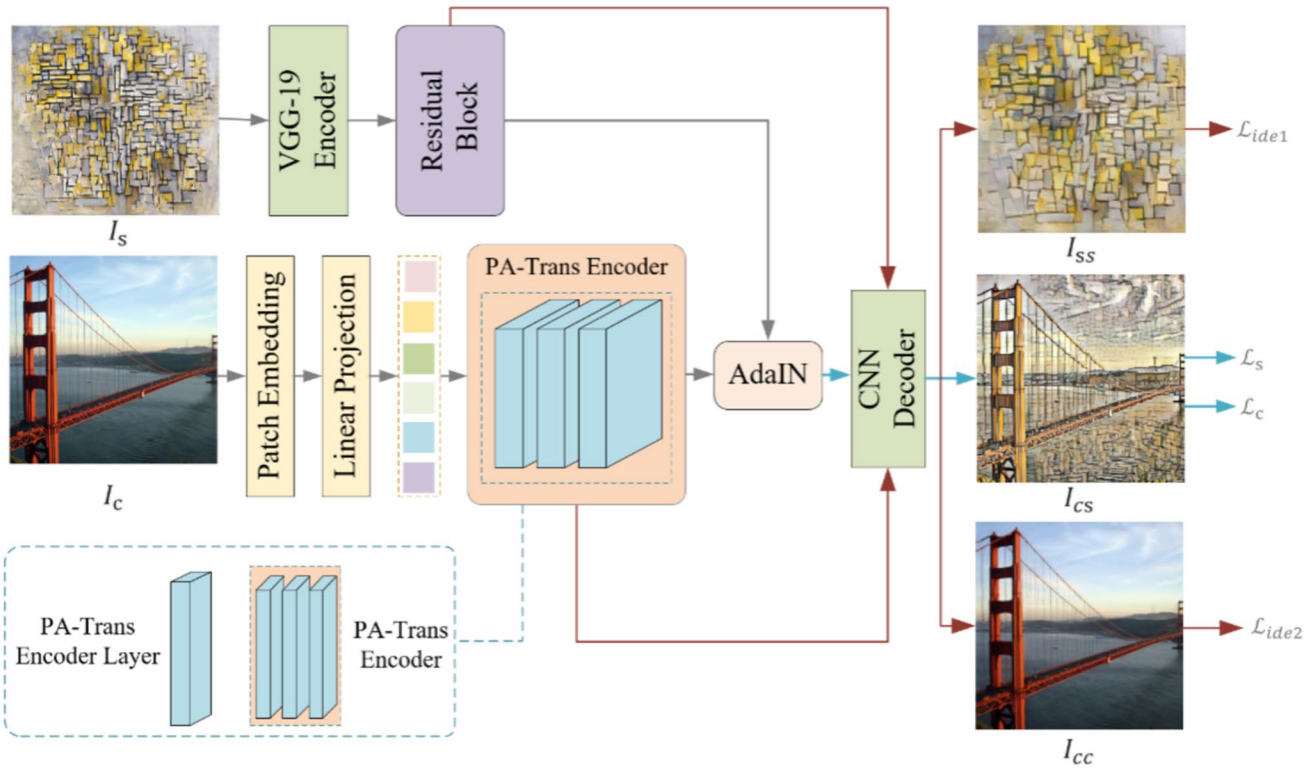


Fig. 2 Proposed architecture of style transfer

position encoding to cater to the demands of multi-visual tasks. These features after CAPE are represented as $P_{CA} = \{P_{CA1}, P_{CA2}, \cdots, P_{CAn}\}$. The specific equations are shown as follows:

$$P_{\mathcal{L}} = POS\left(AvgPool_{n\times n}\left(\varepsilon_C\right)\right) \tag{1}$$

$$P_{\mathcal{CA}}(x,y) = \sum_{m=0}^{s}\sum_{n=0}^{s}\left(a_{mn}P_{\mathcal{L}}\left(x_m, y_n\right)\right) \tag{2}$$

where POS represents learnable function of positional encoding, a_{mn} represents interpolation weights, s represents number of peripheries, $\left(x_m, y_n\right)$ represents two-dimensional coordinates and n is set to 18. The content input with CAPE is defined as $X_C = \left\{\varepsilon_{C1} + P_{\mathcal{CA}1}, \varepsilon_{C2} + P_{\mathcal{CA}2}, \cdots, \varepsilon_{Cn} + P_{\mathcal{CA}n}\right\}$. It is fed into PA-Trans Encoder (a novel block, details will be shown in Section B). The extracted content and style features are then fused by using Eq. (3) to obtain features of generated stylized images (I_{CS}).

$$AdaIN(x) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \mu(y) \tag{3}$$

where x represents content features, $\mu(x)$ and $\mu(y)$ represent variance of content and style features separately, $\sigma(x)$ and $\sigma(y)$ represent standard deviation of content and style features respectively. At last, an iterative CNN decoder will generate the stylized image through the fused features.

In conclusion, the network architecture is concise and simple, ensuring excellent readability. Furthermore, it avoids any compatibility issues associated with different encoder types when utilizing the same decoding format. Identity loss ($\mathcal{L}_{ide1}$ and $\mathcal{L}_{ide2}$) is used to improve the capacity of encoders to extract features except routine content and style loss ($\mathcal{L}_C$ and $\mathcal{L}_S$). The specific parameter settings and calculation formulas are shown in Section D.

3.2 PA-trans encoder layer

A distinct encoder layer, referred to as the PA-Trans Encode Layer, is proposed in Fig. 3. This layer introduces parallel MSA and FFN modules to enhance feature extraction capabilities. Layer normalization is additionally used to speed up model convergence, and the GELU activation function is applied as a precaution against over-fitting.

3.3 Parallel structure

The original Transformer Encoder Layer in VIT arranged the MSA and FFN in a sequential structure. The output of MSA is passed into FFN after Layer Normalization, which increases the training pressure to some extent and trainable

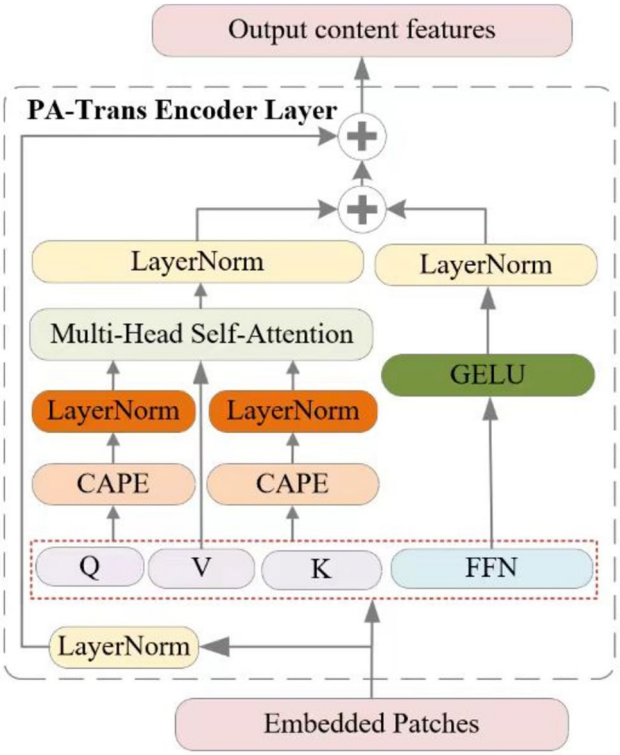


Fig. 3 PA-Trans Encoder Layer

parameters in MSA may influence the backward propagation of FFN. When training large scales of parameters by using the Transformer model in NLP, GPT-J 6B proposed by Ben Wang et al. [41] and Google's PaLM model [42] were all based on a new encoding mode that accelerated the training speed and ensured experimental effectiveness. Referring to it, MSA and FFN are changed into a parallel structure so that the two modules work simultaneously. The part that should be input into MSA is encoded as Q (Query), K (Key), V (Value), where X_C is input into $Q\&K$, ε_C is input into $V\&FFN$. Assumed that the final output is Y, the equation is shown below:

$$Y = LN\left[\left(\varepsilon_C\right) + MSA(Q,K,V) + FFN\left(\varepsilon_C\right)\right] \tag{4}$$

where LN represents Layer Normalization, the equations of the MSA and FFN modules will be discussed in succession. After conducting a series of experiments, it was discovered that this structure also yields good performance in the image style transfer task. The generated stylized images do not compromise performance, but rather decrease training time.

By adopting a parallel structure, the PA-Trans Encoder Layer significantly enhances the model's ability to preserve the content structure during the style transfer process. Unlike sequential models where the processing delays and dependencies between layers could lead to a dilution of content fidelity, the parallel structure ensures that both content and style features are processed simultaneously without compromise. This method not only maintains the integrity of the content's structural details but also aligns closely with the style attributes of the target, resulting in high-quality stylized outputs that faithfully reflect the desired artistic intent.

3.4 Q&K layer normalization in MSA

In Self-Attention mechanism, attention weight is multiplied with Softmax by Q and K and the attention score is equal to the attention weight multiplied by V. If the attention score is too high or too low, it may cause the model to diverge.

According to the theory proposed by Justin Gilmer [38], it is suggested that Q and K undergo Layer Normalization before being fed into the model to achieve better loss convergence. The equations for MSA are shown below:

$$Q\prime = X_c W_q K' = X_c W_k \tag{5}$$

$$Q = LN\left(Q'\right), K = LN\left(K'\right), V = \varepsilon_c W_v \tag{6}$$

$$MSA(Q,K,V) = Concat\left(Attention_1(Q,K,V),\ \cdots, Attention_N(Q,K,V)\right) \tag{7}$$

where W_q, W_k, W_v are learnable weight matrix, $\mathbb{R}\in^{C\times d_{head}}$, C means input channels and d_{head} is the dimension of attention head and set to 8. W_0 is learnable parameter $\mathbb{R}\in^{C\times C}$.

3.5 GELU activate function in FFN

The paper replace the original Transformer Encoding Layer's RELU activation function with the GELU activation function for the FFN to capture nonlinear patterns better. The equations are shown below:

$$\mathrm{ReLU}(x) = \max(0,x) \tag{8}$$

$$GELU(x) = xP(X \le x) = x\Phi(x) \tag{9}$$

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{\frac{-t^2}{2}}dt \tag{10}$$

It is apparent from Eqs. (8) and (9) that utilizing the RELU activation function creates a lot of zero values, which might

contribute to a loss of crucial information. In contrast, the GELU function employs an S-shaped smooth transition around zero, with values approaching zero in the negative region and a form approximating a constant function in the positive region. The equation for FFN is shown below using GELU:

$$FFN(\varepsilon_C) = GELU(W_1\varepsilon_C + b_1)W_2 + b_2 \tag{11}$$

where W_1, W_2 represent weight, $W_1 \in \mathbb{R}^{d_{model} \times C \times H \times W}$, $W_2 \in \mathbb{R}^{d_{model} \times C \times 1 \times 1}$; $b_1 \in \mathbb{R}^{d_{\text{model}} \times R}$, $b_2 \in \mathbb{R}^{d_{\text{model}}' \times R}$; d_{model}, d'_{model} represent the input and output dimension; b_1, b_2 represent bias; R depends on the actual situation. It is valid to use GELU for avoiding the issue of "dead gradient" and accelerating network training speed.

3.6 Residual blocks

When using only the pre-trained VGG-19 model to extract style features from the ReLU4_1 layer, the focus is primarily on local features, resulting in a degradation of style features while involving deep networks. Moreover, the transfer process may not adequately refine the overall style representation, as content features from the input style image can persist. To address this, the extracted style features are incorporated into the Residual Block to achieve the best possible feature representation. The network structure is illustrated in Fig. 4.

Each residual block comprises the following elements: The network architecture includes two ReflectionPad2d layers to preserve boundary information while performing convolutions. Additionally, two Conv2d layers with a kernel size of 3 are employed to maintain consistency in the number of channels between the input and output feature maps. The network further incorporates two InstanceNorm2d layers, which contribute to faster convergence during the training process. Lastly, a RELU layer is included to enhance the network's ability to perform nonlinear fitting. The utilization of Residual Blocks facilitates the flow of information from lower-level features to higher-level layers by establishing connections across layers, leading to a more comprehensive and sophisticated feature representation.

3.7 CNN decoder layer

Due to the use of different encoder structures for style features and content features, fusion of the two is required to decode

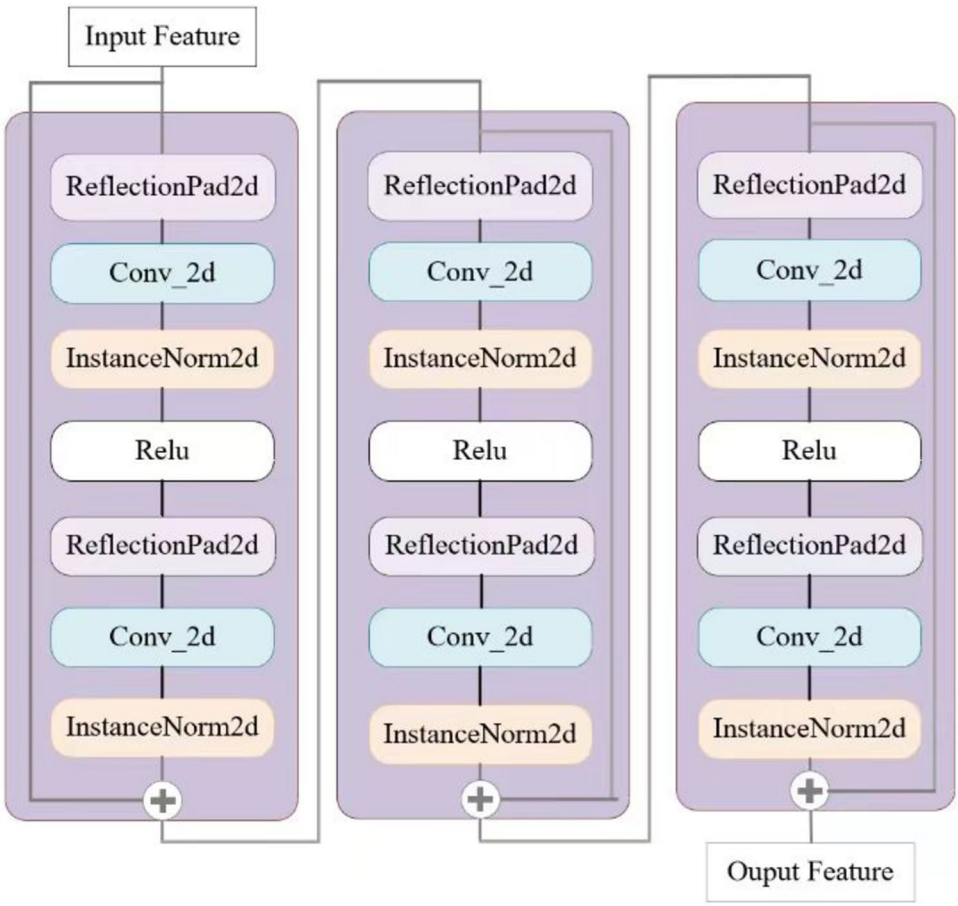


Fig. 4 Residual Block Structure

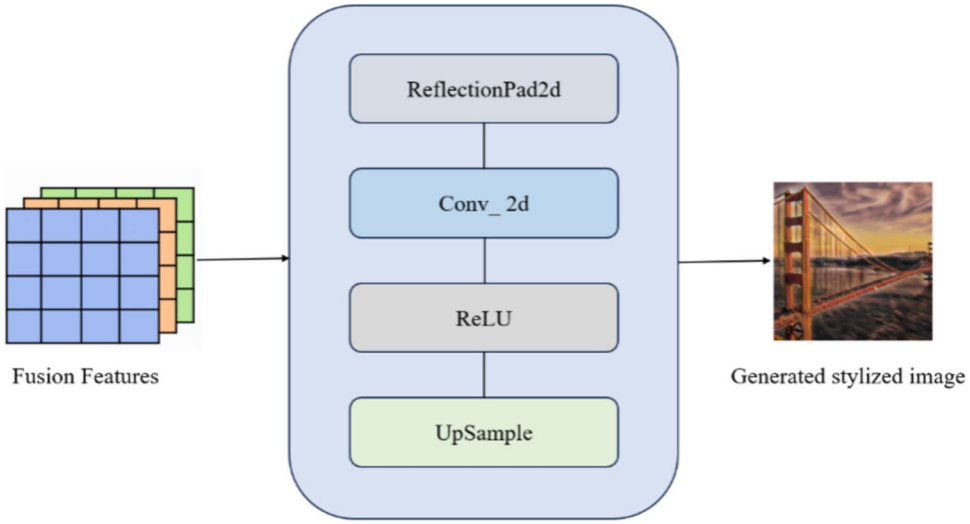


Fig. 5 CNN Decoder Layer

and generate the final stylized image. Referring to the adaptive normalization algorithm proposed in AdaN, the affine parameters originally in instance normalization are learned directly from the style input, allowing for the adaptive adjustment of the mean and variance of the content features.

The final stylized images are generated using a deconvolution network for decoding, the structure of which is illustrated in Fig. 5. The network predominantly consists of a series of 3×3 convolutional layers, bilinear interpolation upsampling layers, and ReLU activation functions. The resulting stylized images are produced with dimensions $H \times W \times C$, where H, W, and C respectively represent the height, width, and number of channels of the image.

3.8 Loss computation

In addition to the conventional style and content loss, identity loss is applied to maximize the retention of both content and style features. To generate the content image I_{CC} and style image I_{SS}, extracted style and content features are re-decoded. The mean–variance loss is calculated with the input content image I_C and style image I_S respectively. The total loss function formula is presented below:

$$\mathcal{L} = \alpha\mathcal{L}_S + \beta\mathcal{L}_C + \gamma_1\mathcal{L}_{ide1} + \gamma_2\mathcal{L}_{ide2} \tag{12}$$

where $\mathcal{L}_S$ and $\mathcal{L}_C$ represent style loss and content loss; $\mathcal{L}_{ide1}$ and $\mathcal{L}_{ide2}$ represent identity loss; α, β, γ_1 and γ_2 represent parameter coefficients for each component. Based on the Stytr2 and SANet methods, the approximate range of each hyperparameter is initially determined. Through repeated experiments and comparisons, parameters are finalized to achieve a balance between style and content feature extraction. The hyperparameters for style and content loss are roughly set to the following combinations: (8,9), (8,10), (9,9), (9,10), (10,9), and (10,10). Upon evaluation, the best performance is achieved with the combination (9,10). Consistency loss is found to have a relatively smaller impact on the final outcome, and after a few additional experiments, it is finalized as (70,1). In the end, α, β, γ_1 and γ_2 is set to 9,10,70 and 1 to minimize the total loss.

Drawing on the content and style loss calculation method proposed in AdaIN, the features extracted from the RELU5_1 layer serve as a reference benchmark for calculating the loss layer by layer with the generated stylized image features. These are achieved by using the following equations:

$$\mathcal{L}_C = \frac{1}{N_n}\sum_{i=0}^{N_n} \left\| \begin{matrix} \Phi_i(I_{CS}) \\ -\Phi_i(I_C) \end{matrix} \right\|_2 \tag{13}$$

$$\begin{aligned} \mathcal{L}_S = \frac{1}{N_n}\sum_{i=0}^{N_n} & \left\| \mu(\Phi_i(I_{SS})) - \mu(\Phi_i(I_S)) \right\|_2 \\ & + \left\| \sigma(\Phi_i(I_{SS})) - \sigma(\Phi_i(I_S)) \right\|_2 \end{aligned} \tag{14}$$

where I_{CS} represents stylized image; $\Phi_i(\cdot)$ represents features extracted by the $i-th$ convolutional layer; N_n represents amount of convolution layers.

The value loss of the ratio of I_{CC} to I_{SS} and the ratio of I_C to I_S are minimized for more precise feature representation. Two types of identity loss are proposed, where one reduces the loss by directly computing the difference between image pixel levels, while the other, similar to Eqs. (13) and (14), reduces the loss by computing the difference of features between images. The equations are shown below:

$$\mathcal{L}_{ide1} = \|I_{CC} - I_C\|_2 + \|I_{SS} - I_S\|_2 \tag{15}$$

$$\mathcal{L}_{ide2} = \frac{1}{N_n}\sum_{i=0}^{N_n} \left\| \phi_i(I_{CC}) - \phi_i(I_C) \right\|_2 + \left\| \phi_i(I_{SS}) - \phi_i(I_S) \right\|_2 \tag{16}$$

4 Experiments

4.1 Implementation details

The proposed model is trained on the CoCoTrain2014 dataset [43], containing 82,784 images as content images, and the WikiArt dataset [44], containing 79,998 images from 27 different art styles as style images. The optimizer used for the experiments is Adam, which dynamically adjusts the learning rate, with an initial learning rate of $5e^{-4}$. In the training process, the batch size is set to 8, and images with various resolutions are randomly cropped to 256×256. The model is trained for a total of 160,000 epochs. During the testing process, the model demonstrated its ability to handle images of different resolutions while ensuring high-quality image generation.

In our training process, two RTX 3090 GPUs are utilized, which takes a total of 12 h. The CPU configuration includes 30 vCPUs with an Intel(R) Xeon(R) Platinum 8358P CPU @ 2.60 GHz, and the GPU is an RTX 3090 (24 GB) with the PyTorch 1.11.0 framework running on Python version 3.8.

4.2 Generation evaluation

Qualitative and quantitative analysis are applied to the evaluate generated stylized images. Deng et al. [45] proposed a graph-based learning method to calculate style representation, but it was only suited for famous artists' paintings. Three methods are employed for quantitative analysis, including LPIPs (Learned Perceptual Image Patch Similarity) for calculating content loss, the inference time of the model, and a user study. A comparative analysis is conducted between the model and six other style transfer models, namely StyTr2 [13], AdaAttN [29], ArtFlow (AdaIN) [12], CAST [23], IEST [46], and AdaIN [16]. These models are primarily based on neural networks, except for CAST, which is based on GAN.

4.3 Qualitative evaluation

Seven diverse painting styles are selected for transfer onto various types of content images, including portraits, animals, landscapes, etc. Figure 6 displays the transfer effects of each model for the same content and style images. From the results, it is evident that the stylized images generated by AdaIN exhibit artifacts (rows 2, 3, and 7) and blocky regions (rows 5 and 6). CAST commonly experiences issues, showing dense speckles and severe visual appearance in most stylized images, with the exception of row 1 in column 6. ArtFlow, which utilizes the flow model, produces images with less prominent style features and even creates "self-invented" styles not present in style images. For instance, rows 5 display a white halo, rows 6 exhibit purple lips on the character, and rows 7 show a light purple kitten's nose. IEST encounters similar issues problems, resulting in less distinct style features (column 8, rows 5, 6, and 7). Moreover, the model significantly disrupts the content structure (rows 2), leading to considerable differences from the reference content image. While Stytr2 generally produces high-quality stylized images, it suffers from significant content loss when transferring ink-style images (rows 2). CAPE in Stytr2 aims to reduce the generation of different styles of the same image semantics, but its performance lacks consistency, as observed from the two red boxes (rows 4). AdaAttN demonstrates relatively excellent content preservation; however, inconsistencies in style are present between the generated images and the input style images (rows 3, 5, and 7).

In contrast to the aforementioned models, the approach integrates the advantages of two distinct, optimized encoders, producing superior overall results. The stylized images generated successfully maintain an effective balance between style and content features (column 3). These images not only closely match the style characteristics of the input style images but also preserve the structural integrity of the input content images. This fidelity to both style and content showcases a significant improvement in visual outcomes compared to other algorithms, demonstrating the model's capacity to combine aesthetic appeal with content preservation effectively.

4.4 Quantitative evaluation

A) Test Time.

Separate tests are conducted on a local GPU (NVIDIA RTX 3060) using arbitrary content and style images at two different resolutions (256×256 and 512×512). The start time T_S and end time T_E are obtained by 'date' command. By utilizing the 'inference' command, the path of the trainable weight, input images, and output images is specified to measure the test time. Finally, the time difference, namely inference time $T_I = T_E - T_S$, is calculated by 'exp' command. These commands are implemented through Shell scripts. The test times for each model are shown in Table 1 below (The bold number indicates the best result, and the underlined number represents the second best):

Our model demonstrates remarkable inference efficiency when processing images with a resolution of 256×256. Although there is a notable increase in inference time when the resolution is raised to 512×512, our model remains approximately three times faster than Stytr2, which employs a similar Content Encoder mechanism. This efficiency demonstrates our model's optimized architectural design, which not only supports faster processing at higher resolutions but

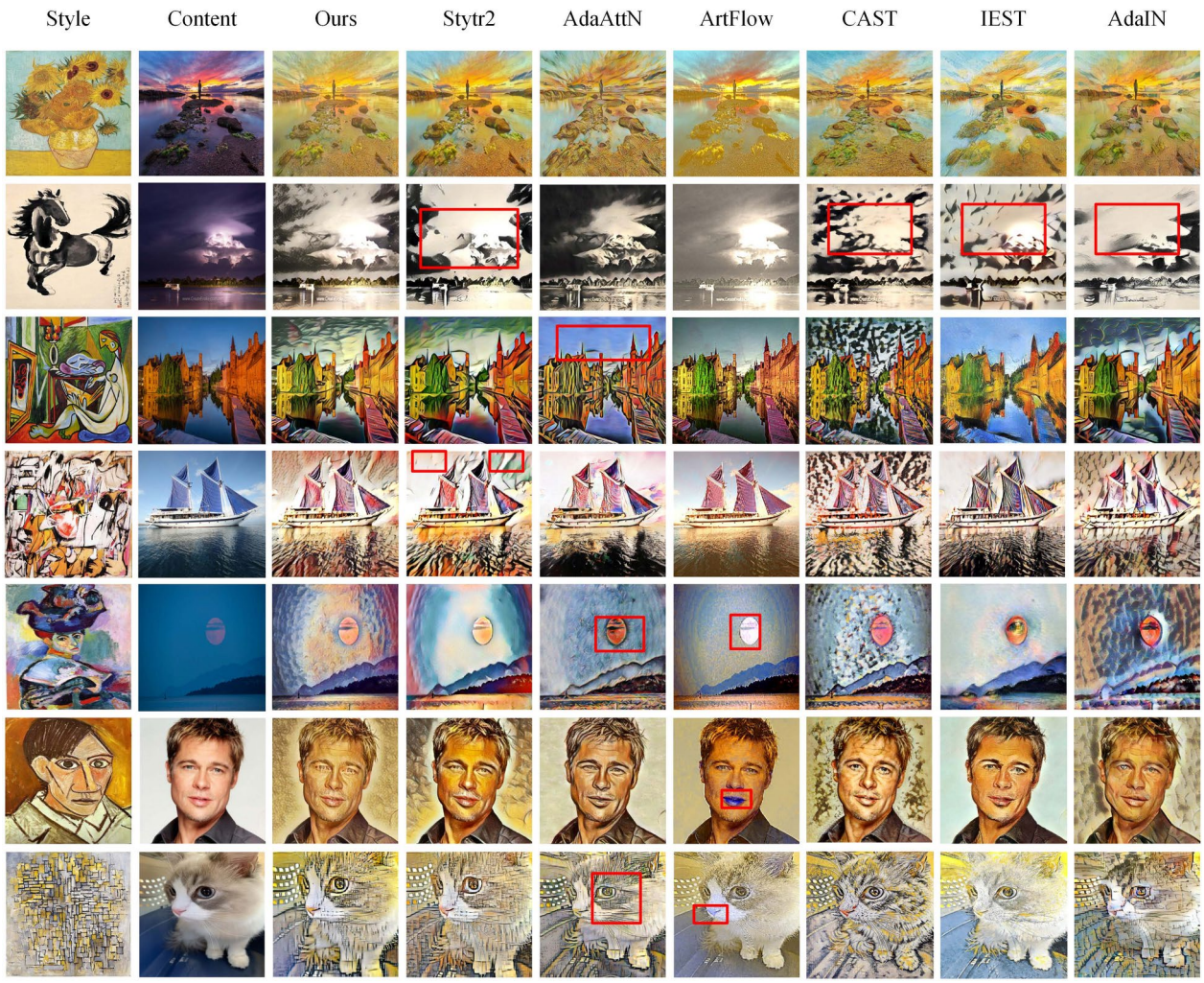


Fig. 6 Comparison of style transfer result with SOTA model

does so with significantly reduced computational delay compared to its peers.

b) LPIPs.

LPIPs [47] is utilized to quantify the content difference between the two images. The underlying principle of this approach involves establishing an inverse mapping from the generated image to the ground truth. Empirical evaluations demonstrate that LPIPs exhibit superior consistency with human perception compared to traditional methods such as L2/PSNR, SSIM, and FSIM. It is worth noting that a smaller LPIPS value indicates a higher degree of similarity between I_C and I_{CS}. The equation is shown below:

$$d(I_{cs}, I_c) = \sum_N \frac{1}{H_N W_N} \sum_{h,w} \| w_l \odot (\hat{y}^l_{hw} - \hat{y}^l_{0hw}) \|_2^2 \tag{17}$$

where w_l represents the learnable parameter which is used to activate channels. Four content images and four style images are randomly selected in succession to generate a set of four stylized images. ArtFlow is no longer involved in the comparison because of its preference for content with bare style features. The calculation results are shown in Table 2 below:

It can be observed that our model exhibits minimal content loss across all four styles, demonstrating its capability to preserve a greater amount of content structure information compared to other existing models during the

Table 1 Inference time comparison

Models	Inference Time/second	
	256*256	512*512
Stytr2	6.377	30.034
AdaAttN	5.043	5.475
ArtFlow	4.441	4.701
CAST	5.134	5.410
IEST	**4.181**	**4.249**
AdaIN	4.202	4.348
Ours	4.236	11.350

Table 2 LPIPs comparison

	I_{cs1}	I_{cs2}	I_{cs3}	I_{cs4}
stytr2	0.565	0.445	0.378	0.473
AdaAttN	0.597	0.532	0.452	0.535
IEST	0.519	0.648	0.649	0.642
CAST	0.673	0.501	0.474	0.606
AdaIN	0.650	0.530	0.464	0.527
Ours	**0.516**	**0.433**	**0.357**	**0.414**

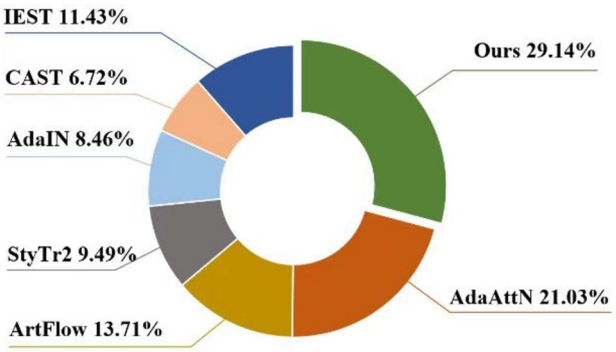


Fig. 7 Result of population

style transfer process. This performance not only reflects the model's advanced capabilities but also underscores its potential to deliver enhanced visual fidelity in diverse artistic transformations.

Both qualitative and quantitative results affirm the superiority of the CaVIT model over other current methods in the field of image style transfer. Its ability to preserve content integrity while effectively managing computational efficiency and stylistic fidelity makes it a formidable tool in artistic image transformation. This evaluation provides a robust framework for future research and potential enhancements in image style transfer technology.

c) User Study.

The evaluation of the generated stylized images is conducted through a questionnaire administered to participants of varying ages, occupations (including art-related practitioners and others), and genders. Our model is compared with StyTr2, AdaAttN, ArtFlow (AdaIN), CAST, IEST, and AdaIN. For each of the 7 models, 20 content images and 20 stylized images are randomly selected, resulting in a collection of 400 stylized images. From these, 10 stylized images are presented in each batch for participants to compare and select the best image. A total of five batches are performed. The specific results can be found in Fig. 7. Moreover, the paper also collects choices from art-related practitioners, with the results presented in Fig. 8 below.

It is concluded that our model achieves a higher preference percentage than the other models. In particular, art-related practitioners show greater preference for our model, with nearly half of the participants selecting it. This demonstrates that our model effectively transfers style features to content images, resulting in the generation of high-quality stylized images.

4.5 Ablation experiments

The network architecture is compared with other similar architectures (AdaIN and StyTr2), as well as with architectures lacking the proposed innovative modules. The same dataset is trained on the same device for a fixed number of epochs, specifically set to 10,000. Batch-size is set to 4, while other hyperparameters remain constant. Subsequently, the same tests are conducted on these architectures, and the generated stylized images are presented in Fig. 9 below.

4.6 Innovative architecture and modules

For the ablation comparison of the encoder-decoder architecture, CNN-based AdaIN and Transformer-based StyTr2 are selected. The outcomes of Stytr2 are displayed in column 3. It can be observed that the Stytr2 model exhibits deficiencies in learning style features, as evidenced by the lackluster style transfer effects (rows 1 and 2). Additionally, the stylized images generated by this model are prone to exhibit cracking (rows 4). Despite achieving better results after extensive training on a considerably large dataset, the model is characterized by slow convergence

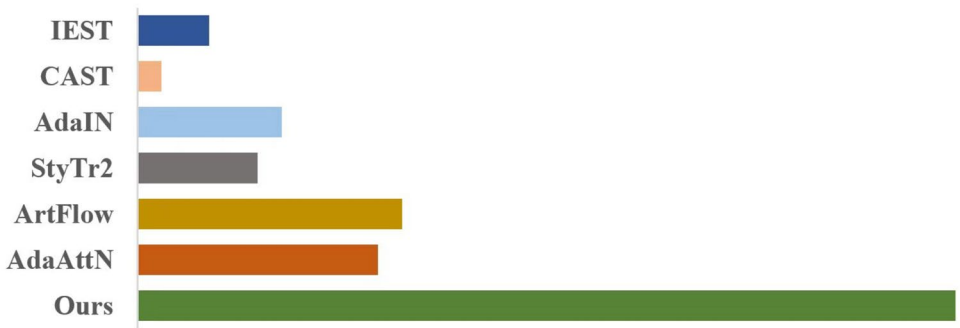


Fig. 8 Result of art-related practitioners

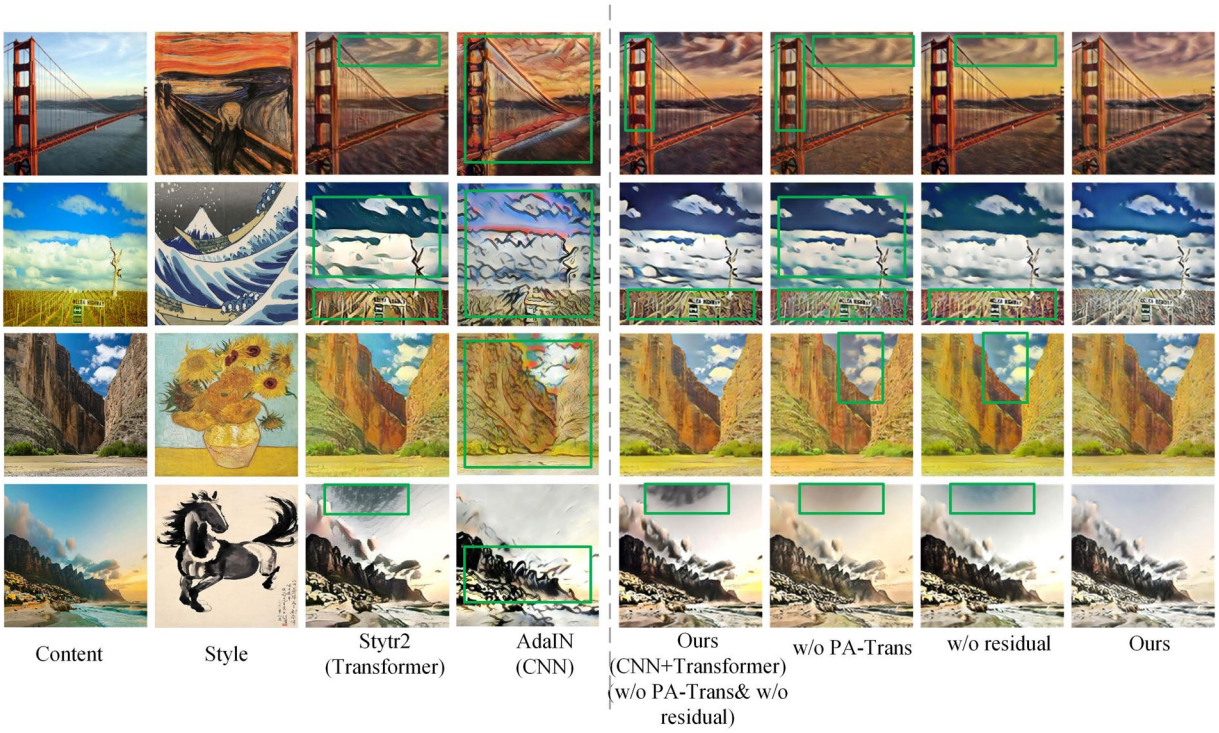


Fig. 9 Ablation experiments without proposed blocks

and high computational resource consumption, which are disadvantageous for practical applications. The outcomes of AdaIN, presented in column 4. The stylized images generated by this model have serious content structure destruction, accompanied by a large number of artifacts and cracks, which still exist even after sufficient training. In contrast, our architecture's outcomes, shown in column 5, effectively strike a balance between content and style features, leading to superior results even with a reduced number of training iterations.

4.7 Removing the PA-Trans encoder layer

In order to verify the effectiveness of the PA-Trans encoder layer, it is replaced by Transformer Encoder for comparative experiments. The experimental results are shown in column 6. It can be seen that the stylized images generated by Transformer Encoder have some defective features, such as color features that do not exist in the content image (rows 2 and 3), and the model's ability to capture texture features will also decrease (rows 1). Furthermore, the use of the PA-Trans encoder drastically reduces the training time from 1.5 h to 1.32 h, a noteworthy reduction of about 12% because of the half trainable parameters of the original Transformer model.

4.8 Remove residual blocks

In order to verify the effectiveness of the residual block, it is removed from the style feature extraction module for comparative experiments. The experimental results are shown in column 7. It can be seen that if the residual block is not added to the style feature extraction module, the migrated style features are insufficient and the style features of the original content image are easily retained (rows 4). Moreover, the generated stylized image lacks detailed information, such as the area shown in the green box (rows 1).

4.9 Remove the PA-Trans encoder layer and residual blocks

The stylized images generated without utilizing the PA-Trans Encoder and the VGG-19 Encoder with Residual Block are displayed in column 5 for reference. However,when comparing the stylized image produced by our final model (column 8), the images in column 5 are highly susceptible to artifacts (rows 1) and patches (rows 4), and the style transfer is incomplete (rows 3). Our model effectively combines the advantages of CNN encoding and Transformer encoding while introducing novel modules to enhance the entire model generation process. Overall, our method achieves superior performance in image style transfer through strategic additions and modifications.

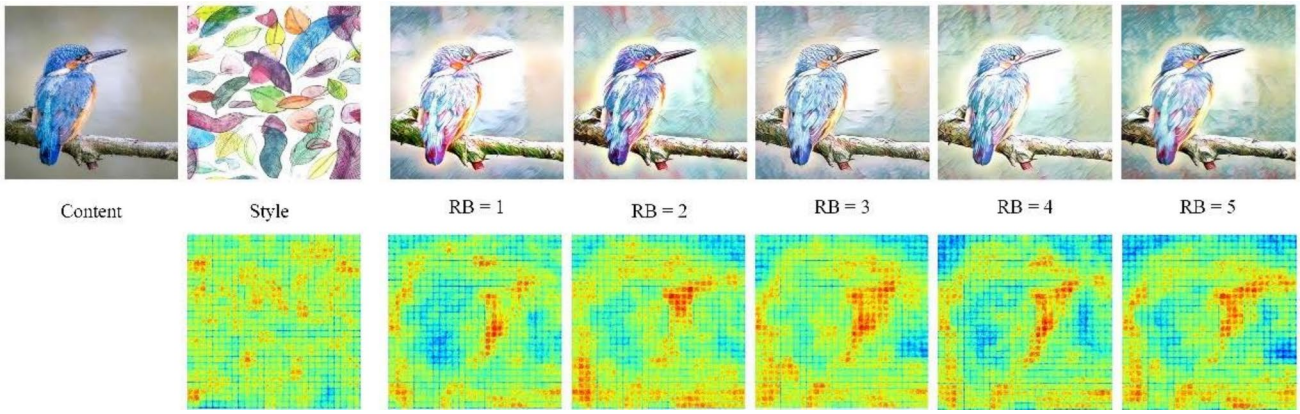


Fig. 10 Numbers of various Residual Blocks (RB)

4.10 Numbers of residual blocks

Choosing the appropriate number of residual blocks is crucial for the proposed model. To determine the optimal configuration, a series of experiments were conducted, varying the number of residual blocks from 1 to 5. Corresponding stylized images were then generated, and style features were visualized under identical experimental conditions. The specific image is shown in Fig. 10 below:

Insufficient residual blocks can lead to artifacts in the generated stylized images, as observed in columns 3 and 4. Conversely, an excessive number of residual blocks can burden the model without substantial improvement in effectiveness, as evident in columns 6 and 7. Furthermore, the visualization diagrams in the corresponding column further demonstrate that an excessive number of residual blocks hampers style consistency, leading to a notable presence of the blue color. Therefore, three residual blocks prove to the optimal choice. Style features can be effectively extracted and high-quality stylized images generated by this configuration, while maintaining the model's lightweight nature.

5 Conclusion

This paper presents CaVIT, an innovative network structure designed for image style transfer. It utilizes two parallel encoders to independently extract features, which are then fused in a sequential manner. The final stylized images are generated using a CNN Decoder. This paper introduce a novel Transformer Encoder called PA-Trans Encoder to extract content features, alongside a VGG-19 encoder with three residual blocks for more detailed style features. Our model ensures compatibility between different encoders and the same decoder, effectively addressing the content leakage problem and achieving superior stylizing outcomes with efficient image style transfer. In terms of inference time, there is minimal difference compared to state-of-the-art (SOTA) models for general resolution during the testing process. However, processing high-resolution images may slightly increase the time required. For the image style transfer task, while ensuring the integrity of feature extraction, future research should focus on optimizing the algorithm's reasoning speed, especially improving high-resolution image generation. In addition, the current quantitative evaluation system mainly focuses on content features, lacks effective evaluation indicators for style features, and the evaluation system is not comprehensive. In subsequent research, it is necessary to specifically reduce the model's computational complexity, reduce the model's reasoning time, and improve the model's performance. It is required to introduce style quantification indicators to more accurately evaluate the degree of style transfer.

Data Availability The data that support the findings of this study are publicly accessible at the official website address: https://cocodataset.org/#download and https://paperswithcode.com/dataset/wikiart.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this study.

References

1. Abed A, Akrout B, Amous I (2022) Semantic heads segmentation and counting in crowded retail environment with convolutional neural networks using top view depth images. SN Comp Sci 4(1):61
2. Abed A, Akrout B, Amous I (2024) Convolutional Neural Network for Head Segmentation and Counting in Crowded Retail Environment Using Top-view Depth Images. Arab J Sci Eng 49(3):3735–3749
3. Jing Y, Yang Y, Feng Z, Ye J, Yu Y, Song M (2019) Neural style transfer: A review. IEEE Trans Visualiz Comp Grap 26(11):3365–85

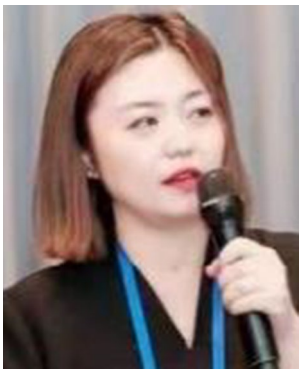
4. Wei LY, Levoy M (2000) Fast texture synthesis using tree-structured vector quantization. In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp 479–488
5. Gao W, Li Y, Yin Y, Yang MH (2020) Fast video multi-style transfer. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp 3222–3230
6. Johnson J, Alahi A, Fei-Fei L (2016) Perceptual losses for real-time style transfer and super-resolution. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, Springer International Publishing, pp 694–711
7. Chen D, Yuan L, Liao J, Yu N, Hua G (2017) Stylebank: an explicit representation for neural image style transfer. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1897–1906
8. Puy G, Pérez P (2019) A flexible convolutional solver for fast style transfers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 8963–8972
9. Dumoulin V, Shlens J, Kudlur M (2016) A learned representation for artistic style. arXiv preprint arXiv:1610.07629.
10. Ulyanov D, Lebedev V, Vedaldi A, Lempitsky V (2016) Texture networks: feed-forward synthesis of textures and stylized images. arXiv preprint arXiv:1603.03417.
11. Zhang H, Dana K (2018) Multi-style generative network for real-time transfer. In: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, pp 0-0
12. An J, Huang S, Song Y, Dou D, Liu W, Luo J (2021) Artflow: Unbiased image style transfer via reversible neural flows. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 862–871
13. Deng Y, Tang F, Dong W, Ma C, Pan X, Wang L, Xu C (2022) Stytr2: image style transfer with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11326–11336
14. Gatys LA, Ecker AS, Bethge M (2016) Image style transfer using convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2414–2423
15. Hendrycks D, Gimpel K (2016) Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415.
16. Huang X, Belongie S (2017) Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE international conference on computer vision, pp 1501–1510
17. Lu J, Barnes C, DiVerdi S, Finkelstein A (2013) Realbrush: Painting with examples of physical media. ACM Trans Grap (TOG) 32(4):1–12
18. Hertzmann A (1998) Painterly rendering with curved brush strokes of multiple sizes. In: Proceedings of the 25th annual conference on Computer graphics and interactive techniques, pp 453–460
19. Portilla J, Simoncelli EP (2000) A parametric texture model based on joint statistics of complex wavelet coefficients. Int J Comput Vision 40:49–70
20. Simonyan K (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
21. Risser E, Wilmot P, Barnes C (2017) Stable and controllable neural texture synthesis and style transfer using histogram losses. arXiv preprint arXiv:1701.08893.
22. Li Y, Wang N, Liu J, Hou X (2017) Demystifying neural style transfer. arXiv preprint arXiv:1701.01036.
23. Zhang Y, Tang F, Dong W, Huang H, Ma C, Lee TY, Xu C (2022) Domain enhanced arbitrary image style transfer via contrastive learning. In: ACM SIGGRAPH 2022 conference proceedings, pp 1–8
24. Park DY, Lee KH (2019) Arbitrary style transfer with style-attentional networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5880–5888
25. Jing Y, Liu X, Ding Y, Wang X, Ding E, Song M, Wen S (2020) Dynamic instance normalization for arbitrary style transfer. In: Proceedings of the AAAI conference on artificial intelligence, 34(04):4369–4376
26. Svoboda J, Anoosheh A, Osendorfer C, Masci J (2020) Two-stage peer-regularized feature recombination for arbitrary image style transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 13816–13825
27. Li X, Liu S, Kautz J, Yang MH (2019) Learning linear transformations for fast image and video style transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3809–3817
28. Yao Y, Ren J, Xie X, Liu W, Liu YJ, Wang J (2019) Attention-aware multi-stroke style transfer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 1467–1475
29. Liu S, Lin T, He D, Li F, Wang M, Li X, Sun Z, Li Q, Ding E (2021) Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 6649–6658
30. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2020) Generative adversarial networks. Commun ACM 63(11):139–144
31. Isola P, Zhu JY, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
32. Zhu JY, Park T, Isola P, Efros AA (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 2223–2232
33. Kotovenko D, Sanakoyeu A, Ma P, Lang S, Ommer B (2019) A content transformation block for image style transfer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 10032–10041
34. Liu MY, Huang X, Mallya A, Karras T, Aila T, Lehtinen J, Kautz J (2019) Few-shot unsupervised image-to-image translation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10551–10560
35. Cao Y, Chandrasekar A, Radhika T, Vijayakumar V (2024) Input-to-state stability of stochastic Markovian jump genetic regulatory networks. Mathemat Comput Simulat. 222:174–87
36. Radhika T, Chandrasekar A, Vijayakumar V et al (2023) Analysis of Markovian jump stochastic Cohen-Grossberg BAM neural networks with time delays for exponential input-to-state stability. Neural Proc Lett 55(8):11055–11072
37. Dosovitskiy A (2020) An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929
38. Zhu X, Su W, Lu L, Li B, Wang X, and Dai J (2020) Deformable DETR: deformable transformers for end-to-end object detection. arXiv preprint arXiv: 2010.04159
39. Xie E, Wang W, Yu Z, Anandkumar A, Alvarez JM, Luo P (2021) SegFormer: Simple and efficient design for semantic segmentation with transformers. Adv Neural Inf Process Syst 34:12077–12090
40. Wu X, Hu Z, Sheng L, Xu D (2021) Styleformer: real-time arbitrary style transfer via parametric style composition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 14618–14627
41. Wang B, Komatsuzaki A (2021) GPT-J-6B: a 6 billion parameter autoregressive language model. URL https://github. com/kingoflolz/mesh-transformer-jax
42. Chowdhery A, Narang S, Devlin J, Bosma M, Mishra G, Roberts A, Barham P, Chung HW, Sutton C, Gehrmann S, Schuh P (2023) Palm: Scaling language modeling with pathways. J Mach Learn Res 24(240):1–13
43. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, Proceedings, Part V 13 2014, Springer International Publishing, pp 740–755
44. Phillips F, Mackintosh B (2011) Wiki Art Gallery Inc: A case for critical thinking. Issues Account Educ 26(3):593–608

45. Deng Y, Tang F, Dong W, Ma C, Huang F, Deussen O, Xu C (2020) Exploring the representativity of art paintings. IEEE Trans Multimedia 23:2794–2805
46. Chen H, Wang Z, Zhang H, Zuo Z, Li A, Xing W, Lu D (2021) Artistic style transfer with internal-external learning and contrastive learning. Adv Neural Inf Process Syst 34:26561–26573
47. Zhang R, Isola P, Efros AA, Shechtman E, Wang O (2018) The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 586–595

ZaiFang Zhang holds a Doctor of Philosophy degree in Industrial Engineering from Shanghai Jiao Tong University. Currently, he serves as the Deputy Director of the Department of Mechanical Automation Engineering in the School of Mechatronic Engineering and Automation at Shanghai University. He is also a Standing Committee Member of the Digital Twin Special Committee of the China Computer Graphics Society, a Council Member of the Shanghai Design Law Society, a Committee Member of the Group Technology and Intelligent Integration Technology Branch of the Chinese Mechanical Engineering Society, and a Member of the Advisory Committee of the international journal "Digital Twins". His research interests mainly focus on intelligent design of product-service systems, digital twins, intelligent learning theories and methods as well as their applications, and the integration of medicine and engineering.

ShunLu Lu is enrolled in the School of Mechatronic Engineering and Automation at Shanghai University. Her main research directions are image style transfer and few-shot image enhancement.

Qing Guo is enrolled in the School of Mechatronic Engineering and Automation at Shanghai University, with a primary research focus on artificial intelligence and multimodal emotion recognition.



Nan Gao a doctoral supervisor in the School of Sciences at Shanghai University, specializes in algebraic representation theory, triangulated categories, derived categories, and Gorenstein homological algebra. She is a distinguished "Oriental Scholar" professor in Shanghai. Additionally, she serves as a member of the organizing committee of the 21st International Conference on Representation Theory of Algebras (ICRA), the deputy secretary-general of the Fifth Committee of the Educational Mathematics Specialty Committee of the China Association of Higher Education, a reviewer for Mathematical Reviews in the United States, and an editorial board member of the journal "Advances in International Applied Mathematics".

YuXiao Yang is enrolled in the School of Science at Shanghai University, with a primary research focus on representation theory of algebra and related applications.