Revisiting and Refining AdaIN: Multi-Level Style Transfer and Attention Module Integration

Chengzhan Gao
Department of Electrical and Computer Engineering
University of California, San Diego
San Diego, CA, 92037
`gchengzhan@ucsd.edu`

Abstract

In this project, I revisited the Adaptive Instance Normalization (AdaIN) style transfer (2017), a significant advancement in the domain of style transfer, that efficiently transforms the aesthetics of an image while maintaining its original content. The AdaIN model, notable for its high-quality real-time multi-style transfers, was not only reimplemented but was also improved upon. I introduced multi-level style transfer, applying the AdaIn module to various encoder levels instead of exclusively at the last level. The updated model generated more holistic and detailed stylized images, better retaining the original content. Further enhancement was made by replacing the AdaIn module with an attention module, improving the model's capacity to stylize images beyond color, also considering other features like texture and shape. Despite the enhancements, more work is needed to fully realize this potential. Overall, this project emphasizes the continued exploration and refinement in the field of style transfer, aiming to push the boundaries of this exciting field and broaden the adaptability of such models.

1 Introduction

This report delves into the adaptation and enhancement of the Adaptive Instance Normalization (AdaIN) style transfer model (2017). Style transfer, a crucial domain within computer vision, facilitates the transformation of an image's style while conserving its essential content. It finds multifarious applications across artistic image synthesis, video editing, fashion design, and beyond. This project navigates through the reimplementation and improvement of the AdaIN model, extending its applicability to other datasets, and unlocking new facets of style transfer.

Despite the profound advancements in style transfer methodologies, certain limitations persist. Initial techniques like the Neural Style Transfer by Gatys et al. (2015) exhibited captivating visual outcomes, yet they grappled with slow processing speeds due to iterative optimization. Other models such as Fast Style Transfer by Johnson et al. (2016) and Texture Networks by Ulyanov et al. (2017) expedited style transfers, albeit being confined to a single style per model. The advent of AdaIN (2017) brought about a considerable breakthrough in this realm, enabling real-time multi-style transfers and generating high-quality stylized results.

In an attempt to push the frontiers of style transfer, this project embarked on two primary endeavors. Firstly, I sought to enhance the original AdaIN model, and secondly, to widen its applicability to other datasets. The enhancements revolved around introducing multi-level style transfer to the model, wherein the AdaIn module was applied at various encoder levels, as opposed to only the last level. This modification led to a more integrated and detailed rendering of stylized images, thereby preserving the original content more effectively.

As a further improvement, the AdaIn module was replaced with an attention module, thereby expanding the model's capability to stylize images on multiple dimensions - not just color, but also considering other aspects such as texture, shape, and artistic features like brushstroke style. While these enhancements mark substantial progress, there still exist areas of potential refinement to maximize the model's capabilities fully.

This report charts the journey through the exploration and augmentation of the AdaIn style transfer model. It provides an in-depth analysis of the original model, outlines the proposed improvements, presents results from the enhanced model, and suggests areas for future work. I believe that the discoveries and improvements presented herein can serve as stepping stones to further innovations in the field of style transfer. I hope this project will inspire further research and development, thus pushing the boundaries of what is possible in this fascinating area of study.

Key contributions of this project include:

- The implementation and enhancement of the Adaptive Instance Normalization (AdaIN) style transfer model, a critical domain within computer vision, extending its applicability to other datasets, and unlocking new facets of style transfer.
- Introduction of a multi-level style transfer to the model, wherein the AdaIn module was applied at various encoder levels, enabling a more detailed and integrated rendering of stylized images while effectively preserving original content.
- Replacement of the AdaIn module with an attention module, enhancing the model's capacity to stylize images not only on color but also considering aspects such as texture, shape, and artistic features like brushstroke style. This change allowed for a more nuanced style representation and a precise matching process between the content and style features.
- An in-depth comparative analysis of the original model and its enhanced version, demonstrating improved image quality with enriched detail preservation and more accurate style transfer. Despite the additional computational demands of the enhanced model, the trade-off is well justified by the improved quality of style transfer. Future work could focus on reducing this computational burden without compromising the model's performance.

2 Background and Related Work

Style transfer is a compelling domain within computer vision that has seen significant advancements over the years. The primary focus has been on the development of techniques that can efficiently transform the aesthetic of an image while preserving its content. This pursuit, however, has been marked by several challenges, notably processing speed and multi-style applicability, which have necessitated a series of evolutionary enhancements in the models deployed.

The foundations of style transfer were laid down by Neural Style Transfer by Gatys et al. (2015). Despite its ability to generate visually stunning results, the model was encumbered with slow processing times, a consequence of its reliance on iterative optimization. To counter this limitation, subsequent models like Fast Style Transfer by Johnson et al. (2016) and Texture Networks by Ulyanov et al. (2017) emerged, which leveraged feed-forward neural networks for faster style transfer. While successful in speeding up the process, these models were constrained to a single style per model, thereby limiting their flexibility and overall applicability.

The potential of generative adversarial networks (GANs) in style transfer has also been explored by researchers like Li and Wand (2016) and Chen and Schmidt (2016). GANs, known for their proficiency in creating realistic images by effectively modeling the distribution of input data, brought new possibilities. However, they also introduced challenges such as high computational requirements, difficulties in training, and sensitivity to hyperparameters and the quality of training data.

The introduction of the Adaptive Instance Normalization (AdaIN) model (2017) marked a significant milestone in the field. AdaIN, capable of real-time multi-style transfers, produced high-quality stylized results. This was achieved by aligning the mean and variance of the content features with those of the style features, allowing the model to generalize to arbitrary styles. Despite its success, AdaIN was found to be susceptible to overfitting to the style image and occasionally produced inconsistent results across diverse content images.

Further advancements in the field have led to the development of methods like Markov Random Field (MRF) based approaches (2016). These methodologies offer an intersection of neural and texture-based style transfers and present a promising perspective on multi-style transfer. However, like their predecessors, they also pose challenges, primarily in terms of computational intensity and optimization difficulties.

In summary, the journey of style transfer has been a landscape of various techniques, each carrying unique strengths and shortcomings. Even though substantial progress has been made, the path for refinement and innovation continues to unfold. This project endeavors to address this need, specifically by mitigating the limitations of the AdaIN model and broadening its applicability. The ultimate aim is to continue pushing the boundaries of style transfer and widen the horizon of possibilities in this fascinating field

3 Analysis of the AdaIN Model

The Adaptive Instance Normalization (AdaIN) model, developed by Huang and Belongie (2017), represents a significant advancement in style transfer techniques, especially with its capability to perform real-time, multi-style transfers. However, to effectively enhance and extend the model, it is crucial to understand its workings in detail.

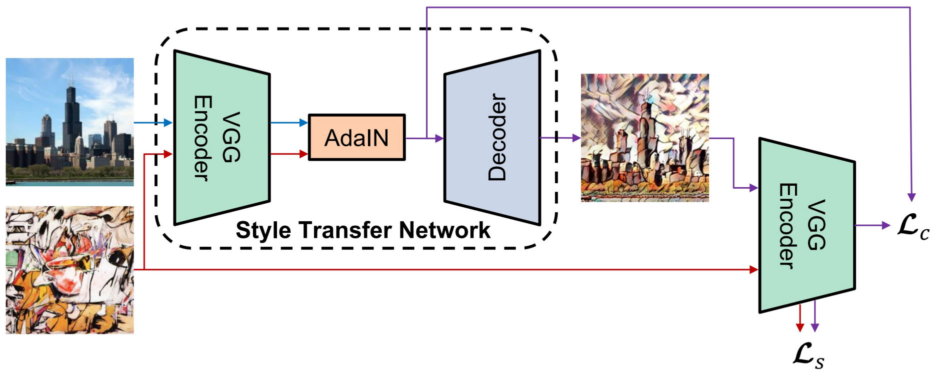


Figure 1: Architecture of Adain Model

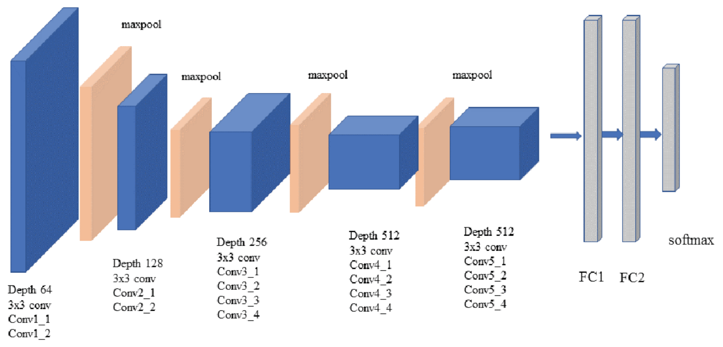


Figure 2: Architecture of VGG Model

3.1 Model Architecture

As shown in figure 1, AdaIN model is primarily composed of three key components: a content encoder, a style encoder, and a decoder.

Content and Style Encoder: The content and style encoders are based on a pre-trained VGG19 network(Figure 2). The network is pre-trained on the ImageNet dataset to learn a wide variety of feature representations. This VGG19 model is stripped of all the fully connected layers, retaining only the convolutional layers to serve as encoders.

The content encoder works by extracting features from the 'relu4_1' layer of the VGG19 network. This layer is deep within the network and is therefore adept at capturing high-level content information. The features derived from this layer preserve the overall structure and semantics of the image, crucial for content representation, while discarding details that are irrelevant to the content, such as color, texture, and style.

Conversely, the style encoder operates across multiple layers, from 'relu1_1' to 'relu5_1'. This approach captures style information at different scales, a necessity given the abstract nature of 'style', which encompasses elements like color, texture, and brush strokes. Unlike content, these stylistic elements do not depend on the spatial layout of the image. Instead, they rely on correlations within and across feature maps. Thus, to capture style information at varying scales and levels of complexity, the encoder leverages features from multiple layers. This strategy results in a richer, more comprehensive style representation that encapsulates stylistic elements at different scales and abstraction levels.

Overall, the choice of layers for content and style extraction reflects the understanding of the hierarchical organization of visual information in CNNs. Lower layers capture simple, local features while higher layers capture more complex, global features, effectively allowing us to differentiate between content and style.

AdaIN Layer: The AdaIN layer stands as a pivotal component within the architecture, taking as input the features derived from both the content and style encoders. The primary function of the AdaIN layer is to align the mean and variance of the content features with those of the style features, thereby facilitating an effective style transfer onto the content. This alignment process is realized through the implementation of the AdaIN equation:

$$AdaIN(x, y) = \sigma(y) \times \frac{(x - \mu(x))}{\sigma(x)} + \mu(y) \tag{1}$$

where x and y represent the content and style features respectively, and μ and σ denote the mean and standard deviation.

This function is derived from the concept of Instance Normalization, a technique initially used in style transfer to remove instance-specific contrast information from the content images. It works by standardizing (i.e., making the mean zero and standard deviation one) the features of each individual instance (image) in a batch. The AdaIN function takes this a step further by not only standardizing the features but also rescaling and shifting them to match the style. This is done by using the mean and standard deviation of the style features. Let's break down the equation for further analysis:

- $\mu(x)$ and $\sigma(x)$ are the mean and standard deviation of the content features (x), respectively. The expression $(x - \mu(x))/\sigma(x)$ standardizes the content features to make it like a standard distribution so as to be prepared to further transfer.
- $\mu(y)$ and $\sigma(y)$ are the mean and standard deviation of the style features (y), respectively.
- The standardized content features are then rescaled by multiplying them with $\sigma(y)$, which matches the standard deviation of the style features. This aligns the content's feature distribution with that of the style, transferring the style's color and texture patterns onto the content.
- Finally, the rescaled features are shifted by adding $\mu(y)$, which matches the mean of the style features. This ensures that the stylized content has the same mean intensity as the style, which affects the overall brightness and color tone of the transferred style.

The underlying theory is that the statistics (mean and standard deviation) of the style features encapsulate the essential stylistic information. By aligning the statistics of the content features with those of the style, the AdaIN function effectively transfers the style onto the content while retaining the content's structure. This is made possible by the hierarchical nature of CNNs, where lower layers capture local patterns (texture, color, etc.) and higher layers capture global patterns (objects, layout, etc.). Thus, modifying lower-level features changes the style without affecting the content, which is encoded in the higher-level features.

Decoder: Following the AdaIN layer, a decoder network is used to transform the stylized features back into an image. This decoder is designed to mirror the architecture of the VGG19 encoder, but in reverse. It's trained to minimize the difference between the features of the original and reconstructed image when passed through the VGG19 encoder.

3.2 Loss Function

The loss function plays a critical role in the training of the AdaIN model, guiding the optimization process to achieve high-quality style transfer results. The overall loss function in the AdaIN model is a combination of two separate losses: content loss and style loss (denoted as L_c and L_s in the Figure 1).

$$L_{total} = L_{content} + \lambda L_{style} \tag{2}$$

where L_{total} denotes the total loss, $L_{content}$ stands for the content loss, L_{style} represents the style loss, and λ is a weighting factor that balances the contribution of the content and style loss to the total loss.

Content Loss: The content loss quantifies the Euclidean distance between the target features and the features of the output image, thereby motivating the output image to maintain the content of the input image. The AdaIN output, t, serves as the target features in this context. The content loss is defined as the Mean Squared Error (MSE) between the AdaIN output t and the final output $f(g(t))$.

$$L_{content} = ||f(g(t))) - t||_2 \tag{3}$$

where $f(\cdot)$,$g(\cdot)$ are the encoder and decoder, t is the Adain output.

Style Loss: The style loss is determined as the Mean Squared Error (MSE) between the style statistics (mean and standard deviation) of the stylized (output) image and the style statistics of the style image. The deployment of the Gram matrix is omitted here because the AdaIN module only transfers the mean and standard deviation of the style features, and the inclusion or exclusion of the Gram matrix does not significantly impact the outcome. The style loss ensures the output image aptly embodies the style of the input style image.

$$L_{style} = \Sigma_{i=1}^{L} ||\mu(\phi_i(g(t))) - \mu(\phi_i(s)||_2 + \Sigma_{i=1}^{L} ||\sigma(\phi_i(g(t))) - \sigma(\phi_i(s)||_2 \tag{4}$$

where $phi_i(\cdot)$ denotes the i^{th} layer in VGG-19 utilized to compute the style loss, μ and σ symbolize the mean and standard deviation, respectively, and s stands for the original style image.

In summary, the total loss ensures that the stylized output image retains the content of the content image and matches the style of the style image. The trade-off between the two is controlled by the weighting factor λ, which can be tuned based on whether more emphasis is to be put on content preservation or style transfer. The AdaIN model is trained by minimizing this total loss, thereby learning to generate images that simultaneously match the content and style targets. The dual nature of the loss function is a reflection of the model's goal: to generate an image that retains the content of one input image while adopting the style of another.

3.3 Limitations

Despite its simplicity, this architecture provides a powerful mechanism for style transfer. However, it isn't devoid of certain drawbacks. One notable limitation is the model's proclivity to overfit to the style image. This overfitting can be traced back to the AdaIN layer's reliance on global statistics, which might not capture the style's intricate details in every instance. Further, an aggressive alignment of content features to style features can occasionally result in the obfuscation of important content details. These issues outline the areas of potential enhancement for this project, underscoring the need for refinement to further optimize the model's performance.

4 Model Enhancement

This project primarily concentrates on the enhancement of the AdaIn style transfer model through two fundamental alterations: the integration of multi-level style transfer and the replacement of the

AdaIn Module with a style-attentional Module. While the conventional AdaIN model employs the COCO Dataset for content images and the Wikiart Dataset for style images, the focus of this project lies in broadening the model's adaptability through the incorporation of new datasets for evaluation. To this end, the project makes use of the Flickr Image Dataset for content images and the Painter by Numbers Dataset for style images.

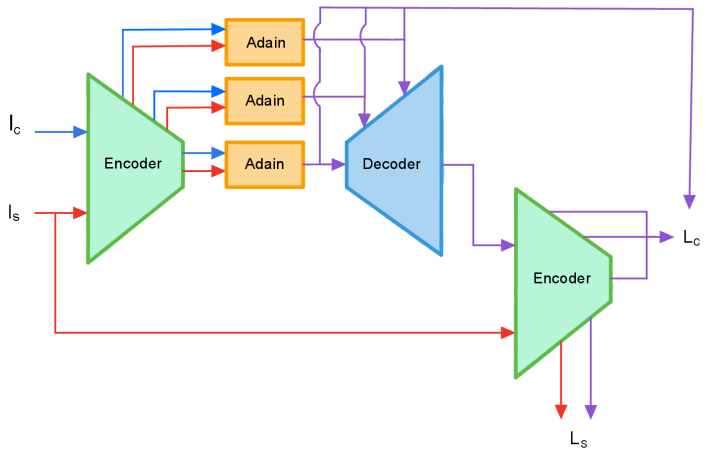


Figure 3: Multi-level AdaIn Style Transfer Architecture

In this figure, I_c and I_S denote the content and style images respectively, while L_C and L_S symbolize the corresponding content and style losses.

4.1 Multi-level Style Transfer

Departing from the traditional single-level approach, the proposed enhancement aims to implement a multi-level style transfer (Figure 3). This approach allows for a more nuanced and enriched stylization process, resulting in superior output images.

The proposed approach exploits the encoder's ability to extract hierarchical features at various levels, with each level capturing unique style information at different scales and complexities. Incorporating features from multiple encoder layers (specifically $'relu2\_1'$ to $'relu4\_1'$ in the implementation) deepens the richness of the style transfer by capturing stylistic elements at varying scales and abstraction levels. However, this method demands greater computational resources due to its comprehensive style representation.

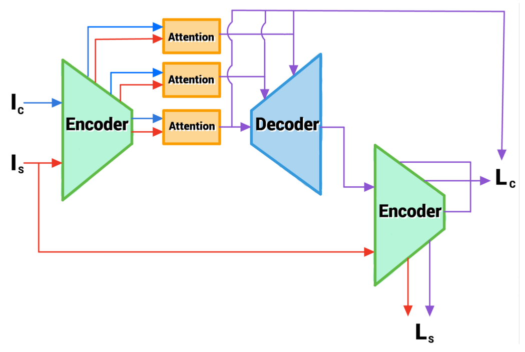
4.2 Loss Function Modification

Transitioning from single-level to multi-level style transfer necessitates an adaptation in the structure of the loss function. Given the AdaIN model's tendency to overfit and compromise content integrity, and the need to accommodate the multi-level architecture, I significantly modify the content loss function while the overall and style loss functions remain consistent:

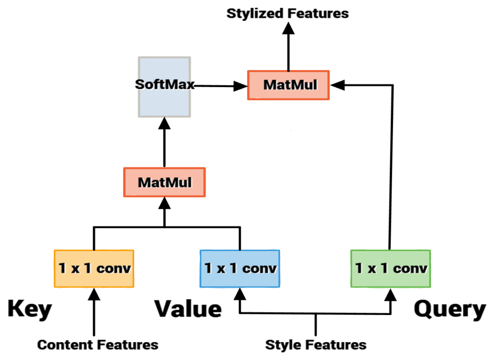
$$L_{content} = \Sigma_{i=2}^{L} ||\phi^i(r) - Adain(\phi^i(c), \phi^i(s))||_2 \tag{5}$$

In the equation above, $\phi_i(\cdot)$ continues to signify the i^{th} layer in the VGG-19 model, while r, c, and s denote the output image post-decoder, the content image, and the style image, respectively.

The content loss now calculates the sum of the Mean Squared Errors (MSE) between the stylized image's feature map and the AdaIN function's transformed results at each level. This updated content loss function ensures consistency between the stylized image's feature map and the AdaIN transformation outputs, with the aim to preserve the original image's content while successfully integrating the multi-level style characteristics. This function takes into account the differences between the content feature maps of the target and output images at multiple levels, thus improving the original image's content preservation quality. Further enhancement could involve adding a weighting coefficient to the MSE at each level, although this might increase the complexity of hyperparameter tuning, it could potentially yield better results.



(a) Model Architecture



(b) Style-Attentional Module

Figure 4: Multi-level Style-Attentional Style Transfer

4.3 Style Attentional Module

To take the proposed enhancements a step further, the AdaIn module was replaced with a more nuanced Style-Attentional Module(Park and Lee [2019]) as depicted in Figure 4a. This alteration addresses a notable limitation of the AdaIn module, which simplifies the transformation process by aligning only the global statistics of the style and content images. While this simplification enables efficient style transfer, it occasionally overlooks intricate style details that could be essential for capturing the authentic essence of the style image. This potential oversight can cause the style transfer to be superficial and even lead to overfitting.

To circumvent this shortcoming, the Style-Attentional Module employs a self-attention mechanism that formulates a more detailed style representation and a precise matching process between the content and style features (Figure 4b). This method enhances the ability of the model to capture and transfer nuanced stylistic details.

The self-attention mechanism weighs the importance of different style features based on their relevance to the content features. In essence, the module identifies which style features are more important or relevant in the context of the given content image. These important features are then emphasized during the style transfer process, ensuring a more faithful and nuanced rendering of the style.

The Style-Attentional Module processes the content and style features separately through two distinct attention mechanisms — one for spatial attention and the other for channel attention. These separate outputs are then combined to generate the final stylized features. The attention mechanisms are implemented as convolutions with trainable weights, which are optimized during the training process to best match the content and style features.

The operation of the module can be represented as follows:

$$S_{out} = h(F_S) \otimes (Softmax(f(Norm(F_C))^T \otimes g(Norm(F_S))))^T \tag{6}$$

In this equation, F_c and F_s denote the content and style features, while f, g, and h are 1×1 learnable convolution layers, Norm denotes channel-wise mean-variance normalization, respectively. $\otimes$ denotes element-wise multiplication.

By enabling the model to adapt the content and style features in a more attentive manner, the Style-Attentional Module leads to a more detailed and accurate style transfer. This module can capture the unique attributes of the style image better and simultaneously preserve the integrity of the content. It is important to note, however, that the integration of attention mechanisms increases the complexity and computational demands of the model. But given the enhanced quality of style transfer, this trade-off seems well justified. Future work could explore ways to reduce this computational burden without compromising the model's performance.

5 Results and Discussion

This section delineates the outcomes from the reimplementation of the original AdaIN model and juxtaposes the enhanced model against the original one.

5.1 Alpha

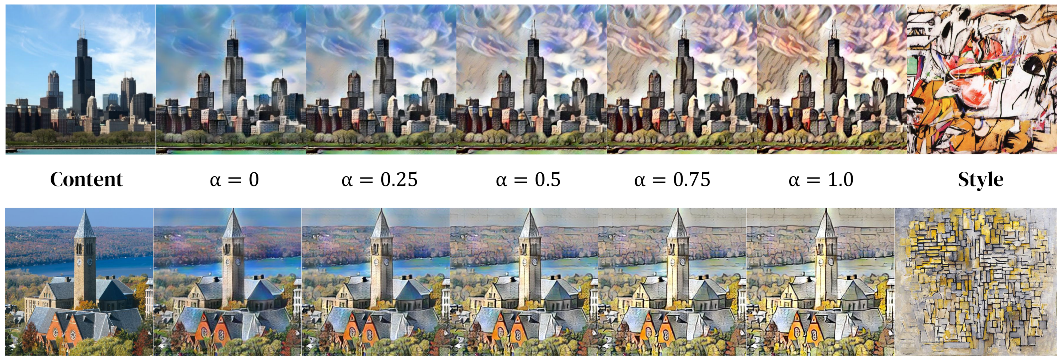


Figure 5: Content–style trade-off

The AdaIN model utilizes a parameter, termed as 'alpha', which dictates the degree of stylization during training. As evident from Figure 5, the stylization intensity of the output image escalates as α elevates from 0 to 1. This variable, therefore, stands crucial in managing the level of style infusion within the content image.

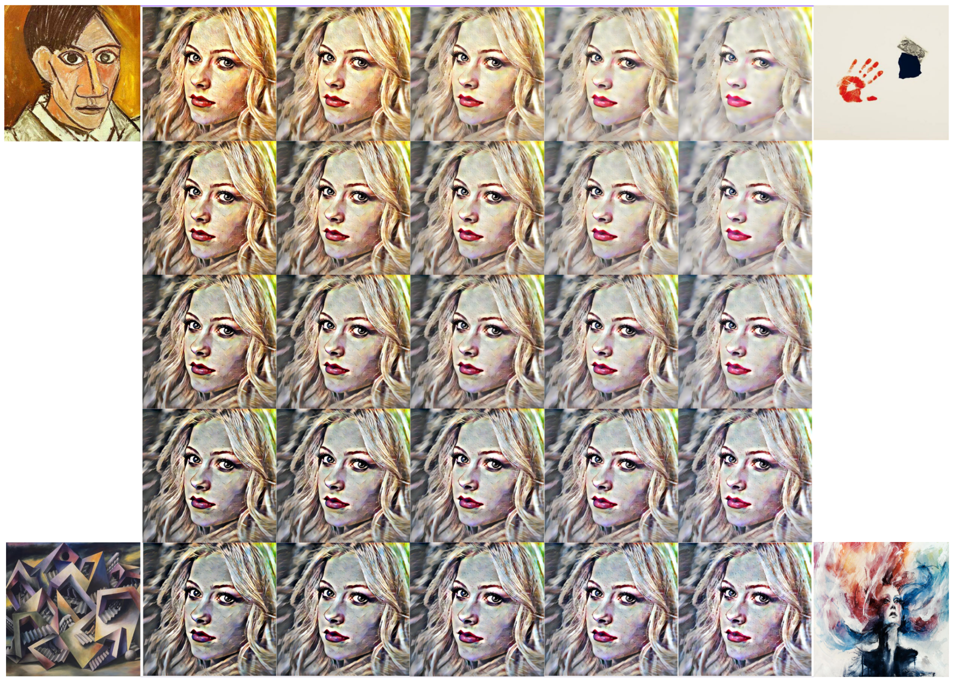


Figure 6: Style interpolation with four different styles

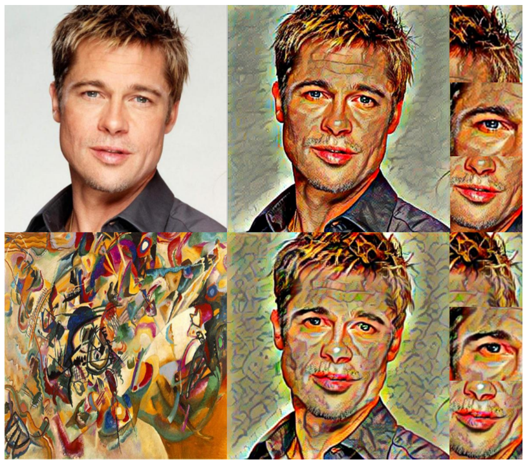
5.2 Style interpolation

The AdaIN model can assimilate multiple styles simultaneously, as depicted in Figure 6. Different style inputs are individually processed with the content image through the AdaIN module. The resultant stylized features are then normalized by the corresponding weights and aggregated. The consolidated feature set finally passes through the decoder to yield the output image.

5.3 Multi-level Style Transfer

Incorporating the AdaIN module at multiple stages of the VGG encoder, the enhanced model demonstrates a marked improvement over its predecessor. A conspicuous divergence between the multi-level and single-layer transfer models lies in their ability to retain the original content. The multi-level style transfer approach maintains the content image in superior detail and refinement, which is evident from Figure8. This enhanced content preservation is credited to the fact that in a multi-level setup, even the lower features of the encoder, encapsulating the original content information, are subjected to style transfer.

The whole-scale comfort and nuanced detail preservation of the enhanced model become apparent upon comparison with the original one, as shown in Figure7. This is further corroborated by the portrait example (Figure 7a), where the lip, hair, and mustache details are well conserved, and the landscape example (Figure 7b), where the bridge and mountain specifics are well preserved. While this detailed retention in most cases is advantageous, it might hamper the artistic style for users seeking a more immersive stylization matching the brushstroke of the style image, as evident in Figure 7b.


(a)

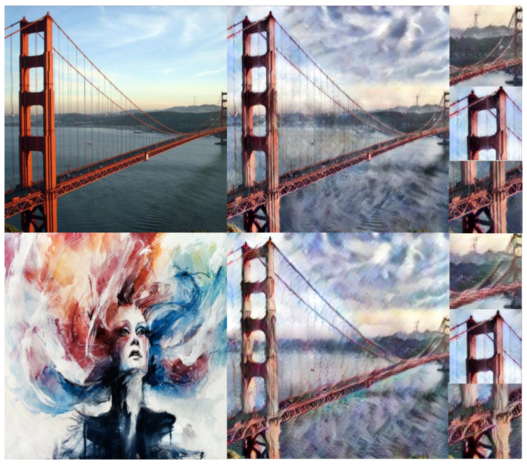

(b)

Figure 7: Comparison between single and multi-level Style Transfer

Upper-left: content image, bottom-left: style image, upper-right: stylized image by multi-level style transfer, bottom-right: stylized image by single-level style transfer, right column: details of the stylized images

5.4 Style-Attentional Module

Substituting the AdaIN Module with the Style-Attentional Module enriches the resultant stylized image by incorporating not only the color cues but also additional aspects such as texture, shape, and brushstroke style from the style input(Figure 9). This advancement is well illustrated in the first column, where the splashed-ink style of the input is effectively imitated by the enhanced model. In the fourth and fifth columns, the stylized outputs of the enhanced model, unlike its counterparts, adeptly recreate the straight lines and circles from their respective style inputs. Similarly, in the last column, the style of the enhanced model's output closely resembles the painting style of the input. The superior performance of the Style-Attentional module underscores its effectiveness in capturing and transposing nuanced stylistic details.

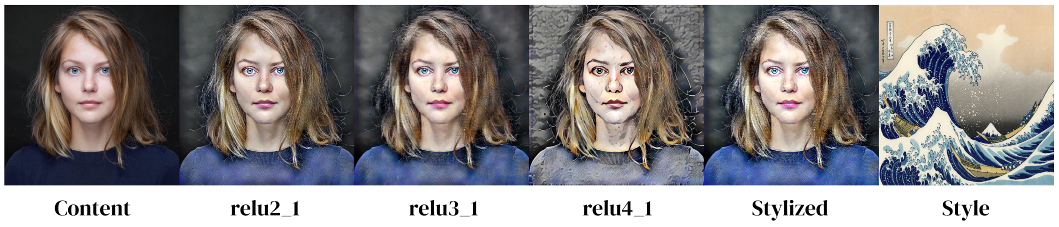


Figure 8: Multi-level Style Transfer

$relu2\_1, relu3\_1, relu4\_1$ are the output of three attention modules, from lower feature to higher feature. Stylized is the final output.

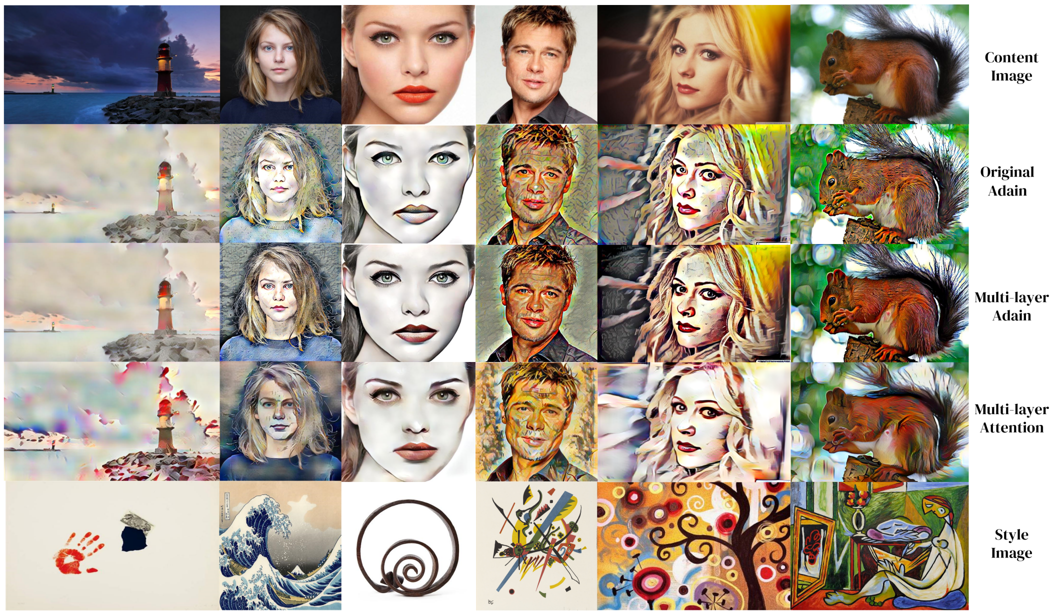


Figure 9: Comparison Of three Models

6 Conclusion

This project constituted a comprehensive exploration and enhancement of the AdaIN style transfer model, a cornerstone of computer vision. The primary aim was to augment the model's capabilities and extend its applications across different datasets. The project was successful in implementing the proposed modifications, leading to improved style transfer performance.

An innovative approach was the introduction of multi-level style transfer, where the AdaIN module was applied at various encoder levels, yielding a more integrated and detailed rendering of stylized images. This approach preserves the original content more effectively compared to single-level style transfers, but the increase in computational resources needed for this more comprehensive style representation is a downside. It would be beneficial for future research to seek methods that optimize this resource demand.

To provide an even more nuanced style transfer, the AdaIN module was replaced with a Style-Attentional Module. The Style-Attentional Module employs a self-attention mechanism for a more detailed style representation and precise matching process between content and style features. This modification enabled a detailed and accurate style transfer, capturing the unique attributes of the style image while simultaneously preserving the integrity of the content.

The successful implementation of these enhancements significantly improved the quality of the output images, which was clearly demonstrated in various examples. This notwithstanding, it is important to note that while the enhanced model's detailed content retention is beneficial in many cases, it might inhibit the artistic style for users seeking a more immersive stylization.

Overall, the modifications made to the AdaIN style transfer model in this project highlight the potential for future innovations in this field. The improvements made, while significant, represent just a fraction of what is possible. Future work could focus on reducing the computational burden without compromising the model's performance, and possibly fine-tuning the balance between content preservation and stylization. It is hoped that the work done here can serve as a stepping stone for further advancements, pushing the boundaries of what is possible in the fascinating realm of style transfer.

References

Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 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*, pages 694–711. Springer, 2016.

Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2479–2486, 2016.

Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5880–5888, 2019.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6924–6932, 2017.

Appendix

The code of this project is available in this Github repository.