Inversion based Style Transfer with Diffusion Models

Mukka Koushik (210101069) **Posa Mokshith** (210101077) **Rangu Rishvanja Simha** (210101084)
Nitish Kumar Pinneti (210101125) **Tammireddy Sri Vallabh** (210101126)

September 6, 2024

---

1 Introduction

Style Transfer is a deep learning technique that merges the content of one image with the artistic style of another (style image). Traditional methods use Convolutional Neural Networks (CNNs) to separate the content (shapes, objects) from the style (colors, textures, patterns) of images, allowing for the creation of a new image that retains the structure of the original but appears in the style of a second style image. This technique has a range of real-world applications...

In artistic image generation, style transfer allows users to transform photos into artistic works by applying styles of famous paintings. It's used in photo and video editing to stylize personal content with unique filters. In augmented reality (AR), style transfer helps in transforming live video feeds with artistic effects in real-time. In fields like fashion and design, it aids in creating novel patterns and designs for textiles or interior decoration. Additionally, advertising and marketing benefit from style transfer by producing visually engaging, stylized content. The film industry also uses it to apply artistic effects to scenes, creating distinctive visual experiences.

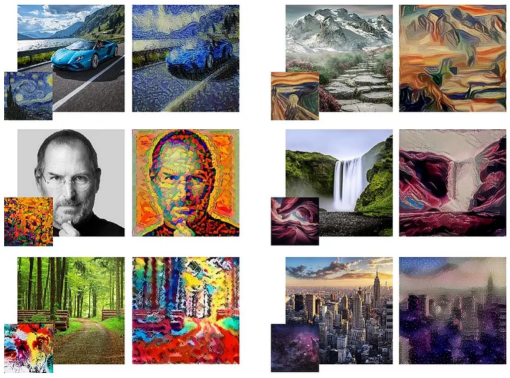


Figure 1: An example of style transfer

2 Problem Statement

Given a content image and a style image, the problem of style transfer is to generate a new image that retains the structural and semantic content of the original image while incorporating the visual style (such as colors, textures, and patterns) of the style image. The goal is to produce an image which blends the content from the source image with the aesthetic qualities of the style image.

3 Related works

In deep learning, style transfer has evolved significantly. Some of the events that shaped this field are mentioned below.

- **Traditional Style Transfer** (CNN-based) - The foundational work in neural style was proposed by Gatys et al. in 2015. They used Convolutional Neural Networks (CNNs), utilising VGG architecture, to separeate and recombine content and style. It optimises two losses - content loss and style loss. Due to the nature of optimisation process, inconsistent style application can occur, thus leading to poor quality of generated image.
- **Image Style Transfer with Transformers (StyTr^2)** - Transformer based approach to address the long-range dependencies, which is an issue in previous CNN approaches. It analyzes the deficiency of existing position encoding methods and proposes Content Aware Positional Encoding (CAPE), which is scale-invariant and more suitable for image style transfer.
- **AdaAttN** (SOTA) - It enhances local visual quality by adaptively normalizing content features based on both shallow and deep features from content and style images. It learns spatial attention scores and calculates per-point weighted statistics to match local feature distributions, improving the overall style transfer quality and reducing distortions. Though, it may not fully capture the interdependencies and relationships between different channels.
- **Contrastive Arbitrary Style Transfer (CAST)** - CAST leverages contrastive learning to directly learn style representations from image features. It utilizes Multi-Layer Style Projector, domain enhancement

modules to accurately learn the style distributions. It also utilises the learned style representations to perform actual style transfer, through generative network (similar to current InST architecture).

As CAST relies on contrastive learning, it requires both positive and negative style examples to enhance the learning of style differences. This process may require careful curation and balancing of datasets, which can be time-consuming and potentially limit the diversity of style representations if suitable negative examples are hard to obtain.

- **StyleShot(Current SOTA)** - A good style representation is crucial and sufficient for generalized style transfer without test-time tuning. With dedicated design for style learning, this style-aware encoder is trained to extract expressive style representation.

4 InST Model architecture and involved steps

Traditional methods like CNN-based style transfer often fail to accurately capture complex elements such as object shapes and high-level semantics, focusing mainly on textures and colors. Text-based methods also struggle, as they rely on vague descriptions that cannot fully represent the detailed attributes of a painting, like brushstrokes or material quality. To address these challenges, the Inversion-based Style Transfer (InST) technique was introduced. InST learns the style directly from a reference painting using diffusion models and textual inversion, eliminating the need for detailed textual descriptions. It captures the painting's attributes such as semantic elements, object shapes, brushstrokes, and colors, and transfers them to a natural image while preserving content.

The model operates in multiple spaces: textual, latent, and pixel spaces, and it performs both textual inversion and stochastic inversion to guide the generative process. The model begins by encoding the style and content images and then performs diffusion and denoising processes to synthesize the final stylized result. During training style and content images are taken same to ensure that the focus is on learning style transfer and image generation, while also keep the structural integrity of content image.

The key steps are mentioned below-

4.1 Textual Inversion

- The style image y is first encoded using the CLIP image encoder into an embedding. This embedding is passed through a multi-layer cross-attention mechanism.
- Key information regarding style image is further optimised through cross-attention layers.
- The final textual embedding v is produced, which is a learnable vector representing the style in the textual space. This embedding guides the generative model by conditioning it on style information.

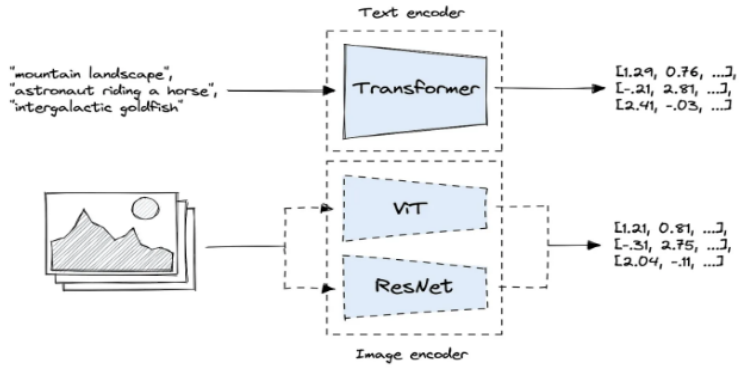


Figure 2: CLIP Architecture used for encoding style images into textual embeddings.

4.2 Stochastic Inversion

- The content image x undergoes a diffusion process where random noise is added to the image. This step perturbs the content image, turning it into a noisy version.
- A denoising process predicts the noise that was added to the image and gradually removes it through reverse diffusion.

- During generation, the predicted noise from this process serves as the initial noise input, allowing the model to preserve the semantic structure of the content image while applying the style from the reference.

Figure 3: Diffusion Process: Adding and removing noise during stochastic inversion.

4.3 Conditioned Synthesis Process on Stable Diffusion Models

- Initially, During Inference, content image is added with the predicted noise found through stochastic inversion.
- With the style embedding v and the noisy latent code from the content image, the model performs a sequence of denoising steps in the latent space.
- This denoising U-Net gradually refines the latent codes, progressively transforming them into the final stylized latent code z.
- Finally, the decoder converts the stylized latent representation back into the pixel space, generating the result image with the style of the reference and the content of the original image.

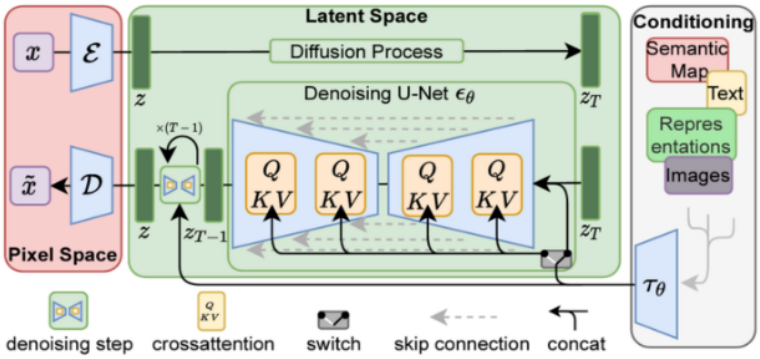


Figure 4: Latent Diffusion Model (LDM) for efficient style transfer in latent space.

The combination of textual inversion (to capture style information) and stochastic inversion (to preserve content) allows the generative model to produce a new image that blends both content and style efficiently.

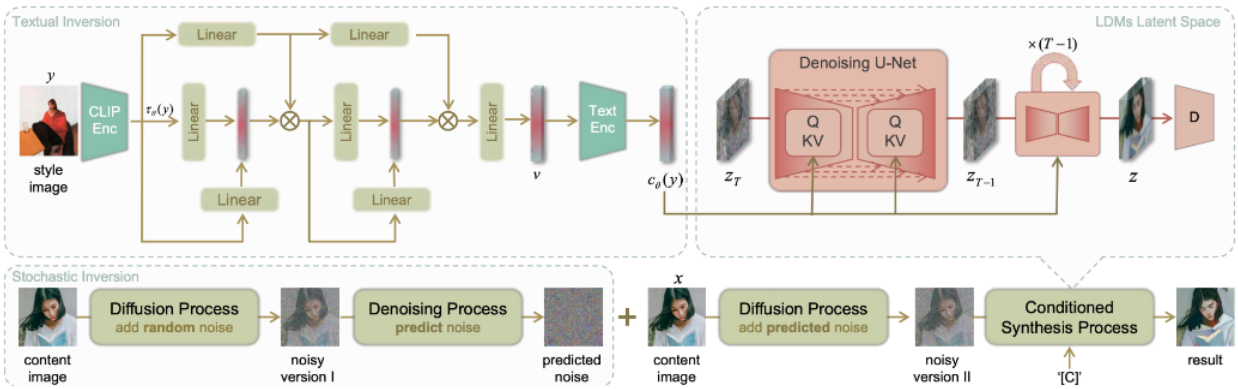


Figure 5: InST Architecture: Combining textual and stochastic inversion with latent diffusion.

5 Implementation and Comparison

We have implemented InST inversion-based Style Transfer, and also implemented StyTr^2. We loaded the pre-trained InST model and tested it using both context and style images. The prompt field was left empty to capture the pure style transfer from the style images. We used the same images provided in the paper to compare our results with the published ones. We are comparing the results with the ones obtained from the original inversion-based Style Transfer method. The comparison includes five images, where each row presents the following columns:
1. Content Image 2. Reference Image 3. InST (Inversion-based Style Transfer) Output
4. Our InST Implementation Output 5. Our StyTr^2 Implementation Output

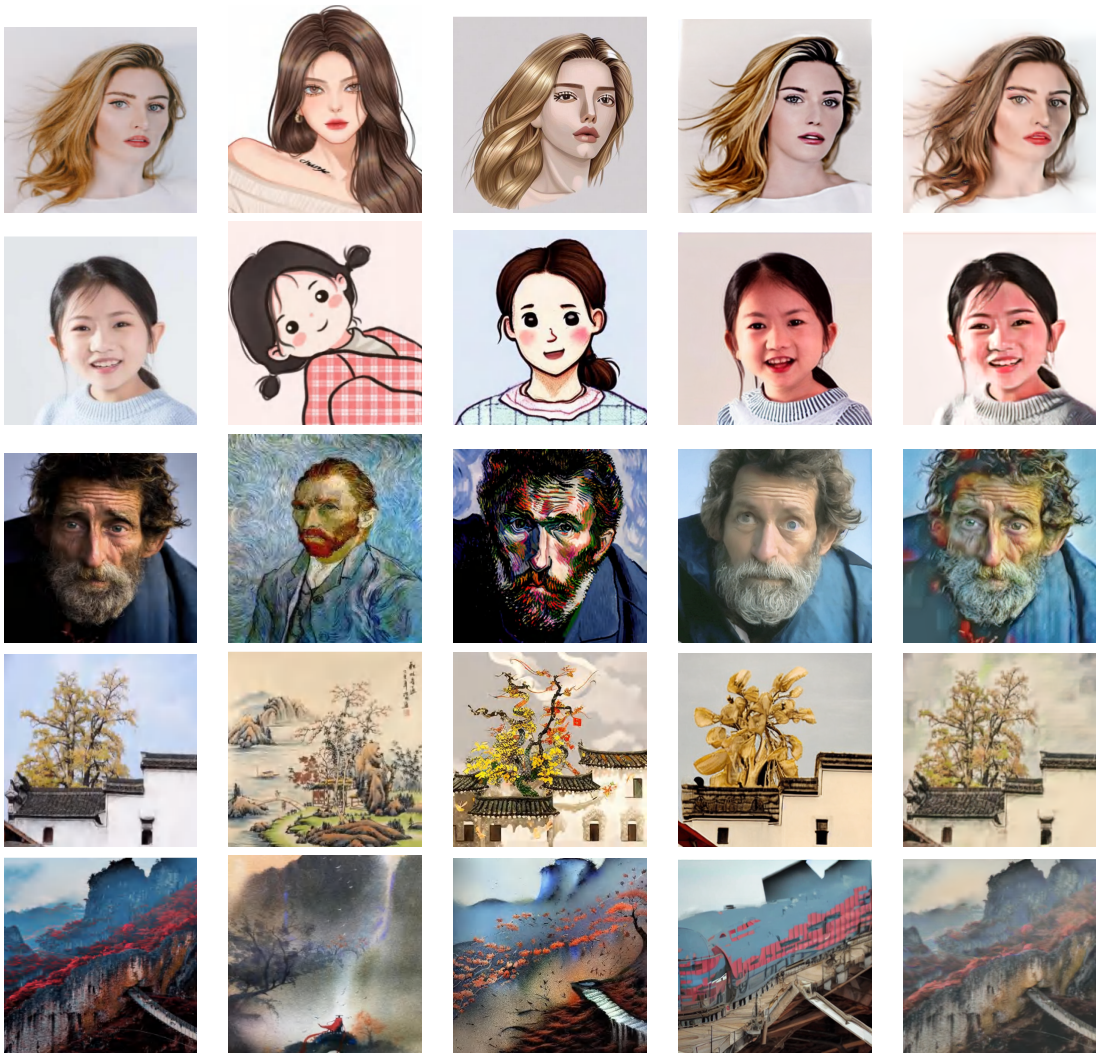


Figure 6: Comparison of different Style Transfer methods across various content and reference images for more comparisions see this doc.

6 Limitations

Diffusion-based approaches, such as InST, encounter challenges in converging towards the optimal style embedding. This issue can result in difficulties in generating a style and content that closely aligns with the input images (as evident from the content deviation in Figure 6 first row) .

Although InST method can transfer typical colors to some extent, it may struggle when there is a significant discrepancy between the colors of the content image and the reference image. In such cases, the color transfer might not achieve a semantically one-to-one correspondence.

The model heavily relies on a textual inversion-based encoder, which may not fully capture style-specific attributes without additional tuning.

The current approach may experience content leakage, where the generated style does not sufficiently mask the content of the original image.

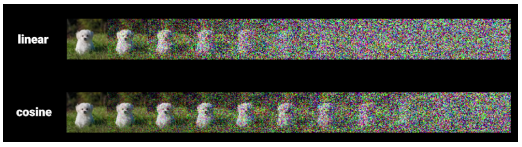
The model's inference process is slow due to the complexity of the inversion and style encoding, leading to inefficiencies in real-time applications.

Figure 7: The green hair of the content is not transferred into brown.

InST exploits a CLIP image encoder to extract visual features serving as style embeddings due to its generalization ability and compatibility with T2I models. However, since CLIP image encoder is primarily trained to extract unified semantic features with intertwined content and style information, these approaches frequently result in poor style representation

7 Proposed Improvements

To address the limitations of the InST method, we propose the following enhancements to refine style embedding, reduce content leakage, and improve inference efficiency:

- **Separate Training for Style Embeddings:** Train the style image encoder and textual inversion module separately to develop specialized, style-aware text embeddings. This approach enhances the model's ability to capture artistic nuances, improving the coherence and fidelity of generated images.
- **Utilizing Specialized Datasets:** Use the SharedGPT4 dataset for content and the WikiArt dataset for style to capture a wide range of artistic characteristics. This combination improves the model's handling of complex styles while maintaining content consistency.
- **Reducing Content Leakage:** Optimize the image encoder and textual inversion module jointly to prioritize style representation, thereby reducing content leakage and preserving the intended artistic style.
- **Embedding Contextual Style Information:** Incorporate additional context such as historical details, themes, and artistic techniques into the embeddings. This enriched context helps the model generate images that not only replicate visual style but also reflect the artistic narrative.
- **Increasing computational performance:** Replace UNet with ENet or BoxENet as their computational performance is up to 10 to 15 times higher than the UNet performance.
- **Better preserving of content structure** - Using cosine schedule instead of linear schedule as it overcomes both limitations of linear schedule which are too rapid in destroying the schedule and too uninformative at the end. This allows for UNet to better capture stylistic elements while preserving content structure. 

These proposed improvements aim to enhance the InST method's ability to generate high-quality, stylistically accurate images, addressing current challenges in style representation and content leakage.