STAT453 Final Project - Image Style Transfer

Callin Dai xdai47@wisc.edu
Yuran Jing jing32@wisc.edu
Haoxi Li hli876@wisc.edu
Tianyi Xu txu223@wisc.edu

Abstract

The task of semantically preserving style transfer in images, where the stylistic elements of one image are applied to the content of another without losing the original message, remains a significant challenge. Despite notable advancements through deep learning techniques, existing methods exhibit limitations. In this case, we finetune multiple Stable Diffusion Models capable of effectuating style transfer via textual directives. Utilizing a conditioned image coupled with text prompts, our model adeptly retains the original content while adapting the prescribed style. Addressing the scarcity of well-paired datasets for training, we utilize other generative models (CycleGAN) and large language model (GPT-4) to produce a thousand of images and instructions that are used to train our models. Our experimental results demonstrate that by finetuning on large-pretrained foundation models, not only does it flexibly execute style transfer but also ensure the preservation of semantic integrity, with better quality and fidelity compare to previous style transfer methods.

1. Introduction

Style transfer techniques in deep learning enable the generation of images that adopt specific artistic or aesthetic styles while preserving the fundamental content and structure of the original base image. Traditionally, graphical analogy is conceptualized by Hertzman et al. with the aim of combining two unfiltered images to create the targeted image. They emphasize its capability on some complex nonlinear graphical variations and do not require programming support [11]. Subsequent developments have introduced deep learning methodologies such as Neural Style Transfer, Generative Adversarial Networks and Diffusion models, significantly improve the effectiveness of texture transfer problems. To further enhance the overall performance on style transferring tasks, we consider to finetune Stable Diffusion Models by applying the learnable weights and training script from large pre-trained foundational model. For addressing the scarcity of pre-existing large scale of paired data, we utilize the capability of large multimodal model GPT-4 [20], image-to-image translation model CycleGAN [31] and arbitrary style transfer method Adaptive Instance normalization(AdaIN) [15] to prepare the text prompts and matched images data respectively.

Our finetuned Stable Diffusion Models on various dataset that, given a conditioned image and text prompts for how to transfer style, generate a new image which is composed by targeted style features and original semantic information. The integration of text instructions within our models frameworks allow for style modifications based on individual preferences, enhancing both flexibility and interactivity. Evaluations using various metrics demonstrate that our models maintain a high level of fidelity in preserving the original content of the input images, thus validating the same or even better effectiveness as only text input models and image to image models.

2. Related Work

Deep Neural Network Approach. Gatys et al. pioneered a novel approach by employing a linear combination of content and style losses within deep convolutional neural networks (VGG networks) to generate optimal images [8]. This method successfully captures both the semantic information of the content image and the distinctive features and representations of the style image. The approach has spurred numerous enhancements in the field.To accelerate the style transfer process, Wang et al. proposed a hierarchical training scheme to combine multiple facets, including RGB colors, coarse texture structure and luminance channels [29].Compared to the singular transfer, this technique successfully tackles with the low speed of online iterative optimization procedure found in Gatys' original works and enables the generation of high-resolution images.Additionally, Adaptive Instance Normalization also presented to contribute on improving the effectiveness of neural style transfer [15]. Normalizing the content images by the parameters from style images significantly reduce the run time of the optimization algorithms.

Generative Adversarial Networks. Ian came up

with a groundbreaking adversarial nets framework for image generation and manipulation [9]. It consists with two sub-models, generators and discriminators that are trained simultaneously by performing zero-sum games. Furthermore, the development of Domain adaptation makes adversarial learning achieve robust performance on cross-modality adaptation tasks [27]. One GAN-based algorithm that is used in style transfer tasks is the CycleGAN [31]. Adding cycle-consistent loss effectively performs the projection between two image domains which is perfectly suited for style transfer. Additional GAN models such as CartoonGAN [6], AnimeGAN [5] are specialized on particular styles transferring. These characteristics demonstrate the robustness and adaptivity of Generative Adversarial Networks on image manipulation tasks.

Diffusion-Based Approach. Diffusion models have recently emerged as the state of art image generation techniques over GANs. It involves a Markov chain process in the forward pass by introducing Gaussian noise, followed by an iterative denoising process using a U-Net architecture [13, 23, 24]. This capability of reconstructing high-quality and realistic images underscores the effectiveness of diffusion models in the image generation domain. Moreover, Contrastive Language–Image Pre-training (CLIP), a recent innovation, greatly motivates the efficiency of understanding and interpreting images in the context of natural language descriptions [21]. The integration of CLIP utility within diffusion models facilitates the generation of images without the need for collecting style-specific images beforehand, thus providing substantial flexibility in image generation tasks. In this case of the Stable Diffusion Model [23], it is done by first encoding the image into a lower-dimensional latent space. Then, it trains the diffusion model with latent representation as the input. The entire process reduces the considerable amount of training resources and inference speed, demonstrating the robust capabilities of diffusion models in generative tasks. Further, previous work has shown finetuning Stable Diffusion model for iamge editing tasks, notably InstructPix2Pix [3], by finetuning on Generated images from Stable Diffusion and using Prompt-to-Prompt [10] for image editing.

Fine-tuning on pre-trained models. The process of refining a pre-trained model by training on a smaller dataset is called Fine-tuning [4]. This method is proficient in suiting more specialized use cases. Furthermore, not only does it preserve the functionality of original pre-trained model, but it also enhances model's ability to deal with variety tasks in diverse and flexible ways. Indeed, fine-tuning on pre-trained model such as Diffusion-based approach provides great convenience on image generation domain [25].

Composition of pre-trained models. Recent research has shown that interacting multiple large pre-trained models enables to tackle with certain tasks from various perspectives. For instance, it has ability to compose zero-shot multimodal reasoning effectively [30]. The synergy between large language models with proficient skills on text-based tasks and image generative models significantly drives progress in the field of style transferring.

3. Proposed Methods

In this section, we describe how we generated the dataset we used for finetuning the diffusion models, as well as our whole finetuning process. In section 3.1, we discuss our datasets‘ full creation process and in section 3.2, we will discuss our finetuning techniques and settings.

3.1. Datasets for finetuning

Because we could not find a paired source images + instruction + transferred images online, or even any paired source images to style transferred images, we decided to manually create our paired dataset.

3.1.1 Source Image Dataset

First of all, we will need to choose some images to use as our dataset so that we can generate the target image in the style transfer using already existing models. To do this, we utilized two image datasets. The photo dataset proposed in the CycleGAN [31] and also MSCOCO [18]. The photo dataset proposed in the CycleGAN was downloaded from Flickr using the combination of tags landscape and landscape photography. Black-and-white photos were pruned. The images were scaled to 256×256 pixels. In total, there were 7038 such photos. We also used subsets of images from MSCOCO, which contain images of dimension 1024 x 1024 that are gathered of complex everyday scenes containing common objects in their natural context. This is also useful in our case as it makes the model difficult to interpret the semantic information. We used a subset of consisting 5226 of those images in our experiments. In addition, we also used the Kaggle dataset "Best Artworks of All Time" [16], which consists of high-quality original artworks of various famous artists. We took all the images by Claude Monet, Paul Cezanne, Vincent van Gogh, Edgar Degas, and Pablo Picasso to use as our possible style images.

3.1.2 Generating Style Transferring Instructions

To finetune a text-guided image-conditioned diffusion model, we need to have text instructions that guide the de-

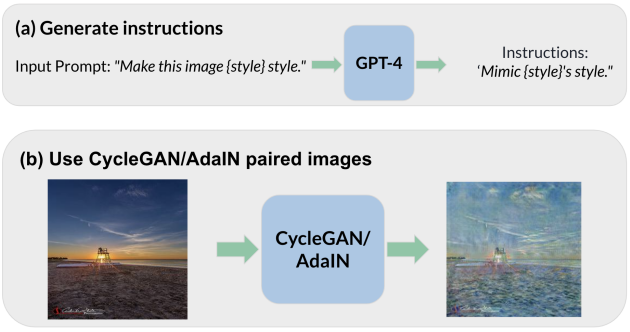

Figure 1. Process showing how we generate text instruction to guide the style transfer (a) and also the target transferred to form our dataset (b).

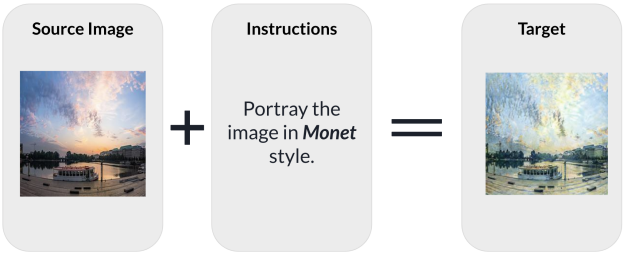

Figure 2. An example of one single paired style transferring training data is for the image-conditioned text-guided diffusion model. In this example, the source image is sampled from the photo dataset, transferring instruction is generated using GPT-4, and the target image is transferred using CycleGAN.

noising process to generate desired outcomes. We followed similar instructions to the process outlined in the InstructPix2Pix paper [3], with a few modifications. We use the SOTA LLM GPT-4 [22] to generate such instructions for us. Specifically, we utilized its in-context learning abilities by providing it styles and a few examples of our desired prompts like $(s, h(s)_1, h(s)_2, \ldots, h(s)_k)$, where s is an input style and h is our mapping from the style to the style transferring instructions, the model would learn to output $h(\hat{s})$, each time with slightly different outputs given the same s. For example, if the input consists of examples like "Make this image monet style", and the style is monet, the GPT-4 would learn from the context we provided to output "Portray the image in Monet style." This diversifies our editing instructions. We generated around 100 different editing instructions for each style we were transferring. Figure 1 (a) shows how this process works concisely.

Here is the list of styles we used: Monet, Vangogh, Cezanne, Ukiyo-e, Degas, and Picasso.

3.1.3 Generating Paired Training Data

Given the input image from various datasets, and our generated transferring instruction, our goal would be to generate the paired images using CycleGAN and Adaptive Instance Normalization method. Figure 1 (b) concisely shows this process. For each of the photo data, we choose a style for it, then we input it into the available pre-trained CycleGAN model for that style to get our output image as the target image when finetuning the model. For each of the MSCOCO dataset images, we choose a style for it, then we randomly sample one available style image for that style and run it through the pre-trained AdaIN model to get our output image. Then, we randomly assign a style transferring instruction corresponding to that style which we generated in section 3.1.2. Then, we can form our finetuning dataset by combining what we have done so far, consisting of the source image (from the CycleGAN photo dataset or MSCOCO), the style transferring instruction (generated using GPT-4), and the target image (generated using CycleGAN or AdaIN). Figure 2 shows one example of how a single paired style transfer data is like in one of our generated datasets. We then generate different numbers of datasets using either model later to finetune our model.

3.2. Finetuning

As described above, we will finetune an already pretrained stable diffusion model using our generated dataset. See Figure 3 for a concise graphical illustration of the process.

However, we are doing it slightly differently here than just naively finetuning a stable diffusion model. In our case, we have an additional input image provided that acts as the condition during the denoising process in the stable diffusion model. The original stable diffusion does not have this as it is a text-to-image generation model. Thus, we make the following alteration to the original model structure: 1. We add a Variational Autoencoder(VAE)'s encoder q_ϕ to encode the input image x into the CLIP latent space, we denote the encoded input as $q_\phi(x)$. 2. At the beginning of the denoising process, instead of just sampling noise $z \sim N(\mu, \sigma^2)$ from the latent space, we, in addition, concatenate the $q_\phi(x)$ to the noise by doubling the number of channels (the dimension went from $[N, C, H, W]$ to $[N, 2C, H, W]$, representing batch size, number of channels, height, width, respectively), which act as the condition in the denoising process. So the final starting point in the denoising step is in the form of $[z|q_\phi(x)]$. We also modify the current convolution layer in the denoising U-net by doubling the number of input channels and also changing the current weight's shape and initializing all the newly added entries in the weight matrix to 0, which will be updated as we finetune the model.

So in the end, our whole model has the following components: A VAE that encodes the input image into the CLIP space and decodes the output latent back to the image rep-

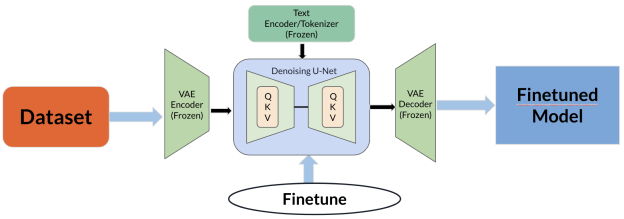

Figure 3. Our finetuning process. We finetune the entire image-conditioned stable diffusion pipeline and when doing so, we freeze everything except for the denoising U-Net.

resentation, a tokenizer and text encoder that encodes the text prompt into the clip space, a denoising U-net that responsible for denoising the encoded image in the latent space. While finetuning, we mainly focused on changing the weight for the denoising U-net as that is what is responsible for learning which noise to remove to construct the desired image conditioned on the editing instruction and input image. While VAE and the text encoder serve a general function of encoding the input into the latent space, so we do not need to change them during the finetuning. Thus, we froze all other parts in the Stable Diffusion Model, and we only finetuned the denoising U-net during the finetuning process.

For the training hyperparameters, we use the following. We resized the input image to 256×256 dimensions for the width and height, applied a random flip, and used a batch size of 2. We also finetuned for 1000 epochs. We used a learning rate of $5 \cdot 10^{-5}$ and Adam optimizer. We also applied a conditional dropout with a probability of 0.05, which means we randomly dropped out the condition during the denoising step with a probability of 0.05 for better generalization for both conditioned and unconditioned image synthesis for the denoising U-net.

4. Experiments

In this section, we discuss how we run our experiments. In section 4.1, we will introduce each of our finetuned models. Then, we will both qualitatively and quantitatively evaluate our 4 finetuned models, and compare them with the baseline, which we choose to be the InstructPix2Pix model since it is also a finetuned Stable Diffusion model for image editing/translating purposes.

4.1. Experiments Setup

We finetuned the modified image-conditioned text-guided Stable Diffusion model in 4 different settings. See Table 1 for details. We started with experimenting using a dataset generated by CycleGAN where we used a subset of 2800 images of our full photo dataset, using a pre-trained weight from Stable Diffusion 1.5 [23]. However, we find that using Stable Diffusion Weight requires a longer epoch for convergence as the weight was trained for image generation, rather than transferring an image to a different style. Thus, we moved on to using InstructPix2Pix's pretrained weight, which is a fine-tuned Stable Diffusion on a large amount of data for the image-editing task. We believe this will give better results and faster convergence as the model already learned pixel-wise image editing and transfer, and it also serves as a baseline for comparison in our later evaluation. Then, for our 2nd model, we finetuned in the same configuration as our previous one but changed from Stable Diffusion 1.5 weight into weight from InstructPix2Pix. We then finetuned 2 more models using 8000 CycleGAN transferred images and 4946 AdaIN-generated images, both using pretrained weights from InstructPix2Pix. In the end, we end up with 4 finetuned models, in addition to the baseline model, which uses pretrained weight from InstructPix2Pix and its weight directly compatible with our proposed model.

4.2. Evaluation Methods and Metrics

We will explain in the following subsections our distinct metrics to objectively evaluate the model in the quality and capabilities of our model in style transferring.

4.2.1 CLIP Similarity

CLIP similarity is calculated by

$$similarity = \frac{X_{CLIP_1} \cdot X_{CLIP_2}}{\max(\|X_{CLIP_1}\|_2, \epsilon) \cdot \max(\|X_{CLIP_2}\|_2, \epsilon)}$$

Where X_{CLIP_i} is a CLIP representation using CLIP image encoder, and we choose ViT-B-32. $\epsilon > 0$ is a small arbitrary real number to avoid dividing by 0. It measures how similar two images are in the CLIP latent space. Since it is difficult to find a true dataset to compare with, as style transfer is a subjective task, the testing set is instead to be chosen of 280 subsets of CycleGAN transferred images of 4 styles (70 each), with source image from MSCOCO dataset. CycleGAN does a relatively accurate job at style transfer so we choose to compare how close our model can get to CycleGAN's result. Figure 4 shows how it is calculated in a graphical simple illustration.

4.2.2 CLIP Directional Similarity

CLIP directional similarity is calculated by

$$dir\_similarity = \frac{(\widehat{X_{CLIP}} - X_{CLIP}) \cdot (\widehat{t_{CLIP}} - t_{CLIP})}{\max(\|\widehat{X_{CLIP}} - X_{CLIP}\|_2, \epsilon) \cdot \max(\|\widehat{t_{CLIP}} - t_{CLIP}\|_2, \epsilon)}$$

Where $\widehat{X_{CLIP}}$ is the output from our model in CLIP representation, X_{CLIP} is the source image in CLIP representation, similarly, $\widehat{t_{CLIP}}$, t_{CLIP} is the caption of the

Model	Model Used to Generate Dataset	Number of Data	Pretrained Weight Used
GAN2800.SD	CycleGAN	2800	Stable Diffusion 1.5
GAN2800.IP2P	CycleGAN	2800	InstructPix2Pix
GAN8000	CycleGAN	8000	InstructPix2Pix
ADAIN4946	Adaptive Instance Normalization	4946	InsturctPix2Pix

Table 1. Configurations of our training process. We finetuned our model in 4 different configurations that differ in what model is used to generate the transferred target image, the number of examples in the dataset, and what pre-trained weight is used. We also give each model a name in the format of {Model Used to Generate Dataset}{Number of Data in the dataset}.{Weight Used} where weight used is optional to differ only the first 2 models since they use the same model to generate the dataset and have the same number of samples. Also, here SD = Stable Diffusion 1.5 and IP2P = InstructPix2Pix.

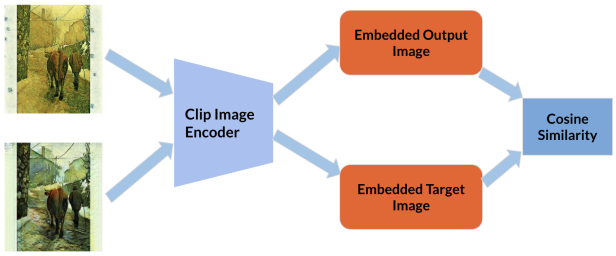

Figure 4. This shows how we calculate the CLIP similarity between the output from our model (top) and the target image (bottom). We encode the two images into the CLIP latent space using a CLIP image encoder and calculate their cosine similarity.

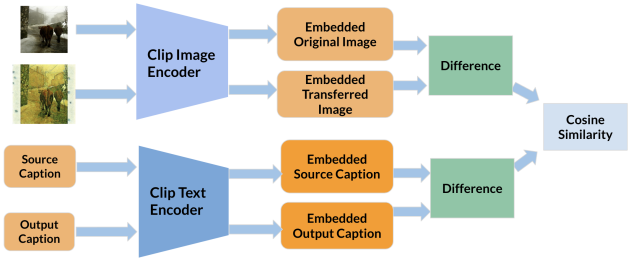

Figure 5. This shows how we calculate the CLIP similarity between the output from our model (top) and the target image (bottom). We encode the two images into the CLIP latent space using a CLIP image encoder and calculate their cosine similarity.

transferred image, and source image, in CLIP representation. Figure 5 shows how it is calculated in a graphical simple illustration.

It is important to note that captions of the transferred images are not included in the MSCOCO dataset. We used a similar trick earlier to use the in-context learning ability to generate the transferred captions from source captions and the style it is transferred into.

4.2.3 Kernel Inception Distance

To evaluate the quality of our generated image, we choose to use Kernel Inception Distance (KID) [2].

Kernel Inception distance is calculated by

$$KID = MMD\left(f_{real}, f_{fake}\right)^2$$

Where f_{real}, f_{fake} are the features extracted from the real image and fake image, and MMD is the maximum mean discrepancy such that $MMD^2(P, Q) = \|\mu_P - \mu_Q\|^2_{\mathcal{F}}$ for any $P, Q \in \mathcal{F}$, where $\mathcal{F}$ is the feature space, μ is the mean of applying kernel k to f the image feature. We can think of μ as the feature mean, MMD as the distance between the mean, and KID as just the square of the distance between the feature mean of real images and fake images. We want smaller KID as that means images transferred from our finetuned model are similar to those actually seen in the real world/drawn by the artist. Normally, we evaluate the quality of our image using Fréchet Inception Distance (FID) [12], however, it normally requires more than 10,000 samples of both real and fake images to get reliable results, thus we instead choose KID, which can be effective on a subset/instance of the real/fake set.

In our experiment, we compare average KID on a subset of 40 images transferred from our finetuned model, against 40 real images from the drawing of the artists for the style we got from the Kaggle Dataset "Best Artworks of All Time".

4.3. Software and Hardware

We use Google Colab for the experiments. The hardware used in the finetuning was a single Nvidia A100 40GB SXM, 84 GB memories, and Intel(R) Xeon(R) CPU @ 2.20GHz CPU, and we were able to achieve a training time of about 30 minutes. In the evaluation process, we changed to using a single NVIDIA Tesla V100 SXM2 16 GB GPU instead since we won't need full 40 GB GPU memory to load the whole dataset.

For the software, all training and analysis is based on Python. The training script is largely based on using Hug-

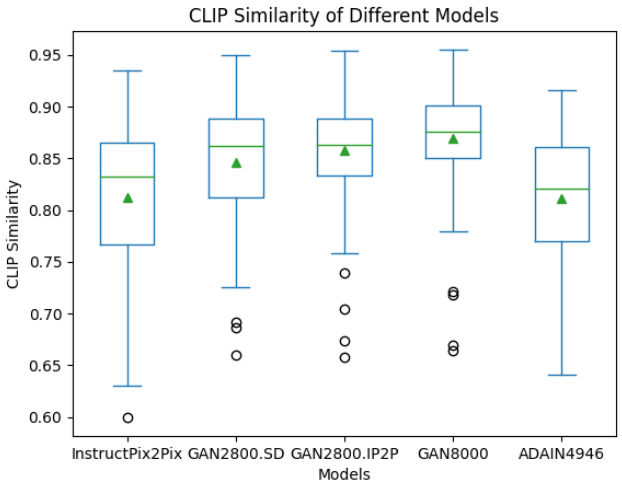

Figure 6. Boxplot for the CLIP similairty between InstructPix2Pix and our 4 fine-tuned models. We notice the variance in our fine-tuned model is small compared to InstructPix2Pix. Further, almost all CLIP similarity scores are concentrated above 0.7-0.75, with a few outliers below that, whereas before fine-tuning, the same line is drawn at ≈0.63.

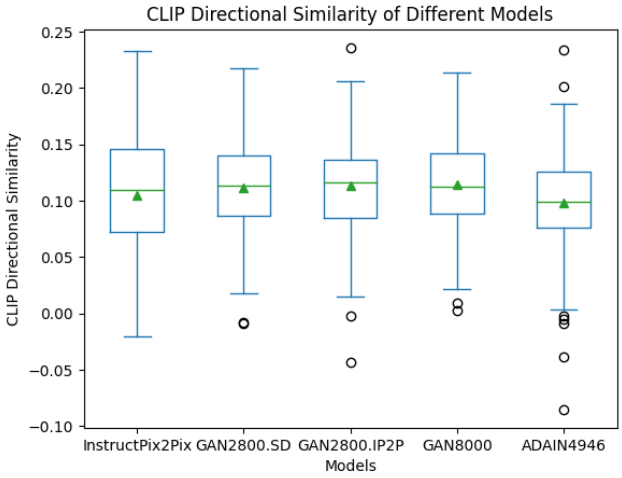

Figure 7. Boxplot for the CLIP directional similarity between InstructPix2Pix and our 4 fine-tuned models. We notice our fine-tuned model has shown less variance compared to fine-tuning but achieves higher but similar average CLIP directional similarity.

gingface's diffuser [28] package and also Pytorch package [19], and we also used Wandb [1] to document our losses. For data generation, we used open implementation of CyleGAN, AdaIN, and also GPT-4 from Openai's API. For evaluation, in addition to using Pytorch function, we also used open-clip's image and text encoder for CLIP similarity/directional similarity [17], and also the torchmetric package from Pytroch Lightning AI [7].

5. Results

We present our overall results from the experiments described above. Table 2 provides a summary of test statistics for each of our models. From these three quantitative evaluation criteria, our model, GAN8000 with the highest average CLIP similarity (0.869) average CLIP directional similarity (0.1144), and lowest for kernel inception distance (KID), has demonstrably outperformed others in style transfer tasks across various image scenarios, compared to other fine-tuned models.

5.1. CLIP Similarity

Our fine-tuned model on the CycleGAN image dataset achieved a higher average CLIP similarity compared to both the solely pre-trained and the AdaIN data-finetuned InstructPix2Pix, as detailed in Table 2. This superior performance, evident from the smaller variance in the boxplot (Figure 6), could stem from the fact that InstructPix2Pix is not specifically trained for image style transfer and is typically trained on artificial images, unlike our model which is trained on real images. Additionally, AdaIN's limitation in only aligning the most basic feature statistics—mean and variance—may not sufficiently capture complex style attributes [15], potentially leading to less effective style transfers. Moreover, within the models fine-tuned using CycleGAN-generated datasets, our CycleGAN8000 model achieves the highest average similarity, indicative of more stable performance, likely due to the larger size of the dataset.

Furthermore, fine-tuning the CycleGAN dataset enabled the Stable Diffusion Model to perform well in style transfer tasks and manage the transformation of multiple styles. Without fine-tuning, Stable Diffusion showed the largest similarity variance among all models, indicating significant instability in style transfer. This highlights that fine-tuning on a large number of datasets not only facilitates style transfer across multiple styles but also enhances the model's performance consistency.

5.2. Directional Similarity

Table 2 and Figure 7 reveal that all models perform comparably on average, suggesting a similar capacity to retain source image information during the style transfer process. However, after fine-tuning, our model shows a slight improvement in average CLIP directional similarity, indicating enhanced consistency in transferring semantic information and style. Similarly, larger datasets contribute to reduced variance, as evidenced in Figure 7.

Furthermore, the higher variance in Instruct Pix2Pix suggests a significant divergence from the source image, resulting in the most noticeable changes in features among all models. This unpredictability may render it less reliable for

Model	Average CLIP Similarity	Avergae CLIP Directional Similarity	Kernel Inception Distance (KID)
Instruct Pix2Pix	0.813	0.105	0.0421
GAN2800.SD	0.846	0.112	0.0300
GAN2800.IP2P	0.857	0.1137	0.0335
GAN8000	**0.869**	**0.1144**	**0.0241**
ADAIN4946	0.811	0.099	0.0315

Table 2. Summary of the result of our experiments.

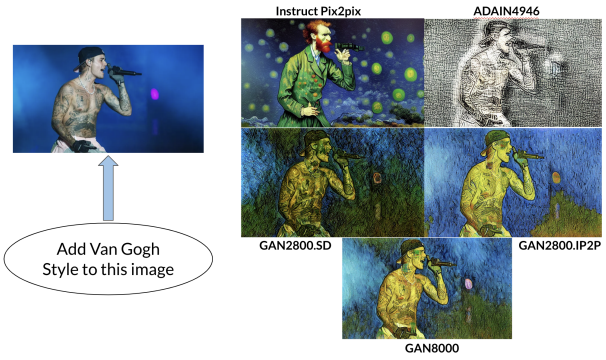

Figure 8. Example of style transferring performed by Instruct-Pix2Pix and all our finetuned model.

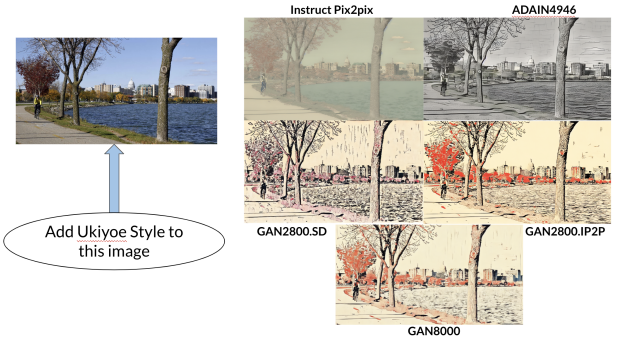

Figure 9. Example of style transferring performed by Instruct-Pix2Pix and all our finetuned model

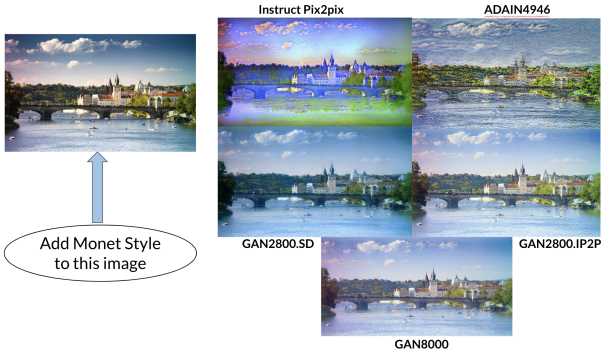

Figure 10. Example of style transferring performed by Instruct-Pix2Pix and all our finetuned model

applications that demand consistency. Therefore, our fine-tuning efforts have led to more consistent and stable style transfers.

5.3. KID

Table 2 demonstrates that fine-tuning enhances the fidelity of our transferred images. GAN8000, in particular, records the lowest Kernel Inception Distance (KID), indicating a higher similarity to actual artworks by artists. This suggests that finetuning a stable diffusion model can effectively yield images that closely mimic the style of specific artists. Additionally, the image quality from the diffusion model surpasses that of other methods. For instance, when comparing KIDs, the diffusion models we finetuned generates more realistic images with a KID of 0.0335 or lower, outperforming the CycleGAN and AdaIN models, which have KIDs of 0.0732 and 0.0430, respectively. This confirms that the diffusion model generally produces superior results in creating realistically transferred images.

5.4. Qualitative Evaluation

We demonstrate our model's performance through several examples of style transfer. In Figure 8, we transferred a real photo of a human figure into Van Gogh's style. Instruct Pix2Pix, which shows the largest variance in CLIP Directional Similarity, often unpredictably retains source image information. When transferring to Van Gogh style, the original human figure significantly transforms, prioritizing style over content fidelity, as commonly portrayed in Van Gogh's artworks. Conversely, our fine-tuned GAN8000 model excels at preserving the original content while adeptly adopting Van Gogh's distinctive color and style. This approach alters aesthetics without significantly changing semantic information. Other models also performed distinctively well in this task.

In another instance, depicted in Figure 9, we tasked the model with transforming a cityscape of Madison into a Ukiyo-e style. Before fine-tuning, the model made minimal changes, indicating unpredictability. However, post-fine-tuning, all models successfully infused their style into

the image while retaining maximum semantic content. Figure 10 further illustrates that post-fine-tuning, the models generate higher-quality images.

Thus, fine-tuning diffusion models with even a minimal dataset can effectively guide style transfer in images. This not only achieves stylistic transformations but also maintains high content fidelity and realism. These results underscore the efficacy of fine-tuning large pre-trained diffusion models for various downstream tasks.

6. Conclusion

By using relatively small datasets, we have successfully fine-tuned a stable diffusion model for image-style transferring tasks. By generating paired training datasets using CycleGAN, AdaIN, and GPT-4, our models demonstrate an adequate understanding and application of various different styles. In addition to comprehension and execution, our models also present the ability to transform input images into different styles with high fidelity. Furthermore, our models demonstrate the ability to transform different styles using a single diffusion model, which is an advantageous feature as it helps avoid the need to deploy multiple models for different styles. By combining textual instructions with large pre-trained base models (e.g., stable diffusion models), we successfully provided models that are comparable to, or even outperforms, previous style transfer models such as CycleGAN and Adaptive Instance Normalization. The advantages of our models over previous methods not only demonstrate the effectiveness of diffusion models over GANs and CNN, and also show the capabilities of fine-tuned stable diffusion models on a wider range of tasks.

6.1. Limitations

Although our models have achieved our expectations and demonstrated the ability to transform image styles, they still has some limitations that need to be addressed. One limitation of the models is diversity of styles and scope of style transformation in the datasets. Our models are only able to transform styles that are included in the training datasets, and they are not capable of transforming styles that are not included in the datasets. In addition, even though we have used small datasets, both the image generation process and the training process of our models are relatively slow. Because of time constraints, we have not been able to evaluate the performance of our model on larger datasets. Not only that, the limitation of long processing time of our models may become a more serious problem and have a negative impact when fast iteration or large amounts of data is required.

6.2. Future Directions

There are some future directions that researchers may consider. First, Researchers may add more styles and use larger datasets to improve the model's understanding and applications of various styles. Moreover, researchers can explore different dataset generation models, training configurations, and model architectures. We used the Denoising Diffusion Probabilistic Model (DDPM) for our model. DDPMs can generate high-quality images, but a drawback of DDPMs is that they require many steps to generate high-quality samples [14]. To address the issue of iteration time, researchers could use another model architecture, Denoising Diffusion Implicit Models (DDIMs), which can provide faster iteration time while maintaining similar quality images [26].

Researchers can also improve and adjust the evaluation methods. Due to the limitation of datasets and time, we used KID to evaluate the fidelity of our output images. To optimize the evaluation methods, researchers can make some changes to the code we used for our evaluation process. If larger datasets are used and more samples are genrated, future researchers can use Fréchet Inception Distance (FID) to evaluate the performance of the model.

7. Contribution

Callin Dai implemented generating text instruction using GPT-4, image captioning model (not used), evaluating CLIP similarity, and interpretation of our model's result from graph and visual output. Also the result part in both presentation and the report.

Yuran Jing assisted with finding the appropriate raw dataset, generating image using CycleGAN, brainstorming model architecture, writing and presenting the conclusion.

Haoxi Li organized generated dataset and forming the paired dataset, generating instructions, image captioning model (not used), CLIP directional similarity, creating boxplot, and writing/presenting the introduction and related work.

Tianyi Xu implemented generating dataset using CycleGAN and AdaIN, finetuning the model, the model architecture, and running the evaluation method as well as plotting other graphs. Also, the method and experiment in presentation/report.

References

[1] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com. 6

[2] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans, 2021. 5

[3] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions, 2023. 2, 3

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

[5] Jie Chen, Gang Liu, and Xin Chen. Animegan: A novel lightweight gan for photo animation. In *Artificial Intelligence Algorithms and Applications: 11th International Symposium, ISICA 2019, Guangzhou, China, November 16–17, 2019, Revised Selected Papers 11*, pages 242–256. Springer, 2020. 2

[6] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[7] Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. TorchMetrics - Measuring Reproducibility in PyTorch, 2022. 6

[8] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[10] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control, 2022. 2

[11] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 557–570. 2023. 1

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. 5

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 8

[15] 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. 1, 6

[16] Ikarus777. Best Artworks of All Time, 2020. Accessed: insert date here. 2

[17] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. If you use this software, please cite it as below. 6

[18] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 2

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 6

[20] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4, 2023. 1

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 2

[22] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. 3

[23] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 2, 4

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 2

[25] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation, 2023. 2

[26] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2020. 8

[27] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017. 2

[28] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. 6

[29] Xin Wang, Geoffrey Oxholm, Da Zhang, and Yuan-Fang Wang. Multimodal transfer: A hierarchical deep convolutional neural network for fast artistic style transfer, 2017. 1

[30] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. Socratic models: Composing zero-shot multimodal reasoning with language, 2022. 2

[31] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020. 1, 2