STATDM: Scene Text Aware style Transfer Diffusion Model

Hyungwook Choi
Seoul National University
chooi221@snu.ac.kr

Geonha Lee
Seoul National University
ghlee420@snu.ac.kr

Dahyun Oh
Seoul National University
qlass33@snu.ac.kr

Sooyong Kim
Seoul National University
ksyint1111@snu.ac.kr

Abstract

*Diffusion model with text prompt(Text-to-Image, T2I) and image prompt(Image-to-Image, **I2I**) has recently established as mainstream in the Generative AI. However, one weaknesses of T2I and I2I models is their inability to generate scene text in a human-readable format within the generated output. In this work, we introduce a seamless pipeline for removing, editing scene text in image prompt. The pipeline consists of 3 main components. (1) Scene Text Segmentation module based on **HI-SAM**(Segment Anything Model). (2) A diffusion model employing fine-grained style transfer that utilizes **cross attention masks** and **selective style injection** within U-Net blocks. (3) Scene Text Editing Module using layout from previous scene text segmentation module. In this study, we introduce a novel approach by applying style transfer to visual text rendering research. By analyzing the cross-attention map and utilizing attention masks, we have significantly improved the preservation and modification of scene text within image prompts. This research has enabled us to refine our design concepts for an end-to-end model and to prepare the necessary datasets for its training. Expanding our work from U-Net-based diffusion models to include transformer-based diffusion models and video generation models would allow us to overcome the current limitations of our model and achieve more innovative outcomes. See our codes at https://github.com/GoGiants1/MLVU-project.*

1. Introduction

As the saying goes, ”imitation is the mother of creation”. There has been an increasing demand for generating new images by referencing existing ones, leading to the emergence and widespread use of techniques such as IP-Adapter [14], ControlNet [16], manipulating attention mechanism [3, 4] and Reference + X [17], which take images as input. The aim of these techniques is to produce high-quality results similar to image prompts. One of the weaknesses of **T2I** and **I2I** models is their inability to generate text in a human-readable format within the generated output. When there is scene text in the image prompt, the problem of generating distorted text undermines the quality of the resulting output, serving as one of the major causes of quality degradation. To address these issues, various methods for generating visual text have been studied. Research such as AnyText[10] and TextDiffuser[1], which involve rendering and editing scene text using font files, and Glyph Control[13] which provide conditional control images to diffusion networks, has been conducted.

These studies have been found to produce reasonable results, but to the best of our knowledge, there is no specific model that performs both text editing and image style transfer. To achieve this, we integrate IP-Adapter[14], an image style transfer diffusion model, and TextDiffuser[1], which focuses on text editing with text prompts, into a single U-Net Network[9]. Specifically, we first construct text region masks and text stroke segmentation using a text segmentation model named Hi-SAM[15]. Second, we update the old text stroke segmentation mask to a new text stroke segmentation mask based on the text prompt. Finally, we modify the U-Net to concatenate the pre-trained models (IP-Adapter and TextDiffuser) and perform inference with both the image input and the text stroke segmentation input.

In this work, our main contributions are as follows:

- We propose a novel approach for handling both style transfer and text editing with using a single U-Net[9].
- We analyze cross attention maps between query from latents, key and value from the text and the image prompt. And we introduce text stroke cross attention mask which enhance generated visual text quality and style alignment.
- The proposed style injection algorithm demonstrates stable results in terms of both image quality and text editing compared to the baselines.

- We have created an English glyph image dataset for contrastive learning between the Visual Text (glyph) Encoder and the Text Encoder.

2. Related Works

2.1. Text Segmentation with SAM

Hi-SAM [15] is the model which tuned SAM [6] for text Segmentation. Hi-SAM use two Decoder which has same architecture with SAM mask decoder. one is S-Decoder and the other is H-Decoder. First, using pretrained Image encoder from SAM, the model converts image to image embedding. unlike SAM, Hi-SAM uses image embedding as prompt by passing self prompt module and S-Decoder gets both image embedding and image prompt. the output for H-Decoder is Text stroke segmentation. So that we can extract text from image with form of binary mask. Second main contribution of Hi-sam [15] is it can extract text bundle with semantic aspect like lines or words. for this, the model use output of S-Decoder as H-Decoder's prompt and use image embedding as H-Decoder's input. by using this architecture, Hi-SAM could reach state of arts in text segmentation and hierarchical text dectection.

2.2. Scene Text Editing

Recently, diffusion based model became states of the arts in many image generative area. Scene text editing has been no exception for this fashion. Glaph-control [13] and Textdiffuser are diffusion based mode that was designed for text editing. Glaph control use controlnet to edit text without distortion(Todo: more specific explanation for Glyph control). On the other hand, textdiffuser used clip tokenizer and layout transformer for text segmentation mask generating and 17-channel U-net [9] to train diffusion denoising network. Specifically, by using layout transformer and clip tokenizer, the model could get box coordinate approximately. then, the model created character level segmention mask for a given font style. Input was composed of 17-channel by using VAE. For tuning 17 channel U-net, textdiffuer used mario-10M dataset which were similar with movie poster images. By doing this process, it could edit scene text without distorting background images.

2.3. Image Style transfer

With the scene text securely masked, we proceed to the image transfer process. The goal here is to transform the image's primary object into a different variant within the same class or style, based on the provided text prompt and image prompt. For instance, if the input image prompt represent a poodle and the text prompt suggests a husky, our algorithm, leveraging a style transfer methods[3, 4, 14], generate the husky's appearance to resemble that of a poodle.

The IP-Adapter method rely on CLIP Vision Transformer [7] to get image embeddings and linearly project image embedding to diffusion model's latent space. In IP-Adapter, decoupled cross-attention occurs between it's attention Key and Value, derived from projected image embeddings, and the shared Query, which originates from text embedding. And other methods based on Style-Aligned[3] manipulate attention mechanism in U-Net of diffusion model.

In InstantStyle[12], a new analysis related to content leakage in the cross-attention layer, previously mentioned in the context of Style-Aligned, Visual Style Prompt[3, 4], was presented. This analysis is similar to the results of our experiments discussed in Section 4. We adopted their layer-wise scaling method shared in the study for our experiments. The key insight they provided is that, during the execution of cross-attention by the IP-Adapter in the U-Net, specific layers contribute to the image's layout and style characteristics.

3. Methods

This study introduces a novel approach for handling both image and text prompts to achieve image style transfer and scene text editing. Our method consists of three main steps: Scene Text Masking, Image Transfer within the Same Class, and Scene Text Editing. Below, we provide detailed descriptions of each step in the process.

3.1. Scene Text Masking and Mask change

We utilize Hi-SAM[15], which is specifically designed for hierarchical text detection masking. Hi-SAM yields two outputs: the text stroke segmentation mask from TSS-SAM and the text region detection mask from Hi-SAM which yields bounding boxes. We use both outputs and modify the text stroke according to the text prompt provided by user using the Python Pillow package. Finally, we obtain the modified text segmentation mask for text alteration.

3.2. Incorporate Text Rendering and Style Transfer

Previous studies on visual text and scene text rendering were designed and trained without considering style transfer. We have successfully integrated the two approaches and identified areas for improvement in the process. As mentioned in Section 2.3, there are various methods of style transfer. We adopted a technique that extracts embeddings from CLIP ViT and performs Cross Attention separately, fusing these embedding to the existing hidden state. The original implementation of Text-diffuser involved calling a Unet twice for Classifier-free Guidance. However, we have re-implemented it in accordance with the standards of Huggingface's open-source library Diffusers.

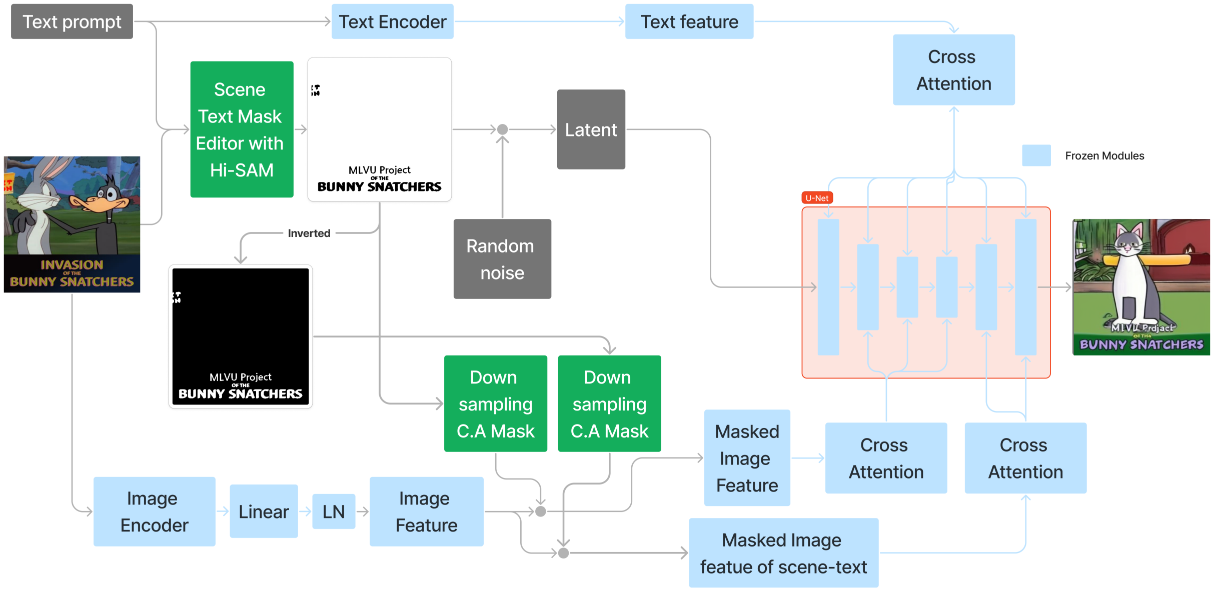


Figure 1. Whole architecture of out proposed model

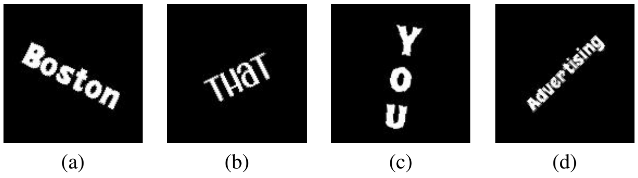
3.3. Fine-Grained Cross Attention for Style Transfer

The naive way to apply the IP-Adapter module is to input image that masked out scene text into the image encoder of the IP-Adapter. However, this approach resulted in output images with black masked areas, so we need to address the problem of image generation around the scene text areas. To resolve this problem, we propose using cross-attention masks that separate the scene text region from the background region. This allows us to more precisely control the cross-attention region and align the style of the scene text.

We investigate two cross-attention maps: one for the original Latent Diffusion Model (LDM) [8] and another for IP-Adapter. The results of our attention weight visualization are presented in Section 4. A key finding from our experiments is that the U-Net blocks in the LDM model perform specific functions. There are blocks that query the entire image and blocks that query specific regions of the image, which supports prior research on content leakage in cross-attention layers [4, 12]. Additionally, because the cross-attention queries in LDM are shared with IP-Adapter, this effect is also evident in cross-attention for style transfer. Ultimately, based on these results, we have improved the quality of generated images in style transfer by meticulously separating layers that inject information about the background and text.

3.4. Future Mehtod: End to end text rendering with glyph-Word encoder

Until now, we have been using text stroke segmentation on input images directly for text editing and text keeping. While this method yields promising results, it is not an end-to-end learning approach and has some limitations, particularly in handling small text. To solve these problems, we propose text rendering with glyph-word encoder. The Ip-Adapter model, designed for creating face layouts, utilizes face embeddings instead of CLIP embeddings. Similarly, if we can train an embedding space between glyph images and text, we could integrate it into a latent diffusion model. To do this goal, we need to train text-glyph image encoder decoder model. Moreover, designing a model to effectively inject these embeddings into the latent diffusion model presents a significant challenge. we will discuss how can deal with this problem in detail at 5.2.



(a) (b) (c) (d)

Figure 2. Examples of glyph-word 1M dataset

3.5. Prepare Glyph-Word 1M Dataset

Training a model directly on glyph images and text line (or paragraphs) is challenging. However training word instead of text line relatively easy. Therefore, our goal is to train a glyph-word encoder-decoder model. To accomplish this, we created a large glyph-word pair dataset. We prepared the top 50,000 English words sorted by frequency usage. We then rendered the word onto a (128, 128, 1) black background with white text using pillow package. At the rendering step, we introduce variations such as capitalizing the first character(Fig. 2 (a)) or the entire word(Fig. 2 (b)), randomly choosing font sizes and styles from over 10 different font files, and applying random rotation angles with $(-\pi/2, \pi/2)$. Also, we incorporated vertical writing(Fig. 2 (c)) into the dataset. Through this augmentation process, generating 20 variations per word, we were able to create a dataset of 1 million glyph image-word pairs. Fig. 2 shows some examples of our dataset.

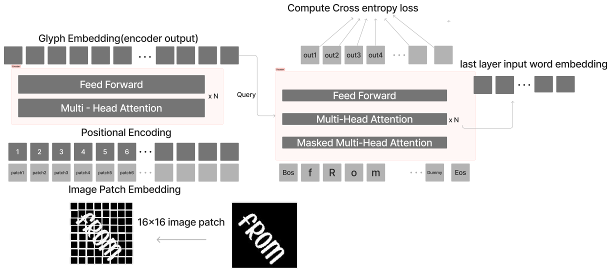
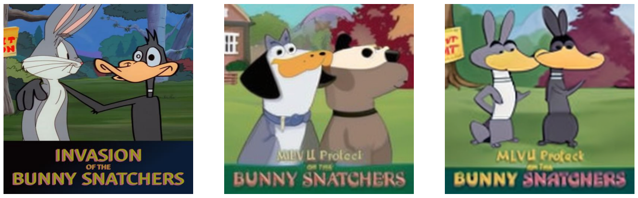


Figure 3. Structure of Glyph-word transformer encoder

3.6. Future Method: Train Glyph-Word model with transformer

Glyph-word model is quite similar to a language translation model. This is because our task can be interpreted as translating glyph image patch tokens in the encoder to the corresponding English word. In this regard, we blend Vision transformer[2] with the original transformer architecture[11]. As depicted in the Fig. 3, we partition the glyph image into 16x16 patches (resulting in 64 tokens per image), while each word is segmented into single characters with a maximum length of 16. for training, we compute cross entropy loss(or noise contrative estimation) between output of the decoder and ground truth word label. After training of this model, we can utilize the encoder output of last layer as embedding for integration into the latent diffusion Unet. Unfortunately, due to time constraints, we were unable to complete the training of this model. Fig. 3 shows overall structure of our glyph-word model. We will use the encoder output as the glyph embedding and the input embedding to the last layer of the decoder as the word embedding. These embeddings will be used in the latent Unet depending on whether the goal is to retain or replace the word. Moreover, this transformer encoder model can be applied to various other areas such as text recognition and scene word classification.



(a) Image prompt (b) Output 1 (0.4) (c) Output 2 (0.7)

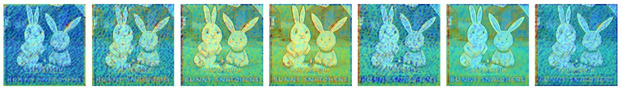
Figure 4. Examples of blurry and entangled outputs from the prompt ’two dogs’ and an image prompt 4a. Style transfer was applied with intensities of 0.4 and 0.7, respectively.

4. Experiments

We have connected three components—Text Stroke Segmentation, Scene Text Rendering, and Style Transfer—to create an initial implementation, hereinafter referred to as our baseline. Our baseline pipeline exhibited two significant issues in the style transfer outputs. First, the results were notably blurry, even making the scene text unrecognizable or disappearing. Second, objects were sometimes entangled in image, resembling Siamese Twins, even worse scene text part became overlapped to the other objects a lot(Fig. 4). To address these challenges, we extracted attention weight heatmaps to analyze the mechanisms of style transfer within a U-Net based diffusion model. Subsequently, by leveraging the Cross Attention Mask and the features of U-Net blocks, we successfully achieved objectives such as Scene Text Removal, Scene Text Editing, and Scene Text Rendering integrated with style transfer.

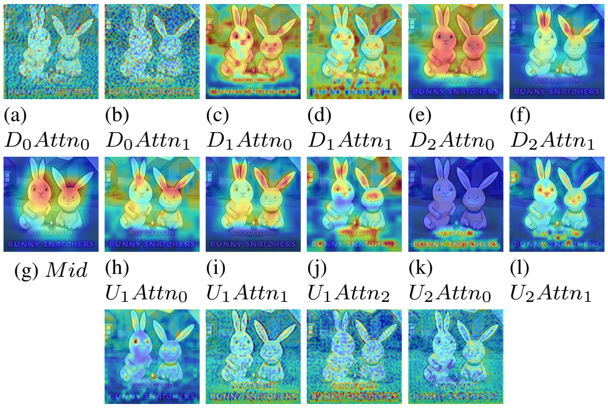
4.1. Visualization of Attention Maps

According to [8], the U-Net of the Latent Diffusion Model includes two attention processors per down block, one per mid block, and three per up block. In individual U-Net attention processor, each text embedding and image embedding perform cross-attention with hidden state separately. We visualized the original cross attention maps and decoupled cross attention(mentioned in section 2.3) maps, which are related with style injection, layer-by-layer in the our baselines U-Net architecture. Initially, the cross attention map for Down Block 0 is depicted in Fig. 5. This is a plot of attention weights corresponding to text queries, rendered as a heatmap on the output image. From the visualization, it is evident that this block attends evenly across detailed regions of the entire image. Additionally, to understand the role of each block at a glance, Fig. 6 visualizes the



(a) two (b) rabbits (c) , (d) each (e) hold (f) a (g) cup

Figure 5. Attention weights in down block 0, Text Diffuser U-Net



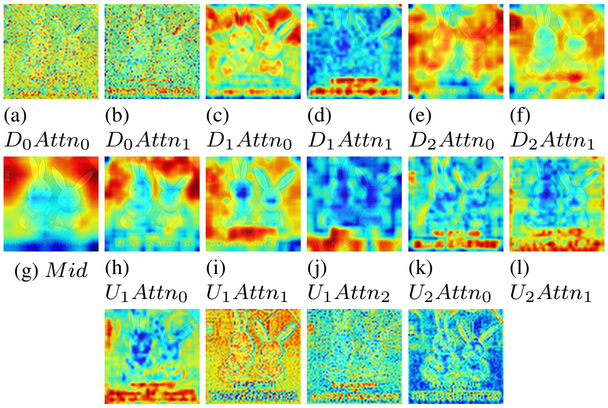
(a) D_0Attn_0 (b) D_0Attn_1 (c) D_1Attn_0 (d) D_1Attn_1 (e) D_2Attn_0 (f) D_2Attn_1 (g) Mid (h) U_1Attn_0 (i) U_1Attn_1 (j) U_1Attn_2 (k) U_2Attn_0 (l) U_2Attn_1 (m) U_2Attn_2 (n) U_3Attn_0 (o) U_3Attn_1 (p) U_3Attn_2

Figure 6. Attention weights by ”rabbits” prompt in our baseline's U-Net blocks (D_0Attn_0 means that Down block 0 and internal attention 0. U_1Attn_0 means that Up block 1 and internal attention 0)

cross attention map in response to the text query ”rabbits”.

When analyzing these visualization results, it is evident that blocks near the Mid block with lower dimensions focus on coarser features, whereas blocks with higher dimensions target the finer details throughout the image. This observation aligns with findings from StyleGAN [5], which demonstrate that injecting the style feature W into lower-resolution layers induces broad style changes, whereas its introduction into higher-resolution layers affects more detailed style transformations. Furthermore, a visualization of the heatmap for the attention weights of the 16 Style Transfer tokens (Fig. 7) demonstrates that each token precisely targets specific features within the 2D space, maintaining the observed pattern of focusing on coarser or more detailed characteristics depending on the depth of the layer, similar to what is depicted in Fig. 6.

Additionally, our baseline has identified U-Net blocks, specifically Fig. 6c and Fig. 6k, that intensively attend to scene text. Therefore, as will be explained in subsequent chapters, this observation has inspired the idea to separate regions generating objects and those generating scene text using a cross attention mask, and to carefully select the layers for performing style transfer.



(a) D_0Attn_0 (b) D_0Attn_1 (c) D_1Attn_0 (d) D_1Attn_1 (e) D_2Attn_0 (f) D_2Attn_1 (g) Mid (h) U_1Attn_0 (i) U_1Attn_1 (j) U_1Attn_2 (k) U_2Attn_0 (l) U_2Attn_1 (m) U_2Attn_2 (n) U_3Attn_0 (o) U_3Attn_1 (p) U_3Attn_2

Figure 7. Attention weights of the first IP-Adapter token in U-Net blocks (D_0Attn_0 means that Down block 0 and internal attention 0. U_1Attn_0 means that Up block 1 and internal attention 0)

4.2. Fine-grained Style Injection

By analyzing the visualization results of the cross attention maps in Style Transfer (Fig. 8), we determined that the initial entanglement occurs when objects and scene text from the prompts or images are attended to simultaneously. Consequently, we introduced a cross attention mask that distinguishes between the Scene Text and Background areas, allowing Scene Text to reference only other Scene Text, and the Background to refer solely to other background elements. This approach enabled the application of a segregated cross attention mechanism. Based on the experimental results discussed later, we applied different style injection philosophies tailored to the distinct style characteristics of the Background and Scene Text.

Fig. 8 showcases our experiments to identify target blocks for style transfer within our implemented baseline's U-Net structure. We analyzed the outputs resulting from style injection at specific blocks. From these analyses, we discovered that Down block 2's Attention 0 successfully injects the overall style from the image prompt. Additionally, we found that Up block 2's Attention 0 and Attention 1 most accurately render Scene Text. Additionally, through experimentation, we were able to identify blocks solely dedicated to style injection within the U-Net architecture of the Stable Diffusion 1.5 model.

After conducting numerous experiments and considering the trade-offs between spatial and style characteristics, we developed a heuristic to refine our style injection process. Style injections for Background areas are executed across U-Net's down blocks, the Mid block, and the lower layers of the up blocks. In contrast, injections specific to Scene Text are precisely performed in Up blocks 2 and 3. This targeted approach has notably enhanced our baseline model. This

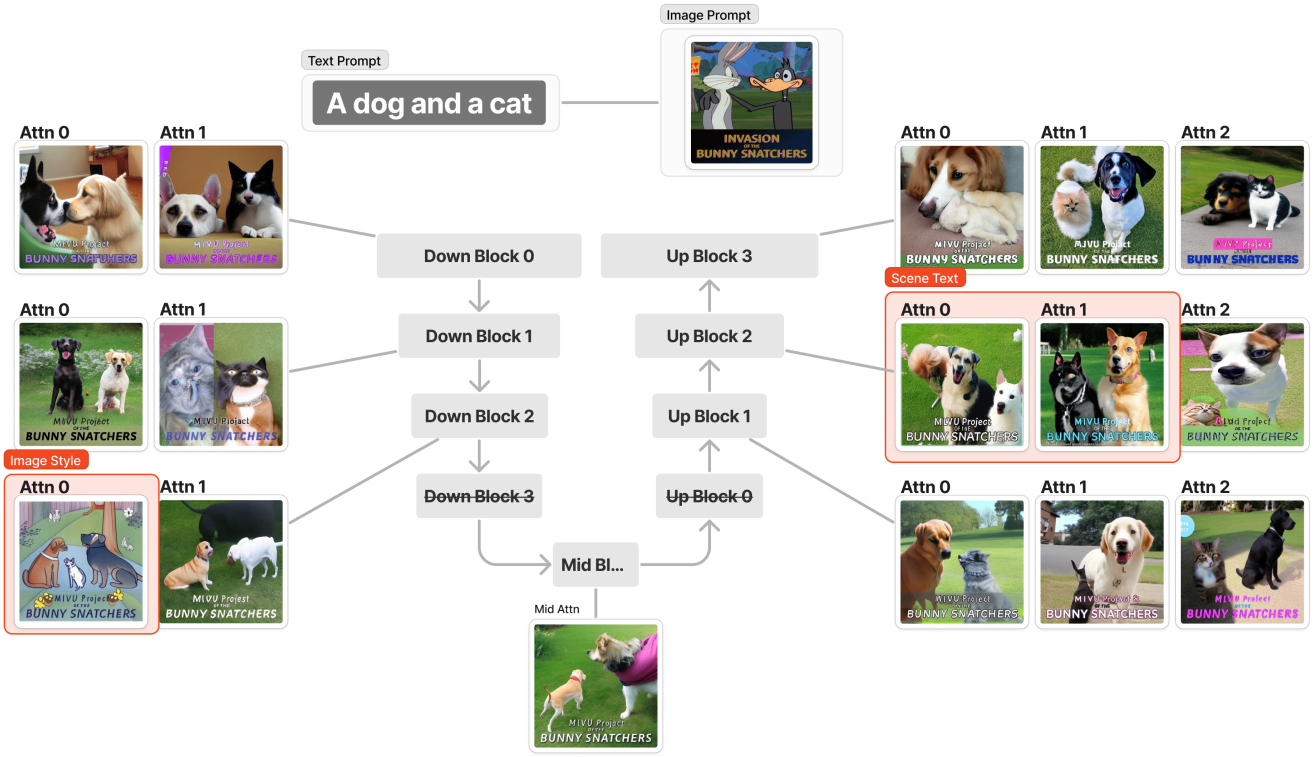


Figure 8. The Result of Style Transfer in baseline's specific U-Net block. For each experimental output, we injected style with an intensity of 0.2 at a specific attention processor within a particular U-Net block

process can be described as thoroughly mixing the features of the text prompt with those of the image prompt. Then, in the Up block, the drawing is guided by the text prompt. Additionally, when drawing text areas, the information from the image prompt is utilized to more accurately render the scene text.

4.3. Comparison Results

We have tackled a novel task that had not been attempted before. Our final pipeline is capable of performing Scene Text Rendering, Editing, and Maintaining while also executing style transfer. Our final implementation utilizes a variant of SD 1.5, the TextDiffuser; therefore, we have also included the SD 1.5 and SD 1.5 + Style Transfer pipelines in our experiments. Additionally, to evaluate the Scene Text Rendering capabilities, we tested the outputs of a Backbone model (TextDiffuser) without style transfer. For Scene Text Segmentation, we utilized our own custom-developed Mask Editing Module. The results of the comparative experiment can be seen in Fig. 9. The issue of style entanglement in the initial implementation has been significantly improved thanks to modifications in the style injection strategy mentioned at Section 4.2.

In the case of SD 1.5, style injection was implemented without any awareness of Scene Text, leading to the generation of unreadable characters or omissions. We selected Glyph-Control[13] for comparison because its methodology most closely aligns with our problem definition. However, it has a significant limitation: the text prompt must include specific information about the Scene Text. However, our method manages to maintain Scene Text quite effectively even without this information in the text prompt, and it is also capable of accurately correcting Scene Text errors that the original Text-Diffuser produced.

4.4. Ablation Study

In this section, we will compare our implemented baseline with two improved style injection strategies. Each improvement method includes (1) Using a cross attention mask to separate the scene text area from the background, and (2) Selective style injection strategy that is applied in specific blocks of the U-Net.

To examine the effects of the cross attention mask, consider the following: Fig. 10 shows an example where only the cross attention mask from (1) is varied while applying the selective style injection strategy from (2). In Fig. 10a, the styles of the scene text do not align well, with a mix of white, yellow, and purple colors, and entanglement is observed in areas like the cat's eyes and tail. However, in Fig. 10b, there is less entanglement in the cat, and the styles

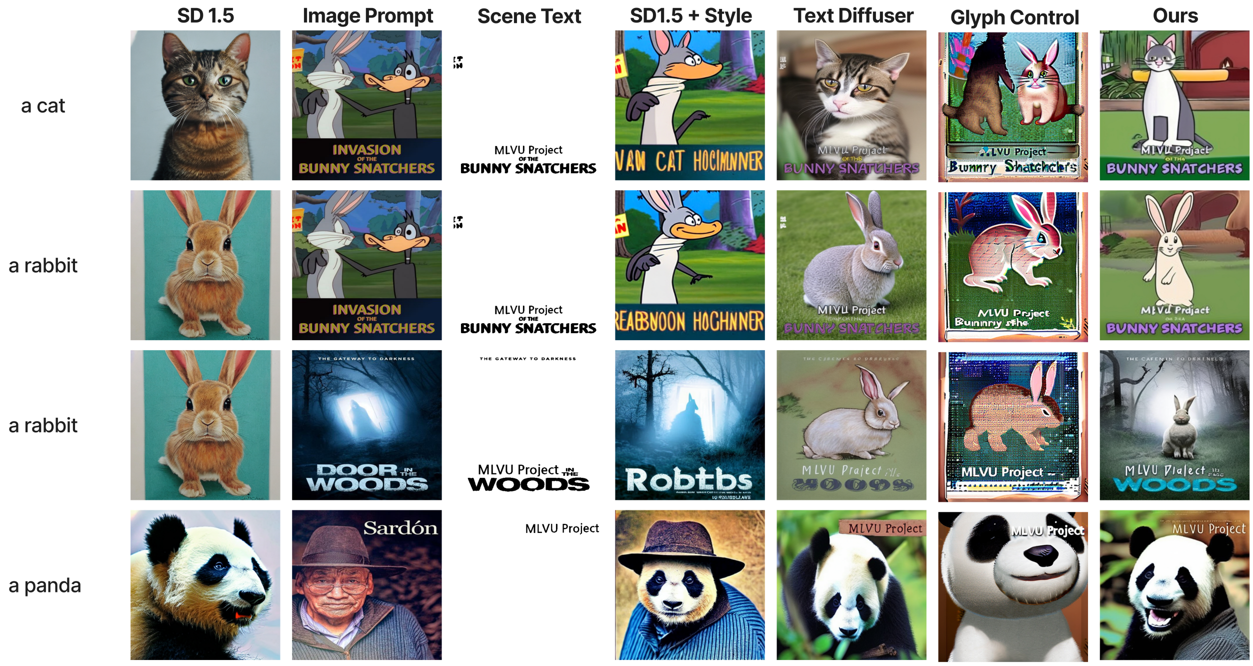
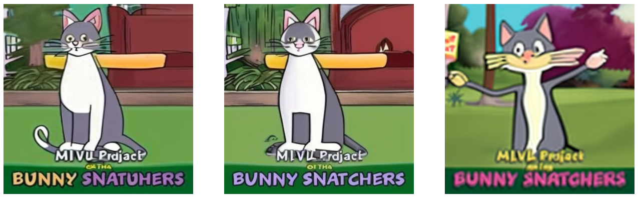


Figure 9. Comparison Result with our final pipeline and others



(a) w/o cross attention mask, w/ selective injection

(b) w/ cross attention mask, w/ selective injection

(c) w/ cross attention mask, w/o selective injection

Figure 10. Effect of separated cross attention masks and selective style injection. (Text prompt: ”a cat”, Image Prompt: Fig. 4a)

among the scene text are well-aligned. This indicates that the strategy of allowing the scene text area to only attend to scene text is effective in resolving issues of entanglement and enhancing scene text alignment. To examine the effects of selective style injection, we analyzed Fig. 10c and 10b. When the scale values are not adjusted for the target region (either background or scene text), as shown in Fig. 10c, the results are blurry and heavily entangled. Our region-wise selective style injection strategy proved effective in achieving disentanglement and enhancing the quality of generation.

We performed an evaluation using the TMDB Evaluation 500 dataset from the TextDiffuser project. This dataset comprised prompts and images, and we modified part of the scene text in the input images to ’MLVU Project’, which we then utilized as a scene text mask. We compared the generated outputs with the target images from the TMDB evaluation dataset.

Method	FID	CLIP	LEV
Ours (w/o style transfer)	415.8	0.54	6.76
Ours (w/ style transfer 1)	389.6	0.54	6.97
Ours (w/ style transfer 2)	385.67	0.54	7.47

Table 1. The first row represents our baseline method without style transfer. The second employs style transfer with a naive style injection strategy. The third incorporates our final style injection strategy and cross attention mask.

We utilized the Inception Network to calculate the distance between the input and output images, both of which had their scene text areas masked. To assess whether the output naturally displays the semantics of the input text, we used CLIP, based on ViT-Base. Finally, we employed the LEV Distance, as measured by EasyOCR, to verify if the scene text content was accurately displayed on the output image.

When applying style transfer with fine-grained mask and selective style injection strategy, our method shows improved performance in maintaining content from the input image. However, when measuring LEV Distance, our final method (third row in Tab. 1) renders the scene text 'MLVU Project' from the input prompt less clearly in the image. This can be considered a limitation of our methodology, which requires manual adjustment of style transfer intensity. If the scale is appropriately adjusted manually, it can lead to superior scene text performance compared to the initial implementation. However, in the evaluation experiments, this scale was fixed, resulting in these specific outcomes.

Lastly, the CLIP scores for these three conditions are similar. This outcome can be attributed to the experimental setup, in which the text prompt serves as a description of the target image, thus ensuring well-aligned conditions. Although there is no clear benchmark for measuring style transfer, we conducted the experiments using the methods available to us.

5. Conclusion

5.1. Limitations of our works

Our final model implementation has a limitation in that it requires manual readjustment of scale values for each text and image prompt. This is due to the disparity between the semantics of the text prompts and the image prompts, and the varying semantic expectations for each combination of inputs. This approach might be a significant disadvantage in areas like video style transfer. Therefore, designing an auxiliary network to bridge this semantic gap could enable the development of a data-driven method that mitigates this limitation.

The second limitation of our model is its inability to maintain and modify small-sized text. Consequently, the model is not suitable for text-heavy images such as book pages or posters. This issue arises because our diffusion model cannot accurately recognize small-sized text stroke segmentation input. There are several reasons for this problem. In our opinion, the most significant reasons are, first, the dataset (Matio-10M) used to train the text diffuser model does not include small-sized text, and second, the model cannot explicitly separate the scene parts from the text parts.

The last limitation of our model is that it relies on preprocessed text stroke segmentation input, which prevents it from being an end-to-end trainable method. Eliminating text stroke segmentation as an input could lead to faster inference speeds and a more interoperable model structure. To solve these problems, we proposed the model in Sec. 3.4. we will discuss how can the model expected to solve following problems below future work session.

5.2. Future Works

This section explains how we expect to address the limitations with the model proposed in 3.4. The first problem is the inability to recognize small-sized text. A straightforward solution is to create a dataset with many small text-sized scene texts and train the text-diffusion model on it. Although this approach might lead to better performance, the model won't explicitly learn the graphical meaning of the text, reducing interpretability. However, if we can train a meaningful embedding space between glyph images and text words, we can explicitly inject these embeddings into the latent Unet, similar to the IP-Adapter method. Thus, this end-to-end text editing model has three different types of embeddings. Each embedding performs cross-attention with the latent hidden state and delivers conditional information to diffusion model.

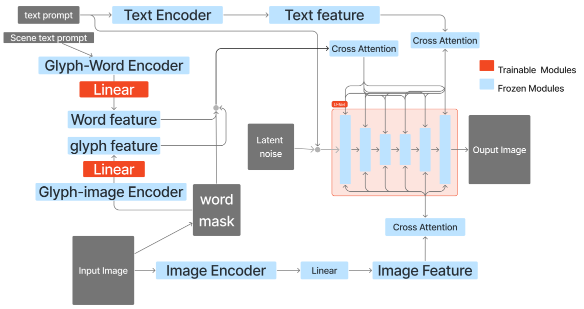


Figure 11. Image-text-glyph style transfer diffusion model

Incorporating three distinct pieces of information without entanglement presents a significant challenge. To address this, we propose a preliminary solution. Initially, a pre-trained OCR encoder is employed to detect text within the image. For text requiring replacement, embeddings are generated using a text encoder. Conversely, for text that needs to be retained, a glyph-image encoder is utilized to produce the corresponding embeddings. Following the IP-Adapter methodology, we introduce trainable linear projection layers that map each embedding to a latent space. Cross-attention is then performed within the latent diffusion U-Net block, incorporating text region masks for targeted image editing. Fig. 11 illustrates the overall architecture of our model. By conveying word-level graphical information, we anticipate enhanced recognition and editing capabilities for small-sized text. Furthermore, by eliminating the text stroke segmentation network, we will enable end-to-end training.

References

[1] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser: Diffusion models as text painters. *arXiv preprint arXiv:2305.10855*, 2023. 1

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner,

Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[3] Amir Hertz, Andrey Voynov, Shlomi Fruchter, and Daniel Cohen-Or. Style aligned image generation via shared attention. *arXiv preprint arXiv:2312.02133*, 2023. 1, 2

[4] Jaeseok Jeong, Junho Kim, Yunjey Choi, Gayoung Lee, and Youngjung Uh. Visual style prompting with swapping self-attention. *arXiv preprint arXiv:2402.12974*, 2024. 1, 2, 3

[5] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 5

[6] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 2

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3, 4

[9] 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. 1, 2

[10] Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. 2023. 1

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[12] Haofan Wang, Qixun Wang, Xu Bai, Zekui Qin, and Anthony Chen. Instantstyle: Free lunch towards style-preserving in text-to-image generation. *arXiv preprint arXiv:2404.02733*, 2024. 2, 3

[13] Yukang Yang, Dongnan Gui, Yuhui Yuan, Weicong Liang, Haisong Ding, Han Hu, and Kai Chen. Glyphcontrol: Glyph conditional control for visual text generation. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2, 6

[14] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023. 1, 2

[15] Maoyuan Ye, Jing Zhang, Juhua Liu, Chenyu Liu, Baocai Yin, Cong Liu, Bo Du, and Dacheng Tao. Hi-sam: Marrying segment anything model for hierarchical text segmentation. *arXiv preprint arXiv:2401.17904*, 2024. 1, 2

[16] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 1

[17] Lvmin Zhang (Lyumin Zhang). Reference only technique. https://github.com/Mikubill/sd-webui-controlnet/discussions/1280, 2023. Accessed on March 21, 2024. 1