Glyph-ByT5: A Customized Text Encoder for Accurate Visual Text Rendering

Zeyu Liu[1,2†‡] Weicong Liang[1,3†] Zhanhao Liang[1,4†]
Chong Luo[1] Ji Li[5] Gao Huang[2] Yuhui Yuan[1‡♯]
†interns at microsoft ‡core contribution ♯project lead

[1] Microsoft Research Asia [2] Tsinghua University [3] Peking University
[4] The Australian National University [5] Microsoft
https://glyph-byt5.github.io

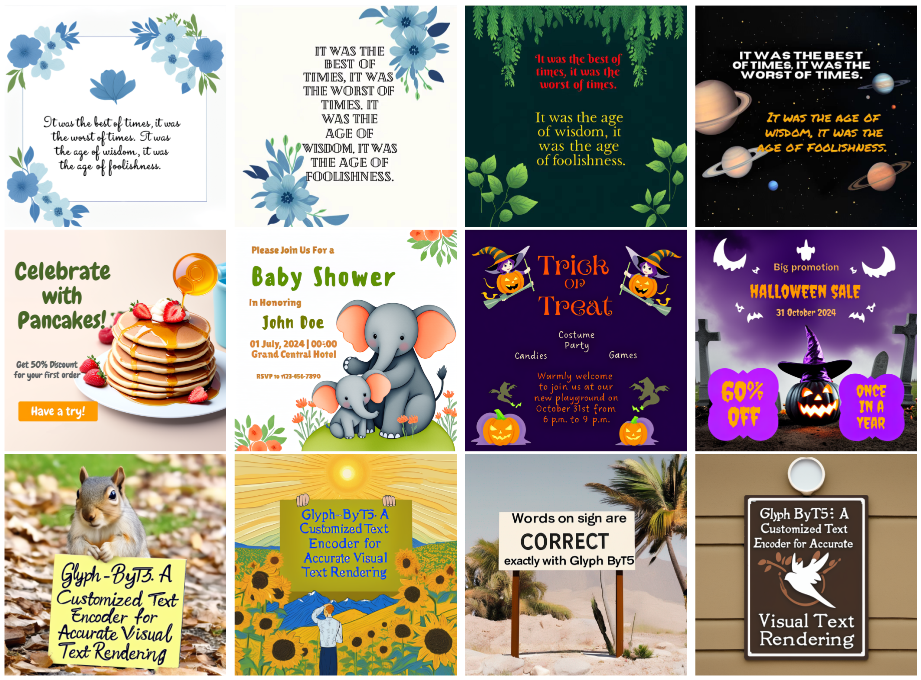

It was the best of times, it was the worst of times. It was the age of wisdom, it was the age of foolishness.
IT WAS THE BEST OF TIMES, IT WAS THE WORST OF TIMES. IT WAS THE AGE OF WISDOM, IT WAS THE AGE OF FOOLISHNESS.
It was the best of times, it was the worst of times.
It was the age of wisdom, it was the age of foolishness.
IT WAS THE BEST OF TIMES. IT WAS THE WORST OF TIMES.
IT WAS THE AGE OF WISDOM, IT WAS THE AGE OF FOOLISHNESS.
Celebrate with Pancakes!
Get 50% Discount for your first order
Have a try!
Please Join Us For a
Baby Shower
In Honoring
John Doe
01 July, 2024 | 00:00
Grand Central Hotel
RSVP to +123-456-7890
Trick or Treat
Costume Party
Candies
Games
Warmly welcome to join us at our new playground on October 31st from 6 p.m. to 9 p.m.
Big promotion
HALLOWEEN SALE
31 October 2024
60% OFF
ONCE IN A YEAR
Glyph-ByT5: A Customized Text Encoder for Accurate Visual Text Rendering
Glyph-ByT5: A Customized Text Encoder for Accurate Visual Text Rendering
Words on sign are
CORRECT
exactly with Glyph ByT5
Glyph ByT5: A Customized Text Encoder for Accurate
Visual Text Rendering

Fig. 1: Illustrating the paragraph rendering capabilities with automatic multi-line layout planning (1st row), text-rich design images (2nd row), and open-domain images with scene text (3rd row), generated with our approach.

Abstract. Visual text rendering poses a fundamental challenge for contemporary text-to-image generation models, with the core problem lying in text encoder deficiencies. To achieve accurate text rendering, we identify two crucial requirements for text encoders: character awareness and alignment with glyphs. Our solution involves crafting a series of customized text encoder, Glyph-ByT5, by fine-tuning the character-aware

‡ ✉ yuhui.yuan@microsoft.com

ByT5 encoder using a meticulously curated paired glyph-text dataset. We present an effective method for integrating Glyph-ByT5 with SDXL, resulting in the creation of the Glyph-SDXL model for design image generation. This significantly enhances text rendering accuracy, improving it from less than 20% to nearly 90% on our design image benchmark. Noteworthy is Glyph-SDXL's newfound ability for text paragraph rendering, achieving high spelling accuracy for tens to hundreds of characters with automated multi-line layouts. Finally, through fine-tuning Glyph-SDXL with a small set of high-quality, photorealistic images featuring visual text, we showcase a substantial improvement in scene text rendering capabilities in open-domain real images. These compelling outcomes aim to encourage further exploration in designing customized text encoders for diverse and challenging tasks.

1 Introduction

Diffusion models have emerged as the predominant approach for image generation. Noteworthy contributions, like DALL·E3 [2, 18] and Stable Diffusion series [21, 23], showcase remarkable proficiency in generating high-quality images in response to user prompts. However, a significant limitation persists in their ability to accurately render visual text, which is a critical element in various image generation applications. These applications range from producing design images for posters, cards, and brochures to synthesizing real-world images featuring scene text found in road signs, billboards, or text-laden T-shirts. The challenge of achieving precise text rendering accuracy has hindered the practical deployment of image generation models in these important domains.

We posit that the primary challenge hindering visual text rendering performance lies in the limitations of text encoders. The widely used CLIP text encoder, trained to align with visual signals, primarily focuses on grasping image concepts rather than delving into image details. Conversely, the commonly adopted T5 text encoder, designed for a comprehensive understanding of language, lacks alignment with visual signals. We argue that a text encoder capable of encoding character-level information and aligning with visual text signals, or glyphs, is essential for achieving high accuracy in visual text rendering. Drawing inspiration from the character-aware ByT5 encoder [15], our approach aims to customize it to better align with visual text or glyphs.

To construct the desired character-aware and glyph-aligned text encoder, we employ a fine-tuning approach based on the ByT5 model using paired text-glyph data. The main challenge arises from the scarcity of high-quality paired text-glyph data, which we overcome by establishing a scalable pipeline capable of generating virtually unlimited paired data based on graphic rendering. Additionally, we incorporate a glyph augmentation strategy to enhance the character awareness of the text encoder, addressing various error types commonly encountered in visual text rendering, as discussed in [15]. Leveraging our meticulously crafted dataset and employing an innovative box-level contrastive loss,

Method	#Params	Char-aware	Glyph-align	Precision (%)			
				≤20 chars	≤20-50 chars	≤50-100 chars	≥100 chars
SDXL (CLIP & OpenCLIP)	817M	✗	✗	21.72	20.98	18.23	19.17
+ T5-L	+ 394M	✗	✗	48.46	44.89	34.59	26.09
+ ByT5-S	+ 292M	✓	✗	60.52	52.79	50.11	42.05
+ Glyph-ByT5-S	+ 292M	✓	✓	92.58	90.38	87.16	83.17
+ Glyph-ByT5-S^{1M}	+ 292M	✓	✓	**93.89**	**93.67**	**91.45**	**89.17**
DeepFloyd-IF (T5-XXL)	4.3B	✗	✗	17.63	17.17	16.42	13.05
DALL·E3	Unknown	✗	✗	23.23	21.59	20.1	15.81

Table 1: Illustrating the improved results achieved with our approach based on SDXL across a varying number of characters, we choose the encoder of T5-Large and ByT5-Small for a relatively fair comparison. We only display the number of parameters for the text encoder components in the second column. Performance is demonstrated through evaluating the word-level precision of each model on different text length ranges. *Char-aware*: using character-aware text encoder. *Glyph-align*: glyph-alignment pre-training. We also report the performance of DeepFloyd-IF and DALL·E3 in our benchmark, which comprises 1,000 prompts, with 250 prompts within each range of character numbers. By default, we compute all precision scores at the word level. The superscript '1M' indicates the use of 1 million training pairs, whereas the preceding four rows use 500K by default.

we efficiently fine-tune ByT5 into a series of customized text encoder for glyph generation, named Glyph-ByT5.

Upon thorough training, Glyph-ByT5 is seamlessly integrated into the SDXL model using an efficient region-wise cross-attention mechanism, significantly enhancing the text rendering performance of the original diffusion model. The resultant Glyph-SDXL model showcases exceptional spelling accuracy, outperforming other state-of-the-art models in the generation of text-rich design images, as illustrated in Table 1. Furthermore, we fine-tuned Glyph-SDXL using a limited set of scene-text images, significantly bolstering its proficiency in generating scene-text images. The examples featured in Fig. 1 demonstrate that the refined model adeptly renders text paragraphs as scene text without perceptible degradation in the image generation capabilities of the original model.

Our investigation reveals that, through the training of a customized text encoder and the implementation of a suitable information injection mechanism, we can transform an open-domain image generator into an outstanding visual text renderer. When presented with a textual paragraph ranging from tens to hundreds of characters, our fine-tuned diffusion model achieves high spelling accuracy for rendering within the designated region, with fully automated handling of multi-line layouts. In essence, this work contributes in three distinct yet complementary ways. First, we train a character-aware, glyph-aligned text encoder, Glyph-ByT5, as the key solution to the accurate visual text rendering problem. Second, we elaborate on the architecture and training of Glyph-SDXL, a robust design image generator that integrates Glyph-ByT5 into SDXL through an efficient region-wise cross-attention mechanism. Lastly, we showcase the potential of fine-tuning Glyph-SDXL into a scene-text image generator, laying the groundwork for the development of a comprehensive, open-domain image generator equipped with exceptional visual text rendering capabilities.

2 Related Work

2.1 Visual Text Rendering

Rendering legible and visually coherent text poses a well-known limitation and a significant challenge for diffusion-based image generation models. It is worth noting that certain contemporary open-domain image generation models, such as Stable Diffusion 3 [9] and Ideogram 1.0[1], have dedicated considerable effort to enhance visual text rendering performance. However, the spelling accuracy of the rendered text remains unsatisfactory. Conversely, there have been endeavors focused on visual text rendering, such as GlyphControl, GlyphDraw, and the TextDiffuser series [5, 6, 15, 17, 28]. While these efforts have shown substantial improvements in spelling accuracy, it is disappointing to note that they are still focusing on rendering single words or text lines with fewer than approximately 20 characters. In this study, we aim to tackle the precise visual text rendering problem, particularly when dealing with textual content longer than a hundred characters, setting forth an ambitious goal in this domain.

2.2 Customized Text Encoder

Several recent efforts [4, 11, 31] have been made to train text-oriented diffusion models and replace or augment the original CLIP encoders with customized text encoders in different manners. However, these methods, like their predecessors, are limited to handling text sequences of a certain length, with UDiffText [31] supporting sequences of no more than 12 characters. In contrast, our methodology distinguishes itself by its ability to generate text sequences of more than 100 characters while achieving exceptionally high accuracy, reaching nearly 90% word-level accuracy. This significant progress addresses the shortcomings of previous methods, providing wider applicability and improved performance in text generation tasks. Another closely related work is Counting-aware CLIP [20], which enhances the original CLIP text encoder with a specialized image-text counting dataset and a counting-focused loss function. However, a significant limitation of their approach is the lack of scalability in their dataset. They choose to replace the original text encoders and train diffusion models from scratch, whereas our data construction pipeline is scalable, and we prioritize integrating GlyphByT5 with the original text encoders to improve efficiency.

Our Contribution Our work aligns with the insights of the previously mentioned studies, identifying that one critical limitation in most current text-to-image generation models resides in the text encoder. The primary contribution

[1] `https://about.ideogram.ai/1.0`

of our work lies in presenting an effective strategy for systematically addressing the glyph rendering task. We first demonstrate that leveraging graphic rendering to create scalable and accurate glyph-text data is crucial for training a high-quality, glyph-aligned, character-aware text encoder. Then, we introduce a simple yet powerful method to integrate our Glyph-ByT5 text encoder with the original CLIP text encoder used in SDXL. Additionally, we illustrate how our approach can be applied to scene-text generation by performing design-to-scene alignment fine-tuning. We anticipate that training the customized text encoder on scalable, high-quality data represents a promising avenue for overcoming fundamental limitations, such as spatial awareness and numeracy.

3 Our Approach

We begin by illustrating the details of our customized glyph-aligned, character-aware text encoder, Glyph-ByT5, which is trained using a substantial dataset of paired glyph images and textual instructions. Subsequently, we demonstrate how Glyph-ByT5 significantly enhances the visual text rendering accuracy when integrated with the SDXL models for the design-text rendering task. Finally, we introduce a straightforward yet effective approach for design-to-scene alignment, enabling the adaptation of Glyph-SDXL for precise scene-text generation.

3.1 Glyph-ByT5: Customized Glyph-Aligned Character-Aware Text Encoder for Design-text Generation

A key factor contributing to inaccuracies in text rendering is the inherent limitations of text encoders in modern diffusion models, especially regarding their interpretation of glyph images. The original CLIP text encoder, for example, is tailored for broad visual-language semantic alignment at the conceptual level, while the T5/ByT5 text encoder focuses on deep language understanding. However, neither is explicitly fine-tuned for glyph image interpretation although the recent works show that T5/ByT5 text encoder is favorable for visual text rendering task. This lack of customized text encoder design can result in less accurate text rendering in various applications.

To bridge the gap between existing text encoders (such as the CLIP text encoder or the T5/ByT5 text encoder) and glyph images, we propose a innovative glyph-alignment methodology for training a series of glyph-aligned character-aware text encoders, i.e., Glyph-ByT5. Our approach is focused on training a series of glyph-aware text encoders, specifically designed to reconcile the disparity between glyph images and text. Drawing inspiration from the LiT framework [29], our strategy involves exclusively fine-tuning the text models while maintaining the pre-trained image models frozen. This approach effectively compels the text encoders to adapt, learning to identify the rich information encoded within the visual glyph representations extracted from the already trained image model. For the vision encoder component, we opt for the pre-trained CLIP vision encoders or the DINOv2 models, leveraging their advanced capabilities in

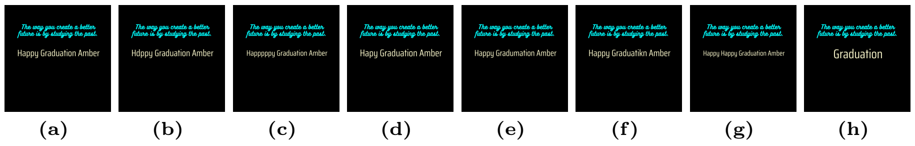


Fig. 2: Illustrating the scheme of glyph augmentation. (a) original glyph. (b) character replacement (Happy → Hdppy). (c) character repeat (Happy → Happpppy). (d) character drop (Happy → Hapy). (e) character add (Graduation → Gradumation). (f) word replacement (Graduation → Gauatikn). (g) word repeat (Happy → Happy Happy). (h) word drop (Happy Graduation Amber → Graduation).

Fig. 3: Illustrating the example images with paragraph visual text in our Paragraph-Glyph-Text dataset. From left to right, # of words: 55, 64, 52, 46, 34, 35, 40, 43; # of characters: : 443, 442, 416, 318, 247, 267, 282, 302.

handling visual data. We also explore the impact of employing vision encoders specifically tailored for scene text recognition or other tasks, and we consider the development and training of more advanced vision encoders for visual text rendering as a future avenue of research.

Creating Scalable and Accurate Glyph-Text Dataset To enable the training of the customized glyph-aware text encoder, we first create a high-quality glyph-text dataset, denoted as $\mathcal{D}$, consisting of approximately ~ 1 million pairs of synthetic data $\{\mathsf{I}_{\text{glyph}}, \mathsf{T}_{\text{text}}\}$. This dataset was developed with the improved graphic render introduced in the recent work by [12]. We construct the initial glyph image set based on the original typographic attributes (including font types, colors, sizes, positions, and others) found in the crawled graphic design images. We compile a large text corpus that can be used to enrich the glyph image set by replacing the words with random text sampled from the corpus. Additionally, we randomly modify the font types and colors within each text box to further enlarge the dataset. Our glyph-text dataset $\mathcal{D}$ encompasses nearly ~ 512 different font types and ~ 100 distinct font colors. To ensure the glyph-aligned text encoder focuses on only the difference on the visual text, we all use black colored background by default.

We present the example of glyph prompts corresponding to the glyph image shown in Figure 2 (a), detailing font types, colors, and text, as illustrated follows: {Text "The way you create a better future is by studying the past." in [font-color-127], [font-type-234]. Text "Happy Graduation Amber" in [font-color-98] [font-type-231]}. In this process, special tokens are utilized to denote font colors and types. Prior to inputting it into the Glyph-ByT5 text encoder, we preprocess the prompt text by substituting special tokens, like the token '[font-color-127]', with a series of global embeddings from the enriched codebook. We have conducted experiments on the Glyph-Text datasets at three distinct scales, expanding from 100K to 500K, and up to 1M. In the future, we aim to significantly expand our datasets, scaling up to 100M given access to more computing resources.

Creating Paragraph-Glyph-Text Dataset To enhance both the generation quality of small-sized fonts and the paragraph-level layout planning capability of customized text encoder, we have additionally compiled a dense-and-small paragraph-level glyph-text dataset, denoted as $\mathcal{D}^{\text{paragraph}}$.

We define a 'paragraph' as a block of text that cannot be accommodated within a single line, typically consisting of more than 10 words or 100 characters. The paragraph-glyph rendering task poses a greater challenge, as it demands not only very high word-level spelling accuracy but also meticulous planning of word-level and line-level layouts within the specified box. This dataset is comprised of 100,000 pairs of synthetic data $\{\mathsf{I}_{\text{glyph}}, \mathsf{T}_{\text{text}}\}$. Empirical findings suggest that fine-tuning on $\mathcal{D}^{\text{paragraph}}$ markedly improves performance in rendering small-sized and paragraph-level visual text.

The capability for paragraph-level layout planning is non-trivial, and we empirically demonstrate that the diffusion model can effectively plan multi-line arrangements and adjust the line or word spacing according to the given text box, regardless of its size or aspect ratios. We display example images of the paragraph glyph-text data in Figure 3, illustrating that each image contains at least one text box with more than 100 characters. Some images even reach 400 characters, arranged into multiple lines with reasonable spacing. We also construct three scales of the paragraph-glyph-text datasets, comprising 100K, 500K, and 1M glyph-text pairs.

Glyph Augmentation Unlike conventional CLIP models, which only consider different glyph-text pairs as negative samples-thereby modeling only the relatively high-level differences caused by multiple words or even paragraphs consisting of more than 10 characters-we propose a simple yet effective character-level and word-level glyph augmentation scheme. This approach constructs more informative negative samples, significantly enhancing training efficiency.

The proposed character-level and word-level augmentation scheme essentially consist of a combination of four different glyph augmentation strategies including *glyph replacement*, *glyph repeat*, *glyph drop*, and *glyph add* at both character-level and word-level. We apply these augmentations to both $\mathsf{I}_{\text{glyph}}$ and T_{text} to ensure consistency. Figure 2 shows some representative examples with these augmentation strategies. We also investigate the effect of constructing different ratios of informative negative samples for each sample. We independently apply these augmentations to each text box. We present statistics on the number of text boxes, words, and characters across the entire glyph-text dataset and the paragraph-glyph-text dataset in the appendix.

Glyph Text Encoder To efficiently capture the text features of each character, we have selected the character-aware ByT5 [26] encoder as the default text encoder for Glyph-CLIP. The original ByT5 model features a robust, heavy encoder paired with a lighter decoder. The ByT5 encoder is initialized using the official pre-trained checkpoints from the mC4 text corpus, as mentioned in [27].

Furthermore, we explore the impact of scaling the text encoders from smaller to larger sizes. This includes the evaluation of various ByT5 models such as ByT5-Small (217M parameters), ByT5-Base (415M parameters), and ByT5-Large (864M parameters) examining their performance enhancements. To distinguish from the original ByT5 series, we refer to these text encoders as Glyph-ByT5, indicating their specialized focus on bridging the gap between glyph images and their corresponding text prompts.

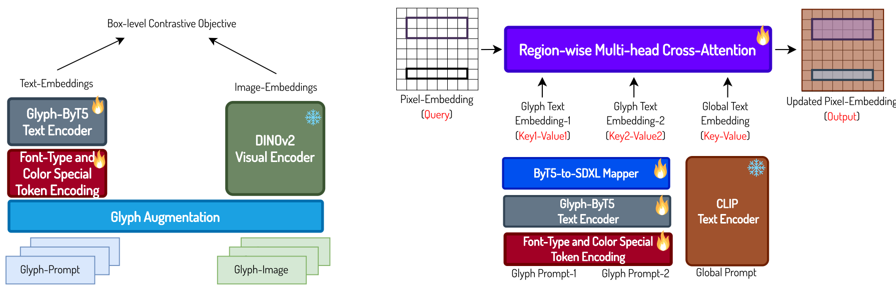
Glyph Vision Encoder For the exploration of the visual encoder, we analyzed the impact of using visual embeddings derived from CLIP [22], or DINOv2 [8, 19], or the variants [1, 30] tailored for visual text recognition task. Our observations revealed that DINOv2 yields the best performance. It was also noted that CLIP's visual embeddings struggled to distinguish between different font types. This finding aligns with recent research efforts, as discussed by [7,33], which demonstrate that DINOv2 excels in preserving identity information. As a result, DINOv2 has been chosen as our primary visual encoder. Furthermore, we explored the effect of scaling visual encoders from smaller to larger sizes on performance. This included assessing variations like ViT-B/14 (86M parameters), ViT-L/14 (300M parameters), and ViT-g/14 (1.1B parameters), aligning them with the above mentioned three ByT5 text encoders of varying scales.

Box-level Contrastive Loss Unlike conventional CLIP, which applies contrastive loss to the entire image, we propose applying a box-level contrastive loss that treats each text box and its corresponding text prompt as an instance. Based on the number of characters or words within the text box, we can categorize them into either a word text box, a sentence text box, or a paragraph text box. Therefore, our box-level contrastive loss is capable of aligning the text with glyph images at different levels of granularity. This alignment aids our customized text encoder in acquiring the capability for paragraph-level layout planning. We illustrate the mathmatical formulation as follows:

$$\mathcal{L}_{\text{box}} = -\frac{1}{2\sum_{i=1}^{|\mathcal{N}|}|\mathcal{B}_i|}\sum_{i=1}^{|\mathcal{N}|}\sum_{j=1}^{|\mathcal{B}_i|}(\log\frac{e^{t\mathbf{x}_i^j\cdot\mathbf{y}_i^j}}{Z_x}+\log\frac{e^{t\mathbf{x}_i^j\cdot\mathbf{y}_i^j}}{Z_y}), \tag{1}$$

where $\mathcal{N} = \{(\mathsf{I}_1, \mathsf{T}_1), (\mathsf{I}_2, \mathsf{T}_2), \dots\}$ represents all image-text pairs within the same batch, where the i-th image-text pair consists of $|\mathcal{B}_i|$ box-sub-text pairs. We compute the box embedding and sub-text embedding of j-th box in i-th image-text pair $(\mathsf{I}_i, \mathsf{T}_i)$ as follows: $\mathbf{x}_i^j = \mathsf{ROIAlign}(\frac{f(\mathsf{I}_i)}{\|f(\mathsf{I}_i)\|_2}, \text{box}_i^j)$ and $\mathbf{y}_i^j = \frac{g(\mathsf{T}_i^j)}{\|g(\mathsf{T}_i^j)\|_2}$. $f(\cdot)$ and $g(\cdot)$ represent the visual encoder and text encoder, respectively. We set the two normalization factors following $Z_x = \sum_{k=1}^{|\mathcal{N}|}\sum_{l=1}^{|\mathcal{B}_k|} e^{t\mathbf{x}_i^j\cdot\mathbf{y}_k^l}$ and $Z_y = \sum_{k=1}^{|\mathcal{N}|}\sum_{l=1}^{|\mathcal{B}_k|} e^{t\mathbf{x}_k^l\cdot\mathbf{y}_i^j}$. t is a learnable temperature parameter.

Hard-negative Contrastive Loss based on Glyph Augmentation: We additionally compute a contrastive loss for the hard-negative samples generated with our


Box-level Contrastive Objective
Text-Embeddings
Image-Embeddings
Glyph-ByT5 Text Encoder
Font-Type and Color Special Token Encoding
DINOv2 Visual Encoder
Glyph Augmentation
Glyph-Prompt
Glyph-Image
Pixel-Embedding (Query)
Region-wise Multi-head Cross-Attention
Glyph Text Embedding-1 (Key1-Value1)
Glyph Text Embedding-2 (Key2-Value2)
Global Text Embedding (Key-Value)
Updated Pixel-Embedding (Output)
ByT5-to-SDXL Mapper
Glyph-ByT5 Text Encoder
Font-Type and Color Special Token Encoding
CLIP Text Encoder
Glyph Prompt-1
Glyph Prompt-2
Global Prompt

(a) Glyph-Alignment Pre-training **(b)** Region-wise Multi-Text-Encoder Fusion

Fig. 4: Illustrating the glyph-alignment pre-training framework and the region-wise multi-head cross attention module

glyph augmentation and the mathematical formulatioin is shown as follows:

$$\mathcal{L}_{\text{hard}} = -\frac{1}{2\sum_{i=1}^{|\mathcal{N}|}|\mathcal{B}_i|}\sum_{i=1}^{|\mathcal{N}|}\sum_{j=1}^{|\mathcal{B}_i|}(\log\frac{e^{t\mathbf{x}_i^j\cdot\mathbf{y}_i^j}}{Z_x^{\text{aug}}} + \log\frac{e^{t\mathbf{x}_i^j\cdot\mathbf{y}_i^j}}{Z_y^{\text{aug}}}), \tag{2}$$

where $Z_x = \sum_{g=1}^{|\mathcal{G}|} e^{t\mathbf{x}_i^j\cdot\mathbf{y}_i^{j,g}}$ and $Z_x = \sum_{g=1}^{|\mathcal{G}|} e^{t\mathbf{x}_i^{j,g}\cdot\mathbf{y}_i^j}$ Here, $\mathcal{G}$ represents the augmented training data based on box $\mathbf{x}_i^j$ and sub-text $\mathbf{y}_i^j$. We investigate the impact of varying the number of augmented data points in the ablation experiments.

We combine the above two losses, i.e., $\mathcal{L}_{\text{box}} + \mathcal{L}_{\text{hard}}$, to facilitate the glyph-alignment pre-training process. We also empirically demonstrate that our design outperforms the image-level contrastive loss in the ablation experiments. We attribute the superior performance to two main factors: the availability of a significantly larger number of effective training samples, and the box-level visual features providing more accurate visual text information. These assertions are corroborated by the findings in two prior studies [3, 32]. Figure 4 depicts the complete framework of Glyph-ByT5, showcasing its glyph-alignment pre-training process that integrates the critical components previously mentioned.

3.2 Glyph-SDXL: Augmenting SDXL with Glyph-ByT5 for Design Image Generation

To verify the effectiveness of our approach in generating accurate text contents in design images and planning visual paragraph layouts within each text box, we integrate our Glyph-ByT5 with the state-of-the-art, open-sourced text-to-image generation model, SDXL [21]. The primary challenge lies in integrating our customized encoder with the existing one to harness the strengths of both without detracting from the original performance. Another challenge is the lack of high-quality graphic design datasets with coherent background image layers.

To address the two mentioned challenges, we first introduce a region-wise multi-head cross-attention mechanism to seamlessly fuse the glyph knowledge encoded in our customized text encoder within the target typography boxes and the prior knowledge carried by the original text encoders in background regions. Additionally, we build a high-quality graphic design dataset to train our Glyph-SDXL model for accurate visual text rendering. Detailed discussions of these two pivotal contributions are provided in subsequent sections.

Region-wise Multi-head Cross-Attention The original multi-head cross-attention is the core component responsible for mapping the rich semantic information of text-space into different positions in the image-space. In other words, it determines generate what object at where by continuely applying multi-head cross-attention across different layers and time steps.

The detailed framework of the region-wise multi-head cross-attention is displayed on the right side of Figure 4. In this mechanism, we first partition input pixel embeddings (Query) into multiple groups. These groups correspond to the target text boxes, which can be either specified by the user or automatically predicted by leveraging the planning capability of GPT-4. Simultaneously, we divide the text prompts (Key-Value) into corresponding sub-sections, which include a global prompt and several groups of glyph-specific prompts. We then specifically direct the pixel embeddings within the target text boxes to attend only to the glyph embeddings extracted with Glyph-ByT5. Similarly, pixel embeddings outside the text boxes are made to attend exclusively to the global prompt embeddings extracted with the original CLIP encoders.

To close the gap between the output embedding space of Glyph-ByT5 with the original SDXL space, we introduce a lightweight mapper, namely the ByT5-to-SDXL mapper. This mapper is equipped with four randomly initialized ByT5 encoder layers and applied to the output of Glyph-ByT5. For efficiency, we implement the above-mentioned region-wise multi-head cross-attention by modulating the attention maps with a mask that ensures the mapping relations between pixel embeddings and the multiple text encoder embeddings. We fine-tune the weights of both text encoder and mapper modules during training, in line with previous research [15] which highlights that refining a character-aware text encoder within a diffusion model can significantly enhance performance.

Visual Design Dataset for Design-text Generation It is important to choose a reliable task to access design-text rendering performance. This work selects the design image generation as this is one of the most representative text-intensive generation task. Therefore, we first build a high-quality visual design image dataset with dense paragraph-level visual text rendered on each image by crawling from graphic design websites following [12]. This task presents two significant challenges, as it demands not only the generation of dense visual text but also necessitates visually appealing background images. We also create three versions of the graphic design datasets, encompassing sizes of 100K, 500K, and 1M, where we utilize LLaVA [14] based on Llama2-13B [24] to generate detailed captions, with the ground-truth glyph text readily accessible. We also conduct data cleaning to ensure that few graphic design images share the same typography as the glyph-text images used for glyph-alignment pre-training.

Glyph-SDXL We train the Glyph-SDXL on the above constructed design-text dataset. To preserve the inherent capabilities of SDXL, we lock the entire model's weights. First, we implement LoRA [10] module exclusively on the UNet components. Second, we introduce a region-wise multi-text-encoder fusion mechanism designed to integrate the glyph-aware capabilities of the Glyph-ByT5 text encoder with the formidable strengths of the two original CLIP text encoders. This approach aims to synergize the unique features of each text encoder, enhancing visual text rendering performance. In implementation, we modify the original multi-head cross-attention module accordingly.

Our tailored Glyph-ByT5 matches the rendering accuracy of conventional tools while leveraging the capabilities of fully diffusion-based models. This allows it to tackle scene-text generation tasks which are beyond the capabilities of standard rendering tools.

3.3 Design-to-Scene Alignment: Fine-tuning Glyph-SDXL for Scene-text Generation

The previous constructed Glyph-SDXL, which mainly trained on design images, encounters difficulties in producing coherent scene texts. Furthermore, we notice a 'language drift' phenomenon, which slightly undermines the model's original proficiency. To tackle these issues and create a superior scene text generation model, we develop a hybrid design-to-scene alignment dataset combining three types of high-quality data: 4,000 scene and design text images from TextSeg [25], 4,000 synthetic images generated by SDXL, and 4,000 design images. We fine-tune Glyph-SDXL on the hybrid design-to-scene alignment dataset for 2 epochs and conduct thorough evaluations of the scene-text rendering capability across three public benchmarks and report significant performance gains compared to previous state-of-the-art methods. To distinguish it from the original Glyph-SDXL, we designate the fine-tuned version as Glyph-SDXL-Scene. Additionally, we demonstrate that each subset is useful for three combined purposes: coherent layout, accurate text rendering, and visual quality, as detailed in the appendix.

4 Experiment

We assess our method's ability to generate accurate design text in graphic design images, which often feature numerous paragraph-level text boxes, as well as scene text within photorealistic images. To facilitate the assessment of paragraph-level visual text rendering, we develop the VisualParagraphy benchmark, which includes multi-line visual text within boxes of diverse aspect ratios and scales.

Our evaluation compares our method against commercial products and the most advanced visual text rendering techniques, such as DALL·E, in the design-text generation task. We report objective OCR metrics and conduct a subjective user study to evaluate visual quality from other aspects. For the scene-text generation task, we compare our method with the representative models GlyphControl [28] and TextDiffuser-2 [5] across three public benchmarks.

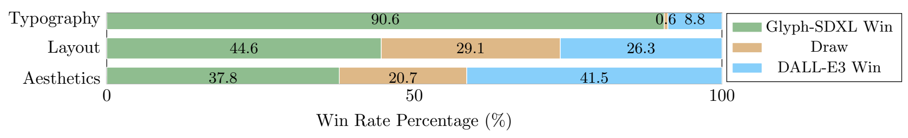


Fig. 5: Glyph-SDXL v.s. DALL·E3 Win Rate Percentage.

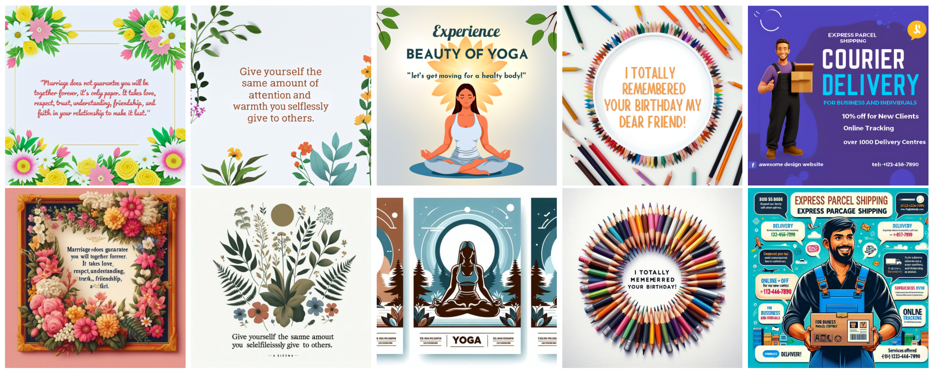

Give yourself the same amount of attention and warmth you selflessly give to others.
Experience
BEAUTY OF YOGA
I TOTALLY REMEMBERED YOUR BIRTHDAY MY DEAR FRIEND!
COURIER
DELIVERY
EXPRESS PARCEL SHIPPING
YOGA
I TOTALLY MEMEMRRED YOUR BIRTHDAY!

Fig. 6: Qualitative comparison results. We show the results generated with our Glyph-SDXL and DALL·E3 in the first row and second row, respectively.

Additionally, we conduct thorough ablation experiments to study the effect of each component of our approach and visualize the cross-attention maps to demonstrate that our customized text encoder can provide a glyph prior to diffusion models. We detail training settings and additional results in the appendix.

4.1 Metrics

In the majority of our experiments, we default to reporting case-sensitive word-level precision, except for comparisons involving GlyphControl and TextDiffuser. In these instances, we align with their original methodologies by reporting case-agnostic metrics and image-level metrics. For instance, as indicated in Table 2, Case-Recall is used as a case-sensitive metric to differentiate between uppercase and lowercase letters. Conversely, all other metrics are case-agnostic. Accuracy [IMG] is utilized to denote image-level accuracy, which depends on the accurate spelling of every visual word within the entire image to achieve a favorable evaluation. Furthermore, we identified a direct correspondence between the OCR Accuracy metric in GlyphControl and the Recall metric in TextDiffuser. As a result, to ensure consistency in metrics reporting for both SimpleBench and CreativeBench, we unify the approach by selecting Recall as the principal metric.

4.2 VisualParagraphy Benchmark

We construct a benchmark for design-text generation task, amassing approximately $\sim 1,000$ prompts covering varying number of characters with different

Method	SimpleBench			CreativeBench			MARIO-Eval			
	Recall	Case-Recall	Edit-Dis.	Recall	Case-Recall	Edit-Dis.	Accuracy [IMG]	Precision	Recall	F-measure
DeepFloyd IF [13]	0.6	33	1.63	1	21	3.09	2.6	14.5	22.5	17.6
GlyphControl [28]	42	48	1.43	28	34	2.40	-	-	-	-
TextDiffuser [6]	-	-	-	-	-	-	56.1	78.5	78.0	78.2
TextDiffuser-2 [5]	-	-	-	-	-	-	57.6	74.0	76.1	75.1
Glyph-SDXL	**93.56**	93.62	0.09	**92.00**	**92.06**	0.16	**74.8**	**88.2**	**92.6**	**90.4**
Glyph-SDXL-Scene	92.69	**95.88**	**0.05**	88.81	91.38	**0.15**	66.5	83.9	89.0	86.4

Table 2: Comparison on SimpleBench, CreativeBench, and MARIO-Eval.

Visual encoder	Precision (%)			
	≤20 chars	≤20-50 chars	≤50-100 chars	≥100 chars
DINOv2 ViT-B/14 + reg	**84.54**	**84.56**	**79.89**	**73.29**
CLIP ViT-B/16	77.17	74.78	74.94	66.34
ViTSTR	79.29	78.2	75.35	68.49
CLIP4STR ViT-B/16	80.38	79.12	77.08	69.24

Table 3: Effect of using different pre-trained visual encoder.

ByT5-to-SDXL mapper	Precision (%)			
	≤20 chars	≤20-50 chars	≤50-100 chars	≥100 chars
w/o mapper	80.22	78.48	72.91	65.02
w/ mapper	**84.54**	**84.56**	**79.89**	**73.29**

Table 4: Effect of the ByT5-to-SDXL mapper within Glyph-SDXL.

# Glyph Image-Text	Precision (%)			
	≤20 chars	≤20-50 chars	≤50-100 chars	≥100 chars
100K	85.6	85.02	81.2	74.58
500K	91.11	93.35	85.43	82.83
1M	**93.54**	**93.96**	**91.0**	**89.96**

Table 5: Effect of scaling the training data for Glyph-ByT5 and Glyph-SDXL.

Text encoder	#Params	Precision (%)			
		≤20 chars	≤20-50 chars	≤50-100 chars	≥100 chars
Glyph-ByT5-S	292M	84.54	84.56	**79.89**	73.29
Glyph-ByT5-B	510M	**87.10**	**84.93**	78.72	72.81
Glyph-ByT5-L	963M	87.07	82.87	79.12	**73.72**

Table 6: Effect of scaling customized text encoder model scales.

difficulty, rendering less than 20, 20 to 50, 50 to 100, and more than 100 characters respectively. We provide some representative examples in the appendix. We use approximately 1,000 design-text prompts in the comparison with DALL·E3, while by default, a smaller subset of approximately 400 design-text prompts are used in all subsequent ablation experiments for efficiency.

4.3 Comparison to Commercial-Product DALL·E3

We compare our approach with the most powerful commercial product in the visual text rendering task, namely, DALL·E3 on VisualParagraphy benchmark. We conducted a user study to assess the results from three critical aspects: visual aesthetics, layout quality, and typography accuracy. We hired 10 users with design backgrounds to rank the results from these aspects and report win-rate results in Figure 5. We conclude that Glyph-SDXL is significantly preferred in terms of typography and comparable or slightly lower on other aspects. Additionally, we visualize representative comparison results in Figure 6. We find that our approach demonstrates significant advantages in design-text rendering. We further improve the visual aesthetics in the follow-up work [16].

4.4 Comparison to State-of-the-Art

Our foremost goal was to confirm the broad applicability of our visual text generation model. To this end, we carefully detail outcomes obtained by applying our method to the representative scene-text rendering benchmarks outlined in earlier research, such as TextDiffuser 1 and 2 [5,6], and GlyphControl [28]. This encompassed comprehensive testing on benchmarks like MARIO-Eval, SimpleBench, and CreativeBench. The comparison results are summarized in Table 2. As indicated, it is evident that our Glyph-SDXL-Scene significantly outperforms the previous state-of-the-art by a substantial margin across these three benchmarks. All of the results of our method represent zero-shot performance.

4.5 Ablation Experiments

We carry out all ablation studies by initially undertaking glyph-alignment pre-training, followed by training the Glyph-SDXL model on our graphic design benchmarks. Furthermore, all ablations are carried out on 100K glyph image-text pairs for Glyph-ByT5 and Glyph-SDXL models respectively unless specified.
Pre-trained Visual Encoder Choice We study the effect of choosing four different pre-trained visual encoders: CLIP visual encoder [22], DINOv2 [8], ViTSTR [1], and CLIP4STR visual encoder [30]. We report the detailed comparison results in Table 3. Notably, we also observe that accurate font type and color controls only occur when using DINOv2 as the pre-trained visual encoder.
Mapper, Loss Design, Glyph Augmentation, Scaling and More Table 4 shows the importance of using the ByT5-to-SDXL mapper to align the gap. Table 5 and Table 6 verify the benefits of scaling up the glyph-text dataset size and text encoder size. We provide more ablations, as well as experiments of Glyph-SDXL-Scene in the appendix.

5 Conclusion

This paper presents the design and training of the Glyph-ByT5 text encoder, tailored for accurate visual text rendering with diffusion models. The two key contributions are: the creation of a scalable, high-quality glyph-text dataset and the implementation of pre-training techniques for glyph-text alignment. These designs efficiently bridge the gap between glyph imagery and text prompts, facilitating the generation of accurate text for both text-rich design images and open-domain images with scene text. The compelling performance achieved by our proposed Glyph-SDXL model suggests that the development of specialized text encoders represents a promising avenue for overcoming some of the fundamental challenges associated with diffusion models.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under Grants 62321005 and 42327901.

References

1. Atienza, R.: Vision transformer for fast and efficient scene text recognition. In: International Conference on Document Analysis and Recognition. pp. 319–334. Springer (2021)
2. Betker, J., Goh, G., Jing, L., Brooks, T., Wang, J., Li, L., Ouyang, L., Zhuang, J., Lee, J., Guo, Y., Manassra, W., Dhariwal, P., Chu, C., Jiao, Y., Ramesh, A.: Improving image generation with better captions (2023), https://cdn.openai.com/papers/dall-e-3.pdf
3. Bica, I., Ilić, A., Bauer, M., Erdogan, G., Bošnjak, M., Kaplanis, C., Gritsenko, A.A., Minderer, M., Blundell, C., Pascanu, R., et al.: Improving fine-grained understanding in image-text pre-training. arXiv preprint arXiv:2401.09865 (2024)
4. Chen, H., Xu, Z., Gu, Z., Lan, J., Zheng, X., Li, Y., Meng, C., Zhu, H., Wang, W.: Diffute: Universal text editing diffusion model
5. Chen, J., Huang, Y., Lv, T., Cui, L., Chen, Q., Wei, F.: Textdiffuser-2: Unleashing the power of language models for text rendering. arXiv preprint arXiv:2311.16465 (2023)
6. Chen, J., Huang, Y., Lv, T., Cui, L., Chen, Q., Wei, F.: Textdiffuser: Diffusion models as text painters. arXiv preprint arXiv:2305.10855 (2023)
7. Chen, X., Huang, L., Liu, Y., Shen, Y., Zhao, D., Zhao, H.: Anydoor: Zero-shot object-level image customization. arXiv preprint arXiv:2307.09481 (2023)
8. Darcet, T., Oquab, M., Mairal, J., Bojanowski, P.: Vision transformers need registers (2023)
9. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis (2024)
10. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)
11. Ji, J., Zhang, G., Wang, Z., Hou, B., Zhang, Z., Price, B., Chang, S.: Improving diffusion models for scene text editing with dual encoders. arXiv preprint arXiv:2304.05568 (2023)
12. Jia, P., Li, C., Liu, Z., Shen, Y., Chen, X., Yuan, Y., Zheng, Y., Chen, D., Li, J., Xie, X., et al.: Cole: A hierarchical generation framework for graphic design. arXiv preprint arXiv:2311.16974 (2023)
13. Lab, D.: Deepfloyd if. https://github.com/deep-floyd/IF (2023)
14. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. Advances in neural information processing systems **36** (2024)
15. Liu, R., Garrette, D.H., Saharia, C., Chan, W., Roberts, A., Narang, S., Blok, I., Mical, R.J., Norouzi, M., Constant, N.: Character-aware models improve visual text rendering. In: Annual Meeting of the Association for Computational Linguistics (2022), https://api.semanticscholar.org/CorpusID:254877579
16. Liu, Z., Liang, W., Zhao, Y., Chen, B., Li, J., Yuan, Y.: Glyph-byt5-v2: A strong aesthetic baseline for accurate multilingual visual text rendering. arXiv preprint arXiv:2406.10208 (2024)
17. Ma, J., Zhao, M., Chen, C., Wang, R., Niu, D., Lu, H., Lin, X.: Glyphdraw: Learning to draw chinese characters in image synthesis models coherently. arXiv preprint arXiv:2303.17870 (2023)

18. OpenAI: Dall·e 3 system card (2023), `https://cdn.openai.com/papers/DALL_E_3_System_Card.pdf`
19. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)
20. Paiss, R., Ephrat, A., Tov, O., Zada, S., Mosseri, I., Irani, M., Dekel, T.: Teaching clip to count to ten. arXiv preprint arXiv:2302.12066 (2023)
21. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)
22. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
23. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
24. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)
25. Xu, X., Zhang, Z., Wang, Z., Price, B., Wang, Z., Shi, H.: Rethinking text segmentation: A novel dataset and a text-specific refinement approach. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12045–12055 (2021)
26. Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., Raffel, C.: Byt5: Towards a token-free future with pre-trained byte-to-byte models. Transactions of the Association for Computational Linguistics **10**, 291–306 (2022)
27. Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., Raffel, C.: mt5: A massively multilingual pre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934 (2020)
28. Yang, Y., Gui, D., Yuan, Y., Liang, W., Ding, H., Hu, H., Chen, K.: Glyphcontrol: Glyph conditional control for visual text generation. Advances in Neural Information Processing Systems **36** (2024)
29. Zhai, X., Wang, X., Mustafa, B., Steiner, A., Keysers, D., Kolesnikov, A., Beyer, L.: Lit: Zero-shot transfer with locked-image text tuning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18123–18133 (2022)
30. Zhao, S., Wang, X., Zhu, L., Yang, Y.: Clip4str: A simple baseline for scene text recognition with pre-trained vision-language model. arXiv preprint arXiv:2305.14014 (2023)
31. Zhao, Y., Lian, Z.: Udifftext: A unified framework for high-quality text synthesis in arbitrary images via character-aware diffusion models. arXiv preprint arXiv:2312.04884 (2023)
32. Zhong, Y., Yang, J., Zhang, P., Li, C., Codella, N., Li, L.H., Zhou, L., Dai, X., Yuan, L., Li, Y., et al.: Regionclip: Region-based language-image pretraining. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16793–16803 (2022)
33. Zhou, Y., Zhang, R., Gu, J., Sun, T.: Customization assistant for text-to-image generation. arXiv preprint arXiv:2312.03045 (2023)