GlyphAnchor: Enhancing Visual Text Rendering via Position-Anchored Glyph Priors

Qiang Xiang[1,2], **Shuang Sun**[2], **Binglei Li**[1,3],
Yibo Chen[2], **Xu Tang**[2], **Yao Hu**[2], **Junping Zhang**[1]*
[1]Fudan University [2]Xiaohongshu Inc. [3]Shanghai Innovation Institute
{qxiang24, blli24}@m.fudan.edu.cn, jpzhang@fudan.edu.cn,
{sunshuang1, zhaohaibo, tangshen, xiahou}@xiaohongshu.com



Figure 1: **Generation results of GlyphAnchor.**

Abstract

Rendering accurate text remains difficult for image generation and editing models, especially when the target contains long, complex, and densely arranged text or rare characters. Existing approaches either improve native text rendering through stronger backbones and data-centric training without explicit glyph priors, or incorporate glyph priors through specialized designs that remain insufficiently accurate and robust under challenging scenarios. We introduce GlyphAnchor, a novel text-rendering enhancement method for both text-to-image and image-editing diffusion transformer models. GlyphAnchor enhances the backbone with lightweight glyph patch conditions whose positions are anchored to the target image through the model's native positional encoding. We train this capability with staged supervised finetuning and further refine it with text-aware post-training to improve robustness. We also introduce InfoTextBench, a benchmark for evaluating text-rich visual text rendering in both generation and editing settings. Experiments across multiple backbones and benchmarks, including long, complex, and densely arranged text and rare character scenarios, show that GlyphAnchor consistently improves text fidelity while preserving overall image quality. Code and models are publicly available at https://github.com/FireRedTeam/GlyphAnchor.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

Accurate visual text rendering has become a key bottleneck for modern text-to-image and image-editing foundation models. Recent image diffusion models [42, 50, 36, 37, 1] can synthesize high-quality scenes and follow complex instructions, but their text rendering remains less reliable for long, complex, and densely arranged text and rare characters. These errors are especially damaging in posters, notes, infographics and other text-rich applications, where a single missing or wrong character can compromise usability. This challenge arises in both text-to-image and image editing, and it involves three coupled requirements: the model must render the correct character shapes, place each text item at its intended location, and preserve the surrounding style and image quality.

Previous work has made substantial progress along several directions. Glyph-prior methods incorporate rendered glyph images along with masks, glyph-aware encoders, or style controls into generation and editing models [24, 46, 40, 39, 21, 48, 41, 45, 17, 49, 44]. These methods show that explicit glyph information is useful for improving text accuracy. Foundation models and data-centric methods [19, 42, 50, 36, 37, 1] improve native text rendering through stronger backbones, large-scale curated data, or character-aware training. Other methods [6, 7, 32, 30, 53, 31] use layout-guided control, post-hoc correction, attention control, or reward-based optimization to improve text accuracy and repair generated text. Despite these advances, accurate visual text rendering remains difficult for long, complex, and densely arranged text and rare characters.

To address this, we introduce **GlyphAnchor**, a method that improves visual text rendering with glyph priors. GlyphAnchor organizes glyph priors as compact glyph patch images and anchors each patch to its intended location in the target image through RoPE positions [34]. Unlike methods that rely on custom attention masks, this design leaves the attention pattern unchanged and remains compatible with standard attention acceleration kernels such as FlashAttention [10]. It provides the backbone with explicit character-shape information and spatial cues, while keeping the conditioning mechanism lightweight and easy to adapt to different diffusion-transformer backbones. We train this representation with staged supervised finetuning, which gradually transitions from ground-truth cropped patches to rendered glyph patches that match inference-time inputs. We then apply text-aware DiffusionNFT post-training [52] to improve stability and visual quality in challenging cases.

Experiments show that GlyphAnchor consistently improves the corresponding backbones in challenging text-rendering scenarios, including long, complex, densely arranged text and rare characters. To evaluate text-rich scenarios more directly, we build **InfoTextBench**, which measures word-level accuracy, phrase hit rate, aesthetics, and prompt following for both image editing and text-to-image settings. On LongTextBench [13], OneIGBench-text [5], CVTG-2K [35], ChineseWord [42], and InfoTextBench, GlyphAnchor consistently improves text accuracy, phrase hit rate, and overall visual quality relative to the corresponding backbones. Our main contributions are listed below.

1. We leverage glyph priors through a position-anchored design that provides glyph guidance with few extra tokens and is easy to adapt across diffusion-transformer backbones.

2. We design a dedicated training recipe for position-anchored glyph priors, including staged supervised finetuning, text-aware rewards for NFT post-training, and train-time glyph augmentation.

3. We build InfoTextBench for text-rich visual text rendering evaluation and show consistent improvements over the corresponding base models on benchmarks covering long, complex, and densely arranged text and rare characters.

2 Related Work

Generation and editing with glyph priors. Many works improve visual text rendering by incorporating explicit glyph priors, such as rendered glyph images or glyph-aware text encoders, into the generation backbones. Some methods [24, 46, 40, 39, 25, 23, 43, 29] inject rendered glyph images, masks, positions, or text attributes as auxiliary conditions for generation. Another line [51, 20, 21] improves text rendering by designing character-aware or glyph-aligned text encoders; and Glyph-ByT5-v2 is later adopted by foundation models such as Seedream 2.0 [14] and GLM-Image [47]. More recent work explores inference-time glyph injection: FreeText [49] injects spectral glyph priors at inference time, and GlyphBanana [44] injects glyph templates into the latent space and attention maps through an agentic workflow. For text editing, prior methods [48, 41, 45, 18, 17, 11] improve

controllability through structure-style disentanglement, dedicated glyph encoders, or segmentation-based font control.

Foundation models and data-centric methods. Another line emphasizes stronger foundation backbones, large-scale curated data, or character-aware training, without relying on an explicit inference-time glyph condition. Liu et al. [19] argues that character-level text representations are key to improving spelling accuracy. Qwen-Image [42], Z-Image [36], LongCat-Image [37], ERNIE-Image [1], and X-Omni [13] demonstrate that large-scale foundation model training can achieve strong native text rendering and editing without any dedicated glyph module. LeX-Art [50] highlights the importance of data synthesis quality and prompt enhancement for these approaches.

Layout-guided control, post-hoc correction, and RL. Some methods improve text rendering through layout-guided control rather than explicit glyph priors. TextDiffuser [6, 7] first predicts text layouts, converts them into segmentation masks, and conditions the diffusion model on these masks for accurate text placement. DCText [32] and TextCrafter [35] instead localize text regions during inference and handle them with attention mask binding or separate denoising. Type-R [30] focuses on post-hoc correction by automatically detecting and repairing typographic errors in generated images. Other methods [53, 31] optimize rendering quality through structural anomaly quantification or region-level DPO with local glyph-correctness annotations.

GlyphAnchor is complementary to these methods, focusing on how to leverage glyph priors in modern diffusion-transformer backbones.

3 Method

To improve text rendering in modern text-to-image and image-editing foundation models, we incorporate explicit glyph priors that provide character-shape guidance beyond prompt tokens. Our goal is to inject these priors into diffusion-transformer backbones in a lightweight and seamless manner while preserving generation quality.

3.1 Problem Formulation

We consider both text-to-image and image editing models. The input contains a prompt p and an optional set of input images $\mathcal{I}_{\text{src}} = \{I_{\text{src}}\}$. The output image should faithfully follow the prompt and render a set of target texts $\mathcal{S} = \{s_i\}_{i=1}^{N}$ specified in the prompt. The texts may be short labels, long sentences, dense information blocks, or mixed Chinese, English, and symbols.

3.2 Design Rationale

To incorporate glyph priors into the backbone, we consider several design choices. We use FireRed-Image-Edit-1.1, an image-editing model derived from Qwen-Image, as the backbone in this section because it performs better than Qwen-Image-Edit-2511 on the ChineseWord benchmark (Table 2).

As illustrated in Fig. 2 (a), default prompt-only generation struggles to render long and complex text. A straightforward design is to finetune the backbone with full-canvas glyph references, where all target texts are rendered on a blank canvas and fed as an additional input image (bottom left in Fig. 2). Fig. 2 (b) shows that this design can improve text accuracy, but it weakens prompt following and suppresses decorative details, background texture, and overall stylistic fidelity. It is also expensive in token overhead, because a full-canvas glyph reference has the same spatial size as the target image and therefore contributes roughly one extra image frame of visual tokens.

To reduce this overhead and remove the unnecessary blank canvas, we next consider a more compact design that renders each target text as a glyph patch image and feeds these images as standard reference image inputs. Fig. 2 (f) shows that even for long and complex text, glyph patches use far fewer tokens than full-canvas glyph references; for a 1536×1536 target canvas, even 1000 Chinese or English characters occupy only about 31% or 15% of a full image frame, respectively. However, compact glyph patches alone are not sufficient. Without explicit spatial grounding, the model can still miss text items or generate corrupted characters, as illustrated in Fig. 2 (d). We therefore introduce a text layout representation:

$$\mathcal{L} = \{(s_i, b_i)\}_{i=1}^{N}, \qquad b_i = (x_i, y_i, w_i, h_i), \tag{1}$$

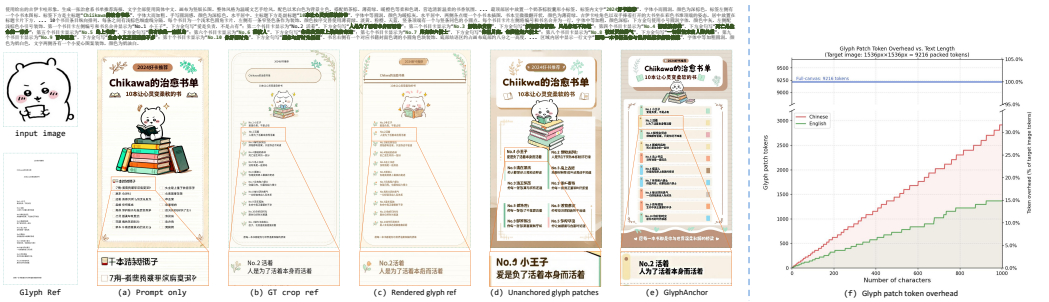


Figure 2: **Design choices for GlyphAnchor**, illustrated with FireRed-Image-Edit-1.1 examples. An English translation of the Chinese prompt is provided in Appendix A.6.

where b_i is the normalized bounding box of s_i on the target canvas. The layout may be specified by the user or obtained from predefined templates, external layout tools, or an MLLM-based layout planner. Our final design pairs compact glyph patch images with layout anchors, as shown in Fig. 2 (e). The full architecture is illustrated in Fig. 3 and detailed in Sec. 3.3.

Fig. 2 also reveals a train-test mismatch for glyph conditioning. At inference, the target image is unknown, so glyph images must always be rendered from the target texts. During training, by contrast, glyph images can instead be constructed from crops of the target image, which provide stronger supervision and lead to superior text accuracy. However, this creates a train-test gap and biases the model toward copy-paste behaviour, degrading stylization, as reflected in Fig. 2 (b). Training directly with rendered glyph images avoids this gap, but provides weaker supervision and leads to lower text accuracy, as illustrated in Fig. 2 (c). These observations motivate our staged training strategy, as described in Sec. 3.4.

3.3 GlyphAnchor

Based on the above design rationale, **GlyphAnchor** adopts position-anchored glyph priors as follows. For each text item (s_i, b_i), we construct a glyph condition

$$c_i = (g_i, b_i), \qquad g_i = E(G_i), \tag{2}$$

where G_i is a glyph patch image and $E(\cdot)$ denotes VAE encoding into latent tokens. During inference, G_i is rendered from s_i using the aspect ratio of b_i, with fixed line height and font size.

GlyphAnchor organizes all glyph patch images in a shared condition frame, which can be viewed as a virtual glyph plane aligned with the target latent grid. Let the target latent grid have size $H \times W$. For each text item, we map its normalized layout box $b_i = (x_i, y_i, w_i, h_i)$ to the $H \times W$ grid by scaling it to the target latent resolution and aligning the center of g_i with the box center, thereby obtaining the position IDs of the glyph patch tokens.

At inference, layouts are either specified by the user or planned by an MLLM, followed by post-processing such as deduplication, overlap adjustment, and text wrapping. The glyph patch tokens are then concatenated with the standard visual token sequence, but their position IDs follow the layout anchors described above rather than the backbone's default positional assignment. We place the glyph conditions in a virtual glyph plane after the noisy target frame and any optional source-image frames. As shown in Fig. 3, these tokens are then processed jointly by the text-to-image or image-editing backbone. Because many recent diffusion-transformer image generation models [42, 36, 2] use 3D RoPE or its variants, GlyphAnchor can be easily adapted to them by only adding glyph tokens and reassigning their position IDs. For classifier-free guidance, the conditional branch receives the glyph conditions and the unconditional branch omits them.

3.4 Staged Supervised Finetuning for GlyphAnchor

We use staged supervised finetuning to adapt the backbone to GlyphAnchor. Let $z_t = (1-\sigma_t)z_0 + \sigma_t\epsilon$ denote the noisy image latent at timestep t under the flow-matching schedule, where $\epsilon \sim \mathcal{N}(0, I)$, and let $v_t = \epsilon - z_0$ be the velocity target. Given the noisy target tokens, prompt embedding, optional

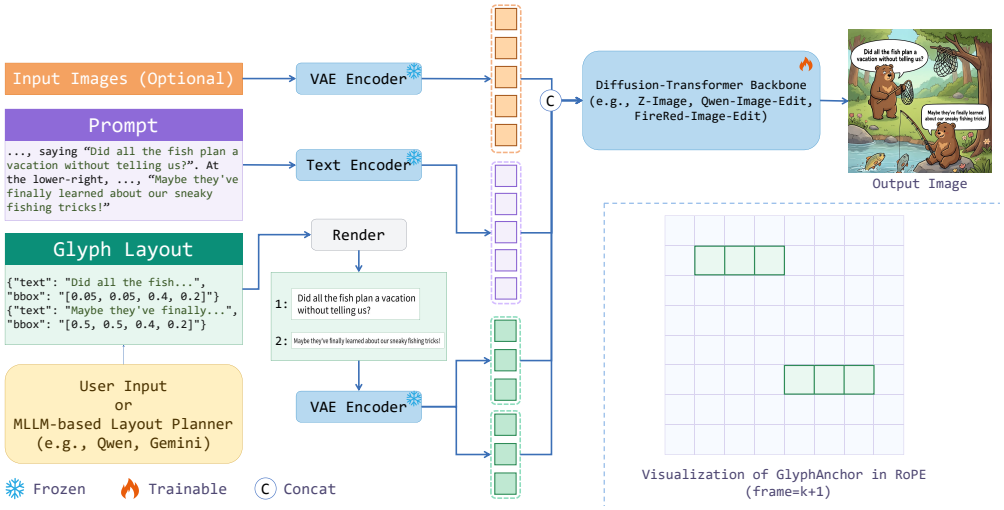


Figure 3: **Overview of GlyphAnchor**. Prompt, optional source images, and rendered glyph patch images are encoded into tokens and concatenated before a text-to-image or image-editing diffusion-transformer backbone. The resulting glyph patch tokens are placed in an additional condition frame, forming a virtual glyph plane whose coordinates are anchored to the target layout.

source-image tokens, and position-anchored glyph patch tokens, the transformer is trained with

$$\mathcal{L}_{\text{sft}} = \mathbb{E}_{\epsilon,\, z_0,\, t,\, p,\, \mathcal{I}_{\text{src}},\, \{c_i\}} \left[\left\| \hat{v}_\theta(z_t, t, p, \mathcal{I}_{\text{src}}, \{c_i\}) - v_t \right\|_2^2 \right]. \tag{3}$$

We use three types of glyph conditions during training: ground-truth image crops within b_i, rendered glyph patch images whose size follows the target box, and rendered glyph patch images with fixed line height and font size. Their sampling ratios are scheduled across training stages. Across stages, the sampling weight gradually shifts from ground-truth crops toward rendered glyph patches. This progression bridges the train-test gap. Ground-truth crops provide stronger supervision but are unavailable at inference, while rendered glyph patch images match inference-time inputs but are harder to learn from directly. During training, we further apply perturbation-based augmentation to the rendered glyph patch images, including random line breaks, resizing, rotation, translation, and dropout. These augmentations prevent the model from overfitting to the exact rendered appearance of glyph patches, while preserving their character-shape and coarse layout cues.

3.5 NFT Post-training with Text-Aware Rewards

After supervised finetuning, the model can still suffer from limited stability and a relatively high failure rate on difficult cases. We therefore adopt the DiffusionNFT framework [52] for post-training to improve stability. We further design three text-aware rewards for text correctness, style consistency, and visual fusion.

Text-Layout Consistency Reward. String-level OCR rewards, typically based on Levenshtein edit distance between recognized text and target text, can be exploited. A model may increase OCR accuracy by producing oversized characters that are easier to recognize. Also, the edit-distance objective does not penalize wrong layout in generated images. We therefore score text correctness and spatial consistency jointly. We parse the OCR outputs of the generated image and the ground-truth image into character instances with labels, locations, and scales. The textual term R_{text} is computed as one minus the normalized Levenshtein edit distance between the recognized text s_{pred} and the target text s_{tgt}. We then form a set $\mathcal{M}$ of matched character pairs between characters from the generated image and those from the ground-truth image, where each pair shares the same character label, and compute a spatial term to penalize center displacement and scale mismatch:

$$R_{\text{layout}} = \frac{\gamma_{\text{cov}}}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \exp(-\lambda d_i - \alpha \Delta s_i), \qquad \gamma_{\text{cov}} = \frac{|\mathcal{M}|}{|s_{\text{tgt}}|}. \tag{4}$$

Here γ_{cov} is a character-level coverage term, d_i is the normalized distance between the i-th matched character pair, and Δs_i measures the difference between their character scales. Then, we define:

$$R_{\text{TLC}} = w_{\text{text}}\, R_{\text{text}} + w_{\text{layout}}\, \text{Gate}(R_{\text{text}})\, R_{\text{layout}}, \tag{5}$$

where $\text{Gate}(\cdot)$ is a thresholding function that activates the layout term only when text recognition is sufficiently accurate.

Style and fusion rewards. Text correctness alone is insufficient; the rendered text should also match the intended typography and blend with the surroundings seamlessly. We use R_{style} to measure whether the generated text style matches the target, and R_{fusion} to measure whether the text is naturally integrated with local background and visual context. Both rewards are scored by an MLLM judge (Qwen3.5-122B-A10B [27] in our implementation). The final reward is

$$R = \lambda_{\text{TLC}} R_{\text{TLC}} + \lambda_{\text{style}} R_{\text{style}} + \lambda_{\text{fusion}} R_{\text{fusion}}. \tag{6}$$

3.6 Training Data

Our training data is composed of several curated datasets spanning both image editing and text-to-image. It covers a broad range of scenarios, including web-curated text-rich images and template-constructed images. In total, the final training mixture contains approximately 500K samples. We control the sampling ratios across these sources during training. Overall, web-curated data and template-constructed data are sampled at an approximately balanced ratio, while the overall mixture of image editing and text-to-image data is maintained at roughly 3:1.

3.7 InfoTextBench

Although existing benchmarks such as LongTextBench [13] evaluate long-text rendering, they do not adequately cover text-rich scenarios with densely arranged and highly structured content. We therefore build **InfoTextBench**. InfoTextBench contains 133 samples, with Chinese and English prompt versions for each. Each sample includes a web-collected reference image, manually curated paired image-editing and text-to-image prompts, and an explicit target text list. Both prompts describe the same poster content, with the editing prompt relying on the reference image and the text-to-image prompt describing it explicitly. The prompts are long and detailed, averaging 1065 and 1127 prompt words for image editing and text-to-image, respectively. Each sample contains 19.3 target text items on average, with an average total text length of 151 words, often mixing Chinese, English, and symbols. The content covers diverse text-rich visual layouts across multiple real-world scenarios. Compared with LongTextBench, InfoTextBench contains substantially longer target texts, as quantified in Appendix A.4.

Evaluation protocol. We evaluate generated results along two dimensions: text accuracy and overall visual quality. For text accuracy, we use PaddleOCR [8] and compare the recognized text against the target text list. We report normalized edit distance (NED, defined as 1 minus the normalized Levenshtein edit distance so that higher is better), together with word-level recall, word-level precision, word-level F1, and phrase hit rate. Here, word-level recall measures how much target text is correctly generated, while word-level precision measures how much generated text is correct rather than wrong or extra. We use word-level F1 to summarize them. Phrase hit rate measures the fraction of target phrases found in the generated image, making it more sensitive to multi-block poster content. For overall visual quality, we use an MLLM judge to score aesthetics and prompt following separately, and take their average as the final overall visual quality score.

4 Experiments

We evaluate GlyphAnchor on both text-to-image and image-editing backbones across general text-rendering benchmarks, ChineseWord for rare characters, and our text-rich InfoTextBench.

4.1 Experimental Setup

Implementation details. We implemented GlyphAnchor on three diffusion-transformer backbones: FireRed-Image-Edit-1.1 [38] and Qwen-Image-Edit-2511 [42] for image editing, and Z-Image [36] for text-to-image. For each backbone, we first conduct staged supervised finetuning for two days on

Table 1: **Quantitative comparison on LongTextBench, OneIGBench-text, and CVTG-2K.**

Method	LongTextBench			OneIGBench-text			CVTG-2K		
	EN	ZH	Avg.	EN	ZH	Avg.	NED	CLIP	WAC
Closed-source models									
GPT-Image-1 [26]	0.956	0.619	0.788	0.857	0.650	0.754	0.948	0.798	0.857
Seedream 3.0 [12]	0.896	0.878	0.887	0.865	0.928	0.897	0.854	0.782	0.592
Seedream 4.0 [28]	0.921	0.926	0.924	0.981	0.982	0.982	0.922	0.798	0.845
Nano Banana 2.0 [16]	0.981	0.949	0.965	0.944	0.983	0.964	0.875	0.737	0.779
Open-source baselines									
TextCrafter [35]	–	–	–	–	–	–	0.868	0.787	0.737
TextDiffuser-2 [7]	–	–	–	–	–	–	0.435	0.677	0.233
AnyText [40]	–	–	–	–	–	–	0.468	0.743	0.180
Qwen-Image [42]	0.943	0.946	0.945	0.891	0.963	0.927	0.912	0.802	0.829
Qwen-Image-2512 [42]	0.956	0.965	0.960	0.988	0.984	0.986	0.929	0.782	0.860
FLUX.2-klein-9B [3]	0.864	0.218	0.541	0.866	0.492	0.679	–	–	–
Emu3.5 [9]	0.976	0.928	0.952	**0.994**	0.941	0.968	0.966	–	0.912
GLM-Image [47]	0.952	0.979	0.966	0.969	0.976	0.973	0.956	0.788	0.912
LongCat-Image [37]	0.871	0.953	0.912	0.965	0.945	0.955	0.936	0.786	0.866
ERNIE-Image [1]	0.968	0.959	0.964	0.967	0.898	0.932	–	–	–
Z-Image-Turbo [36]	0.917	0.926	0.922	**0.994**	0.982	0.988	0.928	0.805	0.859
GlyphAnchor final models									
FireRed-Image-Edit-1.1 [38]	0.895	0.898	0.897	0.813	0.937	0.875	0.865	0.785	0.747
+ GlyphAnchor	**0.990**	0.980	**0.985**	0.961	0.969	0.965	0.968	0.782	0.917
	(+.095)	*(+.082)*	*(+.088)*	*(+.148)*	*(+.032)*	*(+.090)*	*(+.103)*	*(-.003)*	*(+.170)*
Qwen-Image-Edit-2511 [42]	0.900	0.924	0.912	0.893	0.963	0.928	0.850	0.774	0.717
+ GlyphAnchor	0.983	0.984	0.984	0.960	0.975	0.967	0.968	0.801	0.922
	(+.083)	*(+.060)*	*(+.072)*	*(+.067)*	*(+.012)*	*(+.039)*	*(+.118)*	*(+.027)*	*(+.205)*
Z-Image [36]	0.935	0.936	0.936	0.987	0.988	0.988	0.937	0.797	0.867
+ GlyphAnchor	0.975	**0.987**	0.981	0.988	**0.994**	**0.991**	**0.970**	**0.823**	**0.924**
	(+.040)	*(+.051)*	*(+.045)*	*(+.001)*	*(+.006)*	*(+.003)*	*(+.033)*	*(+.026)*	*(+.057)*

Table 2: **Quantitative comparison on ChineseWord.**

Method	Level-1	Level-2	Level-3	Total
Qwen-Image [42]	0.937	0.332	0.026	0.533
Qwen-Image-2512 [42]	0.936	0.382	0.039	0.553
LongCat-Image [37]	0.907	0.714	0.379	0.731
FireRed-Image-Edit-1.1 [38]	0.926	0.411	0.067	0.565
+ GlyphAnchor	**0.992**	0.783	0.416	0.801
	(+.066)	*(+.372)*	*(+.349)*	*(+.236)*
Qwen-Image-Edit-2511 [42]	0.816	0.360	0.051	0.496
+ GlyphAnchor	0.987	**0.795**	**0.438**	**0.807**
	(+.171)	*(+.435)*	*(+.387)*	*(+.311)*
Z-Image [36]	0.852	0.278	0.025	0.476
+ GlyphAnchor	0.939	0.377	0.102	0.565
	(+.087)	*(+.099)*	*(+.077)*	*(+.089)*

24 H800-80GB GPUs using AdamW [22] with a learning rate of 5×10^{-6}. We then perform NFT post-training for another two days on the same hardware setup, using AdamW with a learning rate of 3×10^{-4}. For NFT, we sample 8 images per prompt and set the number of groups to 12, with $(\lambda_{\text{TLC}}, \lambda_{\text{style}}, \lambda_{\text{fusion}}) = (0.5, 0.25, 0.25)$.

Evaluation setup. The evaluation covers LongTextBench [13], OneIGBench-text [5], CVTG-2K [35], ChineseWord [42], and our InfoTextBench, which together test long, complex, and densely arranged text, as well as rare characters. Since GlyphAnchor requires a layout input, we use Qwen3.5-122B-A10B [27] as the MLLM-based layout planner to generate bounding boxes for the target text items. For ChineseWord, we use the *Table of General Standard Chinese Characters* [33] as the source vocabulary, which includes 3,500 Level-1 commonly used characters, 3,000 Level-2 characters, and 1,605 Level-3 characters. We evaluate character rendering by generating one image for each target character. For InfoTextBench, we generate four images for each prompt and report the average score over all generated images. All metrics are higher-is-better unless stated otherwise.

4.2 Quantitative Results

General text-rendering benchmarks. Table 1 shows that GlyphAnchor consistently improves long text rendering across all three backbones and three evaluated benchmarks. The gains are particularly

Table 3: **Quantitative comparison on InfoTextBench.**

Method	NED	Word Recall	Word Precision	Word F1	Phrase Hit	Overall Visual Quality
Editing models						
Seedream 4.5 [4]	0.432	0.865	0.699	0.765	0.674	6.360
Nano Banana Pro [15]	0.649	**0.978**	0.916	0.942	**0.928**	**9.254**
FLUX.2-dev [2]	0.226	0.505	0.529	0.505	0.116	4.311
GLM-Image [47]	0.505	0.719	**0.938**	0.803	0.583	5.787
LongCat-Image Edit [37]	0.253	0.445	0.704	0.505	0.278	4.477
FireRed-Image-Edit-1.1 [38]	0.210	0.443	0.463	0.433	0.154	4.608
+ GlyphAnchor	0.651	0.950	0.864	0.898	0.713	6.480
	(+.441)	*(+.507)*	*(+.401)*	*(+.465)*	*(+.559)*	*(+1.872)*
Qwen-Image-Edit-2511 [42]	0.237	0.466	0.543	0.479	0.167	4.603
+ GlyphAnchor	0.632	0.957	0.817	0.871	0.737	6.704
	(+.395)	*(+.491)*	*(+.274)*	*(+.392)*	*(+.570)*	*(+2.101)*
Text-to-image models						
Qwen-Image [42]	0.234	0.421	0.760	0.490	0.248	4.910
Qwen-Image-2512 [42]	0.500	0.802	0.794	0.788	0.546	6.344
LongCat-Image [37]	0.242	0.432	0.740	0.494	0.295	4.474
Z-Image [36]	0.566	0.949	0.821	0.872	0.807	7.092
+ GlyphAnchor	**0.710**	**0.978**	0.929	**0.950**	0.885	8.822
	(+.144)	*(+.029)*	*(+.108)*	*(+.078)*	*(+.078)*	*(+1.730)*

pronounced for the two editing models, while the already strong Z-Image baseline still benefits on both LongTextBench and CVTG-2K. Notably, Z-Image-based GlyphAnchor achieves the best CVTG-2K results among all compared methods, indicating that explicit glyph anchoring improves text fidelity while maintaining semantic alignment.

Rare character rendering. Table 2 further shows that GlyphAnchor is especially effective on rare character rendering, where prompts alone are insufficient. Across all backbones, GlyphAnchor improves rendering capability for both common (Level 1) and rare (Level 2 and 3) characters. This pattern supports our central hypothesis: explicit glyph evidence becomes most valuable when the prompt alone cannot reliably specify the glyph structure of the target characters.

Text-rich generation and editing. Table 3 evaluates models on InfoTextBench, a more demanding benchmark where models must render many text items while preserving layout and visual coherence. GlyphAnchor improves both word-level and phrase-level text accuracy, as well as overall visual quality, for both editing and text-to-image models. These results suggest that glyph anchoring helps not only local text correctness but also the practical usability of text-rich images.

4.3 Qualitative Results

Fig. 4 presents qualitative comparisons on text-rich cases, where models must render long titles, dense small text, and structured layouts. While closed-source Nano Banana Pro and GlyphAnchor both render these examples well, other baselines still suffer from common text-rendering errors, such as corrupted characters, missing or duplicated words, illegible small text, and incorrect layout. For example, in the third row, Seedream 4.5 fails to follow the prompt-specified layout, GLM-Image also produces an incorrect layout, Qwen-Image-2512 contains missing, incorrect, and corrupted words, and Z-Image omits some words. These comparisons show that GlyphAnchor can produce complete and readable text-rich images.

4.4 Ablation Study

Table 4 presents an ablation study on the FireRed-Image-Edit-1.1 backbone. All SFT variants are trained for the same number of steps on the same training data. Default SFT on the same training data provides only limited gains over the original backbone. Training with glyph patches but without positional anchoring yields substantially larger gains, confirming the value of explicit glyph conditions. Adding positional anchoring further improves performance across all metrics, with a particularly large gain in phrase hit rate on InfoTextBench, showing that spatial grounding is crucial for making glyph patches an effective auxiliary enhancement for text-rich generation and editing. Finally, text-aware NFT post-training yields an additional consistent boost, producing the best results



Figure 4: **Qualitative comparison on challenging text-rich cases.** Best viewed with zoom.

Table 4: **Ablation study on the FireRed-Image-Edit-1.1 backbone.**

Variant	LongTextBench			InfoTextBench					
	EN	ZH	Avg.	NED	Word Recall	Word Precision	Word F1	Phrase Hit	Overall Visual Quality
FireRed-Image-Edit-1.1	0.895	0.898	0.897	0.210	0.443	0.463	0.433	0.154	4.608
+ default SFT with train data	0.907	0.905	0.906	0.242	0.501	0.537	0.510	0.284	4.931
+ glyph patch SFT w/o anchor	0.925	0.935	0.930	0.416	0.794	0.701	0.737	0.399	5.012
+ glyph patch SFT w/ anchor	0.984	0.974	0.979	0.647	0.949	0.856	0.892	0.711	6.395
+ text-aware NFT	**0.990**	**0.980**	**0.985**	**0.651**	**0.950**	**0.864**	**0.898**	**0.713**	**6.480**

on both LongTextBench and InfoTextBench. Together, these results show that glyph priors improve text rendering and that positional anchoring is essential for making them reliable.

5 Conclusion

We presented **GlyphAnchor**, a method for enhancing visual text rendering in both text-to-image and image editing using position-anchored glyph priors. GlyphAnchor organizes glyph priors as compact glyph patch images and anchors them to target layouts through the model's native positional encoding. This provides explicit character-shape information together with spatial cues, while keeping the conditioning mechanism lightweight and easy to adapt across diffusion-transformer backbones. Com-

bined with staged supervised finetuning and text-aware NFT post-training, GlyphAnchor consistently improves the corresponding backbones on long, complex, and densely arranged text and rare character scenarios. We also introduced **InfoTextBench** to evaluate text-rich visual text rendering. Overall, our results show that GlyphAnchor improves text fidelity while preserving image quality.

Broader Impacts. While our work improves visual text generation, it may also introduce misuse risks, such as forged documents, misleading advertisements, and disinformation. Like other image generation models, GlyphAnchor may render harmful text, including obscene, hateful, or illegal content, at specified locations. Potential safeguards include content moderation, watermarking, detection tools, and audit logging, while collaboration with fact-checking organizations remains a possible future direction. We encourage responsible use and further study of safeguards for such systems.

Acknowledgments and Disclosure of Funding

This research was supported by the National Natural Science Foundation of China (NSFC 62576103). The computations were conducted using the CFFF platform at Fudan University. Part of this work was carried out during an internship at Xiaohongshu.

References

[1] Baidu. ERNIE-Image. https://huggingface.co/baidu/ERNIE-Image, 2026.

[2] Black Forest Labs. FLUX.2-dev. https://huggingface.co/black-forest-labs/FLUX.2-dev, 2025.

[3] Black Forest Labs. FLUX.2-klein-9B. https://huggingface.co/black-forest-labs/FLUX.2-klein-9B, 2026.

[4] ByteDance. Seedream 4.5. https://docs.byteplus.com/en/docs/ModelArk/1829186, 2025.

[5] Jingjing Chang, Yixiao Fang, Peng Xing, Shuhan Wu, Wei Cheng, Rui Wang, Xianfang Zeng, Gang Yu, and Hai-Bao Chen. OneIG-Bench: Omni-dimensional nuanced evaluation for image generation, 2025. URL https://arxiv.org/abs/2506.07977.

[6] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. TextDiffuser: Diffusion models as text painters. In *Advances in Neural Information Processing Systems*, 2023.

[7] Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. TextDiffuser-2: Unleashing the power of language models for text rendering. In *Proceedings of the European Conference on Computer Vision*, Lecture Notes in Computer Science, pages 386–402. Springer, 2024.

[8] Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. PaddleOCR 3.0 Technical Report, 2025. URL https://arxiv.org/abs/2507.05595.

[9] Yufeng Cui, Honghao Chen, Haoge Deng, Xu Huang, Xinghang Li, Jirong Liu, Yang Liu, Zhuoyan Luo, Jinsheng Wang, Wenxuan Wang, Yueze Wang, Chengyuan Wang, Fan Zhang, Yingli Zhao, Ting Pan, Xianduo Li, Zecheng Hao, Wenxuan Ma, Zhuo Chen, Yulong Ao, Tiejun Huang, Zhongyuan Wang, and Xinlong Wang. Emu3.5: Native multimodal models are world learners, 2025. URL https://arxiv.org/abs/2510.26583.

[10] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with io-awareness. In *Advances in Neural Information Processing Systems*, 2022.

[11] Zhengyao Fang, Pengyuan Lyu, Jingjing Wu, Chengquan Zhang, Jun Yu, Guangming Lu, and Wenjie Pei. Recognition-Synergistic Scene Text Editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13104–13113. Computer Vision Foundation / IEEE, 2025.

[12] Yu Gao, Lixue Gong, Qiushan Guo, Xiaoxia Hou, Zhichao Lai, Fanshi Li, Liang Li, Xiaochen Lian, Chao Liao, Liyang Liu, Wei Liu, Yichun Shi, Shiqi Sun, Yu Tian, Zhi Tian, Peng Wang, Rui Wang, Xuanda Wang, Xun Wang, Ye Wang, Guofeng Wu, Jie Wu, Xin Xia, Xuefeng Xiao, Zhonghua Zhai, Xinyu Zhang, Qi Zhang, Yuwei Zhang, Shijia Zhao, Jianchao Yang, and Weilin Huang. Seedream 3.0 technical report, 2025. URL https://arxiv.org/abs/2504.11346.

[13] Zigang Geng, Yibing Wang, Yeyao Ma, Chen Li, Yongming Rao, Shuyang Gu, Zhao Zhong, Qinglin Lu, Han Hu, Xiaosong Zhang, Linus, Di Wang, and Jie Jiang. X-Omni: Reinforcement learning makes discrete autoregressive image generative models great again, 2025. URL https://arxiv.org/abs/2507.22058.

[14] Lixue Gong, Xiaoxia Hou, Fanshi Li, Liang Li, Xiaochen Lian, Fei Liu, Liyang Liu, Wei Liu, Wei Lu, Yichun Shi, Shiqi Sun, Yu Tian, Zhi Tian, Peng Wang, Xun Wang, Ye Wang, Guofeng Wu, Jie Wu, Xin Xia, Xuefeng Xiao, Linjie Yang, Zhonghua Zhai, Xinyu Zhang, Qi Zhang, Yuwei Zhang, Shijia Zhao, Jianchao Yang, and Weilin Huang. Seedream 2.0: A native chinese-english bilingual image generation foundation model, 2025. URL https://arxiv.org/abs/2503.07703.

[15] Google. Nano banana pro. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Image-Model-Card.pdf, 2025.

[16] Google. Nano Banana 2. https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/, 2026.

[17] Bowen Jiang, Yuan Yuan, Xinyi Bai, Zhuoqun Hao, Alyson Yin, Yaojie Hu, Wenyu Liao, Lyle H. Ungar, and Camillo Jose Taylor. ControlText: Unlocking controllable fonts in multilingual text rendering without font annotations. In *Proceedings of the Empirical Methods in Natural Language Processing (Findings)*, pages 25414–25425. Association for Computational Linguistics, 2025.

[18] Rui Lan, Yancheng Bai, Xu Duan, Mingxing Li, Dongyang Jin, Ryan Xu, Dong Nie, Lei Sun, and Xiangxiang Chu. FLUX-Text: A simple and advanced diffusion transformer baseline for scene text editing, 2025. URL https://arxiv.org/abs/2505.03329.

[19] Rosanne Liu, Dan Garrette, Chitwan Saharia, William Chan, Adam Roberts, Sharan Narang, Irina Blok, RJ Mical, Mohammad Norouzi, and Noah Constant. Character-aware models improve visual text rendering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16270–16297. Association for Computational Linguistics, 2023.

[20] Zeyu Liu, Weicong Liang, Zhanhao Liang, Chong Luo, Ji Li, Gao Huang, and Yuhui Yuan. Glyph-ByT5: A customized text encoder for accurate visual text rendering. In *Proceedings of the European Conference on Computer Vision*, Lecture Notes in Computer Science, pages 361–377. Springer, 2024.

[21] Zeyu Liu, Weicong Liang, Yiming Zhao, Bohan Chen, Lin Liang, Lijuan Wang, Ji Li, and Yuhui Yuan. Glyph-ByT5-v2: A strong aesthetic baseline for accurate multilingual visual text rendering, 2024. URL https://arxiv.org/abs/2406.10208.

[22] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2019.

[23] Runnan Lu, Yuxuan Zhang, Jiaming Liu, Haofan Wang, and Yiren Song. EasyText: Controllable diffusion transformer for multilingual text rendering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7565–7573. AAAI Press, 2026.

[24] Jian Ma, Mingjun Zhao, Chen Chen, Ruichen Wang, Di Niu, Haonan Lu, and Xiaodong Lin. GlyphDraw: Seamlessly rendering text with intricate spatial structures in text-to-image generation, 2023. URL https://arxiv.org/abs/2303.17870.

[25] Jian Ma, Yonglin Deng, Chen Chen, Nanyang Du, Haonan Lu, and Zhenyu Yang. GlyphDraw2: Automatic generation of complex glyph posters with diffusion models and large language models, 2025. URL https://arxiv.org/abs/2407.02252.

[26] OpenAI. GPT-Image-1. https://openai.com/zh-Hans-CN/index/introducing-4o-image-generation/, 2025.

[27] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[28] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, Xiaowen Jian, Huafeng Kuang, Zhichao Lai, Fanshi Li, Liang Li, Xiaochen Lian, Chao Liao, Liyang Liu, Wei Liu, Yanzuo Lu, Zhengxiong Luo, Tongtong Ou, Guang Shi, Yichun Shi, Shiqi Sun, Yu Tian, Zhi Tian, Peng Wang, Rui

Wang, Xun Wang, Ye Wang, Guofeng Wu, Jie Wu, Wenxu Wu, Yonghui Wu, Xin Xia, Xuefeng Xiao, Shuang Xu, Xin Yan, Ceyuan Yang, Jianchao Yang, Zhonghua Zhai, Chenlin Zhang, Heng Zhang, Qi Zhang, Xinyu Zhang, Yuwei Zhang, Shijia Zhao, Wenliang Zhao, and Wenjia Zhu. Seedream 4.0: Toward next-generation multimodal image generation, 2025. URL https://arxiv.org/abs/2509.20427.

[29] Wenda Shi, Yiren Song, Dengming Zhang, Jiaming Liu, and Xingxing Zou. FonTS: Text rendering with typography and style controls. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18463–18474, October 2025.

[30] Wataru Shimoda, Naoto Inoue, Daichi Haraguchi, Hayato Mitani, Seiichi Uchida, and Kota Yamaguchi. Type-R: Automatically retouching typos for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2745–2754. Computer Vision Foundation / IEEE, 2025.

[31] Xincheng Shuai, Ziye Li, Henghui Ding, and Dacheng Tao. GlyphPrinter: Region-grouped direct preference optimization for glyph-accurate visual text rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[32] Jaewoo Song, Jooyoung Choi, Kanghyun Baek, Sangyub Lee, Daemin Park, and Sungroh Yoon. DCText: Scheduled attention masking for visual text generation via divide-and-conquer strategy, 2025. URL https://arxiv.org/abs/2512.01302.

[33] State Council of the People's Republic of China. Circular of the state council on releasing the general standard chinese characters table. https://www.gov.cn/zwgk/2013-08/19/content_2469793.htm, 2013. Guo Fa [2013] No. 23. Accessed: 2026-05-05.

[34] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063, 2024.

[35] Ying Tai, Nikai Du, Rui Xie, Zhennan Chen, Qian Wang, Zhengkai Jiang, Kai Zhang, and Jian Yang. Investigating text insulation and attention mechanisms for complex visual text generation, 2026. URL https://arxiv.org/abs/2503.23461.

[36] Image Team, Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Xin Jin, Liangchen Li, Zhen Li, Zhong-Yu Li, David Liu, Dongyang Liu, Junhan Shi, Qilong Wu, Feng Yu, Chi Zhang, Shifeng Zhang, and Shilin Zhou. Z-Image: An efficient image generation foundation model with single-stream diffusion transformer, 2025. URL https://arxiv.org/abs/2511.22699.

[37] Meituan LongCat Team, Hanghang Ma, Haoxian Tan, Jiale Huang, Junqiang Wu, Jun-Yan He, Lishuai Gao, Songlin Xiao, Xiaoming Wei, Xiaoqi Ma, Xunliang Cai, Yayong Guan, and Jie Hu. LongCat-Image Technical Report, 2025. URL https://arxiv.org/abs/2512.07584.

[38] Super Intelligence Team, Changhao Qiao, Chao Hui, Chen Li, Cunzheng Wang, Dejia Song, Jiale Zhang, Jing Li, Qiang Xiang, Runqi Wang, Shuang Sun, Wei Zhu, Xu Tang, Yao Hu, Yibo Chen, Yuhao Huang, Yuxuan Duan, Zhiyi Chen, and Ziyuan Guo. FireRed-Image-Edit-1.0 Technical Report, 2026. URL https://arxiv.org/abs/2602.13344.

[39] Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. AnyText2: Visual text generation and editing with customizable attributes, 2024. URL https://arxiv.org/abs/2411.15245.

[40] Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. AnyText: Multilingual visual text generation and editing. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2024.

[41] Tong Wang, Ting Liu, Xiaochao Qu, Chengjing Wu, Luoqi Liu, and Xiaolin Hu. GlyphMastero: A glyph encoder for high-fidelity scene text editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28523–28532. Computer Vision Foundation / IEEE, 2025.

[42] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image Technical Report, 2025. URL https://arxiv.org/abs/2508.02324.

[43] Yu Xie, Jielei Zhang, Pengyu Chen, Weihang Wang, Longwen Gao, Peiyi Li, Qian Qiao, and Zhouhui Lian. TextFlux: An ocr-free dit model for high-fidelity multilingual scene text synthesis, 2026. URL https://arxiv.org/abs/2505.17778.

[44] Zexuan Yan, Jiarui Jin, Yue Ma, Shijian Wang, Jiahui Hu, Wenxiang Jiao, Yuan Lu, and Linfeng Zhang. GlyphBanana: Advancing precise text rendering through agentic workflows, 2026. URL https://arxiv.org/abs/2603.12155.

[45] Zhenyu Yan, Jian Wang, Aoqiang Wang, Yuhan Li, Wenxiang Shang, and Zhu Hangcheng. TextMaster: A unified framework for realistic text editing via glyph-style dual-control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16112–16121, October 2025.

[46] Yukang Yang, Dongnan Gui, Yuhui Yuan, Weicong Liang, Haisong Ding, Han Hu, and Kai Chen. GlyphControl: Glyph conditional control for visual text generation. In *Advances in Neural Information Processing Systems*, 2023.

[47] Z.AI. GLM-Image. https://z.ai/blog/glm-image, 2026.

[48] Weichao Zeng, Yan Shu, Zhenhang Li, Dongbao Yang, and Yu Zhou. TextCtrl: Diffusion-based scene text editing with prior guidance control. In *Advances in Neural Information Processing Systems*, 2024.

[49] Ruiqiang Zhang, Hengyi Wang, Chang Liu, Guanjie Wang, Zehua Ma, and Weiming Zhang. FreeText: Training-free text rendering in diffusion transformers via attention localization and spectral glyph injection, 2026. URL https://arxiv.org/abs/2601.00535.

[50] Shitian Zhao, Qilong Wu, Xinyue Li, Bo Zhang, Ming Li, Qi Qin, Dongyang Liu, Kaipeng Zhang, Hongsheng Li, Yu Qiao, Peng Gao, Bin Fu, and Zhen Li. LeX-Art: Rethinking text generation via scalable high-quality data synthesis, 2025. URL https://arxiv.org/abs/2503.21749.

[51] Yiming Zhao and Zhouhui Lian. UDiffText: A unified framework for high-quality text synthesis in arbitrary images via character-aware diffusion models. In *Proceedings of the European Conference on Computer Vision*, Lecture Notes in Computer Science, pages 217–233. Springer, 2024.

[52] Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. In *Proceedings of the International Conference on Learning Representations*, 2026.

[53] Hanshen Zhu, Yuliang Liu, Xuecheng Wu, An-Lan Wang, Hao Feng, Dingkang Yang, Chao Feng, Can Huang, Jingqun Tang, and Xiang Bai. TextPecker: Rewarding structural anomaly quantification for enhancing visual text rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

A Technical appendices and supplementary material

A.1 Qualitative ablation on positional anchoring

Fig. 5 provides a qualitative comparison of glyph patch SFT with and without positional anchoring on the FireRed-Image-Edit-1.1 backbone. Both GlyphAnchor variants use the same staged SFT strategy, training data, rendered glyph patch images, layout inputs, and inference pipeline. They differ only in whether glyph tokens are assigned position IDs according to their layout anchors. Consistent with the quantitative ablation in Table 4, text accuracy improves progressively from the backbone to glyph patch SFT without anchoring and further to position-anchored glyph patch SFT. Without anchoring, glyph patches are only weakly associated with the target regions, often leading to incomplete text or corrupted characters; positional anchoring instead aligns glyph priors with their target layout, enabling more accurate rendering.



Figure 5: **Qualitative ablation of positional anchoring.** Best viewed with zoom.

A.2 Qualitative ablation on staged SFT

Fig. 6 qualitatively compares three SFT variants of GlyphAnchor. All variants use FireRed-Image-Edit-1.1 as the backbone, with the same position-anchored glyph prior strategy, training data, layout inputs, and inference pipeline. They differ only in the glyph patch images used during SFT: GT crop SFT uses cropped text regions from the ground-truth target images, rendered glyph SFT uses rendered black-on-white glyph patches, and staged SFT follows our proposed staged supervision strategy. At inference time, all variants are conditioned on rendered glyph patches. GT crop SFT encourages the model to transfer the appearance of the conditioning patches. Since inference uses rendered glyph patches rather than real text crops, this supervision introduces a train–test mismatch, which manifests as white-background copy-paste artifacts. Rendered glyph SFT alleviates this mismatch and improves visual consistency, but remains less faithful to the prompt-specified style. In contrast, staged SFT better preserves the target image style while maintaining comparable text accuracy, yielding the most consistent overall results.



Figure 6: **Qualitative ablation of staged SFT.** Best viewed with zoom.

A.3 Robustness to Layout Perturbations

We evaluate the robustness of GlyphAnchor to inaccurate layout estimates on InfoTextBench using FireRed-Image-Edit-1.1 as the backbone. We randomly perturb the bounding boxes by shifting their positions by up to $\pm 10\%$ of their respective widths and heights and resizing them with factors in $[0.9, 1.1]$. All other generation and evaluation settings remain unchanged. As shown in Tab. 5, GlyphAnchor maintains nearly identical performance under these perturbations, indicating robustness to moderate layout estimation errors. This observation is consistent with the train-time glyph augmentations described in Sec. 3.4.

Table 5: Robustness to layout perturbations on InfoTextBench using FireRed-Image-Edit-1.1 as the backbone.

Setting	NED	Word Recall	Word Precision	Word F1	Phrase Hit	Overall Visual Quality
GlyphAnchor	0.651	0.950	0.864	0.898	0.713	6.480
+ Layout Perturbation	0.645	0.955	0.861	0.898	0.710	6.456

A.4 Comparison of InfoTextBench and LongTextBench

Table 6 compares the text-length statistics of InfoTextBench and LongTextBench. InfoTextBench contains substantially longer target texts, with a mean length of 150.99 compared to 39.94 in LongTextBench. We measure text length in words for English and characters for Chinese.

Table 6: Statistical comparison of LongTextBench and InfoTextBench.

Benchmark	# Samples	Mean Length	Median Length	Length Range
LongTextBench	320	39.94	33	[10, 121]
InfoTextBench	266	150.99	135	[18, 392]

A.5 Prompt Templates

A.5.1 Prompt templates for NFT rewards.

During NFT post-training in Sec. 3.5, we use two MLLM prompt templates for reward computation: one for style consistency and one for visual fusion. The style consistency prompt evaluates whether the generated text style matches the target typography and visual style, while the visual fusion prompt evaluates whether the rendered text is naturally integrated with the local background and surrounding visual context. The Text-Layout Consistency Reward is computed separately from OCR recognition and layout matching, and therefore does not require an MLLM prompt. The two prompt templates are provided below.

Style consistency reward prompt template.

> You are an expert in evaluating visual text style consistency. You are skilled at comparing the style consistency of corresponding text regions across two images. You should focus on typography, font weight, stroke shape, color, outlines, shadows, texture and material, decorative details, layout style, and the overall visual language. Your task is not to evaluate whether the text content is correct, nor to evaluate how naturally the text is fused into the image, but only to assess whether the styles of corresponding text regions are consistent across the two images.
> Image A: target reference image (ground-truth image)
> Image B: model-generated image
> Please evaluate whether the styles of the corresponding text regions are consistent across the two images according to the following prompt. Note that different text regions within the same image may naturally have different styles, which is acceptable. You should focus on the style similarity between corresponding text regions across the two images, rather than requiring all text regions within one image to share the same style.
> Prompt: [PROMPT]
> If the prompt is a text removal task, such as deleting, removing, erasing, or wiping out text, this style consistency evaluation is not applicable. In that case, directly reply with 3 and do not provide any explanation.
> Please consider the following aspects: typography, font weight, stroke thickness and stroke structure, handwritten vs. printed appearance, color, outlines, shadows, texture and material, decorative details, layout style, and overall visual tone.

Please assign an integer score from 0 to 5 according to the following criteria:
0: The styles of the corresponding text regions are completely inconsistent. The font type, color, or decorative language are clearly incorrect, and the overall result looks like a completely different design system.
1: Only a very small amount of superficial similarity is present, while the overall style is still clearly incorrect. Most key aspects (glyph form, color, outlines, brushwork, decoration) differ substantially.
2: The generated image shows some attempt to imitate the target style, but only some aspects are similar. Noticeable style drift remains, and the consistency between corresponding text regions is weak.
3: The overall style is basically similar, and the main stylistic direction is correct, but there are still differences in several important details, such as stroke characteristics, decorative treatment, color relationships, or typographic tone.
4: The styles are highly similar. Most corresponding text regions are consistent with the target image, with only minor differences that do not affect the overall style judgment.
5: The styles are almost perfectly consistent. The corresponding text regions closely match the target image in typography, stroke language, color treatment, decorative details, and overall visual tone.
Please note: evaluate only text style consistency. Do not evaluate text correctness, and do not evaluate how naturally the text blends with the background.
Please directly reply with a single integer score from 0 to 5.

Visual fusion reward prompt template.

You are an expert in evaluating visual text fusion in images. You are skilled at comparing the degree of harmony and integration between text and the overall image in a target reference image and a generated image. You can distinguish between real-world images and graphic design images such as posters, covers, or layout-based designs, and evaluate text fusion according to the appropriate visual standard for each case. Your task is not to evaluate whether the text content is correct, nor to evaluate whether the text style is consistent, but to assess whether the text in the generated image achieves the same level of visual fusion as in the target reference image.
Image A: target reference image (ground-truth image)
Image B: model-generated image
Please compare how well the text is integrated with the overall image in the two images, and evaluate whether the generated image reaches the level of fusion shown in the target reference image. Here, "fusion" refers to the degree of coordination between the text and the background, texture, lighting, perspective, edges, clarity, occlusion, surface/material attachment, and overall visual style.
If the prompt is a text removal task, such as deleting, removing, erasing, or wiping out text, this visual fusion evaluation is not applicable. In that case, directly reply with 3 and do not provide any explanation.
Prompt: [PROMPT]
Please pay special attention to the following two evaluation standards:
1. If the image is a real-world scene image (e.g., a photographed scene, a real object surface, a street view, a product image, or an environmental image), focus on whether the text looks as if it naturally exists in the scene, and whether it is consistent with realistic lighting, perspective, occlusion, material attachment, and edge details.
2. If the image is a graphic design image, poster, cover, sticker-like image, or layout-based design, focus on whether the text is visually coordinated with the design system, and whether it achieves the same level of decoration, layout quality, visual balance, and design completeness as the target image. Do not penalize it harshly based on a "must look physically real" standard.
Please assign an integer score from 0 to 5 according to the following criteria:
0: The text in the generated image is clearly disconnected from the overall image, and its fusion is far worse than that in the target reference image. Under either the real-world standard or the design-image standard, it looks very unnatural and awkward.
1: A few local regions may be somewhat acceptable, but the overall result is still clearly uncoordinated. There are serious problems with edge quality, spatial attachment, lighting/perspective, or design consistency.
2: The text shows partial integration with the image, but noticeable awkwardness remains. There is still a large gap compared with the target reference image.
3: The overall fusion is basically acceptable, and the major text regions look reasonably integrated. Compared with the target reference image, it reaches a moderate level, although noticeable stiffness or insufficient design coordination remains upon closer inspection.
4: The fusion quality is high. Most text regions are well coordinated with the image, and the overall effect is close to that of the target reference image, with only minor shortcomings.
5: The fusion quality reaches or almost reaches the level of the target reference image. In real-world images, the text looks very natural and believable; in design images, it looks highly coordinated and polished.
Please note: evaluate only visual fusion relative to the target reference image. Do not evaluate text correctness, and do not separately evaluate whether the text style is consistent.
Please directly reply with a single integer score from 0 to 5.

A.5.2 Prompt templates for InfoTextBench evaluation.

For InfoTextBench in Sec. 3.7, we use an MLLM prompt template to evaluate overall visual quality. The prompt asks the MLLM judge to score two aspects of each generated image: aesthetics and

prompt following. The aesthetics score reflects visual and design quality, while the prompt-following score reflects consistency with the input prompt. The final overall visual quality score is obtained by averaging these two scores. The prompt template is provided below.

You are a professional digital artist and graphic designer. Your task is to evaluate an AI-generated image.
The system will provide two images: Image A: the reference image, when applicable. Image B: the generated image based on the reference image.
Please return your evaluation in the following format (keep the reasoning concise):
{
"score": [aesthetic_score, prompt_follow_score],
"reasoning": "..."
}
Scoring criteria (0–10):
The first score (aesthetic, 0–10): Evaluate the overall visual quality of the generated image, including composition and layout, color harmony, visual hierarchy, level of detail, and overall stylistic coherence.
0 = Extremely poor. The image is visually chaotic or severely flawed, with major problems such as broken composition, inconsistent or unpleasant colors, weak visual hierarchy, crude details, obvious artifacts, or a generally unfinished appearance. It does not meet even a basic visual design standard.
10 = Excellent. The image demonstrates outstanding visual quality at a highly polished professional level. The composition is well balanced, the layout is refined, the colors are harmonious and effective, the visual hierarchy is clear, the details are carefully rendered, and the overall style is highly coherent and visually impressive.
The second score (prompt following, 0–10): Evaluate whether the generated image follows the prompt below, including layout structure, text content (whether all required text appears and is correct), style and color scheme, and the use of the reference image content.
0 = Completely fails to follow the prompt. Most key requirements are missing, incorrect, or ignored. The layout, text, style, or major visual elements do not match the prompt description in any meaningful way.
10 = Perfectly follows the prompt. All major and minor requirements are faithfully satisfied, including the intended layout, required text content, stylistic description, color design, and visual elements. The generated image closely matches the prompt in both content and presentation.
Generation prompt: <PROMPT>
Required text list (provided to help judge whether the text is rendered correctly): <TEXTS>

A.6 English Translation of the Prompt in Fig. 2

For readability, all Chinese text, including the target text to be rendered, has been translated into English. The original prompt was entirely in Chinese.

Use the provided Chiikawa character image to generate a healing-themed book recommendation poster. All text should be in Simplified Chinese. The canvas should be a long vertical poster. The overall style should be warm, artistic, and hand-drawn, with an off-white background as the main color, accented by soft tones such as milk-tea brown, mint green, and warm orange to create a fresh, gentle, bookish atmosphere. ... At the very top center, place a small milk-tea-brown capsule-shaped label containing the text “2024 Good Book Recommendations” in a small, rounded dark-brown font, with a small book icon on the left. Below the label is the main title “Chiikawa’s Healing Reading List,” set in a large, bold, rounded handwritten-style font in dark brown and centered horizontally. Below the main title is the subtitle “10 Books That Make the Heart Softer,” set in a medium-sized rounded font in warm brownish-gray and centered horizontally, with a small, slightly open book illustration on each side in mint green. The Chiikawa character should be placed centrally below the title card, sitting on top of a colorful stack of books while holding an open book with both hands. ... Arrange 10 book entries vertically, separated by thin light-brown dashed lines. Each entry should be a light beige card with rounded corners. On the left side of each card, place a narrow vertical decorative color bar, alternating among mint green, pale yellow, pink-orange, and sky blue, with a small dot of the same color at the top. On the left side of each book card, combine the index number and book title into a single line in a medium-sized, bold, dark-brown font. Below it, place a quote in a small, rounded, medium-gray font, with a small light-brown quotation mark decoration on the left. The first book card should display “No.1 The Little Prince,” with the quote “Love means responsibility, not possession.” The second book card should display “No.2 To Live,” with the quote “People live for the sake of living itself.” The third book card should display “No.3 The Miracles of the Namiya General Store,” with the quote “Every trouble has an answer; you just do not know it yet.” The fourth book card should display “No.4 Norwegian Wood,” with the quote “Death is a part of life.” The fifth book card should display “No.5 The Storied Life of A.J. Fikry,” with the quote “No one is an island.” The sixth book card should display “No.6 Ferryman,” with the quote “You are the most beautiful miracle in my wasteland.” The seventh book card should display “No.7 The Moon and Sixpence,” with the quote “Look up at the moon, but also pick up the sixpence.” The eighth book card should display “No.8 The Courage to Be Disliked,” with the quote “All troubles come from interpersonal relationships.” The ninth book card should display “No.9 One Hundred Years of Solitude,” with the quote “There are not many things in life that are truly important.” The tenth book card should display “No.10 Hello, Old Times,” with the quote “May you encounter beautiful moments.” To the right of each

book title, place a small rounded decorative color block corresponding to the color palette of that book's cover. The concluding section at the bottom should occupy approximately one-eighth of the canvas height. ... Center a line of text in this area reading "May every book be a bridge that gently reconciles you with the world," in a medium-sized, bold, rounded font in creamy white, with a small heart icon on each side, also in creamy white.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We introduce our GlyphAnchor method in the abstract and introduction, provide a detailed description in Section 3, and present experimental results in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification: The paper has no limitation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The architecture of our module is introduced in Section 3, and the implementation details are presented at the beginning of Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will release our code upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The implementation details are presented at the beginning of Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Although we report multiple evaluation metrics (e.g., WAC, CLIP, phrase hit) showing substantial improvements over baselines, we did not include error bars, standard deviations, confidence intervals, or statistical significance tests across multiple training runs because they would be too computationally expensive. For InfoTextBench, we report averages over four generations per prompt. As such, our results do not fully meet the criteria outlined in this question for statistical rigor and uncertainty quantification.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We present the computer resources needed for the experiments at the beginning of Section 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We conducted the research in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discussed the broader impacts in the last part of the article.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: GlyphAnchor builds on existing image generation and editing backbones and follows their safety mechanisms and release constraints. We do not expect it to substantially expand the overall risk profile beyond the base models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite the publicly available backbone models used in this work. For InfoTextBench, we will release only the original image URLs, prompts, annotations, and evaluation code, without redistributing the reference images.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The dataset defined for our benchmark will be made publicly available, in case of acceptance, together with the documentation required for reproducing the experiments. Moreover, in case of acceptance, we will also release the source code of our model.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We use multimodal language models for layout planning when layouts are not user-specified, rewards, and visual-quality evaluation, as described in Sections 3.5 and 4.1.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.