PosterMaker: Towards High-Quality Product Poster Generation with Accurate Text Rendering

Supplementary Material

Due to space limitations, we were unable to present all experimental results in the main text. In this supplementary material, we will give more details about our experiments and present additional results.

1. Implementation Details

Training and Inference. We fully follow the settings of SD3 [7]. During training, the denoise loss $\mathcal{L}_{\text{denoise}}$ uses simplified flow matching, also known as 0-rectified flow matching loss [9]. In inference, we also use the inference method of flow matching, with 28 inference steps.

TextRenderNet and SceneGenNet. TextRenderNet and SceneGenNet have an architecture similar to SD3 [7], composed of multiple MM-DiT Blocks. In our implementation, TextRenderNet consists of 12 layers of MM-DiT Blocks, while SceneGenNet consists of 23 layers of MM-DiT Blocks. The output of the N_i-th block of SceneGenNet is first added with the output of the $\lceil \frac{N_i}{2} \rceil$-th block of TextRenderNet, and then add to the N_i-th SD3 block.

Classifier-Free Guidance. We use CFG during inference, with a CFG scale of 5. Additionally, since the "prompt" inputted to TextRenderNet is not a caption but a text representation, the negative one for CFG is set to a zero vector. During training, we randomly drop the text representation to a zero vector with 10% probability.

The Setting of t_1 in Reward Loss. We follow [17] to train the reward loss at the last 10 inference steps, i.e., we set t_1 to 10. Within the range of $t' \sim [1, t_1]$, the result of the image x_0 obtained by one-step inference is close to the result of complete inference.

Details about Metric Calculation. Our evaluation benchmark contains samples generated by LLM [5] thus there is no ground truth for these samples. Therefore, we exclude these LLM-generated samples when calculating metrics that depend on ground truth images, i.e., FID metric for all experiments, text accuracy metrics for GT (with and without VAE reconstruction) and results for ablation on different text features.

About ground truth for training Foreground Extension Detector. We treat the task of detecting foreground extension as a binary classification problem and ask annotators to manually label the ground truth.

2. Baseline Details

We carefully designed 6 baseline approaches based on existing techniques for comparative analysis. The details are shown in Fig. 1. For 1) SD3_inpaint_byt5, 2) SD3_canny&inpaint, and 4) AnyText, we fine-tune them on our 160K dataset for the poster generation task. Meanwhile, 3) SD3_inpaint_Anytext is a two-stage inference method. In the first stage, the pre-trained Inpaint ControlNet gener-

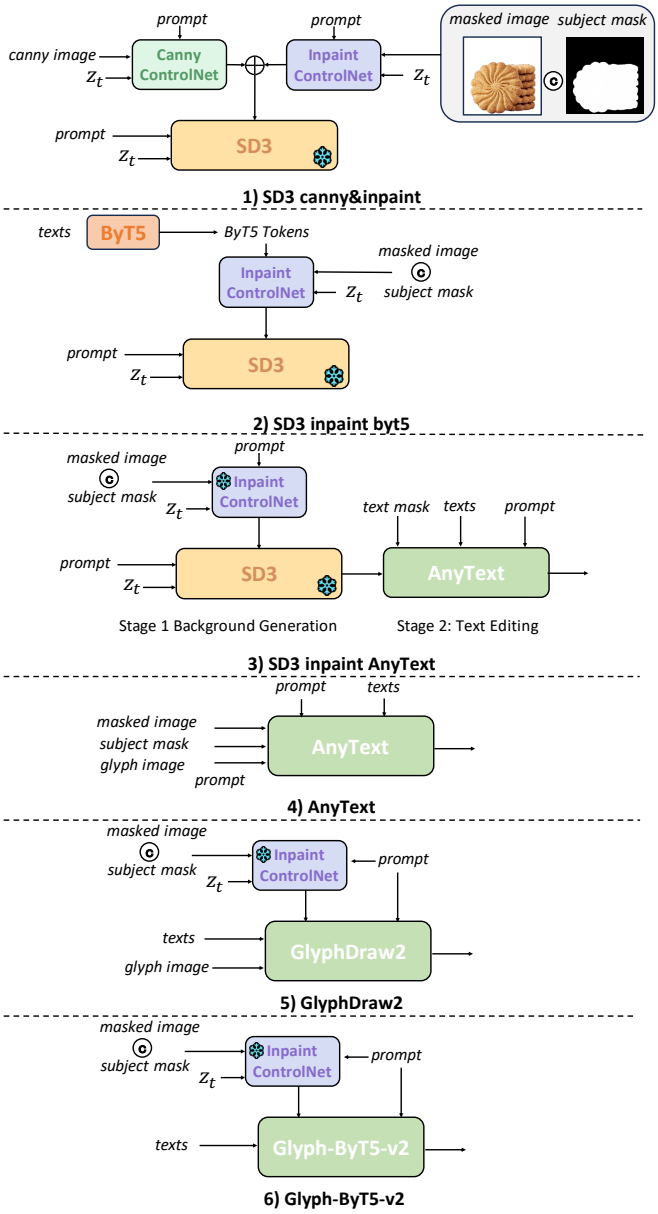

Figure 1. Detailed illustration of the implementation of the different baseline methods.

ates the background, and in the second stage, AnyText performs the text editing task, with AnyText also fine-tuned on the 160K dataset specifically for the text editing task. The Inpainting ControlNet is initialized from pre-trained SD3 Inpainting-Controlnet [2] and Canny ControlNet is initialized from [3]. For 5) GlyphDraw2 [12] and 6) Glyph-ByT5-v2 [11] are both the SOTA T2I methods that support multilingual text rendering. However, they neither have open-source pre-trained weights nor support subject input, so we reproduced them on our dataset by adding the pre-trained inpainting controlnet [4] to support the subject input.

3. Scalable Training for Text Rendering

Our proposed two-stage training strategy allows the model to learn two different capabilities (i.e., text rendering and scene generation) separately, enabling more flexibility with distinct datasets for each phase. Recent text rendering methods [1, 10, 11, 16] typically train their models on datasets containing millions of samples. To verify the potential of further improving our performance with more training data, we build a large dataset with 1 million samples and we directly obtain the text annotations with PPOCRv4 [15] without manually annotating. And we use this dataset for the first stage of text rendering training and use the same 160k data for the second stage of scene generation learning. Compared to using 160k data in both of the previous stages, the text sentence accuracy significantly improved by 4.48% (as shown in Tab. 1), demonstrating that the multi-stage training strategy is flexible and scalable. However, in the main experiments, we select to report the performance of our model training only on 160k data for fair comparison with the baselines.

Data Size (St.1 & St.2)	Sen. ACC	NED
160k & 160k	93.11%	98.21%
1M & 160k	**97.59%**	**99.38%**

Table 1. Quantitative comparison with different data sizes for text rendering training.

4. Discussion on advantages of end-to-end over two-stage methods.

The main weakness of two-stage methods (first inpaint background, then render text) is their inability to consistently provide a clean background for texts (see Fig. 2, reducing text readability, especially with complex backgrounds. In contrast, one-stage methods generate texts and backgrounds simultaneously, enabling them to create a clean backdrop or underlays that enhance text visibility.

5. Text Position Control

The position control of PosterMaker uses a very straightforward approach (as shown in Fig. 3), mapping the text bounding box to cosine position encoding, which is then concatenated with text features and used as the input to TextRenderNet. To demonstrate our method's effectiveness, we evaluate the bounding box IoU (Intersection of Union) metric as follows: 1) we employ OCR model to extract texts from the generated image. 2) For each ground truth text, we identify the best-matched OCR-detected text based on edit distance and then calculate the IoU between their corresponding bounding boxes. We average the IoU score over all the samples to obtain mean IoU (termed mIoU). And we also report IoU@R which indicates the proportion of samples with IoU higher than R. As shown in Tab. 2, our method achieves a high mIoU of 84.65% and 93.94% samples have an IoU score higher than 0.7. These promising results prove that our text position control method is simple yet effective.

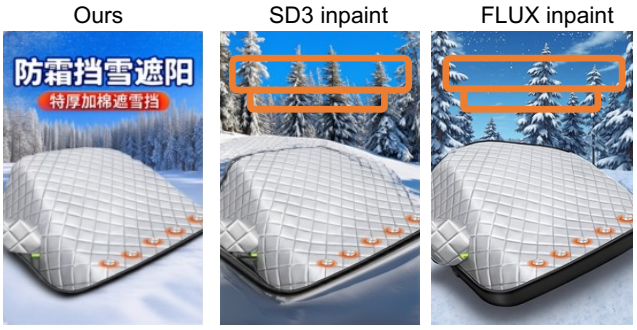

Figure 2. Showcases for end-to-end and two-stage methods.

Method	mIoU	IoU@0.5	IoU@0.7
Ours	84.65%	97.18%	93.94%

Table 2. Evaluation on text location accuracy.

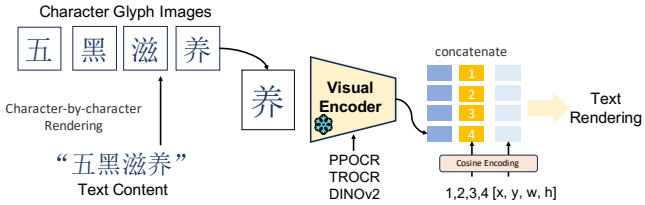

Figure 3. Detailed illustration of how we construct the position embedding for controlling the text position.

6. Comparison Between GlyphByT5 and PosterMaker

GlyphByT5 [10, 11] are recently proposed visual text rendering methods that achieve high text rendering accuracy. And we will discuss some differences and internal connections between our PosterMaker and GlyphByT5 on how to control text rendering.

- Text position control: GlyphByT5 achieve text position

control by modifying the original cross-attention module with their proposed region-wise multi-head cross-attention. In contrast, our PosterMaker encodes the text location directly into the character-level text representation to accomplish text position control. As discussed in Sec. 5, our approach is both simple and effective for precise text location control.

- Text content control: both GlyphByT5 and our PosterMaker control the generation of text content by constructing suitable text representation. Specifically, in this work, we claim that the key to achieve accurate text rendering is to extract **character-level visual** features as the control condition and carefully construct a robust text representation based on off-the-shelf OCR model [15]. In GlyphByT5, the authors also extract **character-level text** features, but with a *textual* encoder named ByT5 [18]. Then they propose glyph-alignment pre-training to align these *textual* features with pre-trained *visual* encoders DINOv2 [14]. Additionally, they employ box-level contrastive learning with complex augmentations and a hard-mining strategy to enhance *character-level* discriminativeness. We hypothesize that the primary reason both our method and GlyphByT5 achieve high text rendering accuracy is our shared goal of constructing a robust **character-level visual** representation. In fact, the ability of GlyphByT5's character-level visual representation is distilled from the pre-trained *visual* encoder DINOv2, rather than inherited from the pre-trained *textual* encoder ByT5 itself. In order to verify our hypothesis and insights, we adopt a more direct approach to directly replace the PPOCR encoder in PosterMaker with DINOv2. As shown in Tab. 3, simply extracting character-wise visual features with DINOv2 can also achieve precise text rendering. This result further verifies our claim: the key to precise text rendering is to extract **character-level visual** features as the control condition.

Text Feature	Type	Sen. ACC	NED
PPOCR Line	visual feat.	38.91%	53.86%
PPOCR Char	visual feat.	95.15%	98.75%
DINOv2 Line	visual feat.	4.25%	20.59%
DINOv2 Char	visual feat.	94.92%	98.66%
GT (w/o Rec.)	-	98.53%	99.59%
GT (w/ SD3 Rec.)	-	98.09%	99.36%

Table 3. Quantitative comparison using various text features.

7. Visualization of Training Samples

We present example training images from our dataset in Fig. 7. The dataset predominantly consists of Chinese text, with a small portion of English text. Additionally, it includes challenging cases with small-sized text elements.

8. The Generalization of Text Representation.

PosterMaker is trained primarily on common Chinese data, with only a minimal amount of English data. Despite this, it demonstrates a notable level of generalization, enabling it to generate English, Japanese, and uncommon Chinese characters that were not included in the training set (as shown in Fig. 6). In order to quantitatively evaluate the generalization capability of PosterMaker, we compared the accuracy of different text representations on uncommon characters using a randomly sampled uncommon character benchmark. The results show that our method can also generalize well to some characters that are unseen in the training set. Our performance is inferior to the canny baseline, likely because the canny baseline has been pre-trained on large-scale image data.

Text Feature	Type	Sen. ACC	NED
ByT5	textual feat.	2.01%	10.27%
Canny	img	**65.12%**	**74.56%**
PPOCR Line	visual feat.	8.34 %	15.84%
PPOCR Char	visual feat.	61.54%	70.38%

Table 4. Quantitative comparison of the rendering results of different text features on uncommon characters.

9. Ablation about Foreground Extension Detector

We collected 20k manually annotated images to train the foreground extension detector. We randomly selected 10% samples as a validation set, while using the remaining 90% for model training. We conduct ablation experiments on different architecture designs of the detector to verify the effectiveness of the proposed architecture. We implement 2 baselines: 1) **RFNet** [6]: we reimplemented RFNet based on the description in their paper [6]. Since we could not access their depth and saliency detection models, we modified our implementation to only use the product image and generated image as input, excluding the depth and saliency maps. 2) **RFNet(SAM)** : in this baseline, we replace the image encoder used in RFNet with the same SAM[8] image encoder used in our method. As summarized in Tab. 5, our proposed foreground extension detector outperforms

Method	Precision	Recall	F1 Score
RFNet (our impl.)	76.52%	75.52%	76.02%
RFNet (SAM)	81.35%	80.99%	81.17%
Ours	**83.52%**	**84.81%**	**84.16%**

Table 5. Evaluation on different architectures of foreground extension detector.

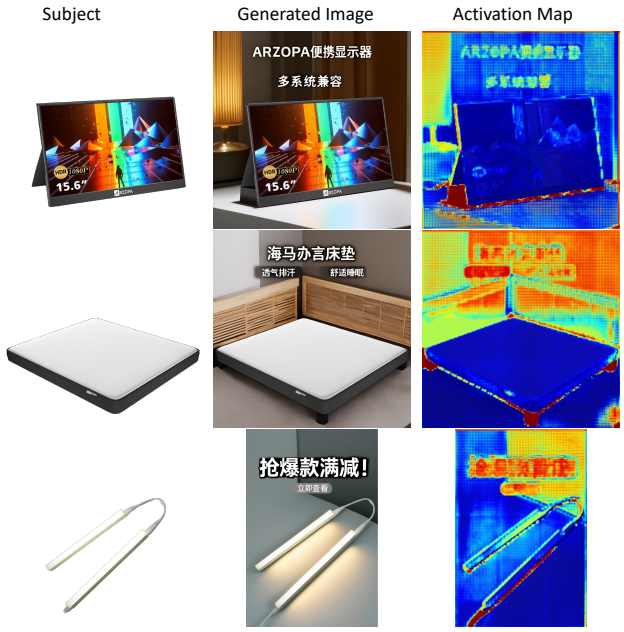

Figure 4. Class activation map of the foreground extension detector.

the baselines by a considerable margin, which demonstrates its effectiveness.

In Fig. 4, we visualize the class activation map [19] of our proposed foreground extension detector. As shown, we can observe a notably higher activation score in the extended foreground regions compared to other areas. This compelling evidence demonstrates that our detector has effectively learned to discern foreground extension cases, thereby it can serve as a robust reward model for fine-tuning PosterMaker to mitigate the foreground extension problem.

10. Ablation about SceneGenNet

SceneGenNet enables our model to perform background inpainting while preserve the subject so we cannot directly remove it. We replace it by SDEdit [13] to achieve inpainting. As the results shown in Sec. 10, replacing it results in a significant drop of performance.

Model	Sen. ACC ↑	NED ↑	FID ↓	CLIP-T ↑
Ours w/o SceneGenNet	90.53%	97.95%	79.44	26.67
Ours	**93.36%**	**98.39%**	**65.35**	**27.04**

Table 6. Comparison between SceneGenNet and SDEdit

11. Discussion on the impact of the test set size.

To ensure a fairer comparison between PosterMaker and the baseline methods, we expanded the test set to 5,000 samples(10x the previous PosterBenchmark). The results are shown in Tab. 7, and the experimental conclusions remain consistent with the previous test set. Due to the calculation principle of the FID metric, increasing the test size leads to a significant decrease in the FID scores for all methods, but still maintains the same conclusion.

Model	Sen. ACC ↑	NED ↑	FID ↓	CLIP-T ↑
Glyph-ByT5-v2	67.87%	86.23%	20.37	21.08
SD3_canny&inpaint	74.49%	88.78%	17.91	20.79
GlyphDraw2	83.81%	96.49%	15.24	20.67
Ours	**90.20%**	**97.58%**	**13.36**	**21.36**

Table 7. Comparison with baseline methods on 5,000 test samples.

12. Discussion on the meaningless texts generated outside target position.

In our early experimental attempts about text rendering in poster generation, we found that the trained model sometimes generates meaningless texts outside the target area of the text, which will seriously affect the aesthetics. We conjecture that the main reason is that the ground truth images sometimes contain text outside the specified position. To solve this problem, we masked out the extra text during training and it solved most cases.

Specifically, SceneGenNet is initialized from pre-trained SD3 Inpainting-Controlnet [2]. In the second stage of training, we simultaneously mask out the regions of the untrained texts (usually those that are too small or just logos) both in the subject mask input to SceneGenNet and in the ground truth image used for loss calculation(as shown in Fig. 5). It is worth noting that although these small texts and logos are not included in the training, we have also annotated them to address the aforementioned issues. Finally, this technique makes the loss corresponding to the masked-out regions very close to zero so that the model will not learn these meaningless texts.

Figure 5. Example of our solution technique for meaningless texts and logos that generated outside target position.

Figure 6. Visualization results on texts in English, Japanese, and uncommon Chinese characters.

Figure 7. Visualization of ground truth for some samples in the dataset.

References

[1] Haoxing Chen, Zhuoer Xu, Zhangxuan Gu, Jun Lan, Xing Zheng, Yaohui Li, Changhua Meng, Huijia Zhu, and Weiqiang Wang. Diffute: Universal text editing diffusion model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 2

[2] Alimama Creative. Sd3-controlnet-inpainting. https://huggingface.co/alimama-creative/SD3-Controlnet-Inpainting, 2024. 2, 4

[3] Alimama Creative. Sd3-controlnet-softedge. https://huggingface.co/alimama-creative/SD3-Controlnet-Softedge, 2024. 2

[4] Alimama Creative. Ecomxl-controlnet-inpaint. https://huggingface.co/alimama-creative/EcomXL_controlnet_inpaint, 2024. 2

[5] Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Bin Wang, Linke Ouyang, Xilin Wei, Songyang Zhang, Haodong Duan, Maosong Cao, Wenwei Zhang, Yining Li, Hang Yan, Yang Gao, Xinyue Zhang, Wei Li, Jingwen Li, Kai Chen, Conghui He, Xingcheng Zhang, Yu Qiao, Dahua Lin, and Jiaqi Wang. Internlm-xcomposer2: Mastering free-form text-image composition and comprehension in vision-language large model. *arXiv preprint arXiv:2401.16420*, 2024. 1

[6] Zhenbang Du, Wei Feng, Haohan Wang, Yaoyu Li, Jingsen Wang, Jian Li, Zheng Zhang, Jingjing Lv, Xin Zhu, Junsheng Jin, et al. Towards reliable advertising image generation using human feedback. In *European Conference on Computer Vision*, pages 399–415. Springer, 2024. 3

[7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 1

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023. 3

[9] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1

[10] 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 *European Conference on Computer Vision*, pages 361–377. Springer, 2024. 2

[11] Zeyu Liu, Weicong Liang, Yiming Zhao, Bohan Chen, Ji Li, and Yuhui Yuan. Glyph-byt5-v2: A strong aesthetic baseline for accurate multilingual visual text rendering. *arXiv preprint arXiv:2406.10208*, 2024. 2

[12] Jian Ma, Yonglin Deng, Chen Chen, Haonan Lu, and Zhenyu Yang. Glyphdraw2: Automatic generation of complex glyph posters with diffusion models and large language models. *arXiv preprint arXiv:2407.02252*, 2024. 2

[13] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 4

[14] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3

[15] PaddlePaddle. https://github.com/PaddlePaddle/PaddleOCR, 2023. 2, 3

[16] Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. 2

[17] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. In *Advances in Neural Information Processing Systems*, pages 15903–15935. Curran Associates, Inc., 2023. 1

[18] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022. 3

[19] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Computer Vision and Pattern Recognition*, 2016. 4