POSTERTEXT: UNIFYING TEXT PATCH GENERATION AND EDITING FOR E-COMMERCE POSTER DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated e-commerce poster design requires not only the generation of high-quality posters from scratch but also the ability to flexibly and precisely edit existing designs. However, most existing methods either focus on end-to-end poster generation or follow multi-stage design pipelines, with limited capability for flexible and fine-grained editing of existing posters. To enable unified generation and editing of e-commerce posters, we introduce Text Patch Generation and Editing, a unified task formulation that treats text patches as atomic units and covers four fundamental operations: poster generation, patch addition, patch deletion, and patch modification, with optional reference-guided style condition control. Based on this formulation, we propose PosterText, a unified model trained with a four-stage curriculum, including text rendering pretraining, instruction-following training, reinforcement learning for preference alignment, and spatial guidance self-distillation for execution refinement. To facilitate systematic training and evaluation, we further construct a large-scale dataset with fine-grained patch-level annotations, together with a comprehensive benchmark covering both generation and editing scenarios. Extensive experiments demonstrate that PosterText achieves competitive performance against existing generation and editing approaches, validating the effectiveness of the proposed framework.

1 INTRODUCTION

E-commerce posters are a prevalent form of visual content in online commerce, integrating product images, visual backgrounds, and text elements such as slogans, prices, and brand names (Lin et al., 2023; Fan et al., 2026; Qin et al., 2026). These text elements are structured visual units with specific typography, colors, decorations, and spatial layouts, rather than simple strings. In practical scenarios, designers often modify existing posters instead of creating them from scratch, including replacing promotional texts, adjusting typography, adding new campaign elements, or removing outdated information. Therefore, an effective poster automation system should support both high-quality generation and flexible, style-controllable editing based on existing designs.

Existing poster automation methods mainly follow two paradigms. The first adopts a plan-and-render pipeline, which decomposes poster creation into multiple stages such as layout planning and rendering, providing explicit control over the generation process (Li et al., 2023; Hsu et al., 2023). However, these methods are constrained by the dependency between stages, where errors can accumulate throughout the pipeline, and local modifications often require re-executing multiple components. The second explores end-to-end diffusion-based generation (Gao et al., 2025; Chen et al., 2025b; Qin et al., 2026), which directly synthesizes complete posters with improved visual quality. Despite their strong generation capability, these methods mainly focus on zero-to-one creation and provide limited support for localized editing of existing designs. As a result, existing approaches remain insufficient for practical poster design workflows, where designers frequently need to iteratively refine existing layouts and introduce new content while preserving visual consistency. Therefore, developing a unified model that supports both poster generation and editing remains challenging. Such a model needs to achieve accurate character-level text rendering, reconcile the different requirements of open-ended generation and precise local editing, and enable reference-guided style control while maintaining overall visual coherence.

To address these challenges, we formulate *Text Patch Generation and Editing*, where text patches serve as the fundamental units for both generation and editing operations. The proposed framework

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: Overview of PosterText framework and qualitative comparison. (a) Visual demonstration of four fundamental operations: poster generation, patch addition, patch deletion, and patch modification. (b) PosterText enables both poster generation and localized editing within a single unified framework. (c) Qualitative comparison with closed-source methods across multiple tasks.

unifies four operations within a single model: poster generation, patch addition, patch deletion, and patch modification, with optional reference-patch conditioning for style-controlled editing and generation. Based on this formulation, we propose **PosterText**, a unified model trained with a progressive four-stage curriculum. The model first learns accurate text rendering and patch synthesis, then acquires instruction-driven generation and editing capabilities. Subsequently, reinforcement learning aligns the model with human preferences through text accuracy, aesthetic quality, and background preservation rewards, and Spatial Guidance Self-Distillation (SGSD) further improves execution reliability by transferring mask-guided spatial knowledge to mask-free inference. To facilitate research in this direction, we construct **PosterText-320K**, a large-scale poster dataset with patch-level structural annotations and automatically generated editing pairs, and further develop **PosterText-Bench** with a comprehensive evaluation protocol.

The main contributions are summarized as follows:

1. We introduce the task of **text patch generation and editing**, which treats text patches as fundamental units and unifies poster generation, patch addition, patch deletion, and patch modification with reference-guided style control. Extensive experiments across diverse generation and editing scenarios demonstrate the effectiveness of our proposed approach.
2. We propose **PosterText**, a unified model trained with a four-stage curriculum covering text rendering, instruction following, reinforcement learning alignment, and self-distillation refinement, enabling both generation and editing within a unified framework.
3. We construct a large-scale dataset and benchmark with an automated data construction pipeline and comprehensive evaluation protocols.

2 Related Work

2.1 E-commerce and Poster Generation

E-commerce posters are a crucial form of visual content in online commerce, requiring the harmonious integration of products, backgrounds, and marketing text. Automated poster generation

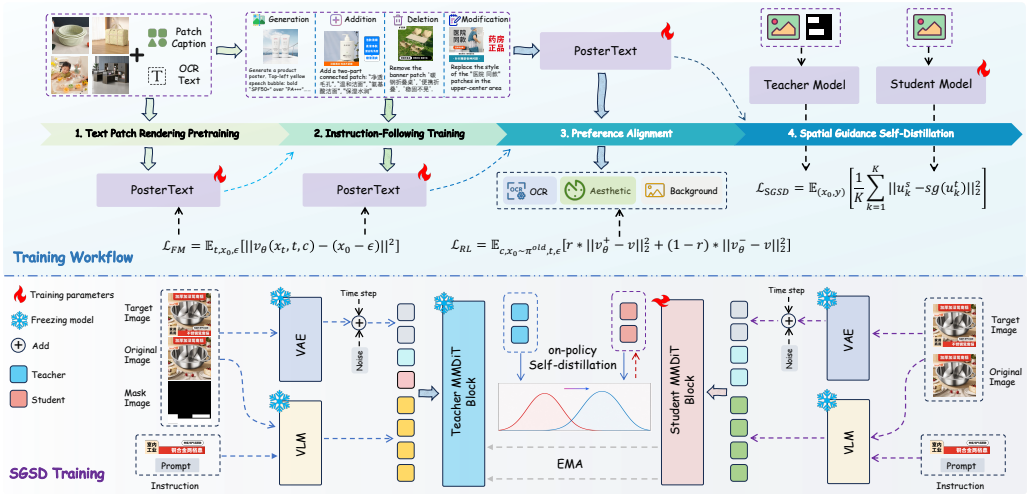


Figure 2: Overview of the proposed PosterText framework. PosterText is trained through a four-stage curriculum : text patch rendering pretraining, instruction-following training, reinforcement learning with human preferences, and Spatial Guidance Self-Distillation (SGSD). The framework supports four fundamental operations: poster generation, patch addition, patch deletion, and patch modification. For SGSD training, the teacher model uses mask-guided inputs for spatial supervision, while the student learns mask-free localization through on-policy self-distillation.

has thus attracted increasing attention, with the key challenge being the creation of visually appealing posters while maintaining product fidelity and text accuracy (Hsu et al., 2023; Chen et al., 2025a; Liu et al., 2026; Chen et al., 2026a). Early approaches typically follow a plan-and-render paradigm, decomposing generation into layout planning and background rendering stages (Li et al., 2023). While these methods provide explicit layout control, the errors accumulated across stages and limited text rendering capability hinder their practical applications. Recent works have explored end-to-end generation by jointly modeling text, layout, and visual appearance, achieving impressive performance in generating complete posters from scratch (Gao et al., 2025; Chen et al., 2025b; Qin et al., 2026). However, real-world e-commerce scenarios often require localized modifications to existing designs, such as replacing promotional texts, adjusting typography styles, or adding new campaign elements. Existing approaches primarily focus on zero-to-one generation and lack the ability to perform localized and controllable modifications on specific poster components, making fine-grained editing challenging and limiting their practical usability. Developing a unified model that can both generate high-quality posters and enable flexible, controllable, and fine-grained content editing remains an open challenge.

2.2 Visual Text Rendering and Editing

Text rendering in images has attracted increasing attention, as diffusion models still struggle with accurate text generation, particularly for complex glyphs (Ma et al., 2023; Tuo et al., 2024a; Ma et al., 2025a; Chen et al., 2026b; Lu et al., 2026). This challenge is especially critical for e-commerce posters, where text conveys essential product information and marketing semantics. Existing studies have improved text rendering from different perspectives. AnyText (Tuo et al., 2024b) and FLUX-Text (Lan et al., 2025) enhanced rendering accuracy through glyph-aware conditioning and improved text-image alignment. Recent works further investigate controllable text generation, with FonTS (Shi et al., 2025) and Calligrapher (Ma et al., 2025b) enabling typography and style customization, while UM-Text (Ma et al., 2026) introduced vision-language guidance for instruction-driven text editing. Despite these advances, existing methods mainly treat text as an independent rendering target, overlooking its interaction with the overall poster composition. In practical e-commerce designs, text is tightly coupled with typography, color, decoration, and spatial layout, requiring coordinated generation and editing with surrounding visual elements. To address this limitation, we propose a unified framework that integrates poster generation with patch-level text editing. By incorporating reference patch conditions, our framework enables style-controllable text creation and modification while maintaining visual consistency.

3 METHOD

Overview. In this paper, we present PosterText, a unified framework for text patch generation and editing in e-commerce posters, as shown in Figure 2. Given a product image and a natural-language instruction, PosterText generates or edits posters by treating text patches as atomic units. The model is trained with a multi-stage curriculum that progressively learns text rendering and patch synthesis, extends to multi-task generation and editing, and further improves alignment and execution accuracy through reinforcement learning and self-distillation.

3.1 TASK FORMULATION

Text Patch Definition. We represent a text patch as $p = (t, s, b, r)$, where t denotes the text content, s encodes style attributes (e.g., font, size, and color), b represents an optional decorative background (set to empty for plain text), and $r = (x, y, w, h)$ specifies its spatial region on the canvas. An e-commerce poster I is composed of a product image I_{prod} and a set of text patches $\mathcal{P} = \{p_1, \ldots, p_n\}$.

Unified Framework. We model all tasks using a single function f_θ, conditioned on an input image I_{in}, a natural-language instruction c, and an optional set of reference patches $\mathcal{R} = \{p_{\text{ref}}^{(1)}, \ldots, p_{\text{ref}}^{(K)}\}$, where $\mathcal{R} = \emptyset$ when no reference patch is provided:

$$I_{\text{out}} = f_\theta(I_{\text{in}}, c, \mathcal{R}). \tag{1}$$

Specifically, the task types are determined by the natural-language instruction c, which specifies the desired operation and its corresponding content or editing requirements: (1) **Poster generation** synthesizes a complete poster from I_{prod}, guided by c describing the desired content, layout, and visual style. (2) **Patch addition** inserts new text patches into an existing poster, where c specifies content and optionally location, while the visual style can be further guided by a reference patch $\mathcal{R}$. (3) **Patch deletion** removes regions specified in c. (4) **Patch modification** updates text content or style within a target region defined by c, optionally conditioned on $\mathcal{R}$.

3.2 STAGE I: TEXT PATCH RENDERING PRETRAINING

Accurate text rendering is fundamental for poster generation, as diffusion models often suffer from character-level errors. Directly training instruction following tends to prioritize global layout while degrading text fidelity. We therefore first establish text rendering and patch generation through pretraining. Stage I includes two progressive sub-tasks. **(A) Text Rendering:** the model receives prompts containing OCR text and spatial positions, and renders the corresponding glyph sequence. **(B) Patch Reconstruction:** the model reconstructs complete text patches from natural-language descriptions, learning to compose content, style, and layout. Training starts with Task A and gradually increases the proportion of Task B. Both tasks are optimized with the flow matching objective:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t, x_0, \epsilon}\left[\|v_\theta(x_t, t, c) - (x_0 - \epsilon)\|^2\right], \tag{2}$$

where $x_t = (1-t)\epsilon + t \cdot x_0$ denotes the interpolated sample at time t. This stage establishes reliable character-level rendering capabilities and then transitions to structured text patch synthesis, forming the foundation for subsequent generation and editing tasks.

3.3 STAGE II: INSTRUCTION-FOLLOWING TRAINING

Building upon the rendering capabilities from Stage I, the model is further trained to follow instructions and perform diverse operations, including poster generation, patch addition, modification, and deletion. We formulate all tasks with a unified input format $(I_{\text{in}}, c, \mathcal{R})$, where the instruction c specifies the desired operation, target content, and optional style conditions. Training data is constructed by decomposing existing posters into operation tuples, allowing the model to learn different generation and editing behaviors under the same instruction framework. During Stage II, we jointly train the model with both mask-guided and mask-free samples, allowing it to acquire stronger spatial editing capabilities with explicit masks while supporting mask-free inference.

All operations are jointly optimized with the flow matching objective, with balanced sampling across tasks to ensure stable multi-task learning. After this stage, the model can interpret instructions and perform different poster generation and editing operations, including style-controllable generation and modification with optional reference patches.

3.4 Stage III: Preference Alignment

The first two stages optimize pixel-level reconstruction objectives, which are insufficient to capture higher-level perceptual criteria in e-commerce posters, including text readability, visual aesthetics, and background consistency. Building upon the instruction-following capability acquired in Stage II, we further align the model with human preferences through reinforcement learning.

Optimization Objective. We adopt DiffusionNFT (Zheng et al., 2025), an online RL algorithm compatible with the flow matching framework. Given online rollouts with normalized rewards r, the model is optimized by jointly improving high-reward behaviors and suppressing low-reward behaviors through implicit positive and negative policies:

$$\mathcal{L}_{\text{RL}} = \mathbb{E}_{c,x_0\sim\pi^{\text{old}},t,\epsilon}\left[r\cdot\|v_\theta^+ - v\|_2^2 + (1-r)\|v_\theta^- - v\|_2^2\right], \tag{3}$$

where $v = x_0 - \epsilon$ denotes the target velocity and v^{old} is the EMA reference policy. The positive and negative branches are derived from the same velocity field, allowing the model to improve toward preferred outputs while preserving the flow matching formulation.

Reward Design. We design a composite reward specifically tailored to the requirements of e-commerce poster generation and editing:

$$R = \alpha R_{\text{text}} + \beta R_{\text{aesthetic}} + \delta R_{\text{background}}, \tag{4}$$

where R_{text} measures OCR-based text accuracy to evaluate character-level rendering fidelity and textual correctness; $R_{\text{aesthetic}}$ assesses the overall visual quality and visual coherence; and $R_{\text{background}}$ measures the preservation of non-edited regions to ensure consistency with the original image.

3.5 Stage IV: Spatial Guidance Self-Distillation

After preference alignment, the model demonstrates significant improvements in overall quality, but still suffers from execution errors in complex scenarios such as ghost text, missing characters, and incorrect operations. These failures mainly arise from insufficient localization and decomposition of complex instructions, making it difficult for the model to accurately determine where and how each operation should be performed (Baron et al., 2026). The sparse reward signal in RL only evaluates the final output and making it difficult to effectively attribute global feedback to specific intermediate-step errors. To improve execution accuracy, we introduce SGSD, which transfers the spatial guidance from mask-guided editing to a mask-free setting. Since explicit masks can provide precise region information and improve editing quality (Guo et al., 2026), SGSD aims to enable the model to achieve comparable execution quality without additional mask inputs.

During training, we construct teacher-student pairs with asymmetric conditions. The teacher receives an mask-guided condition $M_t = (I_{\text{in}}, c, \mathcal{R}, \text{mask})$, where the mask provides explicit spatial information for the target operation, while the student receives a mask-free condition $M_s = (I_{\text{in}}, c, \mathcal{R})$. The student samples intermediate states along its generation trajectory, and the teacher provides velocity predictions at the same states as distillation targets. The training objective is:

$$\mathcal{L}_{\text{SGSD}} = \mathbb{E}_{(x_0,y)}\left[\frac{1}{K}\sum_{k=1}^{K}\left\|u_k^s - \text{sg}(u_k^t)\right\|_2^2\right], \tag{5}$$

$$u_k^s = \nu_\theta(x_{t_k}^s, t_k, M_s), \quad u_k^t = \nu_{\bar{\theta}}(x_{t_k}^s, t_k, M_t), \tag{6}$$

where $\text{sg}(\cdot)$ represents the stop-gradient operation, K denotes the number of intermediate timesteps sampled along each student trajectory, and $\bar{\theta}$ denotes the teacher parameters updated by EMA. Un-

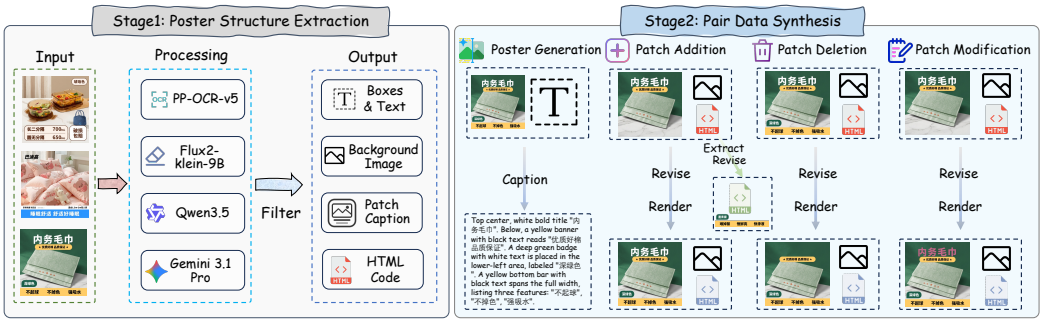


Figure 3: Overview of data pipeline. The pipeline extracts patch-level structural annotations from raw posters and constructs operation pairs for generation, addition, deletion, and modification.

like teacher-forcing based SFT, SGSD provides supervision along the student's own generation trajectory, reducing the train-inference discrepancy. The dense intermediate supervision across multiple time steps provides fine-grained correction for intermediate steps that sparse RL rewards cannot reach, further improving execution precision in complex mask-free scenarios.

4 Dataset and Benchmark

Existing text generation datasets mainly focus on text rendering from scratch, while lacking editing operation pairs and patch-level structural annotations. To support systematic research on text patch generation and editing, we construct **PosterText-320K**, a large-scale annotated poster dataset containing 120K posters for editing tasks and 200K posters for generation tasks. In addition, we introduce **PosterText-Bench** with a comprehensive evaluation protocol. The overall data construction pipeline first extracts fine-grained patch-level structural annotations from raw posters, which are then used to derive diverse operation pairs covering generation, addition, deletion, and modification.

4.1 Data Construction Pipeline

Phase I: Poster Structure Extraction. We collect a large number of e-commerce posters from public platforms and automatically recover patch-level structural information through a multi-stage pipeline, as illustrated in Figure 3: (1) text regions are detected and OCR is applied to obtain text content and spatial locations; (2) Qwen3.5-27B (Team, 2026) VLM is used to describe visual attributes of each text patch, including font, color, background blocks, and text effects; (3) Flux.2-Klein-9B (Labs, 2025) is applied to remove all text patches from posters, producing corresponding product images; (4) Gemini-3.1 Pro (Google, 2025) is used to extract structured HTML representations from the posters; (5) low-quality samples are filtered based on OCR confidence, image resolution, perceptual similarity, and residual text artifacts in the generated product images.

Phase II: Paired Data Synthesis. Based on the extracted poster structures, we construct training pairs for four operations: generation, addition, deletion, and modification. We introduce an HTML-based structured representation that encodes each text patch with its content, visual attributes, and spatial layout, enabling precise patch-level manipulation and reference-based editing. Using the structured representation, we construct training pairs for four operations: (1) **Generation:** patch annotations are converted into natural-language descriptions to create product-conditioned poster generation pairs. (2) **Addition:** existing patches are removed and re-rendered to construct addition pairs, with HTML attributes providing reference patch information for style-conditioned generation. (3) **Deletion:** target patches are removed according to their structural locations to generate deletion pairs. (4) **Modification:** patch content or visual attributes are changed while preserving the original layout, with HTML attributes enabling reference-guided style-controlled editing.

4.2 Evaluation Protocol

We evaluate model performance from three perspectives: text fidelity, background preservation, and editing quality, covering both the accuracy of generated content and the visual quality of posters.

Table 1: Quantitative results on PosterText-Bench. **Bold** and underline indicate the best and second-best results, respectively. S.A.: Sentence Accuracy; M-L.: Masked-LPIPS; I.F.: Instruction Following; V.Q.: Visual Quality; B.P.: Background Preservation. Higher ↑ is better for S.A., NED, I.F., V.Q., and B.P., while lower ↓ is better for M-L.

	Patch Addition					Patch Deletion			Patch Modification					Poster Generation		
Method	**S.A.**	**NED**	**M-L.**	**I.F.**	**V.Q.**	**M-L.**	**I.F.**	**V.Q.**	**S.A.**	**NED**	**M-L.**	**I.F.**	**V.Q.**	**I.F.**	**V.Q.**	**B.P.**
Closed-Source Methods																
Seedream 5.0	0.403	0.585	0.195	4.40	4.64	0.122	4.68	4.69	0.735	0.884	0.102	4.33	4.65	4.10	4.50	4.80
Nano Banana 2	0.500	0.635	0.064	4.46	4.67	0.051	4.62	4.84	**0.819**	**0.907**	0.034	4.47	**4.80**	**4.63**	4.76	4.86
Nano Banana Pro	0.543	0.679	0.184	**4.55**	**4.75**	0.061	4.66	**4.91**	0.815	0.891	0.102	4.36	4.64	4.58	**4.88**	**4.92**
Open-Source Methods																
FireRed-Image-Edit	0.309	0.481	0.497	3.53	3.61	0.185	4.29	4.04	0.673	0.803	0.305	3.71	3.78	3.86	4.21	4.42
Flux.2-Klein-9B	0.163	0.284	0.100	1.51	2.55	0.098	4.02	4.57	0.375	0.573	0.062	2.16	3.02	3.12	3.68	4.24
Qwen-Image-Edit	0.378	0.563	0.647	3.31	3.31	0.323	3.08	2.76	0.602	0.769	0.280	3.12	3.36	3.58	4.06	2.92
PosterText	**0.763**	**0.899**	**0.039**	4.51	4.61	**0.040**	**4.70**	4.78	0.771	0.902	**0.028**	**4.53**	4.64	4.21	4.46	4.88

Text Fidelity. Sentence Accuracy (Sen. Acc) and Normalized Edit Distance (NED) are used to evaluate text rendering accuracy, measuring sentence-level exact correctness and character-level similarity between generated and ground-truth texts, respectively.

Background Preservation. Masked-LPIPS (Zhang et al., 2018) is used to evaluate background preservation by measuring perceptual differences between input and output images in non-edited regions. For poster generation, a VLM-based scorer is instead used to assess whether the product subject and content are faithfully preserved in the generated output.

Editing Quality. A vision-language model is used to evaluate editing quality from two aspects: instruction following, which measures whether the requested operation is correctly executed, and visual quality, which evaluates the coherence, style consistency, and naturalness of the edited results.

5 Experiments

Implementation Details. We build PosterText upon Qwen-Image-Edit (Wu et al., 2025) and fine-tune it with LoRA. Stage I is trained for 50K steps with a cosine learning rate schedule and an initial learning rate of 1×10^{-4}. The sampling ratio between text rendering task and patch reconstruction task gradually changes from 4:1 to 1:1 during the first 20K steps, with LoRA rank set to 128. Stage II is trained for 10K steps for instruction following. Stage III applies DiffusionNFT for RL alignment with 200 training steps, while Stage IV performs self-distillation with $K = 20$ and 500 training steps. More implementation details are provided in the supplementary material.

Compared Methods. For evaluation, we compare PosterText with representative open-source and closed-source methods that support both poster generation and editing. The open-source methods include Flux.2-Klein-9B (Labs, 2025), Qwen-Image-Edit (Wu et al., 2025), and FireRed-Image-Edit (Zhou et al., 2025). The closed-source methods include Seedream 5.0 (Seedream Team, 2025), Nano Banana 2 (DeepMind, 2025a), and Nano Banana Pro (DeepMind, 2025b).

5.1 Experimental Results

Overall Comparison. We evaluate the compared methods across tasks on PosterText-Bench, with results summarized in Table 1. PosterText achieves competitive performance in text fidelity, background preservation, and instruction following, outperforming open-source baselines and even surpassing closed-source models on key metrics. In terms of visual quality, PosterText narrows the gap with closed-source systems on localized editing, while remaining behind them in poster generation, demonstrating competitive editing quality and promising generation performance. As shown in Figure 1 (c) and 4, PosterText can better leverage complex style references to generate corresponding text patches while maintaining accurate text rendering and clean backgrounds. In contrast, open-source baselines often produce blurry or corrupted characters, while closed-source models generally achieve higher visual quality but may modify non-target regions or introduce unintended content.



Figure 4: Visual comparison of PosterText with existing methods.

Results on Poster Editing. For Patch Addition, PosterText achieves the best text fidelity among all methods, with Sen.Acc of 0.763 and NED of 0.899, outperforming the closed-source systems. It also obtains the lowest Masked-LPIPS score, demonstrating that PosterText can insert new patches while effectively preserving the original poster structure. For Patch Deletion, which mainly evaluates region localization and background restoration rather than text rendering, PosterText achieves competitive Masked-LPIPS and instruction-following scores, indicating more accurate spatial control and cleaner content removal. Patch Modification represents the most challenging setting, as it requires simultaneous text regeneration, precise target localization, and preservation of surrounding regions. PosterText achieves strong performance in background preservation and instruction following, demonstrating effective localized editing while maintaining overall visual consistency. Some closed-source models achieve higher text fidelity on this task, which may benefit from large-scale proprietary training data and broader text rendering supervision.

Results on Poster Generation. In the Poster Generation task, closed-source models maintain advantage in instruction following and visual quality, benefiting from larger-scale training data, stronger aesthetic alignment, and more capable foundation models. Meanwhile, PosterText achieves comparable performance in product fidelity, demonstrating that the region control and content preservation abilities learned from editing tasks can effectively transfer to the generation setting. Despite their strong performance, closed-source models still face practical limitations in e-commerce deployment, including high costs for large-scale generation due to pay-per-use pricing, limited adaptability caused by the lack of fine-tuning access, and potential concerns regarding commercial data privacy when uploading product images to external services. In contrast, PosterText provides a more controllable and deployment-friendly solution for practical poster generation scenarios.

Analysis and Discussion. The comparison results demonstrate that PosterText addresses the key challenge: unifying poster generation and fine-grained patch-level editing within a single model. Across three editing tasks, PosterText achieves strong performance in text fidelity and region preservation, two essential requirements often underemphasized by existing generation-oriented methods. By combining accurate text rendering with precise local control under a unified instruction framework, PosterText bridges the gap between text rendering models and general-purpose image editing systems. For poster generation, the performance gap with closed-source models mainly reflects differences in training scale and optimization focus, while the strong product fidelity of PosterText confirms that editing-oriented region control can effectively transfer to generation scenarios.

Table 2: Ablation study of different training stages, averaged across all tasks.

Model	S.A.↑	NED↑	M-L.↓	I.F.↑	V.Q.↑
Base Model	0.490	0.666	0.417	3.27	3.37
+SFT Stage					
Stage II-only	0.679	0.786	0.088	3.63	3.81
Stage I(no cur.)+II	0.702	0.831	0.061	4.00	4.12
Stage I + II	0.712	0.866	0.053	4.03	4.24
+SFT+RL	0.735	0.883	0.037	4.22	4.51
+SFT+RL+SGSD	0.767	0.901	0.036	4.49	4.62

5.2 Ablation Study

To evaluate the contribution of each training stage, we conduct a systematic ablation study using task-averaged metrics, with the results summarized in Table 2. Specifically, Sen.ACC and NED are averaged over the Addition and Modification tasks for text rendering; Masked-LPIPS is averaged over Addition, Deletion, and Modification for region preservation; instruction following and visual quality scores are averaged across all four tasks for overall performance.

Analysis of the SFT stage. Without task-specific training, the base model achieves only 0.490 S.A. and 0.417 M-L., indicating limited capability for fine-grained patch-level editing. Stage II-only substantially improves performance, increasing S.A. to 0.679 and reducing M-L. to 0.088, demonstrating that instruction-following training establishes basic editing capabilities. Adding Stage I pretraining (Stage I(no cur.)+II) further improves NED from 0.786 to 0.831, confirming that text rendering pretraining strengthens character-level representation. Introducing the curriculum sampling strategy (Stage I+II) brings additional but modest improvements, mainly by providing a more stable initialization for subsequent RL optimization.

Analysis of the RL stage. Building on full SFT, RL further improves editing performance and overall visual quality. This demonstrates that pixel-level reconstruction objectives alone cannot fully capture human preferences, such as text correctness, aesthetic quality, and instruction compliance. The multi-dimensional reward design effectively guides different aspects of optimization: OCR-based rewards improve text fidelity, aesthetic rewards enhance visual quality, and instruction-based rewards improve task execution. The significant reduction in M-L. (30.2%) further indicates that the background preservation reward explicitly constrains editing boundaries, thereby reducing unintended modifications in non-target regions.

Analysis of the SGSD stage. SGSD further enhances text accuracy and instruction following through dense trajectory-level supervision, which corrects intermediate-step errors beyond the reach of sparse reward signals. The improvement in instruction following indicates that spatial knowledge from the mask-guided teacher can be effectively transferred to mask-free inference for more accurate region localization. Since RL already provides strong background preservation, SGSD mainly improves text rendering precision and execution reliability. Together, RL and SGSD offer complementary benefits: RL optimizes overall editing quality, while SGSD improves execution accuracy.

6 Conclusion

In this work, we formulate the task of Text Patch Generation and Editing, establishing text patches as fundamental units to unify poster generation, patch addition, deletion, and modification within a single architecture. Built upon this formulation, we introduce PosterText, a unified framework trained through a progressive four-stage curriculum encompassing accurate text rendering, instruction following, multi-reward preference alignment, and spatial guidance self-distillation for reliable execution. To support benchmarked evaluation, we construct the large-scale PosterText-320K dataset alongside PosterText-Bench with systematic assessment protocols. Extensive experiments demonstrate that PosterText achieves superior performance across visual aesthetics, text rendering accuracy, and localized editing fidelity, confirming the effectiveness of unifying poster generation and fine-grained patch-level editing in a single model.

REFERENCES

Yoav Baron, Sara Dorfman, Roni Paiss, Daniel Cohen-Or, and Or Patashnik. Analysis-by-proxy: Localization signals in vlms operating as condition encoders. *arXiv preprint arXiv:2607.06445*, 2026.

Haoyu Chen, Xiaojie Xu, Wenbo Li, Jingjing Ren, Tian Ye, Songhua Liu, Ying-Cong Chen, Lei Zhu, and Xinchao Wang. Posta: A go-to framework for customized artistic poster generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 28694–28704, 2025a.

SiXiang Chen, Jianyu Lai, Jialin Gao, Tian Ye, Haoyu Chen, Hengyu Shi, Shitong Shao, Yunlong Lin, Song Fei, Zhaohu Xing, et al. Postercraft: Rethinking high-quality aesthetic poster generation in a unified framework. *arXiv preprint arXiv:2506.10741*, 2025b.

Sixiang Chen, Jianyu Lai, Jialin Gao, Hengyu Shi, Zhongying Liu, Tian Ye, Junfeng Luo, Xiaoming Wei, and Lei Zhu. Posteromni: Generalized artistic poster creation via task distillation and unified reward feedback. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5978–5987, 2026a.

Zeyu Chen, Fangmin Zhao, Yan Shu, Yichao Liu, Liu Yu, and Yu Zhou. Styletextgen: Style-conditioned multilingual scene text generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7643–7653, 2026b.

Google DeepMind. Gemini image: High-quality image generation, 2025a. URL `https://deepmind.google/models/gemini-image/flash`. Accessed 2026-06-15.

Google DeepMind. Gemini image pro: High-quality image generation, 2025b. URL `https://deepmind.google/models/gemini-image/pro`. Accessed 2026-06-15.

Jiahao Fan, Yuxin Qin, Wei Feng, Yanyin Chen, Yaoyu Li, Ao Ma, Yixiu Li, Li Zhuang, Haoyi Bian, Zheng Zhang, et al. Autopp: Towards automated product poster generation and optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 3768–3776, 2026.

Yifan Gao, Zihang Lin, Chuanbin Liu, Min Zhou, Tiezheng Ge, Bo Zheng, and Hongtao Xie. Postermaker: Towards high-quality product poster generation with accurate text rendering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8083–8093, 2025.

Google. Gemini 3: Introducing the latest gemini ai model from googles, 2025. URL `https://blog.google/products/gemini/gemini-3`. Accessed 2026-05-18.

Zipeng Guo, Xiaoan Liu, Lichen Ma, Cheng Wang, Yu He, Xiaolong Fu, Jingling Fu, Xinyuan Shan, Shaojie Guo, Luohang Liu, et al. Gmo-e^{2}dit: Grounded multi-operation editing for e-commerce images. *arXiv preprint arXiv:2607.00920*, 2026.

Hsiao Yuan Hsu, Xiangteng He, Yuxin Peng, Hao Kong, and Qing Zhang. Posterlayout: A new benchmark and approach for content-aware visual-textual presentation layout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6018–6026, 2023.

Black Forest Labs. Flux.2 [klein]: Towards interactive visual intelligence, 2025. URL `https://bfl.ai/blog/flux2-klein-towards-interactive-visual-intelligence`.

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. *arXiv preprint arXiv:2505.03329*, 2025.

Zhaochen Li, Fengheng Li, Wei Feng, Honghe Zhu, Yaoyu Li, Zheng Zhang, Jingjing Lv, Junjie Shen, Zhangang Lin, Jingping Shao, et al. Planning and rendering: Towards product poster generation with diffusion models. *arXiv preprint arXiv:2312.08822*, 2023.

Jinpeng Lin, Min Zhou, Ye Ma, Yifan Gao, Chenxi Fei, Yangjian Chen, Zhang Yu, and Tiezheng Ge. Autoposter: A highly automatic and content-aware design system for advertising poster generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 1250–1260, 2023.

Junle Liu, Peirong Zhang, Yuyi Zhang, Pengyu Yan, Hui Zhou, Xinyue Zhou, Fengjun Guo, and Lianwen Jin. Posterverse: A full-workflow framework for commercial-grade poster generation with html-based scalable typography. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 7197–7205, 2026.

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*, volume 40, pp. 7565–7573, 2026.

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. *arXiv preprint arXiv:2303.17870*, 2023.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 5955–5963, 2025a.

Lichen Ma, Xiaolong Fu, Gaojing Zhou, Zipeng Guo, Ting Zhu, Yichun Liu, Yu Shi, Jason Li, and Junshi Huang. Um-text: A unified multimodal model for image understanding. *arXiv preprint arXiv:2601.08321*, 2026.

Yue Ma, Qingyan Bai, Hao Ouyang, Ka Leong Cheng, Qiuyu Wang, Hongyu Liu, Zichen Liu, Haofan Wang, Jingye Chen, Yujun Shen, et al. Calligrapher: Freestyle text image customization. *arXiv preprint arXiv:2506.24123*, 2025b.

Yuxin Qin, Ke Cao, Haowei Liu, Ao Ma, Fengheng Li, Honghe Zhu, Zheng Zhang, Run Ling, Wei Feng, Xuanhua He, et al. Innoads-composer: Efficient condition composition for e-commerce poster generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32988–32999, 2026.

Seedream Team. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint arXiv:2509.20427*, 2025.

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*, pp. 18463–18474, 2025.

Qwen Team. Qwen3.5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. Anytext2: Visual text generation and editing with customizable attributes. *arXiv preprint arXiv:2411.15245*, 2024a.

Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. In *International Conference on Learning Representations*, volume 2024, pp. 56783–56799, 2024b.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

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. *arXiv preprint arXiv:2509.16117*, 2025.

Jun Zhou, Jiahao Li, Zunnan Xu, Hanhui Li, Yiji Cheng, Fa-Ting Hong, Qin Lin, Qinglin Lu, and Xiaodan Liang. Fireedit: Fine-grained instruction-based image editing via region-aware vision language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

SUPPLEMENTARY MATERIAL FOR POSTERTEXT

In this supplementary material, we provide additional technical details and qualitative evidence omitted from the main paper due to space constraints, together with information that supports reproducibility. We first report the implementation details and stage-wise training setup because they are prerequisite to reproducing the main results. We then expand the method design, the PosterText-320K data construction pipeline, and the PosterText-Bench evaluation protocol, and conclude with additional visual comparisons and a discussion of limitations. Specifically, the supplementary material contains:

- A statement of large language model usage during manuscript preparation and of reproducibility;
- Implementation details of PosterText, including the shared configuration and the four-stage training setup;
- An expanded description of the method design, covering rendering pretraining, instruction-following training, preference alignment, and Spatial Guidance Self-Distillation (SGSD);
- The construction procedure of PosterText-320K;
- The PosterText-Bench evaluation protocol;
- Additional visual comparisons across poster generation, patch addition, patch deletion, and patch modification;
- Discussion of current limitations and potential directions for future work.

STATEMENT ON THE USE OF LARGE LANGUAGE MODELS

In the preparation of this paper, we used large language models (LLMs) as an auxiliary tool. The primary role of LLMs was to assist with language polishing, phrasing improvements, grammar checking, and enhancing the clarity and fluency of certain sentences, ensuring that our research ideas and technical details are expressed more accurately and professionally. Importantly, LLMs were not involved in the core aspects of this work, including research conception, result generation, or data analysis.

REPRODUCIBILITY STATEMENT

We state that PosterText is highly reproducible. Complete implementation details, network architectures, and training hyperparameters across all four curriculum stages are thoroughly documented in the Method section and Appendix. In addition, the dataset construction pipeline and data processing procedures for PosterText-320K are detailed in the supplementary material to support full reproducibility of our work.

A IMPLEMENTATION DETAILS

PosterText follows the same unified formulation as the main paper: all operations are represented by an input image I_{in}, a natural-language instruction c, and an optional set of reference patches $\mathcal{R}$. The implementation keeps this interface unchanged across poster generation, patch addition, patch deletion, and patch modification, so the same model can be used for both localized editing and generation from a product image. This unified interface is important because it avoids task-specific model branches: the difference between adding a new text patch, removing an outdated patch, modifying an existing patch, or generating a complete poster is expressed through the instruction and optional references rather than through separate architectures.

A.1 SHARED CONFIGURATION

PosterText is built on Qwen-Image-Edit and trained with LoRA fine-tuning. The base model weights are frozen, while the LoRA parameters are updated across the four stages. Each stage starts from the

converged LoRA weights of the previous stage rather than from a separate initialization, which forms a curriculum from basic glyph rendering to instruction following, preference alignment, and mask-free execution refinement. Unless otherwise specified, training uses bf16 precision and AdamW. At inference time, we use 40 sampling steps and a CFG scale of 3.5.

Table 3: Shared implementation and inference configuration.

Item	Configuration
Base model	Qwen-Image-Edit
Fine-tuning	LoRA (base frozen)
Precision	bf16
Optimizer	AdamW
Sampling steps	40
CFG scale	3.5

A.2 Stage-wise Training Setup

Table 4 gives the stage-wise hyperparameters. Stage I uses the largest number of steps because it establishes character-level rendering and structured patch synthesis from reconstruction supervision. Stage II then converts the rendering capability into instruction-following behavior over the four task types. Stage III and Stage IV require fewer steps because they refine an already trained policy: Stage III aligns outputs with text accuracy, aesthetics, and background preservation, while Stage IV transfers mask-guided spatial behavior into the mask-free inference setting.

Table 4: Stage-wise training hyperparameters. Stage I anneals the sampling ratio of Task A to Task B from 4:1 to 1:1 within the first 20K steps.

Stage	Batch	Steps	LR	Key setting
I. Rendering	128	50K	1×10^{-4}	cosine; rank 128; A:B $4{:}1\rightarrow1{:}1$
II. Instruction	128	10K	5×10^{-5}	cosine; joint mask/mask-free
III. RL (NFT)	32	200	1×10^{-5}	EMA 0.99; group-norm reward
IV. SGSD	16	500	1×10^{-5}	EMA 0.99; $K{=}20$

B Additional Method Details

This section provides the details that are compressed in the main paper. The goal is not to introduce a different method, but to make explicit how the four-stage pipeline is implemented and why each stage supports the unified poster generation and editing framework. The design follows a progressive principle: Stage I builds a glyph and patch prior, Stage II turns this prior into instruction-following behavior, Stage III aligns outputs with poster-level preferences, and Stage IV improves mask-free execution by distilling spatial behavior from a stronger mask-guided model.

B.1 Stage I: Rendering Pretraining

Stage I separates the low-level challenge of rendering readable characters from the broader challenge of synthesizing a complete styled text patch. Both sub-tasks use the flow matching objective in the main paper, but they differ in the information provided to the model.

Task A: Text Rendering. The model receives the target OCR string and the spatial region in which the text should appear. The supervision emphasizes whether the correct glyph sequence is rendered in the specified area. This task is intentionally simple: it removes most decorative and contextual variation so that the model first learns stable character shapes, spacing, and short text composition.

Task B: Patch Reconstruction. The model then reconstructs a full text patch from a natural-language description that contains text content, style attributes, and optional decorative backgrounds.

Compared with Task A, this task requires the model to bind content with font, color, layout, and background blocks, which more closely matches the patches used in e-commerce posters.

During the first 20K steps, the sampling ratio of Task A to Task B is annealed from 4:1 to 1:1. This schedule gives the model a strong character-rendering prior at the beginning and gradually increases the proportion of structured patch synthesis. The ablation results in the main paper show that this curriculum is important: removing the curriculum weakens both text fidelity and downstream instruction-following quality.

B.2 Stage II: Instruction-Following Data Interface

Stage II uses the same input format for all tasks, $(I_{\text{in}}, c, \mathcal{R})$. The instruction c specifies the operation type, target content, and spatial or semantic target when available. The reference set $\mathcal{R}$ is empty for operations that do not require a style reference, and contains one or more reference patches for style-guided addition or modification. During training, we mix mask-guided and mask-free samples. The mask-guided samples teach accurate localization and region preservation, while mask-free samples match the final inference setting in which users only provide an instruction and optional references.

This design is consistent with the main paper's task formulation. Poster generation uses I_{prod} as the input image and asks the model to synthesize a complete poster layout. Patch addition inserts new text patches into an existing poster. Patch deletion removes the specified patch regions and restores the background. Patch modification changes text content or visual attributes while preserving the surrounding layout.

B.3 Stage III: Preference Alignment

Pixel reconstruction alone cannot fully capture whether a poster edit is useful: an output can be close in pixels but still contain wrong characters, visually awkward typography, or unwanted changes outside the edit region. Stage III therefore adopts DiffusionNFT Zheng et al. (2025), an online reinforcement learning method compatible with flow matching, to align the model with higher-level preferences.

Following the main paper, the composite reward is

$$R = \alpha R_{\text{text}} + \beta R_{\text{aesthetic}} + \delta R_{\text{background}}, \tag{7}$$

where R_{text} measures OCR-based text correctness, $R_{\text{aesthetic}}$ scores overall visual quality with Qwen3.5-27B Team (2026), and $R_{\text{background}}$ measures preservation of non-edited regions with Masked-LPIPS. Here Qwen3.5-27B is used only as the reward scorer during alignment, while the benchmark evaluation in Section D uses a separate VLM judge. We empirically set $\alpha = 0.4$, $\beta = 0.2$, and $\delta = 0.4$. This setting is used to balance the two requirements that are most critical for poster editing, namely correct text rendering and preservation of non-target regions, while keeping aesthetics as an auxiliary quality signal. We do not treat this weighting as a separate contribution; it simply defines the reward used in Stage III.

For each prompt, multiple online rollouts are scored and group-normalized. The normalized reward r is then used in the DiffusionNFT objective in the main paper. This normalization makes the update depend on the relative quality of samples under the same instruction, reducing sensitivity to the absolute scale of OCR, aesthetic, and preservation scores.

B.4 Stage IV: Spatial Guidance Self-Distillation

Stage IV addresses a gap between training and deployment. Explicit masks provide strong spatial cues during editing, but a practical poster editing system should also work from natural-language instructions without requiring the user to draw masks. SGSD transfers the spatial behavior of a mask-guided teacher to a mask-free student.

The teacher and student share the same base architecture but receive different conditions. The teacher condition is $M_t = (I_{\text{in}}, c, \mathcal{R}, \text{mask})$, where the mask marks the target region. The student condition is $M_s = (I_{\text{in}}, c, \mathcal{R})$, which matches inference. The student first rolls out intermediate states $x^s_{t_k}$ along its own trajectory; the teacher then predicts velocities at these same states under the

stronger mask-guided condition. The objective is

$$\mathcal{L}_{\text{SGSD}} = \mathbb{E}_{(x_0,y)}\left[\frac{1}{K}\sum_{k=1}^{K}\left\|u_k^s - \text{sg}(u_k^t)\right\|_2^2\right], \tag{8}$$

$$u_k^s = \nu_\theta(x_{t_k}^s, t_k, M_s), \quad u_k^t = \nu_{\bar{\theta}}(x_{t_k}^s, t_k, M_t), \tag{9}$$

where $\text{sg}(\cdot)$ is the stop-gradient operator, $\bar{\theta}$ denotes the EMA teacher, the EMA decay is 0.99, and $K = 20$. Because supervision is applied on the student's own trajectory, SGSD reduces train-inference mismatch more directly than teacher-forcing SFT. It also provides dense intermediate-step correction that complements the sparse final-output reward used in Stage III.

Algorithm 1 Spatial Guidance Self-Distillation (SGSD)

Require: student LoRA parameters θ; EMA teacher $\bar{\theta}$; dataset $\mathcal{D} = \{(I_{\text{in}}, c, \mathcal{R}, \text{mask}, x_0)\}$; steps K; EMA decay m=0.99
Ensure: mask-free student parameters θ

for iter $= 1$ to 500 **do**
  sample $(I_{\text{in}}, c, \mathcal{R}, \text{mask}, x_0) \sim \mathcal{D}$
  $M_s \leftarrow (I_{\text{in}}, c, \mathcal{R})$; $M_t \leftarrow (I_{\text{in}}, c, \mathcal{R}, \text{mask})$
  sample timesteps $\{t_1, \ldots, t_K\}$ in $[0, 1]$
  for $k = 1$ to K **do**
    roll out student state $x_{t_k}^s$ under M_s
    $u_k^s \leftarrow \nu_\theta(x_{t_k}^s, t_k, M_s)$
    $u_k^t \leftarrow \nu_{\bar{\theta}}(x_{t_k}^s, t_k, M_t)$
  end for
  $\mathcal{L} \leftarrow \frac{1}{K}\sum_k \|u_k^s - \text{sg}(u_k^t)\|_2^2$
  update θ with AdamW using $\nabla_\theta \mathcal{L}$
  $\bar{\theta} \leftarrow m\,\bar{\theta} + (1-m)\,\theta$
end forθ

Mask-guided transfer analysis. Table 5 provides an editing-only auxiliary analysis of the mask-free NFT model, the mask-guided teacher after NFT alignment, and the final mask-free student after SGSD. S.A. and NED are averaged over addition and modification, while M-L., I.F., and V.Q. are averaged over addition, deletion, and modification. This table is intended to isolate spatial transfer in localized editing, rather than to replace the main ablation table. The mask-guided teacher provides the strongest spatial control because it receives explicit region masks. This setting is not the intended deployment interface, but it reveals the benefit of accurate spatial guidance for patch-level poster editing. SGSD then transfers this capability to the mask-free student, allowing the final model to approach the teacher while preserving the mask-free inference interface. Therefore, the comparison supports the main paper's conclusion from a complementary angle: the final model improves execution reliability not by requiring extra user-provided masks, but by learning from a stronger mask-guided trajectory during training.

Table 5: Effect of mask-guided spatial transfer. NFT denotes the preference-aligned model after Stage III. The mask-guided teacher uses explicit region masks, while the SGSD student keeps the mask-free inference interface.

Model	S.A.↑	NED↑	M-L.↓	I.F.↑	V.Q.↑
Stage I+II+NFT (mask-free)	0.735	0.883	0.037	4.22	4.51
Stage I+II+NFT (mask-guided)	0.781	0.904	0.033	4.58	4.64
Stage I+II+NFT+SGSD (mask-free)	0.767	0.901	0.036	4.49	4.62

C PosterText-320K Construction

PosterText-320K contains 120K posters for editing tasks and 200K posters for generation tasks, all with patch-level annotations. The construction follows the two-phase pipeline summarized in the main paper: first recover structured poster annotations, then convert the annotations into operation pairs for the four tasks. We keep the two subsets conceptually separate because they serve different

learning signals. The generation subset teaches the model to synthesize a complete poster from a clean product image, while the editing subset emphasizes localized patch manipulation on an existing design. Both subsets, however, share the same patch schema so that the model can learn a unified interface rather than task-specific input formats.

C.1 Phase I: Poster Structure Extraction

We collect raw e-commerce posters from public platforms and recover text-patch structure with the following pipeline. First, text detection and OCR locate each text region and recognize its content, producing bounding boxes (x, y, w, h). Second, Qwen3.5-27B Team (2026) describes visual attributes of each patch, including font, color, decorative blocks, and text effects. Third, Flux.2-Klein-9B Labs (2025) removes all text patches to produce a clean product image I_{prod}. Fourth, Gemini-3.1 Pro Google (2025) extracts a structured HTML representation that records text content, layout, and visual attributes. Finally, we filter low-quality samples using OCR confidence, resolution, and residual-text checks.

The structured representation is designed to be both machine-operable and instruction-friendly. Each text patch stores its recognized string, bounding box, coarse reading order, and visual descriptors such as foreground color, approximate font style, decoration type, and background block information. These fields are sufficient to recover the target region for deletion and modification, to identify style references for addition, and to serialize the whole poster into natural-language instructions for generation. We use HTML as an intermediate representation because it naturally expresses nested text elements, spatial layout, and style attributes, while remaining easier to validate and edit than free-form captions.

The filtering stage applies four main checks. We retain OCR annotations only when the confidence is at least 0.9, discard images whose short side is below 600 pixels, remove near-duplicate samples according to perceptual similarity, and re-run OCR on the erased product image to detect residual text artifacts. If any text is still detected on I_{prod}, the sample is removed. For perceptual-similarity filtering, we use pHash to convert each image into a perceptual fingerprint and compute the Hamming distance between fingerprints. Unlike exact duplicate detection, this check focuses on whether two posters look visually similar; pairs with Hamming distance ≤ 10 are treated as overly similar and filtered to improve dataset diversity. The residual-text check is particularly important for generation training because it prevents the model from copying leftover text artifacts from the product image rather than generating text patches from the instruction. In addition, samples with clearly inconsistent structure, such as empty OCR content inside a large detected text box or severe mismatch between the HTML layout and detected patch locations, are excluded during automatic validation.

Table 6: Main filtering criteria used in PosterText-320K construction.

Check	**Threshold**	**Purpose**
OCR confidence	≥ 0.9	remove unreliable text labels
Image resolution	short side ≥ 600	keep readable patch details
Perceptual similarity	pHash Hamming distance > 10	remove overly similar samples
Residual text	none detected	ensure clean product images

C.2 Phase II: Operation Pair Construction

After structure extraction, each poster can be decomposed into operation pairs. For poster generation, we convert the patch annotations into a full-poster instruction conditioned on the product image I_{prod}, without using reference patches. For patch addition, we remove existing text patches and train the model to insert them back according to the instruction; when style control is needed, the removed or matched patch can be provided as a reference patch in $\mathcal{R}$. For patch deletion, the target patch regions are removed according to their structured locations and no reference patch is required. For patch modification, the layout is kept fixed while the text content or visual attributes are changed, and reference patches are optionally used when the instruction asks for style transfer or style-consistent replacement.

Table 7: Operation-pair construction from structured poster annotations.

Operation	Pair construction	Reference patches
Generation	product image → full poster	not used
Addition	remove patch, then insert by instruction	optional/style-guided
Deletion	remove target patch and restore background	not used
Modification	change content or attributes in place	optional/style-guided

All four operations are encoded with unified natural-language templates. Each template specifies the operation type, target text, target region when available, and reference style information when $\mathcal{R}$ is used. We avoid exposing HTML code directly to the final model; instead, the structured fields are verbalized into natural instructions so that the model learns the same type of input expected from users. This keeps the training distribution aligned with the inference interface in Eq. (1) of the main paper.

D PosterText-Bench Evaluation Protocol

PosterText-Bench contains 750 records and is de-duplicated from PosterText-320K to avoid overlap with training samples. The benchmark covers the same four operations as the training formulation: poster generation, patch addition, patch deletion, and patch modification. During construction, we remove exact duplicates and visually near-duplicate posters from the training pool before selecting benchmark samples, so evaluation does not reward memorization of poster templates.

The benchmark evaluates both objective text correctness and perceptual editing quality. This is necessary because e-commerce poster editing requires three properties at the same time: the requested text must be correct, the edited patch must look natural, and the product/background outside the target area should remain unchanged. We therefore report task-specific metric groups rather than forcing every metric onto every task. S.A. and NED are used when an added or modified text string has a known target; M-L. is used for localized editing tasks with a non-edited region; B.P. is used for poster generation, where the main concern is whether the product subject remains faithful after synthesizing a complete poster.

D.1 Objective Metrics

Sentence Accuracy (S.A.) measures the proportion of generated text strings that exactly match the ground-truth sentence after OCR recognition. It is strict and therefore reflects whether the final text is directly usable.

Normalized Edit Distance (NED) is 1 minus the normalized character-level edit distance between the recognized text and the target text. It captures partial text correctness when a prediction is close but not exactly correct.

Masked-LPIPS (M-L.) computes LPIPS only over non-edited regions. For editing tasks, we use the complement of the target mask to measure whether the model changes the background, product, or unrelated patches outside the requested edit.

For S.A. and NED, the generated image is first processed by the same OCR pipeline used for evaluation. We normalize recognized strings by removing obvious OCR spacing artifacts before computing sentence-level exact match and character-level edit distance. For M-L., the edited target region is excluded by the binary operation mask, and LPIPS is computed on the remaining region so that a method is penalized for unintended changes outside the requested patch rather than for successfully changing the target area.

D.2 VLM-based Scores

We use Gemini-3.1 Pro Google (2025) as the VLM judge. Each output is scored on a 1–5 integer scale. The judge receives the input image, instruction, optional reference patches, and model output. For editing tasks, it is also asked to focus on whether the requested local operation is correctly

executed while unrelated regions remain stable. For generation, it focuses on whether the product image is faithfully incorporated into a complete poster and whether the generated poster follows the instruction.

Instruction Following (I.F.) measures whether the requested operation, content, count, and location are correctly executed. A score of 5 indicates that the operation is fully correct, a score of 3 indicates that the main intent is satisfied with minor errors, and a score of 1 indicates that the operation is missing or wrong. Visual Quality (V.Q.) measures typography, style consistency, naturalness, and visual artifacts. A score of 5 corresponds to a natural layout with consistent style, a score of 3 corresponds to a usable result with visible but minor artifacts, and a score of 1 corresponds to distorted, unnatural, or style-conflicting outputs. Background/Product Preservation (B.P.) is used for poster generation to evaluate whether the product subject and major content remain faithful in the generated poster. A score of 5 means the product and background are faithfully preserved, a score of 3 means there are noticeable but limited changes, and a score of 1 means the subject is damaged or heavily altered.

The rubric uses scores 1, 3, and 5 as anchors, with intermediate scores indicating quality between adjacent anchors.

We further audit the automatic judge against human annotations. The rank agreement is strong across the three evaluation dimensions: Spearman correlations are 0.80 for I.F., 0.74 for V.Q., and 0.82 for B.P. These values show that the VLM judge closely tracks human preference trends, while we still treat it as an evaluation proxy rather than a replacement for human assessment.

E Additional Qualitative Results

Figures 5, 6, and 7 provide additional visual comparisons with representative baselines on real e-commerce posters. The examples cover the same four operations evaluated in the main paper: generation, addition, deletion, and modification. Reference patches are shown when the instruction requires reference-guided style control.

F Limitations and Future Work

Generation ceiling. PosterText is optimized primarily for patch-level poster editing after Stage I. Although the model supports end-to-end poster generation, the generation data and the later-stage optimization signals are smaller than those available to closed-source systems. Increasing generation-side data and adding stronger aesthetic alignment may further improve instruction following and visual quality for full-poster synthesis.

Dependence on the base model. Because PosterText is built on Qwen-Image-Edit, the glyph prior, image resolution, and general editing ability of the base model affect the final upper bound. A stronger base model with better native text rendering and higher-resolution generation could further improve all four tasks.

Dependence on automatic evaluation. I.F., V.Q., and B.P. rely on a VLM judge, which can introduce model-specific scoring bias. The human-correlation audit supports the reliability of the current protocol, but a specialized poster-editing reward model or a larger human evaluation would provide a more fine-grained assessment in future work.

Data construction noise. PosterText-320K is built with an automatic pipeline, so OCR errors, imperfect text erasure, and incomplete visual-attribute descriptions can still appear after filtering. These errors are mitigated by confidence filtering and residual-text checks, but they cannot be fully removed without costly manual annotation. Future work can combine automatic construction with targeted human verification for hard cases such as dense small text, overlapping decorations, and highly stylized fonts.

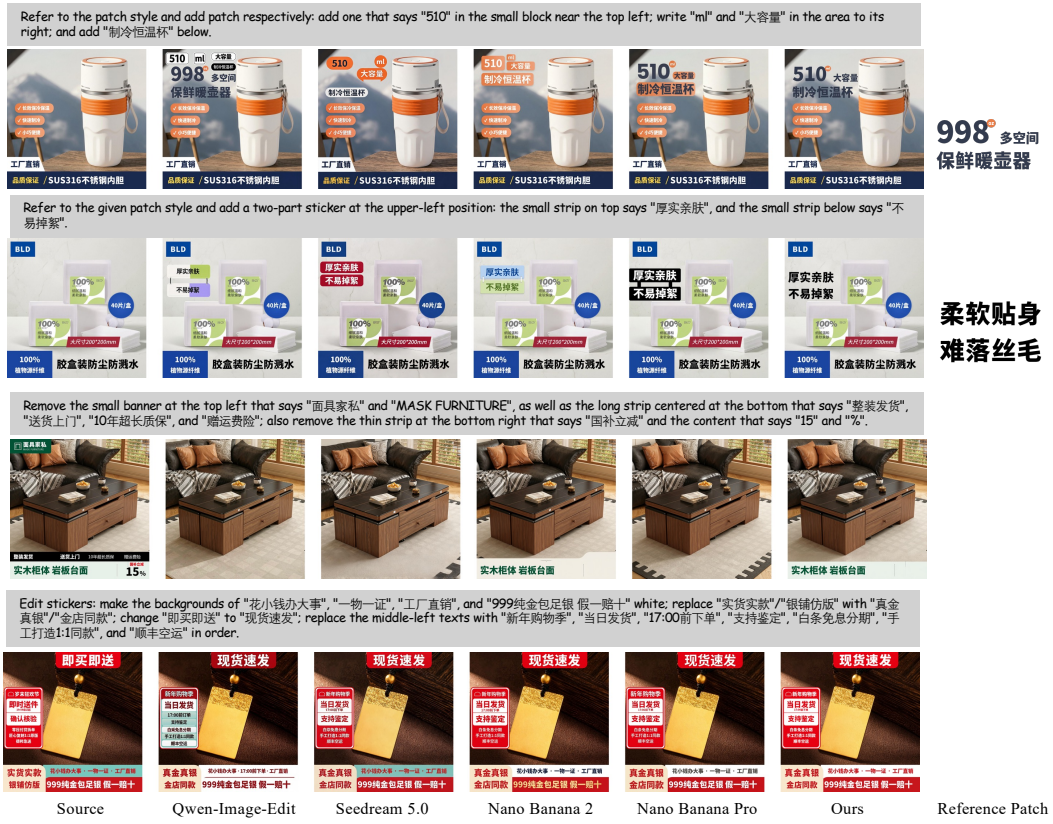


Figure 5: Additional visual comparison of PosterText with existing methods.



Figure 6: Additional visual comparison of PosterText with existing methods.

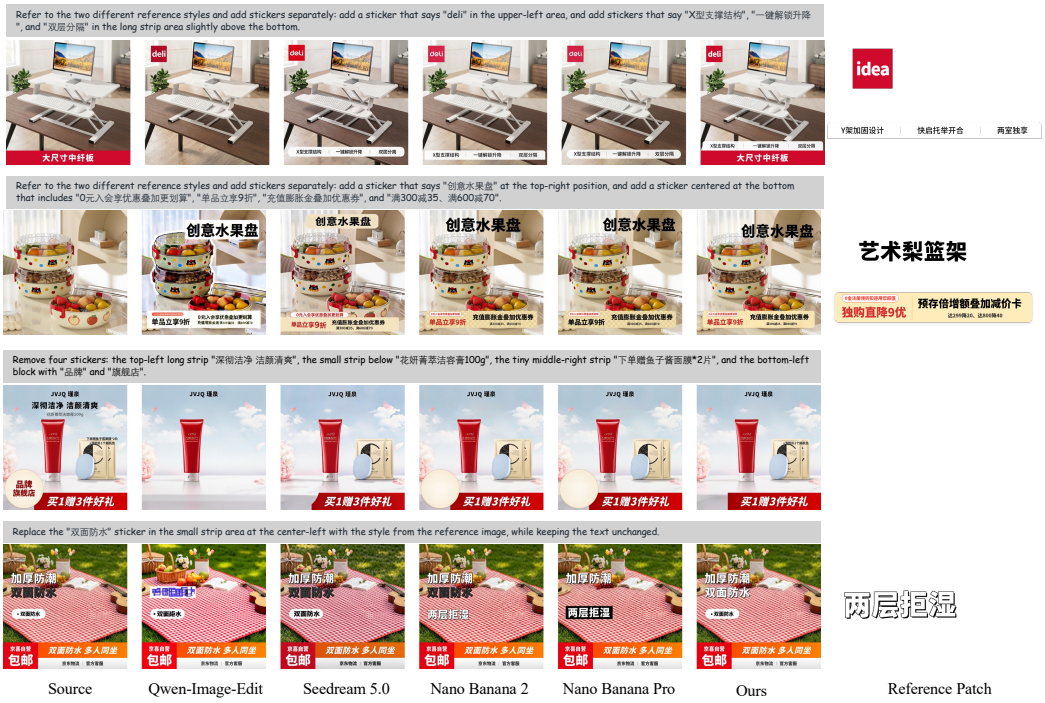


Figure 7: Additional visual comparison of PosterText with existing methods.