CommerceGen-Studio: An End-to-End Framework for Commercial-Grade Main-Image Generation

AI-Native Applications Group
Xiaoli Xu, Zhengtao Wu, Zhengze Xu, Tao Liu, Yongchao Du, Zuan Gao, Xingjian Wang, Hao Yan, Mengting Chen*, Wei Li, Bo Zheng
Alibaba Group
*Project Lead

Taobao App
CommerceGen-Bench (HuggingFace)
CommerceGen-Bench (ModelScope)

arXiv:2610.20603v1 [cs.CV] 14 Oct 2026

Abstract

We present CommerceGen-Studio, an end-to-end framework for generating commercial-grade main images for e-commerce Stock Keeping Units (SKUs). Our framework is organized along a three-stage pipeline of information extraction, semantic rewriting, and multi-condition generation: Prompt-Eval-Refine-Loop Agent (PERL-Agent) distills heterogeneous SKU metadata into concise selling-point text lists through an iterative closed loop of VLM and LLM roles scored by a reference-free rubric. Layout Thinker Optimization (LTO) aligns the rewriter with both caption and rendered-image quality via alternating domain-specific policy optimization. SKU-DiT, trained under Reinforce-Distill-Reinforce (RDR) progressive training, conditions jointly on the white-background product image, the trademark crop, and the structured layout to hold subject fidelity, trademark integrity, and character-level text accuracy together. The framework runs in production at a scale of millions of main images per day with a latency of a few seconds per image, which we believe makes it among the first systems of this kind to operate at commercial scale. We also release CommerceGen-Bench, a benchmark of hard cases collected from live traffic across all product categories. On it, our framework achieves the highest overall scores among the evaluated open-source baselines on both editing tracks, with and without prompt enhancement. It also matches much larger closed-source models, and beats several of them at Instruction Following and Product Integration - the two properties that determine whether an image can be published.

1 Introduction

On e-commerce search result pages (SRPs), shoppers judge relevance almost entirely from two signals: the product image and the title. As search engines increasingly resolve a query to one specific Stock Keeping Unit (SKU) rather than to an item containing multiple SKUs, the SRP has to render display assets at the precise SKU granularity rather than the coarser item granularity. Merchants, however, rarely maintain them. SKU-level imagery is dominated by plain white-background product photographs, and dropping such photographs directly into the SRP measurably lowers click-through rate. Therefore, automated production of high-quality main images at SKU granularity is both a critical need for experience optimization and an inevitable choice for supply gap remediation.

General-purpose text-to-image and image-to-image foundation models have advanced rapidly Hu et al. (2026a;b); Chen et al. (2024c); Wu et al. (2025); Ma et al. (2025); Song et al. (2026); Qiao et al. (2026); Black Forest Labs (2025); Liu et al. (2026b), yet deploying them for SKU product cards exposes four requirements they were not designed to meet. First, selling points must be constructed from heterogeneous sources. SKU information is scattered across titles, OCR results from product images, structured attributes, and backend parameters. These sources may overlap, conflict, or be incomplete, and the relevant information varies by category. They must therefore be reconciled into a concise and reliable set of selling points before generation begins. Second, information consistency is essential: subjects, trademarks, and selling-point text must match the source SKU metadata exactly, since even a hallucinated digit or conflicting attribute may mislead a purchase decision. Third, long-tail robustness is required because live traffic spans diverse product categories and merchant images of highly uneven quality. Fourth, multi-condition layout control must jointly coordinate the product, trademark, and selling-point text within a coherent composition. Existing methods address these requirements only partially: cascaded "plan-then-render" systems accumulate errors across stages, while end-to-end poster models often trade text accuracy against



Figure 1: **Showcase of CommerceGen-Studio**. Given a product main image, an SKU image, and structured product information, the system automatically generates commercial-grade images with subject fidelity, trademark consistency, and precise selling-point text rendering.

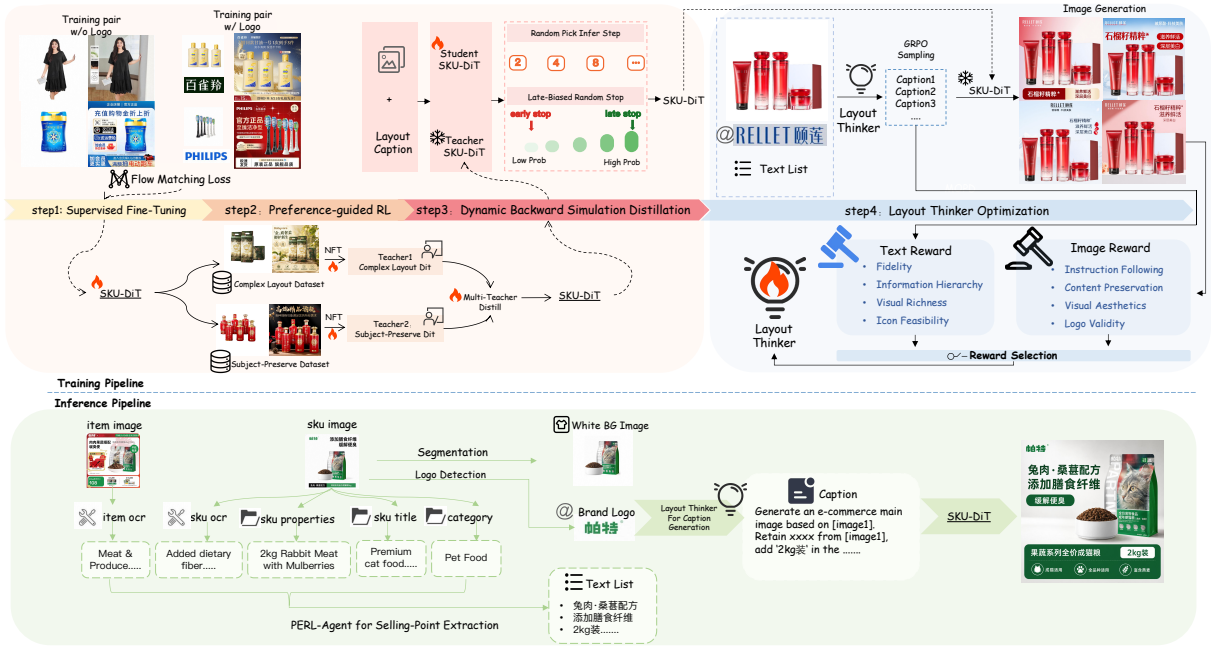


Figure 2: Overall framework of CommerceGen-Studio. The training pipeline combines supervised fine-tuning, RDR progressive training, and LTO. The inference pipeline performs selling-point text list extraction, Layout Thinker rewriting, and multi-condition SKU-DiT generation. Dashed lines connect training-time components to their inference-time roles.

subject fidelity. Figure 1 illustrates our target setting: given a primary product image and an original SKU image, the system must produce a ready-to-use SKU image that satisfies all four requirements.

We address these requirements with a system designed end to end rather than assembled from independently tuned parts. Our contributions are as follows. (1) We present CommerceGen-Studio, a fully automated production framework organized around three stages – information extraction, semantic rewriting, and multi-condition generation – that turns heterogeneous SKU metadata into executable layout semantics and renders it with a single multi-image editing Diffusion Transformer (DiT). (2) We present Prompt-Eval-Refine-Loop Agent (PERL-Agent), which recasts selling-point extraction as an iterative closed loop over four roles (Generator, Judge, Analyzer, and Refiner; one VLM and three LLMs) scored by a six-dimensional reference-free rubric. One category-adaptive prompt serves every category, and attribution over the dominant failure dimension lets the loop roll back to a specific weakness instead of restarting. (3) We present Layout Thinker Optimization (LTO), a VLM layout rewriter trained on two independently sampled reward domains – one scoring captions directly and one scoring rendered images. Rewards are normalized within each domain's rollout groups, the two policy losses are computed separately, and each optimization step samples one reward domain for backpropagation according to the configured domain probability. (4) We present SKU-DiT together with Reinforce-Distill-Reinforce (RDR) progressive training, which sequences preference alignment, Dynamic Backward Simulation Distillation, and a second round of reinforcement on the few-step student, reaching a few seconds latency without conceding the text precision and subject fidelity earned earlier. (5) We release CommerceGen-Bench, a hard-sample benchmark spanning all e-commerce categories, with a five-dimensional rubric scored by a multimodal judge model so that results on it are reproducible across systems.

2 Related Work

2.1 E-commerce Poster Generation

Recent e-commerce poster systems have moved beyond fixed templates toward learned composition and text rendering. PosterMaker Gao et al. (2025) and PosterCraft Chen et al. (2025) generate complete posters with accurate or stylized typography, but PosterCraft starts from a text prompt rather than a product reference, while PosterMaker does not address heterogeneous SKU metadata or separate trademark conditioning. Among 2026 systems, InnoAds-Composer Qin et al. (2026) jointly conditions on a subject image, a glyph image, and a style reference, yet receives the text and approximate placement as conditions instead of extracting selling points or planning a complex layout. AutoPP Fan et al. (2026)

predicts background, selected copy, and layout from one product image and a supplied candidate-copy set, then optimizes the design model with online click feedback; it selects from prepared copy rather than reconciling raw titles, OCR, attributes, and parameters, and it has no separate trademark reference. PosterVerse Liu et al. (2026a) converts a user request into a blueprint and editable HTML poster with scalable typography, but its interface is prompt-driven and does not preserve multiple SKU reference assets. PosterText Liu et al. (2026b) unifies poster creation and editing by treating text patches as atomic units; it accepts one primary product image or poster together with an optional set of reference patches, but does not extract selling points from heterogeneous SKU sources or provide a metadata-driven adaptive layout planner. TransAnyText Liu et al. (2026c) localizes an existing e-commerce image by translating its text and generating an HTML patch whose position, font, color, and size adapt to the target language. Its documented inference input is a single source image, and its objective is localization rather than SKU-level selling-point construction or multi-reference product generation.

Reference-guided editing can transfer appearance from unconstrained visual exemplars Chen et al. (2024c). Recent open-source image editors provide a stronger comparison for subject-preserving generation. LongCat-Image Ma et al. (2025) and the base JoyAI-Image-Edit interface Song et al. (2026) document a single source image, whereas JoyAI-Image-Edit-Plus, FireRed-Image-Edit Qiao et al. (2026), and FLUX.2 [klein] Black Forest Labs (2025) extend editing to multiple references. Qwen-Image Wu et al. (2025) supports precise editing and Chinese text rendering but does not document multi-reference conditioning in its technical report. These general editors nevertheless lack the upstream conversion of heterogeneous SKU metadata into verified selling points and a layout policy trained against the downstream generator. Their remaining domain-specific gaps are also visible on CommerceGen-Bench: FLUX.2 [klein] 4B/9B frequently fail on Chinese selling-point text, FireRed loses logos or fuses multiple product conditions unreliably, and Qwen-Image-Edit shows occasional product-identity drift. We state these as observations from our evaluation rather than as limitations claimed by the respective papers. CommerceGen-Studio addresses the full chain jointly: multi-source selling-point extraction, adaptive layout rewriting, dual-reference product-and-trademark conditioning, and character-level text fidelity.

2.2 Visual Text Rendering

Diffusion models have long suffered from missing characters, incorrect characters, and character hallucinations in the accurate rendering of complex glyphs. AnyText and AnyText2 Tuo et al. (2024b;a) enhance multilingual text rendering precision through glyph-aware conditioning, but treat text as an independent rendering target while neglecting the coordination among text, overall layout, subject, and trademark, and cannot guarantee that the product subject remains unaltered. TextDiffuser Chen et al. (2023) introduces character-level segmentation masks to guide per-character rendering, while its successor TextDiffuser-2 Chen et al. (2024b) relaxes this constraint by adopting line-level text regions and LLM-based layout planning for more flexible composition, but still targets single-image text rendering and lack multi-condition joint modeling with product subject and trademark. FLUX-Text Lan et al. (2025) uses a diffusion Transformer backbone to improve scene text editing but only supports single-image text editing, not multi-condition (white-background image + trademark image + text layout) joint generation. EasyText Lu et al. (2026) improves the controllability of multilingual rendering but is not designed for the information consistency needs of e-commerce scenarios and lacks trademark fidelity and product subject constraints. Our work treats text layout as an explicit condition path in multi-condition generation and jointly models it with subject and trademark within a unified diffusion Transformer, filling the above gaps.

2.3 Reinforcement Learning and Distillation for Diffusion Models

Reconstruction losses defined at the pixel level say nothing about whether text is legible or a composition is attractive, which is why reinforcement learning has moved into the post-training of diffusion models. Diffusion Transformers and flow-matching objectives Peebles & Xie (2023); Esser et al. (2024); Lipman et al. (2023) form the backbone on which current generative systems are built. DDPO Black et al. (2024) applies policy gradients to diffusion post-training by treating denoising as a Markov decision process, where trajectories are sampled on-policy from the current policy and rewards are assigned at the final step. However, it requires per-step policy probability computation and is designed for stochastic diffusion sampling, making it not directly applicable to the deterministic ODE rollout of flow matching. GRPO Shao et al. (2024) replaces the learned value function of PPO Schulman et al. (2017) with in-group relative advantages; it was designed for language models and does not transfer directly to the continuous-time setting of diffusion and flow matching. Flow-GRPO Liu et al. (2025) and DanceGRPO Xue et al. (2025) adapt GRPO to flow-matching models and a unified visual generation framework respectively, but the former requires converting a deterministic ODE into a stochastic SDE to inject exploration and introduces additional approximation errors and is prone to reward hacking without KL regularization, and the latter

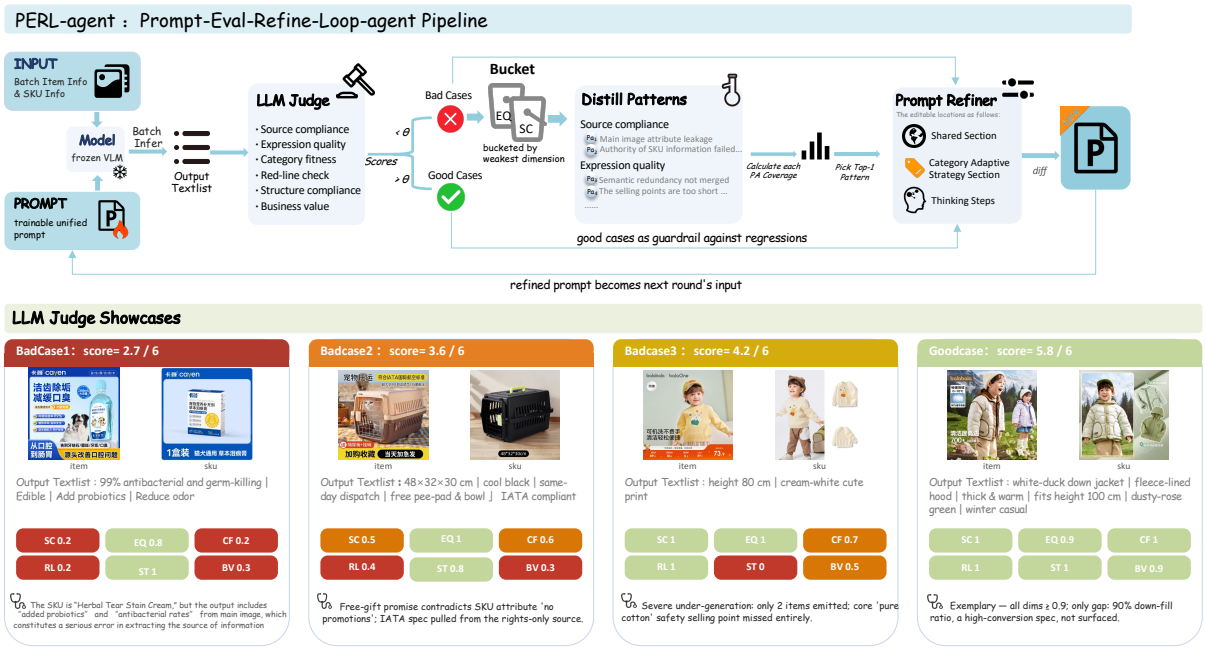


Figure 3: Prompt-Eval-Refine-Loop Agent (PERL-Agent): the prompt-evaluation-refinement pipeline. Generator produces selling-point text list candidates from the current prompt; Judge scores them against a six-dimensional reference-free rubric (SC: Source Compliance · EQ: Expression Quality · CF: Category Fitness · RL: Red-line Check · ST: Structure Compliance · BV: Business Value); Analyzer attributes low-scoring samples by dominant failure dimension to obtain coverage-tagged patterns; Refiner selects the top-1 pattern and applies a minimal single-point diff to produce the next-round prompt.

suffers from task interference and seesaw effects in multi-task mixed training. DiffusionNFT Zheng et al. (2025) proposes an online reinforcement algorithm compatible with flow matching, performing online reward optimization directly through the diffusion forward process without offline trajectory collection and with stable training; therefore, we adopt it as our policy optimization framework. DiffusionOPD Li et al. (2026) provides a diffusion-specific framework for on-policy capability integration: task-specialized teachers are trained independently, while the current student generates its own diffusion trajectories and is distilled toward the corresponding teacher on the states it actually visits. This separates capability-specific exploration from joint integration and reduces interference between heterogeneous objectives.

For inference acceleration, DMD Yin et al. (2024b) pioneers distribution matching distillation by compressing multi-step sampling to a single step, but single-step distillation yields limited generation quality in complex scenarios. DMD2 Yin et al. (2024a) raises few-step quality by refining the distribution matching strategy and adding backward simulation, but does not explicitly protect structural integrity. Based on DMD2, we propose Dynamic Backward Simulation Distillation, which increases the probability of stopping at timesteps near the clean-image endpoint to maintain structural integrity while enhancing generation quality, and introduces discriminator-based adversarial training to strengthen trademark generation, thus balancing quality and latency.

3 Method

3.1 Overview

CommerceGen-Studio comprises a training pipeline and an inference pipeline, both shown in Figure 2. At inference the input is a single product ID. PERL-Agent extracts a structured selling-point text list (TextList) from that SKU's multi-source metadata, and Layout Thinker rewrites the TextList into a layout caption. The text encoder encodes that caption, and SKU-DiT consumes it alongside the white-background product image and the trademark crop to synthesize the final image in one multi-image editing pass.

Training proceeds in three cascaded stages over our in-house Swift-Image 6B backbone Hu et al. (2026b). Stage I moves the general backbone onto the e-commerce SKU distribution through pretraining and continual training. Stage II applies Reinforce-Distill-Reinforce to align the generator with human preference and cut inference cost. Stage III trains the VLM layout rewriter under Layout Thinker Optimization so

that TextList and SKU-DiT meet through an explicit, executable layout rather than an implicit one.

An item is the parent product listing, and its item image is the listing-level main image; an SKU is a concrete purchasable variant under that item, represented by its own SKU image. For selling-point extraction, we collect and filter heterogeneous information from five sources: text recognized from the item image, text recognized from the SKU image, SKU properties (combining attributes and structured backend fields), the SKU title, and the SKU category. The item image often contains shipping offers and other promotional benefits, but its product attributes may conflict with the selected SKU. We therefore retain useful benefit information while filtering conflicting product facts. For example, if the item image says "iPhone 17" but the selected SKU image shows an iPhone 18, preserving "iPhone 17" would inject an incorrect selling point into generation. The other four sources are less likely to conflict with the selected SKU, but they are still heterogeneous and verbose; only information that attracts buyers or expresses the product's core value should be retained. PERL-Agent performs this conflict resolution and salient selling-point extraction. Formally, we denote the five-source metadata by $\mathcal{M} = \{T_{item_ocr}, T_{sku_ocr}, P_{properties}, T_{title}, C_{category}\}$ and map it to a refined selling-point list $L = \pi_{\text{PERL}}(\mathcal{M})$. Through segmentation and detection, we obtain the white-background product image I_{prod} and trademark image I_{logo} respectively; Layout Thinker then rewrites L into $c_{layout} = g_\phi(L, I_{prod}, I_{logo})$. Our objective is to learn a multi-condition diffusion generation model

$$I_{out} = f_\theta\left(I_{prod}, I_{logo}, c_{layout}\right), \tag{1}$$

such that I_{out} is a commercial-grade main image directly deployable to production and satisfies: (a) each selling-point fact is supported by the SKU metadata, and the rendered text is character-for-character consistent with the approved TextList; (b) product subject color, texture, quantity, and viewpoint remain unaltered; (c) the trademark does not deform or drift; (d) the layout is typographically coherent at the standard of professionally designed posters.

3.2 PERL-Agent for TextList Extraction

PERL-Agent produces the selling-point text list that drives everything downstream. The conventional alternatives scale poorly: per-category `CASE WHEN` rules require someone to anticipate every category, and hand-maintained prompts give no way to attribute a regression to a cause or to roll back to a known-good state. We instead treat prompt construction as an optimization problem solved by a closed loop of VLM and LLM roles, scored by a six-dimensional reference-free rubric, with one category-adaptive prompt covering all categories (Figure 3). Appendix A.2 specifies the rubric, PERL-Bench-6.

Four roles divide the work: a Generator (VLM) drafts TextList candidates from the current prompt, a Judge scores them against the rubric without reference answers, an Analyzer attributes low scores to a dominant failure dimension, and a Refiner edits the prompt accordingly. We optimize the worst dimension rather than the mean, since a TextList that is fluent but non-compliant is unusable regardless of how well its other dimensions score:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}_{\text{cat}}}\left[\min_{d\in\mathcal{R}_6} S_d\left(G_\pi(x), x\right)\right], \tag{2}$$

where $G_\pi(x)$ is the TextList generated by Generator using prompt π on input x, $\mathcal{R}_6$ denotes the 6-dimensional rubric, and S_d is the score on the d-th dimension. The prompt is updated by iterative refinement: $\pi^{(t+1)} = \pi^{(t)} \oplus \text{Refine}(\text{Analyze}(\text{Judge}(G_{\pi^{(t)}})))$, where $\oplus$ denotes minimal-diff merging, and Refiner selects the top-1 failure pattern output by Analyzer and applies a single-point modification. Intermediate artifacts are persisted at each round to support comparison and rollback.

3.3 Stage I: Pretraining & Continual Training

The generation backbone adopts our in-house Swift-Image 6B unified multi-image editing Diffusion Transformer Hu et al. (2026b). We then run a further round of continual training on the SKU vertical domain. Starting from high-quality main images contributed by both premium and long-tail merchants, we recover white-background product images by matting and caption the difference between the white-background and original images, which yields triplets of the form (white-background image, prompt, ground truth). Candidates are filtered twice, by an in-house aesthetics model and by high-saturation color detection, then sampled in a balanced manner across leaf category, background scene type, and in-image text length as described in Section 4.2. This stage uses approximately 4M product-domain T2I samples, 2.67M single-image I2I editing pairs, and 274K multi-image I2I editing pairs to move the general backbone onto the e-commerce SKU distribution and to stabilize multi-condition text rendering and subject fidelity.

3.4 Stage II: Progressive Reinforce-Distill-Reinforce Training

Stage I provides SKU-DiT with general multi-condition generation capabilities, but its reconstruction objective does not directly optimize text accuracy, selling-point completeness, subject fidelity, or visual quality. To address these limitations, Reinforce-Distill-Reinforce (RDR) consists of three phases: reinforcement learning aligns the model with human preferences, distillation reduces the number of sampling steps for faster inference, and post-distillation reinforcement recovers the visual details and fidelity lost during distillation. Human preferences rl combines DiffusionNFT-based specialist optimization and DiffusionOPD-based capability integration.

Phase I: Specialist Alignment and Capability Integration. We first define the reward used for DiffusionNFT-based specialist optimization Zheng et al. (2025). The reward is evaluated by a VLM-based judge Chen et al. (2024a) along three primary dimensions. Instruction following (S_1) measures text accuracy, selling-point completeness, trademark restoration, and outpainting compliance. Content fidelity (S_2) evaluates whether the color, texture, quantity, and viewpoint of the white-background product remain unchanged. Aesthetic plausibility (S_3) assesses shadow grounding, lighting consistency, clean edges, and typography quality. To further improve character-level text accuracy, we introduce an additional OCR penalty computed by PP-OCRv6 Zhang et al. (2026). The resulting reward is

$$\mathcal{R} = \sqrt[3]{S_1 \cdot S_2 \cdot S_3} - w \cdot \text{penalty}_{\text{OCR}}, \tag{3}$$

where w controls the OCR penalty strength and $\text{penalty}_{\text{OCR}} = 1$ when an OCR error is detected in non-product regions and 0 otherwise. The geometric mean penalizes imbalanced quality across dimensions, so that a serious failure on one dimension substantially reduces the overall reward even when other dimensions score highly.

Using this reward framework, we independently optimize two specialist teachers with DiffusionNFT. The complex-layout teacher is trained on composition-intensive samples to improve typography, spatial organization, and multi-element coordination, while the subject-preservation teacher is trained on fidelity-critical samples to preserve product identity, structure, color, texture, and trademark details.

After the two teachers converge, we follow the diffusion-specific on-policy distillation framework of DiffusionOPD Li et al. (2026) to integrate their complementary capabilities into a single student SKU-DiT. The student generates its own diffusion trajectories, and each trajectory is supervised by the teacher corresponding to its capability domain. This separates capability-specific exploration from joint capability integration, allowing the student to acquire both complex-layout generation and subject preservation without directly mixing the teachers' parameters. Approximately 5K grouped trajectories are used across specialist optimization and on-policy capability integration.

Phase II: Distillation. After specialist alignment and on-policy capability integration,the student model has acquired both complex-layout generation and subject-preservation capabilities, but still requires expensive multi-step sampling at every step. Dynamic Backward Simulation Distillation folds both costs into few-step inference. We build on DMD2 Yin et al. (2024a) and depart from it in three respects. In the training mixture, general editing data and trademark-domain editing data are combined at a 3:1 ratio; either source alone is worse, as general data underfits domain-specific fonts while trademark-only data yields poorly scaled type. In the sampling schedule, we raise the probability of terminating at timesteps near the clean-image endpoint, which preserves structural integrity that uniform stopping tends to erode, and we add a discriminator trained against high-quality ground-truth images so that trademark strokes are supervised adversarially rather than only in expectation. In the inputs, trademark reference crops are often only tens of pixels across, so we upsample them before conditioning to keep fine stroke detail recoverable. Together with CFG and few-step distillation, single-image latency falls to the 5-second level.

Phase III: Post-Distillation Reinforcement. Distillation buys latency at the cost of a distribution shift, and the damage is not uniform: it concentrates in exactly the high-frequency detail we care most about, blunting font strokes and softening trademark edges. Rather than accept that as the price of speed, we run a second round of reinforcement directly on the few-step student, which re-establishes the subject fidelity and text precision learned during continual training and Phase I, but now under the few-step sampling constraint the student will actually face at inference. This phase uses roughly 2.5K grouped trajectories. The data is drawn from the same pool as continual training with tighter thresholds on aesthetics, composition, and OCR precision – a smaller and cleaner subset that sits closer to the online distribution than the broad training corpus does.

3.5 Stage III: Layout Thinker Optimization

A TextList states what must appear in the image but not where or how, and feeding it to the generator directly forces the generator to invent its own layout, which is what makes composition unstable across samples. Layout Thinker Optimization (LTO) inserts an explicit rewriting layer between the TextList and generator (Figure 2). A VLM takes the TextList from PERL-Agent together with the white-background product image and, when available, the trademark crop, and emits a layout caption that decides subject position and orientation, background style and lighting, text hierarchy and color, and trademark position and size. Training proceeds in two stages, supervised fine-tuning (SFT) followed by reinforcement fine-tuning (RFT). Unlike the reinforcement phases of the generator, which are LoRA-adapted Hu et al. (2022), both rewriter stages fully fine-tune the upper transformer blocks while keeping the lower blocks frozen. The reason is that layout generation is not a matter of adopting a new output format—a change a low-rank adapter can absorb—but of changing how the model reads a product image: it must re-attend to the subject's extent, saliency, and free space to decide where elements go. In our experiments, low-rank updates were too restrictive to capture this representational shift. The frozen lower blocks preserve the general visual-linguistic grounding from pretraining, and the trainable upper blocks specialize to layout reasoning.

Given the ground-truth layout caption c^*, TextList L, the white-background product image I_{prod}, and the trademark image I_{logo} sampled from the dataset $\mathcal{D}_{SFT}$, the SFT stage optimizes the VLM by maximizing the likelihood of the ground-truth layout caption, as follows,

$$\mathcal{L}_{SFT} = -\mathbb{E}_{(L,I,c^*)\in\mathcal{D}_{SFT}}\Big[\frac{1}{T}\sum_{t=1}^{T}\log p_\phi(c_t^* \mid c_{<t}^*, L, I_{prod}, I_{logo})\Big], \tag{4}$$

where ϕ denotes the trainable upper-layer parameters of the VLM, c_t^* is the t-th token of the caption c^* with length T. To mitigate distribution bias, $\mathcal{D}_{SFT}$ is constructed through balanced sampling across different scenes and product categories. At this stage, we primarily emphasize data balancing and prompt engineering, enabling the VLM to acquire the basic mapping capability from structured selling-point information to executable layout descriptions.

The RFT stage aligns the rewriter with both caption quality and rendered-image quality using group-relative policy optimization (GRPO Shao et al. (2024)). The text domain scores sampled captions directly on fidelity, information hierarchy, visual richness, and icon feasibility. The image domain renders a separate caption group with the frozen SKU-DiT and scores the resulting images along instruction following, content preservation, visual aesthetics, and logo validity. The two domains normalize rewards within their own rollout groups and define independent KL-regularized policy losses.

For the text-reward domain, the loss is

$$\mathcal{L}_{\text{text}} = -\mathbb{E}_{\mathcal{G}_{\text{text}}}\left[\frac{1}{N_{\text{text}}}\sum_{i=1}^{N_{\text{text}}}\min\big(r_{\text{text},i}(\phi)\hat{A}_{\text{text},i}, \text{clip}(r_{\text{text},i}(\phi), 1-\epsilon, 1+\epsilon)\,\hat{A}_{\text{text},i}\big)\right] + \beta_{\text{text}} D_{\text{KL}}\big(\pi_\phi \| \pi_{\text{ref}}\big). \tag{5}$$

For the image-reward domain, the loss is

$$\mathcal{L}_{\text{img}} = -\mathbb{E}_{\mathcal{G}_{\text{img}}}\left[\frac{1}{N_{\text{img}}}\sum_{i=1}^{N_{\text{img}}}\min\big(r_{\text{img},i}(\phi)\hat{A}_{\text{img},i}, \text{clip}\big(r_{\text{img},i}(\phi), 1-\epsilon, 1+\epsilon\big)\,\hat{A}_{\text{img},i}\big)\right] + \beta_{\text{img}} D_{\text{KL}}\big(\pi_\phi \| \pi_{\text{ref}}\big). \tag{6}$$

At each optimization step, only one reward domain is activated to avoid gradient interference between the text and image objectives. We sample a scalar $u \sim \mathcal{U}[0,1]$ and select the domain according to the configured text-domain probability p_{text}:

$$\mathcal{L}_{\text{RFT}}(u) = \begin{cases} \mathcal{L}_{\text{text}}, & u < p_{\text{text}}, \\ \mathcal{L}_{\text{img}}, & u \geq p_{\text{text}}. \end{cases} \tag{7}$$

Accordingly, each optimization step backpropagates through either the text-domain loss or the image-domain loss, but never both. For $d \in \{\text{text}, \text{img}\}$, $\hat{A}_{d,i} = \big(R_d(g_{d,i}) - \bar{R}_d\big)/(\sigma_d + \delta)$ is the reward standardized within domain d, while $r_{d,i}(\phi) = \pi_\phi(c_{d,i})/\pi_{\phi_{\text{old}}}(c_{d,i})$ is the corresponding importance-sampling ratio. Here, R_d denotes the domain-specific reward, δ is a numerical-stability constant, ϵ is the PPO clipping range Schulman et al. (2017), β_d controls the domain-specific KL regularization, and π_{ref} is the fixed reference policy. The sampling probability p_{text} controls the relative optimization frequency of the two domains, with $p_{\text{img}} = 1 - p_{\text{text}}$. This stochastic domain selection prevents the two reward signals from producing conflicting gradients within the same update.

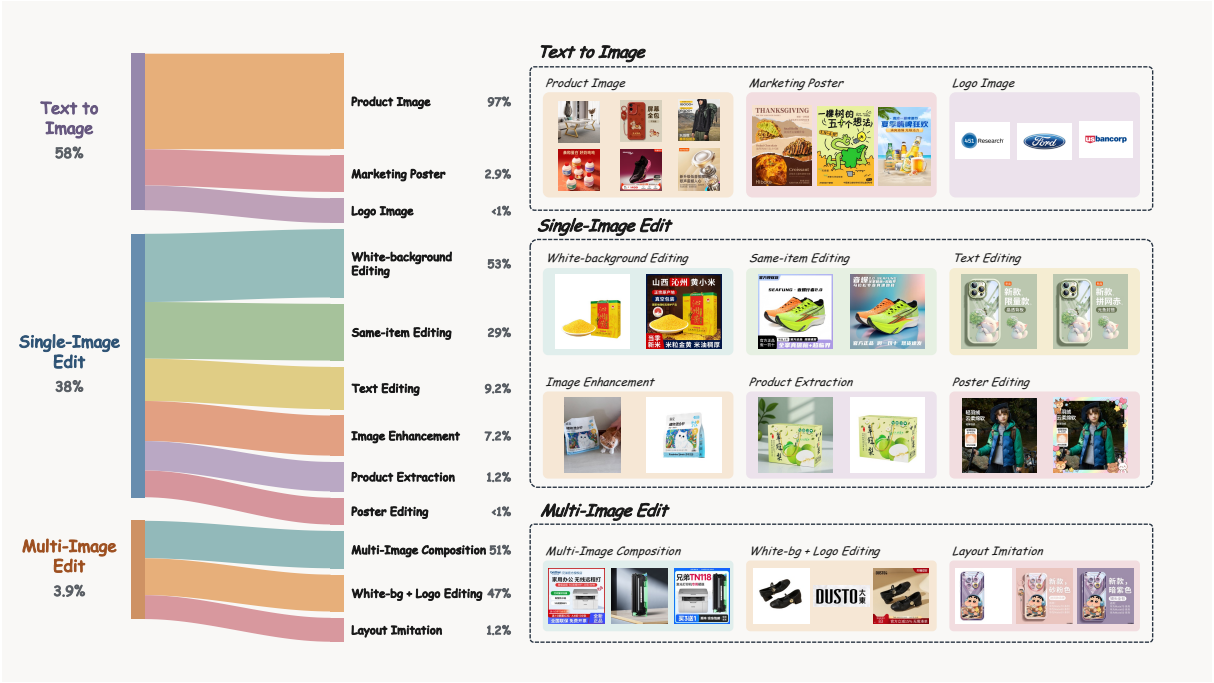


Figure 4: **Task taxonomy of the SKU vertical-domain training corpus.** From left to right: three top-level tasks with their corpus share, their sub-tasks with shares, and representative samples.

Throughout RFT, SKU-DiT remains frozen and serves only as the renderer for the image-reward domain. The caption emitted by the VLM is sampled immediately by SKU-DiT, and the resulting image reward updates only the VLM policy. This closes the semantic gap between the rewriter and the fixed generator without interleaving additional SFT steps or modifying SKU-DiT. The RFT-trained VLM is deployed for prompt rewriting at inference.

4 Dataset and CommerceGen-Bench

4.1 Training Model

Our generation backbone is the in-house Swift-Image 6B unified multi-image editing DiT Hu et al. (2026b). We initialize it from the checkpoint obtained after the 500K-step foundational pretraining and 200K-step continual-training schedule reported in Swift-Image Hu et al. (2026b). The data curriculum follows the capability-centric design of Wang et al. (2026), which progressively increases the training resolution from 256 to 1024 pixels through a five-stage curriculum. We then adapt this checkpoint to the SKU domain using the stage-wise datasets summarized in Table 4.

4.2 Training Data

Figure 4 summarizes the task composition of the SKU-domain training corpus, covering text-to-image generation, single-image editing, and multi-image editing. The editing data include both complex-layout samples, which emphasize the organization of products, selling-point text, trademarks, and decorative elements, and subject-preservation samples, which require the product identity, structure, color, texture, and quantity to remain unchanged during editing.

After task construction, we perform balanced sampling along three independent dimensions. First, samples are balanced across leaf product categories to prevent popular categories from dominating the training distribution. Second, backgrounds are assigned to 50 predefined background classes, and samples are balanced across these classes to improve scene diversity. Third, samples are grouped by the number of selling points in the target image and balanced across groups, ensuring sufficient coverage of both sparse and text-dense layouts.

For data cleaning, we train an in-house aesthetic scorer based on the SwinV2-T architecture Liu et al. (2022) using e-commerce image data. We combine its aesthetic score with a high-saturation color detector to remove low-quality or excessively saturated merchant images. This filtering stage reduces visually

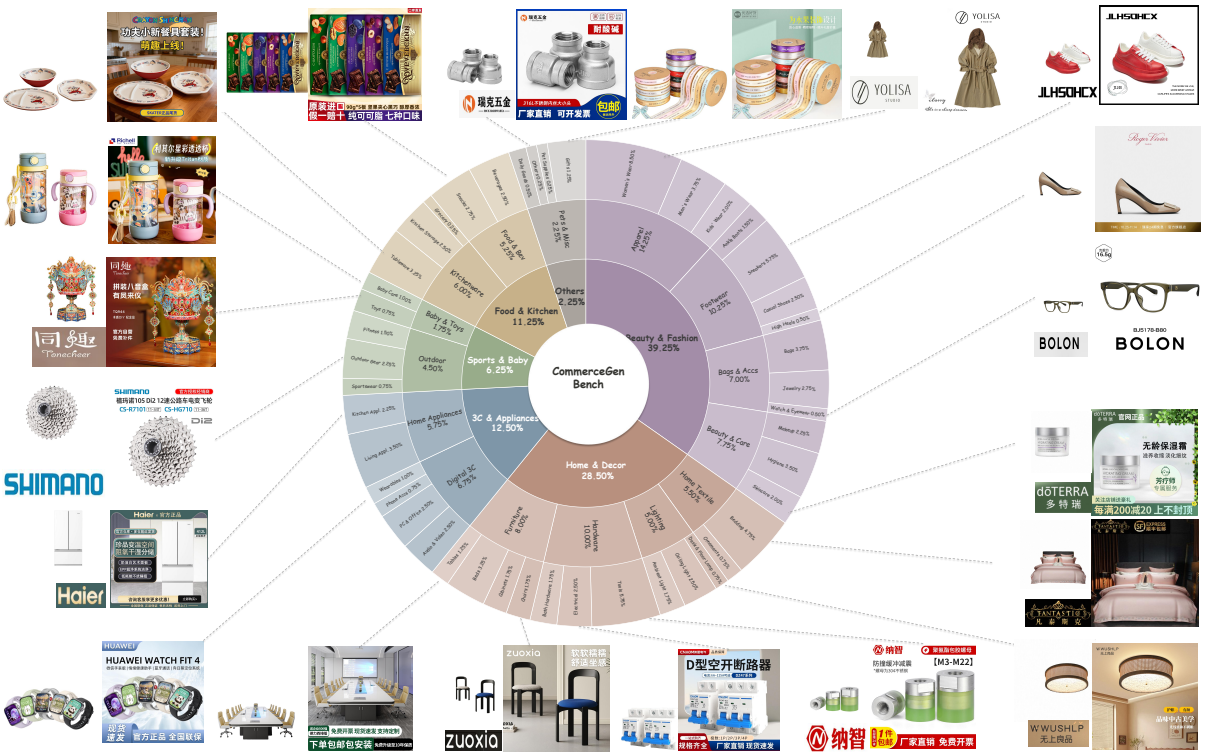


Figure 5: **Distribution of CommerceGen-Bench.** The benchmark comprises 400 common e-commerce scenarios covering 6 top-level categories and 15 sub-categories, with the percentage share of each category visualized.

undesirable supervision before balanced sampling and model training.

PERL-Agent uses two data stages. The cold-start stage contains 100 heterogeneous SKU records spanning the five information sources above. It is used to initialize the extraction prompt, expose cross-source conflicts, and establish rules for retaining important selling points. The same 100 records also calibrate the LLM Judge: after the cold-start extractor produces its first-round TextLists, the Judge and multiple domain experts score each result independently. We compute Spearman rank correlation Spearman (1904) between the sample ordering induced by the mean expert scores and that induced by the Judge scores. Spearman correlation does not require experts to annotate ranks manually; it converts the two sets of scalar scores into ranks and measures whether the Judge orders good and bad TextLists consistently with the experts. A Judge configuration enters the closed loop only when $\rho > 0.7$, which serves as a screening threshold for sufficient ranking agreement with domain experts. After cold start and calibration, the scale-up stage expands directly to approximately 1,000 full-category records for prompt optimization and failure-pattern coverage.

4.3 CommerceGen-Bench

Average-case benchmarks do not adequately reflect model performance in real-world e-commerce scenarios, where difficult and long-tail cases are particularly important. CommerceGen-Bench comprises 400 real online cases collected from live search result pages, evenly split into 200 single-image editing samples and 200 multi-image editing samples. The benchmark spans 6 top-level categories and 15 subcategories: 3C electronics and home appliances, home goods and home improvement, cosmetics and apparel, general merchandise, sports and maternity, and a residual category. A sample qualifies as hard when at least one of the following holds: the subject has thin or transparent structure that matting tends to destroy, the trademark is small enough that its strokes fall near the resolution limit, or the selling-point text is long enough that character-level errors become likely. Figure 5 gives the category distribution.

4.4 Evaluation Protocol

Recent benchmarks emphasize practical and reasoning-intensive single- and multi-image editing scenarios Zhou et al. (2026). Following this evaluation setting, we use Gemini 3 Flash Google (2026) as the multimodal judge model to assess each generated image against its reference product image and

editing instruction. To reduce the influence of input resolution, all images are resized to approximately 1024×1024 before evaluation. Five dimensions are assessed on $[0, 10]$. Product Fidelity (PF) asks whether the identity, structure, color, and key attributes of the core product match the reference. Instruction Following (IF) asks whether the requested edit was actually carried out. Product Integration (PI) asks whether the edited product sits plausibly in its new scene. Naturalness Detection (ND) measures overall realism and the consistency of lighting and materials. Artifacts Detection (AD) identifies blur, distortion, broken edges, and garbled text. Overall is the arithmetic mean of PF, IF, PI, ND, and AD. Appendix A.1 gives the abridged prompt.

5 Experiments

5.1 Training Configuration

For matting of white-background product images, we use Mask2Former Cheng et al. (2022) as the segmentation backbone, fine-tuned on e-commerce product data to achieve robust segmentation for complex backgrounds and multi-subject scenarios.

For trademark detection, we use Ultralytics YOLO26 Jocher et al. (2026), fine-tuned on e-commerce trademark data, to accurately localize and crop trademark regions in main and SKU images. Rendered text is verified throughout with PP-OCRv6 Zhang et al. (2026), which supplies the OCR penalty in the Phase I reward.

PERL-Agent. PERL-Agent instantiates all four roles with Qwen3.8-27B Qwen Team (2026a) using role-specific prompts. The Generator receives both product images and textual SKU metadata and therefore uses the model's multimodal input capability. The Judge uses the same backbone in text-only mode: it receives the generated TextList, and the six-dimensional evaluation rubric, but no image input. The Analyzer and Refiner also operate in text-only mode, attributing low scores to failure patterns and updating the extraction prompt, respectively. No model parameters are updated during the PERL-Agent loop.The refinement loop runs for at most 10 rounds and terminates early when the aggregate rubric score does not improve for three consecutive rounds. In practice, the best score is typically reached within the first three rounds. After termination, we retain the prompt with the highest aggregate rubric score over all completed rounds.

Generator. SKU-DiT is trained as a chain in which each stage initializes from the converged weights of the previous one, and the training method alternates according to what the stage needs. Since transferring a general backbone onto the product distribution is a change of representation that a low-rank update cannot express, continual training on the SKU vertical domain updates all parameters of the Swift-Image 6B backbone Hu et al. (2026b). When in RL Phase I, we train two LoRA-based specialist teachers with DiffusionNFT: a complex-layout teacher and a subject-preservation teacher, both using rank 32 and $\alpha = 64$. Each teacher uses 8 rollout samples per group, 16 inference steps, and a guidance scale of 4.0. The OCR penalty weight in the reward is set to $w = 0.4$ in both reinforcement phases. After the two specialist teachers converge, their capabilities are integrated into a single student SKU-DiT through DiffusionOPD-based on-policy distillation. This student initializes the full-parameter few-step distillation in Phase II, because folding classifier-free guidance and a multi-step sampler into a few-step student alters the sampling behavior of the whole network rather than adjusting a preference. Phase III post-distillation reinforcement again uses LoRA over the distilled student at the same rank 32 and $\alpha = 64$, with 8 rollout samples per group, 8 inference steps to match the student sampler, and a guidance scale of 1.0, guidance having already been distilled into the model by that point.

This alternation between full-parameter updates and low-rank adaptation follows a consistent principle: stages that must acquire or restructure capability update all parameters, whereas the two reward-driven stages are confined to a low-rank subspace. Restricting reinforcement in this way bounds how far reward optimization can pull the model from the grounding established by the preceding supervised or distillation stage, which is the failure mode we most needed to avoid: an unconstrained policy update that chases the aesthetic components of the reward will readily trade away the subject and text fidelity that the earlier stages installed.

Layout rewriter. LTO uses Qwen3.8-27B Qwen Team (2026a) as its backbone. Its reinforcement stage is not LoRA-adapted as the generator's is; instead both LTO stages train the upper transformer blocks in full while the lower blocks stay frozen, so that general visual-linguistic grounding is preserved while the upper blocks specialize to layout planning. Producing a layout requires reasoning jointly over the product image and the TextList to decide placement, hierarchy, and scale, and in our experiments a low-rank update did not give the upper blocks enough capacity to acquire that behavior; the same trainable parameter set is then carried into the RFT stage. In RFT, the two reward domains are activated

Model	Params.	Single-Image Editing						Multi-Image Editing					
		PF	IF	PI	ND	AD	Overall	PF	IF	PI	ND	AD	Overall
Open-source Models													
Qwen-Image-2.1 Qwen Team (2026b)	–	8.32	6.35	8.52	7.21	6.06	7.29	7.65	5.48	8.28	6.66	5.72	6.76
FireRed-Image-Edit-1.0 Qiao et al. (2026)	20B	7.49	3.72	7.84	5.56	4.35	5.79	6.63	1.97	6.81	3.70	2.62	4.35
Qwen-Image-Edit Wu et al. (2025)	20B	6.15	2.80	7.09	4.86	3.63	4.91	6.05	1.85	6.57	3.96	2.92	4.27
FLUX.2 [klein] 9B Black Forest Labs (2025)	9B	8.60	2.02	8.63	5.24	3.30	5.56	7.99	2.38	8.63	5.22	3.39	5.52
FLUX.2 [klein] 4B Black Forest Labs (2025)	4B	8.24	1.15	8.51	4.75	3.09	5.15	6.60	1.75	8.37	4.63	3.16	4.90
JoyAI-Image-Edit-Plus Song et al. (2026)	16B	6.95	3.56	7.59	5.78	4.56	5.69	5.74	2.85	7.72	5.52	4.33	5.23
Closed-source Models													
GPT-Image-2 OpenAI (2026a)	–	9.36	**9.79**	9.60	9.65	9.67	**9.61**	9.39	**9.72**	9.70	9.73	9.87	**9.68**
GPT-Image-2.5 OpenAI (2026b)	–	9.36	9.63	**9.67**	**9.68**	9.71	9.61	9.32	9.59	**9.73**	**9.74**	**9.89**	9.66
Seedream 5.0 Pro ByteDance Seed Team (2026b)	–	**9.44**	9.47	9.50	9.61	**9.82**	9.57	**9.41**	9.49	9.70	9.66	9.72	9.60
Nano Banana Pro Google Cloud (2025)	–	8.70	9.28	9.33	9.35	9.34	9.20	8.61	8.93	9.35	9.37	9.58	9.17
Qwen-Image-2.0-Pro Alibaba Cloud (2026)	–	8.77	8.72	9.21	9.06	8.67	8.89	8.65	8.08	9.41	9.17	9.14	8.89
Seedream 4.5 ByteDance Seed Team (2026a)	–	8.70	7.87	9.33	8.88	8.76	8.71	8.35	7.45	9.19	8.73	8.90	8.53
Ours	6B	9.22	9.54	9.42	9.44	9.37	9.40	9.27	9.29	9.54	9.54	9.57	9.44

Table 1: Quantitative results on CommerceGen-Bench without prompt enhancement (PE). **Bold** and underline denote the best and second-best results across all open-source and closed-source baselines; Ours is reported separately and excluded from this ranking. Overall is the mean of PF, IF, PI, ND, and AD within each editing track. Rankings are determined from unrounded values; all displayed values are rounded to two decimal places.

Model	Params.	Single-Image Editing						Multi-Image Editing					
		PF	IF	PI	ND	AD	Overall	PF	IF	PI	ND	AD	Overall
Open-source Models													
Qwen-Image-2.1 Qwen Team (2026b)	–	8.80	9.02	9.22	9.16	8.56	8.95	8.74	8.71	9.28	9.17	8.60	8.90
FireRed-Image-Edit-1.0 Qiao et al. (2026)	20B	8.99	7.95	9.21	9.25	8.81	8.84	8.66	6.27	9.14	8.67	7.40	8.03
Qwen-Image-Edit Wu et al. (2025)	20B	8.84	7.85	9.05	9.04	8.56	8.67	8.34	5.85	8.60	7.91	6.69	7.48
FLUX.2 [klein] 9B Black Forest Labs (2025)	9B	8.45	3.65	8.70	6.87	4.08	6.35	8.70	4.12	8.90	7.04	4.22	6.60
FLUX.2 [klein] 4B Black Forest Labs (2025)	4B	8.33	3.47	8.75	6.50	3.97	6.20	8.05	3.78	8.83	6.59	3.96	6.24
JoyAI-Image-Edit-Plus Song et al. (2026)	16B	8.65	7.90	9.18	9.09	8.63	8.69	8.40	7.20	9.31	9.17	8.50	8.52
Closed-source Models													
GPT-Image-2 OpenAI (2026a)	–	9.64	9.34	9.69	9.77	9.85	9.66	**9.65**	**9.30**	9.74	9.79	9.92	**9.68**
GPT-Image-2.5 OpenAI (2026b)	–	9.59	**9.52**	**9.76**	**9.80**	**9.92**	**9.72**	9.51	9.27	**9.79**	**9.81**	**9.94**	9.66
Seedream 5.0 Pro ByteDance Seed Team (2026b)	–	**9.82**	9.20	9.76	9.79	9.88	9.69	9.56	9.17	9.71	9.72	9.83	9.60
Nano Banana Pro Google Cloud (2025)	–	9.41	8.69	9.51	9.57	9.75	9.39	9.18	8.64	9.50	9.54	9.79	9.33
Qwen-Image-2-Pro Alibaba Cloud (2026)	–	8.95	8.86	9.21	9.35	9.16	9.11	9.19	8.50	9.61	9.64	9.78	9.34
Seedream 4.5 ByteDance Seed Team (2026a)	–	9.20	8.52	9.45	9.47	9.54	9.24	9.03	8.18	9.48	9.44	9.63	9.15
Ours	6B	9.48	8.83	9.63	9.66	9.55	9.43	9.60	8.72	9.58	9.62	9.70	9.45

Table 2: Quantitative results on CommerceGen-Bench with prompt enhancement (PE). The notation follows Table 1; Ours is reported in green and excluded from baseline ranking.

Layout Model	Generation Model	Score
VLM base	SKU-DiT (50 steps)	6.509
VLM base + SFT	SKU-DiT (50 steps)	7.832
VLM base + SFT	RD SKU-DiT (50 steps)	8.296
VLM base + SFT	RD SKU-DiT (8 steps)	8.253
VLM base + SFT	RDR SKU-DiT (8 steps)	8.271
LTO VLM	**RDR SKU-DiT (8 steps)**	**9.440**

Table 3: Ablation study on CommerceGen-Bench. RDR means Reinforce-Distill-Reinforce, RD means Reinforce-Distill.

Stage	Data Type	Scale
CT: Product T2I	T2I triplets	∼4.07 M
CT: Single-image I2I	Editing pairs	∼2.67 M
CT: Multi-image I2I	Editing pairs	∼274 K
LTO: SFT	Layout captions	∼20 K
LTO: RFT	Prompt groups	∼5 K
RDR I: RL	Group traj.	∼5 K
RDR II: Distill.	Mixed data	∼200 K
RDR III: RL	Group traj.	∼2.5 K
PERL-Agent	Cold start / scale-up	100 / 1K

Table 4: Data scale of each training stage.

uniformly at random, i.e., $p_{\text{text}} = 0.5$, with each optimization step backpropagating through exactly one domain.

5.2 Quantitative Comparison

We conduct a quantitative comparison between CommerceGen-Studio and multiple closed- and open-source models on CommerceGen-Bench; Prompt enhancement(PE) is performed by Qwen3.8-27B Qwen Team (2026a): in the with-PE setting, all systems share the same enhanced input, whereas in the without

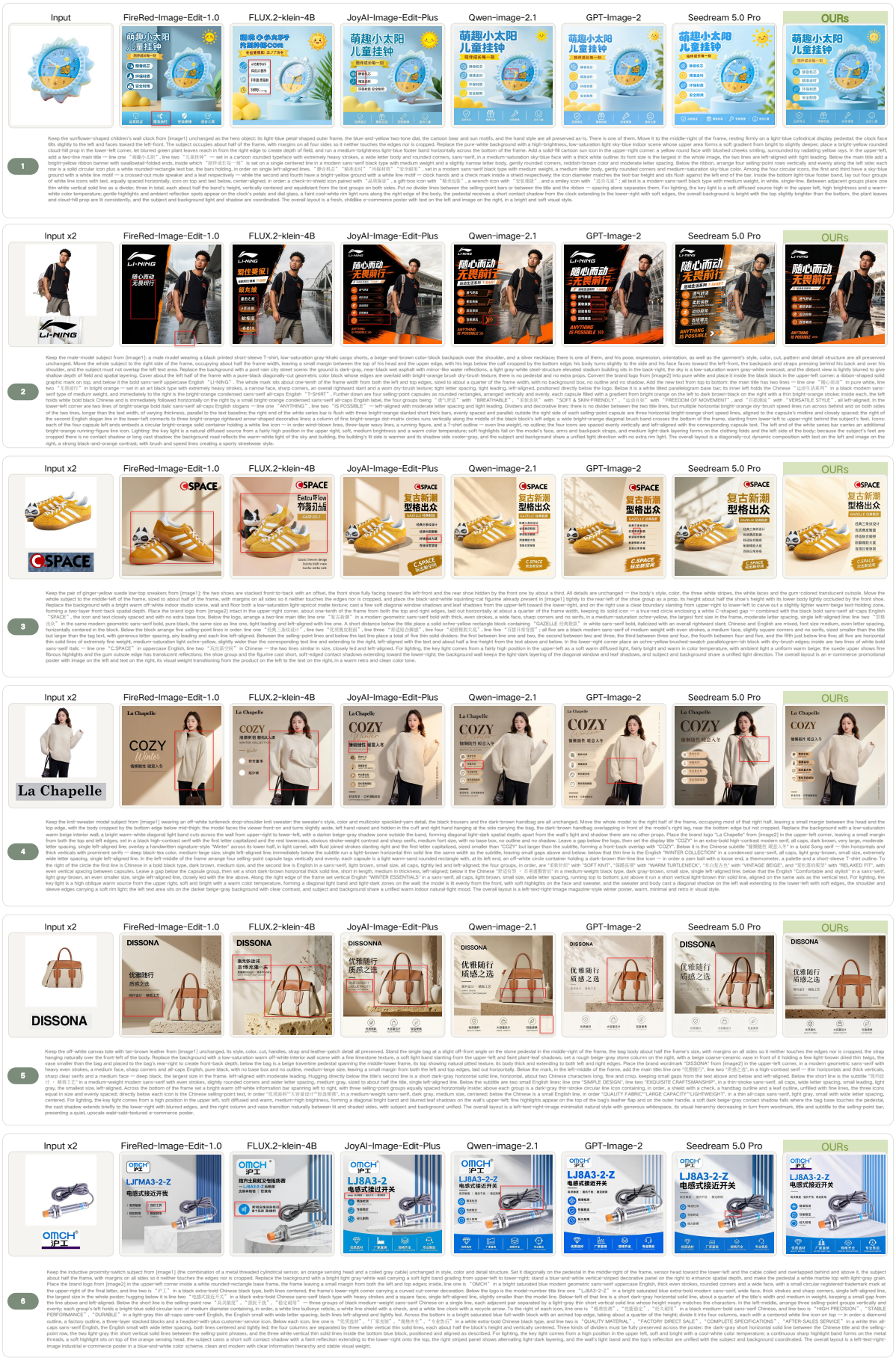


Figure 6: Qualitative comparison of CommerceGen-Studio and several models on single-image editing and multi-image editing.

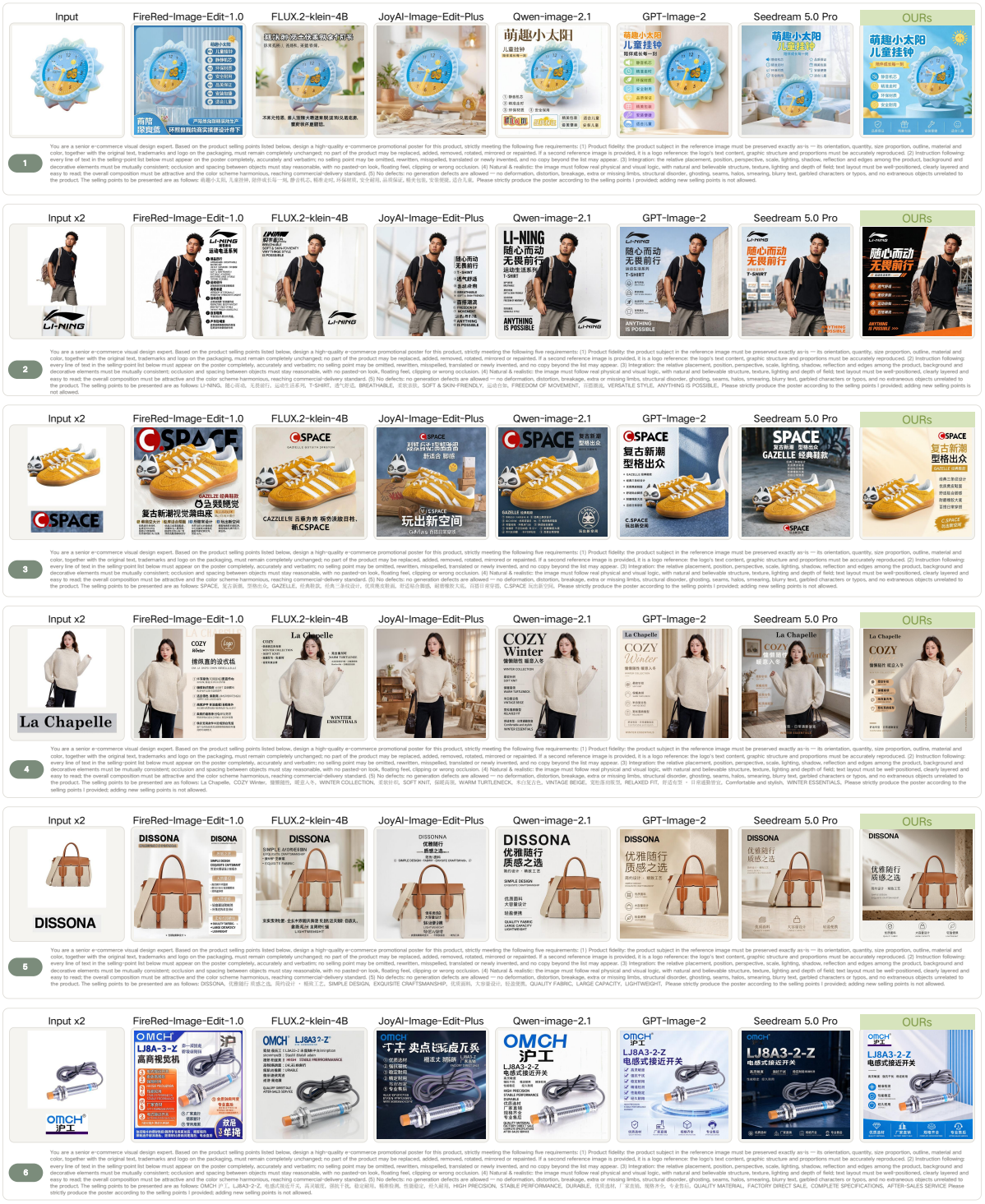


Figure 7: Qualitative results of several models without prompt enhancement (PE).

Results without and with PE are shown in Tables 1 and 2, respectively. the exact without-PE prompt format is shown in Figure 7.

-PE setting we use the original instruction that directly contains the selling-point TextList and basic requirements; Without PE, CommerceGen-Studio achieves Overall scores of 9.40 and 9.44 on the single and multi-image tracks, respectively; with PE, the scores are 9.43 and 9.45. Among open-source baselines, Qwen-Image-2.1 is strongest overall in both settings, reaching 7.29/6.76 without PE and 8.95/8.90 with PE on the single-/multi-image tracks, but it remains below CommerceGen-Studio in product fidelity, integration, and artifact control. GPT-Image-2 and GPT-Image-2.5 provide the strongest closed-source results. Comparing Tables 1 and 2 shows that PE substantially improves instruction following for most

baselines, while domain-specific training remains critical for faithful product-and-trademark composition.

For Ours, the enhanced prompt introduces additional explicit placement, hierarchy, and style requirements, thereby increasing the number of verifiable constraints. Consequently, cross-setting IF changes reflect both generation quality and prompt difficulty and should not be interpreted as a pure capability change.

5.3 Qualitative Examples

Figures 6 and 7 complement the quantitative results with representative single and multi-image editing cases. Figure 6 shows that CommerceGen-Studio preserves product appearance, logos, and selling-point text more consistently than general editors, particularly when product and trademark references must be fused. Figure 7 isolates the compared models without PE: the most common failures are product-identity drift, omitted or corrupted Chinese text, and unstable multi-reference composition. These visual patterns are consistent with Tables 1 and 2, where PE improves instruction following for several baselines but does not eliminate domain-specific fidelity and integration errors.

5.4 Ablation Study

To assess the contribution of each training stage, we start from an open-source VLM baseline and the SKU-DiT base model, and progressively add each training stage; the results are shown in Table 3. The data scale of each SKU vertical-domain training stage is summarized in Table 4. The score is computed on 400 samples from CommerceGen-Bench with PE.

The measured endpoints show a substantial gap between the open-source layout baseline(6.509) and the complete CommerceGen-Studio configuration (9.440). SFT provides the first substantial gain by adapting the layout rewriter to the e-commerce domain. Distillation enables 8-step sampling with a small score decrease from 8.296 to 8.253, after which post-distillation reinforcement partially recovers the loss to 8.271. LTO then provides the largest late-stage gain, raising the score from 8.271 to 9.440.

6 Conclusion and Future Work

We presented CommerceGen-Studio, an end-to-end framework for producing commercial-grade main images at SKU granularity. Its three components address the four requirements we set out with: PERL-Agent extracts selling points from heterogeneous metadata under a reference-free rubric, Layout Thinker Optimization turns those selling points into layouts the generator can execute, and SKU-DiT trained under Reinforce-Distill-Reinforce renders them with a latency of a few seconds without surrendering fidelity. On CommerceGen-Bench, our hard-sample benchmark, the framework achieves the highest Overall scores among the evaluated open-source baselines on both editing tracks, with and without PE, and remains competitive with leading closed-source systems, outperforming several of them on selected dimensions. Two limitations point to future work. First, our reward signals are defined by rubrics and therefore capture what we can articulate; incorporating real user behavior would let the model learn aesthetic preferences that resist explicit specification. Second, the system currently optimizes for a single notion of quality per SKU, and conditioning on audience segment and style would let one product yield genuinely different assets rather than one canonical treatment.

A Appendix

A.1 Evaluation Prompt

Figure 8 reproduces the rubric given to the judge model, abridged here for space. Each dimension additionally carries per-band anchor descriptions at the 10, 9, 7–8, 5–6, 3–4, 1–2, and 0 levels, written in the same conservative style as the calibration rules below; we omit those anchor tables and retain the instructions that govern scoring behavior.

Evaluation Prompt (Forensic Rubric Scoring)

`You are a conservative, forensic evaluator of commercial product image editing. Compare every visible detail in the generated image against the reference image(s) and the editing instruction. Return ONLY one JSON object.`

CALIBRATION RULES

- Score five dimensions independently from 0 to 10; decimals are allowed and higher is better.
- Start each dimension at 5.0, then move it only when visible evidence satisfies an anchor. Do not start from 10 and subtract.
- 5 means mixed/borderline, not catastrophic. 7 means good but clearly flawed. 8 means strong with limited defects. 9 means excellent with only negligible defects. 10 means essentially flawless after close inspection and must be rare.
- Never award 9 or 10 from a favorable overall impression alone. A score above 8 requires naming why there is no material defect in that dimension.
- When a relevant small detail or text is visibly unreadable, do not assume it is correct. Do not reward beauty, resolution, or style in dimensions where it does not prove correctness.
- Use the lower end of a band when several defects exist and the upper end when only one limited defect exists. When uncertain between two scores, choose the lower score.

DIMENSIONS

- **PF – Product Fidelity:** identity, silhouette, geometry, proportions, materials, colors, fine details, original packaging text, trademarks, and product logos. Layout or camera changes alone are not fidelity errors.
- **IF – Instruction Following:** completion and correctness of every explicit requested edit, object, placement, count, relation, style, and newly added or modified text.
- **PI – Product Integration:** how coherently the preserved product and optional logo belong in the edited composition, including placement, scale, perspective, lighting, shadows, reflections, edges, occlusion, and scene interaction.
- **ND – Naturalness Detection:** perceptual realism and visual plausibility appropriate to the requested photo, advertising, illustration, or graphic-design style. Judge anatomy, physics, texture, lighting, depth, typography placement, and composition, not mere attractiveness.
- **AD – Artifacts Detection:** absence of malformed geometry, duplicated or missing parts, garbled pixels or text, halos, seams, smearing, ghosting, unintended objects, broken edges, and generation residue. A higher score means fewer artifacts.

Overall Score: the arithmetic mean of the five dimension scores, $(\mathrm{PF} + \mathrm{IF} + \mathrm{PI} + \mathrm{ND} + \mathrm{AD})/5$.

Figure 8: Forensic evaluation prompt for CommerceGen-Bench (abridged): a conservative five-dimensional rubric (PF, IF, PI, ND, AD) scored 0–10, where the overall score is the arithmetic mean of the five dimensions. Per-band anchor descriptions are omitted for space.

A.2 PERL-Bench-6

The six-dimensional ground-truth-free rubric used to evaluate PERL-Agent's TextList quality is defined as follows: Source Compliance (SC) measures the consistency between text information and SKU metadata; Expression Quality (EQ) measures the fluency and professionalism of language expression; Category Fitness (CF) measures how well the content matches category-specific characteristics; Red-line Check (RL) detects whether the content violates rules or contains sensitive material; Structure Compliance (ST) measures the structural regularity of the text list; Business Value (BV) measures the commercial conversion potential of selling-point information. Each dimension is scored from 0 to 1, and a rule module applies reverse penalty scoring for specific patterns. By default, an aggregate score ≤ 5 out of 6 is classified as a bad case, triggering Analyzer attribution and Refiner rollback.

References

Alibaba Cloud. Qwen-image-2.0-pro-2026-06-22. Official model release announcement, 2026. URL https://www.aliyun.com/product/news/29626.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. arXiv preprint arXiv:2305.13301, 2024.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. Official model release and documentation, 2025.

ByteDance Seed Team. Seedream 4.5. Official model documentation, 2026a. URL https://seed.bytedance.com/en/seedream4_5.

ByteDance Seed Team. Seedream 5.0 pro. Official model documentation, 2026b. URL https://seed.bytedance.com/en/seedream5_0_pro.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. In Proceedings of the 41st International Conference on Machine Learning, 2024a.

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.

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser-2: Unleashing the power of language models for text rendering. arXiv preprint arXiv:2311.16465, 2024b.

Shuhao Chen, Jia Lai, Jia Gao, Haonan Shi, Zhi Liu, Tian Ye, Jie Luo, Xing Wei, and Lei Zhu. Postercraft: Rethinking high-quality aesthetic poster generation in a unified framework. arXiv preprint arXiv:2506.10741, 2025.

Xi Chen, Yutong Feng, Mengting Chen, Yiyang Wang, Shilong Zhang, Yu Liu, Yujun Shen, and Hengshuang Zhao. Zero-shot image editing with reference imitation. In Advances in Neural Information Processing Systems, volume 37, pp. 84010–84032, 2024c.

Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1290–1299, 2022.

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.

Jia Fan, Yuxuan Qin, Wei Feng, Yikai Chen, Yifan Li, An Ma, Yixiang Li, Ling Zhuang, Hui Bian, Zhi 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.

Yichen Gao, Zefeng Lin, Chang Liu, Ming Zhou, Tao Ge, Biao Zheng, and Hong Xie. Postermaker: Towards high-quality product poster generation with accurate text rendering. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8083–8093, 2025.

Google. Gemini 3 flash. Official model documentation, 2026. URL https://ai.google.dev/gemini-api/docs/models/gemini-3-flash-preview.

Google Cloud. Gemini 3 pro image (nano banana pro). Official model documentation, 2025. URL https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/gemini/3-pro-image.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In International Conference on Learning Representations, 2022.

Taihang Hu, Mengting Chen, Jinsong Lan, Xiaoyong Zhu, Kaifu Zhang, Ming-Ming Cheng, Bo Zheng, and Yaxing Wang. ORION: Decoupling and alignment for unified autoregressive understanding and generation. In International Conference on Learning Representations, 2026a.

Taihang Hu, Zhao Wang, Zuan Gao, Tao Liu, Hao Yan, Zhengze Xu, Yuhang Yu, Yongchao Du, Xingjian Wang, Jun Zheng, Qinye Zhou, Zhengrui Chen, Chao Lin, Yefeng Shen, Zhengtao Wu, Xiaoli Xu, Denghui Yang, Huayu Zhang, Mingzhou Zhang, and Mengting Chen. Swift-image: Exploring the performance frontier of compact unified image generation models. arXiv preprint arXiv:2608.20334, 2026b.

Glenn Jocher, Jing Qiu, Mengyu Liu, Shuai Lyu, Fatih Cagatay Akyon, and Muhammet Esat Kalfaoglu. Ultralytics yolo26: Unified real-time end-to-end vision models. arXiv preprint arXiv:2606.03748, 2026.

Rui Lan, Yifan Bai, Xiang Duan, Ming Li, Di Jin, Rui Xu, Dong Nie, Li Sun, and Xiaojie Chu. Flux-text: A simple and advanced diffusion transformer baseline for scene text editing. arXiv preprint arXiv:2505.03329, 2025.

Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. Diffusionopd: A unified perspective of on-policy distillation in diffusion models. arXiv preprint arXiv:2605.15055, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In International Conference on Learning Representations, 2023.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. arXiv preprint arXiv:2505.05470, 2025.

Jie Liu, Pei Zhang, Yixiao Zhang, Pan Yan, Hao Zhou, Xin Zhou, Feng 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, 2026a.

Xiaoan Liu, Lichen Ma, Zipeng Guo, Yu He, Xiaoyan Su, Shaojie Guo, Jingling Fu, Xiaolong Fu, Hao Yang, Tongxuan Liu, et al. Postertext: Towards unified visual text generation and editing for e-commerce poster. arXiv preprint arXiv:2608.16289, 2026b.

Xiaoan Liu, Lichen Ma, Zipeng Guo, Yu He, Xiaoyan Su, Shaojie Guo, Hao Yang, Jingling Fu, Xiaolong Fu, Zhen Chen, Yu Guo, Fei Wang, Xinyi Liu, Yongjun Zhang, Ke Zhang, and Junshi Huang. Transanytext: Translating arbitrary text in e-commerce images via structured visual generation. arXiv preprint arXiv:2608.16284, 2026c.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12009–12019, 2022.

Run Lu, Yifan Zhang, Jia Liu, Hao Wang, and Yizhi Song. Easytext: Controllable diffusion transformer for multilingual text rendering. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 40, pp. 7565–7573, 2026.

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. arXiv preprint arXiv:2512.07584, 2025.

OpenAI. Gpt-image-2. Official model documentation, 2026a. URL https://platform.openai.com/docs/models/gpt-image-2.

OpenAI. Gpt-image-2.5. Official model documentation, 2026b.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4195–4205, 2023.

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. arXiv preprint arXiv:2602.13344, 2026.

Yuxuan Qin, Kai Cao, Hui Liu, An Ma, Fan Li, Hui Zhu, Zhi Zhang, Rui Ling, Wei Feng, Xiang 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.

Qwen Team. Qwen3.8-27b. Official model repository, 2026a. URL https://github.com/QwenLM/Qwen3.8.

Qwen Team. Qwen-image-2.1. Official model repository, 2026b. URL https://github.com/QwenLM/Qwen-Image-2.1.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024.

Lin Song, Wenbo Li, Guoqing Ma, Wei Tang, Bo Wang, Yuan Zhang, Yijun Yang, Yicheng Xiao, Jianhui Liu, Yanbing Zhang, et al. Joyai-image: Awaking spatial intelligence in unified multimodal understanding and generation. arXiv preprint arXiv:2605.04128, 2026.

C. Spearman. The proof and measurement of association between two things. The American Journal of Psychology, 15(1):72–101, 1904.

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, 2024b.

Xingjian Wang, Zhao Wang, Taihang Hu, Jun Zheng, Qing Jin, Qinye Zhou, Zhengtao Wu, Yongchao Du, Zuan Gao, Chao Lin, Yefeng Shen, Xiaoli Xu, Zhengze Xu, Hao Yan, Yuhang Yu, Mingzhou Zhang, and Mengting Chen. From corpora to co-evolving capabilities: Capability-centric data design for generalist image generation. arXiv preprint arXiv:2608.18076, 2026.

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.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. Dancegrpo: Unleashing grpo on visual generation. arXiv preprint arXiv:2505.07818, 2025.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. arXiv preprint arXiv:2405.14867, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In International Conference on Learning Representations, 2024b.

Yubo Zhang, Xueqing Wang, Manhui Lin, Yue Zhang, Penglongyi Deng, Ting Sun, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Changda Zhou, Hongen Liu, Suyin Liang, Cheng Cui, Yi Liu, Dianhai Yu, and Yanjun Ma. Pp-ocrv6: From 1.5m to 34.5m parameters, surpassing billion-scale vlms on ocr tasks. arXiv preprint arXiv:2606.13108, 2026.

Kaiwen Zheng, Huayu Chen, Hao Ye, Hang 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.

Qinye Zhou, Jun Zheng, Yongchao Du, Yuan Wang, Zhengrui Chen, Zuan Gao, Taihang Hu, Chao Lin, Yefeng Shen, Xingjian Wang, Zhao Wang, Zhengtao Wu, Xiaoli Xu, Zhengze Xu, Hao Yan, Denghui Yang, Yuhang Yu, Huayu Zhang, Mingzhou Zhang, and Mengting Chen. CPI-Bench: A comprehensive, practical and intelligent benchmark for real-world image editing. arXiv preprint arXiv:2608.14546, 2026.