LEARNING TO DRAW INTERFACES: WHAT MAKES SYNTHETIC UI DATA WORK FOR TEXT-TO-IMAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating interface screenshots requires text-to-image models to reproduce both the specified text and the arrangement of components. In this paper, we propose SCREENFORGE, a data synthesis system for studying how training data shapes these capabilities. The system renders application templates and retains source documents and component trees, allowing different captions to be derived for the same image. We use SCREENFORGE to compare caption recipes, application coverage, and UI-to-background mixtures in single-run continued-training experiments, each starting from the same 2B checkpoint and using 10 million examples. These comparisons show that caption choice depends on the inference prompt format and that broader application coverage improves most metrics at a fixed UI sample count. Increasing the UI share does not improve all metrics: moderate shares favor text and layout adherence, while larger shares yield modestly higher visual quality. On a benchmark of 1,300 structured screen specifications, a 50% UI mixture raises required-text recall from 0.465 to 0.813 and VLM-rated layout adherence from 3.24 to 3.84 compared with background-only continued training. We will release the rendered training images and captions, data manifests, and all 1,300 benchmark reference screens, together with the evaluation code, to support further research on UI generation.

1 INTRODUCTION

Generating an application screen requires a text-to-image (T2I) model to place components in the intended regions and render their text correctly. The open-weight baselines evaluated here often merge components or misrender labels (§6.2). Model rankings alone do not isolate the effects of training data. In this paper, we study which choices in constructing synthetic UI data improve text and layout fidelity under a fixed model and example budget.

Answering this question requires control over both the training images and the information in their captions. Re-captioning images with a vision-language model can provide richer descriptions (Betker et al., 2023), but component attributes absent from the stored annotations must be inferred from pixels. Capturing live applications, as in RICO (Deka et al., 2017), provides view hierarchies for recorded screens. Rendering parameterized templates gives us access to the source needed to vary screen content and produce new images.

We introduce SCREENFORGE, a synthesis system that renders application templates and stores each image with its source document and measured component tree. These records let us derive prose, flat JSON, and hierarchical JSON captions for the same images (Figure 1). We can therefore compare caption recipes without changing the underlying screens, while selecting from the rendered pool lets us vary application coverage and the amount of UI data used in training.

Our study varies caption recipes, application coverage, and the proportion of UI examples in the training mixture (Figure 2). All controlled runs start from the same 2B reference checkpoint pre-trained on BLIP3-o (Chen et al., 2025) and process 10M examples; background-only continued training provides the control for adding UI data. We evaluate the resulting models on structured specifications of 1,000 real screens and 300 synthetic screens from held-out applications. Most real

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.

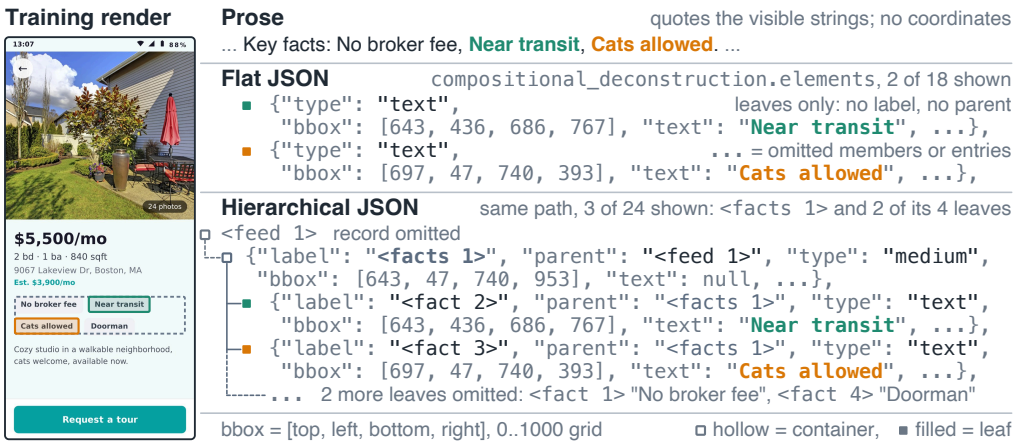


Figure 1: A training render and excerpts of three captions derived in stage 5 (§4). The example follows one container and two leaves. Prose quotes visible strings; flat JSON records leaf types, boxes, and text; hierarchical JSON also records the container and parent links. Matching colors connect screen regions to caption entries.

screens come from applications used to build the templates, so this benchmark tests both transfer to real screens and generation for held-out applications (§4.3).

The comparisons show that the preferred caption recipe depends on the inference prompt format: hierarchical-caption training gives stronger prompt and layout adherence under hierarchical prompts on the primary benchmark, while mixed-caption training is more robust across the tested formats (Appendix G.1, G.2). At a fixed UI sample count, broader application coverage improves most benchmark metrics (§6.3). The preferred UI proportion also depends on the metric: moderate shares favor text and layout adherence, while larger shares yield modestly higher visual quality (Appendix G.3). These findings describe single training runs on the tested model and benchmark.

Adding synthetic UI data improves both text rendering and layout adherence over background-only continued training. Under the same example budget, a 50% UI mixture raises required-text recall from 0.465 to 0.813 and VLM-rated layout adherence from 3.24 to 3.84 (§6.3).

Our contributions are:

- **A synthetic UI data system** that retains rendering sources and component structure, enabling alternative captions to be derived for the same images.
- **Empirical findings on UI training data**: caption choice depends on the inference format, broader application coverage benefits most metrics, and the preferred UI share depends on the evaluation metric.
- **A UI generation evaluation suite** for text, layout, and visual quality. We will release the evaluation code and all 1,300 benchmark reference screens alongside training data and captioning code.

2 Related Work

Re-captioning images improves T2I supervision (Betker et al., 2023; Esser et al., 2024). RICO pairs captured screens with view hierarchies (Deka et al., 2017), while WebSight pairs synthetic webpages with HTML source for screenshot-to-code learning (Laurençon et al., 2024). ScreenForge retains rendering sources and measured component trees to derive alternative captions for the same images.

UI generation includes element-layout modeling (Li et al., 2019; Gupta et al., 2021; Inoue et al., 2023) and layout-conditioned image synthesis (Li et al., 2023; Zheng et al., 2023). UI-Diffuser first generates a layout from component specifications, then synthesizes a mobile screen (Wei et al., 2023). UISim predicts the next screen from a current screenshot and a user action (Xiang et al., 2025). We study synthetic training data for generating screens from structured specifications.

DataComp and DataComp-LM compare data choices with fixed models and training procedures (Gadre et al., 2023; Li et al., 2024). We adopt this approach for T2I with a shared model and example budget. Our UI benchmark measures required text, layout, and visual quality (§3), complementing general T2I evaluations such as GenEval, T2I-CompBench, and DPG-Bench (Ghosh et al., 2023; Huang et al., 2023; Hu et al., 2024). Appendix A gives a fuller comparison.

3 TASK AND STUDY DESIGN

For the primary benchmark, a T2I model generates a static screenshot from a hierarchical JSON specification of components, visible text, and normalized regions. The reference screenshot is used for evaluation, not supplied to the generator. We do not test design from a short brief or interactive behavior. Additional experiments evaluate caption-training recipes under different inference prompt formats (§6.3).

We compare sixteen existing T2I systems on the 1,300-prompt benchmark. Separately, we compare synthetic-data recipes by continued training from the same reference checkpoint, fixing the model, example budget (one pass over 10M examples), and evaluation protocol. This follows DataComp's fixed-model, varied-data design (Gadre et al., 2023).

A recipe specifies which applications to include, how to derive captions from the stored source and component tree, and the UI share of the training mixture. Caption schemas change content and serialization together. Evaluation combines reference-based image and text metrics with a fixed VLM judge (§5.2). The layout evaluation uses GT-defined regions without detecting and matching UI components in the generated screens.

4 THE SCREENFORGE PIPELINE AND UI POOL

4.1 GENERATION PIPELINE

The pipeline retains the source document and measured component tree for each render, so captions can be derived without changing the image. Its six stages are shown in Figure 2 and detailed in Appendix B.

(1) Layout capture. A scripted capture extracts layout patterns, view hierarchies, and iconography from real applications. **(2) Parameterized templates.** An LLM coding agent builds templates from each application's captures, covering the observed screen types (100 training applications and ten held out). **(3) Content filling.** An LLM writes a deduplicated copy bank for each application. Numeric fields are sampled programmatically; imagery comes from an index of more than 500M images. Each render combines copy and imagery afresh. **(4) Rendering.** A headless browser renders each screen. We store its source document and measured component tree, including component types, layout boxes, visibility, and visible text. **(5) Caption derivation.** Captions are derived deterministically from the stored document and tree: a prose summary, a flat JSON list of visible components, and hierarchical JSON preserving the tree (Figure 1). The mixed arm of §6.3 uses a fixed, balanced rotation to pair each image with one schema. New schemas supported by the stored fields require rerunning this stage. **(6) Quality control.** Each render undergoes deterministic checks for an empty or underfilled main area or a placeholder image. Passing renders form the pool. A VLM distinct from the evaluation judge of §5.2 reviews random samples of about 60 renders per application. Given a sampled render and its hierarchical caption, the reviewer scores faithfulness to the described elements, coverage of regions, text legibility, and rendering realism on a 1–5 scale. A render passes only when all four scores are at least 4; ambiguous failures are re-judged by a stronger judge. Templates or content are revised and fresh samples reviewed until at least 95% pass. The authors inspect the final round. The 97% in Table 1 is the mean final-round pass rate across applications, measured on these samples rather than the full pool.

4.2 SAMPLE RECORDS

Each synthetic sample stores the image, the stage 4 component tree as structured ground truth, the source document, all three captions, and provenance: application, screen type, locale, theme, and viewport. The source records the template, parameters, copy, numeric values, and image choices.

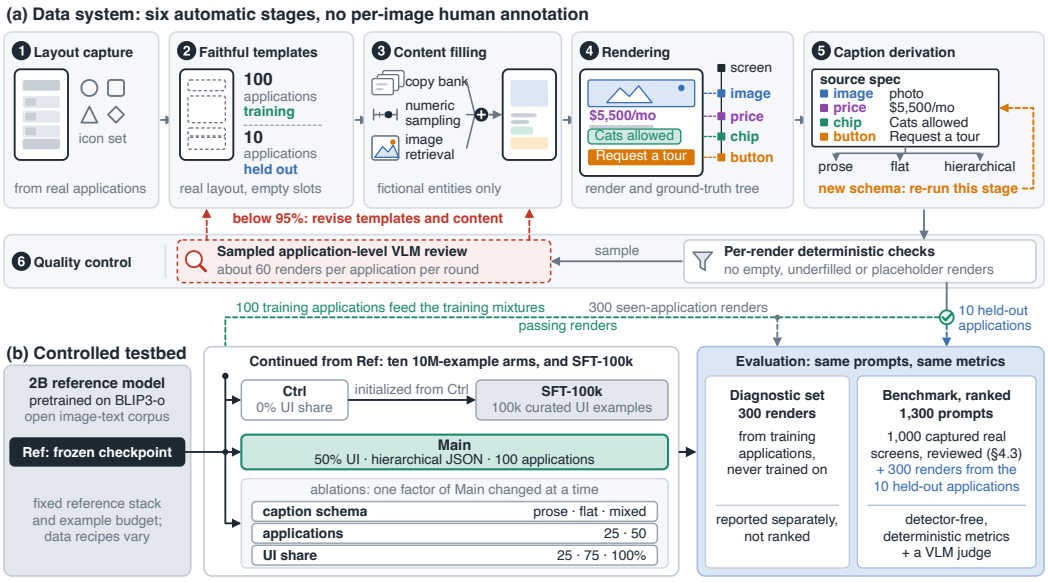


Figure 2: SCREENFORGE overview. (a) The pipeline renders captured layouts and derives captions from the stored specification at stage 5. (b) Ten arms compare data recipes using the same 2B reference model, 10M-example budget, and 1,300-prompt benchmark. The supervised fine-tuning baseline follows its own training protocol.

Table 1: **UI pool statistics**, with the held-out list frozen before any training run (§4.3).

Applications (train / held-out)	100 / 10
Images (produced / to release)	110M / 10M
Caption schemas per image	3 (prose / flat / hierarchical JSON)
Screen copy	LLM-written per-application banks; recombined per render
Content imagery	retrieved from an index of >500M images; no repeat within a screen
Sampled review pass rate	97% mean over applications; ≥95% of a fresh sample of ∼60 renders required per application

Table 2 compares the information retained by this pool and existing screenshot and visual-text datasets.

4.3 SPLITS AND RELEASE

We split synthetic data by application to separate templates across training and held-out evaluation. Ten applications are selected at random and frozen with a timestamp before any training run. They are excluded from all training mixtures; 300 of their renders test generalization to unseen applications in the primary benchmark (§5.2). The 25- and 50-application arms (§6.3) use uniformly random subsets of the training applications.

A separate 300-render set from training applications is excluded from training. These screens share templates with training data and serve as a seen-application reconstruction diagnostic (Appendix F.3), excluded from headline results and rankings.

The primary benchmark also includes 1,000 automatically captured real screens from 130 applications. Their structural GT annotations were generated by Gemini and then manually verified. With real-world content and visual details, these screens provide a complementary test of transfer from synthetic renders to real interfaces.

Table 2: **Stored information and rendering support in existing datasets.** *Struct. GT*: per-image structural ground truth; partial denotes text annotations without a component structure. *Re-derivable*: fields recoverable from stored source or annotations without re-annotation; geometry after re-render requires rendering the source. *Re-render*: stored source renders the screen again. *Recipe protocol*: data variants compared under a fixed model and example budget. The SCREENFORGE row describes retained records; release contents are specified in §4.3.

Dataset	Domain	Scale	Struct. GT	Re-derivable	Re-render	Recipe protocol
RICO (Deka et al., 2017)	mobile UI	72K	✓	text, structure, geometry	—	—
WebSight (Laurençon et al., 2024)	web	∼2M	✓	text, structure, style; geometry after re-render	✓	—
MARIO-10M (Chen et al., 2023)	scene text	10M	partial	text, geometry	—	—
TextAtlas5M (Wang et al., 2025)	dense text	5M	partial	text, geometry; style in some subsets	—	—
SCREENFORGE (ours)	app UI	**10M**	✓	text, structure, style, geometry (measured)	✓	✓

We will release every UI render appearing in a training manifest (10M distinct images), with all three captions, structured ground truth, and provenance. The evaluation release will include all 1,300 benchmark reference screens: 1,000 real screenshots and 300 held-out synthetic renders.

5 TRAINING AND EVALUATION PROTOCOL

5.1 REFERENCE STACK

We use a 2B-scale latent text-to-image model (Qwen3.5-2B (Qwen Team, 2026) text encoder, JiT (Li & He, 2026) denoising transformer, FLUX.2 latent autoencoder); architecture, initialization, and resolution policy are fixed across all controlled experiments. The reference model is pretrained on BLIP3-o (Chen et al., 2025); a 10M-example BLIP3-o subset is the background pool BG, and the UI pool follows §4. From the reference checkpoint, CTRL is continued-trained on 10M BG examples and MAIN on 5M UI plus 5M BG examples. Under the same budget we vary three data factors: caption format (prose, flat JSON, mixed, or hierarchical JSON), application diversity (25, 50, or 100 applications at a fixed UI example count), and UI-to-background mixture ratio (0%, 25%, 50%, 75%, or 100%). We train each model on 8×H200 GPUs in BF16 with token-budgeted variable-length packing; all matched arms use learning rate 2×10^{-4}, gradient clipping at 1.0, and EMA decay 0.9999; each primary endpoint is evaluated after the same pass through its fixed 10M-example manifest without replacement.

5.2 CAPABILITY METRICS

The primary UI benchmark contains 1,300 structured JSON prompts, each with a GT reference screen: 1,000 real mobile-application screens and 300 synthetic screens from ten held-out applications excluded from training. Metrics are averaged within each application, then equally across applications. A further 300 synthetic screens from training applications form a seen-application diagnostic set; since they may share application-specific patterns with training data, they serve only to study reconstruction and are excluded from headline results, model ranking, and statistical comparisons.

Our deterministic metrics do not detect or match components in generated screens. We instead use fixed GT-defined regions and detector-free full-screen measurements. Reference Layout Fidelity, S_{RefLay}, compares text-suppressed edge maps at three scales; Region Visual Similarity, S_{Region}, compares frozen SigLIP (Zhai et al., 2023) patch features within atomic non-text GT regions; Global Visual Similarity, S_{Global}, compares DreamSim (Fu et al., 2023) representations on a common can-

vas; required-text recall, S_{TRec}, is the fraction of required strings PaddleOCR (Du et al., 2020) matches; and text-position error, S_{TPos}, is coverage-aware, charging a unit error per unmatched string. A Gemini 3.6 Flash (Google, 2026a) judge, shown projected GT boxes rather than detected ones, scores Prompt Adherence on a 1–5 scale and Layout Adherence as a $\{0, 1, 2\}$ status per GT region and derived relation, aggregated to the same scale; Visual Quality comes from a separate candidate-only call. Appendix C gives the definitions.

6 EXPERIMENTS

We first benchmark existing text-to-image models on UI generation (§6.2), then run matched-budget continued-training ablations over four data choices (§6.3).

6.1 EXPERIMENTAL DESIGN

Except for the frozen reference checkpoint REF and the SFT baseline, all arms follow the matched 10M-example continued-training protocol defined in §5.1. CTRL spends the budget on background (BG) data alone, and MAIN provides the common setting across the ablations: hierarchical JSON captions, 100 applications, and a 50% UI mixture. The caption-format arms (CAP-PROSE, CAP-FLAT, CAP-MIX) use identical images and vary only their captions; the application-diversity arms (DIV-25, DIV-50) keep the total number of UI examples fixed while drawing them from 25 or 50 applications; and the mixture-ratio arms (SHARE-25, SHARE-75, SHARE-100) keep the UI pool and caption format fixed while varying the UI:BG ratio over 25:75, 75:25, and 100:0. SFT-100K is an adaptation baseline outside the matched protocol: it starts from CTRL and trains for three epochs on 100k curated UI examples. Table A1 in Appendix D lists every arm.

6.2 THE CAPABILITY GAP IN EXISTING MODELS

We benchmark commercial API systems and locally evaluated open-weight models under the shared protocol in §5.2. Every model in Table 3 receives the same unabridged structured JSON prompt and is evaluated on the 1,300-prompt primary UI benchmark. For each metric, we first average scores within each source-aware application group and then macro-average over 130 real and 10 held-out synthetic application groups. Real and synthetic applications remain distinct even when their normalized identifiers coincide. The commercial systems comprise Ideogram 4 (Ideogram, 2026); Seedream 5.0 Lite (ByteDance Seed, 2026); Gemini 2.5 Flash Image (Nano Banana), Gemini 3 Pro Image (Nano Banana Pro), and Gemini 3.1 Flash Image (Nano Banana 2) (Google, 2025; 2026c;b); and GPT-Image-1, GPT-Image-1.5, GPT-Image-2, and GPT-Image-2.5 Sunburst (OpenAI, 2025a;b; 2026a;b).

The locally evaluated open-weight models comprise NextStep-1.1 (Han et al., 2026), HiDream-O1-Image (Cai et al., 2026), LLaDA-Image (Chen et al., 2026), BAGEL (Deng et al., 2025), Emu3.5 (Cui et al., 2025), FLUX.2-dev (Black Forest Labs, 2025), and SenseNova-U1.5 (Diao et al., 2026). Only configurations verified to consume the complete structured input are included; Qwen-Image (Wu et al., 2025), SD3.5-Large (Stability AI, 2024), Lumina-Image 2.0 (Qin et al., 2025), and Z-Image (Z-Image Team et al., 2025) are excluded because theirs did not, and prompt truncation or compression would alter the task.

Among the open-weight models, MAIN leads all five deterministic and all three VLM-based metrics. Its reference-layout fidelity nearly matches GPT-Image-2.5 (0.203 versus 0.204), and its coverage-aware text-position error of 0.210 is the lowest among the open-weight systems. Across all evaluated models, it ranks third in both regional visual similarity and VLM Layout Adherence, behind only GPT-Image-2 and GPT-Image-2.5. The strongest commercial systems retain an advantage in text rendering and visual quality, with a Quality score of 4.94 compared with 2.69 for MAIN. This gap may reflect the capacity of our 2B base model and the emphasis of our training data on interface structure rather than typography and visual polish. Appendix F reports results for real screens, held-out synthetic applications, and the seen-application diagnostic set. The representative examples in Figure 3 reflect the aggregate results: MAIN preserves component hierarchy and relative placement more consistently than the open-weight baselines, while the strongest commercial systems produce cleaner typography, icons, and fine visual details.

Table 3: **UI generation performance.** All scores are application-macro averages over the 1,300-prompt primary UI benchmark, comprising 130 real and 10 held-out synthetic application groups. Deterministic metrics measure reference-layout fidelity (*Ref.-Lay.*), regional and global visual similarity (*Region / Global*), required-text recall (*T-Rec.*), and coverage-aware text-position error (*T-Pos.*); the VLM evaluation reports 1–5 ratings of prompt adherence, layout adherence, and visual quality. Only models verified to consume the complete structured JSON prompt are included. Bold and underlined values denote the best and second-best results within each group.

	Structure	Visual fidelity		Text fidelity		VLM-based evaluation		
Model	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Commercial APIs								
Ideogram 4	0.091	0.647	0.875	0.906	0.194	3.91	3.38	4.46
Seedream 5.0 Lite	0.080	0.616	0.858	0.883	0.304	3.00	2.43	3.47
Nano Banana	0.133	0.643	0.879	0.859	0.216	3.75	3.47	3.83
Nano Banana Pro	0.122	0.620	0.869	0.873	0.218	3.69	3.31	3.49
Nano Banana 2	0.148	0.621	0.872	0.856	0.219	3.62	3.68	2.86
GPT-Image-1	0.090	0.601	0.853	0.595	0.482	2.98	2.16	3.90
GPT-Image-1.5	0.114	0.619	0.879	0.807	0.297	3.84	3.17	4.37
GPT-Image-2	<u>0.186</u>	<u>0.686</u>	<u>0.903</u>	**0.949**	<u>0.108</u>	<u>4.88</u>	<u>4.68</u>	**4.94**
GPT-Image-2.5	**0.204**	**0.698**	**0.903**	<u>0.943</u>	**0.106**	**4.90**	**4.70**	**4.94**
Open-weight models								
NextStep-1.1	0.110	0.552	0.634	0.033	0.975	1.02	1.02	1.01
HiDream-O1-Image	0.006	0.571	0.640	0.003	0.998	1.08	1.08	1.10
LLaDA-Image	0.055	0.565	0.678	0.066	0.954	1.10	1.12	1.22
BAGEL	0.093	0.563	0.720	0.002	0.998	1.12	1.10	1.55
Emu3.5	0.110	0.596	0.839	0.561	0.540	2.18	1.98	2.31
FLUX.2-dev	<u>0.111</u>	<u>0.621</u>	0.848	0.369	0.707	2.23	1.77	<u>2.54</u>
SenseNova-U1.5	0.102	0.617	<u>0.856</u>	<u>0.783</u>	<u>0.362</u>	<u>2.57</u>	<u>2.42</u>	2.30
Ours (MAIN)	**0.203**	**0.680**	**0.868**	**0.813**	**0.210**	**3.44**	**3.84**	**2.69**

Table 4: **Effectiveness of synthetic UI data and an SFT baseline.** SFT-100K† is initialized from CTRL. Bold and underlined values denote the best and second-best results in each column.

	Structure	Visual fidelity		Text fidelity		VLM evaluation		
Arm	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
REF	<u>0.192</u>	0.648	0.829	0.245	0.759	2.64	3.11	1.52
CTRL	0.171	0.649	0.818	0.465	0.539	2.69	3.24	1.58
SFT-100K†	0.171	**0.689**	<u>0.852</u>	<u>0.747</u>	<u>0.275</u>	<u>3.07</u>	<u>3.63</u>	<u>2.27</u>
MAIN	**0.203**	<u>0.680</u>	**0.868**	**0.813**	**0.210**	**3.44**	**3.84**	**2.69**

6.3 DATA AND TRAINING ABLATIONS

Effectiveness and an SFT baseline. Table 4 compares MAIN with REF, CTRL, and SFT-100K. Relative to REF, MAIN increases required-text recall from 0.245 to 0.813, Prompt Adherence from 2.64 to 3.44, and Layout Adherence from 3.11 to 3.84. Its gains in regional and global similarity further show that the learned capability is not limited to text rendering, while the smaller improvements from CTRL indicate that additional training on background data alone is insufficient. SFT-100K achieves slightly higher regional similarity than MAIN (0.689 versus 0.680), but remains lower on the other seven metrics. Because it is initialized from CTRL and lies outside the matched 10M-example protocol, we treat it as an adaptation baseline rather than a controlled comparison of training paradigms (Appendix D).

Application diversity. Under the fixed UI sample budget, the 100-application arm performs best on seven of the eight metrics in Table 5. Prompt Adherence improves from 3.34 with 25 applications to 3.45 with 50 applications and remains effectively saturated at 3.44 with 100 applications. As

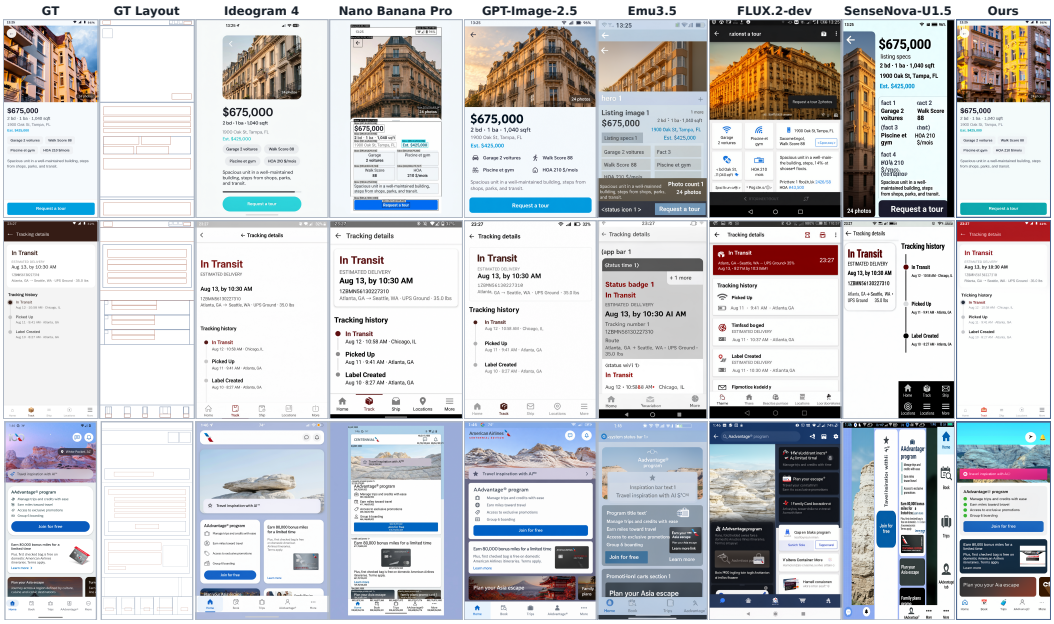


Figure 3: **Qualitative comparison on our benchmark.** Rows show representative examples from our benchmark, spanning different interface types and layout structures. Columns show the GT reference, its annotated layout, selected commercial systems, representative open-weight baselines, and MAIN. Compared with the open-weight baselines, MAIN more consistently preserves the requested component hierarchy, relative placement, and overall interface organization, while the strongest commercial systems render typography, icons, and fine visual details more cleanly.

Table 5: **Application diversity at a fixed UI sample budget.** All arms use the same total UI sample budget, hierarchical caption format, and 50:50 UI–background mixture. Results use the 1,300-prompt primary UI benchmark. Bold values denote the best result in each column.

	Structure	Visual fidelity		Text fidelity		VLM evaluation		
Applications	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
25	0.180	0.677	0.860	0.765	0.257	3.34	3.69	2.55
50	0.173	0.680	0.858	0.798	0.227	**3.45**	3.75	2.64
100	**0.203**	**0.680**	**0.868**	**0.813**	**0.210**	3.44	**3.84**	**2.69**

coverage increases from 25 to 50 and 100 applications, required-text recall rises from 0.765 to 0.798 and 0.813, coverage-aware text-position error decreases from 0.257 to 0.227 and 0.210, and Layout Adherence improves from 3.69 to 3.75 and 3.84. Reference-layout and global similarity exhibit small non-monotonic variations between 25 and 50 applications before reaching their highest values at 100 applications.

Caption representation. Figure 4a reports an independently generated cross-format sweep on 300 screens from ten held-out applications. Within this independently generated cross-format sweep, matched training and inference formats obtain the strongest scores across the evaluated prompt views. In particular, hierarchical training achieves Prompt Adherence, Layout Adherence, and Visual Quality scores of 4.13, 4.43, and 3.80 under hierarchical JSON prompting. Mixed-format training is the most robust when the inference representation differs from the training format: averaged over the three off-diagonal cells in the mixed-training row, it achieves 3.38, 3.60, and 3.52, respectively, outperforming the other caption arms on all three cross-format averages. Appendix G reports the complete caption-ablation and cross-format results.

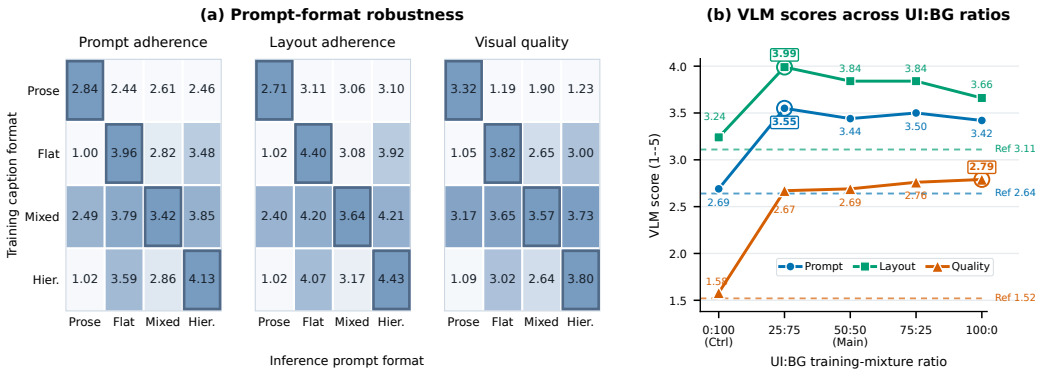


Figure 4: **Prompt-format robustness and UI-mixture ablations.** **(a)** Cross-format evaluation of the caption-representation arms on 300 screens from ten held-out applications. Rows indicate the caption format used during training, and columns indicate the prompt format used at inference time. The three heatmaps report Prompt Adherence, Layout Adherence, and Visual Quality. Cell colors are normalized within each inference-format column, with darker shades indicating stronger relative performance; the printed values are absolute scores on a 1–5 scale. Outlined diagonal cells denote matched training and inference formats. **(b)** Prompt Adherence, Layout Adherence, and Visual Quality on the complete 1,300-prompt primary UI benchmark, reported across UI:BG ratios in $\{0{:}100, 25{:}75, 50{:}50, 75{:}25, 100{:}0\}$. The $0{:}100$ setting is the background-only continued-training control, and $50{:}50$ is MAIN. Dashed horizontal lines show the frozen reference model, and outlined markers identify the best mixture result for each metric.

UI mixture ratio. Figure 4b reports the three VLM metrics on the complete 1,300-prompt primary benchmark. The background-only control remains close to the frozen reference checkpoint: its Prompt Adherence, Layout Adherence, and Visual Quality scores are 2.69, 3.24, and 1.58, compared with 2.64, 3.11, and 1.52 for REF. Introducing a 25% UI share produces the largest gains, raising the three scores to 3.55, 3.99, and 2.67. Both adherence metrics reach their highest values at this mixture ratio; at 100% UI, they decrease to 3.42 and 3.66. Visual Quality instead increases gradually from 2.67 at 25% UI to 2.79 at 100% UI. Thus, the 25% mixture provides the strongest adherence performance among the evaluated ratios, while larger UI shares trade some prompt and layout control for a modest improvement in rendering quality. MAIN (50%) remains competitive but is not the VLM-optimal operating point in this sweep.

7 CONCLUSION

In this paper, we studied how to construct synthetic UI data for training text-to-image models. We introduced SCREENFORGE, which renders interface screens and retains their source documents and component trees for caption derivation. Using the same 2B model and a fixed example budget, we compared caption recipes, application coverage, and UI-to-background mixtures.

Our experiments show that synthetic UI data improves text rendering and layout adherence over background-only continued training. The preferred caption recipe depends on the inference prompt format, while broader application coverage improves most metrics at a fixed UI sample count. The UI share also requires a trade-off: moderate shares favor text and layout adherence, while larger shares yield modestly higher visual quality. These findings come from single training runs on one model architecture; layout fidelity remains lower on real interfaces than on held-out synthetic screens.

The limited amount of data used to train our model may constrain its general text-to-image capabilities and contribute to weaknesses in typography and fine visual detail on the UI benchmark (§6.2). Limited training coverage may also affect generalization to unfamiliar content and layouts. The relative benefits of the tested data recipes may change with substantially larger training datasets.

We will release 10M training renders, captioning code, and the evaluation suite.

AI USE STATEMENT

We used generative AI in the data pipeline and evaluation (§4.1, §5.2) and to help draft sections, and edit the manuscript. An LLM coding agent builds application templates and generators from captured layouts, and an LLM writes the screen copy banks. Synthetic-image captions are derived deterministically from stored structure and quote that copy. For real benchmark screens, Gemini generated the structural GT annotations, which were then manually verified. A VLM reviews sampled renders; a separate model, Gemini 3.6 Flash, serves as the evaluation judge.

We have reviewed all AI-assisted work. Renders from the AI-written templates and copy enter the pool only after passing the deterministic checks of stage 6. VLM review of about 60 renders per application repeats until at least 95% pass, after which the authors inspect the judged sample (§4.1). All technical content, experiments, and claims were authored and verified by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI. The work contains no theoretical models, mathematical claims, or proofs, so those disclosure items do not apply.

ETHICS STATEMENT

Our templates follow real application layouts, and the synthetic renders inherit their styling. Fonts and templates undergo a license audit before release. Real screenshots serve as references for layout analysis and evaluation (§4.3) and are not used to train our models. The 1,000 real benchmark screenshots will be released as part of the evaluation suite.

Seen-application diagnostics are reported separately (Appendix F.3) to show reconstruction of familiar patterns. The synthetic-data pipeline uses no user data or per-image human annotation: an LLM writes copy about fictional entities, interface labels come from locale dictionaries, and imagery comes from a stock image index. Human review covers the sampled pipeline outcomes (§4.1) and the benchmark's real screens and their structural GT annotations (§4.3).

REPRODUCIBILITY STATEMENT

We will release the 10M synthetic training renders with all three captions, structured ground truth, and provenance, together with caption derivation code, per-arm image-ID manifests, the BG manifest, and the evaluation suite. The release will also include the held-out application list with its freeze timestamp. BLIP3-o is the public source of the background data, identified by image ID in the BG manifest. The released renders, captions, and manifests provide fixed inputs for further studies of UI training data. The synthesis generators and model checkpoints will not be released. Training and evaluation settings are given in §5.1 and §5.2.

The evaluation suite will include the 1,300 benchmark prompts with their reference screens, comprising the 300 held-out synthetic renders and the 1,000 captured real screens (§4.3), to support evaluation on the same benchmark (Appendix F.1). Each arm is one run over a fixed manifest.

REFERENCES

Gilles Baechler, Srinivas Sunkara, Maria Wang, Fedir Zubach, Hassan Mansoor, Vincent Etter, Victor Cărbune, Jason Lin, Jindong Chen, and Abhanshu Sharma. ScreenAI: A vision-language model for UI and infographics understanding. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 3058–3068, 2024. doi: 10.24963/ijcai.2024/339.

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. Improving image generation with better captions. Technical report, OpenAI, 2023. URL https://cdn.openai.com/papers/dall-e-3.pdf.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. Official Technical Report, 2025. URL https://bfl.ai/blog/flux-2. Accessed September 20, 2026.

ByteDance Seed. Deeper Thinking, More Accurate Generation: Introducing Seedream 5.0 Lite. Official Technical Report, 2026. URL https://seed.bytedance.com/en/blog/

deeper-thinking-more-accurate-generation-introducing-seedream-5-0-lite. Evaluated model: seedream-5-0-260128; accessed September 20, 2026.

Qi Cai, Jingwen Chen, Chengmin Gao, Zijian Gong, Yehao Li, Yingwei Pan, Yi Peng, Zhaofan Qiu, Kai Yu, Yiheng Zhang, Hao Ai, Siying Bai, Yang Chen, Zhihui Chen, Fengbin Gao, Ying Guo, Dong Li, Zhen Shen, Leilei Shi, Jing Wang, Siyu Wang, Yimeng Wang, Rui Zheng, Ting Yao, and Tao Mei. HiDream-O1-Image: A natively unified image generative foundation model with pixel-level unified transformer. *arXiv preprint arXiv:2605.11061*, 2026. URL https://arxiv.org/abs/2605.11061.

Chuyan Chen, Haoxing Chen, Kun Chen, Zhenglin Cheng, Long Cui, Ruishan Fang, Zhangxuan Gu, Zhicheng Huang, Zhenzhong Lan, Yuanting Lei, Haoquan Li, Jianguo Li, Rongchuan Li, Sidu Li, Tao Lin, Deyuan Liu, Jiacheng Liu, Lin Liu, Yuxuan Lou, Zhisheng Lu, Yuxin Ma, Shuheng Shen, Peng Sun, Chaoyang Wang, Hongjun Wang, Xiaomei Wang, Yongxin Wang, Chengzhang Wu, Hongru Wu, and Jun Xie. LLaDA-Image: Building strong image generators with fully open training recipes. *arXiv preprint arXiv:2609.03796*, 2026. URL https://arxiv.org/abs/2609.03796.

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. TextDiffuser: Diffusion models as text painters. *Advances in Neural Information Processing Systems*, 36, 2023.

Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, Le Xue, Caiming Xiong, and Ran Xu. BLIP3-o: A family of fully open unified multimodal models — architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025. URL https://arxiv.org/abs/2505.09568.

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

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pp. 845–854, 2017. doi: 10.1145/3126594.3126651.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025. URL https://arxiv.org/abs/2505.14683.

Haiwen Diao, Jiahao Wang, Chenjing Ding, Hanming Deng, Jiangnan Chen, Ruixi Zhang, Ruohui Wang, Wenwen Tong, Xiangyu Fan, Yubo Wang, Yue Zhu, Yuwei Niu, Zhengqi Bai, Zhiqian Lin, Zhitao Yang, Zhongang Cai, Bo Yang, Chen Feng, Chengguang Lv, Guangjia Liu, Guanlin Wang, Hanyu Zhang, Haojia Yu, Hongcan Xiao, Hongli Wang, Huan Wu, Huaping Zhong, Jian Fang, Jianan Fan, Jiaqi Li, Jiefan Lu, Jing Zuo, Jingcheng Ni, Junxiang Xu, Linjun Dai, Mutian Xu, Peishen Yan, Penghao Wu, Ruijie Mao, Ruisi Wang, Shihao Bai, Shuang Yang, Shuya Yang, Shuyan Zheng, Silei Wu, Siying Li, Tao Chu, Tianbo Zhong, Tongxi Zhou, Weichao Luo, Weichen Fan, Wenhao Jia, Wenjie Gao, Xiangli Kong, Yan Li, Yang Yong, Zimo Wen, Zixuan Qian, Wenxiu Sun, Ruihao Gong, Quan Wang, Lewei Lu, Lei Yang, Ziwei Liu, and Dahua Lin. SenseNova-U1.5: Towards native unified visual intelligence. *arXiv preprint arXiv:2609.11929*, 2026. URL https://arxiv.org/abs/2609.11929.

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. PP-OCR: A practical ultra lightweight OCR system. *arXiv preprint arXiv:2009.09941*, 2020. URL https://arxiv.org/abs/2009.09941.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In

Proceedings of the 41st International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 2024.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, volume 36, pp. 50742–50768, 2023.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. DataComp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems*, 36, 2023.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36, 2023.

Google. Gemini 2.5 Flash Image (Nano Banana). Gemini API Documentation, 2025. URL https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash-image. Evaluated model: gemini-2.5-flash-image; accessed September 20, 2026.

Google. Gemini 3.6 Flash. https://ai.google.dev/gemini-api/docs/models/gemini-3.6-flash, 2026a. Official model documentation; accessed September 20, 2026.

Google. Gemini 3.1 Flash Image (Nano Banana 2). Gemini API Documentation, 2026b. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.1-flash-image. Evaluated model: gemini-3.1-flash-image; accessed September 20, 2026.

Google. Gemini 3 Pro Image (Nano Banana Pro). Gemini API Documentation, 2026c. URL https://ai.google.dev/gemini-api/docs/models/gemini-3-pro-image. Evaluated model: gemini-3-pro-image; accessed September 20, 2026.

Kamal Gupta, Justin Lazarow, Alessandro Achille, Larry S. Davis, Vijay Mahadevan, and Abhinav Shrivastava. LayoutTransformer: Layout generation and completion with self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1004–1014, 2021.

Chunrui Han, Guopeng Li, Jingwei Wu, Quan Sun, Yan Cai, Yuang Peng, Zheng Ge, Deyu Zhou, Haomiao Tang, Hongyu Zhou, Kenkun Liu, Shu-Tao Xia, Binxing Jiao, Daxin Jiang, Xiangyu Zhang, and Yibo Zhu. NextStep-1: Toward autoregressive image generation with continuous tokens at scale. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/974ff7b5bf08dbf9400b5d599a39c77f-Abstract-Conference.html.

Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip diffusion models with LLM for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024. URL https://arxiv.org/abs/2403.05135.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2I-CompBench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2023.

Ideogram. Ideogram 4.0 Technical Details: Open Model at the Forefront of Design. Official Technical Report, 2026. URL https://ideogram.ai/blog/ideogram-4.0/. Accessed September 20, 2026.

Naoto Inoue, Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. LayoutDM: Discrete diffusion model for controllable layout generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10167–10176, 2023.

Peidong Jia, Chenxuan Li, Yuhui Yuan, Zeyu Liu, Yichao Shen, Bohan Chen, Xingru Chen, Yinglin Zheng, Dong Chen, Ji Li, Xiaodong Xie, Shanghang Zhang, and Baining Guo. COLE: A hierarchical generation framework for multi-layered and editable graphic design. *arXiv preprint arXiv:2311.16974*, 2023. URL https://arxiv.org/abs/2311.16974.

Hugo Laurençon, Léo Tronchon, and Victor Sanh. Unlocking the conversion of web screenshots into HTML code with the WebSight dataset. *arXiv preprint arXiv:2403.09029*, 2024. URL https://arxiv.org/abs/2403.09029.

Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2Struct: Screenshot parsing as pretraining for visual language understanding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18893–18912, 2023.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, et al. DataComp-LM: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37, 2024.

Jianan Li, Jimei Yang, Aaron Hertzmann, Jianming Zhang, and Tingfa Xu. LayoutGAN: Generating graphic layouts with wireframe discriminators. In *International Conference on Learning Representations*, 2019.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2511.13720.

Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. GLIGEN: Open-set grounded text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22511–22521, 2023.

Zeyu Liu, Weicong Liang, Zhanhao Liang, Chong Luo, Ji Li, Gao Huang, and Yuhui Yuan. Glyph-ByT5: A customized text encoder for accurate visual text rendering. In *Computer Vision – ECCV 2024*, pp. 361–377. Springer, 2024. doi: 10.1007/978-3-031-73226-3_21.

Pratyush Maini, Sachin Goyal, Zachary C. Lipton, J. Zico Kolter, and Aditi Raghunathan. T-MARS: Improving visual representations by circumventing text feature learning. In *International Conference on Learning Representations*, 2024.

Mark Mazumder, Colby Banbury, Xiaozhe Yao, Bojan Karlaš, et al. DataPerf: Benchmarks for data-centric AI development. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0235.

OpenAI. GPT-Image-1 Model. OpenAI API Documentation, 2025a. URL https://developers.openai.com/api/docs/models/gpt-image-1. Accessed September 20, 2026.

OpenAI. GPT-Image-1.5 Model. OpenAI API Documentation, 2025b. URL https://developers.openai.com/api/docs/models/gpt-image-1.5. Accessed September 20, 2026.

OpenAI. GPT-Image-2 Model. OpenAI API Documentation, 2026a. URL https://developers.openai.com/api/docs/models/gpt-image-2. Accessed September 20, 2026.

OpenAI. GPT-Image-2.5 Sunburst Model. OpenAI API Documentation, 2026b. URL https://developers.openai.com/api/docs/models/gpt-image-2.5-sunburst. Evaluated model: gpt-image-2.5-sunburst; accessed September 20, 2026.

Qi Qin, Le Zhuo, Yi Xin, Ruoyi Du, Zhen Li, Bin Fu, Yiting Lu, Xinyue Li, Dongyang Liu, Xiangyang Zhu, Will Beddow, Erwann Millon, Victor Perez, Wenhai Wang, Yu Qiao, Bo Zhang, Xiaohong Liu, Hongsheng Li, Chang Xu, and Peng Gao. Lumina-Image 2.0: A unified and efficient image generative framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20031–20042, 2025.

Qwen Team. Qwen3.5-2B. https://huggingface.co/Qwen/Qwen3.5-2B, 2026. Official model card; accessed September 20, 2026.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y.

Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. Design2Code: Benchmarking multimodal code generation for automated front-end engineering. *arXiv preprint arXiv:2403.03163*, 2024. URL https://arxiv.org/abs/2403.03163.

Stability AI. Introducing Stable Diffusion 3.5. Official Model Release, 2024. URL https://stability.ai/news/introducing-stable-diffusion-3-5.

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*, 2024.

Alex Jinpeng Wang, Dongxing Mao, Jiawei Zhang, Weiming Han, Zhuobai Dong, Linjie Li, Yiqi Lin, Zhengyuan Yang, Libo Qin, Fuwei Zhang, Lijuan Wang, and Min Li. TextAtlas5M: A large-scale dataset for dense text image generation. *arXiv preprint arXiv:2502.07870*, 2025. URL https://arxiv.org/abs/2502.07870. Accepted to ICML 2026.

Jialiang Wei, Anne-Lise Courbis, Thomas Lambolais, Binbin Xu, Pierre Louis Bernard, and Gérard Dray. Boosting GUI prototyping with diffusion models. In *Proceedings of the 31st IEEE International Requirements Engineering Conference (RE), RE@Next! track*, 2023. URL https://arxiv.org/abs/2306.06233.

Haohan Weng, Danqing Huang, Yu Qiao, Zheng Hu, Chin-Yew Lin, Tong Zhang, and C. L. Philip Chen. Desigen: A pipeline for controllable design template generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12721–12732, 2024.

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

Jiannan Xiang, Yun Zhu, Lei Shu, Maria Wang, Lijun Yu, Gabriel Barcik, James Lyon, Srinivas Sunkara, and Jindong Chen. UISim: An interactive image-based UI simulator for dynamic mobile environments. *arXiv preprint arXiv:2509.21733*, 2025. URL https://arxiv.org/abs/2509.21733.

Yukang Yang, Dongnan Gui, Yuhui Yuan, Weicong Liang, Haisong Ding, Han Hu, and Kai Chen. GlyphControl: Glyph conditional control for visual text generation. *Advances in Neural Information Processing Systems*, 36, 2023.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-UI: Grounded mobile UI understanding with multimodal LLMs. In *Computer Vision – ECCV 2024*, pp. 240–255, 2024.

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

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11975–11986, 2023.

Tianming Zhao, Chunyang Chen, Yuanning Liu, and Xiaodong Zhu. GUIGAN: Learning to generate GUI designs using generative adversarial networks. In *Proceedings of the 43rd IEEE/ACM International Conference on Software Engineering*, pp. 748–760, 2021. doi: 10.1109/ICSE43902.2021.00074.

Guangcong Zheng, Xianpan Zhou, Xuewei Li, Zhongang Qi, Ying Shan, and Xi Li. LayoutDiffusion: Controllable diffusion model for layout-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22490–22499, 2023.

A Extended related work

Synthetic training data for generative models. Re-captioning images with stronger models provides synthetic *text* supervision that improves T2I prompt following (Betker et al., 2023; Esser et al., 2024). Training recursively on synthetic *images* sampled from a model's own outputs can degrade performance (Shumailov et al., 2024). Rendering provides another source of training images, with labels derived from the renderer's source specification. We study how to construct and use such data at scale.

Screenshots for UI understanding. Screen corpora support tasks such as mobile design mining with RICO (Deka et al., 2017), screenshot-to-HTML with WebSight's LLM-written pages (Laurençon et al., 2024), web reconstruction with Design2Code (Si et al., 2024), and pretraining screenshot encoders and UI-focused VLMs. These include Pix2Struct, ScreenAI, and Ferret-UI (Lee et al., 2023; Baechler et al., 2024; You et al., 2024).

Layout and interface generation. LayoutGAN, LayoutTransformer, and LayoutDM generate element boxes and types (Li et al., 2019; Gupta et al., 2021; Inoue et al., 2023). Layout-to-image models such as GLIGEN and LayoutDiffusion condition image generation on given boxes (Li et al., 2023; Zheng et al., 2023). GUIGAN generates GUI designs with a GAN (Zhao et al., 2021); related work generates graphic-design templates (Weng et al., 2024; Jia et al., 2023). UI-Diffuser renders mobile screens from text and component specifications with a diffusion model (Wei et al., 2023), and UISim synthesizes the next screen state after a user action (Xiang et al., 2025). Our study concerns the caption schema, application count, and UI share of synthetic training data for a general T2I model, compared under a fixed model and example budget.

Visual text generation. Work on visual text improves text rendering inside natural images through OCR-annotated corpora and glyph conditioning: TextDiffuser, GlyphControl, AnyText, GlyphByT5, and dense-text datasets such as TextAtlas5M (Chen et al., 2023; Yang et al., 2023; Tuo et al., 2024; Liu et al., 2024; Wang et al., 2025). Interfaces require accurate text placed within a component hierarchy. Our text-fidelity metrics, required-text recall and text placement, follow this work's OCR-based matching approach, using PaddleOCR.

Data-centric protocols and T2I evaluation. DataComp and its language-model successor compare data under a fixed model and training procedure (Gadre et al., 2023; Li et al., 2024; Mazumder et al., 2023). We adapt this design to T2I training. GenEval, T2I-CompBench, and DPG-Bench (Ghosh et al., 2023; Huang et al., 2023; Hu et al., 2024) evaluate models on natural-image prompts. Our 1,300-prompt UI benchmark comprises 1,000 real screens and 300 screens from ten held-out applications. It compares existing systems and evaluates data recipes under a fixed model and example budget, using reference-based deterministic metrics and a fixed VLM judge.

Crawled and instrumented screenshots. Web-scraped corpora are not built for interfaces. Text-aware curation may also remove screenshots: T-MARS masks image text and drops pairs whose caption aligns with the image only through that text (Maini et al., 2024). Screenshots can be mined and re-captioned with a vision-language model (Betker et al., 2023), although many web examples are marketing images, annotated tutorials, or low-resolution crops. Without the source component tree, a captioning model must infer text, hierarchy, and geometry from pixels, with possible annotation errors. Schemas requiring attributes absent from the stored captions need additional annotation.

Instrumented captures can preserve view hierarchies, as in RICO (Deka et al., 2017). Accessibility trees may omit custom-rendered widgets and visual styling. These captures cover the screens reached by the crawler and cannot be re-rendered with new content. Our generators retain the source needed to render new content and derive captions in schemas supported by the stored source and component tree.

B Pipeline and pool details

In the pipeline of §4.1, layout capture, template construction, and copy banks are organized by application. Each render draws content from these banks, then undergoes rendering, caption derivation, and deterministic checks. Stage 6 additionally reviews samples from each application; the authors inspect the final judged sample.

Stage 1: layout capture. A scripted capture extracts layout patterns, view hierarchies, and iconography from real applications.

Stage 2: parameterized templates. An LLM coding agent builds templates for each application (100 for training plus ten held out) from its captures. These cover the observed screen types: home feeds and grids, search and result lists, item detail pages, and, where captured, carts, account pages, and booking forms.

Stage 3: content filling. An LLM supplies product names, messages, menu items, and other copy to a deduplicated bank for each application. Prices, counts, and other numeric fields are sampled programmatically. Content imagery comes from an index of more than 500M images, mostly photographs. Each render draws copy and photographs afresh, without repeating a photograph within a screen. Individual strings and photographs recur across renders in varying combinations within each template.

Stage 4: rendering. A headless browser renders each screen. We store its source document and measured component tree, including types, layout boxes, visibility, and visible text. The source records the template, parameters, copy, numeric values, and image choices needed to rerun caption derivation.

Stage 5: caption derivation. Captions are derived deterministically from the stored source document and measured component tree: a prose description, a flat JSON list of visible components, and hierarchical JSON preserving the tree. The schemas differ in both format and represented structure (Figure 1). For schemas supported by the stored fields, only this stage needs to be rerun.

For CAP-MIX, each UI training example is paired with exactly one of the three schemas. After the UI records are placed in a fixed application-major order, the schema is assigned cyclically by the zero-based global record index: prose, flat JSON, and hierarchical JSON for indices congruent to 0, 1, and 2 modulo 3, respectively. This produces a balanced mixture whose schema counts differ by at most one example. The assignment is fixed when the dataset is materialized and is not resampled during training. Thus, mixed-format training combines the three concrete caption schemas rather than introducing a fourth schema.

Stage 6: quality control. Each render undergoes deterministic checks for an empty or underfilled main area or a placeholder in an image slot. Passing renders form the pool. A VLM distinct from the evaluation judge of §5.2 reviews a random sample of about 60 renders per application. Given a render and its hierarchical caption, it scores faithfulness of the described elements, coverage (no region omitted or invented), text legibility, and realism (no render defects such as overflow or missing thumbnails) from 1 to 5. A render passes when all four scores are at least 4; ambiguous failures are re-judged by a stronger judge, whose verdict stands. An application's pass rate is the passing share of its sample.

While the rate is below 95%, its templates or content are revised and a fresh sample of about 60 renders is drawn, independently of earlier rounds, and judged. Once the rate reaches 95%, the authors inspect that round's judged sample. The 97% in Table 1 averages final-round pass rates across applications. It describes those fresh review samples; the VLM does not score the full pool.

C METRIC DEFINITIONS

The eight metrics in §5.2 are defined below.

Detector-free deterministic metrics. We compare generated screens using fixed GT-defined regions and full-screen measurements. Generated components have no explicit correspondence with the reference annotation; detecting and matching them would introduce detector and localization errors into the evaluation.

Let $\mathcal{S}$ denote the structural comparison scales, $\mathcal{R}$ the atomic non-text GT regions, and $\mathcal{P}(r)$ the visual-feature patches assigned to region r. Let $\mathcal{T}$ denote the required GT text strings and $\mathcal{M}$ the

one-to-one matches between required strings and OCR candidates. Following the order used in our result tables, we compute

$$S_{\mathrm{RefLay}} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \exp\left(- \frac{d_{\mathrm{sym}}\left(E^{(s)}_{\mathrm{gen}}, E^{(s)}_{\mathrm{ref}}\right)}{\tau_s} \right), \qquad \tau_s = 0.01 D_s, \tag{1}$$

$$S_{\mathrm{Region}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \frac{1}{|\mathcal{P}(r)|} \sum_{p \in \mathcal{P}(r)} \frac{1 + \cos(\phi_p(I_{\mathrm{gen}}), \phi_p(I_{\mathrm{ref}}))}{2}, \tag{2}$$

$$S_{\mathrm{Global}} = \frac{1 + \cos\left(\psi(\widetilde{I}_{\mathrm{gen}}), \psi(\widetilde{I}_{\mathrm{ref}})\right)}{2}, \tag{3}$$

$$S_{\mathrm{TRec}} = \frac{|\mathcal{M}|}{|\mathcal{T}|}, \qquad S_{\mathrm{TPos}} = \frac{\sum_{(\hat{t},t) \in \mathcal{M}} d_{\mathrm{pos}}(\hat{t}, t) + |\mathcal{T}| - |\mathcal{M}|}{|\mathcal{T}|}. \tag{4}$$

Reference-layout fidelity. Reference-layout fidelity compares text-suppressed edge maps at three width-by-height scales:

$$\mathcal{S} = \{(512, 1024), (256, 512), (128, 256)\}.$$

At each scale, the reference and generated screens are converted to grayscale using bilinear resizing, smoothed with a 5×5 Gaussian kernel with $\sigma = 1.0$, and processed with Canny thresholds of 60 and 160. The interiors of all GT text boxes are suppressed in both edge maps with a one-pixel inset, reducing sensitivity to glyph shapes while preserving nearby component boundaries.

The symmetric edge distance d_{sym} is the mean of the two directed edge-to-edge distances computed with an L_2 distance transform. For a scale with width w_s and height h_s, $D_s = \sqrt{w_s^2 + h_s^2}$ and $\tau_s = 0.01 D_s$. If both edge maps are empty, the scale score is 1; if exactly one is empty, it is 0. The final reference-layout fidelity is the equal average of the three scale scores.

Regional and global visual similarity. Regional similarity uses the frozen SigLIP vision encoder (Zhai et al., 2023). Both screens are converted to RGB and resized bicubically to 224×224, producing a 14×14 patch grid. Patch embeddings are L_2-normalized, and their cosine similarities are mapped from $[-1, 1]$ to $[0, 1]$. The evaluated regions are atomic non-text GT elements; text elements and hierarchical containers are excluded to avoid counting the same pixels repeatedly. A patch belongs to a region when its normalized center lies inside the region. If no patch center lies inside a narrow region, the nearest grid row or column is used. Patch scores are first averaged within each region and then equally averaged across regions.

Global similarity uses the pretrained DreamSim ensemble (Fu et al., 2023). The reference and generated screens are mapped by bicubic resizing to a common 512×1024 logical canvas. Their DreamSim embeddings are L_2-normalized, and the cosine similarity is mapped to $[0, 1]$ using $(1 + \cos)/2$.

Required-text recall and text-position error. Text is detected with PaddleOCR (Du et al., 2020), with angle classification enabled. OCR detections with confidence below 0.5 are discarded. Required and recognized strings are normalized using Unicode NFKC normalization and case folding, followed by removal of all non-alphanumeric characters.

Matching candidates include individual OCR spans and merges of up to three geometrically adjacent spans. A required string and OCR candidate form an eligible pair when their normalized Levenshtein similarity,

$$1 - \frac{d_{\mathrm{edit}}(t, \hat{t})}{\max(|t|, |\hat{t}|, 1)},$$

is at least 0.70. Eligible pairs are sorted by decreasing text similarity and then by increasing center distance. A greedy global assignment consumes each required string and each underlying OCR span at most once. Consequently, repeated required strings must be supported by distinct OCR detections.

For a matched pair, text-position displacement is

$$d_{\mathrm{pos}}(\hat{t}, t) = \frac{\left\| c(\hat{b}_{\hat{t}}) - c(b_t) \right\|_2}{\sqrt{2} \times 1000},$$

where b_t and $\hat{b}_{\hat{t}}$ are the GT and OCR boxes and $c(\cdot)$ returns the box center on the normalized 1000×1000 canvas. Required-text recall is the fraction of required strings that receive a match. Text-position error is coverage-aware: a matched string contributes its normalized center displacement, whereas each unmatched required string contributes the maximum unit error. Thus, $S_{\mathrm{TPos}} \in [0,1]$, lower values are better, and $S_{\mathrm{TPos}} = 1$ when none of the required strings is matched.

VLM evaluation. Gemini 3.6 Flash evaluates generated screens. For Prompt and Layout Adherence, the judge receives the structured JSON prompt, the unannotated GT reference and generated screens, and paired annotation panels in which the same normalized GT boxes and numeric identifiers are projected onto both images. No candidate-side boxes are detected or predicted.

Prompt Adherence, $S_{\mathrm{Prompt}} \in \{1,\dots,5\}$, measures whether the generated screen realizes the requested purpose, content, and components. Scores of 1, 3, and 5 indicate failure, a recognizable but materially incomplete result, and complete fulfillment, respectively.

Layout Adherence evaluates every valid GT annotation region and every derived containment or horizontal/vertical ordering relation. Each region and relation receives a status in $\{0,1,2\}$, denoting failure, approximate satisfaction, or correct satisfaction. Let $\mathcal{R}_J$ and $\mathcal{Q}_J$ denote the complete sets of evaluated regions and relations, with region status u_r and initial relation status v_q. For a relation q whose endpoint regions are $a(q)$ and $b(q)$, we define

$$\bar{v}_q = \min\big(v_q,\, u_{a(q)},\, u_{b(q)}\big)\,,$$

preventing a relation from receiving credit when either required component is absent. We then compute

$$A_{\mathrm{Region}} = \frac{\sum_{r\in\mathcal{R}_J} u_r}{2|\mathcal{R}_J|}, \qquad A_{\mathrm{Relation}} = \frac{\sum_{q\in\mathcal{Q}_J} \bar{v}_q}{2|\mathcal{Q}_J|}, \qquad |\mathcal{Q}_J|>0, \quad (5)$$

$$A_{\mathrm{Layout}} = \begin{cases} \dfrac{A_{\mathrm{Region}} + A_{\mathrm{Relation}}}{2}, & |\mathcal{Q}_J|>0, \\ A_{\mathrm{Region}}, & |\mathcal{Q}_J|=0, \end{cases} \qquad S_{\mathrm{Layout}} = 1 + 4A_{\mathrm{Layout}}. \quad (6)$$

The affine transformation $1 + 4A_{\mathrm{Layout}}$ maps the normalized layout score from $[0,1]$ to the same $[1,5]$ scale used by the other VLM metrics.

Visual Quality is evaluated in a separate candidate-only call. The judge receives only the raw generated screen, without the prompt, GT reference, annotations, or layout relations, and assigns 1–5 ratings for whole-screen coherence and local rendering quality:

$$S_{\mathrm{Quality}} = \frac{S_{\mathrm{Coherence}} + S_{\mathrm{Rendering}}}{2}.$$

The prompt-and-layout rubric permits stylistic variation when the requested content, hierarchy, spatial relationships, navigation, and functional organization are preserved. Differences caused solely by supported output resolution or padding are ignored, whereas missing components, cropping, distortion, and displaced content remain penalized. The candidate-only quality score separately evaluates typography, icons, imagery, spacing, clipping, overlap, and rendering artifacts without rewarding similarity to the GT style.

D TRAINING ARMS

Table A1 lists the arms used in §6.1. The application-diversity arms keep the total number of UI examples fixed by sampling each retained application $4\times$ (DIV-25) or $2\times$ (DIV-50) as often as under MAIN. Each ablation changes one recipe factor with the remaining training settings fixed; budgets are matched in examples, not tokens.

Scope of the comparisons. The testbed of §3 adapts DataComp's fixed-model design (Gadre et al., 2023) to generative training. Generated screens are evaluated for layout fidelity, regional and global visual similarity, and required-text recall and placement using reference-based image and text metrics and a fixed VLM judge (§5.2). A recipe can select data, derive different captions, or render new

Table A1: **Experimental design.** MAIN provides the common setting for the caption-format, application-diversity, and mixture-ratio ablations. All continued-training arms use a 10M-example budget unless otherwise noted.

Arm	Study	UI subset	UI caption format	UI:BG	Budget
REF	Effectiveness	n/a	n/a	n/a	Frozen
CTRL	Effectiveness	None	n/a	0:100	10M
MAIN	Common setting	100 applications	Hierarchical JSON	50:50	10M
CAP-PROSE	Caption format	Same images as MAIN	Prose	50:50	10M
CAP-FLAT	Caption format	Same images as MAIN	Flat JSON	50:50	10M
CAP-MIX	Caption format	Same images as MAIN	Fixed prose/flat/hierarchical mix	50:50	10M
DIV-25	Application diversity	25 applications, $4\times$ per-app sampling	Hierarchical JSON	50:50	10M
DIV-50	Application diversity	50 applications, $2\times$ per-app sampling	Hierarchical JSON	50:50	10M
SHARE-25	Mixture ratio	100 applications	Hierarchical JSON	25:75	10M
SHARE-75	Mixture ratio	100 applications	Hierarchical JSON	75:25	10M
SHARE-100	Mixture ratio	100 applications	Hierarchical JSON	100:0	10M
SFT-100K	Adaptation baseline	100k curated examples	Hierarchical JSON	100:0	3 epochs

Table A2: **Configuration for the matched continued-training arms.** These settings apply to CTRL, MAIN, and the caption-representation, application-diversity, and mixture-ratio arms. SFT-100K follows its separately specified adaptation protocol.

Setting	Value
Initialization	Same frozen 2B reference checkpoint for every matched arm
Training endpoint	10,000,000 unique sample exposures in one complete pass without replacement; no fixed optimizer-step endpoint
Hardware	One node with eight NVIDIA H200 GPUs
Distributed training	FSDP2 with one replica and eight-way parameter sharding; parameters and gradient reductions use BF16
Numerical precision	BF16 mixed precision
Image resolution	UI renders retain their native approximately 1.05-megapixel resolution; background examples retain their native 512- or 1024-resolution tier
Image-token limit	At most 1,024 image tokens per example; native 512-resolution background examples use 256 image tokens
Text-token limit	At most 8,192 text tokens per example
Variable-length packing	Packed-token capacity of 16,896 tokens per rank and optimizer update
Batching	One packed item per rank, eight ranks, and gradient accumulation of one
Optimizer	HybridMuonAdamW with Muon and AdamW learning rates of 2×10^{-4}; Muon momentum 0.95, Nesterov momentum, five Newton–Schulz steps, and AdamW coefficients $(\beta_1, \beta_2) = (0.9, 0.95)$ with $\epsilon = 10^{-8}$
Weight decay	0 for both Muon and AdamW parameter groups
Learning-rate schedule	Constant learning rate without warm-up or decay
Gradient clipping	Global gradient-norm clipping at 1.0
EMA	Exponential moving average with decay 0.9999
Conditioning dropout	Classifier-free-guidance conditioning dropout probability 0.1

content. The generators also support changes to application templates, content filling, and rendering code, including new applications and screen types.

We include the frozen reference and a small supervised fine-tuning baseline (Table A1). The latter differs from the matched arms in initialization, UI volume, mixture, and epochs; it measures adaptation under that configuration and does not isolate a training-paradigm effect.

Continued-training configuration. All matched arms start from the same frozen 2B reference checkpoint and use the configuration in Table A2. Each consumes exactly ten million unique training examples in one pass without replacement. Variable-length packing accommodates caption length and image resolution differences within the same per-rank token capacity.

Table A3: **Inference configurations for external systems.** All systems receive the same structured prompt and generate one output per prompt. Commercial systems use the listed API model and request settings. Open-weight entries give repository identifiers.

System	Model or checkpoint version	Output size	Inference setting
Commercial APIs			
Ideogram 4	`ideogram-v4`	1664×2496	DEFAULT rendering speed
Seedream 5.0 Lite	`seedream-5-0-260128`	1664×2496	PNG
Nano Banana	`gemini-2.5-flash-image`	2:3, 1K	Official API defaults, one candidate
Nano Banana Pro	`gemini-3-pro-image`	2:3, 1K	Official API defaults, one candidate
Nano Banana 2	`gemini-3.1-flash-image`	2:3, 1K	Official API defaults, one candidate
GPT-Image-1	`gpt-image-1`	1024×1536	Medium quality, PNG
GPT-Image-1.5	`gpt-image-1.5-2025-12-16`	1024×1536	Medium quality, PNG
GPT-Image-2	`gpt-image-2-2026-04-21`	1024×1536	Medium quality, PNG
GPT-Image-2.5	`gpt-image-2.5-sunburst-2026-09-08`	1024×1536	Medium quality, PNG
Open-weight models			
NextStep-1.1	`stepfun-ai/NextStep-1.1`	512×768	28 steps, CFG 7.5
HiDream-O1-Image	`HiDream-ai/HiDream-O1-Image`	1024×1536	50 steps, CFG 5.0
LLaDA-Image	`inclusionAI/LLaDA-Image`	1024×1536	50 steps, CFG 5.0
BAGEL	`ByteDance-Seed/BAGEL-7B-MoT`	576×1024	50 steps, CFG 4.0
Emu3.5	`BAAI/Emu3.5-Image`	752×1360	5,120 image tokens, CFG 2.0
FLUX.2-dev	`black-forest-labs/FLUX.2-dev`	1024×1536	28 steps, guidance 4.0
SenseNova-U1.5	`sensenova/SenseNova-U1.5-8B-MoT`	1536×2720	50 steps, CFG 4.0

External-system inference. All 16 external systems receive the same frozen structured prompt and schema adapter, without access to the reference image, and generate one output per prompt. Each system uses its official inference pipeline and recommended settings, with a portrait output configuration where supported and no model-specific prompt rewriting or parameter tuning. Table A3 lists the model identifiers and inference settings.

Table A4: **Application-family split for the synthetic UI corpus.** The 100 training families are used to construct the synthetic UI training mixtures, while the ten held-out families are excluded from every training mixture and used for cross-application evaluation. The real-screen evaluation set is not part of this split.

Training applications (100)				
accuweather	comaps	ihg	perplexity	ticketmaster
adidas	costco	imdb	picsart	tiktok
adobe_reader	douban	instagram	pinterest	tripadvisor
airbnb	ebay	jetblue	popeyes	trulia
alaskaair	espn	jingdong	quizlet	tunein
alibaba	etsy	khanacademy	realtor	ulta
aliexpress	eventbrite	kohls	reddit	united
amazon	expedia	letterboxd	redfin	usps
americanair	fema	libby	reuters	vrbo
amtrak	flipboard	linkedin	sephora	walmart
bestbuy	fotmob	lowes	shein	waze
bilibili	gmaps	mcdonalds	slice	wholefoods
bleacherreport	grubhub	mercari	snapseed	wikipedia
booking	guardian	miniflutt	sofascore	xhs
burgerking	hilton	mnn	sportcaster	ximalaya
capcut	homedepot	mymta	spothero	yahoosports
chickfila	hopper	myshake	stubhub	yelp
chipotle	hotels	nba	taobao	zara
choicehotels	hoteltonight	newsbreak	target	zillow
cnn	hotpads	nhl	temu	zocdoc
Held-out applications (10)				
apartments	coolapk	coursera	delta	hm
meituan	redcross_firstaid	twitter	ups	youtube

E DATASET COMPOSITION AND STRUCTURED REPRESENTATIONS

The synthetic corpus is organized by application family. The split below defines which families contribute training screens and which are reserved for evaluation; the accompanying examples show the screens and their hierarchical captions.

E.1 TRAINING AND HELD-OUT APPLICATION SPLITS

We use 100 application families to construct the training mixtures and reserve 10 disjoint families for application-level generalization. Every screen associated with a held-out application is excluded from all training mixtures. The real-screen test set comprises 1,000 automatically captured mobile-application screens. Their structural GT annotations were first generated by Gemini and then manually verified. Table A4 lists the normalized application identifiers defining the synthetic training and held-out splits.

E.2 TRAINING-APPLICATION EXAMPLES

Figure A1 shows training screens from commerce, travel, food, media, navigation, sports, social, and utility applications. The gallery illustrates differences in page structure, visual style, content density, and navigation. These examples were selected for display and are not used for model selection or evaluation.

E.3 HIERARCHICAL STRUCTURED-CAPTION EXAMPLES

Hierarchical captions record each element's normalized bounding box, granularity, parent relation, and text when applicable. Figure A2 shows 3 screen–caption pairs with excerpts shortened for readability and omissions marked explicitly. Training and evaluation use the complete captions.

F DOMAIN-SPECIFIC UI GENERATION RESULTS

We report results separately for 300 synthetic screens from ten held-out applications, 1,000 captured real screens, and a 300-screen synthetic diagnostic set from application families represented during training. These subsets assess unseen synthetic applications, performance on real screens, and reconstruction of familiar application families, respectively. All models receive the complete structured JSON prompts.

For each metric, scores are first averaged within each application and then macro-averaged across applications. Application identity includes the source domain, so real and synthetic applications remain separate even when their normalized identifiers coincide. The held-out synthetic, real-screen, and seen-application tables contain ten, 130, and 100 application groups, respectively.

The open-weight systems and our models have complete coverage on these subsets. A small number of commercial-API requests did not return valid images, so their rows are aggregated over the available valid generations; coverage ranges from 299 to 300 of 300 held-out synthetic screens, from 976 to 994 of 1,000 real screens, and from 296 to 300 of 300 seen-application screens. The table captions report subset sizes; the number of valid generations can therefore be smaller for a commercial model.

The deterministic metrics measure reference-layout fidelity (*Ref.-Lay.*), regional and global visual similarity (*Region / Global*), required-text recall (*T-Rec.*), and coverage-aware text-position error (*T-Pos.*). Each unmatched required string contributes unit error to *T-Pos.*, allowing this metric to be ranked independently; lower values are better. The VLM metrics assess Prompt Adherence, Layout Adherence, and Visual Quality on a scale from 1 to 5.

F.1 SYNTHETIC SCREENS FROM HELD-OUT APPLICATIONS

Table A5 reports results for screens from ten applications, generated by the same synthetic-data process as the training corpus. These applications and all their screens are excluded from every training mixture.

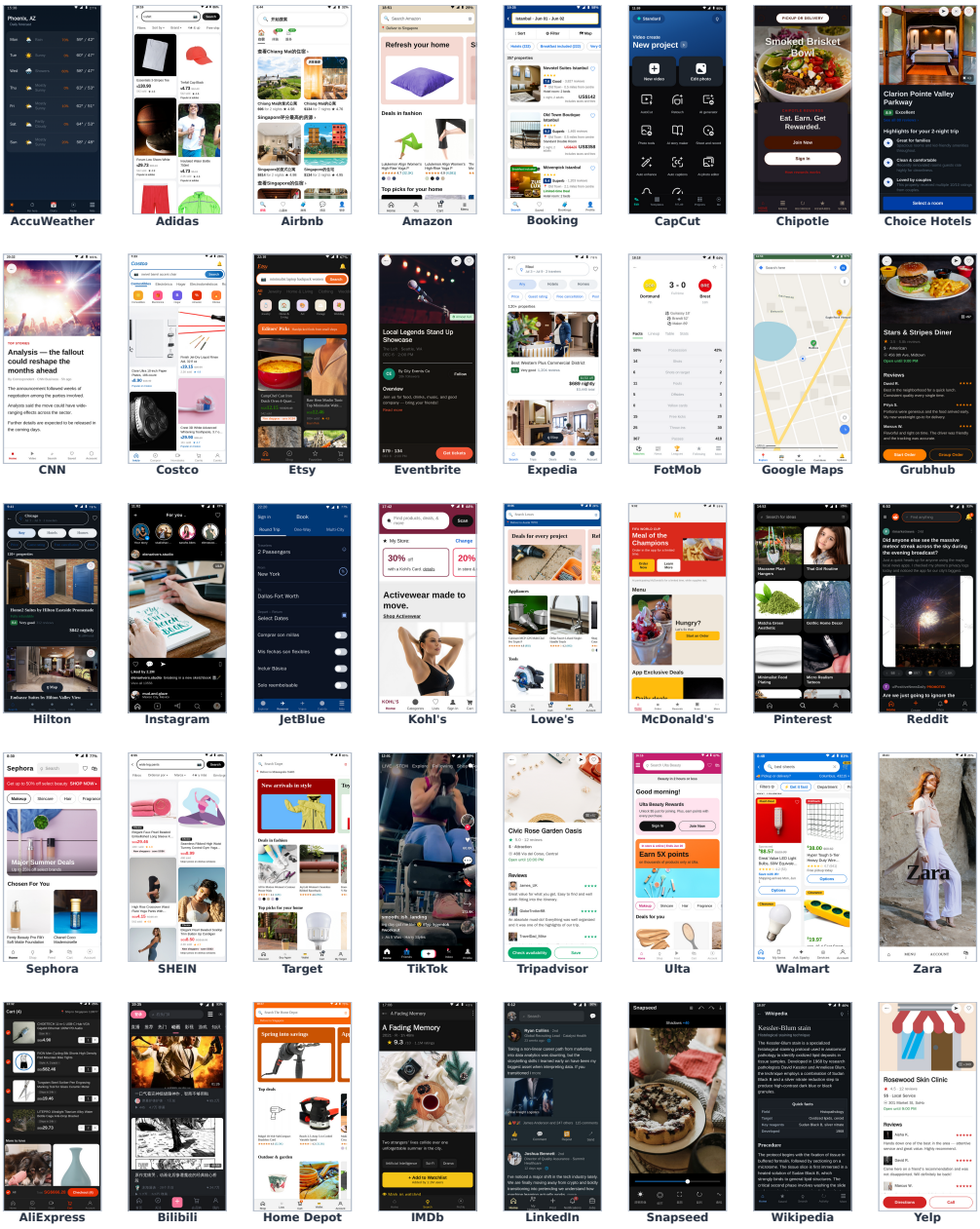


Figure A1: **Examples from the synthetic training corpus.** Screens from 40 of the 100 training application families, with one screen per family. Each screen is displayed in full without cropping or aspect-ratio distortion.

MAIN leads the evaluated systems in reference-layout fidelity and regional and global similarity. It also leads the open-weight systems in text recall, coverage-aware text positioning, and the three VLM metrics. The higher text-rendering and Visual Quality scores of the strongest commercial systems show where a performance gap remains.

CapCut

bbox overlays · labels omitted for clarity

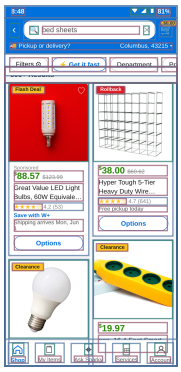


Hierarchical JSON caption

{
"high_level_description": "A portrait Android screenshot. An Android CapCut tools home in dark mode, with a top app bar and a bottom navigation bar. The main regions are the status bar, <status bar 1>, at the top; a hero banner, <hero banner 1>, at the top; a create cards grid, <create cards 1>, in the upper half; a tool tiles grid, <feed 1>, in the lower half; a bottom navigation bar, <bottom navigation 1>, at the bottom. Everything sits on a solid #0A0A0C app background.",
"style_description": ["aesthetics":"clean modern mobile UI, dark theme","color_palette":["#F2F3F5","#1C232C","#FFFFFF","#0A285A","#D6DBE1","#7A838D"],"lighting":"flat, screen-rendered, no shadows","medium":"ui_screenshot"],
"compositional_deconstruction": [
"background": "A solid #0A0A0C app screen background, flat and untextured, fully covered by the UI regions above it.",
"elements": [
{"bbox": [0, 0, 33, 1000], "color_palette": ["#0A0A0C", "#171719", "#6F6F71", "#09090B", "#F2F3F5"], "desc": "The Android status bar, with Wi-Fi, cellular-----", "label": "<status bar 1>", "parent": "ROOT", "text": null, "type": "medium", "z": 1},
{"bbox": [10, 849, 27, 885], "color_palette": ["#F2F3F5"], "desc": "The battery status icon.", "label": "<status battery icon 1>", "parent": "<status icons 1>", "text": null, "type": "fine", "z": null},
{"bbox": [33, 0, 295, 1000], "color_palette": ["#154380", "#215A9D", "#172D4A", "#174A8B"], "desc": "A hero banner, plus.", "label": "<hero banner 1>", "parent": "ROOT", "text": null, "type": "medium", "z": null},
{"bbox": [33, 44, 146, 956], "color_palette": ["#20599D", "#1D5497", "#154483", "#174B8B", "#FFFFFF"], "desc": "A top bar, plus a 114×114px dark blue icon---", "label": "<top bar 1>", "parent": "<hero banner 1>", "text": null, "type": "medium", "z": 1},
{"bbox": [46, 850, 95, 956], "color_palette": ["#0A285A", "#FFFFFF"], "desc": "A 114×114px dark blue icon button.", "label": "<navigation search 1>", "parent": "<top bar 1>", "text": null, "type": "fine", "z": null},
{"bbox": [49, 44, 92, 372], "color_palette": ["#FFFFFF", "#0A285A"], "desc": "A text label in an extrabold sans-serif white---", "label": "<plan badge 1>", "parent": "<top bar 1>", "text": "Standard", "type": "text", "z": null},
{"bbox": [249, 39, 391, 961], "color_palette": ["#1C232C", "#2C3847", "#A8AAAE", "#161C24", "#F2F3F5"], "desc": "A 2-column grid of 2 create cards.", "label": "<create cards 1>", "parent": "ROOT", "text": null, "type": "medium", "z": null},
{"bbox": [414, 0, 903, 1000], "color_palette": ["#0A0A0C", "#09090B", "#18181A", "#4C4C4E", "#F2F3F5"], "desc": "A masonry grid of 11 tool tiles.", "label": "<feed 1>", "parent": "ROOT", "text": null, "type": "coarse", "z": null},
...
]
}
}

Walmart

bbox overlays · labels omitted for clarity



Hierarchical JSON caption

{
"high_level_description": "A portrait Android screenshot. An Android Walmart search results in light mode for "bed sheets", showing a 2-column grid of 2 product cards, with a top app bar and a bottom navigation bar. The main regions are the status bar, <status bar 1>, at the top; a top bar, <top bar 1>, at the top; a chip row, <filter chips 1>, at the top; a feed, <feed 1>, filling the body; a bottom navigation bar, <bottom navigation 1>, at the bottom. Everything sits on a solid #F5F7F9 app background.",
"style_description": ["aesthetics":"clean modern mobile UI, light theme","color_palette":["#FFFFFF","#1A1A1A","#0071DC","#FFC220","#555555","#2A2A2A"],"lighting":"flat, screen-rendered, no shadows","medium":"ui_screenshot"],
"compositional_deconstruction": [
"background": "A solid #F5F7F9 app screen background, flat and untextured, fully covered by the UI regions above it.",
"elements": [
{"bbox": [0, 0, 33, 1000], "color_palette": ["#FFFFFF"], "desc": "The Android status bar, with Wi-Fi, cellular-----", "label": "<status bar 1>", "parent": "ROOT", "text": null, "type": "medium", "z": 1},
{"bbox": [6, 833, 24, 872], "color_palette": ["#FFFFFF"], "desc": "The battery status icon.", "label": "<status battery icon 1>", "parent": "<status icons 1>", "text": null, "type": "fine", "z": null},
{"bbox": [33, 0, 133, 1000], "color_palette": ["#1A1A1A", "#0071DC"], "desc": "A top bar, plus a search field and.", "label": "<top bar 1>", "parent": "ROOT", "text": null, "type": "medium", "z": 1},
{"bbox": [42, 110, 94, 874], "color_palette": ["#555555"], "desc": "A pill-shaped search bar with a placeholder and---", "label": "<search pill 1>", "parent": "<top bar 1>", "text": null, "type": "medium", "z": null},
{"bbox": [56, 806, 80, 838], "color_palette": ["#888888"], "desc": "A search field.", "label": "<search pill clear 1>", "parent": "<search pill 1>", "text": null, "type": "fine", "z": null},
{"bbox": [57, 219, 79, 782], "color_palette": ["#555555"], "desc": "A search bar with a placeholder in a sans-serif---", "label": "<search pill text 1>", "parent": "<search pill 1>", "text": "bed sheets", "type": "text", "z": null},
{"bbox": [51, 897, 86, 971], "color_palette": ["#FFFFFF"], "desc": "A cart.", "label": "<cart 1>", "parent": "<top bar 1>", "text": null, "type": "medium", "z": null},
{"bbox": [169, 0, 1000, 1000], "color_palette": ["#FFFFFF", "#F7F8F6", "#E2E3DA", "#D16D52", "#1A1A1A"], "desc": "A feed, plus a cut-off text label.", "label": "<feed 1>", "parent": "ROOT", "text": null, "type": "coarse", "z": null},
...
]
}
}

Target

bbox overlays · labels omitted for clarity



Hierarchical JSON caption

{
"high_level_description": "A portrait Android screenshot. An Android Target home feed in light mode, showing a 2-column grid of 2 product cards, with a top app bar and a bottom navigation bar. The main regions are the status bar, <status bar 1>, at the top; a top bar, <top bar 1>, at the top; a feed, <feed 1>, filling the body; a bottom navigation bar, <bottom navigation 1>, at the bottom. Everything sits on a solid #FFFFFF app background.",
"style_description": ["aesthetics":"clean modern mobile UI, light theme","color_palette":["#F1111","#EEEEEE","#990000","#FFFFFF","#F3F3F3","#D6E9D8"],"lighting":"flat, screen-rendered, no shadows","medium":"ui_screenshot"],
"compositional_deconstruction": [
"background": "A solid #FFFFFF app screen background, flat and untextured, fully covered by the UI regions above it.",
"elements": [
{"bbox": [0, 0, 27, 1000], "color_palette": ["#0F1111"], "desc": "The Android status bar, with Wi-Fi, cellular-----", "label": "<status bar 1>", "parent": "ROOT", "text": null, "type": "medium", "z": 1},
{"bbox": [4, 863, 19, 895], "color_palette": ["#0F1111"], "desc": "The battery status icon.", "label": "<status battery icon 1>", "parent": "<status icons 1>", "text": null, "type": "fine", "z": null},
{"bbox": [27, 0, 112, 1000], "color_palette": ["#0F1111", "#FFFFFF"], "desc": "A top bar containing a search bar and.", "label": "<top bar 1>", "parent": "ROOT", "text": null, "type": "medium", "z": 1},
{"bbox": [34, 24, 78, 976], "color_palette": ["#565959"], "desc": "A search bar with a placeholder in a serif dark---", "label": "<search pill 1>", "parent": "<top bar 1>", "text": "Search Target", "type": "text", "z": null},
{"bbox": [114, 19, 1000, 981], "color_palette": ["#7F463F", "#DCE0DA", "#C7AC96", "#FFFFFF", "#0F1111"], "desc": "A feed.", "label": "<feed 1>", "parent": "ROOT", "text": null, "type": "coarse", "z": null},
{"bbox": [126, 44, 386, 833], "color_palette": ["#FFFFFF"], "desc": "A hero card, with a photo of this is a product---", "label": "<hero card 1>", "parent": "<hero 1>", "text": null, "type": "medium", "z": null},
{"bbox": [200, 83, 368, 794], "color_palette": ["#68B8FD", "#CAC874", "#7D9E99", "#68B8FC"], "desc": "A large featured image showing this is a---", "label": "<hero image 1>", "parent": "<hero card 1>", "text": null, "type": "fine", "z": null},
{"bbox": [143, 896, 173, 981], "color_palette": ["#0F1111"], "desc": "A text label, mostly cut off at the screen edge---", "label": "<hero title 2>", "parent": "<hero card 2>", "text": null, "type": "text", "z": null},
...
]
}
}

Figure A2: **Examples of hierarchical structured captions.** Each row pairs a synthetic reference screen with its hierarchical JSON caption. The screen overlays visualize the annotated regions. Each displayed element retains the full field schema, including `bbox`, `color_palette`, `desc`, `label`, `parent`, `text`, `type`, and `z`. The `parent` field encodes containment, `type` distinguishes annotation granularities, and `bbox` specifies coordinates on the normalized 1000×1000 canvas. Ellipses denote omitted element records or abbreviated free-text values.

F.2 Real screens

On the 1,000-screen real subset (Table A6), Main leads the open-weight systems on all reported metrics. Its reference-layout fidelity is below that of GPT-Image-2.5 (0.191 versus 0.205). Its coverage-aware text-position error is 0.205, lower than that of every other open-weight baseline.

Table A5: **UI generation on held-out synthetic applications.** The benchmark subset contains 300 screens from ten applications excluded from all training mixtures. Values are application-macro averages. Bold and underlined values mark the best and second-best results within each model group, as determined before rounding.

	Structure	Visual fidelity		Text fidelity		VLM-based evaluation		
Model	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Commercial APIs								
Ideogram 4	0.079	0.642	0.870	0.851	0.240	3.64	3.22	4.31
Seedream 5.0 Lite	0.111	0.630	0.874	0.885	0.224	3.53	3.16	3.87
Nano Banana	0.123	0.667	0.876	0.749	0.314	3.54	3.51	3.63
Nano Banana Pro	0.114	0.634	0.871	0.863	0.216	3.76	3.50	3.66
Nano Banana 2	0.142	0.643	0.873	0.831	0.235	3.79	4.02	2.97
GPT-Image-1	0.094	0.595	0.856	0.427	0.627	2.71	1.81	3.46
GPT-Image-1.5	0.114	0.631	0.881	0.669	0.403	3.61	3.11	4.19
GPT-Image-2	0.170	0.684	0.895	**0.945**	**0.116**	4.76	4.59	4.89
GPT-Image-2.5	**0.188**	**0.712**	**0.897**	0.924	0.127	**4.79**	**4.81**	**4.90**
Open-weight models								
NextStep-1.1	0.100	0.555	0.637	0.021	0.984	1.01	1.01	1.00
HiDream-O1-Image	0.007	0.577	0.636	0.000	1.000	1.02	1.05	1.04
LLaDA-Image	0.021	0.564	0.639	0.017	0.989	1.03	1.03	1.25
BAGEL	0.097	0.568	0.718	0.000	1.000	1.05	1.03	1.49
Emu3.5	0.104	0.606	0.839	0.526	0.576	2.18	1.94	2.20
FLUX.2-dev	0.117	0.633	0.843	0.250	0.811	2.08	1.69	2.36
SenseNova-U1.5	0.108	0.631	0.848	0.739	0.375	2.59	2.71	2.30
Ours (MAIN)	**0.368**	**0.802**	**0.906**	**0.772**	**0.262**	**3.96**	**4.41**	**3.35**

Compared with the held-out synthetic subset, MAIN's reference-layout fidelity falls from 0.368 to 0.191 and Layout Adherence from 4.41 to 3.79. The subsets differ in both visual domain and application composition, so these changes cannot be attributed to either factor alone. GPT-Image-2 and GPT-Image-2.5 retain higher text-rendering and Visual Quality scores on the real screens.

F.3 SEEN-APPLICATION DIAGNOSTIC EVALUATION

This diagnostic subset contains 300 synthetic screens from application families represented in the training pool. The screens may share structure, style, and recurring visual patterns with the training data. We use them to examine reconstruction within familiar application families and exclude them from headline model ranking and model selection.

Relative to the held-out synthetic subset, MAIN increases from 0.368 to 0.515 in reference-layout fidelity, from 0.802 to 0.879 in regional similarity, and from 4.41 to 4.84 in Layout Adherence. Its coverage-aware text-position error decreases from 0.262 to 0.118, while Prompt Adherence and Visual Quality increase from 3.96 and 3.35 to 4.55 and 3.75. Performance is better on familiar application families, although this comparison does not establish which shared properties account for the difference.

Qualitative diagnostic examples. Figure A3 shows examples from the diagnostic results in Table A7, including event listings and booking, food ordering, and educational study interfaces. In these examples, REF and the background-only CTRL recover the page type but distort text and merge neighboring components. MAIN preserves more of the repeated list items, section boundaries, action controls, and bottom navigation, with fewer typography and local rendering errors.

G COMPLETE DATA AND TRAINING ABLATIONS

The tables below expand the ablations in Section 6.3 to all eight metrics and report the VLM results across prompt formats. Unless stated otherwise, evaluation uses the 1,300-prompt primary

Table A6: **UI generation on real screens.** The benchmark subset contains 1,000 captured real screens. Values are application-macro averages. Bold and underlined values mark the best and second-best results within each model group, as determined before rounding.

	Structure	Visual fidelity		Text fidelity		VLM-based evaluation		
Model	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Commercial APIs								
Ideogram 4	0.091	0.647	0.876	0.910	0.190	3.94	3.39	4.48
Seedream 5.0 Lite	0.078	0.615	0.857	0.883	0.310	2.96	2.37	3.44
Nano Banana	0.134	0.640	0.880	0.867	0.209	3.77	3.47	3.85
Nano Banana Pro	0.122	0.619	0.868	0.873	0.218	3.69	3.29	3.48
Nano Banana 2	0.148	0.619	0.872	0.858	0.218	3.61	3.66	2.85
GPT-Image-1	0.090	0.602	0.853	0.608	0.471	3.00	2.19	3.93
GPT-Image-1.5	0.114	0.618	0.879	0.818	0.289	3.86	3.18	4.39
GPT-Image-2	0.187	0.686	0.903	**0.950**	0.108	4.88	4.69	**4.95**
GPT-Image-2.5	**0.205**	**0.697**	**0.904**	0.945	**0.104**	**4.90**	**4.69**	4.94
Open-weight models								
NextStep-1.1	0.111	0.551	0.634	0.034	0.974	1.02	1.03	1.01
HiDream-O1-Image	0.006	0.571	0.641	0.003	0.998	1.08	1.08	1.11
LLaDA-Image	0.058	0.565	0.682	0.070	0.951	1.10	1.13	1.21
BAGEL	0.093	0.562	0.721	0.002	0.998	1.12	1.10	1.55
Emu3.5	0.111	0.595	0.839	0.564	0.537	2.18	1.99	2.32
FLUX.2-dev	0.111	0.620	0.849	0.378	0.699	2.25	1.77	2.56
SenseNova-U1.5	0.101	0.616	0.856	0.787	0.361	2.57	2.40	2.30
Ours (MAIN)	**0.191**	**0.670**	**0.865**	**0.817**	**0.205**	**3.40**	**3.79**	**2.64**

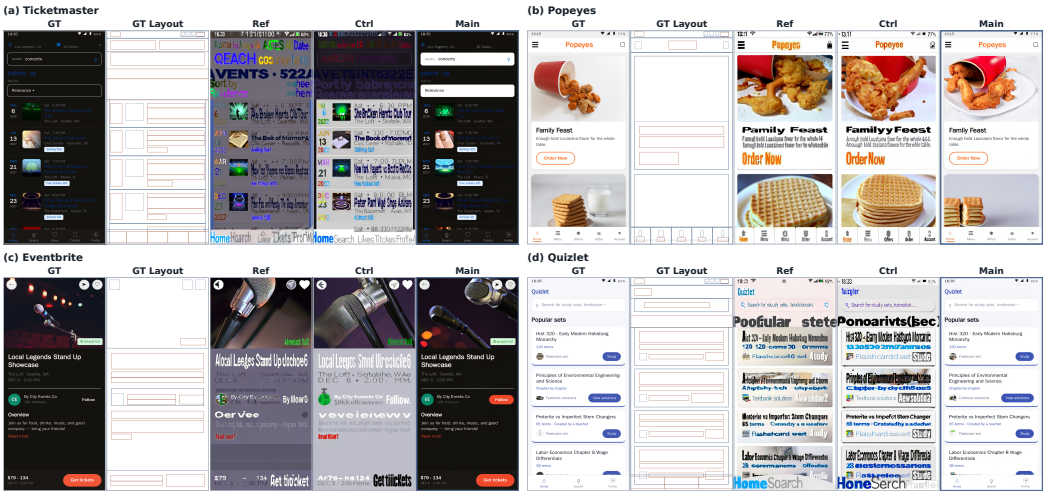


Figure A3: **Seen-application diagnostic examples.** Each panel shows the GT reference, its annotated layout, the frozen reference checkpoint (REF), the background-only continued-training control (CTRL), and MAIN. The examples cover event discovery, food ordering, product browsing, and educational study interfaces. Their application families are represented during training.

UI benchmark: scores are averaged within each source-aware application group and then macro-averaged over 130 real and ten held-out synthetic application groups. Real and synthetic applications remain distinct even when their normalized identifiers coincide.

Table A7: **Seen-application diagnostic results.** The benchmark subset contains 300 synthetic screens from application families represented during training. Values are application-macro averages. Bold and underlined values identify the best and second-best results within each model group. These rankings are diagnostic and are excluded from headline model comparison and model selection.

	Structure	Visual fidelity		Text fidelity		VLM-based evaluation		
Model	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Commercial APIs								
Ideogram 4	0.094	0.641	0.872	0.891	0.195	3.76	3.29	4.22
Seedream 5.0 Lite	0.107	0.617	0.870	0.897	0.233	3.44	3.02	3.69
Nano Banana	0.135	0.659	0.873	0.845	0.220	3.82	3.70	3.68
Nano Banana Pro	0.115	0.630	0.867	0.880	0.205	3.74	3.43	3.51
Nano Banana 2	0.145	0.640	0.873	0.857	0.210	3.75	3.85	3.07
GPT-Image-1	0.089	0.594	0.852	0.482	0.584	2.89	1.91	3.60
GPT-Image-1.5	0.112	0.617	0.876	0.793	0.304	3.69	3.06	4.27
GPT-Image-2	<u>0.182</u>	<u>0.683</u>	<u>0.889</u>	**0.937**	<u>0.113</u>	<u>4.75</u>	<u>4.60</u>	<u>4.76</u>
GPT-Image-2.5	**0.192**	**0.703**	**0.893**	<u>0.936</u>	**0.109**	**4.82**	**4.74**	**4.84**
Open-weight models								
NextStep-1.1	0.102	0.553	0.637	0.029	0.978	1.02	1.03	1.01
HiDream-O1-Image	0.007	0.574	0.630	0.000	1.000	1.03	1.04	1.08
LLaDA-Image	0.033	0.564	0.659	0.022	0.985	1.10	1.06	1.47
BAGEL	0.088	0.567	0.704	0.000	1.000	1.12	1.04	1.42
Emu3.5	0.103	0.601	0.837	0.575	0.538	2.17	1.88	2.23
FLUX.2-dev	0.117	0.627	0.836	0.309	0.769	2.20	1.75	2.20
SenseNova-U1.5	<u>0.119</u>	<u>0.629</u>	<u>0.851</u>	<u>0.812</u>	<u>0.320</u>	<u>2.80</u>	<u>2.71</u>	<u>2.50</u>
Ours (MAIN)	**0.515**	**0.879**	**0.940**	**0.913**	**0.118**	**4.55**	**4.84**	**3.75**

Table A8: **Caption-representation ablation.** All arms use identical UI images, a 50:50 UI-to-background mixture, and the same continued-training budget. Evaluation uses hierarchical JSON prompts on the 1,300-prompt primary UI benchmark. Values are source-aware application-macro averages. Bold and underlined values denote the best and second-best results, respectively, as determined before rounding.

	Structure	Visual fidelity		Text fidelity		VLM evaluation		
Training caption	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Prose	0.185	0.651	0.801	0.515	0.493	2.82	3.29	1.48
Flat JSON	<u>0.203</u>	0.656	0.818	0.598	<u>0.414</u>	2.92	3.39	1.84
Mixed formats	0.199	<u>0.680</u>	<u>0.853</u>	<u>0.607</u>	0.414	<u>3.29</u>	<u>3.54</u>	**2.80**
Hierarchical JSON (MAIN)	**0.203**	**0.680**	**0.868**	**0.813**	**0.210**	**3.44**	**3.84**	<u>2.69</u>

G.1 CAPTION REPRESENTATION

The caption arms use identical UI images, training budgets, and UI-to-background proportions. They differ in the captions paired with each image, including both content and serialization. With the hierarchical JSON evaluation prompts in Table A8, hierarchical training leads seven of the eight metrics, including both text metrics and both reference-assisted VLM metrics. Mixed-format training has the highest Visual Quality. These results compare the complete caption recipes under hierarchical evaluation prompts; they do not isolate the effect of hierarchy.

G.2 CROSS-FORMAT PROMPT ROBUSTNESS

To assess sensitivity to the inference format, we evaluate each of the four caption-training arms under four prompt views: Prose, Flat JSON, Mixed, and Hierarchical JSON. The Prose, Flat JSON, and

Table A9: **Off-diagonal prompt-view averages on held-out applications.** Each value averages the three off-diagonal entries in a training-arm row. Mixed training includes all three concrete prompt formats. Bold values denote the best result.

Training caption	Prompt↑	Layout↑	Quality↑
Prose	2.51	3.09	1.44
Flat JSON	2.44	2.67	2.23
Mixed formats	**3.38**	**3.60**	**3.52**
Hierarchical JSON	2.49	2.76	2.25

Hierarchical JSON views use the corresponding schema for every screen. The Mixed view selects one of these three concrete schemas for each screen using a fixed assignment.

For the Mixed view, screens are placed in a frozen order and assigned a prompt schema cyclically according to their zero-based benchmark index. Indices congruent to 0, 1, and 2 modulo 3 use Prose, Flat JSON, and Hierarchical JSON, respectively. The selected prompt is copied from the corresponding concrete prompt view and is neither concatenated with the other representations nor converted into a separate mixed schema. Across the 600 screen identities, the Mixed view therefore contains exactly 200 prompts in each concrete schema.

Combining four caption-training arms with four evaluation prompt views produces $4 \times 4 = 16$ training-arm and evaluation-view combinations. Each combination contains the same 600 screen identities: 300 screens from ten held-out applications and 300 screens from application families represented during training. The experiment therefore contains 9,600 generated candidates.

This cross-format sweep was generated independently from the primary benchmark evaluation. For the nominally identical hierarchical-training and hierarchical-prompt condition, the two evaluations use the same checkpoint, screen identities, prompt contents, sampling steps, guidance scale, and output resolution. However, the cross-format generation uses a prediction batch size of 4 and a separate manifest ordering, whereas the primary benchmark generation uses a prediction batch size of 1. Because sampling randomness is assigned within each run rather than deterministically keyed to sample identity, the two evaluations produce different candidate images. We therefore use the cross-format scores only for comparisons within the 4×4 matrix; the domain-specific benchmark tables provide the canonical absolute scores for MAIN. All comparisons and conclusions drawn from this sweep are therefore based on differences within the same cross-format evaluation, rather than on absolute score differences between independently generated runs.

Each candidate is scored using the Prompt and Layout protocol and, in a separate call, the Visual Quality protocol. For a training arm f and metric m, the mean over the three off-diagonal evaluation views is

$$\overline{S}_m^{\text{off}}(f) = \frac{1}{3} \sum_{g \neq f} S_m(f \to g), \tag{7}$$

where $S_m(f \to g)$ is the score obtained from training arm f under evaluation prompt view g.

G.2.1 HELD-OUT APPLICATIONS

Figure 4a reports the matrices for 300 screens from ten held-out applications. Among the diagonal entries, Hierarchical JSON has the highest Prompt and Layout scores, while Flat JSON has slightly higher Quality. Table A9 averages the three off-diagonal entries in each training-arm row. Mixed-format training leads these averages, with 3.38 for Prompt, 3.60 for Layout, and 3.52 for Quality. The results support hierarchical training when inference uses hierarchical JSON and mixed training when inference uses the tested range of formats. Mixed training has already encountered all three concrete formats, so its scores do not establish transfer to unseen formats.

G.2.2 SEEN-APPLICATION DIAGNOSTIC

Table A10 reports the same experiment on the 300-screen seen-application diagnostic subset. Scores are generally higher than on the held-out applications, particularly for structured inference prompts.

Table A10: **Cross-format VLM evaluation on the seen-application diagnostic set.** Rows indicate the caption-training arm and columns indicate the evaluation prompt view. Each cell reports Prompt / Layout / Quality. Bold diagonal cells pair each training arm with its corresponding evaluation view. This subset is excluded from headline comparisons and model selection.

	Inference prompt format			
Training caption	**Prose**	**Flat JSON**	**Mixed**	**Hierarchical JSON**
Prose	**3.69/3.75/3.53**	2.50/3.19/1.26	2.92/3.40/2.01	2.66/3.29/1.30
Flat JSON	1.01/1.04/1.15	**4.59/4.86/4.46**	3.13/3.38/2.92	3.77/4.14/3.13
Mixed formats	2.93/3.06/3.08	4.43/4.80/4.26	**3.93/4.21/3.89**	4.52/4.83/4.32
Hierarchical JSON	1.01/1.04/1.11	3.73/4.26/3.00	3.09/3.41/2.77	**4.51/4.80/4.08**

Mixed-format training has the highest off-diagonal averages: 3.96, 4.23, and 3.89 for Prompt, Layout, and Quality. The corresponding averages are 2.69/3.29/1.52 for prose, 2.63/2.85/2.40 for flat JSON, and 2.61/2.90/2.29 for hierarchical JSON. Because these screens come from application families represented during training, these results are diagnostic only and are excluded from headline model ranking and model selection.

G.3 UI PROPORTION IN THE TRAINING MIXTURE

Table A11 reports all metrics for the mixture ratios in Figure 4b. Table A12 gives the corresponding results on 300 synthetic screens from application families represented during training. The primary benchmark evaluates the experimental configurations; it was not used for hyperparameter or operating-point selection. The seen-application results serve only as a diagnostic.

Table A11: **UI proportion on the primary UI benchmark.** All trainable arms use the same total budget, UI pool, and hierarchical caption format. The frozen reference is included for context. Values are source-aware application-macro averages over 130 real and ten held-out synthetic application groups. Bold and underlined values denote the best and second-best results among the mixture-ratio arms, respectively, as determined before rounding.

	Structure	**Visual fidelity**		**Text fidelity**		**VLM evaluation**		
UI:BG	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Frozen REF	0.192	0.648	0.829	0.245	0.759	2.64	3.11	1.52
0:100 (CTRL)	0.171	0.649	0.818	0.465	0.539	2.69	3.24	1.58
25:75	<u>0.199</u>	**0.683**	0.862	**0.825**	**0.200**	**3.55**	**3.99**	2.67
50:50 (MAIN)	**0.203**	<u>0.680</u>	**0.868**	<u>0.813</u>	<u>0.210</u>	3.44	3.84	2.69
75:25	0.186	0.680	<u>0.865</u>	0.801	0.224	<u>3.50</u>	<u>3.84</u>	<u>2.76</u>
100:0	0.171	0.678	0.862	0.757	0.268	3.42	3.66	**2.79**

Primary UI benchmark. The 25:75 mixture leads regional similarity, text recall, Prompt Adherence, and Layout Adherence, and has the lowest coverage-aware text-position error. The 50:50 mixture leads reference-layout fidelity and global similarity. Above 50% UI data, Visual Quality improves modestly, while structure and text scores do not improve. The choice between 25:75 and 50:50 therefore depends on which of these metrics matters most.

Seen-application diagnostic. On familiar application families, the 75:25 mixture leads regional similarity, text recall, coverage-aware text positioning, and all three VLM metrics; the 100:0 mixture leads reference-layout fidelity and global similarity. Although 75:25 leads six of the eight diagnostic metrics, adherence and text scores on the primary benchmark favor a smaller UI share. We report the seen subset only as a diagnostic because its application families are represented during training.

Table A12: **UI proportion on the seen-application diagnostic set.** Results use 300 synthetic screens from application families represented during training. Bold and underlined values denote the best and second-best results among the mixture-ratio arms, respectively. These rankings are diagnostic and are excluded from headline model comparison and model selection.

	Structure	Visual fidelity		Text fidelity		VLM evaluation		
UI:BG	**Ref.-Lay.↑**	**Region↑**	**Global↑**	**T-Rec.↑**	**T-Pos.↓**	**Prompt↑**	**Layout↑**	**Quality↑**
Frozen REF	0.203	0.666	0.824	0.209	0.793	2.77	3.18	1.50
0:100 (CTRL)	0.175	0.665	0.814	0.379	0.625	2.71	3.31	1.54
25:75	0.450	0.867	0.929	<u>0.915</u>	<u>0.115</u>	<u>4.62</u>	<u>4.87</u>	<u>3.95</u>
50:50 (MAIN)	0.515	0.879	0.940	0.913	0.118	4.55	4.84	3.75
75:25	<u>0.530</u>	**0.891**	<u>0.946</u>	**0.924**	**0.107**	**4.70**	**4.91**	**4.04**
100:0	**0.553**	<u>0.890</u>	**0.948**	0.912	0.119	4.57	4.84	3.82

H VLM JUDGE PROMPTS AND IMPLEMENTATION DETAILS

The prompts and response schemas below implement the VLM metrics defined in Section 5.2. We use two independent Gemini 3.6 Flash calls. The first evaluates Prompt Adherence and the GT-defined layout constraints using the actual inference prompt supplied to the generator, the GT reference, the generated screen, and paired annotation panels. The second evaluates Visual Quality from the raw generated screen alone, without the prompt, reference image, or GT annotations.

Prompt representation supplied to the judge. For Prompt Adherence, the judge receives the same prompt representation that was supplied to the generator for that evaluation view. In the primary benchmark, this is the complete hierarchical JSON prompt. In the cross-format evaluation, a Prose example remains prose, while Flat JSON and Hierarchical JSON examples retain their respective JSON schemas. JSON prompts may be parsed and serialized again for transport, but their semantic content and schema are preserved; they are not converted into hierarchical JSON. For the Mixed view, the judge receives the concrete Prose, Flat JSON, or Hierarchical JSON prompt selected for that screen by the fixed assignment described in §G.2. The layout targets are constructed separately from the canonical hierarchical annotation. Consequently, the evaluated regions and spatial relations remain identical across the four prompt views for a given screen, even though the prompt representation supplied to the generator and Prompt Adherence judge changes.

H.1 PROMPT AND LAYOUT EVALUATION

Prompt A: reference-assisted Prompt and Layout evaluation

You are a strict, consistent evaluator of generated Android UI screenshots. Evaluate every anonymous candidate independently. Never compare candidates or infer model identity.

1. Prompt Adherence (integer 1–5).
Determine whether the raw generated screen realizes the page purpose and the requested content and components in the generation prompt. Judge what is shown, rather than exact placement or rendering polish. Exact OCR spelling is evaluated separately.

- 1: failed or unrelated;
- 2: weak partial match;
- 3: recognizable, but with material omissions;
- 4: all core content is present, with only minor deviations;
- 5: complete fulfillment.

2. Layout constraints.
The numbered region and relation panels jointly cover every valid GT annotation region and every derived non-redundant spatial relation. The GT and generated-screen panels use identical

target regions. Colored rectangles are query windows, not evidence that a component exists or is correctly placed. Ignore the rectangle lines themselves and judge the visible UI content.
For every requested region, return:

- 2: the described component is unmistakably present, semantically correct, and its visible extent closely agrees with the target region;
- 1: the component is present but incomplete, displaced, or only approximately aligned;
- 0: the component is absent, incorrect, or replaced by unrelated generic content.

For every requested relation, return:

- 2: the relation is correctly satisfied;
- 1: the relation is approximate or ambiguous;
- 0: the relation is violated or either endpoint component is absent.

Judge relations from the visible components rather than from the geometry of the drawn query windows. Do not predict new boxes for the generated screen.
Differences caused solely by supported canvas resolution or padding are not errors. Cropping, distortion, missing components, displaced content, and altered spatial relationships remain errors. Return every candidate identifier and every requested region and relation identifier exactly once.

Sample-specific inputs

```
Generation prompt: <ACTUAL-INFERENCE-PROMPT>
Target regions: <REGION-ID, DESCRIPTION, GT-BBOX>
Region-panel index: <PANEL-ID, REGION-IDS>
Required spatial relations: <RELATION-ID, ENDPOINTS,
DESCRIPTION>
Relation-panel index: <PANEL-ID, RELATION-IDS>
Candidate identifiers: <ANONYMOUS-CANDIDATE-IDS>
```

The multimodal content is supplied in the following order:

1. the textual rubric and sample-specific structured information;
2. the raw GT reference;
3. all GT region and relation panels;
4. the raw generated screen for each anonymous candidate;
5. the corresponding generated-screen region and relation panels.

For each generated screen, the response contains an anonymous candidate identifier, an integer Prompt Adherence score, one status for every requested region, one status for every requested relation, and a short explanation. Its logical structure is:

Structured response for Prompt and Layout evaluation

```
{
  "candidates": [
    {
      "candidate_id": "<ID>",
      "prompt_adherence": <integer 1--5>,
      "region_checks": [
        {"id": "E1", "status": <0, 1, or 2>}, ...
      ],
      "relation_checks": [
        {"id": "R1", "status": <0, 1, or 2>}, ...
      ],
      "reason": "<short explanation>"
    }
```

```
  ]
}
```

Every expected identifier is validated in code. A response is rejected and retried if a candidate, region, or relation is missing, duplicated, or assigned an out-of-range status. Before aggregation, each relation status is capped by the statuses of its two endpoint regions. Prompt Adherence and Layout Adherence are then computed as defined in Section 5.2.

H.2 Visual Quality evaluation

The reported Visual Quality score is obtained from a separate request that receives only the raw generated screen. The request does not include the structured prompt, GT reference, annotation panels, bounding boxes, or spatial relations.

Prompt B: Visual Quality evaluation

You are a strict, consistent evaluator of generated Android UI screenshots.
For each anonymous generated screen, assess visual quality from the raw image alone. You do not receive a generation prompt, reference image, bounding boxes, or spatial relations. Do not infer missing requirements or judge similarity to an unseen target. Judge each screen independently, never compare screens, and never infer model identity.
Return two integer ratings from 1 to 5.

1. Global coherence.
Evaluate overall composition, visual hierarchy, spacing, balance, density, color harmony, and consistency of the design system.

- 1: visually incoherent or unusable;
- 2: major composition or consistency problems;
- 3: recognizable and acceptable, but materially flawed;
- 4: coherent and polished, with only minor defects;
- 5: exceptionally coherent, with no salient global defect.

2. Local rendering.
Evaluate the legibility and rendering quality of typography, icons, controls, imagery, borders, and fine details, including blur, malformed glyphs, clipping, overlap, and generation artifacts.

- 1: severe corruption or pervasive illegibility;
- 2: major artifacts or malformed local content;
- 3: usable, but with noticeable local defects;
- 4: crisp and polished, with only minor defects;
- 5: consistently crisp, with no salient local defect.

Do not reward semantic content, prompt compliance, or reference-layout similarity. Return every candidate identifier exactly once and provide a short reason grounded only in visible rendering evidence.

`Candidate identifiers: <ANONYMOUS-CANDIDATE-IDS>`

For each generated screen, the response contains separate ratings for global coherence and local rendering:

Structured response for Visual Quality evaluation

```
{
  "candidates": [
    {
      "candidate_id": "<ID>",
```

```
      "global_coherence":  <integer 1--5>,
      "local_rendering":  <integer 1--5>,
      "reason":  "<short explanation>"
    }
  ]
}
```

For each generated screen, Visual Quality is computed as the arithmetic mean of global coherence and local rendering.

H.3 Inference and aggregation settings

Both evaluations use Gemini 3.6 Flash with temperature zero and schema-constrained JSON output. The judge is explicitly instructed to evaluate each generated screen independently.

Scores are first computed per sample. They are then averaged within each application and finally macro-averaged across applications, following the same aggregation protocol used for the deterministic metrics. Requests are retried when the returned candidate identifiers, region identifiers, relation identifiers, or score ranges fail validation.