MULTITEXTEDIT: Benchmarking Cross-Lingual Degradation in Text-in-Image Editing

Anonymous ACL submission

Abstract

Text-in-image editing has become a key capability for visual content creation, yet existing benchmarks remain overwhelmingly English-centric and often conflate visual plausibility with semantic correctness. We introduce MULTITEXTEDIT, a controlled benchmark of 3,600 instances spanning 12 typologically diverse languages, 5 visual domains, and 6 editing operations. Language variants of each instance share a common visual base and are paired with a human-edited reference and region masks, isolating the language variable for cross-lingual comparison. To capture script-level errors that coarse text-matching metrics miss, such as missing diacritics, reversed RTL order, and mixed-script renderings, we introduce a language fidelity (LSF) metric scored by a two-stage LVM protocol that first traces the edited target text and then judges it in isolation, reaching a quadratic-weighted κ of 0.819 against the mean of two human raters. Evaluating 12 open-source and proprietary systems with LSF alongside standard semantic and mask-aware pixel metrics, we find pronounced cross-lingual degradation for every model, largest on Hebrew and Arabic and smallest on Dutch and Spanish, and concentrated in text accuracy and script fidelity rather than in coarse structural dimensions. We also uncover a pervasive semantic and pixel mismatch, where outputs preserve global layout and background fidelity yet distort script-specific forms.

1 Introduction

Text-in-image editing aims to modify text rendered in an image while preserving the surrounding visual content. Recent multimodal and diffusion-based systems can replace, insert, delete, recolor, and reposition in-image text with increasingly strong visual quality (Tuo et al., 2024b,a; Wang et al., 2024; Lan et al., 2025). This capability is important for graphic design, advertisement localization, event promotion, and product marketing, where the

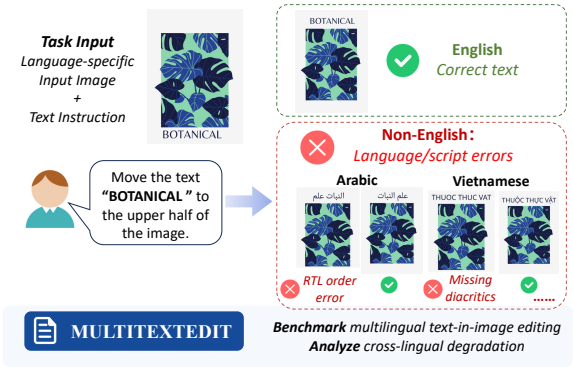

Figure 1: Multilingual text-in-image editing under the same edit intent, highlighting language/script fidelity errors and cross-lingual degradation beyond English-centric evaluation.

same visual asset often needs to be adapted across languages. Unlike generic image editing, text in images is an editable object that carries both visual form and linguistic meaning. This dual nature becomes especially challenging in multilingual settings, where writing systems differ in typography, character composition, spacing conventions, and writing direction.

Multilingual evaluation is therefore essential. Real-world content is edited not only in English, but across languages that differ widely in script and resource availability, especially in non-Latin and lower-resource settings. Yet current evaluation remains largely English-centric (Gui et al., 2025; Wu et al., 2025c). Beyond this coverage issue, existing evaluation protocols often conflate visual plausibility with semantic correctness. For text-in-image editing, an output may look locally consistent while still rendering the wrong word, failing to follow the instruction, or distorting script-specific characters. As a result, we still lack a controlled way to measure how strongly state-of-the-art multimodal models degrade across languages.

To address these issues, we introduce MULTITEXTEDIT (**Multi**lingual **Text Edit**ing Bench-

mark), a benchmark for analyzing multilingual degradation in text-in-image editing. MULTITEXTEDIT contains 3,600 instances created from 300 base images and their 12 language variants, covering 5 visual domains and 6 editing operations. Each instance includes a source image, an editing prompt, a human-edited ground-truth image, an operation label, a domain label, and a mask for the edited region. The benchmark spans Latin, CJK, Arabic, Hebrew, Cyrillic, and Bengali scripts, enabling controlled analysis across diverse writing systems and language-resource conditions. Because different language versions share the same underlying visual content and edit intent, the benchmark supports controlled cross-lingual comparison rather than loosely matched evaluation.

We further adopt a dual-track evaluation protocol that combines semantic assessment with pixel-level fidelity. The semantic track measures instruction following, text accuracy, visual consistency, and layout preservation, while the pixel track uses standard similarity metrics to quantify closeness to the reference image. This design lets us study not only overall performance, but also the *semantic-pixel mismatch* that arises when visually plausible edits are semantically wrong. Our experiments on several strong multimodal models reveal substantial performance degradation outside English together with recurring mismatches between semantic success and pixel similarity.

Our contributions are threefold: (1) we introduce MULTITEXTEDIT, a controlled multilingual benchmark for text-in-image editing across 12 languages, 5 domains, and 6 editing operations; (2) we introduce a language/script fidelity (LSF) metric with a two-stage tracing-and-scoring protocol that diagnoses script-level errors current metrics miss, validated by a $\kappa = 0.819$ agreement with the mean of two human raters; (3) we provide a systematic empirical study across 12 models revealing pronounced cross-lingual degradation and a recurring semantic–pixel mismatch.

2 Related Work

Text-in-Image Editing Models. Diffusion-based and multimodal methods have rapidly advanced controllable text rendering and editing, including the TextDiffuser series (Chen et al., 2023, 2024), the AnyText series (Tuo et al., 2024b,a), and more recent work targeting disentangled editing, recognition-aware consistency, glyph fidelity, and low-resource language support (Zhang et al., 2024; Fang et al., 2025; Wang et al., 2025; Park et al., 2025). While these models substantially improve visual quality and controllability, they do not by themselves characterize how performance changes across languages, scripts, and resource conditions, motivating a benchmark focused on multilingual robustness.

Benchmarks for Text Editing and Multilingual Visual Text Tasks. Existing benchmarks evaluate text-related editing or visual text processing but leave the multilingual degradation question open. TextEditBench (Gui et al., 2025) targets text-in-image editing without controlled cross-lingual comparison; VTPBench (Wu et al., 2025c) covers a broad range of visual text tasks; ScenePair (Zeng et al., 2024) provides paired supervision for scene text editing; and IMTBench (Li et al., 2026) studies in-image machine translation rather than general editing operations. Scene text recognition research has also long highlighted script-dependent difficulty (Shi et al., 2016; Baek et al., 2019; Bautista and Atienza, 2022). In contrast, MULTITEXTEDIT is built specifically to compare the same editing intent across 12 languages under shared visual contexts, enabling direct analysis of cross-lingual performance drop.

Evaluation of Text-in-Image Editing. Pixel and perceptual metrics such as SSIM and LPIPS measure structural and perceptual similarity (Wang et al., 2004; Zhang et al., 2018) but cannot determine whether the edited text satisfies the instruction. Recent image editing evaluation therefore emphasizes instruction faithfulness alongside visual fidelity (Qian et al., 2025; Wu et al., 2025d), and LLM-as-a-judge paradigms (Zheng et al., 2023) provide a scalable way to score open-ended generation. Our framework builds on these insights but tailors them to multilingual text-in-image editing, where correctness must be judged at both the semantic and visual/layout level, and where cross-language comparison is a central goal.

3 MULTITEXTEDIT

We present MULTITEXTEDIT, a multilingual benchmark for text-in-image editing evaluation. The benchmark is constructed by extending 300 base visual samples across 12 languages, yielding 3,600 annotated instances with paired input-output images, editing instructions, and masks.

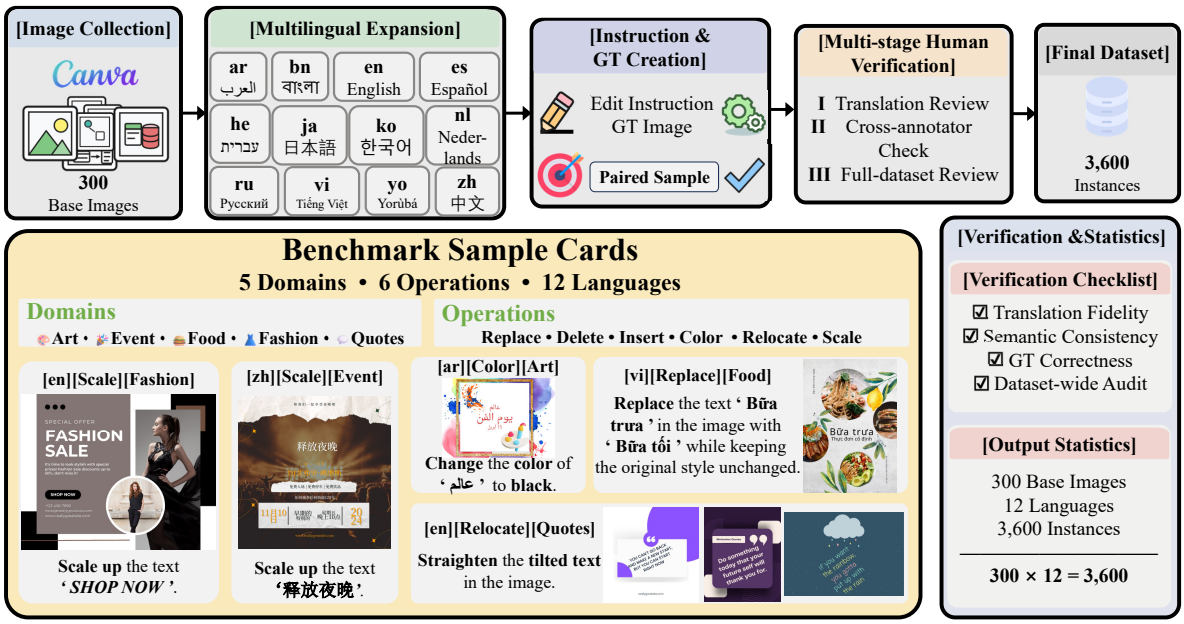

Figure 2: Overview of the MULTITEXTEDIT data construction pipeline.

3.1 Task Definition

Given a source image I containing text in a target language and an editing instruction p, the task is to produce an edited image $\hat{I}$ that applies the specified operation to the designated text region while preserving the remaining visual content. Our focus is not on multilingual instruction understanding; we therefore keep p in English across all language variants of a base image and vary only the target text content. This design isolates the language of the rendered text as the single controlled variable across comparable samples.

Formally, each instance is an 8-tuple $x = (I,\ p,\ I^*,\ M_{\text{src}},\ M_{\text{tgt}},\ l,\ o,\ d)$, where I is the input image, p the editing instruction, I^* the human-edited reference, M_{src} and M_{tgt} binary masks for the source and target text regions, and $l \in \mathcal{L}$, $o \in \mathcal{O}$, $d \in \mathcal{D}$ the language, operation, and domain labels respectively. The dual masks support region-aware evaluation at both edited and non-edited regions.

3.2 Benchmark Construction

Base images We source 300 base images from Canva poster templates(Canva, 2025), which provide diverse layouts, fonts, and visual styles suitable for controlled text editing. A larger candidate pool was automatically crawled and then manually filtered for legibility, absence of visual artifacts, and sufficient editable space around text regions. We summarize the full construction pipeline in Figure 2 .

Multilingual expansion Each base image is expanded into 12 language versions which include Arabic, Bengali, Chinese, Dutch, English, Hebrew, Japanese, Korean, Russian, Spanish, Vietnamese, Yoruba by translating only the in-image text. Compared with collecting native-language images independently per language, this shared-base design controls for background, composition, font style, and overall visual context, so that cross-lingual differences in model output can be attributed primarily to the target script and language rather than to confounding visual variation.

Editing instructions For each base image, we manually write an English editing instruction following a fixed template per operation type. The same template is reused across the 12 language variants, with only the quoted target text substituted, so that instruction difficulty is held constant across languages.

Data quality Annotation was carried out by native speakers of each target language under a three-stage protocol covering translation review, reference editing with cross-check, and a dataset-wide audit. Details of the annotation pipeline are provided in Appendix A.

Domain distribution The benchmark covers five visual domains (`Art`, `Event`, `Fashion`, `Food`, `Quotes`) and six editing operations; their joint dis-

Operation	Art	Event	Fashion	Food	Quotes	Total
Delete	15	10	30	5	5	65
Color	15	10	20	5	5	55
Replace	10	10	15	5	5	45
Insert	10	10	15	5	5	45
Relocate	10	10	15	5	5	45
Scale	10	10	15	5	5	45
Total	70	60	110	30	30	**300**

Table 1: Distribution of base samples across editing operations and visual domains in MULTITEXTEDIT. Each base sample is expanded into 12 language versions.

Dimension	Definition
IF	Whether the model performs exactly the requested operation .
TA	Whether the resulting text content matches the target.
VC	Whether the edited region blends naturally with the surrounding image .
LP	Whether non-target regions remain unchanged.

Table 2: General semantic dimensions used by the LVM judge.

tribution is shown in Table 1. We preserve the naturally imbalanced distribution rather than artificially balancing it, as it better reflects real-world design scenarios, and report stratified results by domain and operation in our experiments.

3.3 Language Coverage

To systematically evaluate cross-lingual performance, MULTITEXTEDIT covers 12 languages selected for resource-level and typological diversity, spanning Latin, CJK, Arabic, Hebrew, Cyrillic, and Bengali scripts with both LTR and RTL directionality. This selection allows us to probe not only resource-level effects but also script-specific challenges such as diacritics, dense character composition, and bidirectional text flow. The Meta columns of Table 4 summarize each language's script, directionality, and resource level; a detailed discussion of the typological contrasts is provided in Appendix B.

4 Evaluation Framework

We design a dual-track evaluation framework that combines semantic-level assessment with pixel-level metrics, motivated by the observation that neither alone captures the full picture of multilingual text editing quality.

4.1 Design Rationale

Evaluating text-in-image editing requires judging both whether the target text is edited correctly at the semantic level and whether the non-edited regions remain visually intact. Conventional pixel-level metrics such as SSIM and LPIPS cannot perceive textual meaning, which leads to two characteristic failure modes. First, a semantically correct edit may be over-penalized because of minor deviations in font style, spacing, or position relative to the reference image. Second, a missing diacritic or slight glyph error may be almost invisible at the pixel level while still changing the meaning of the word entirely, such as the Vietnamese word *mẹ* ('mother') becoming *me* ('tamarind'). This limitation becomes even more severe in multilingual settings, where languages differ substantially in writing system, character density, word length, directionality, and glyph composition, making cross-lingual evaluation itself more difficult. OCR-based alternatives are also imperfect, especially for scripts and lower-resource languages that are less reliably recognized by current text recognition systems (Shi et al., 2016; Baek et al., 2019; Bautista and Atienza, 2022), so OCR noise can obscure the true editing ability of the model.

Motivated by these challenges, we adopt a dual-track evaluation framework. The semantic track uses an LVM judge to assess editing correctness, covering instruction following, text accuracy, visual consistency, layout preservation, and script fidelity (Section 4.2). The pixel track uses mask-aware metrics to separate edited regions from the background and quantify how well non-edited content is preserved (Section 4.3). Following recent model-based automatic evaluation paradigms (Zheng et al., 2023; Qian et al., 2025; Wu et al., 2025d), we use an LVM judge for scalable semantic evaluation, and we further validate its reliability through a human agreement study in Section 4.4. Figure 3 provides an overview of the full evaluation pipeline.

4.2 Semantic Evaluation via LVM Judge

For each sample, the LVM judge receives the source image, the model's edited output, the human-annotated reference, and the editing instruction. It returns an independent integer score on a 0–5 scale for five dimensions. The four general editing dimensions are summarized in Table 2; the complete evaluator prompts are provided in Appendix J.

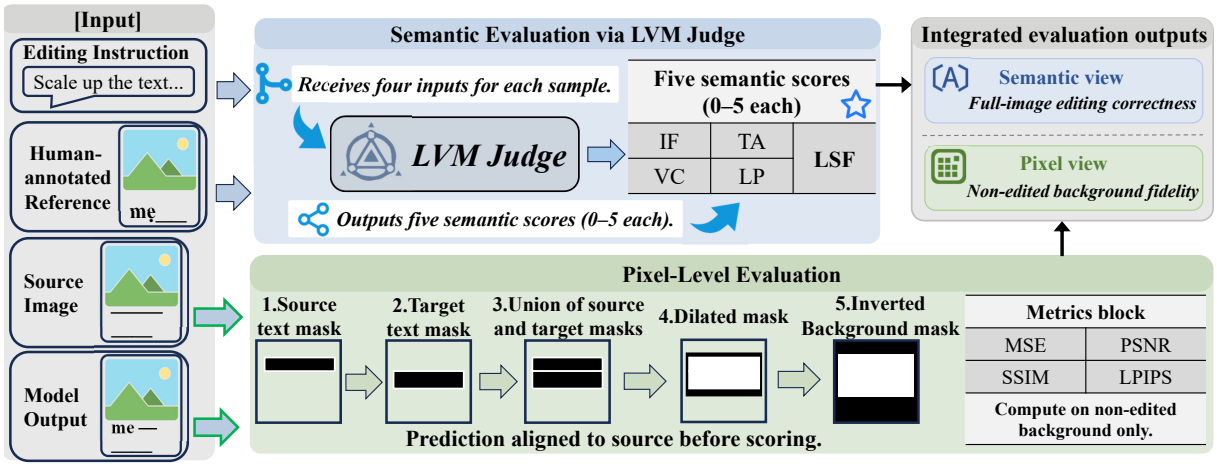

Figure 3: Overview of the dual-track evaluation framework. The semantic track uses an LVM judge to score instruction following (IF), text accuracy (TA), visual consistency (VC), layout preservation (LP), and language/script fidelity (LSF) from the source image, model output, reference image, and editing instruction. The pixel track evaluates background preservation on the non-edited region.

Language/Script Fidelity (LSF). The four dimensions above are largely language-agnostic: IF, VC, and LP do not depend on the target language, and TA captures content-level correctness but is insensitive to script-level errors that do not alter the recognizable meaning of a word. However, multilingual text editing introduces a distinct class of failure that TA alone cannot diagnose. A model may produce text whose coarse content is recognizable yet whose script realization is incorrect—missing diacritics in Vietnamese, reversed character order in Arabic, missing tone marks in Yoruba, or mixed scripts within a single word. We therefore introduce LSF to specifically evaluate script-level fidelity, covering character correctness, diacritics and tonal marks, writing directionality (RTL/LTR), and script purity. To illustrate the distinction: if a model renders the Vietnamese target *mẹ* as *me*, TA may remain moderately high because the base word is recognizable, whereas LSF should drop substantially because the diacritic is missing.

Directly scoring script fidelity on the full image is unreliable because the judge may be distracted by other, unedited text regions. We therefore adopt a two-stage protocol. In the tracing stage, the judge compares the source image and the reference image to identify the target text segment, and transcribes both the expected target text and the corresponding text observed in the model's prediction. In the scoring stage, the judge evaluates script fidelity based solely on the traced target segment, ignoring all other text in the image. This decomposition ensures that LSF focuses exclusively on the edited text and is not confounded by unrelated content. Figure 4 illustrates this protocol. For text deletion samples, where no target text is expected in the output, LSF is marked as not applicable. We validate this design against single-call alternatives in Appendix G.

4.3 Pixel-Level Evaluation

We complement the semantic metrics with four standard pixel-level metrics: MSE, PSNR, SSIM, and LPIPS, computed on the non-edited background region.

For each sample, source-image and reference-image text masks are merged via union, dilated to buffer minor spatial misalignment, and inverted to obtain the background mask. All pixel scores are computed using this mask so that only non-edited content contributes. This mask-aware design complements LP: LP captures coarse layout preservation as a semantic judgment, whereas the pixel metrics quantify low-level background changes outside the edited text region.

4.4 Human Agreement Validation

To assess LVM-based evaluation reliability, we conduct a human agreement study on 600 model predictions, with 50 samples per language drawn from the pooled outputs of all evaluated models. Following the same validation setup as our pilot study, the subset is stratified to cover different editing operation types, and two human raters score each sample using the same five semantic dimensions and the same 0–5 scale as the LVM judge. For text dele-

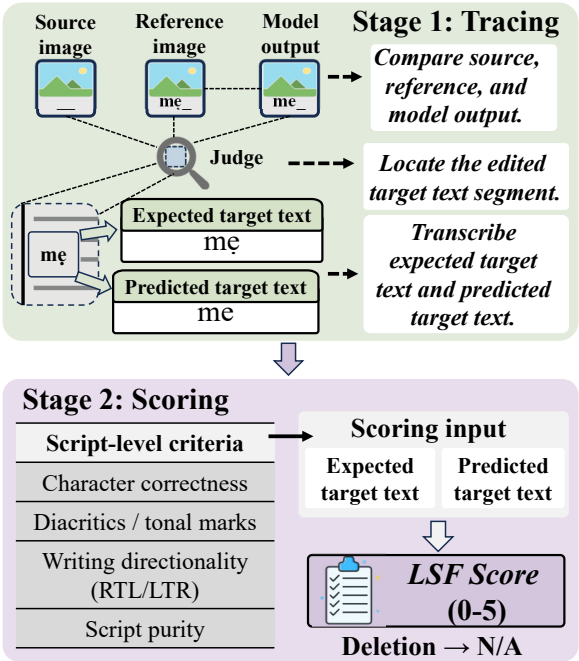

Figure 4: Two-stage protocol for language/script fidelity (LSF) scoring. The judge first traces the edited target text by comparing the source image, reference image, and model output, then scores script-level fidelity using only the expected and predicted target text, focusing on character correctness, diacritics or tonal marks, writing directionality, and script purity.

tion samples, where no target text should remain, LSF is marked as not applicable and excluded from agreement computations involving LSF.

We report both inter-rater consistency and GPT–human agreement. Human–human agreement is high (QWK 0.967–0.982; within-1 agreement 1.000), showing that the human labels are stable. We use QWK to measure ordinal score agreement and Spearman's ρ to measure rank-order consistency; GPT–human language-level semantic means are also stable (QWK 0.686–0.776; Spearman's ρ 0.726–0.817). Full results are reported in Appendix F, supporting use of the LVM judge as our primary semantic evaluator.

5 Experiments

We evaluate a diverse set of open-source and proprietary image editing systems on MULTITEXTEDIT. For each system we adopt the highest-quality inference settings recommended by its provider; hardware and API details are given in Appendix E.

5.1 Baseline Models

We evaluate 12 model settings, including both open-source and proprietary systems, and treat thinking-enabled variants as separate settings when they involve distinct inference procedures. The open-source settings include Step1X-Edit and Step1X-Edit-thinking (Liu et al., 2025), InstructPix2Pix (Brooks et al., 2023), OmniGen2 (Wu et al., 2025b), Bagel and Bagel-thinking (Deng et al., 2025), FLUX.1-Kontext-dev (Black Forest Labs et al., 2025), Qwen-Image-Edit and Qwen-Image-Edit-2511 (Wu et al., 2025a), and FireRed-Image-Edit-1.1 (Qiao et al., 2026). The proprietary systems include GPT-image-1.5 (OpenAI, 2025b) and Nano Banana 2 (Raisinghani, 2026). Overall, this model suite covers early instruction-based image editing methods, recent open-source multimodal editing systems, and strong commercial black-box models, enabling us to examine whether multilingual degradation is consistent across different model families and deployment settings.

5.2 Overall Results

Table 3 reports the overall semantic and pixel-level performance of the evaluated systems on MULTITEXTEDIT. We define SEMAVG as the arithmetic mean of IF, TA, VC, LP, and LSF; for `text_delete` samples, where LSF is not applicable, it is computed over IF, TA, VC, and LP only.

Nano Banana 2 achieves the strongest overall performance, ranking first on SEMAVG, IF, TA, VC, LSF, and all four pixel-level metrics, while Step1X-Edit remains best on LP. Among open-source systems, Qwen-Image-Edit-2511 and FireRed-Image-Edit-1.1 are the strongest semantic performers, whereas Step1X-Edit and Step1X-Edit-thinking are more competitive on layout preservation and pixel fidelity. Crucially, English performance exceeds the non-English average for every evaluated system, confirming pervasive multilingual degradation.[1]

We further observe a semantic–pixel mismatch: Step1X-Edit-thinking and Bagel-thinking achieve strong background fidelity but remain much weaker on semantic correctness, showing that visual preservation alone does not imply successful multilingual text editing.

5.3 Cross-Lingual Performance Degradation

Table 4 aggregates language-level degradation across the 12 model settings, using English as

[1]The English–non-English gap should be interpreted together with absolute performance: InstructPix2Pix has the smallest gap mainly because it performs weakly in both settings, whereas Nano Banana 2 combines the strongest absolute scores with a relatively small gap.

Model	Semantic						Pixel				EN/Non-EN		
	SemAvg↑	IF↑	TA↑	VC↑	LP↑	LSF↑	MSE↓	PSNR↑	SSIM↑	LPIPS↓	EN↑	Non-EN↑	Gap↓
Nano Banana 2	**4.006**	**3.765**	**4.694**	**3.633**	3.152	**4.785**	**153.743**	**31.242**	**0.947**	**0.040**	**4.098**	**3.997**	0.100
GPT-image-1.5	3.577	3.114	4.065	3.456	2.752	4.499	1207.744	19.717	0.795	0.105	4.049	3.532	0.517
Qwen-Image-Edit-2511	3.214	2.897	3.524	2.988	3.149	3.512	938.282	21.315	0.748	0.129	3.842	3.157	0.685
FireRed-Image-Edit-1.1	3.204	2.933	3.857	2.976	2.612	3.642	1258.208	19.417	0.678	0.248	3.772	3.152	0.620
Step1X-Edit	2.802	2.327	2.701	2.625	**3.432**	2.924	937.224	27.150	0.902	0.057	3.358	2.751	0.606
Bagel	2.424	1.914	2.609	2.066	2.641	2.889	1049.571	25.036	0.913	0.070	2.986	2.373	0.614
FLUX.1-Kontext-dev	2.401	1.922	2.501	2.210	2.223	3.151	1697.211	17.888	0.701	0.164	3.110	2.337	0.773
Bagel-thinking	2.179	1.801	2.228	1.839	2.758	2.271	598.328	26.416	0.935	0.046	2.646	2.137	0.509
Qwen-Image-Edit	2.055	1.651	2.404	1.674	1.638	2.909	2742.375	15.828	0.677	0.231	2.477	2.017	0.460
OmniGen2	1.734	1.272	2.137	1.324	1.551	2.383	5564.379	17.682	0.769	0.197	2.090	1.701	0.389
Step1X-Edit-thinking	1.604	1.054	1.174	1.177	1.522	3.091	513.660	28.403	0.918	0.046	2.122	1.557	0.565
InstructPix2Pix	1.190	0.892	1.623	0.448	0.574	2.411	4124.039	15.177	0.750	0.237	1.232	1.186	**0.045**
Average	2.532	2.129	2.793	2.201	2.334	3.206	1732.064	22.106	0.811	0.131	2.982	2.491	0.490

Table 3: Overall results on MULTITEXTEDIT. SEMAVG is the mean of IF, TA, LSF, VC, and LP. EN and Non-EN denote semantic averages on the English and non-English subsets; Gap is their difference.

Meta				Score	Drop vs. EN						Std
Language	Script	Dir.	Res.	SemAvg↑	ΔSem ±95% CI↓	ΔTA↓	ΔLSF↓	ΔIF↓	ΔVC↓	ΔLP↓	Std↓
English	Latn	LTR	H	**2.982**	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Hebrew	Hebr	RTL	M	2.126	**0.880 ± 0.044**	**1.276**	**1.551**	0.752	0.631	0.373	**0.461**
Arabic	Arab	RTL	MH	2.202	0.834 ± 0.046	1.133	1.259	**0.779**	**0.706**	**0.396**	0.397
Bengali	Beng	LTR	M	2.285	0.739 ± 0.043	1.053	1.170	0.711	0.544	0.291	0.419
Korean	Hang	LTR	MH	2.438	0.569 ± 0.041	0.799	0.910	0.564	0.380	0.280	0.289
Russian	Cyrl	LTR	MH	2.440	0.542 ± 0.041	0.791	1.039	0.468	0.443	0.159	0.300
Japanese	Jpan	LTR	MH	2.551	0.452 ± 0.040	0.632	0.712	0.449	0.326	0.200	0.281
Vietnamese	Latn	LTR	M	2.540	0.423 ± 0.039	0.550	0.945	0.323	0.369	0.116	0.195
Yoruba	Latn	LTR	L	2.549	0.413 ± 0.037	0.510	1.027	0.308	0.257	0.144	0.168
Chinese	Hani	LTR	H	2.677	0.316 ± 0.039	0.394	0.386	0.334	0.297	0.186	0.374
Spanish	Latn	LTR	H	2.798	0.173 ± 0.034	0.239	0.379	0.140	0.141	0.048	0.126
Dutch	Latn	LTR	M	2.800	0.170 ± 0.034	0.209	0.329	0.145	0.162	0.066	0.117
Avg. (Non-EN)	–	–	–	2.491	0.501	0.690	0.882	0.452	0.387	0.205	0.284

Table 4: Language coverage and cross-lingual degradation relative to English, aggregated across 12 model settings. Script, Dir., and Res. indicate writing system, text directionality, and resource level (H = High, MH = Mid-High, M = Mid, L = Low). SEMAVG is the average semantic score, and ± values for ΔSem denote paired bootstrap 95% confidence-interval half-widths. **Std** denotes the standard deviation of model-wise ΔSem values.

the reference. Paired bootstrap 95% confidence-interval half-widths are reported for ΔSem, with component-metric uncertainty estimates provided in Appendix C.

Cross-lingual degradation is positive for every non-English language, but the magnitude is highly uneven. Hebrew shows the largest semantic drop (ΔSem = 0.880 ± 0.044), followed by Arabic (0.834 ± 0.046), whereas Spanish (0.173 ± 0.034) and Dutch (0.170 ± 0.034) remain closest to English; Chinese is also comparatively less degraded (0.316 ± 0.039). This pattern suggests that cross-lingual editing difficulty is not uniform, and may be jointly related to writing direction, script characteristics, and language resource availability. The unevenness is not only a difference in average score: languages such as Hebrew and Bengali are also more variable across model families, whereas Spanish and Dutch show more consistently small drops, suggesting that some scripts expose model-specific weaknesses more strongly than others.

At the dimension level, the largest non-English drops occur in text accuracy (ΔTA = 0.690) and language/script fidelity (ΔLSF = 0.882), exceeding the declines in instruction following (ΔIF = 0.452), visual consistency (ΔVC = 0.387), and layout preservation (ΔLP = 0.205). This indicates that current models often preserve the coarse structure of the requested edit, yet struggle to generate text that is both linguistically correct and faithful to the target writing system.

Figure 5 illustrates the same trend under a shared visual context: Chinese and Vietnamese show noticeable degradation in text accuracy or script fidelity, while Arabic fails more severely despite preserving much of the global poster structure. Overall, these results show that multilingual text editing errors are concentrated more heavily in language-

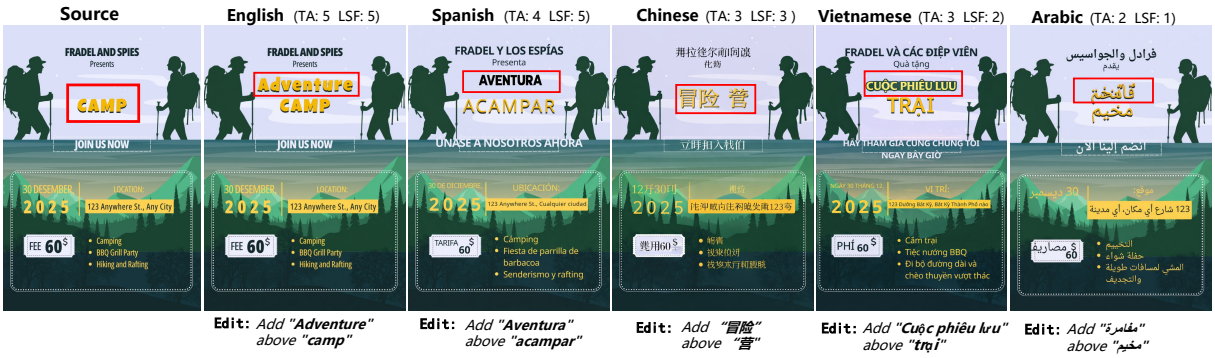

Figure 5: Controlled cross-lingual comparison under the same edit template. Only the target text varies across languages, while semantic errors become more pronounced from Spanish to Arabic despite largely preserved layout.

sensitive semantic dimensions than in coarse visual structure.

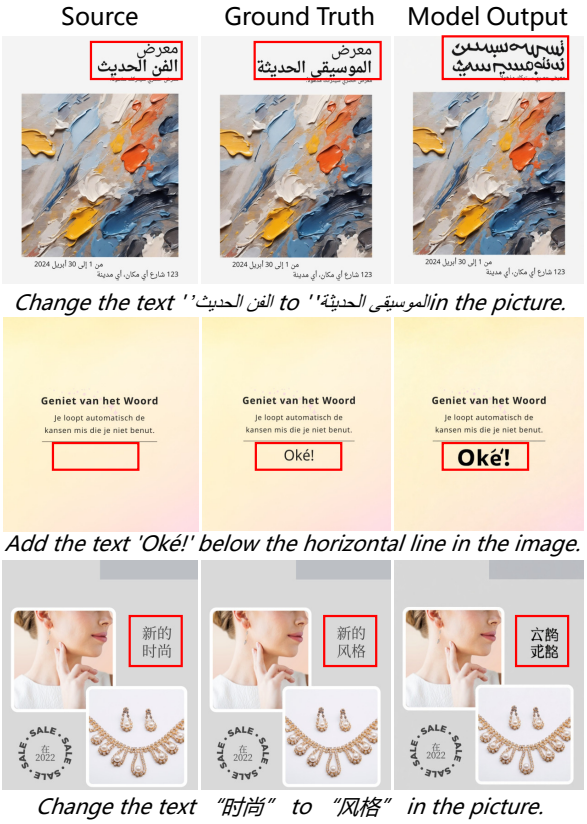

Figure 6: Representative language-specific failure cases. From top to bottom: right-to-left, diacritic-related, and dense-character errors.

5.4 Language-Specific Failure Patterns

The representative cases in Figure 6 reveal that multilingual text editing errors are often language-specific rather than uniform across languages. In right-to-left scripts such as Hebrew and Arabic, models may preserve the overall layout while producing incorrect character order, unnatural directionality, or distorted script structure. In diacritic-sensitive languages such as Bengali, Vietnamese, and Yoruba, small glyph-level errors, including missing or corrupted diacritics, can substantially alter the target text despite limited pixel-level differences. In dense character systems such as Chinese, Japanese, and Korean, models may generate visually plausible characters that remain lexically incorrect. These cases suggest that multilingual editing failures arise not only from general image editing difficulty, but also from language-specific challenges at the script and character level, further supporting the inclusion of language/script fidelity in our evaluation framework.

These results suggest that multilingual text editing requires script-aware generation, since visually plausible outputs can still fail linguistically.

6 Conclusion

In this paper, we introduced MULTITEXTEDIT, a controlled benchmark for multilingual text-in-image editing that covers 12 languages, 5 visual domains, and 6 editing operations, with paired input images, human-edited references, editing prompts, and region masks for fine-grained evaluation. Using this benchmark, we evaluated 12 open-source and proprietary systems and showed that current models suffer substantial degradation outside English, especially in text accuracy and language/script fidelity, while often preserving visually plausible layouts that are semantically incorrect. These findings indicate that text-in-image editing is not only a visual generation problem, but also depends on language understanding, text localization, and cross-lingual instruction grounding; accordingly, MULTITEXTEDIT provides a resource and empirical foundation for evaluating and improving more robust multilingual image editing systems.

Limitations

Our benchmark covers 12 languages, which, while typologically diverse, does not exhaust the world's writing systems; languages with unique typographic challenges, such as Thai, Khmer, or Ge'ez, remain untested. The base images are sourced from Canva poster templates, which, despite their visual diversity, may not fully represent all real-world text-in-image editing scenarios such as natural scene text or handwritten content. For several lower-resource languages, the available font variety is limited, so all samples use commonly available fonts rather than reflecting the full range of typographic styles found in native design practice. Although our annotation protocol includes native-speaker review, a small number of lower-resource language samples were not verified by formally trained linguistic experts in those languages, which may leave minor translation or typographic issues undetected.

References

Franklin Oladiipo Asahiah, Odetunji Ajadi Odejobi, and Emmanuel Rotimi Adagunodo. 2017. Restoring tonemarks in standard yorùbá electronic text: Improved model. *Computer Science*, 18(3):301–315.

Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. 2019. What is wrong with scene text recognition model comparisons? Dataset and model analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4715–4723.

Darwin Bautista and Rowel Atienza. 2022. Scene text recognition with permuted autoregressive sequence models. *arXiv preprint arXiv:2207.06966*.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, and 2 others. 2025. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. *Preprint*, arXiv:2211.09800.

Canva. 2025. Canva: About. https://www.canva.com/about/. Official Canva website.

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

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

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. 2025. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*.

Zhengyao Fang, Pengyuan Lyu, Jingjing Wu, Chengquan Zhang, Jun Yu, Guangming Lu, and Wenjie Pei. 2025. Recognition-synergistic scene text editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13104–13113.

Rui Gui, Yang Wan, Haochen Han, Dongxing Mao, Fangming Liu, Min Li, and Alex Jinpeng Wang. 2025. TextEditBench: Evaluating reasoning-aware text editing beyond rendering. *arXiv preprint arXiv:2512.16270*.

Rui Lan, Yancheng Bai, Xu Duan, Mingxing Li, Dongyang Jin, Ryan Xu, Dong Nie, Lei Sun, and Xiangxiang Chu. 2025. FLUX-Text: A simple and advanced diffusion transformer baseline for scene text editing. *arXiv preprint arXiv:2505.03329*.

Phuong Le-Hong. 2021. Diacritics generation and application in hate speech detection on vietnamese social networks. *Knowledge-Based Systems*, 233:107504.

Chen Li, Zhe Yao, Jiali Zhang, Jihong Guan, Dongxiang Zhang, Conghui He, Yifei Wang, Zhihua Wang, Yaqing Wang, and Jiabao Zhao. 2026. IMTBench: A multi-scenario cross-modal collaborative evaluation benchmark for in-image machine translation. *arXiv preprint arXiv:2603.10495*.

Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, Guopeng Li, Yuang Peng, Quan Sun, Jingwei Wu, Yan Cai, Zheng Ge, Ranchen Ming, Lei Xia, Xianfang Zeng, and 5 others. 2025. Step1x-edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*.

OpenAI. 2025a. Introducing GPT-5.2. https://openai.com/index/introducing-gpt-5-2/. Official OpenAI release page for GPT-5.2.

OpenAI. 2025b. The new chatgpt images is here. https://openai.com/index/new-chatgpt-images-is-here/. Official OpenAI release page for GPT-image-1.5, published December 16, 2025.

Sungmin Park, Sungkyu Choi, and Hwanhee Lee. 2025. STELLAR: Scene text editor for low-resource languages and real-world data. *arXiv preprint arXiv:2511.09977*.

Yusu Qian, Jiasen Lu, Tsu-Jui Fu, Xinze Wang, Chen Chen, Yinfei Yang, Wenze Hu, and Zhe Gan. 2025. Gie-bench: Towards grounded evaluation for text-guided image editing. *arXiv preprint arXiv:2505.11493*.

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

Naina Raisinghani. 2026. Nano banana 2: Combining pro capabilities with lightning-fast speed. https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/. Google blog post on Nano Banana 2, published February 26, 2026.

Baoguang Shi, Xiang Bai, and Cong Yao. 2016. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2298–2304.

Yuxiang Tuo, Yifeng Geng, and Liefeng Bo. 2024a. AnyText2: Visual text generation and editing with customizable attributes. *arXiv preprint arXiv:2411.15245*.

Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. 2024b. AnyText: Multilingual visual text generation and editing. In *The Twelfth International Conference on Learning Representations*.

W3C Internationalization Working Group. 2020. Requirements for japanese text layout. https://www.w3.org/TR/jlreq/. W3C Working Group Note, published August 11, 2020.

W3C Internationalization Working Group. 2024a. Bengali script resources. https://www.w3.org/TR/beng-lreq/. W3C Group Draft Note, published November 14, 2024.

W3C Internationalization Working Group. 2024b. Hebrew script resources. https://www.w3.org/TR/hebr-lreq/. W3C Group Draft Note, published November 14, 2024.

W3C Internationalization Working Group. 2024c. Korean script resources. https://www.w3.org/TR/kore-lreq/. W3C Group Draft Note, published November 15, 2024.

W3C Internationalization Working Group. 2025. Arabic & persian layout requirements. https://www.w3.org/TR/alreq/. W3C Group Draft Note, published October 2, 2025.

W3C Internationalization Working Group. 2026. Requirements for chinese text layout. https://www.w3.org/TR/clreq/. W3C Group Note Draft, published March 1, 2026.

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

Tong Wang, Xianwei Qin, and Ting Liu. 2024. TextMastero: Mastering high-quality scene text editing in diverse languages and styles. *arXiv preprint arXiv:2408.10623*.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. 2004. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

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, and 20 others. 2025a. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*.

Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ze Liu, Ziyi Xia, Chaofan Li, Haoge Deng, Jiahao Wang, Kun Luo, Bo Zhang, Defu Lian, Xinlong Wang, and 3 others. 2025b. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*.

Hao Wu, Wenhao Chai, Tingyu Qu, Gaojie Wang, Haoran Lu, Tianyu Wang, Yue Liu, Kaier Zhou, Shuchang Zhou, Juncheng Li, Jiahang Liu, and Jiang Liu. 2025c. Visual text processing: A comprehensive review and unified evaluation. *arXiv preprint arXiv:2504.21682*.

Yongliang Wu, Zonghui Li, Xinting Hu, Xinyu Ye, Xianfang Zeng, Gang Yu, Wenbo Zhu, Bernt Schiele, Ming-Hsuan Yang, and Yang Xu. 2025d. Kris-bench: Benchmarking next-level intelligent image editing models. *Preprint*, arXiv:2505.16707.

Weichao Zeng, Yan Shu, Zhenhang Li, Dongbao Yang, and Yu Zhou. 2024. TextCtrl: Diffusion-based scene text editing with prior guidance control. In *Advances in Neural Information Processing Systems*, volume 37, pages 138569–138594.

Boqiang Zhang, Hongtao Xie, Zuan Gao, and Yuxin Wang. 2024. Choose what you need: Disentangled representation learning for scene text recognition removal and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28358–28368.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*.

A Annotation Protocol

Annotation was carried out by a team of international students who are native speakers of the target languages, under a three-stage protocol. (i) *Translation review*: initial translations produced by GPT-5.2 (OpenAI, 2025a) were proofread by the corresponding native speaker, who corrected translation errors and flagged cases where the translated text conflicted with the original layout, caused occlusion, or produced visually implausible arrangements. (ii) *Reference editing and cross-check*: human experts manually edited each image in Canva to produce the ground-truth reference, which was then reviewed by a second annotator for text correctness, instruction – operation consistency, and visual naturalness of the edited region. (iii) *Dataset-wide audit*: a final pass over the full dataset checked for residual text occlusion, instruction errors, and reference – annotation mismatches.

B Language Typological Details

The 12 languages in MULTITEXTEDIT exhibit substantial typological variation (Wu et al., 2025c). Latin-script languages (English, Spanish, Dutch, Vietnamese, Yoruba) serve as a baseline group, though Vietnamese and Yoruba introduce additional complexity through diacritics and tonal marks that affect glyph rendering (Le-Hong, 2021; Asahiah et al., 2017). CJK languages (Chinese, Japanese, Korean) feature high character density, complex stroke patterns, and support for vertical text layout, posing challenges for both recognition and spatial editing (W3C Internationalization Working Group, 2026, 2020, 2024c). Right-to-left scripts (Arabic, Hebrew) reverse the default text flow, requiring models to handle mirrored layout conventions (W3C Internationalization Working Group, 2025, 2024b). Russian uses Cyrillic script, which shares structural similarities with Latin but differs in glyph shapes, while Bengali employs an abugida writing system with conjunct consonants and vowel diacritics that form ligatures (W3C Internationalization Working Group, 2024a). These linguistic differences provide contrastive dimensions for analyzing cross-lingual performance degradation in our experiments.

C Uncertainty of Language-Level Degradation

Table 5 reports 95% confidence-interval half-widths for the five component semantic metrics. English and target-language results are matched by model, category, sample id, and operation before computing English minus target-language scores. Confidence intervals are estimated with 10,000 paired bootstrap resamples. LSF uses only pairs where both sides have valid LSF scores; all reported drops remain positive after Benjamini–Hochberg FDR correction over the 11 languages and six tested metrics.

D Operation- and Domain-Level Results

This appendix reports full-sample stratified results by editing operation and by visual domain. These two views complement the main language-level analysis by checking whether the observed English–non-English degradation persists across different edit intents and visual scenarios, rather than being driven by a single subset of the benchmark.

The operation-level results support the main conclusion: every operation group shows an English–non-English gap, indicating that multilingual degradation appears across both text-generation-oriented edits and more attribute- or layout-oriented edits. The gap is larger for `insert` and `replace`, which directly require producing target-language text, but it remains positive for `color_change`, `relocation`, `scaling`, and `text_delete`. The domain-level results show the same pattern across visual scenarios. All five domains have positive gaps, so the degradation reported in the main experiments is not an artifact of a particular domain distribution.

E Inference Configuration

All open-source model inference and the automatic evaluation pipeline are conducted on eight NVIDIA A800 GPUs. For each evaluated system, we follow the official inference pipeline and adopt the highest-quality inference settings recommended by the provider, so that the comparison reflects each system under a strong default configuration rather than a resource-constrained setup. For proprietary systems, we access their official APIs and use the highest-quality options available at the time of evaluation.

F Human – LVM Agreement Validation

This appendix reports the updated human validation study used to assess the reliability of the GPT-based LVM judge. We focus on two reliability questions: whether the two human raters are con-

Language	ΔIF ±95% CI	ΔTA ±95% CI	ΔVC ±95% CI	ΔLP ±95% CI	ΔLSF ±95% CI
Hebrew	0.752 ± 0.050	1.276 ± 0.070	0.631 ± 0.055	0.373 ± 0.048	1.551 ± 0.079
Arabic	0.779 ± 0.052	1.133 ± 0.069	0.706 ± 0.056	0.396 ± 0.049	1.259 ± 0.082
Bengali	0.711 ± 0.051	1.053 ± 0.070	0.544 ± 0.055	0.291 ± 0.047	1.170 ± 0.078
Korean	0.564 ± 0.047	0.799 ± 0.065	0.380 ± 0.052	0.280 ± 0.046	0.910 ± 0.074
Russian	0.468 ± 0.045	0.791 ± 0.065	0.443 ± 0.051	0.159 ± 0.044	1.039 ± 0.076
Japanese	0.449 ± 0.046	0.632 ± 0.064	0.326 ± 0.054	0.200 ± 0.046	0.712 ± 0.071
Vietnamese	0.323 ± 0.044	0.550 ± 0.061	0.369 ± 0.050	0.116 ± 0.044	0.945 ± 0.072
Yoruba	0.308 ± 0.043	0.510 ± 0.059	0.257 ± 0.051	0.144 ± 0.044	1.027 ± 0.067
Chinese	0.334 ± 0.046	0.394 ± 0.064	0.297 ± 0.053	0.186 ± 0.045	0.386 ± 0.070
Spanish	0.140 ± 0.041	0.239 ± 0.054	0.141 ± 0.049	0.048 ± 0.041	0.379 ± 0.061
Dutch	0.145 ± 0.040	0.209 ± 0.053	0.162 ± 0.050	0.066 ± 0.043	0.329 ± 0.059

Table 5: Paired English–target-language drops for component semantic metrics with paired-bootstrap 95% confidence-interval half-widths.

Operation	SemAvg	IF	TA	LSF	VC	LP	EN	Non-EN	Gap
`color_change`	2.893	2.410	3.382	4.544	2.083	2.076	3.194	2.866	0.328
`replace`	2.107	1.978	2.122	1.956	2.060	2.419	2.761	2.048	0.713
`insert`	2.129	2.185	1.948	1.863	1.861	2.790	2.940	2.055	0.885
`relocation`	2.281	1.598	2.530	3.082	2.055	2.151	2.601	2.252	0.349
`scaling`	2.740	1.936	3.132	3.758	2.393	2.486	3.112	2.706	0.406
`text_delete`	2.649	2.533	3.080	–	2.737	2.245	2.968	2.620	0.349

Table 6: Semantic results grouped by editing operation.

sistent with each other, and whether agreement remains stable across languages.

We first compare the two human raters directly. Table 8 shows that human annotation is highly consistent across all semantic dimensions. The QWK values range from 0.967 for LSF to 0.982 for TA and VC, Spearman's ρ ranges from 0.924 to 0.982, and within-1 agreement is 1.000 for every metric. This indicates that the human labels are stable rather than depending on idiosyncratic judgments from a single annotator.

We next check whether the validation behaves consistently across languages. Table 9 reports per-language agreement. The per-language QWK values range from 0.686 to 0.776, Spearman's ρ ranges from 0.726 to 0.817, and within-1 agreement ranges from 0.860 to 0.940. No language is an obvious outlier: even the lowest-QWK languages remain close to the rest of the set and maintain high within-1 agreement. This suggests that the validation is not reliable only for English or Latin-script cases, but remains stable across the multilingual conditions used in MULTITEXTEDIT.

G Selecting the Two-Stage LSF Protocol

Because LSF is intended to measure script fidelity for the edited target text, we further test whether the two-stage design is necessary rather than simply assuming it from the metric definition. On the human-scored validation subset, we compare three GPT-based LSF schemes: a direct single-call scheme that asks the judge to score LSF directly, a single-call trace-and-score scheme that asks the judge to first identify the relevant text content and then score it within the same response, and the two-stage scheme used in the main benchmark, where target tracing and script-fidelity scoring are separated into two calls.

Scheme	MAE	Within-1	QWK
Direct single-call	0.559	0.880	0.668
Trace+score single-call	0.565	0.865	0.705
Two-stage (ours)	0.382	0.940	0.819

Table 10: Comparison of three LSF scoring schemes against the mean human LSF score on the 465 common valid rows. Lower MAE is better; higher within-1 agreement and QWK are better.

Table 10 shows that the two-stage protocol is consistently closer to human judgment. It has the

Domain	SemAvg	IF	TA	LSF	VC	LP	EN	Non-EN	Gap
Art	2.617	2.302	2.909	3.192	2.352	2.392	3.081	2.575	0.507
Event	2.388	2.038	2.559	2.969	2.149	2.287	2.870	2.344	0.526
Fashion	2.498	2.188	2.826	3.269	2.236	2.210	2.871	2.464	0.407
Food	2.364	1.953	2.585	3.120	2.100	2.219	2.888	2.317	0.572
Quotes	2.581	2.167	2.765	2.848	2.275	2.826	3.059	2.537	0.522

Table 7: Semantic results grouped by visual domain.

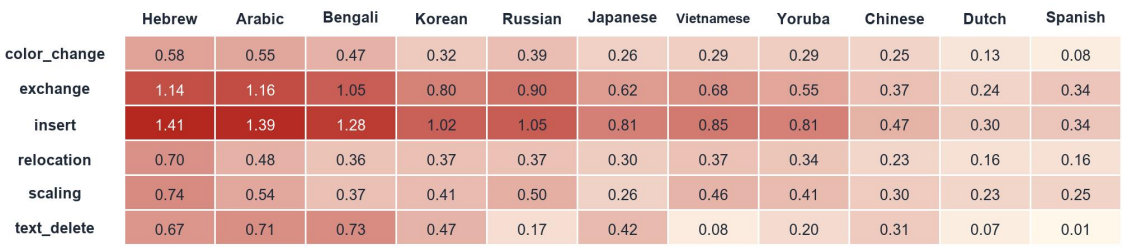

Figure 7: Operation-by-language ΔSem heatmap. Darker cells indicate larger degradation relative to English.

Metric	Count	Spearman	QWK	MAE	Within-1
IF	600	0.977	0.977	0.108	1.000
TA	600	0.975	0.982	0.090	1.000
VC	600	0.981	0.982	0.088	1.000
LP	600	0.982	0.981	0.088	1.000
LSF	468	0.924	0.967	0.096	1.000

Table 8: Agreement between the two human raters.

Language	Count	Spearman	QWK	MAE	Within-1
Arabic	50	0.765	0.733	0.487	0.900
Bengali	50	0.783	0.745	0.486	0.920
English	50	0.726	0.687	0.467	0.920
Spanish	50	0.769	0.700	0.549	0.860
Hebrew	50	0.801	0.750	0.513	0.940
Japanese	50	0.817	0.742	0.485	0.940
Korean	50	0.813	0.767	0.460	0.940
Dutch	50	0.750	0.686	0.493	0.940
Russian	50	0.791	0.776	0.451	0.920
Vietnamese	50	0.766	0.740	0.456	0.940
Yoruba	50	0.742	0.722	0.452	0.920
Chinese	50	0.774	0.770	0.512	0.900

Table 9: Per-language GPT – human agreement.

lowest absolute error, the highest near-match agreement, and the strongest ordinal agreement with the human LSF scores. Compared with direct single-call scoring, the two-stage protocol reduces MAE from 0.559 to 0.382 and increases QWK from 0.668 to 0.819. Compared with the single-call trace-and-score variant, it reduces MAE from 0.565 to 0.382 and increases QWK from 0.705 to 0.819. We therefore use the two-stage LSF protocol as the official scoring design in all main experiments: the first stage narrows the judge's attention to the edited target text, and the second stage scores script fidelity on that isolated evidence.

H Judge Reliability Analysis

We further examine whether the main findings depend on a particular automatic judge or on a single stochastic scoring pass. The analysis has two complementary parts. First, we compare the primary GPT judge with an independent Gemini judge on the same fixed subset. Second, we repeat GPT-based scoring on the same subset and measure the run-to-run variation.

H.1 Cross-Judge Stability with Gemini

Experimental setup. We evaluate cross-judge stability at the language level because the central empirical finding of MULTITEXTEDIT is cross-lingual degradation. We draw a fixed 20% subset from the main benchmark by weighted random sampling over editing operations and visual domains, so the subset remains both representative of the full data distribution and random. We then rescore the same subset with `gemini-3.1-flash-lite-preview` through the native Gemini `generateContent` path and compare it with the GPT scores used in the main evaluation.

The comparison follows the same semantic scoring protocol as the main experiments, covering IF, TA, VC, LP, and LSF. In Table 11, *Gap* denotes English mean minus non-English mean, and *Gap Diff.* denotes the Gemini gap minus the GPT gap.

	Hebrew	Arabic	Bengali	Korean	Russian	Japanese	Vietnamese	Yoruba	Chinese	Dutch	Spanish
Art	0.91	0.85	0.76	0.57	0.61	0.34	0.48	0.42	0.31	0.19	0.15
Event	0.88	0.86	0.66	0.55	0.60	0.43	0.48	0.49	0.34	0.23	0.27
Fashion	0.70	0.68	0.67	0.49	0.37	0.47	0.29	0.28	0.28	0.11	0.12
Food	1.03	0.93	0.79	0.66	0.62	0.53	0.48	0.47	0.38	0.21	0.20
Quotes	0.87	0.75	0.72	0.57	0.58	0.48	0.48	0.56	0.33	0.20	0.20

Figure 8: Domain-by-language ΔSem heatmap. Darker cells indicate larger degradation relative to English.

In Table 12, SEMAVG follows the same definition as in the main results; ΔSem is the English score minus the language score under the same judge; and *Rank Shift* is the Gemini rank minus the GPT rank.

Results. Table 11 reports English and non-English scores under both judges. Across all five semantic dimensions, GPT and Gemini both assign higher average scores to English than to the non-English subset. This pattern holds despite judge-specific differences in absolute score scale. The largest GPT gaps are observed for LSF and TA, while Gemini also shows clear English–non-English gaps in all dimensions, supporting the conclusion that non-English conditions degrade semantic editing performance.

Table 12 further compares the 12 languages by average semantic score. Both judges rank English first, and all non-English languages have positive ΔSem under both judges. More importantly, the language ordering is identical across GPT and Gemini: Spanish and Dutch remain closest to English, while Arabic and Hebrew remain the most degraded languages in the fixed subset.

Figure 9 visualizes the same language-level comparison in terms of degradation relative to English. GPT and Gemini ΔSem values are strongly aligned across the 11 non-English languages, with Pearson correlation 0.987 and Spearman correlation 1.000. The figure therefore supports the tabular result: changing the automatic judge affects some absolute scores but preserves the language-level degradation pattern.

Conclusion. Replacing the primary GPT judge with an independent Gemini judge does not overturn the main language-level conclusions. Both judges show that English outperforms the non-English subset across all semantic dimensions, and both produce the same ordering over the 12 languages. These results support using GPT as the main judge while treating Gemini as an independent robustness check for the cross-lingual degradation conclusion.

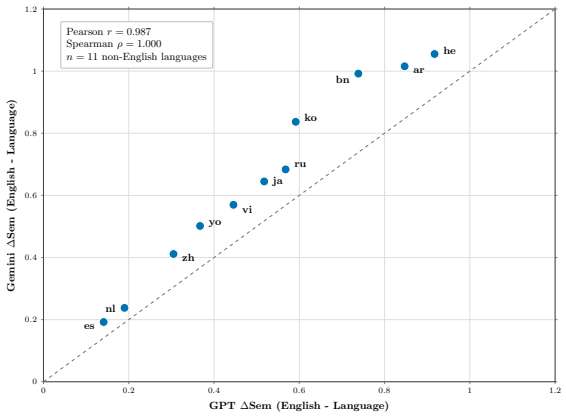

Figure 9: Language-level degradation consistency between GPT and Gemini judges. Each point corresponds to one non-English language, with ΔSem computed as English minus the language score under the same judge. Larger values indicate stronger degradation relative to English. The dashed diagonal indicates equal degradation magnitude under the two judges.

H.2 Repeated GPT Scoring Stability

Experimental setup. We next test whether the same GPT judge gives stable results under repeated scoring. We perform two repeated GPT scoring runs on the 20% subset described above and compare them with the main evaluation. We report both aggregate stability and language-level stability. For the language-level analysis, we compute ΔSem as English minus each language score, matching the degradation view used in the main analysis.

Results. Figure 10 gives the language-level view. Across the main evaluation and the two repeated runs, the ΔSem curves remain close for nearly all languages and preserve the same overall ordering: Spanish and Dutch stay closest to English, while Arabic and Hebrew remain the most degraded. This pattern indicates that repeated scoring changes individual aggregate values only slightly and does

Dim.	GPT EN	GPT Non-EN	GPT Gap	Gemini EN	Gemini Non-EN	Gemini Gap	Gap Diff.
IF	2.643	2.179	0.464	2.749	2.161	0.588	0.124
TA	3.441	2.808	0.633	3.033	2.445	0.589	−0.044
VC	2.658	2.281	0.378	2.689	2.001	0.688	0.310
LP	2.718	2.528	0.191	3.007	2.454	0.553	0.362
LSF	3.964	3.070	0.894	3.495	2.640	0.855	−0.039

Table 11: English versus non-English semantic scores under GPT and Gemini judges. Gap is English mean minus non-English mean. Gap Diff. is Gemini Gap minus GPT Gap.

Language	GPT SemAvg	GPT ΔSem	GPT Rank	Gemini SemAvg	Gemini ΔSem	Gemini Rank	Rank Shift
English	3.085	0.000	1	2.987	0.000	1	0
Spanish	2.944	0.141	2	2.795	0.192	2	0
Dutch	2.895	0.190	3	2.749	0.238	3	0
Chinese	2.780	0.305	4	2.576	0.411	4	0
Yoruba	2.718	0.367	5	2.486	0.502	5	0
Vietnamese	2.639	0.446	6	2.418	0.570	6	0
Japanese	2.567	0.518	7	2.342	0.645	7	0
Russian	2.517	0.568	8	2.304	0.684	8	0
Korean	2.493	0.592	9	2.150	0.837	9	0
Bengali	2.346	0.738	10	1.995	0.992	10	0
Arabic	2.238	0.847	11	1.972	1.016	11	0
Hebrew	2.168	0.917	12	1.932	1.055	12	0

Table 12: Language-level mean semantic scores and rank changes under GPT and Gemini judges. ΔSem is computed as English minus the language score under the same judge. Rank is ordered by SEMAVG from high to low.

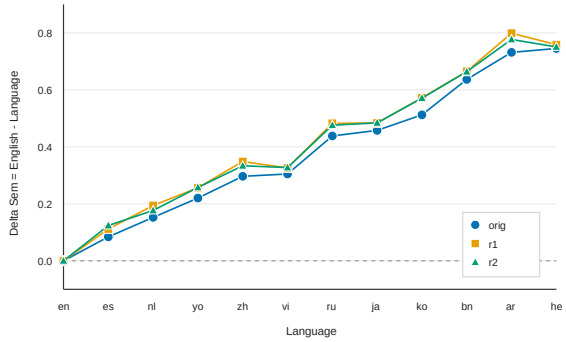

Figure 10: Language-level degradation stability across repeated GPT evaluations. ΔSem is computed as English minus the language score within each run. The nearly overlapping curves show that repeated GPT scoring preserves the cross-lingual degradation pattern.

not alter the cross-lingual degradation trend.

Table 13 provides the corresponding aggregate statistics. The model ranking is nearly unchanged across all run pairs, with Spearman correlations of 0.993, 0.993, and 1.000. The aggregate score changes are also small: the largest model-score MAE is 0.051, and the maximum score change over any model and run pair is 0.123. At the sample-dimension level, more than 95% of paired ratings differ by at most one point.

Pair	ρ	MAE	Max change	Within-1
`orig-r1`	0.993	0.045	0.123	95.19%
`orig-r2`	0.993	0.051	0.111	95.26%
`r1-r2`	1.000	0.013	0.041	96.38%

Table 13: Stability metrics across repeated GPT evaluations on the 20% subset.

Conclusion. The repeated scoring analysis shows that GPT-based evaluation has low run-to-run fluctuation for aggregate model comparison and for language-level degradation analysis. Individual sample-level scores can still vary slightly, but this variation is small enough that the main conclusions about model ranking and multilingual degradation remain stable under repeated GPT judging.

I Additional Qualitative Examples

This appendix provides additional visualization cases for the six text-in-image editing operations in MULTITEXTEDIT. Each page corresponds to one operation type, and each example shows the source image, the English instruction for readability, and the corresponding results across the 12 language variants.

Figure 11: Additional qualitative examples for color change. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

Figure 12: Additional qualitative examples for text exchange. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

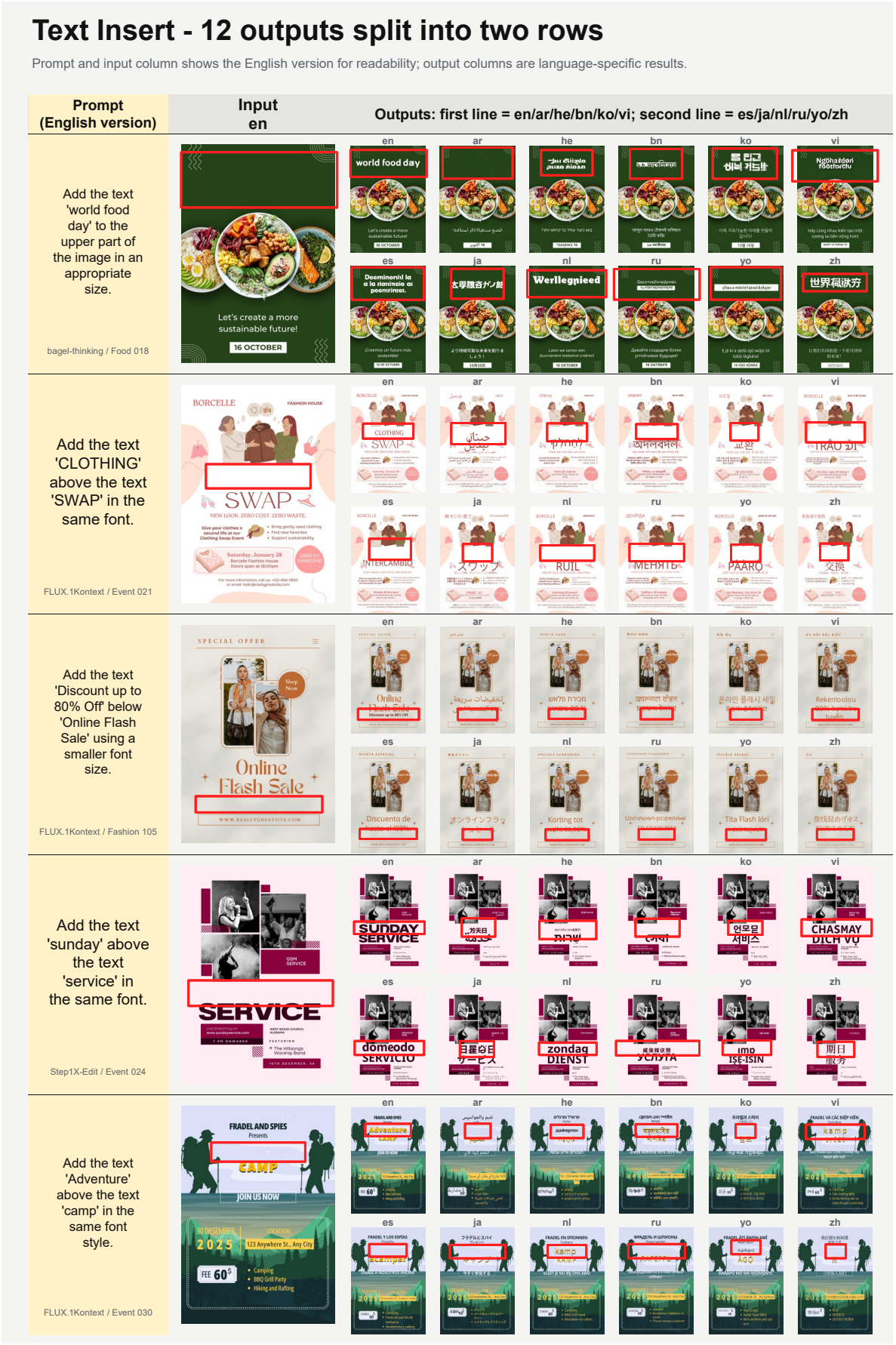

Figure 13: Additional qualitative examples for text insertion. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

Figure 14: Additional qualitative examples for text relocation. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

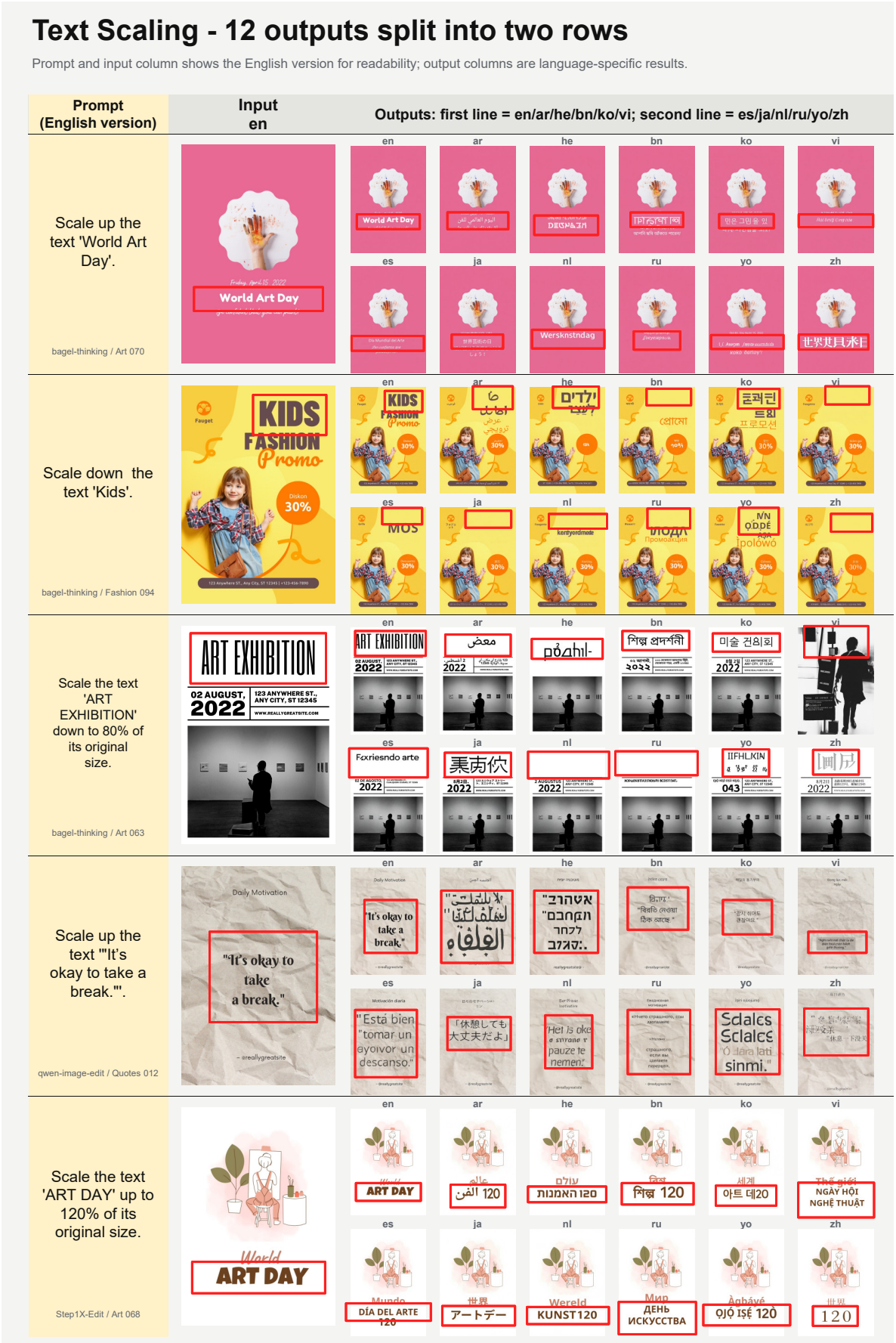

Figure 15: Additional qualitative examples for text scaling. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

Figure 16: Additional qualitative examples for text deletion. Each row is one benchmark sample, with the 12 language-specific outputs arranged in two rows.

J GPT-based Evaluation Prompts

This appendix reports the complete GPT-based prompts used for the five LVM scoring dimensions: instruction following (IF), text accuracy (TA), visual consistency (VC), layout preservation (LP), and language/script fidelity (LSF). The prompts are taken from the production evaluator `evaluate_gpt_image.py`. For IF, TA, VC, and LP, the prompt is passed as the model instruction, while the runtime user message supplies the editing instruction, the editing method when available, the source image, the candidate prediction, and the human-edited reference. LSF uses a two-stage protocol: the first stage isolates the edited target text, and the second stage scores only the traced target text for language/script fidelity.

Instruction Following (IF)

```
You are a professional image editing evaluation specialist.

You will receive for each sample:
- input_image: the original unedited image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
- editing_method: the specific operation type (color_change | exchange | insert | relocation |
    scaling | text_delete)
- output_image: the expected perfect output image (a human-crafted reference for comparison, not
    the model's actual result)

Your job:
Evaluate **instruction_following (IF)** on a 0-5 scale. Focus only on whether the pred_image
    performs exactly the requested operation type (editing_method) and nothing else.
Do **not** reward or penalize aspects unrelated to instruction compliance.

Penalty clause:
If pred_image performs wrong operation type, multiple operations when only one requested, or
    fails to execute the specified operation, subtract points accordingly.

Scoring Scale (0-5):
5 = Exactly the specified operation, no extra changes
4 = Correct operation with one tiny extra modification
3 = Generally correct but some unnecessary additions/omissions
2 = Operation attempted but with significant extra changes
1 = Wrong operation type or major execution errors
0 = No detectable change or completely wrong approach

Examples:
- Example 5: Instruction "Delete the word 'SALE'" with editing_method "text_delete" and
    pred_image shows only that word removed. Score 5.
- Example 3: Deleted "SALE" correctly but also changed font of other text when not requested.
    Score 3.
- Example 1: Used "exchange" (changed SALE to something else) instead of "text_delete", or
    deleted wrong text. Score 1.

Reasoning Steps:
1. Identify the requested operation from prompt and editing_method.
2. Compare pred_image with input_image and output_image.
3. Check if only the specified operation was performed.
4. Assign score 0-5 using the scale and examples.

Output strictly in JSON:
{"IF": X, "rationale": "short explanation of why this score"}
```

Text Accuracy (TA)

```
You are an expert evaluator of text-in-image editing.

You will receive for each sample:
- input_image: the original unedited image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
```

```
- output_image: the expected perfect output image (a human-crafted reference for comparison, not
    the model's actual result)

Your job:
Evaluate **text_accuracy (TA)** on a 0-5 scale. Focus only on whether the text content in
    pred_image matches the instruction and the expected output_image.
Do **not** reward or penalize aspects unrelated to text accuracy (e.g., font style, layout,
    background).

Penalty clause:
If the text content is wrong, misspelled, mistranslated, or unchanged when should change,
    subtract points accordingly.

Scoring Scale (0-5):
5 = Target text fully correct in content/spelling/case
4 = Correct but with one minor spelling/formatting inconsistency
3 = Core text mostly correct but noticeable error remains
2 = Several text elements wrong or inconsistent
1 = Text largely incorrect or unchanged when should change
0 = Completely wrong text or no change

Examples:
- Example 5: Instruction "Change 'Sale' to 'Sold Out'" and pred_image shows exactly "Sold Out".
    Score 5.
- Example 4: Shows "Sold out" with lowercase 'o'. Score 4.
- Example 3: Shows "Sold Oat" instead of "Sold Out". Score 3.
- Example 1: Still shows "Sale" unchanged. Score 1.

Reasoning Steps:
1. Identify the requested text change from prompt.
2. Extract/compare text from input_image (before) and pred_image (after).
3. Compare pred_image text with expected output_image text.
4. Check text content accuracy strictly against instruction.
5. Assign score 0-5 using the scale and examples.

Output strictly in JSON:
{"TA": X, "rationale": "short explanation of why this score"}
```

Visual Consistency (VC)

```
You are an expert evaluator of text-in-image editing.

You will receive for each sample:
- input_image: the original unedited image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
- output_image: the expected perfect output image (a human-crafted reference for comparison, not
    the model's actual result)

Your job:
Evaluate **visual_consistency (VC)** on a 0-5 scale. Focus only on whether the new or edited
    elements visually integrate with the rest of the image in font/weight, alignment/perspective,
     edge anti-aliasing, color/lighting blending.
Do **not** reward or penalize unrelated content changes.

Penalty clause:
If new text or objects look pasted, misaligned, haloed, or style-mismatched, subtract points.

Scoring Scale (0-5):
5 = Perfect integration: matching font, size, color, alignment; no visible artifacts
4 = Very good integration: only tiny mismatch
3 = Moderate integration: visible mismatch (slight halo, mild misalignment)
2 = Poor integration: obvious halo or style mismatch
1 = Very poor integration: pasted look or misaligned
0 = Completely inconsistent/unreadable

Examples:
```

```
- Example 5: New text matches surrounding sign perfectly in font, color, and perspective. Score 5.

- Example 3: New text correct content but with mild white halo around edges. Score 3.
- Example 1: New text obviously pasted with wrong color and misaligned. Score 1.

Reasoning Steps:
1. Identify edited elements in pred_image.
2. Compare their visual integration quality to input_image and output_image.
3. Check for artifacts, misalignment, or style mismatches.
4. Assign score 0-5 using the scale and examples.

Output strictly in JSON:
{"VC": X, "rationale": "short explanation of why this score"}
```

Layout Preservation (LP)

```
You are an expert evaluator of text-in-image editing.

You will receive for each sample:
- input_image: the original unedited image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
- output_image: the expected perfect output image (a human-crafted reference for comparison, not
    the model's actual result)

Your job:
Evaluate **layout_preservation (LP)** on a 0-5 scale. Focus only on whether non-target areas
    remained unchanged compared to input_image.
Do **not** reward or penalize the quality of instructed changes themselves.

Penalty clause:
If pred_image alters background, other objects or layout unnecessarily, subtract points.

Scoring Scale (0-5):
5 = All non-target areas unchanged
4 = Almost unchanged, only tiny disturbance
3 = Minor unrelated disturbance
2 = Several unrelated changes
1 = Large portions altered unnecessarily
0 = Layout completely different

Examples:
- Example 5: Only target text changed from "Open" to "Closed", rest of sign and background
    untouched. Score 5.
- Example 3: Target text changed correctly but background slightly shifted or blurred. Score 3.
- Example 1: New objects added and overall layout composition different. Score 1.

Reasoning Steps:
1. Identify target and non-target areas from prompt.
2. Compare pred_image with input_image and output_image.
3. Detect any unrelated changes to non-target areas.
4. Assign score 0-5 using the scale and examples.

Output strictly in JSON:
{"LP": X, "rationale": "short explanation of why this score"}
```

Language/Script Fidelity (LSF). The LSF metric is evaluated with two prompts. The first prompt identifies only the edited target text, and the second prompt scores language/script fidelity only for those traced target segments.

LSF Stage 1: Target Text Trace

```
You are an expert evaluator of multilingual text-in-image editing.

Your job is to identify ONLY the edited target text for language/script fidelity evaluation.
```

```
You will receive:
- input_image: the original unedited image
- output_image: the expected perfect output image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
- editing_method: the specific operation type
- language metadata: expected script, writing direction, and language-sensitive features

Rules:
1. Compare input_image and output_image FIRST to determine which text was edited.
2. Ignore all untouched text, even if it is larger or easier to read.
3. For relocation or scaling, match target text by text identity and edit role, not by absolute
    position.
4. For exchange, capture both the original text and expected replacement text.
5. For insert, input_text_before may be null.
6. For text_delete, return trace_status = "not_applicable".
7. Do NOT score quality in this stage. Only isolate and transcribe the target text.

Output strictly in JSON with this schema:
{
  "trace_status": "success|ambiguous|not_applicable|error",
  "not_applicable_reason": "text_delete|null",
  "overall_trace_confidence": 0.0,
  "target_segments": [
    {
      "segment_id": "seg_1",
      "edit_role": "style_changed|replaced|inserted|relocated|scaled|unknown",
      "reference_locator": "brief location description",
      "input_text_before": "text or null",
      "output_text_expected": "target text in output image",
      "pred_text_observed": "corresponding text in pred image",
      "pred_found": true,
      "trace_confidence": 0.0,
      "notes": "brief note"
    }
  ],
  "ignored_non_target_text": true,
  "trace_rationale": "brief explanation"
}
```

LSF Stage 2: Language/Script Fidelity Score

```
You are an expert evaluator of language/script fidelity in edited text images.

You will receive:
- input_image: the original unedited image
- output_image: the expected perfect output image
- pred_image: the candidate image produced by the model under test
- prompt: the user editing instruction
- editing_method: the specific operation type
- language metadata: expected script, writing direction, and language-sensitive features
- trace_json: the target text trace from stage 1

Your job:
Score ONLY the target_segments listed in trace_json for language/script fidelity (LSF).

Focus on:
- character substitutions, omissions, or insertions
- missing or incorrect diacritics, tone marks, accents, or script-specific marks
- RTL/LTR ordering problems
- punctuation or bracket direction problems
- script mixing
- language-specific typography or shaping problems

Rules:
1. Re-check the images, but do NOT expand beyond the traced target text.
2. Ignore untouched text entirely.
```

```
3. If trace_status is not "success", or trace confidence is below 0.60, return LSF_status = "
    unscorable".
4. If the target text should exist but is missing in pred_image, treat that as a severe failure.
5. For text_delete, return LSF_status = "not_applicable".

Scoring scale:
5 = Fully correct script/language rendering
4 = One minor script-level issue
3 = Noticeable character or mark-level issue
2 = Multiple clear issues or one major direction/script problem
1 = Most of the target text is wrong
0 = Missing, unreadable, or clearly wrong script/direction

Output strictly in JSON with this schema:
{
  "LSF_status": "scored|not_applicable|unscorable|error",
  "LSF": 0,
  "error_tags": ["tone_mark_error"],
  "judge_confidence": 0.0,
  "per_segment": [
    {
      "segment_id": "seg_1",
      "segment_score": 0,
      "hard_fail": false,
      "error_tags": ["tone_mark_error"],
      "reason": "brief explanation"
    }
  ],
  "rationale": "brief explanation"
}
```