FORRAIL: FORENSIC RETRIEVAL-AUGMENTED VISUAL IN-CONTEXT LEARNING FOR TRAINING-FREE IMAGE FORGERY LOCALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Image forgery localization (IFL) has become increasingly challenging as modern image editing models produce realistic and semantically coherent manipulations, while existing methods often rely on task-specific training and struggle to generalize to unseen manipulation techniques and generators. Multimodal large language models (MLLMs) offer broad visual reasoning capabilities and cross-domain knowledge, yet their potential for precise, training-free forgery localization remains largely unexplored. In this work, we investigate whether a frozen MLLM can perform pixel-level IFL through visual in-context learning. We propose Retrieval-Augmented Visual In-Context Learning for IFL (ForRAIL), a training-free framework that retrieves informative forgery demonstrations for each query to guide localization without updating model parameters. To improve demonstration selection, we further introduce CoSFoR, a coarse-to-fine semantic–forgery retrieval strategy that jointly considers semantic relevance and manipulation-sensitive characteristics. Extensive experiments across diverse out-of-domain datasets, generators, and multiple MLLM backbones show that ForRAIL substantially outperforms vanilla MLLMs and existing training-free MLLM-based approaches and exceeds the best specialized trained forensic models 0.03 IoU on OOD average. It also maintains strong localization performance across different tampered-region sizes and under common image perturbations, including Gaussian blur, Gaussian noise, JPEG compression, and scale. These results establish retrieval-augmented visual in-context learning as a promising training-free paradigm for generalizable image forgery localization.

1 INTRODUCTION

Recent advances in generative image editing have made local manipulations increasingly realistic and semantically coherent. Modern editors can selectively modify objects, attributes, or regions while preserving most of the original image, making manipulated content increasingly difficult to identify through visual inspection alone. Image forgery localization (IFL) therefore goes beyond determining whether an image has been manipulated and aims to identify manipulated regions at the pixel level, providing fine-grained evidence for forensic analysis.

Despite substantial progress, generalization remains a central challenge for IFL. Existing systems typically learn forensic representations from specific manipulation distributions, and their performance can degrade when transferred to unseen generators, editing pipelines, and datasets. Representative approaches such as PSCC-Net (Liu et al., 2022), TruFor (Guillaro et al., 2023), Mesorch (Zhu et al., 2025), and DetectiveSAM (Lek et al., 2026) rely on task-specific training to acquire forensic representations. Adapting such systems to newly emerging manipulation techniques generally requires additional training, fine-tuning, or newly collected forgery data. This dependence becomes increasingly restrictive as image editing models and manipulation distributions continue to evolve.

Multimodal large language models (MLLMs) provide a different opportunity. Pretrained on large-scale and diverse visual–textual data, they possess broad semantic knowledge and flexible visual reasoning that are not tied to a single forgery dataset. Recent MLLM-based forensic systems such as SIDA (Huang et al., 2025) and FakeShield (Xu et al., 2025) demonstrate that MLLMs can support

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.

forgery detection, localization, and explanation. However, these approaches still rely on task-specific training, instruction tuning, or specialized localization components. Meanwhile, Foresee (Zuo et al., 2025), a training-free approach, suggests that vanilla MLLMs already possess useful intrinsic forensic capabilities. Nevertheless, recognizing suspicious content does not necessarily provide reliable estimates of the location and extent of manipulated regions. This motivates investigating how informative visual demonstrations can support spatial localization across unseen editing distributions.

Visual in-context learning (VICL) offers a promising route toward such inference-time adaptation. By conditioning a frozen MLLM on a small set of multimodal input–output demonstrations, VICL can communicate task structure without updating model parameters. For forensic localization, however, demonstrations must convey not only why a region is suspicious, but also where it lies and how its spatial extent is delineated. We therefore use polygon-based demonstrations that associate manipulated images with polygon annotations and forensic rationales, providing explicit image–annotation correspondences for spatially grounded prediction. Even so, the effectiveness of such demonstrations depends critically on their relevance to the query. For IFL in particular, semantic similarity and forensic relevance are not equivalent: semantically similar images may exhibit different manipulation traces, while visually different images may share similar forensic characteristics. Conversely, relying solely on forensic similarity may overlook the semantic context needed for informative visual demonstrations. Consequently, effective demonstration selection should consider both semantic compatibility and manipulation-sensitive regional evidence.

To address these challenges, we propose **ForRAIL**, a retrieval-augmented visual in-context learning framework for generalizable image forgery localization. ForRAIL maintains a fixed pool of localization-aware demonstrations containing manipulated images, polygon annotations, and forensic rationales, from which informative examples are dynamically retrieved for each query image. To improve demonstration selection, we introduce **CoSFoR**, a coarse-to-fine semantic–forgery retrieval strategy that first identifies semantically compatible candidates and then incorporates manipulation-sensitive regional evidence for reranking. Conditioned on the retrieved demonstrations, the frozen MLLM predicts the locations and spatial extents of manipulated regions as structured polygons. These predictions are subsequently converted into pixel-level masks based on the Whole Object Label (WOL): polygons corresponding to complete objects are refined using pretrained SAM2 (Ravi et al., 2025), while the remaining polygons are directly rasterized.

Extensive experiments across diverse out-of-domain datasets and generators show that ForRAIL consistently improves over vanilla MLLMs and existing training-free MLLM-based approaches while remaining competitive with specialized, task-trained forensic models. Notably, these results are obtained using the same demonstration pool across OOD datasets, without incorporating target-domain examples or updating model parameters. Ablation studies further show that query-dependent retrieval improves localization performance over fixed and random demonstration selection, highlighting the importance of selecting informative examples rather than merely providing additional data. Similar improvements are observed across multiple MLLM backbones, suggesting that the benefits of retrieval-augmented VICL are not limited to a single model. ForRAIL also maintains strong localization performance across a range of tampered-region sizes and under common image degradations, including Gaussian blur, Gaussian noise, JPEG compression, and scale. Together, these findings highlight the potential of retrieval-augmented VICL for generalizable image forgery localization without task-specific parameter updates.

Our contributions are summarized as follows:

- We propose **ForRAIL**, a visual in-context learning framework for generalizable image forgery localization that combines localization-aware demonstrations, structured polygon prediction, and selective mask refinement without task-specific parameter updates.
- We introduce **CoSFoR**, a coarse-to-fine semantic–forgery retrieval strategy that selects informative demonstrations by jointly considering high-level semantic compatibility and manipulation-sensitive forensic characteristics.
- Extensive evaluations across diverse out-of-domain datasets, generators, MLLM backbones, tampered-region sizes, and image degradations show that ForRAIL consistently improves over vanilla MLLMs and existing training-free MLLM-based approaches while remaining competitive with specialized, task-trained forensic models.

2 Related Work

Image forgery localization. Image forgery localization aims to identify manipulated regions at the pixel level. Early methods exploit forensic traces and spatial inconsistencies (Wu et al., 2019; Kwon et al., 2021; Dong et al., 2022; Hu et al., 2020; Hao et al., 2021; Wang et al., 2022), later work develops stronger forensic representations and architectures (Guillaro et al., 2023; Li et al., 2024b; Zhu et al., 2025; Liu et al., 2025b; Sheng et al., 2025), and recent approaches improve generalization through contrastive learning, diffusion priors, unified representations, non-semantic features, and multi-spectral cues (Zhou et al., 2023; Yu et al., 2024; Li et al., 2024a; Su et al., 2025; Nam et al., 2025). Foundation-model-based methods such as SAFIRE (Kwon et al., 2025), Omni-IML (Qu et al., 2026), and DetectiveSAM (Lek et al., 2026) further expand localization capability. Unlike these trained or adapted models, ForRAIL keeps the MLLM frozen and adapts through query-specific demonstrations at inference time.

MLLM-based and training-free forensics. MLLMs and VLMs have been used for forgery detection, localization, and explanation (Xu et al., 2025; Huang et al., 2025; Zhang et al., 2024; Sun et al., 2026), sometimes with explicit forensic evidence from frequency features, verification modules, or vision-language reasoning (Liu et al., 2026; Zhang et al., 2026; Sheng et al., 2026), but these systems rely on task-specific training or specialized modules. Training-free alternatives include diffusion-based localization (Zhang et al., 2025b), few-shot prompting of MLLMs (He et al., 2025), intrinsic forensic prompting and feature matching (Zuo et al., 2025), and retrieval of textual forensic rules (Chen et al., 2025b). In contrast, ForRAIL retrieves visual input–output demonstrations containing images, localization annotations, and forensic rationales, enabling a frozen MLLM to infer localization behavior directly from examples.

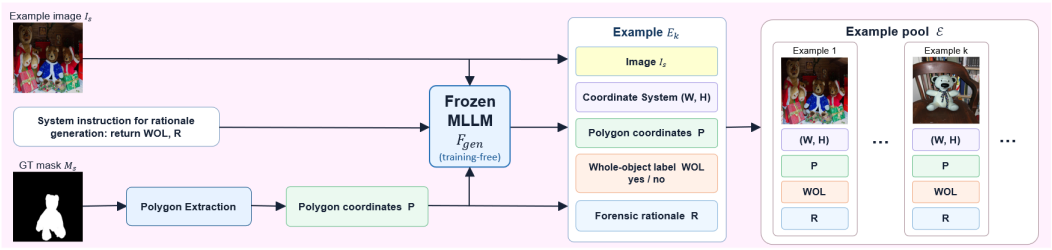
Visual in-context learning and demonstration retrieval. Multimodal in-context learning adapts without parameter updates (Zhao et al., 2024a), but performance hinges on choosing good demonstrations (Zhang et al., 2023b). Prior work studies retrieval, reranking, and composition for LVLMs (Zhou et al., 2024), plus model-aware, RL, exploration–exploitation, and coreset-based selection (Peng et al., 2024; Wang et al., 2025; Chen et al., 2025a; Wang et al., 2026). Beyond generic similarity, effective selection needs task-specific relationships (Liu et al., 2025a). In IFL, semantic similarity may not match forensic similarity, so CoSFoR pairs semantic relevance with manipulation-sensitive evidence.

3 ForRAIL

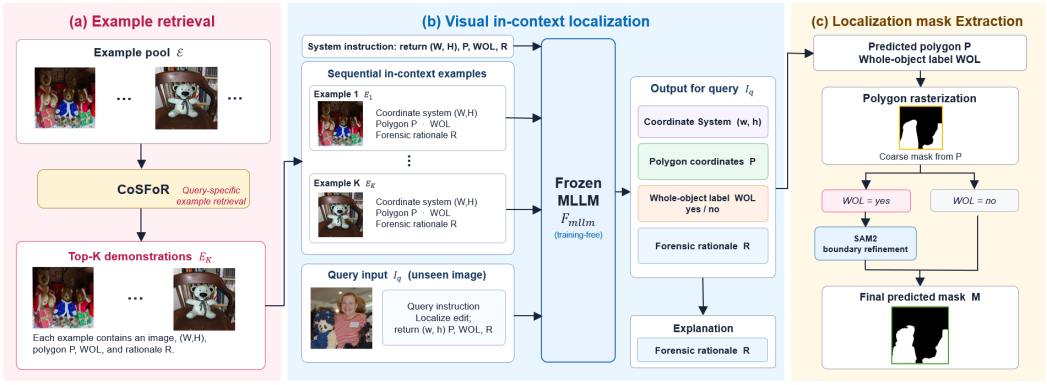
We define image forgery localization as follows. Given an image $\mathcal{I}_q \in \mathbb{R}^{3\times H\times W}$ with 3 channels, height H, and width W, the goal is to predict a binary mask $\mathcal{M} \in \{0,1\}^{H\times W}$, where $\mathcal{M}_{i,j} = 1$ if pixel (i,j) of $\mathcal{I}_q$ has been tampered with, edited, or manipulated, and $\mathcal{M}_{i,j} = 0$ otherwise. In this work, we focus on images edited by diffusion-based models, currently the most powerful family of image editing methods. ForRAIL consists of an offline stage that constructs the examples and an online stage that localize tampered regions with MLLM. The online stage includes three phases: example retrieval, MLLM Localization, and localization mask extraction. CoSFor in the example retrieval phase further breaks down into the semantic shortlist stage followed by the forensic-aware reranking stage.

3.1 Offline: Example pool construction

Figure 1a illustrates this stage. Each example E_k is built from a source tampered image $\mathcal{I}_s$ and its ground-truth mask $\mathcal{M}_s$. First, we convert the ground-truth mask $\mathcal{M}_s$ into a set of polygon coordinates $P = \{P_p\}_{p=1}^{N_P}$, where N_P denotes the number of polygons. Each polygon is represented as $P_p = \{P_{p,i}\}_{i=1}^{N_p}$, where p is the polygon index, i the vertex index, and N_p the number of vertices in P_p; each vertex is given by its spatial coordinate $P_{p,i} = (x_{p,i}, y_{p,i})$. The area enclosed by a polygon corresponds to a tampered region. Fragmented regions that do not belong to the main tampered areas are discarded. Second, the source tampered image $\mathcal{I}_s$ and the polygon coordinates P are fed into a frozen MLLM $\mathcal{F}_{gen}$, which produces a rationale R explaining why each region is regarded as tampered, together with a Whole Object Label $WOL \in \{\text{yes}, \text{no}\}^{N_P}$ indicating whether each



(a) Offline example pool construction pipeline for ForRAIL.



(b) Online MLLM Localization pipeline for ForRAIL.

Figure 1: (a) describes the offline example construction procedure for ForRAIL. (b) describes the online MLLM localization for ForRAIL. The MLLM localization have three phases: (a)example retrieval, (b)visual in-context localization, and (c)localization mask extraction.

tampered region forms a complete object that SAM can readily segment. Finally, we assemble the source tampered image $\mathcal{I}_s$, the coordinate system (H, W), the polygon coordinates P, the Whole Object Label WOL, and the rationale R into an example E_k. Repeating this procedure over a set of source tampered images yields the example pool $\mathcal{E}$.

3.2 Online: MLLM Localization

Localizing tampered regions with an MLLM requires the example pool $\mathcal{E}$ constructed offline and a query image $\mathcal{I}_q$. As shown in Figure 1b, the online stage consists of three phases: example retrieval, visual in-context localization, and localization mask extraction.

Phase (a): Example retrieval We apply CoSFoR, our example retrieval method, to select K suitable examples from the example pool $\mathcal{E}$, forming the retrieved example set $\mathcal{E}_K$. CoSFoR will be further illustrated in section 3.3.

Phase (b): MLLM localization For each query sample, we feed the system instruction Sys, the retrieved example set $\mathcal{E}_K$, and the query image $\mathcal{I}_q$ to the MLLM $\mathcal{F}_{mllm}$ over multiple turns. The MLLM replies 'Understood' after receiving the system instruction and 'Example k received' after each example. Once the query image $\mathcal{I}_q$ is received, it outputs the coordinate system (w, h) in which the polygon coordinates $P_{p,i}$ are expressed, the predicted polygons P, the Whole Object Label WOL, and a rationale R explaining why the predicted regions are considered tampered.

Phase (c): Localization mask extraction In the final phase, we parse the predicted polygons P and the Whole Object Label WOL from the MLLM response. For each polygon P_p, if $WOL =$ yes, it is converted into a bounding box B_p and passed to SAM2 for refinement; if $WOL =$ no, it is rasterized directly onto the output mask $\mathcal{M}$. By applying this mechanism, the predicted object-

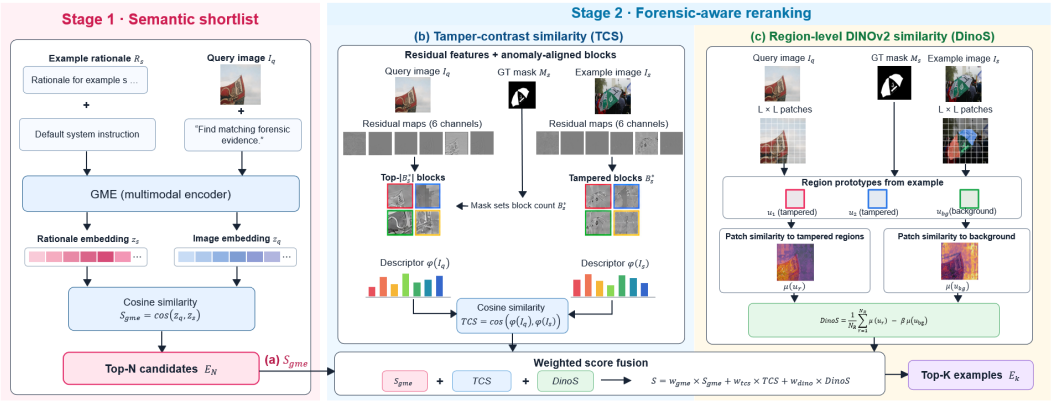


Figure 2: Overview of CoSFoR. Stage 1 uses GME (Zhang et al., 2025a) to retrieve a semantic shortlist through cross-modal similarity between the query image and candidate forensic rationales. Stage 2 reranks the shortlisted candidates by combining GME similarity, Tampered-Contrast Similarity (TCS), and region-level DINOv2 similarity (DinoS). The top-K candidates are selected as demonstrations for visual in-context localization.

like regions will become more precise, while the predicted non-object regions will not be modified falsely by SAM.

3.3 CoSFoR: Coarse-to-fine Semantic-Forgery Retrieval

As illustrated in Figure 2, CoSFoR proceeds in two stages. Since MLLMs are more sensitive to semantic information, the first stage performs a semantic similarity check. To further account for low-level forgery traces, the second stage computes a semantic-forgery hybrid similarity.

Stage 1: Semantic shortlist. On both the query and the example side, we take the last-layer hidden state of the final token produced by GME (Zhang et al., 2025a) as the feature. On the query side, the input is the image $\mathcal{I}_q$ together with our instruction; on the example side, it is the rationale R together with GME's original instruction. We then compute the similarity S_{gme} between the query image $\mathcal{I}_q$ and each example E_k, and keep the top-N examples $\mathcal{E}_N$ as the intermediate result of Stage 1.

Stage 2: Forensic-aware reranking. In Stage 2, every candidate retained by Stage 1 is re-scored by the semantic-forgery hybrid similarity

$$S = w_{gme} \times S_{gme} + w_{tcs} \times TCS + w_{dino} \times DinoS, \tag{1}$$

where S_{gme} is the semantic similarity from Stage 1. The acquisitions of TCS and $DinoS$ are illustrated below.

Tampered-Contrast Similarity (TCS). TCS measures whether the query and a candidate example carry similar low-level forgery traces (Algorithm 1). Both $\mathcal{I}_q$ and $\mathcal{I}_k$ are converted into N_s=6 residual maps (one noise residual, three SRM (Fridrich & Kodovsky, 2012), one Noiseprint++ (Guillaro et al., 2023), and one Laplacian high-pass), each partitioned into $G \times G$ blocks. For block b and map s, we compute the residual energy $e_{b,s} = \log(1 + \mathrm{std}(r_s|_b))$ and its deviation from the image's median block, $d_{b,s} = e_{b,s} - \mathrm{med}_{b'} e_{b',s}$, which cancels the camera and compression baseline shared by all blocks. On the example side, we select the blocks B_k^* more than half covered by the ground-truth mask; on the query side, where no mask exists, we keep the $|B_k^*|$ blocks with the largest self-anomaly $a_b = \|(d_{b,s}/(\sigma_s + \epsilon))_{s=1}^{N_s}\|_2$, where $\sigma_s = \mathrm{std}_b\, d_{b,s}$, so that both sides summarize regions of comparable size. The selected blocks are aggregated into an 18-dimensional descriptor of the signed and absolute mean deviations and the global dispersion σ_s of each map, and TCS is the cosine similarity between the two descriptors.

Region-DINOv2 Score ($DinoS$). $DinoS$ measures whether the query contains content resembling the candidate's tampered regions in DINOv2 (Oquab et al., 2023) feature space (Algorithm 2).

Both images are encoded into $L \times L$ grids of ℓ_2-normalized patch tokens. On the example side, the mask partitions the grid into up to $N_{\max}$ tampered regions (the largest connected components) and one background region, summarized by normalized mean tokens u_r and u_{bg}. Each is matched against all query patches, with $\mu(\cdot)$ denoting the mean of the top-κ cosine similarities, which is more robust to outliers than a single maximum:

$$DinoS(\mathcal{I}_q, E_k) = \frac{1}{N_R} \sum_{r=1}^{N_R} \mu(u_r) - \beta\, \mu(u_{bg}). \tag{2}$$

The background term prevents a shared scene alone from producing a high score. The matching is asymmetric by design: regions are defined only on the example, whose mask is known, and searched for anywhere in the query.

4 Experiments

Implementation details. To build our example pool, we sample 1k instances each from the training splits of SID-Set (Huang et al., 2025) and MagicBrush (Zhang et al., 2023a), and generate rationales and WOLs with Claude Opus 5 (Anthropic, 2026a) using high reasoning effort. Unless stated otherwise, we retrieve $K = 5$ examples (ablation in Appendix K); system instructions are given in Appendix C. Each session is initialized with the system instruction and the K examples together with their corresponding MLLM replies, so the MLLM only needs to generate the localization for the query image. We evaluate three backbones: Claude Opus 5.5 (Anthropic, 2026b) and GPT-5.6 Luna (OpenAI, 2026), both via the official batch API with low reasoning effort, and Gemma4 31B-IT (Team et al., 2026) in thinking mode with Q4_K_M quantization via Ollama. Unless noted otherwise, Gemma4 31B-IT serves as the backbone. Full configurations and query dates are listed in Appendix D.

Evaluation datasets & metrics. We evaluate ForRAIL on SID-Set (Huang et al., 2025), MagicBrush (Zhang et al., 2023a), UltraEdit (Zhao et al., 2024b), COCOGlide (Guillaro et al., 2023), TGIF (six subsets) (Mareen et al., 2024; 2026), and DiffSeg30k (Ci et al., 2025), which cover diverse generators and editing mechanisms. Since DetectiveSAM is pretrained on SID-Set and MagicBrush, we treat them as in-domain and the rest as out-of-domain (OOD). Ablations use a 250-image mixed dataset drawn from five OOD subsets (Appendix E). We report pixel-level mean IoU and mean F1.

Baselines. ForRAIL was compared against comprehensive previous state-of-the-art baselines, including DetectiveSAM (Lek et al., 2026), TruFor (Guillaro et al., 2023), PSCC-Net (Liu et al., 2022), Mesorch (Zhu et al., 2025), SIDA (Huang et al., 2025), and FakeShield (Xu et al., 2025). We use their official weights and for DetectiveSAM, we use the state-of-the-art model version, which is finetuned on its FLUX and Qwen datasets. Besides the trained methods, we also compare against the current MLLM based training-free method, Foresee (Zuo et al., 2025), and vanilla MLLMs mentioned in **Implementation details**.

4.1 Results

Comparison with state-of-the-art methods. Table 1 compares ForRAIL with trained and training-free methods across three MLLM backbones. Among training-free methods, ForRAIL is the best on every OOD dataset for every backbone. It raises the OOD-average IoU of the vanilla MLLM by 0.12–0.15 (e.g., Gemma4 31B: 0.1760 $\rightarrow$ 0.3222; Opus 5.5: 0.2593 $\rightarrow$ 0.3803) and exceeds Foresee on the same backbone by 0.10–0.15 IoU. Part of Foresee's deficit stems from empty predictions, whose frequency varies with the backbone (e.g., 45% of UltraEdit with Gemma4 31B, 45% of COCOGlide with Opus 5.5). Restricting the comparison to the images on which Foresee does produce a mask (Appendix I) removes this effect but not the ranking: ForRAIL still leads on every dataset and backbone, by 0.05–0.11 OOD-average IoU, so the advantage reflects the retrieved demonstrations rather than Foresee's failure mode alone.

Against trained methods, ForRAIL with Opus 5.5 backbone surpasses every trained method on the OOD average (0.3803 vs. 0.3491 IoU; 0.4639 vs. 0.4427 F1) and is the best method overall on UltraEdit (0.3595 IoU) and TGIF (0.4614 IoU). With GPT5.6 Luna and Gemma4 31B backbone, they achieve close OOD average performance against best trained methods.

Table 1: Comparison of image forgery localization performance across different datasets. The best scores are **bold**; trained and training-free methods are annotated separately.

Methods	SID-Set		MagicBrush		UltraEdit		COCOGlide		TGIF		DiffSeg30k		OOD Avg.		OOD Worst-Case	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Trained Methods																
DetectiveSAM$_{sota}$	**0.4865**	**0.5809**	**0.5770**	**0.6948**	**0.3373**	**0.4298**	**0.4857**	**0.5864**	0.1999	0.2652	0.3734	0.4895	**0.3491**	**0.4427**	**0.1999**	**0.2652**
TruFor	0.1865	0.2647	0.2696	0.3751	0.0865	0.1197	0.2917	0.3596	**0.4278**	**0.5063**	0.5684	0.6632	0.3436	0.4122	0.0865	0.1197
PSCC-Net	0.0261	0.0403	0.1888	0.2807	0.1750	0.2322	0.3324	0.4215	0.0306	0.0436	**0.6128**	**0.7298**	0.2877	0.3568	0.0306	0.0436
Mesorch	0.1293	0.1673	0.1375	0.1990	0.0504	0.0674	0.3819	0.4494	0.1280	0.1623	0.2524	0.3209	0.2032	0.2500	0.0504	0.0674
SIDA	0.3315	0.3994	0.2136	0.3207	0.0481	0.0587	0.1419	0.1725	0.0473	0.0589	0.1176	0.1601	0.0887	0.1126	0.0473	0.0587
FakeShield	0.2137	0.2628	0.1490	0.2214	0.2235	0.2839	0.2688	0.3583	0.0179	0.0276	0.0999	0.1342	0.1525	0.2010	0.0179	0.0276
Training-Free Methods and Vanilla MLLMs																
Gemma4 31B	0.2203	0.2943	0.1863	0.2849	0.1867	0.2622	0.2119	0.2958	0.1619	0.2217	0.1436	0.2193	0.1760	0.2498	0.1436	0.2193
Foresee(Gemma4 31B)	0.2263	0.2745	0.1436	0.1503	0.1680	0.1990	0.2937	0.3507	0.2767	0.3329	0.1478	0.2097	0.2216	0.2706	0.1478	0.1990
Ours(Gemma4 31B)	0.3811	0.4431	0.2338	0.3390	0.3277	0.4031	0.3515	0.4394	0.3581	0.4299	0.2514	0.3498	0.3222	0.4056	0.2514	0.3498
GPT-5.6 Luna	0.1865	0.2399	0.0733	0.1131	0.1493	0.2059	0.2201	0.2913	0.2628	0.3239	0.1904	0.2767	0.2057	0.2745	0.1493	0.2059
Foresee(GPT-5.6 Luna)	0.2161	0.2601	0.1076	0.1623	0.2186	0.2619	0.2895	0.3462	0.2131	0.2489	0.1560	0.2187	0.2193	0.2689	0.1560	0.2187
Ours(GPT-5.6 Luna)	0.3944	0.4441	0.1801	0.2655	0.2955	0.3620	0.3487	0.4292	0.3925	0.4592	0.2571	0.3572	0.3235	0.4019	0.2571	0.3572
Opus 5.5	0.3236	0.3859	0.2312	0.3338	0.2324	0.3009	0.2657	0.3439	0.3201	0.3811	0.2191	0.3101	0.2593	0.3340	0.2191	0.3009
Foresee(Opus 5.5)	0.3196	0.3785	0.1807	0.2683	0.2971	0.3535	0.2063	0.2462	0.2374	0.2765	0.1743	0.2459	0.2288	0.2805	0.1743	0.2459
Ours(Opus 5.5)	**0.4722**	**0.5291**	**0.2499**	**0.3555**	**0.3595**	**0.4338**	**0.4251**	**0.5094**	**0.4614**	**0.5342**	**0.2752**	**0.3780**	**0.3803**	**0.4639**	**0.2752**	**0.3780**

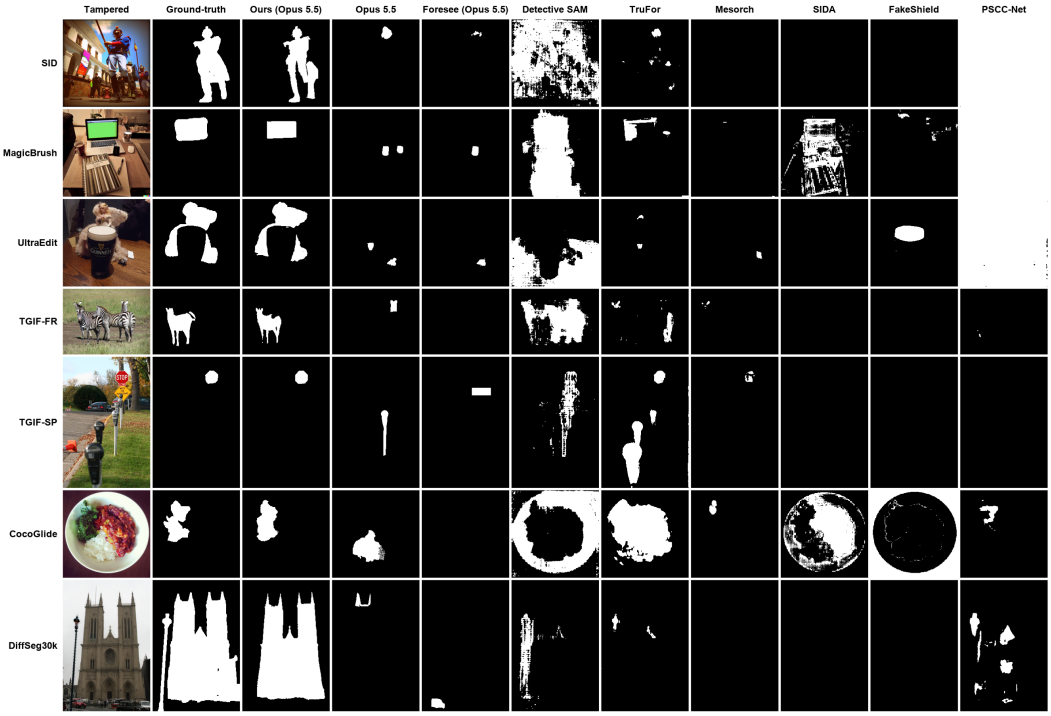


Figure 3: Qualitative comparison of image forgery localization across evaluation datasets. Each row shows a manipulated image, its ground-truth mask, and the predicted masks from ForRAIL and the baselines. Opus 5.5 is used as the MLLM backbone for ForRAIL, Foresee, and vanilla MLLM inference. The TGIF examples are selected from the FLUX subsets. White pixels indicate manipulated regions.

The clearest advantage appears in the worst-case column. Trained methods are uneven across domains: TruFor falls to 0.0865 IoU on UltraEdit and PSCC-Net to 0.0306 on TGIF, so their averages are carried by a few favorable datasets, and even DetectiveSAM, the most balanced of them, bottoms out at 0.1999 on TGIF. ForRAIL never falls below 0.2514 IoU with any backbone, and reaches 0.2752 with Opus 5.5, 0.0753 above the best trained method. This flat profile holds across backbones and is precisely what deployment requires, where the source generator of a query image is unknown in advance.

Visualization. Figure 3 shows the qualitative results of each baseline and ForRAIL. We observe that previous baselines fail to localize most of the tampered regions. DetectiveSAM appears to have

Table 2: Comparison of the performance on fully regenerated samples across TGIF subsets. The best scores are **bold**; trained and training-free methods are annotated separately.

Methods	TGIF-FLUX		TGIF-FLUX-Fill		TGIF-SD2	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Trained Methods						
DetectiveSAM$_{\text{sota}}$	0.1941	0.2557	**0.1379**	**0.1911**	**0.3260**	**0.4139**
TruFor	**0.2086**	**0.2802**	0.1206	0.1742	0.1784	0.2380
PSCC-Net	0.0069	0.0122	0.0095	0.0169	0.0207	0.0337
Mesorch	0.1530	0.1883	0.0758	0.0982	0.1291	0.1705
SIDA	0.0380	0.0461	0.0199	0.0255	0.1052	0.1317
FakeShield	0.0036	0.0066	0.0072	0.0117	0.0711	0.1078
Training-Free Methods and Vanilla MLLMs						
Gemma4 31B	0.1515	0.2062	0.1307	0.1828	0.2465	0.3323
Foresee(Gemma4 31B)	0.2966	0.3369	0.2411	0.2823	0.3310	0.3934
Ours(Gemma4 31B)	0.3617	0.4264	0.3114	0.3778	0.4096	0.4999
GPT-5.6 Luna	0.2483	0.3041	0.2393	0.2971	0.2964	0.3728
Foresee(GPT-5.6 Luna)	0.2043	0.2317	0.2086	0.2451	0.3186	0.3769
Ours(GPT-5.6 Luna)	0.3794	0.4324	0.3182	0.3789	0.4461	0.5356
Opus 5.5	0.2941	0.3406	0.2978	0.3559	0.4107	0.4954
Foresee(Opus 5.5)	0.2182	0.2458	0.1806	0.2130	0.4081	0.4798
Ours(Opus 5.5)	**0.4701**	**0.5306**	**0.4133**	**0.4842**	**0.5210**	**0.6131**

Table 3: IoU grouped by the relative size of the ground-truth tampered region. Results are computed on the pooled OOD set, i.e., the union of UltraEdit, CocoGlide, TGIF, and DiffSeg30k from Table 1 (8,368 images); the "All" row is therefore a sample-weighted average over the pooled set and differs from the per-dataset OOD average in Table 1. n denotes the number of samples in each group. DSAM denotes DetectiveSAM$_{\text{sota}}$. The best scores are **bold**.

GT size	n	DSAM	Vanilla			Ours		
			Gemma4 31B	GPT-5.6 Luna	Opus 5.5	Gemma4 31B	GPT-5.6 Luna	Opus 5.5
<1%	664	0.0451	0.0640	0.1546	0.1937	0.1616	0.2369	**0.3293**
1–5%	1287	0.1641	0.1744	0.2854	0.3555	0.3444	0.3720	**0.4603**
5–20%	2203	0.3203	0.2195	0.2646	0.3147	0.3517	0.3698	**0.4075**
20–50%	2453	**0.3967**	0.1710	0.1659	0.2201	0.3108	0.2976	0.3304
>50%	1761	**0.4545**	0.1329	0.0985	0.1614	0.2923	0.2329	0.2802
All	8368	0.3251	0.1678	0.1952	0.2514	0.3110	0.3096	**0.3600**

the best performance among the baselines, however, it tends to be distracted by the surrounding noise or uniform appearances.

Fully regenerated robustness. Table 2 evaluates the fully regenerated (FR) samples of the three TGIF subsets, where the entire image is re-generated by the generator and no splicing boundary remains. ForRAIL is the best method on every subset in both IoU and F1. With Opus 5.5 backbone, it exceed the best trained method by 0.20–0.28 IoU and Foresee by 0.20 IoU on average. This result also accounts for the one dataset in Table 1 on which a trained method leads: TruFor's advantage on TGIF comes from spliced samples, whose boundary artifacts it is trained to detect, and it falls to 0.12–0.21 IoU once those boundaries are removed. ForRAIL does not rely on such artifacts and therefore holds up when the whole image is regenerated.

Performance across Different Ground-Truth Size of Tampered Regions. Table 3 groups the pooled OOD set by the relative area of the ground-truth tampered region. Small regions are the hard case, and they expose a sharp failure mode in trained localizers: DetectiveSAM's IoU falls monotonically from 0.4545 on regions above 50% to 0.0451 below 1%, so its strong pooled average is carried almost entirely by large edits. ForRAIL degrades far more gracefully: with Opus 5.5 as the backbone, it reaches 0.3293 IoU on regions below 1% (7.3× DetectiveSAM) and stays within 0.28–0.46 across all groups, while demonstrations raise the vanilla baseline by 0.09–0.14 IoU in every group. Similar trends hold for Gemma4 31B-IT and GPT-5.6 Luna. DetectiveSAM retains an

Table 4: Ablation studies on the Mixed dataset with Gemma4 31b. (a) CosFoR Variants. btm-5 indicates using bottom-5 instead of top-5 for w_{gme}. (b) In-context example composition. (c) Visual in-context variants. The best scores are **bold**.

(a) CoSFoR Variants

w_{gme}-w_{tcs}-w_{dino}	IoU ↑	F1 ↑
100-00-00	0.3266	0.4057
100-00-00(btm-5)	0.2635	0.3307
00-50-25	0.3448	0.4233
25-00-25	0.3612	0.4393
25-50-00	0.3343	0.4133
50-25-25	0.3374	0.4174
25-25-50	**0.3646**	**0.4427**
25-50-25 (Ours)	0.3496	0.4275

(b) Example composition

Setting	IoU ↑	F1 ↑
Rationale	0.2513	0.3153
Image + Rationale	0.3143	0.3918
polygons + Rationale	0.2883	0.3546
Image + Polygons	0.3045	0.3775
Ours	**0.3496**	**0.4275**

(c) Visual in-context learning variants

Setting	IoU ↑	F1 ↑
w/o example	0.1680	0.2330
Fixed example	0.2787	0.3512
Random example	0.2946	0.3697
Ours	**0.3496**	**0.4275**

advantage on regions above 20%, where a large tampered area offers abundant low-level evidence. F1 results are provided in Appendix G.

Impact of external perturbation. We evaluate the robustness of the methods under external Gaussian blur, Gaussian noise, JPEG compression, and scale pre-processing perturbations as in Figure 8 (Appendix L). ForRAIL appears to be solid under perturbed input.

4.2 Ablation Study

Different score fusion weights in CoSFoR. Table 4a demonstrates the performances for different weights of w_{gme}, w_{tcs}, and w_{dino} in CoSFoR step 2. 100-00-00 variants are equal to step-1-only in CoSFoR. The 0.0631 IoU gap between 100-00-00 and 100-00-00(btm-5) shows that semantic information plays a critical role in our task. The performance of 25-25-50 demonstrates that each component of our example is important.

Importance of the components in the examples. Table 4b shows that images, polygons, and rationales are critical components in examples. Missing either of these components will result in degradation.

Different visual in-context learning styles. Table 4c demonstrates that ForRAIL is the most suitable one among visual in-context learning variants, exceeding the random example setting by 0.06 IoU.

5 Conclusion

ForRAIL demonstrates the MLLM capablility of conducting Image forgery Localization, reaching 0.3803 mean out-of-distribution IoU with Opus 5.5 MLLM backbone and exceeding the best trained method by 0.03 IoU. It also maintains 0.28 IoU on OOD worst-case, showing strong robustness toward cross-dataset, cross-generator scenario. We further exhibit the robustness of ForRAIL under fully regenerated cases, tiny artifacts, and common pre-processing perturbations. Overall, we present a training-free IFL conducted by MLLMs via visual in-context learning, showing strong performance and robustness against current trained methods.

Limitations. ForRAIL requires a long inference time dominated by the MLLM backbone. It takes roughly 60-70 seconds per sample with Gemma4 31B-IT backbone on a single RTX 5090 GPU, while DetectiveSAM can finish 29 samples per second on an NVIDIA H100 according to its paper. ForRAIL also relies deeply on the MLLMs' ability to learn from visual context.

AI USE STATEMENT

In this work, we used generative AI tools for helping develop theoretical models or conceptual frameworks, proposing and refining hypotheses, designing and providing feedback on research methodology and experiments, implementing methods, cleaning and reformating dataset, interpreting results. We have not used generative AI tools for generating synthetic data sets, assisting with translation. Formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, supporting qualitative and thematic data analysis, are not applicable to this work. Additionally, we used generative AI tools for create or modify scientific figures or images, create or edit software code, draft parts of a research paper, summarize or analyse existing literature, discover research topics or identify gaps, brainstorming, sourcing/searching for information, edit a research paper to improve readability, identify relevant literature, propose a title or keywords for a research paper. We have reviewed all AI-assisted work. We have reviewed the AI-generated codebase along with testing. We have checked LLM-generated ideas for ForRAIL through experiments. We have reviewed the relevant literature manually. We have reviewed all of the AI-polished paper content. We checked the correctness of the reformatted datasets. Applying MLLM to generate the rationale for our examples is part of our method, so we do not consider it as generating synthetic data sets. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

ForRAIL requires no training, so reproducing our results amounts to re-running a fixed pipeline over public data. The method is fully specified in Section 3; the two retrieval scores are given as pseudocode in Algorithms 1 and 2. The system instruction and the prompt used to generate rationales and Whole Object Labels during example pool construction, are provided in Appendix C and the source code in supplementary material. Dataset sources, the construction of the six TGIF subsets and the 250-image mixed evaluation set, and the protocol for the non-empty subset comparison are described in Appendix E and I. Because rationale generation and MLLM inference are non-deterministic, we will release the exact dataset and example pool used in all experiments after the paper is published, including the generated rationales and labels, so that retrieval results can be reproduced without regenerating them. For datasets subject to copyright restrictions, we will release only the list of samples used in out experiments, rather than redistributing the data. For the two proprietary backbones, we report the model identifiers, reasoning efforts, and query dates in **Implementation details**; since hosted models may change over time, Gemma4 31B is provided as an open-weight reference configuration.

REFERENCES

Anthropic. Introducing claude opus 5. https://www.anthropic.com/news/claude-opus-5, July 2026a. Accessed: 2026-09-26.

Anthropic. Claude Opus 5.5. https://www.anthropic.com/claude-opus-5-5, September 2026b.

Cheng Chen, Yunpeng Zhai, Yifan Zhao, Jinyang Gao, Bolin Ding, and Jia Li. Provoking multimodal few-shot lvlm via exploration-exploitation in-context learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3826–3835, 2025a.

Rui Chen, Bin Liu, Changtao Miao, Xinghao Wang, Yi Li, Tao Gong, Qi Chu, and Nenghai Yu. Training-free in-context forensic chain for image manipulation detection and localization. *arXiv preprint arXiv:2510.10111*, 2025b.

Hai Ci, Ziheng Peng, Pei Yang, Yingxin Xuan, and Mike Zheng Shou. Diffseg30k: A multi-turn diffusion editing benchmark for localized aigc detection. *arXiv preprint arXiv:2511.19111*, 2025.

Chengbo Dong, Xinru Chen, Ruohan Hu, Juan Cao, and Xirong Li. Mvss-net: Multi-view multi-scale supervised networks for image manipulation detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3539–3553, 2022.

Jessica Fridrich and Jan Kodovsky. Rich models for steganalysis of digital images. *IEEE Transactions on information Forensics and Security*, 7(3):868–882, 2012.

Fabrizio Guillaro, Davide Cozzolino, Avneesh Sud, Nicholas Dufour, and Luisa Verdoliva. Trufor: Leveraging all-round clues for trustworthy image forgery detection and localization. In *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 20606–20615. IEEE, 2023.

Jing Hao, Zhixin Zhang, Shicai Yang, Di Xie, and Shiliang Pu. Transforensics: Image forgery localization with dense self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15055–15064, 2021.

Yiran He, Yun Cao, Bowen Yang, and Zeyu Zhang. Can gpt tell us why these images are synthesized? empowering multimodal large language models for forensics. In *Proceedings of the 2025 ACM Workshop on Information Hiding and Multimedia Security*, pp. 24–34, 2025.

Xuefeng Hu, Zhihan Zhang, Zhenye Jiang, Syomantak Chaudhuri, Zhenheng Yang, and Ram Nevatia. Span: Spatial pyramid attention network for image manipulation localization. In *European Conference on Computer Vision*, pp. 312–328. Springer, 2020.

Zhenglin Huang, Jinwei Hu, Xiangtai Li, Yiwei He, Xingyu Zhao, Bei Peng, Baoyuan Wu, Xiaowei Huang, and Guangliang Cheng. Sida: Social media image deepfake detection, localization and explanation with large multimodal model. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28831–28841. IEEE, 2025.

Myung-Joon Kwon, In-Jae Yu, Seung-Hun Nam, and Heung-Kyu Lee. Cat-net: Compression artifact tracing network for detection and localization of image splicing. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 375–384, 2021.

Myung-Joon Kwon, Wonjun Lee, Seung-Hun Nam, Minji Son, and Changick Kim. Safire: Segment any forged image region. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 4437–4445, 2025.

Gert Lek, Nicolas van Schaik, Chaoyi Zhu, Pin-Yu Chen, Robert Birke, and Lydia Y. Chen. Detective SAM: Adaptive AI-image forgery localization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GKJHPHNFIx.

Shuaibo Li, Wei Ma, Jianwei Guo, Shibiao Xu, Benchong Li, and Xiaopeng Zhang. Unionformer: Unified-learning transformer with multi-view representation for image manipulation detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12523–12533, 2024a.

Yuxi Li, Fuyuan Cheng, Wangbo Yu, Guangshuo Wang, Guibo Luo, and Yuesheng Zhu. Adaifl: Adaptive image forgery localization via a dynamic and importance-aware transformer network. In *European conference on computer vision*, pp. 477–493. Springer, 2024b.

Hui Liu, Wenya Wang, Hao Sun, Chris Xing Tian, Chenqi Kong, Xin Dong, and Haoliang Li. Unraveling the mechanics of learning-based demonstration selection for in-context learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2623–2641. Association for Computational Linguistics, 2025a. doi: 10.18653/v1/2025.acl-long.132.

Xiaohong Liu, Yaojie Liu, Jun Chen, and Xiaoming Liu. Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(11):7505–7517, 2022.

Yaqi Liu, Shuhuan Chen, Haichao Shi, Xiao-Yu Zhang, Song Xiao, and Qiang Cai. Mun: image forgery localization based on m^3 encoder and un decoder. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 5685–5693, 2025b.

Zhou Liu, Tonghua Su, Hongshi Zhang, Fuxiang Yang, Donglin Di, Yang Song, and Lei Fan. Foca: Frequency-oriented cross-domain forgery detection, localization and explanation via multi-modal large language model. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 12332–12336. IEEE, 2026.

Hannes Mareen, Dimitrios Karageorgiou, Glenn Van Wallendael, Peter Lambert, and Symeon Papadopoulos. Tgif: Text-guided inpainting forgery dataset. In *2024 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–6. IEEE, 2024.

Hannes Mareen, Dimitrios Karageorgiou, Paschalis Giakoumoglou, Peter Lambert, Symeon Papadopoulos, and Glenn Van Wallendael. Tgif2: extended text-guided inpainting forgery dataset and benchmark. *Journal on Information Security*, 2026(1):13, 2026.

Ju-Hyeon Nam, Dong-Hyun Moon, and Sang-Chul Lee. M2sformer: Multi-spectral and multi-scale attention with edge-aware difficulty guidance for image forgery localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15927–15938, 2025.

OpenAI. GPT-5.6: Frontier intelligence that scales with your ambition. `https://openai.com/index/gpt-5-6/`, 2026.

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

Keqin Peng, Liang Ding, Yancheng Yuan, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. Revisiting demonstration selection strategies in in-context learning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9090–9101. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.492.

Chenfan Qu, Yiwu Zhong, Fengjun Guo, and Lianwen Jin. Omni-iml: Towards unified interpretable image manipulation localization. In *The Fourteenth International Conference on Learning Representations*, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, volume 2025, pp. 28085–28128, 2025.

Ziqi Sheng, Wei Lu, Xiangyang Luo, Jiantao Zhou, and Xiaochun Cao. Sumi-ifl: An information-theoretic framework for image forgery localization with sufficiency and minimality constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 720–728, 2025.

Ziqi Sheng, Junyan Wu, Wei Lu, and Jiantao Zhou. Weakly-supervised image forgery localization via vision-language collaborative reasoning framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 988–996, 2026. doi: 10.1609/aaai.v40i2.37068.

Lei Su, Xiaochen Ma, Xuekang Zhu, Chaoqun Niu, Zeyu Lei, and Ji-Zhe Zhou. Can we get rid of handcrafted feature extractors? sparsevit: Nonsemantics-centered, parameter-efficient image manipulation localization through spare-coding transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7024–7032, 2025. doi: 10.1609/aaai.v39i7.32754.

Zhihao Sun, Haoran Jiang, Haoran Chen, Yixin Cao, Xipeng Qiu, Zuxuan Wu, and Yu-Gang Jiang. Forgerysleuth: Empowering multimodal large language models for image manipulation detection. *Advances in Neural Information Processing Systems*, 38:153136–153175, 2026.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Junke Wang, Zuxuan Wu, Jingjing Chen, Xintong Han, Abhinav Shrivastava, Ser-Nam Lim, and Yu-Gang Jiang. Objectformer for image manipulation detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2364–2373, 2022.

Xubin Wang, Jianfei Wu, Yichen Yuan, Deyu Cai, Mingzhe Li, and Weijia Jia. Demonstration selection for in-context learning via reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 64937–64954. PMLR, 2025.

Zihua Wang, Jiarui Wang, Haiyang Xu, Ming Yan, Fei Huang, Xu Yang, Xiu-Shen Wei, Siya Mi, and Yu Zhang. Efficient and effective in-context demonstration selection with coreset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 10458–10466, 2026.

Yue Wu, Wael AbdAlmageed, and Premkumar Natarajan. Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9543–9552, 2019. doi: 10.1109/CVPR.2019.00977.

Zhipei Xu, Xuanyu Zhang, Runyi Li, Zecheng Tang, Qing Huang, and Jian Zhang. Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. In *International Conference on Learning Representations*, volume 2025, pp. 31186–31216, 2025.

Zeqin Yu, Jiangqun Ni, Yuzhen Lin, Haoyi Deng, and Bin Li. Diffforensics: Leveraging diffusion prior to image forgery detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12765–12774, 2024.

Fanrui Zhang, Jiawei Liu, Jiaying Zhu, Esther Sun, Dong Li, Qiang Zhang, and Zheng-Jun Zha. Forgerygpt: A multimodal llm for interpretable image forgery detection and localization. *arXiv preprint arXiv:2410.10238*, 2024.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. In *Advances in Neural Information Processing Systems*, 2023a.

Keyang Zhang, Chenqi Kong, Hui Liu, Bo Ding, Xinghao Jiang, and Haoliang Li. Propose and rectify: A forensics-driven mllm framework for image manipulation localization. *IEEE Transactions on Information Forensics and Security*, 21:3396–3409, 2026. doi: 10.1109/TIFS.2026.3677584.

Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan Zhang, Wenjie Li, and Min Zhang. Bridging modalities: Improving universal multimodal retrieval by multimodal large language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9274–9285. IEEE, 2025a.

Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. What makes good examples for visual in-context learning? *Advances in Neural Information Processing Systems*, 36:17773–17794, 2023b.

Zhenfei Zhang, Ming-Ching Chang, and Xin Li. Training-free image manipulation localization using diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 10376–10384, 2025b.

Haozhe Zhao, Zefan Cai, Shuzheng Si, Xiaojian Ma, Kaikai An, Liang Chen, Zixuan Liu, Sheng Wang, Wenjuan Han, and Baobao Chang. Mmicl: Empowering vision-language model with multi-modal in-context learning. In *International Conference on Learning Representations*, 2024a.

Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. Ultraedit: Instruction-based fine-grained image editing at scale. *Advances in Neural Information Processing Systems*, 37:3058–3093, 2024b.

Jizhe Zhou, Xiaochen Ma, Xia Du, Ahmed Y. Alhammadi, and Wentao Feng. Pre-training-free image manipulation localization through non-mutually exclusive contrastive learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22346–22356, 2023.

Yucheng Zhou, Xiang Li, Qianning Wang, and Jianbing Shen. Visual in-context learning for large vision-language models. In *Findings of the association for computational linguistics: ACL 2024*, pp. 15890–15902, 2024.

Xuekang Zhu, Xiaochen Ma, Lei Su, Zhuohang Jiang, Bo Du, Xiwen Wang, Zeyu Lei, Wentao Feng, Chi-Man Pun, and Ji-Zhe Zhou. Mesoscopic insights: orchestrating multi-scale & hybrid architecture for image manipulation localization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pp. 11022–11030, 2025.

Rui Zuo, Qinyue Tong, Zhe-Ming Lu, and Ziqian Lu. Unlocking the forgery detection potential of vanilla mllms: A novel training-free pipeline. *arXiv preprint arXiv:2511.13442*, 2025.

A NOTATION

Table 5: Notation used in ForRAIL and CoSFoR.

Symbol	Meaning	Type or shape
Task definition		
$\mathcal{I}_q$	Query image	$\mathbb{R}^{3\times H\times W}$
H, W	Image height and width	$\mathbb{N}$
$\mathcal{M}$	Predicted binary forgery mask	$\{0,1\}^{H\times W}$
Offline example pool construction		
$\mathcal{I}_s$	Source tampered image	$\mathbb{R}^{3\times H\times W}$
$\mathcal{M}_s$	Ground-truth mask of $\mathcal{I}_s$	$\{0,1\}^{H\times W}$
$P=\{P_p\}_{p=1}^{N_P}$	Set of polygons converted from a mask	Set
N_P	Number of polygons	$\mathbb{N}$
$P_p=\{P_{p,i}\}_{i=1}^{N_p}$	The p-th polygon	Set of vertices
N_p	Number of vertices of P_p	$\mathbb{N}$
$P_{p,i}=(x_{p,i}, y_{p,i})$	The i-th vertex of P_p	$\mathbb{N}^2$
$\mathcal{F}_{gen}$	Frozen MLLM for rationale generation	Function
R	Rationale explaining why regions are tampered	Text
WOL	Whole Object Label per polygon	$\{\text{yes}, \text{no}\}^{N_P}$
E_k	The k-th example $(\mathcal{I}_s, (H, W), P, WOL, R)$	Tuple
$\mathcal{E}$	Example pool	Set of E_k
Online MLLM localization		
K	Number of retrieved examples	$\mathbb{N}$
$\mathcal{E}_K$	Retrieved example set	Subset of $\mathcal{E}$, $\lvert\mathcal{E}_K\rvert = K$
Sys	System instruction	Text
$\mathcal{F}_{mllm}$	MLLM used for localization	Function
(w, h)	Coordinate system of the predicted polygons	$\mathbb{N}^2$
B_p	Bounding box of polygon P_p	$\mathbb{N}^4$
CoSFoR		
$\mathcal{I}_k$	Source tampered image of example E_k	$\mathbb{R}^{3\times H\times W}$
$\mathcal{M}_k$	Ground-truth mask of example E_k	$\{0,1\}^{H\times W}$
N	Shortlist size of Stage 1	$\mathbb{N}$
$\mathcal{E}_N$	Semantic shortlist from Stage 1	Subset of $\mathcal{E}$, $\lvert\mathcal{E}_N\rvert = N$
S_{gme}	Semantic similarity from GME	$[-1, 1]$
TCS	Tampered-Contrast Similarity	$[-1, 1]$
$DinoS$	Region-DINOv2 Score	$\mathbb{R}$
S	Semantic-forgery hybrid similarity	$\mathbb{R}$
$w_{gme}, w_{tcs}, w_{dino}$	Weights of the hybrid similarity	$\mathbb{R}_{\geq 0}$
Tampered-Contrast Similarity (Algorithm 1)		
G	Block grid size (G^2 blocks per image)	$\mathbb{N}$ (G=8)
N_s	Number of residual maps	$\mathbb{N}$ (N_s=6)
τ	Mask coverage threshold for block selection	$[0, 1]$ ($\tau=\frac{1}{2}$)
ϵ	Numerical stabilizer	$\mathbb{R}_{>0}$ ($\epsilon=10^{-8}$)
r_s	The s-th residual map	$\mathbb{R}^{H\times W}$
$r_s\lvert_b$	Restriction of r_s to block b	Sub-array
$\text{cov}_b(\cdot)$	Mask coverage ratio of block b	$[0, 1]$
$e_{b,s}$	Block-level residual energy	$\mathbb{R}$
$d_{b,s}$	Deviation of $e_{b,s}$ from the image's median block	$\mathbb{R}$
σ_s	Dispersion of $d_{b,s}$ over blocks	$\mathbb{R}_{\geq 0}$
$\mathcal{B}_k^*$	Tampered block set of example E_k	Set of block indices
$N_k = \lvert\mathcal{B}_k^*\rvert$	Block budget borrowed by the query	$\mathbb{N}$

Symbol	Meaning	Type or shape
a_b	Self-anomaly score of block b (query side)	$\mathbb{R}_{\geq 0}$
$\mathcal{B}^*$	Selected block set (either side)	Set of block indices
$m_s^{\mathrm{sgn}},\ m_s^{\mathrm{abs}},\ v_s$	Signed mean, absolute mean, and dispersion of the s-th map	$\mathbb{R}$
$\phi(\cdot\,;\cdot)$	Tampered-contrast descriptor	$\mathbb{R}^{3N_s}$
Ω	Second argument of ϕ: block set (example side) or block budget (query side)	Set or $\mathbb{N}$
Region-DINOv2 Score (Algorithm 2)		
$\mathcal{T}(\cdot)$	DINOv2 patch-token extractor	Image $\mapsto \mathbb{R}^{L^2\times D}$
L	Token grid size (L^2 patch tokens)	$\mathbb{N}$ (L=37)
Z_q, Z_k	ℓ_2-normalized patch tokens of $\mathcal{I}_q$ and $\mathcal{I}_k$	$\mathbb{R}^{L^2\times D}$
cov_c	Mask coverage of token c	$[0,1]$
$\mathcal{F}, \mathcal{G}$	Foreground / background token sets of the example	Set of token indices
$\mathcal{C}_r$	The r-th tampered region (connected component of $\mathcal{F}$)	Set of token indices
N_R	Number of tampered regions used	$\mathbb{N}$, $N_R \leq N_{\max}$
$N_{\max}$	Region cap	$\mathbb{N}$ ($N_{\max}$=4)
u_r	Normalized mean token of region $\mathcal{C}_r$	$\mathbb{R}^D$
u_{bg}	Normalized mean token of the background	$\mathbb{R}^D$
κ	Match size (top-κ)	$\mathbb{N}$ (κ=10)
$\mu(\cdot)$	Mean of the top-κ cosine similarities to query patches	$[-1,1]$
β	Background penalty weight	$\mathbb{R}_{\geq 0}$ (β=0.5)

B Algorithm for Tampered-Contrast Similarity and Region-DINOv2 Score

Algorithm 1 Tampered-Contrast Similarity $TCS(\mathcal{I}_q, E_k)$

Require: query image $\mathcal{I}_q$; pool example E_k with source tampered image $\mathcal{I}_k$ and ground-truth tamper mask $\mathcal{M}_k$; block grid G=8 (i.e. G^2 non-overlapping blocks); number of residual maps N_s=6; coverage threshold $\tau=\frac{1}{2}$; stabilizer $\epsilon=10^{-8}$
Ensure: similarity $TCS(\mathcal{I}_q, E_k) \in [-1,1]$

$\mathcal{B}_k^* \leftarrow \{\, b : \mathrm{cov}_b(M_k) \geq \tau \,\}$; $N_k \leftarrow |\mathcal{B}_k^*|$ ▷ cov_b: mask coverage of block b; if empty, take the max-coverage block(s)
return $\dfrac{\langle\, \phi(\mathcal{I}_q;\, N_k),\ \phi(\mathcal{I}_k;\, \mathcal{B}_k^*)\,\rangle}{\|\phi(\mathcal{I}_q;\, N_k)\|_2\ \|\phi(\mathcal{I}_k;\, \mathcal{B}_k^*)\|_2}$ ▷ the query borrows the candidate's budget N_k
procedure $\phi(\mathcal{I}_q'; \Omega)$ ▷ $\phi \in \mathbb{R}^{3N_s}$; S: block set (pool) or budget (query)
  $g \leftarrow \mathrm{gray}(\mathcal{I}_q')$
  $r_1 \leftarrow g - \mathrm{med}_{3\times 3}(g)$; $r_2, r_3, r_4 \leftarrow k_{\mathrm{SRM}}^{(1)} * g,\ k_{\mathrm{SRM}}^{(2)} * g,\ k_{\mathrm{SRM}}^{(3)} * g$; $r_5 \leftarrow \mathrm{NP{+}{+}}(\mathcal{I}_q')$; $r_6 \leftarrow \nabla^2 g$
  for $b = 1, \ldots, G^2$ **and** $s = 1, \ldots, N_s$ **do**
    $e_{b,s} \leftarrow \log\bigl(1 + \mathrm{std}(r_s|_b)\bigr)$ ▷ $r_s|_b$: restriction of r_s to block b
  end for
  for $b = 1, \ldots, G^2$ **and** $s = 1, \ldots, N_s$ **do**
    $d_{b,s} \leftarrow e_{b,s} - \mathrm{med}_{b'}\, e_{b',s}$ ▷ cancels the shared camera/compression baseline
  end for
  $\sigma_s \leftarrow \mathrm{std}_b\ d_{b,s}, \quad s = 1, \ldots, N_s$
  if Ω is a block set **then**
    $\mathcal{B}^* \leftarrow \Omega$ ▷ pool side
  else
    $a_b \leftarrow \bigl\|\bigl(d_{b,s}/(\sigma_s + \epsilon)\bigr)_{s=1}^{N_s}\bigr\|_2, \quad b = 1, \ldots, G^2$ ▷ self-anomaly
    $\mathcal{B}^* \leftarrow$ the Ω blocks with largest a_b ▷ query side
  end if
  $m_s^{\mathrm{sgn}} \leftarrow \dfrac{1}{|\mathcal{B}^*|}\sum_{b\in\mathcal{B}^*} d_{b,s}$; $\quad m_s^{\mathrm{abs}} \leftarrow \dfrac{1}{|\mathcal{B}^*|}\sum_{b\in\mathcal{B}^*} |d_{b,s}|$; $\quad v_s \leftarrow \sigma_s$ $\quad (s = 1, \ldots, N_s)$
  return $\bigl[\, m_{1:N_s}^{\mathrm{sgn}}\ ;\ m_{1:N_s}^{\mathrm{abs}}\ ;\ v_{1:N_s} \,\bigr]$
end procedure

Algorithm 2 Region-DINOv2 example score $DinoS(\mathcal{I}_q, E_k)$

Require: query image $\mathcal{I}_q$; pool example E_k with source tampered image $\mathcal{I}_k$ and ground-truth mask $\mathcal{M}_k$; token extractor $\mathcal{T}(\cdot)$: DINOv2 ViT-B/14 at 518×518 px $\to L^2$ patch tokens on an $L\times L$ grid (L=37, L^2=1369), each ℓ_2-normalized; match size κ=10; background weight β=0.5; region cap $N_{\max}$=4

Ensure: score $DinoS(\mathcal{I}_q, E_k) \in \mathbb{R}$

Pool side (offline, once per example):

$Z_k \leftarrow \mathcal{T}(\mathcal{I}_k)$

$\mathrm{cov}_c \leftarrow$ area-downsampled coverage of $\mathcal{M}_k$ on the $L\times L$ token grid, $c = 1, \ldots, L^2$

$\mathcal{F} \leftarrow \{\, c : \mathrm{cov}_c \geq \frac{1}{2} \,\}$; $\mathcal{G} \leftarrow \{\, c : \mathrm{cov}_c < \frac{1}{2} \,\}$ ▷ foreground and background token sets

$\mathcal{C}_1, \ldots, \mathcal{C}_{N_0} \leftarrow$ connected components of $\mathcal{F}$, sorted by decreasing size; $N_R \leftarrow \min(N_0,\, N_{\max})$

if $N_R = 0$ **then** ▷ tamper thinner than one grid cell

  $\mathcal{C}_1 \leftarrow \{\arg\max_c \mathrm{cov}_c\}$; $N_R \leftarrow 1$

end if

$u_r \leftarrow \mathrm{norm}\big(\frac{1}{|\mathcal{C}_r|}\sum_{c\in\mathcal{C}_r} Z_k[c]\big)$, $r = 1, \ldots, N_R$ ▷ $\mathrm{norm}(x) = x/\|x\|_2$

if $\mathcal{G} \neq \emptyset$ **then**

  $u_{\mathrm{bg}} \leftarrow \mathrm{norm}\big(\mathrm{mean}\{\, Z_k[c] : c \in \mathcal{G} \,\}\big)$

else

  $u_{\mathrm{bg}} \leftarrow \mathrm{norm}\big(\mathrm{mean}\{\, Z_k[c] : c = 1, \ldots, L^2 \,\}\big)$ ▷ fallback when the mask covers every token

end if

Query side:

$Z_q \leftarrow \mathcal{T}(\mathcal{I}_q)$

Asymmetric mean-of-top-κ matching:

for $v \in \{u_1, \ldots, u_{N_R},\, u_{\mathrm{bg}}\}$ **do**

  $\mu(v) \leftarrow$ mean of the κ largest values of $\{\, \langle v,\, Z_q[c]\rangle : c = 1, \ldots, L^2 \,\}$

end for

return $DinoS(\mathcal{I}_q, E_k) \leftarrow \dfrac{1}{N_R}\sum_{r=1}^{N_R} \mu(u_r) \;-\; \beta\, \mu(u_{\mathrm{bg}})$

C PROMPTS

System Instruction Template

You are the Reasoner for an image-forgery localization pipeline.

You will first receive K examples, then the target sample will be presented after those examples.

The image you will be asked about HAS BEEN TAMPERED — some pixels were manipulated (copy–move, splicing, removal/inpainting, or AI generation) and the exact region is hidden from you. Do NOT default to "authentic" or to an empty answer. Your job is to localize the tampered region(s) as one or more **polygons** in the SOURCE image's pixel coordinate system. The only question is WHERE.

You have NO tool-calls and NO forensic heatmaps. You are given ONLY the SOURCE image. Reason directly from what you can see in it — lighting and shadows, perspective and geometry, texture and noise, edges and halos, colour, and semantic plausibility.

=== Output format ===
When you are asked to PRODUCE THE ANSWER, emit a single JSON object with exactly these keys:

```
{
 "coordinate_space": [width, height],   // REQUIRED, FIRST KEY — the pixel frame your
                                        //  vertices are expressed in (the SOURCE image's
                                        //  true size if you know it)
 "polygons": [[[x,y],[x,y],...], ...],  // 0+ free-shape regions; each polygon is
                                        //  a list of >=3 [x,y] vertices in `coordinate_space`
 "whole_object": ["yes" | "no", ...],   // one per polygon, SAME ORDER — "yes" if the
                                        //  region is one whole discrete object
 "region_evidence": [                   // REQUIRED — one reason string per polygon,
  "<why this region is tampered — in your own words>",  //  in the SAME ORDER as `polygons`
  ...
 ],
 "rationale": "<= 200 words; scene survey + one block per region; see rules below"
}
```

Polygons only. Do NOT emit boxes or points. Trace each tampered region with a polygon of roughly 8–30 vertices that hugs its boundary — vertex placement IS the mask edge, so place them precisely. A single polygon covering more than ~30% of the image is almost always wrong; split large or disjoint regions into separate polygons. There is no cap on the number of polygons, but every one must be defensible.

Coordinate rule. All vertices are in the SOURCE image's pixel coordinate system (origin top-left, x right, y down), and `coordinate_space` states that frame's [width, height]. If for any reason your vertices are expressed in a different frame (e.g. a normalized 1024x1024 grid), `coordinate_space` MUST state THAT frame's [width, height] instead, so the vertices can be mapped back onto the source image. Declare `coordinate_space` BEFORE writing any vertex.

Evidence rule (every polygon must be defensible). Each `region_evidence` entry MUST be a non-empty reason — explain, in your own words, why you believe this region is tampered. Point to whatever is convincing: a visible inconsistency (e.g. "cast shadow points the opposite way from neighbouring objects", "skin texture markedly smoother than the rest of the face", "hard non-scene edge with a thin halo", "garbled text on the sign", "colour/lighting mismatch with the surrounding region"), a comparison with neighbouring regions, or an inference. A polygon you cannot justify is worthless — drop it instead of emitting it.

Whole-object rule. For each polygon, `whole_object` is "yes" when the region is one whole, discrete object (a person, an animal, a vehicle, a plate — something a segmentation model could cut out cleanly along its natural boundary), and "no" when it is an object part, a diffuse area, a texture/background patch, or a compound region spanning several objects.

Rationale rules.
1. Start with a `Scene survey` subsection: list 3–6 candidate regions you CONSIDERED — BOTH nameable objects ("the dog at ~(820,455)") AND non-object patches by position ("upper-left sky, x=0–300 y=0–200"). For each, one sentence: what made you look, and whether you KEEP or DISMISS it (with the reason). Listing only 0–2 candidates means you anchored too quickly.
2. Then one block per KEPT region, each citing its approximate (x,y) location and its reason.

No prose after the JSON.

Figure 4: System instruction template.

Analysis Instruction Template for Example Pool Construction

You are authoring a GOLD-STANDARD worked example that will teach another model to localize image tampering FROM THE SOURCE IMAGE ALONE.
Accuracy and honesty matter more than fluency: this example will be imitated, so any claim you invent will be learned as a bad habit.

You are given ONLY the SOURCE image (Image 1). Reason purely from what is visible — lighting and shadows, perspective and geometry, texture and noise, edges and halos, color, and semantic plausibility.

GROUND TRUTH (given to you — do NOT relocate it):
- The tampered region(s) are exactly these polygons, in SOURCE pixel coords: {GOLD_POLYGONS}
Treat the given polygon as the confirmed tampered region; explain it as such, do not question or relabel it.

Your job: produce the COMPLETE gold answer JSON for this image, in this exact schema (the same one the student model must produce):

```
{
 "coordinate_space": [W, H],   // copy the given coordinate_space VERBATIM — do not edit coordinate_space, the pixel frame the vertices are expressed in.
 "polygons": {GOLD_POLYGONS},          // copy the given polygons VERBATIM — do not edit coordinates
 "whole_object": ["yes" | "no", ...],   // one per polygon, SAME ORDER — "yes" if the region is one whole discrete object
 "region_evidence": {                  // one entry per polygon, SAME ORDER as "polygons"
  "polygons": [
   {"reason": "<why this region is tampered — in your own words>"}
  ]
 },
 "rationale": "<= 200 words; Scene survey first, then one block per region"
}
```

Honesty rules — follow strictly:
1. **polygons**: paste the given coordinates unchanged.
2. **reason**: explain, in your own words, why you believe this region is tampered. You are free to point to whatever is convincing — a visible cue (shadow direction off, texture too smooth, hard non-scene edge with a halo, garbled text, lighting/color mismatch, duplicated/cloned patch, broken perspective), a comparison with neighboring regions, or an inference. Be honest and specific (say where and what); don't invent a cue just to fill the field. Every polygon MUST have a non-empty **reason**.
3. **rationale** → Scene survey: list the true region (KEEP) plus 2–4 plausible decoys you DISMISS with a one-line reason each — this teaches the student to survey before committing. Then one block per kept region with its (x,y) and its reason. Do NOT mention heatmaps or forensic tools (there are none).

Output the JSON only — nothing before or after it.

Figure 5: Analysis instruction template for example pool construction

D MLLM BACKBONE DETAILS

Table 6: MLLM backbones and the rationale generator used in all experiments. Dashes denote information not disclosed by the provider.

	Gemma4 31B	**GPT-5.6 Luna**	**Opus 5.5**	**Opus 5**
Role	Localization	Localization	Localization	Pool construction
Provider	Google	OpenAI	Anthropic	Anthropic
Model identifier	`google/gemma-4-31b-it`	`gpt-5.6-luna`	`claude-opus-5.5`	`claude-opus-5`
Weights	Open	Proprietary	Proprietary	Proprietary
Parameters	31B	–	–	–
Access	Local inference	API	API	Claude Code CLI
Inference framework	`Ollama v0.34.0`	–	–	–
Precision	`Q4_K_M`	–	–	–
Image input	`original resolution`	`original resolution`	`original resolution`	`original resolution`
Decoding	`temperature=1`	`effort=low`	`effort=low`	`effort=high`
Retries on parse failure	`10`	`10`	`10`	`10`
Query dates	`local, n/a`	21–25 Sep 2026	21–25 Sep 2026	14–31 Aug 2026 (3 batches)

E DETAILS OF DATASETS FOR EVALUATION

SID-Set. SID-Set contains real, fully synthetic, and locally tampered images, with the tampered images generated using a Latent Diffusion Model (LDM). Since our task focuses on image forgery localization, we randomly sample 1,000 images from the tampered test split for evaluation.

MagicBrush. MagicBrush is a manually annotated instruction-guided image editing dataset, where target images are generated using DALL-E 2 according to natural-language editing instructions and free-form masks. We evaluate on the complete test split.

UltraEdit. UltraEdit is a large-scale automatically constructed instruction-based image editing dataset. We use its Region-Based 100K subset, which contains localized edits generated using SDXL-Turbo, and randomly sample 3,000 examples for evaluation.

COCOGlide. COCOGlide is constructed from MS-COCO images using GLIDE-based text-guided inpainting to regenerate local image regions. We evaluate on the complete dataset.

TGIF. TGIF (Mareen et al., 2024) and TGIF2 (Mareen et al., 2026) contain text-guided image manipulations generated by multiple diffusion-based editors. We evaluate the spliced (SP) and fully regenerated (FR) subsets generated using Stable Diffusion 2 (SD2), FLUX.1 Dev, and FLUX.1 Fill Dev. For each subset–editor combination, we use 343 samples. The only exception is the

FLUX.1 Fill Dev FR subset, for which 327 valid samples are retained after excluding cases whose corresponding edited images are blank.

DiffSeg30k. DiffSeg30k contains locally manipulated images generated using eight diffusion-based editing models: Stable Diffusion 2 Inpainting, Kolors, Stable Diffusion 3.5 Medium, FLUX, Stable Diffusion XL Inpainting, GLIDE, HunyuanDiT, and Kandinsky 2.2 Inpainting. From its test split, we select 2,817 samples whose ground-truth masks contain both manipulated and unmanipulated pixels.

Mixed Dataset. We construct a mixed dataset of 250 samples from UltraEdit, COCOGlide, TGIF-FLUX-SP, TGIF-FLUX-FR, and Diffseg30k, with 50 images sampled from each dataset.

F OOD AVERAGE V.S. WORST-CASE

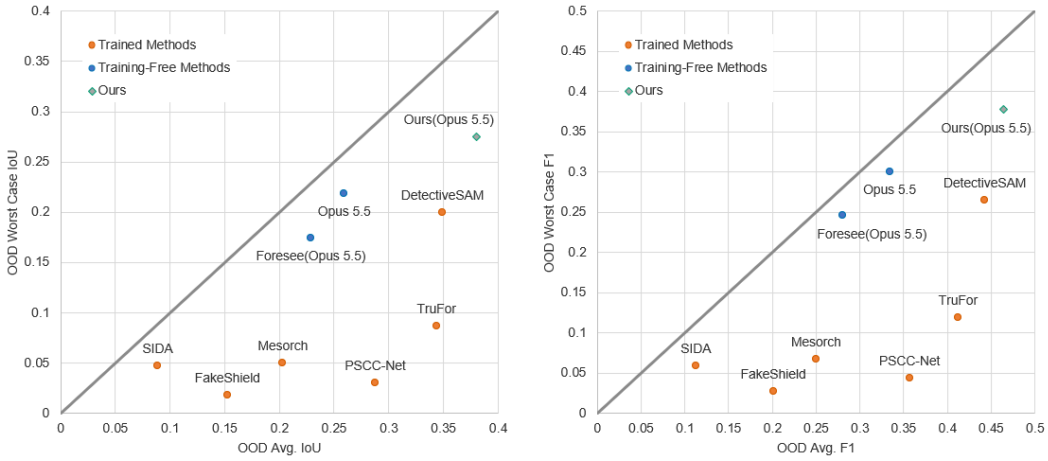


Figure 6: IoU and F1 of OOD average v.s. worst-case. Showing the robustness of ForRAIL.

G PERFORMANCE ACROSS DIFFERENT GROUND-TRUTH SIZE OF TAMPERED REGIONS (F1)

Table 7: F1 grouped by the relative size of the ground-truth tampered region, on the same pooled OOD set as Table 3. Results are computed on the pooled OOD set, i.e., the union of UltraEdit, CocoGlide, TGIF, and DiffSeg30k from Table 1 (8,368 images); the "All" row is therefore a sample-weighted average over the pooled set and differs from the per-dataset OOD average in Table 1. n denotes the number of samples in each group. DSAM denotes DetectiveSAM$_{\text{sota}}$. The best scores are **bold**.

GT size	n	**DSAM**	**Vanilla**			**Ours**		
			Gemma4 31B	GPT-5.6 Luna	Opus 5.5	Gemma4 31B	GPT-5.6 Luna	Opus 5.5
<1%	664	0.0680	0.0991	0.2005	0.2430	0.2148	0.2942	**0.3990**
1–5%	1287	0.2288	0.2435	0.3607	0.4304	0.4245	0.4438	**0.5400**
5–20%	2203	0.4295	0.2999	0.3438	0.3951	0.4395	0.4525	**0.4952**
20–50%	2453	**0.5089**	0.2471	0.2357	0.2994	0.3971	0.3824	0.4187
>50%	1761	**0.5542**	0.2059	0.1559	0.2325	0.3782	0.3114	0.3632
All	8368	0.4195	0.2400	0.2638	0.3262	0.3940	0.3884	**0.4442**

H Comparison of the performance on fully regenerated samples and spliced samples across TGIF subsets

TGIF provides each forgery in two forms: a spliced (SP) version, in which the edited region is pasted back into the original image, and a fully regenerated (FR) version, in which the entire image is re-generated by the generator and no splicing boundary remains. Table 2 in the main text reports FR results; Table 8 reports SP results.

Comparing the two tables exposes what each family of methods relies on. TruFor is the strongest method on spliced samples (0.6753, 0.6719, and 0.7117 IoU on TGIF-FLUX, TGIF-FLUX-Fill, and TGIF-SD2), yet its IoU falls by 0.47–0.55 on the FR counterparts, where the low-level inconsistencies along the splicing boundary no longer exist. The remaining trained methods follow the same pattern at lower absolute levels. ForRAIL, in contrast, is nearly unaffected by the presence or absence of a splicing boundary (0.4679 vs. 0.4701 IoU on TGIF-FLUX; 0.4333 vs. 0.4133 on TGIF-FLUX-Fill with Opus 5.5 backbone), and on FR samples it is the best method overall on every subset, exceeding the best trained method by 0.20–0.28 IoU. Vanilla MLLMs and Foresee are likewise insensitive to the splicing boundary, indicating that this stability is a general property of MLLM-based localization; ForRAIL retains it while adding a substantial margin over both.

Note that the FR samples of TGIF-SD2 released by Mareen et al. (2024) are cropped from the source image, whereas the SP samples are not, so the TGIF-SD2 columns of the two tables are not directly comparable, and their difference should not be read as an FR-versus-SP effect. We nevertheless retain TGIF-SD2 because it contributes a third generator and thereby strengthens the cross-generator evaluation.

Table 8: Comparison of the performance on spliced samples across TGIF subsets. The best scores are **bold**; trained and training-free methods are annotated separately.

Methods	TGIF-FLUX		TGIF-FLUX-Fill		TGIF-SD2	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Trained Methods						
DetectiveSAM$_{\text{sota}}$	0.2021	0.2659	0.1311	0.1841	0.2084	0.2803
TruFor	**0.6753**	**0.7671**	**0.6719**	**0.7660**	**0.7117**	**0.8120**
PSCC-Net	0.0101	0.0149	0.0452	0.0624	0.0911	0.1212
Mesorch	0.1551	0.1905	0.1230	0.1514	0.1317	0.1750
SIDA	0.0386	0.0458	0.0327	0.0405	0.0493	0.0636
FakeShield	0.0094	0.0136	0.0079	0.0121	0.0081	0.0136
Training-Free Methods and Vanilla MLLMs						
Gemma4 31B	0.1529	0.2056	0.1474	0.2023	0.1426	0.2009
Foresee(Gemma4 31B)	0.2820	0.3196	0.2714	0.3146	0.2378	0.2903
Ours(Gemma4 31B)	0.3831	0.4471	0.3474	0.4164	0.3353	0.4115
GPT-5.6 Luna	0.2557	0.3081	0.2727	0.3329	0.2643	0.3284
Foresee(GPT-5.6 Luna)	0.1931	0.2180	0.1938	0.2242	0.1602	0.1973
Ours(GPT-5.6 Luna)	0.4211	0.4775	0.4008	0.4678	0.3893	0.4632
Opus 5.5	0.2889	0.3360	0.3023	0.3604	0.3267	0.3980
Foresee(Opus 5.5)	0.2146	0.2414	0.1919	0.2226	0.2107	0.2563
Ours(Opus 5.5)	**0.4679**	**0.5282**	**0.4333**	**0.5014**	**0.4625**	**0.5479**

I Comparison between ForRAIL and Foresee without Empty Masks

Even after removing the explicit instruction in Foresee's prompt that permits an empty mask, its pipeline still returns empty predictions for a substantial fraction of images. Each such case scores zero IoU and F1 and therefore depresses its averages in Table 1. To rule out the possibility that our margin merely reflects this failure mode, Tables 9 restricts the comparison to the images on which Foresee produces a non-empty mask, with both methods evaluated on the same retained subset; "Kept / Total" reports the subset size. This filter is deliberately chosen in Foresee's favor, since it discards exactly the cases in which Foresee fails outright.

Under this protocol, ForRAIL still leads on every dataset. On the retained OOD images, the average IoU gap of Opus 5.5 backbone narrows from 0.1515 to 0.1127 (0.4097 vs. 0.2970), indicating that roughly a quarter of the margin in the main table stems from Foresee's empty predictions, while the remainder reflects a genuine localization advantage. Its worst-case IoU of 0.2860 (DiffSeg30k) also remains 0.0820 above Foresee's 0.2040.

Table 9: Comparison of ForRAIL and Foresee on the kept subsets, i.e., images for which Foresee produced a non-empty prediction. Since the kept subset depends on the backbone, Kept / Total is reported per backbone. OOD Avg. and OOD Worst-Case are computed over the four OOD datasets as in Table 1. The best scores within each backbone are **bold**.

Methods	SID-Set		MagicBrush		UltraEdit		COCOGlide		TGIF		DiffSeg30k		OOD Avg.		OOD Worst-Case	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Kept / Total (Gemma4 31B)	917 / 1000		1050 / 1053		1648 / 3000		443 / 512		1905 / 2039		2316 / 2817		-		-	
Foresee(Gemma4 31B)	0.2467	0.2994	0.1440	0.2160	0.3058	0.3623	0.3394	0.4053	0.2960	0.3455	0.1797	0.2550	0.2803	0.3420	0.1797	0.2550
Ours(Gemma4 31B)	**0.3819**	**0.4420**	**0.2339**	**0.3393**	**0.3318**	**0.4061**	**0.3630**	**0.4484**	**0.3642**	**0.4360**	**0.2470**	**0.3436**	**0.3265**	**0.4085**	**0.2470**	**0.3436**
Kept / Total (GPT-5.6 Luna)	916 / 1000		936 / 1053		2635 / 3000		415 / 512		1569 / 2039		2287 / 2817		-		-	
Foresee(GPT-5.6 Luna)	0.2359	0.2839	0.1211	0.1826	0.2489	0.2981	0.3572	0.4271	0.2750	0.3211	0.1922	0.2694	0.2683	0.3289	0.1922	0.2694
Ours(GPT-5.6 Luna)	**0.4056**	**0.4555**	**0.1848**	**0.2711**	**0.2968**	**0.3626**	**0.3675**	**0.4483**	**0.4026**	**0.4701**	**0.2627**	**0.3639**	**0.3324**	**0.4112**	**0.2627**	**0.3626**
Kept / Total (Opus 5.5)	932 / 1000		1053 / 1053		2932 / 3000		281 / 512		1560 / 2039		2407 / 2817		-		-	
Foresee(Opus 5.5)	0.3429	0.4061	0.1807	0.2683	0.3040	0.3617	0.3759	0.4486	0.3043	0.3542	0.2040	0.2877	0.2970	0.3631	0.2040	0.2877
Ours(Opus 5.5)	**0.4888**	**0.5466**	**0.2499**	**0.3555**	**0.3614**	**0.4354**	**0.4900**	**0.5803**	**0.5013**	**0.5790**	**0.2860**	**0.3910**	**0.4097**	**0.4964**	**0.2860**	**0.3910**

J Impact of SAM integration

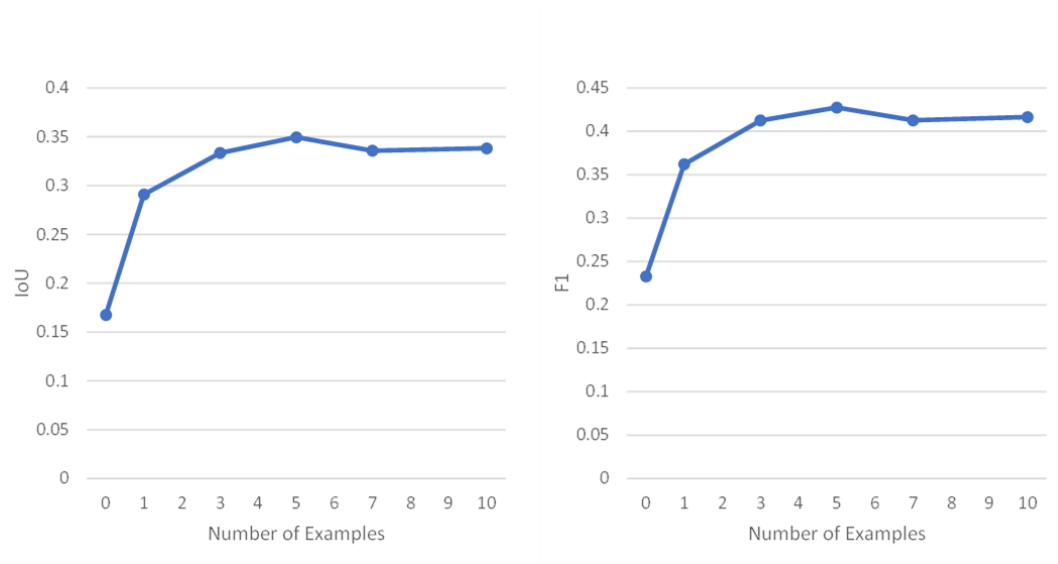
Table 10 compares three settings: no SAM refinement, SAM applied to every polygon, and our WOL-gated refinement, where the MLLM decides per polygon whether the region is a whole object that SAM can segment. It shows that with more powerful MLLM backbones, the gain from SAM refinement decreases. The performance of Opus 5.5 increased only 0.005 IoU on OOD average with SAM refinements, while Gemma4 31B benefits 0.03 IoU from it. We can conclude that powerful MLLMs don't need addition segmentation models in IFL to reach a high performance.

Table 10: Ablation on the use of SAM across MLLM backbones. WOL denotes Whole Object Label from ForRAIL. TGIF reports the average over the six subsets. The best scores within each backbone are **bold**.

Setting	UltraEdit		COCOGlide		TGIF		DiffSeg30k		OOD Avg.	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Gemma4 31B										
w/o SAM	0.3039	0.3939	0.3249	0.4255	0.2741	0.3620	**0.2585**	**0.3617**	0.2904	0.3858
w/o WOL w/ SAM	0.3243	0.3945	0.3437	0.4279	**0.3627**	**0.4343**	0.2403	0.3354	0.3178	0.3980
w/ WOL w/ SAM (Ours)	**0.3277**	**0.4031**	**0.3515**	**0.4394**	0.3581	0.4298	0.2514	0.3498	**0.3222**	**0.4055**
GPT-5.6 Luna										
w/o SAM	0.2733	0.3530	0.3237	0.4177	0.3377	0.4198	**0.2668**	**0.3719**	0.3004	0.3906
w/o WOL w/ SAM	**0.2963**	0.3599	0.3475	0.4251	**0.3991**	**0.4654**	0.2503	0.3473	0.3233	0.3994
w/ WOL w/ SAM (Ours)	0.2955	**0.3620**	**0.3487**	**0.4292**	0.3925	0.4592	0.2571	0.3572	**0.3234**	**0.4019**
Opus 5.5										
w/o SAM	0.3513	0.4315	0.4190	0.5092	0.4482	0.5269	**0.2848**	**0.3890**	0.3758	**0.4642**
w/o WOL w/ SAM	0.3571	0.4277	0.4195	0.5004	**0.4671**	**0.5388**	0.2665	0.3678	0.3775	0.4586
w/ WOL w/ SAM (Ours)	**0.3595**	**0.4338**	**0.4251**	**0.5094**	0.4614	0.5342	0.2752	0.3780	**0.3803**	0.4639

K Ablation study on number of examples provided to MLLM

Figure 7 shows that the performance converges and reach the top when providing 5 examples to Gemma4 31B-IT.



(a) IoU (b) F1

Figure 7: Effect of the number of examples on localization performance with Gemma4 31B backbone.

L External perturbation robustness

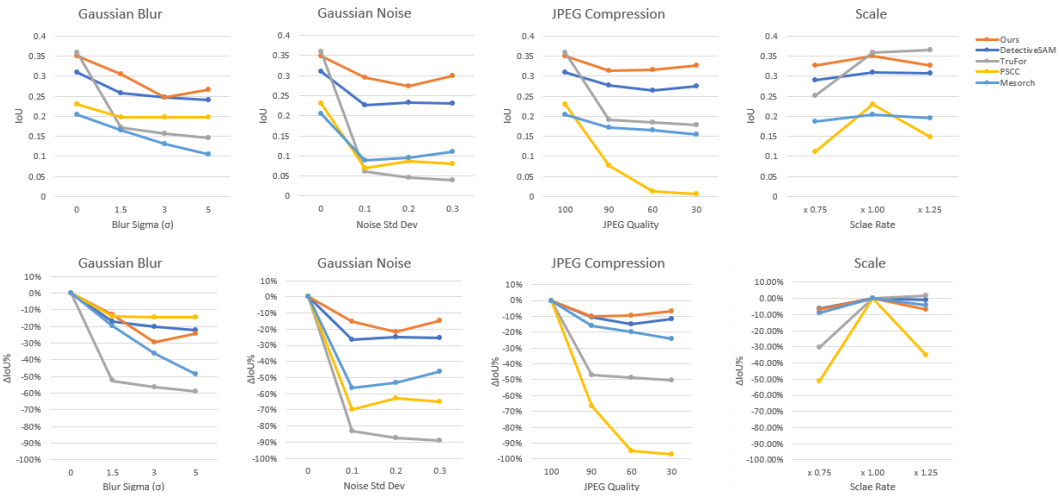


Figure 8: Localization robustness under Gaussian blur, Gaussian noise, JPEG compression, and image scaling. The top row reports absolute IoU, while the bottom row reports the percentage change in IoU relative to the corresponding unperturbed input. ForRAIL uses Gemma4 31B-IT as the MLLM backbone.

M Additional backbone

Table 11 and Table 12 show the result of Qwen3.8 27B as an additional MLLM backbone. Although ForRAIL improves the OOD Avg. of vanilla Qwen 3.8 27B by 0.08 IoU, it is only comparable with Foresee under the protocol of kept subsets.

Table 11: Comparison of image forgery localization performance across different datasets with Qwen3.8 27B as the MLLM backbone. The best scores are **bold**.

Methods	SID-Set		MagicBrush		UltraEdit		COCOGlide		TGIF		DiffSeg30k		OOD Avg.		OOD Worst-Case	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Training-Free Methods and Vanilla MLLMs																
Qwen3.8 27B	0.1984	0.2658	0.1530	0.2315	0.1665	0.2384	0.1743	0.2609	0.1494	0.2045	0.1609	0.2441	0.1628	0.2370	0.1494	0.2045
Foresee(Qwen3.8 27B)	0.2204	0.2665	0.1135	0.1725	0.1993	0.2387	0.2539	0.3044	0.2140	0.2511	0.1355	0.1933	0.2007	0.2469	0.1355	0.1933
Ours(Qwen3.8 27B)	**0.3325**	**0.3852**	**0.1643**	**0.2467**	**0.2552**	**0.3230**	**0.2675**	**0.3413**	**0.2620**	**0.3167**	**0.1988**	**0.2844**	**0.2459**	**0.3164**	**0.1988**	**0.2844**

Table 12: Comparison of ForRAIL and Foresee on the kept subsets with Qwen3.8 27B as the MLLM backbone, i.e., images for which Foresee produced a non-empty prediction. OOD Avg. and OOD Worst-Case are computed over the four OOD datasets as in Table 1. The best scores are **bold**.

Methods	SID-Set		MagicBrush		UltraEdit		COCOGlide		TGIF		DiffSeg30k		OOD Avg.		OOD Worst-Case	
	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑	IoU ↑	F1 ↑
Kept / Total (Qwen3.8 27B)	916 / 1000		937 / 1053		2482 / 3000		362 / 512		1624 / 2039		2223 / 2817		-		-	
Foresee(Qwen3.8 27B)	0.2406	0.2910	0.1275	0.1938	0.2409	0.2886	**0.3590**	**0.4305**	0.2677	0.3139	0.1717	0.2450	**0.2598**	0.3195	0.1717	0.2450
Ours(Qwen3.8 27B)	**0.3451**	**0.3989**	**0.1655**	**0.2490**	**0.2556**	**0.3220**	0.2925	0.3699	**0.2753**	**0.3314**	**0.2010**	**0.2874**	0.2561	**0.3277**	**0.2010**	**0.2874**

N FORRAIL FAILURE MODES

Several low performance samples are shown in Figure 9.

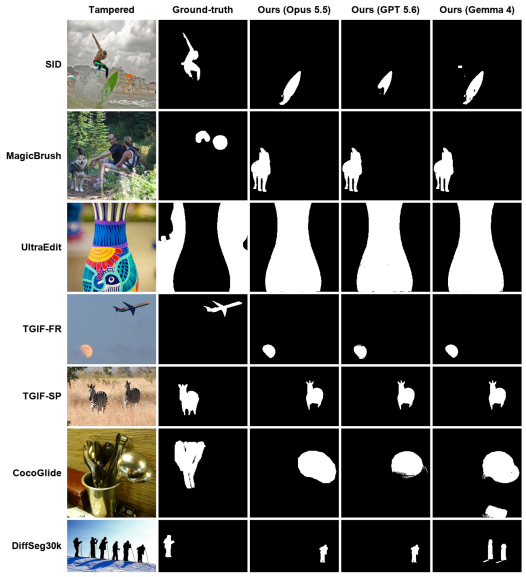


Figure 9: ForRAIL low performance samples.