FragFake: A Dataset for Fine-Grained Detection of Edited Images with Vision Language Models

Zhen Sun[1] **Ziyi Zhang**[1] **Zeren Luo**[1] **Zeyang Sha**[2]
Tianshuo Cong[3] **Zheng Li**[4] **Shiwen Cui**[2] **Weiqiang Wang**[2]
Jiaheng Wei[1] **Xinlei He**[1*] **Qi Li**[3] **Qian Wang**[5]
[1]Hong Kong University of Science and Technology (Guangzhou) [2]Ant Group
[3]Tsinghua University [4]Shandong University [5]Wuhan University

Abstract

Fine-grained edited image detection of localized edits in images is crucial for assessing content authenticity, especially given that modern diffusion models and image editing methods can produce highly realistic manipulations. However, this domain faces three challenges: (1) Binary classifiers yield only a global real-or-fake label without providing localization; (2) Traditional computer vision methods often rely on costly pixel-level annotations; and (3) No large-scale, high-quality dataset exists for modern image-editing detection techniques. To address these gaps, we develop an automated data-generation pipeline to create *FragFake*, the first dedicated benchmark dataset for edited image detection, which includes high-quality images from diverse editing models and a wide variety of edited objects. Based on *FragFake*, we utilize Vision Language Models (VLMs) for the first time in the task of edited image classification and edited region localization. Experimental results show that fine-tuned VLMs achieve higher average Object Precision across all datasets, significantly outperforming pretrained models. We further conduct ablation and transferability analyses to evaluate the detectors across various configurations and editing scenarios. To the best of our knowledge, this work is the first to reformulate localized image edit detection as a vision-language understanding task, establishing a new paradigm for the field. We anticipate that this work will establish a solid foundation to facilitate and inspire subsequent research endeavors in the domain of multimodal content authenticity. The code[1] and dataset[2] are open-source.

1 Introduction

Owing to the swift progress of diffusion models, the images they generate, commonly referred to as AI-Generated Content (AIGC), have become remarkably lifelike [1, 2, 3]. At the same time, image editing techniques have also made significant progress [4, 5, 6]. Unlike traditional approaches that generate fully synthetic images, modern editing techniques enable localized modifications guided by natural language, while preserving the rest of the image [4].

These partially edited versions based on real images often appear highly authentic, as most of their visual content remains unaltered. However, such realism can also post severe safety threats. Such images can be exploited to craft highly convincing fake news or manipulate public opinion [7]. For instance, in July 2024, a genuine Associated Press photograph showed Secret Service agents solemnly protecting Donald Trump following an assassination attempt. A manipulated version of this image circulated on social media, in which the agents were deceptively edited to appear smiling. As a result of this modification, some users misconstrued the event as a "political performance", which in turn fueled widespread misinformation and public confusion [8]. Accordingly, developing the capacity to accurately detect and distinguish partially edited images from genuine ones has become both a pressing and indispensable challenge.

Most traditional image forgery detectors are trained on datasets consisting entirely of either real or fully generated images. As a result, their performance degrades significantly when faced with images that contain only localized edits. For example, with open-source AIGC detector Hive Moderation [9], only 55 out of 100 partially edited images are correctly identified as AI-generated (as described in Appendix B). This limitation stems from the presence of a large amount of authentic content in the image, which easily misleads the classifier into predicting it as real. Moreover, most existing detection methods rely on binary classification strategies that provide only a decision of "real" or "fake" for the entire image. These methods lack the ability to indicate which specific regions have been edited, making them unsuitable for real-world forensic and provenance applications. While some computer vision approaches explore edited region localization, they typically rely on costly pixel-level annotations [10]. Moreover, the datasets used often involve outdated editing models, which fail to reflect the realism of modern generation techniques.

*Corresponding author(xinleihe@hkust-gz.edu.cn).
[1]The code's link: https://github.com/Vincent-HKUSTGZ/FragFake.
[2]The dataset's link: https://huggingface.co/datasets/Vincent-HKUSTGZ/FragFake.

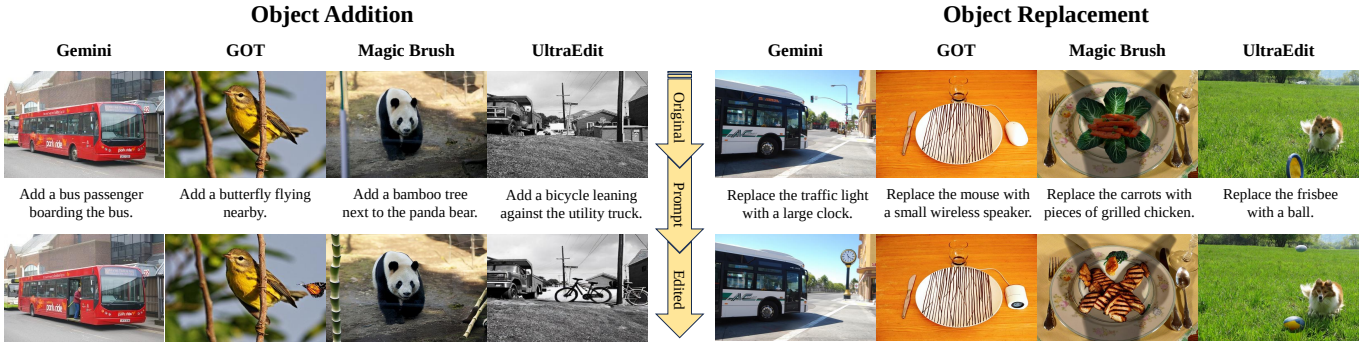


Figure 1: Examples of edited images generated by four different models, showcasing two types of operations: Object Addition and Object Replacement.

1.1 Our Contribution

To address these problems, we propose leveraging Vision Language Models (VLMs) [11] for edited images detection, including both classification and localization. Pretrained on large-scale image–text pairs, VLMs possess strong visual understanding and can be efficiently fine-tuned for diverse downstream tasks.

Since using VLMs for edited image detection is a novel task and no high-quality public dataset currently exists, we construct a dedicated image dataset, referred to as *FragFake*, which consists of edited images generated by several advanced image editing models, including three open-source models (MagicBrush [6], GoT [12], and UltraEdit [12]) and one commercially deployed model (Gemini, hereafter referred to as Gemini-IG [13] for "Image Generation"). The dataset covers two major types of editing operations: object addition and object replacement.

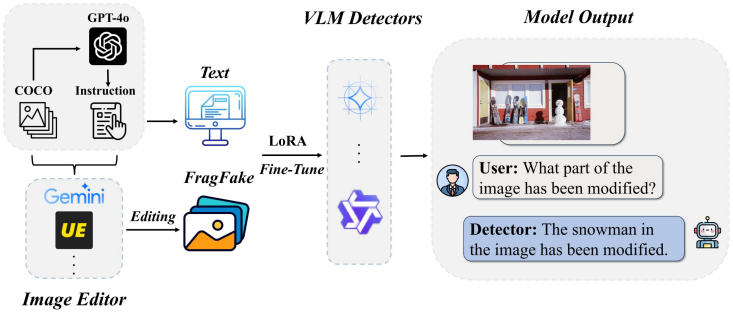
To ensure diversity among source images, we sample 20 images from each of the 80 categories in the COCO dataset [14], yielding 1,600 originals. Since most modern image editing methods support natural language-driven editing, we use GPT-4o [15] to generate a total of 3,200 editing instructions based on these original images. During this process, we observe that many of the target objects specified in the instructions are repeated. We therefore refer to this version as the Easy version. Building upon it, we further refine the dataset by filtering and replacing overlapping target objects to produce the Hard version, which contains 1,930 unique instructions where all target objects are non-redundant. Combined with four editing models, these instructions produce 20,222 edited images. This image generation, together with instruction creation, is fully automated, forming a scalable and extensible pipeline. It is important to note that the current dataset is primarily designed to validate and demonstrate the feasibility of our pipeline; thus, only around 20,000 images are generated. Nevertheless, the pipeline is highly scalable and can be extended to produce hundreds of thousands of samples to support larger-scale research. The resulting edited images and their corresponding instructions are then converted into image–text pairs for training VLMs. We fine-tune four widely used models for this task: Llava-1.5 [16], Qwen2-VL [17], Qwen2.5-VL [18] and Gemma3 [19].

Our evaluation is conducted at two levels: edited image classification performance (using Accuracy and F1-score) and edited region localization performance, using Region Precision (RP) and Object Precision (OP). Among all pretrained VLMs, GPT-4o performs best, with 0.810 Accuracy and 45.0% Object Precision (OP) on the Hard version, and 0.825 Accuracy with 46.0% OP on the Easy version of the Gemini-IG dataset. In contrast, the Qwen2.5-VL (7B) detector, fine-tuned with LoRA, achieves significantly the best performance, with Accuracy scores of 0.990 and 0.965 and OP values of 69.0% and 74.0% on the Hard and Easy versions, respectively. Compared to the initial performance of pretrained models, which achieved only around 5.0% OP, the fine-tuned Qwen2.5-VL shows a substantial improvement. This highlights the effectiveness of our constructed dataset.

In addition, we conduct ablation studies to examine how data balancing strategies, training size, and LoRA rank affect model performance. Results show that larger models perform better with lower LoRA ranks, suggesting fewer trainable parameters suffice for effective adaptation. We also conduct transfer experiments, showing that detectors trained on Gemini-IG and GoT generalize well across domains, while those from MagicBrush and UltraEdit perform worse. Detectors trained on one editing type also generalize well to the other, indicating strong cross-task transferability.

In conclusion, our main contributions are as follows:

- We are the first to propose reframing edited image detection (classification and edited region localization) as a vision-language understanding task to reduce reliance on costly annotations. To support this, we construct *FragFake*, a large-scale dataset of over 20,000 edited image–text pairs generated via a fully automated pipeline with diverse editing models and target objects.
- To address the poor performance of pretrained VLMs, we finetune and benchmark several widely used VLMs (Llava-1.5, Qwen2-VL/2.5-VL, Gemma3), achieving substantial improvements in detection performance that far exceed GPT-4o's performance.
- We conduct ablation and transferability studies: ablation experiments reveal that an optimal LoRA rank, balanced data strategies, and sufficient training scale significantly enhance detection performance; transferability tests demon-



Number of images in the FragFake

Editing Model	Hard			Easy		
	Original	Edited Image		Original	Edited Image	
		OA	OR		OA	OR
Gemini-IG	1600	881	1009	1600	1469	1473
MagicBrush	1600	906	1024	1600	1600	1600
GoT	1600	906	1024	1600	1600	1600
UltraEdit	1600	906	1024	1600	1600	1600

Figure 2: Dataset construction pipeline and statistics (OA: Object Addition OR: Object Replacement).

strate robust cross-domain and cross-task generalization, shedding light on future research on this domain.

2 Related Work

2.1 Image Editing

Recently, image editing techniques have significantly evolved, enabling users to intuitively modify images by selectively editing specific regions [5, 6, 12]. This differs from traditional image generation, as it demands understanding user intent and preserving original image semantics. However, the ease of creating realistic edited images has also increased misuse, including misinformation, fraud, and defamation, highlighting the urgent need for effective detection methods [7]. This work focuses on two main editing techniques: diffusion model-based editing and closed-source model editing.

- **Diffusion Model-Based Editing.** Diffusion models have greatly advanced image editing. MagicBrush fine-tunes InstructPix2Pix on a large-scale annotated dataset, significantly improving image quality [6, 5]. UltraEdit automatically generates extensive editing instructions using large language models (LLMs) and real images, enhancing dataset diversity [12]. GoT integrates reasoning-guided language analysis with diffusion models to enhance semantic and spatial coherence in edited outputs, demonstrating superior performance [20].
- **Closed-Source Model Editing.** Closed-source models, such as Google's Gemini-IG [13] and Flux AI's Magic Edit [21], provide advanced multimodal image generation and editing capabilities. Gemini-IG notably supports multimodal input and sophisticated editing tasks, while Magic Edit excels at interactive, chat-based editing. However, limited API access restricts their broader usage.

2.2 Fake Image Detection and Edited Region Localization

The proliferation of AI-generated content, particularly realistic manipulated images, has intensified misinformation risks [22, 23]. DE-FAKE integrates detection and attribution models to differentiate between real and fake images [24]. Systematic evaluations highlight that both humans and automated tools can effectively identify AI-generated images [25], but traditional binary classifiers struggle with subtle edits. To address this, zero-shot approaches like ZeroFake leverage stability differences during image inversion [26]. Although binary classification methods perform well, fine-grained detection is more important for edited images. Prior work, such as [10], trains segmentation models using pixel-level annotations, often automated with SAM [27], but still incurs high resource costs. To reduce this burden, we replace pixel-level masks with VLM-based inference of edited regions and objects, significantly lowering annotation overhead.

3 Dataset Construction

In this section, we describe the construction of the edited image detection dataset *FragFake*.

3.1 Construction Goals

As stated in Section 1.1, we first need to construct the dataset that can be used to train and evaluate the performance of edited image detection. The built dataset should have the following properties:

- **Diversity of Editing Models:** We include 4 image editing models: 1 closed-source commercial model (Gemini-IG [13]) and 3 open-source models (MagicBrush [6], GoT [12], and UltraEdit [12]).
- **Quality:** All four models produce highly realistic images, as illustrated in Figure 1. To ensure test set quality, we manually annotate 100 representative samples per subset, covering all 8 subsets (2 versions × 4 models), for a total of 800 images.
- **Diversity of Edited Objects:** As described in Section 3.2, we use GPT-4o to generate a wide range of editing instructions. To eliminate repetition of target objects, we apply filtering and re-query steps to create the Hard version, in which every target object is unique.

3.2 Construction Pipeline

To build a comprehensive dataset, we use the widely used COCO dataset [14], randomly sampling 20 images from each category ($20 \times 80 = 1600$ images in total) to serve as our base. Then we apply four image editing models to edit images, Figure 2 presents the pipline and dataset statistics.

Editing Instruction Creation. These image editing models operate via natural language instructions. Manually writing these instructions is both time-consuming and labor-intensive, so we use the pretrained VLM GPT4o-2024-11-20 (temperature set to 1) to generate them automatically. First, we apply a unified task template (refer to Appendix A.1) to produce initial editing prompts for 1 600 original images. Analysis shows that many target objects to be added or used to replace existing ones are repeated, for example "potted plant" appears 58 times (We refer to this subset as the Easy version). To reduce redundancy, we implement a target-object cache: if a newly generated instruction's target object is already in the cache, we append the prompt "Important: Please do NOT use the following object: [object]" and prompt GPT4o to regenerate the instruction. If the same object still recurs after three attempts, we discard this instruction. The remaining instructions and images form the Hard version, in which every target object is unique.

***FragFake*.** After generating the editing instructions, we process the images with each of the four editing models, i.e., MagicBrush, UltraEdit, GoT, and Gemini-IG. The former three are open-source models that we run locally on our environment, while Gemini-IG (also known as Gemini-2.0-flash-exp) is a closed-source commercial model whose inputs and outputs undergo content filtering. This filtering prevents some images from being edited, so the number of edited outputs from Gemini-IG is slightly lower than that of the other models, as shown in Figure 2. Once all edited images are obtained, we convert them into the image–text pair format required for VLM training. Each pair consists of the edited image and a corresponding model response that explicitly identifies the edited object; these pairs constitute the complete *FragFake*. We then manually review the edited images and their associated responses for each editing method to ensure correctness. For each subset, we randomly select 100 reviewed, edited images and their original counterparts (200 images in total) to form the test set. All remaining images are included in the training set without further filtering.

4 Experimental Settings

Evaluation Metrics. We evaluate image editing detection using two levels of metrics. The first level involves binary classification to determine whether an image has been edited, where we use *Accuracy* and *F1-score* as the metrics. These metrics are computed based on keyword matching. The second level includes fine-grained evaluation metrics: *Region Precision* and *Object Precision*. Region Precision measures whether the VLM correctly identifies the edited location, while Object Precision assesses whether the model correctly identifies the edited object. Since these metrics are difficult to evaluate using pretrained VLMs alone, we employ human annotation for accurate assessment. Specifically, Region Precision is a broader evaluation criterion than Object Precision. For example, during prediction, VLMs may generate an output that differs from the ground truth (GT) in wording, but human annotators can determine that the predicted object is located in the same region as the GT. Figure 9 provides an example of such human evaluation. Object Precision focuses on semantic alignment between the model's output and the ground truth. In human evaluation, we consider expressions with equivalent or highly similar meanings as correct matches. For instance, "vase of flowers" and "bouquet of flowers", or "fruit basket" and "bowl of fresh fruit", are judged as semantically consistent and therefore counted as true positives. The human evaluation results are based on cross-validation by two authors. On average, evaluating 5 samples takes approximately 1 minute. We utilize the Label Studio platform for manual annotation, as illustrated in Figure 12.

VLMs for Training. We select four VLMs for fine-tuning. For Llava-1.5, we use the llava-1.5-7b-hf version [28]. For Qwen2-VL [29] and Qwen2.5-VL [30], we use their respective 7B versions. For Gemma3, we use the gemma-3-4b-it model [31] for fine-tuning.

VLMs for Testing. In addition to the four open-source models mentioned above, we also select four powerful commercial closed-source models for evaluation: GPT-4o-mini (2024-07-18) [32], GPT-4o (2024-11-20) [15], GLM-4V (glm-4v-plus-0111) [33], and Gemini-2.5 (gemini-2.5-flash-preview-04-17) [34]. All models are accessed via their APIs with temperature set to 0.1.

Training Hyperparameters. We adopt LoRA [35] for model fine-tuning, which is a widely used and efficient parameter-efficient tuning method. The implementation is based on the Llama-Factory framework [36]. We utilize a single NVIDIA L20 GPU for training. The hyperparameter settings are as follows: the rank is set to 64, the learning rate is 5e-4, the number of training epochs is 5, and the batch size is 16. We use the final checkpoint obtained after training for evaluation. Since the Hard and Easy versions of *FragFake* contain different numbers of samples, we randomly sample 3,000 image pairs (edited image and corresponding original image in a 1:1 ratio) from each version for training. When the number of original images in a version is less than edited images, we supplement with non-redundant images from the COCO dataset to maintain sample balance.

5 Evaluation

5.1 Comparison of Different VLMs

Performance of Pretrained VLMs. VLMs with strong image understanding capabilities often perform well on downstream visual question answering tasks even without fine-tuning. To investigate their ability to directly identify whether an object in an image has been edited, we evaluate them on the Gemini-IG subset of the *FragFake* test set. We test two categories of models: (1) popular proprietary production VLMs, including GPT4o-mini, GPT4o, GLM-4V, and Gemini-2.5; and (2) widely used open-source VLMs, including Llava-1.5, Qwen2-VL, Qwen2.5-VL, and Gemma3. For this detection task, we design a unified prompt as demonstrated in Appendix A.2.

As shown in Table 1, GPT4o achieves the best performance among all detectors, reaching an Accuracy of 0.825 and an Object Precision of 46.0% on the Easy version of Gemini-IG, and an Accuracy of 0.810 with an Object Precision of 45.0% on the Hard version. GPT4o-mini also performs relatively

Table 1: Performance of pre-trained VLMs on Gemini-IG subset.

	Hard				Easy			
Model	Acc.	F1	RP	OP	Acc.	F1	RP	OP
GPT4o-mini	0.620	0.703	42.0%	42.0%	0.620	0.686	30.0%	30.0%
GPT4o	**0.810**	0.798	**45.0%**	**45.0%**	**0.825**	**0.813**	**46.0%**	**46.0%**
GLM-4V	0.535	0.614	35.0%	34.0%	0.470	0.558	26.0%	26.0%
Gemini-2.5	0.605	0.368	20.0%	20.0%	0.575	0.286	11.0%	11.0%
Llava-1.5	0.500	0.565	8.0%	8.0%	0.505	0.571	9.0%	9.0%
Qwen2-VL	0.805	**0.804**	25.0%	25.0%	0.810	0.808	22.0%	22.0%
Qwen2.5-VL	0.535	0.256	5.0%	5.0%	0.515	0.157	4.0%	4.0%
Gemma3	0.500	0.667	28.0%	27.0%	0.500	0.667	30.0%	30.0%

Note: Acc.: Accuracy, F1: F1-score, RP: Region Precision, OP: Object Precision.

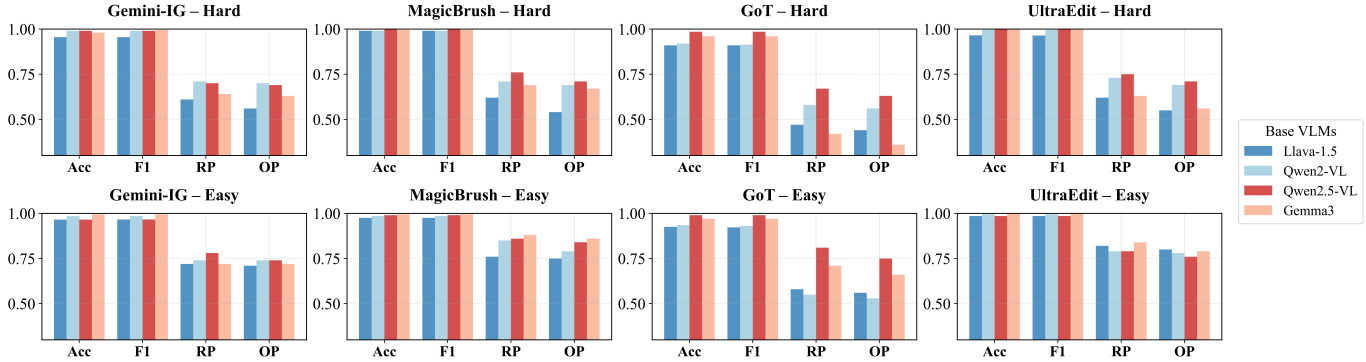


Figure 3: Performance comparison of different detectors on Hard and Easy versions.

well with an Object Precision of 42.0% on the Hard version but exhibits weaker binary classification performance (0.620 Accuracy). Qwen2-VL also demonstrates fair detection capability, achieving an Accuracy of 0.805 on the Hard version but only 25.0% Object Precision, indicating its limitations in fine-grained classification.

The remaining detectors perform rather poorly under both settings. For example, GLM-4V records an Accuracy of 0.535 and an Object Precision of 34.0% on the Hard version. Gemini-2.5 and Llava-1.5 achieve Object Precision values of only 20.0% and 8.0%, respectively, on the Hard version, showing almost no capacity for fine-grained object recognition. Qwen2.5-VL performs close to random guessing, with Object Precision close to 5.0% for both Hard and Easy versions, and is entirely unable to handle this task. Although Qwen2.5-VL outperforms Qwen2-VL on standard Visual Question Answering (VQA) benchmarks [18], it achieves lower scores in our fine-grained object detection task, demonstrating that our dataset differs fundamentally from traditional VQA benchmarks. Interestingly, although Gemma3 achieves only 0.500 in Accuracy, it still attains 27.0–30.0% in Object Precision, which is comparable to models like GPT4o-mini. This indicates that while Gemma3 struggles with classification tasks, it may still retain a certain degree of edited region localization ability. Overall, apart from the GPT4o series, existing pretrained VLMs still exhibit clear deficiencies in fine-grained classification and localization when faced with complex scenarios generated by Gemini-IG.

Performance of Fine-Tuned VLMs. We then fine-tune four open-source VLMs as detectors, Llava-1.5, Qwen2-VL, Qwen2.5-VL, and Gemma3, to verify whether fine-tuning enhances their ability to detect edited images. As shown in Figure 3, all detectors achieve excellent binary classification performance in terms of Accuracy and F1-score, demonstrating strong overall edit-detection capabilities. Qwen2.5-VL stands out most prominently. On the Hard version, it attains an average Accuracy/F1 of 0.994/0.994 and an average Region/Object Precision of 72.0%/68.5%; on the Easy version, the corresponding averages rise to 0.983/0.983 and 81.0%/77.3%, substantially outperforming the other detectors in fine-grained localization and object recognition. Moreover, on the Gemini-IG subset, Qwen2.5-VL attains Object Precision values of 69.0% for the Hard version and 74.0% for the Easy version, corresponding to improvements of 64.0% and 70.0% over the baseline in Table 1.

Gemma3 and Qwen2-VL follow closely behind, both maintaining strong performance on the Easy version, with average Object Precision exceeding 70.0%. However, Gemma3 shows a notable decline on the Hard version, where its average Object Precision drops to approximately 56.0%. In the Hard version of the GoT editing method, their Region Precision scores drop to 42.0% and 58.0% and their Object Precision scores to 36.0% and 56.0%, indicating that subtle modifications in this subset present greater challenges. In comparison, Llava-1.5 exhibits the weakest performance. Its Accuracy inches up from 0.955 on the Hard version to 0.963 on the Easy version, and its Object Precision stays low in both cases, with only 52.3% on the Hard and 70.5% on the Easy versions.

Table 2: Performance comparison of fine-tuned Llava-1.5 model trained on a 4,000-sample Gemini-IG subset using different data preparation strategies.

Sample Strategy	Hard				Easy			
	Acc.	F1	RP	OP	Acc.	F1	RP	OP
No Processing	0.885	0.878	54.0%	51.0%	0.670	0.731	73.0%	72.0%
Image Augmentation Only	0.970	0.970	59.0%	54.0%	0.950	0.950	67.0%	65.0%
Sampling from COCO Extras	0.965	0.965	64.0%	59.0%	0.975	0.975	74.0%	71.0%
Bootstrap Resampling	0.955	0.955	62.0%	57.0%	0.955	0.956	72.0%	70.0%

Note: Acc.: Accuracy, F1: F1-score, RP: Region Precision, OP: Object Precision.

Table 3: Performance comparison of popular vision backbones on the Gemini-IG (Easy version).

Metric	ResNet-50 [37]	DenseNet-121 [38]	MobileNet-V2 [39]	ViT-B/16 [40]	Inception-V3 [41]	ConvNeXt-Base [42]	Swin-B/4W7 [43]
Accuracy	0.890	0.905	0.855	0.940	0.905	0.995	**1.000**
F1-score	0.890	0.907	0.857	0.940	0.903	0.995	**1.000**

Note: ViT-B/16 = `vit_base_patch16_224`; Swin-B/4W7 = `swin_base_patch4_window7_224`.

Overall, the choice of editing method significantly influences detection difficulty. Compared to their pretrained counterparts, the fine-tuned VLMs not only achieve substantial gains in binary edit detection but also reach impressive levels in fine-grained localization and object identification. Although the Hard version's training and test sets contain disjoint editing targets, leading to relatively lower performance, each model still maintains an average binary classification Accuracy above 0.950 and satisfies practical requirements for fine-grained detection.

5.2 Ablation Study

Effect of LoRA Rank on Detection Performance. We use LoRA for fine-tuning, where different LoRA ranks correspond to varying numbers of additional trainable parameters. We therefore investigate how the LoRA rank affects detection performance.

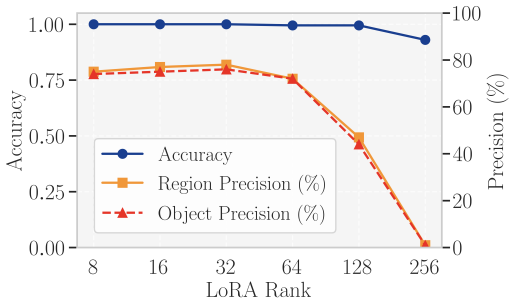


Figure 4: Gemma3 detector: performance across different rank settings on the Gemini-IG (Easy version) dataset.

Figures 4 and 14 show the trends in classification Accuracy, Region Precision and Object Precision on the Gemini-IG Easy version dataset as the LoRA rank increases. For Gemma3 (4 B parameters), performance improves with increasing rank and peaks at rank 32 with an Accuracy of 1.000, a Region Precision of 78.0% and an Object Precision of approximately 76.0%. Beyond this rank, all three metrics decline. For Qwen2.5-VL (7 B parameters), the best performance occurs at rank 8, with a Region Precision of 81.0% and an Object Precision of 78.0%. These results indicate that detectors of different scales and architectures require different amounts of trainable parameters, since larger models already possess greater expressive capacity and can adapt effectively with a smaller LoRA rank, whereas setting the rank too high can disrupt the pretrained capabilities of the base model.

Comparison of Different Data Balancing Strategies. Referring to Figure 2, our original dataset contains 1,600 images, with 100 held out as a test set. We now explore a scenario where the number of edited images in the training set (set to 2,000) exceeds the number of original images. In this case, we aim to determine which data balancing strategies yield the best performance. For the No Processing baseline, we train on all available data (1,500 original + 2,000 edited = 3,500 images), which yields 0.885 Accuracy with 51.0% Object Precision on the Hard version and 67.0% Accuracy with 72.0% Object Precision on the Easy version. To utilize the full 4,000-image budget and keep original and edited images balanced at 2,000 each, we expand the original set from 1,500 to 2,000 via one of three strategies: (1) Image Augmentation Only generates 400 new samples by applying random rotations, horizontal flips, and center crops to the 1,500 originals; (2) Sampling from COCO Extras draws 400 images from the remainder of the COCO dataset not used in the original split; (3) Bootstrap Resampling samples with replacement from the 1,500 originals until the set reaches 2,000 images.

As shown in Table 2, Image Augmentation Only raises Hard Accuracy to 0.970 (54.0% OP) and Easy Accuracy to 0.950 (65.0% OP). Sampling from COCO Extras achieves 0.965 Accuracy (59.0% OP) on Hard and 0.975 Accuracy (71.0% OP) on Easy, and delivers the highest Hard version Region Precision of 64.0%. Bootstrap Resampling yields 0.955 Accuracy (57.0% OP) on Hard and 0.956 Accuracy (70.0% OP) on Easy, demonstrating that even simple resampling of the limited original set can substantially improve performance.

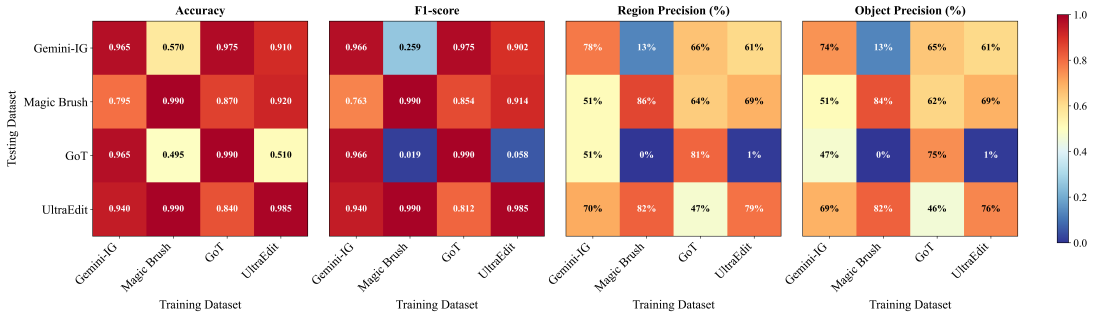


Figure 5: Cross-evaluation results of four datasets (Gemini, MagicBrush, GoT, UltraEdit) under the Easy setting, showing Accuracy, F1-score, Region Precision, and Object Precision. (Note: Fine-tuned model is Qwen2.5-VL)

Effect of Data Scale on Detection. We evaluate how model performance scales as the number of training images increases from 1,000 to 4,000 in the Gemini-IG subset, while maintaining a 1:1 ratio between original and edited images. We use the Gemma3 model fine-tuned with LoRA at rank 64.

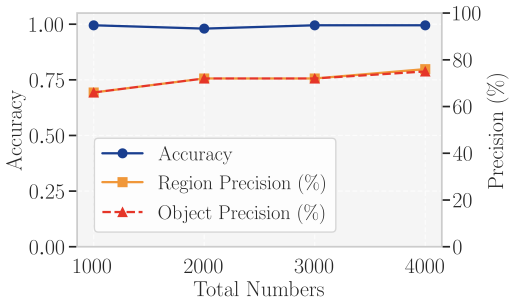


Figure 6: Scaling behavior of training dataset on Gemma3.

As shown in Figure 6, overall classification Accuracy remains nearly 1.000 across all sample sizes, with only a minor fluctuation at 2,000 images. In contrast, both Region Precision and Object Precision improve steadily as the dataset grows. Region Precision increases from 66.0% at 1,000 images to 76.0% at 4,000 images, while Object Precision rises from 66.0% to 75.0%. These findings indicate that although classification Accuracy saturates at an early stage, the more detailed metrics continue to benefit from larger training sets.

Comparison of Different Vision Backbones. We evaluate the performance of traditional vision backbones on the edited image detection task using the Gemini-IG Easy version dataset. The results are shown in Table 3. We compare seven visual backbones in two groups: traditional convolutional networks and transformer-based networks. The Accuracy of traditional convolutional networks (ResNet-50, DenseNet-121, MobileNet-V2 and Inception-V3) ranges from 0.855 to 0.905. MobileNet-V2 achieves the lowest Accuracy at 0.855, while DenseNet-121 and Inception-V3 both reach 0.905. Transformer-based backbones (ViT-B/16, ConvNeXt-Base and Swin-B/4W7) exhibit greater variation: ViT-B/16 attains 0.940, ConvNeXt-Base achieves 0.995, and Swin-B/4W7 achieves a score of 1.000. In VLMs, the current top performer Gemma3 also reaches 1.000 Accuracy on this task (see Figure 4 for the rank 32 results). However, as noted above, only detectors with more precise object descriptions can deliver greater value in practical applications.

5.3 Zero-Shot Transferability

In this section, we investigate the detectors' ability to generalize image edit detection to unseen editing scenarios without any additional fine-tuning.

Transferability on Different Datasets. We evaluate the zero-shot transferability of Qwen2.5-VL, as it consistently outperforms other fine-tuned models. As shown in Figures 5 and 13, detectors trained on Gemini-IG or GoT generalize best. Under the Easy setting, Gemini-IG achieves 0.965 Accuracy and 47.0% Object Precision when transferred to GoT, and retains 69.0% on UltraEdit but drops to 51.0% on MagicBrush. GoT-trained detectors reach 65.0%, 62.0%, and 46.0% Object Precision when transferred to Gemini-IG, MagicBrush, and UltraEdit, respectively. In contrast, MagicBrush-trained detectors show poor generalization, with Object Precision dropping to 13.0% on Gemini-IG and 0.0% on GoT, though retaining 82.0% on UltraEdit, suggesting similarity between those two. The same pattern holds for the Hard version, albeit with generally lower scores.

In summary, single-dataset detectors perform well in-domain but degrade across domains, highlighting the need for joint fine-tuning on diverse datasets for robust open-world detection.

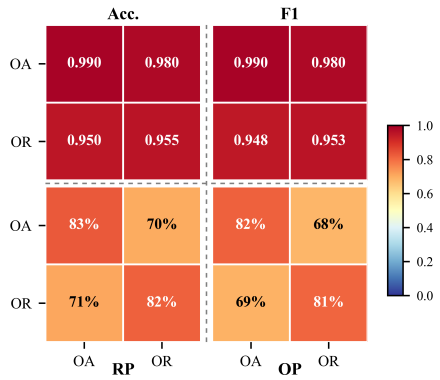


Figure 7: Transferability of Qwen2.5-VL detector on OA and OR tasks in the Gemini-IG dataset (Easy version).

Transferability on Different Editing Tasks. We train and evaluate the two task types, Object Addition (OA) and Object

Replacement (OR), separately using distinct training and test sets.

We select the Gemini-IG and UltraEdit datasets (Easy version) for this experiment. Each task-specific training set contains 2,000 images, and the corresponding test set contains 200 images, with a 1:1 ratio of original to edited images. The Qwen2.5-VL is also fine-tuned using LoRA for all settings.

As shown in Figure 7, on the Gemini-IG dataset, the detector trained on OA achieves an Accuracy of 0.990 and an Object Precision of 82.0%. When transferred to the OR task, it still maintains an Accuracy of 0.950 and an Object Precision of 69.0%. Similarly, the detector trained on OR also preserves high accuracy and transferability. These results indicate that the two task types share certain features and exhibit potential for cross-task generalization. Comparable findings are observed on the UltraEdit dataset, as shown in Figure 11.

6 Limitations

We construct the first systematic dataset, *FragFake*, specifically designed for detecting AI-edited images. However, several limitations remain:

- **Limited Types of Editing Targets.** In this paper, we only consider two types of editing operations: *Object Addition* and *Object Replacement*. However, image editing encompasses a broader range of modifications [4], such as *background changes* and *modification of emotional expressions*, which can be explored in future work.
- **Limited Diversity of Editing Methods.** Our dataset includes four representative image editing methods. While these cover a range of practical cases, many other editing techniques exist in the field and could be incorporated in subsequent studies to improve coverage and robustness.
- **Lack of Automated Filtering for Training Samples.** We do not apply filtering to the training data, as manual verification is highly labor-intensive. However, we observe that some editing outputs deviate from the original instructions, occasionally modifying unintended objects. These instances introduce noise into the training set. We believe that applying an efficient data filtering or quality assurance mechanism could further enhance the performance of the fine-tuned models.

7 Conclusion

Our work is the first to achieve edited region localization using VLMs without manual annotations. We design a fully automated and scalable dataset construction pipeline and release FragFake, the first VLM-oriented dataset for edited image detection with high diversity. We fine-tune VLMs using LoRA and conduct extensive evaluations, demonstrating that VLMs can effectively perform high-precision image edit detection and localization. Our results show strong transferability across domains and editing tasks, laying a solid foundation for future research in automated image forensics and tampering detection. We also discuss limitations and future directions in the Section 6.

References

[1] Hang Chen, Qian Xiang, Jiaxin Hu, Meilin Ye, Chao Yu, Hao Cheng, and Lei Zhang. Comprehensive exploration of diffusion models in image generation: a survey. *Artif. Intell. Rev.*, 58(4):99, 2025. 1

[2] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4):105:1–105:39, 2024. 1

[3] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE, 2022. 1

[4] Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, Wei Xiong, He Zhang, Shifeng Chen, and Liangliang Cao. Diffusion model-based image editing: A survey. *CoRR*, abs/2402.17525, 2024. 1, 8

[5] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 18392–18402. IEEE, 2023. 1, 3

[6] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 1, 2, 3

[7] Alexander Loth, Martin Kappes, and Marc-Oliver Pahl. Blessing or curse? A survey on the impact of generative AI on fake news. *CoRR*, abs/2404.03021, 2024. 1, 3

[8] MELISSA GOLDIN. Photo edited to make it appear secret service agents were smiling after attempt on trump's life. https://apnews.com/article/fact-check-trump-shooting-secret-service-smiling-photo-427049284678, 2024. Accessed: 2025-05-03. 1

[9] Hive Moderation. https://hivemoderation.com/. Accessed: 2025-05-15. 1

[10] Zhihao Sun, Haipeng Fang, Juan Cao, Xinying Zhao, and Danding Wang. Rethinking image editing detection in the era of generative AI revolution. In Jianfei Cai, Mohan S. Kankanhalli, Balakrishnan Prabhakaran, Susanne Boll, Ramanathan Subramanian, Liang Zheng, Vivek K. Singh, Pablo César, Lexing Xie, and Dong Xu, editors, *Proceedings of the 32nd ACM International Conference on Multimedia, MM 2024, Melbourne, VIC, Australia, 28 October 2024 - 1 November 2024*, pages 3538–3547. ACM, 2024. 1, 3

[11] Zongxia Li, Xiyang Wu, Hongyang Du, Fuxiao Liu, Huy Nghiem, and Guangyao Shi. A survey of state of the art large vision language models: Alignment, benchmark, evaluations and challenges, 2025. 2

[12] Haozhe Zhao, Xiaojian (Shawn) 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. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. 2, 3

[13] gemini-2.0-flash-exp. https://developers.googleblog.com/en/experiment-with-gemini-20-flash-native-image-generation/. 2, 3

[14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014. 2, 3

[15] OpenAI. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/, 2024. Accessed: 2025-05-13. 2, 4

[16] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 2

[17] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *CoRR*, abs/2409.12191, 2024. 2

[18] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *CoRR*, abs/2502.13923, 2025. 2, 5

[19] Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean-Bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Róbert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucinska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, and Ivan Nardini. Gemma 3 technical report. *CoRR*, abs/2503.19786, 2025. 2

[20] Rongyao Fang, Chengqi Duan, Kun Wang, Linjiang Huang, Hao Li, Shilin Yan, Hao Tian, Xingyu Zeng, Rui Zhao, Jifeng Dai, Xihui Liu, and Hongsheng Li. Got: Unleashing reasoning capability of multimodal large language model for visual generation and editing. *CoRR*, abs/2503.10639, 2025. 3

[21] Magic edit. https://flux1.ai/magic-edit/. 3

[22] Zhen Sun, Zongmin Zhang, Xinyue Shen, Ziyi Zhang, Yule Liu, Michael Backes, Yang Zhang, and Xinlei He. Are we in the ai-generated text world already? quantifying and monitoring AIGT on social media. *CoRR*, abs/2412.18148, 2024. 3

[23] Tao Zhang. Deepfake generation and detection, a survey. *Multim. Tools Appl.*, 81(5):6259–6276, 2022. 3

[24] Zeyang Sha, Zheng Li, Ning Yu, and Yang Zhang. DE-FAKE: detection and attribution of fake images generated by text-to-image generation models. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, pages 3418–3432. ACM, 2023. 3

[25] Anna Yoo Jeong Ha, Josephine Passananti, Ronik Bhaskar, Shawn Shan, Reid Southen, Hai-Tao Zheng, and Ben Y. Zhao. Organic or diffused: Can we distinguish human art from ai-generated images? In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, pages 4822–4836. ACM, 2024. 3

[26] Zeyang Sha, Yicong Tan, Mingjie Li, Michael Backes, and Yang Zhang. Zerofake: Zero-shot detection of fake images generated and edited by text-to-image generation models. In Bo Luo, Xiaojing Liao, Jun Xu, Engin Kirda, and David Lie, editors, *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, pages 4852–4866. ACM, 2024. 3

[27] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloé Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment anything. *CoRR*, abs/2304.02643, 2023. 3

[28] LLaVA Team. llava-1.5-7b-hf. https://huggingface.co/llava-hf/llava-1.5-7b-hf. Accessed: 2025-05-14. 4

[29] Qwen Team. Qwen2-vl-7b-instruct. https://huggingface.co/Qwen/Qwen2-VL-7B-Instruct. Accessed: 2025-05-14. 4

[30] Qwen Team. Qwen2.5-vl-7b-instruct. https://huggingface.co/Qwen/Qwen2.5-VL-7B-Instruct. Accessed: 2025-05-14. 4

[31] Google. gemma-3-4b-it. https://huggingface.co/google/gemma-3-4b-it. Accessed: 2025-05-14. 4

[32] OpenAI. Gpt-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, 2024. Accessed: 2025-05-13. 4

[33] Zhipu AI. Glm-4v model, 2024. Accessed: 2025-05-11. 4

[34] Google. Gemini 2.5 flash overview, 2024. Accessed: 2025-05-11. 4

[35] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. 4

[36] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *CoRR*, abs/2403.13372, 2024. 4

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June*

27-30, 2016, pages 770–778. IEEE Computer Society, 2016. 6

[38] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016. 6

[39] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018. 6

[40] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 6

[41] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision . In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826. IEEE Computer Society, 2016. 6

[42] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976. IEEE Computer Society, 2022. 6

[43] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021. 6

[44] Gabriela Ben Melech Stan, Estelle Aflalo, Raanan Yehezkel Rohekar, Anahita Bhiwandiwalla, Shao-Yen Tseng, Matthew Lyle Olson, Yaniv Gurwicz, Chenfei Wu, Nan Duan, and Vasudev Lal. Lvlm-interpret: An interpretability tool for large vision-language models. *arXiv preprint arXiv:2404.03118*, 2024. 12

A Task Template

A.1 Editing Instruction Creation Template

Editing Instruction Creation Template

VLM:

You are an expert visual instruction generator specialized in image editing tasks.
Task: Advanced Image Modification Instruction Generation
Description: Given a COCO dataset object label corresponding to an object in the image, along with the image and a modification goal (e.g., object addition), generate the following four outputs:
1. **Object Caption:** Produce a natural language caption describing the object (e.g., "A woman in the forest wearing a shirt with a drawing.").
2. **Brief Modification Instruction:** Provide a concise, imperative modification instruction (e.g., "Add a fighter jet flying nearby.").
3. **Symbolic Modification:** Output a succinct representation indicating only the addition or removal of objects. Use '+' for additions and '-' for removals. - For additions: "+fighter jet" - For removals: "-fighter jet" - For replacements: "-woman +man"
—
Input: - **COCO Object Label:** {coco_label} - **Image:** [Provided image] - **Modification Goal:** {object addition/object replacement}
—
Output (in valid JSON format): { "object_caption": "", "brief_modification_instruction": "", "symbolic_modification": "" }

A.2 Edited Image Detection Template

Edited Image Detection Template

VLM:

Task Description:
You are a visual analysis assistant. Your task is to examine a given image and determine whether any object in the image has been digitally modified or manipulated. Pay close attention to subtle inconsistencies in lighting, shadows, textures, edges, perspective, or logical composition. Carefully analyze these visual cues before making a judgment.
Instructions:
1. Provide a **detailed explanation** of your reasoning.
2. Then, based on your analysis, provide a **final result** in one of the following two formats: - If something has been edited: **"The thing in the image has been modified."** (Replace 'thing' with a brief, clear description of the modified object, e.g., "the cat," "the sky," "the left-side tree," etc.) - If nothing has been edited: **"Nothing has been modified in this image."**
Important Constraints:
- Your explanation must come before the result statement.
- Do not output both statements; only one final result should appear based on your judgment.
- Be cautious: minor edits may be hard to detect but should still be flagged if visible.

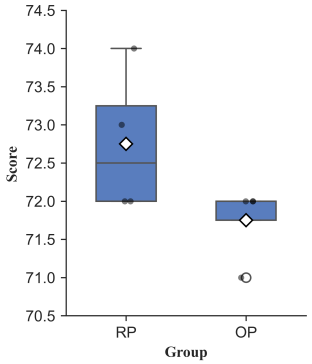


Figure 8: Box plot of repeated experiment results for RP and OP groups.

B Result of Classification in Hive Moderation

Hive Moderation is a commercial fake image detector. We conduct a manual test using 100 edited images from the Gemini-IG Easy test set. We observe that for some edited images, the AI likelihood reported by Hive Moderation is close to zero, as illustrated in Figure 10. Overall, only 55 out of 100 edited images are successfully detected. This result indicates that detectors trained primarily on fully generated images lack robustness when applied to partially edited content.

B.1 Interpretability Analysis

We perform interpretability analysis on edited images using LVLM-Interpret [44]. As shown in Figure 15, the model concentrates its attention on the cabin while generating the detection result. This alignment between the predicted modification ("cabin") and the corresponding visual region demonstrates the reliability and interpretability of the model's output.

C Repeated Experiment Validation

Since our evaluation dataset requires manual annotation, repeated testing is labor-intensive. Therefore, we do not perform multiple evaluation runs for each experiment. However, we conduct repeated trials using the Llava-1.5 detector trained on the Gemini-IG Easy version dataset, and observe that both RP and OP exhibit only minor fluctuations, as shown in Figure 8.

D Broader Impacts

Our work aims to improve the detection of partially edited images, which can help combat misinformation and support media integrity in an era of increasingly realistic AI-generated content. By enabling fine-grained localization of edits without human annotation, our method provides scalable tools for content verification and digital forensics.

However, such tools could also be misused—for instance, to improve the subtlety of manipulations by adversaries aware of detection mechanisms. Additionally, care must be taken to ensure fair and unbiased model behavior, as the dataset generation process may reflect biases in underlying models or prompts.

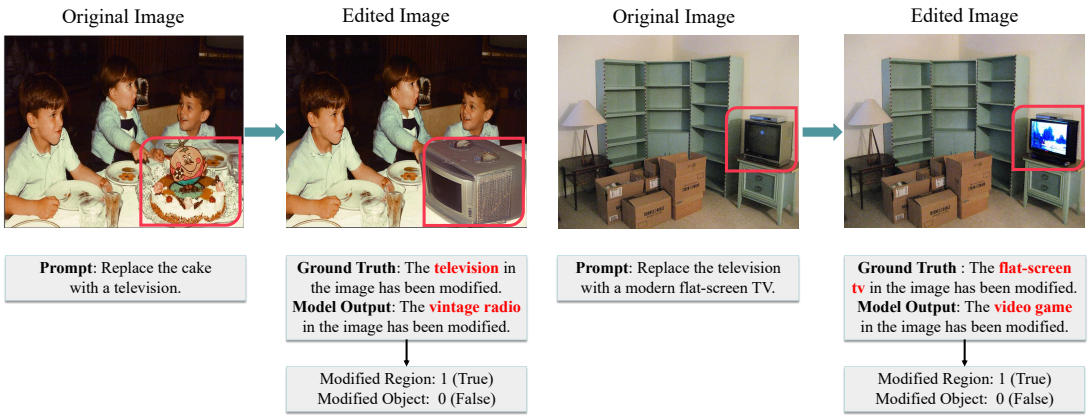


Figure 9: An example of human annotation about the Region Precision.

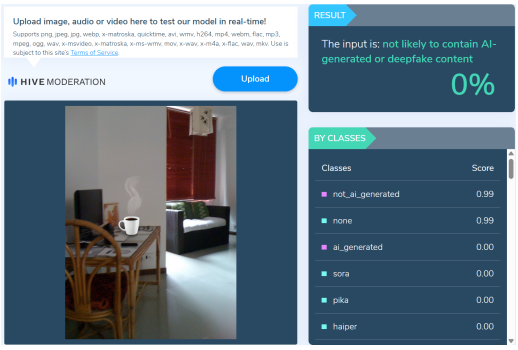


Figure 10: Detection of an edited image using Hive Moderation (ground truth: The coffee cup in the image has been modified.)

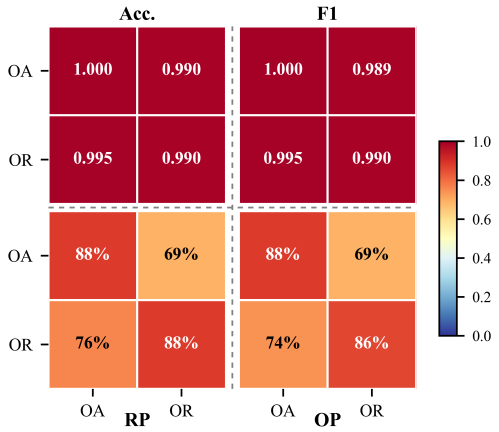


Figure 11: Cross-task performance of Qwen2.5-VL detector comparison between Object Addition (OA) and Object Replacement (OR) tasks on on UltraEdit.

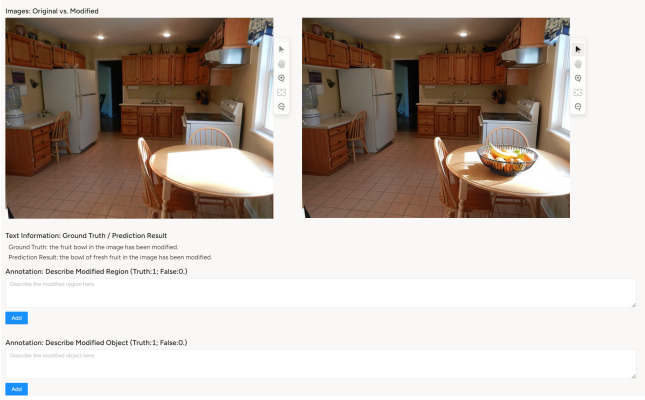


Figure 12: The annotation platform we built using Label Studio.

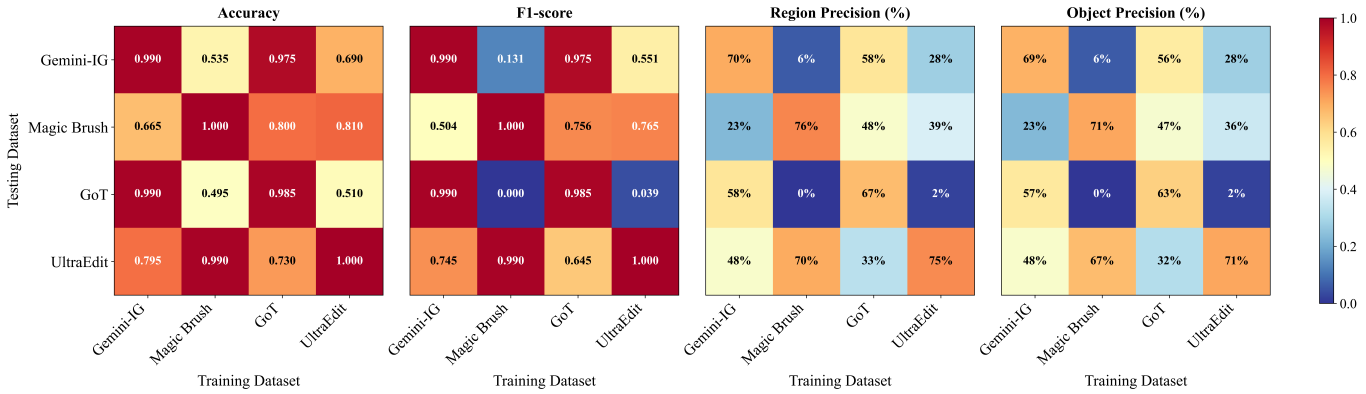


Figure 13: Cross-evaluation results of four datasets (Gemini, MagicBrush, GoT, UltraEdit) under the Hard setting, showing Accuracy, F1-score, Region Precision, and Object Precision. (Note: Finetuned model is Qwen2.5-VL)

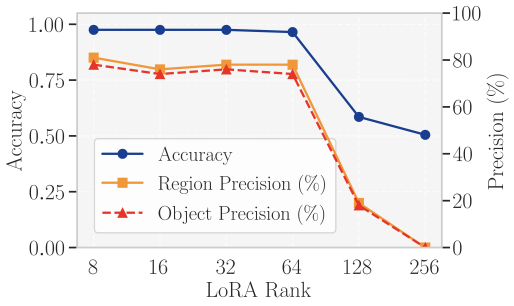


Figure 14: Qwen2.5-VL detector: performance across different rank settings on the Gemini-IG (Easy version) dataset.

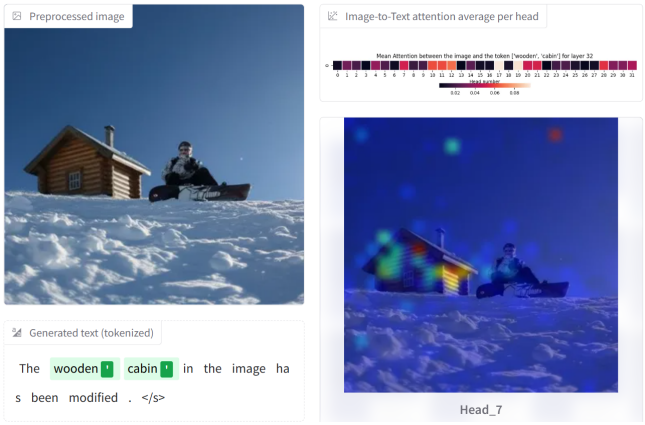


Figure 15: LVLM-Interpret is used to show the model's output for the edited image.