ManipShield: A Unified Framework for Image Manipulation Detection, Localization and Explanation

Zitong Xu
Shanghai Jiao Tong University
xuzitong@sjtu.edu.cn

Huiyu Duan *
Shanghai Jiao Tong University
huiyuduan@sjtu.edu.cn

Xiaoyu Wang
University of Electronic and Science Technology of China
2024300903005@std.uestc.edu.cn

Zhaolin Cai
Shanghai Jiao Tong University
sebqaq21@gmail.com

Kaiwei Zhang
Shanghai Jiao Tong University
zhangkaiwei@sjtu.edu.cn

Xiongkuo Min
Shanghai Jiao Tong University
minxiongkuo@sjtu.edu.cn

Guangtao Zhai
Shanghai Jiao Tong University
zhaiguangtao@sjtu.edu.cn

Abstract

With the rapid advancement of generative models, powerful image editing methods now enable diverse and highly realistic image manipulations that far surpass traditional deepfake techniques, posing new challenges for manipulation detection. Existing image manipulation detection and localization (IMDL) benchmarks suffer from limited content diversity, narrow generative-model coverage, and insufficient interpretability, which hinders the generalization and explanation capabilities of current manipulation detection methods. To address these limitations, we introduce **ManipBench**, a large-scale benchmark for image manipulation detection and localization focusing on AI-edited images. ManipBench contains over 450K manipulated images produced by 25 state-of-the-art image editing models across 12 manipulation categories, among which 100K images are further annotated with bounding boxes, judgment cues, and textual explanations to support interpretable detection. Building upon ManipBench, we propose **ManipShield**, an all-in-one model based on a Multimodal Large Language Model (MLLM) that leverages contrastive LoRA fine-tuning and task-specific decoders to achieve unified image manipulation detection, localization, and explanation. Extensive experiments on ManipBench and several public datasets demonstrate that ManipShield achieves state-of-the-art performance and exhibits strong generality to unseen manipulation models. The project is available at https://github.com/IntMeGroup/ManipShield.

1 Introduction

Image manipulation detection and localization (IMDL) have long attracted significant attention due to their crucial role in preserving image integrity and ensuring security [10, 40, 18, 76]. With the rapid advancement of generative models [13, 32], numerous powerful artificial intelligence editing techniques have emerged, such as NanoBanana [3] and Qwen-Image-Edit [73]. Unlike early deepfake techniques, which are typically limited to straightforward, malicious facial or identity

*Corresponding Author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: An overview of the constructed image manipulation database and the proposed image manipulation detection model, termed ManipBench and ManipShield, respectively. (a) We collect 22K real images and generate editing prompts via a multimodal large language models across 12 manipulation categories, producing 455K manipulated images using 25 editing models. (b) 100K images are further annotated with localization and explanations. (c) We propose ManipShield for detection, localization, and explanation. (d) We perform model comparisons based on ManipBench.

manipulations [10, 18, 40], recent new editing methods enable a broader range of modifications, including background changes, gesture alterations, text manipulations, *etc.*, potentially posing new security and privacy risks [63]. Furthermore, these editing methods are increasingly imperceptible, maintaining strong subject consistency while producing highly realistic images [74, 29, 73], which underscores the urgent need for more advanced image manipulation detection and localization approaches.

As shown in Table 1, existing IMDL datasets and benchmarks exhibit several critical limitations. **(i) Limited content diversity:** Existing datasets primarily focus on facial or identity forgeries [11, 51, 9, 80], whereas modern editing techniques allow personalized and localized modifications guided by flexible instructions [8, 73, 47]. The absence of such diverse editing categories limits the generalization of IMDL models on real-world manipulations. **(ii) Limited generative models:** most datasets rely on a small number of generative methods [76, 23, 7, 63], causing detection models to learn model-specific artifacts rather than generalizable features. Moreover, many of these methods are outdated, producing obvious artifacts, making detection easy but hindering the generalization of detection models to newer, more complex fakes. **(iii) Limited interpretability:** most benchmarks provide only binary real-or-fake labels [51, 9, 35, 80]. While some improved benchmarks include annotations of manipulated regions [11, 7], they still lack textual explanations to justify detection decisions, which limits the interpretability of model reasoning and constrains the development of more transparent manipulation detection methods. **(iv) Region-first pipeline:** Existing benchmarks predefine the manipulation region and conduct editing within a fixed mask [76, 23, 7], thereby restricting evaluation to mask-guided diffusion models and limiting their applicability to direct, instruction-driven editing scenarios where the manipulation region is not specified in advance.

To address the limitations of existing datasets, we introduce ManipBench, as shown in Figure 1, featuring diverse AI-edited images and fine-grained annotations, which has not been explored in prior benchmarks. **(i) Abundant content diversity:** ManipBench contains over 22K real source images covering diverse visual content. **(ii) Various manipulation tasks and numerous generative methods:** 21 state-of-the-art open-source editing models building on diverse diffusion backbones and 4 advanced closed-source editing methods are selected to produce manipulated images across 12 editing categories, resulting in over 450K manipulated images with diverse generative characteristics. **(iii) Fine-grained annotation:** ManipBench provides over 100K images meticulously annotated by trained annotators, using bounding boxes and corresponding judgment cues, along with textual explanations supporting the detection decisions. **(iv) Manipulation-first pipeline:** Unlike the region-first pipeline, we first apply the manipulations and then directly annotate the manipulated regions on the resulting images, benchmarking localization in more realistic, instruction-driven editing scenarios. As a multimodal dataset, ManipBench further enables comprehensive evaluation of MLLMs on detection, localization, and explanation.

Based on ManipBench, we propose an all-in-one IMDL model, ManipShield, which is capable of (1) discriminating manipulated images, (2) localizing manipulated regions, (3) identifying unrealistic attributes within each region as evidential support, and (4) providing comprehensive analysis. Building upon a Multimodal Large Language Model (MLLM), ManipShield leverages contrastive low-rank adaptation (LoRA) fine-tuning to train the vision encoder, utilizes layer discrimination selection to identify the most informative large language model layer, and decodes the hidden state of this layer using task-specific decoders to generate the target outputs. Extensive experiments on our ManipBench and other IMDL datasets demonstrate that ManipShield achieves state-of-the-art performance. Moreover, we show that it achieves strong zero-shot performance on unseen generative models, highlighting its superior generalization ability. In summary, our main contributions are:

- We introduce ManipBench, a large-scale IMDL dataset focusing on AI-edited images, comprising over 450K manipulated images produced by 25 advanced editing models, with fine-grained annotations including bounding boxes, judgment cues, and textual explanations.
- We benchmark the image manipulation detection, localization, and explanation capabilities of MLLMs based on ManipBench.
- We propose ManipShield, an MLLM-based model that unifies manipulation detection, localization and explanation, offering a transparent manipulation detection method.
- Extensive experiments demonstrate that our method outperforms most existing approaches on our ManipBench and other image manipulation datasets. We also validate its strong zero-shot performance on unseen editing models.

2 Related work

2.1 AI editing

With the advancement of generative models such as Stable Diffusion [58] and FLUX [13], numerous editing methods have emerged [22]. According to the type of editing prompts, these methods can be broadly categorized into description-based approaches (*e.g.*, "dog" $\rightarrow$ "cat") and instruction-based approaches (*e.g.*, "change the dog to a cat") [22, 77]. Description-based methods primarily rely on inversion-based modification [29, 15, 50, 27, 75], which first performs image-to-noise inversion to obtain a latent representation, then conducts generation according to the new prompt. Instruction-based methods, such as InstructPix2Pix [5] and MagicBrush [86] generally utilize large-scale paired editing datasets to train editing models. Recent models [73, 74] further adopt the DiT [32] architecture, enabling diverse and flexible editing instructions for generating highly realistic edited images.

2.2 Image manipulation detection datasets

As shown in Table 1, early datasets [11, 17, 71, 51] mainly focus on manual manipulations such as splicing, removal, and copy-move. With the rise of generative models, datasets [9, 61, 20, 4, 80] use GANs and diffusion models to produce deepfakes, but most are limited to facial content. Some datasets [35, 67, 66] extend manipulations to general images, yet still lacking text-guided edited images that incorporate novel manipulations and distinctive visual features. Some recent image editing datasets [76, 23, 7, 63] have incorporated text-guided edited images, but typically cover only a limited range of models and manipulation types. Moreover, since they rely on region-first pipeline, most datasets only include mask-guided editing methods [76, 23, 7], restricting to simple edits. Recent advances in MLLMs [70, 1, 37] have driven some benchmarks [35, 76, 23] to include textual explanations, offering interpretable insights into the manipulations, but the limited manipulation categories and models may restrict the generalization ability.

2.3 Image manipulation detection and localization

Early studies on IMDL mainly focus on localizing specific types of image manipulations, often limited to facial content [59, 36, 89, 31, 25]. However, since manipulation types vary widely in real-world scenarios, subsequent research has shifted toward universal manipulation localization methods [10, 40, 18, 85], which aim to identify artifacts and inconsistencies across diverse manipulation types. There are also various advanced network architectures that can be adapted for localization tasks [46, 62, 91, 57, 6, 82, 72]. With the rapid emergence of advanced editing models, recent methods [23, 63, 76, 7] have focused on detecting AI-edited images. However, most methods are trained on only a few editing models, and because different models produce distinct editing

Table 1: An overview of image manipulation detection datasets.

Databases	Image Content	Manipulation Categories	Manipulation Methods	Text-guided Editing Models	Manipulated Images	Text-guided Edited Images	Localization Annotations	Textual Explanations
IMD2020 [51]	General	3	3	✗	35,000	✗	✗	✗
DDFT [9]	Face	4	7	✗	240,336	✗	✗	✗
DF40 [80]	Face	4	40	✗	1M+	✗	✗	✗
FakeBench [35]	General	6	10	✗	3,000	✗	✗	3,000
FakeShield [76]	General	7	3	1	196,123	181,000	196,123	34,000
SID-Set [23]	General	2	1	1	200,000	200,000	200,000	3,000
GIM [7]	General	2	3	3	320,000	320,000	320,000	✗
FragFake [63]	General	2	4	4	20,000	20,000	✗	✗
ManipBench (Ours)	**General**	**12**	**25**	**25**	**455,000**	**455,000**	**100,000**	**100,000**

artifacts, their generalization is limited. Some works [23, 35, 76] further leverage LLMs to explain the rationale behind their detection and localization decisions, providing human-interpretable reasoning. Nevertheless, these approaches are typically designed for simple editing scenarios involving single manipulated regions [23, 76].

3 ManipBench

In this section, we introduce ManipBench, a large-scale IMDL dataset focusing on AI-edited images. The images are generated using multiple state-of-the-art editing methods and are annotated with bounding boxes, judgment cues, and detailed textual explanations.

3.1 Image collection

We first collect 22K high-resolution real-world images from publicly available photography websites along with their corresponding textual descriptions. we define 12 manipulation categories, including add, remove, replace, color, background, size, action, material, expression, scene text, handwritten text, and scanning text.

For each category, we employ the advanced MLLM, InternVL3.5 [70], to generate suitable editing prompts based on the image content and the manipulation task, then perform manual examination and revision. The prompts contain both an instruction type and a description type to accommodate different editing models. We employ 21 state-of-the-art editing models to perform manipulation. The details of these editing models are provided in Section 9.2 of the supplementary material. The whole generation process is shown in Figure 1(a). Finally, a total of over 455K AI-edited images are produced, forming a rich set suitable for benchmarking image manipulation detection.

3.2 Metadata acquisition

To enable fine-grained localization analysis, we select 100K images for detailed annotation. Unlike the region-first pipeline, which can automatically obtain the manipulation region, the manipulation-first pipeline requires human annotation. Defining the manipulation region is therefore critical. Image editing often alters areas beyond the target region to maintain global consistency, resulting in widespread pixel-level changes. For this reason, we define the manipulated regions as those exhibiting perceptual semantic changes relative to the original image. Here, semantic changes also include alterations in texture, color, or other visual attributes that differ from the original identity, even if the object category remains the same. We emphasize perceptual changes because modifications that do not produce perceptible semantic differences are unlikely to be significant and generally do not pose safety or functional risks.

Since the annotated regions are defined at the semantic level rather than pixel-level, precise masks are unnecessary. Instead, we use bounding boxes, which provide a flexible and practical way to capture complex or ambiguous modifications. During annotation, annotators are presented with paired real and manipulated images, draw bounding boxes around each modified region, and record cues that indicate the region has been manipulated. These cues are categorized into high-level features (shape, structure, relation, text, pose, expression, *etc.*) and low-level features (texture, blur, noise, light, detail, color, *etc.*).

Annotations are performed using a custom Python-based interface. Before starting, annotators have to receive training and finish a test to ensure consistency. Annotators work under controlled conditions, with images shown in randomized order and sessions divided into short rounds to minimize fatigue. Reliability is ensured through a three-stage protocol of independent labeling, cross-verification, and expert voting. For explanatory analysis, we first use ChatGPT-4o [53] to generate textual descriptions based on bounding boxes and cues, which are then manually reviewed and

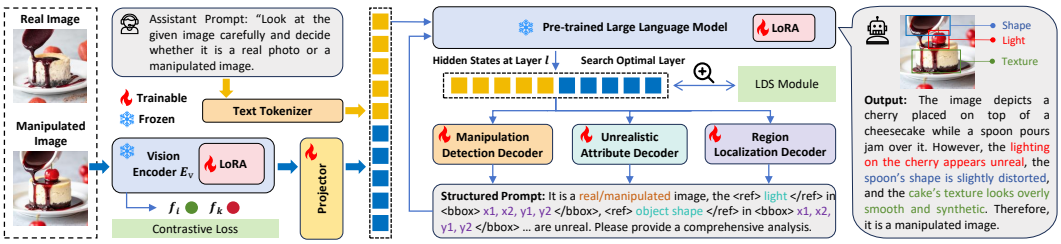


Figure 2: Overview of ManipShield. We first perform contrastive LoRA fine-tuning on paired real and manipulated images to train the vision encoder. The encoder is then frozen, and the projected image features, along with an assistant prompt, are fed into the LLM. The layer discrimination selection (LDS) module identifies the most discriminative layer, whose hidden state is used by three decoders for manipulation detection, unrealistic attribute extraction, and region localization. The outputs are combined into a structured prompt to generate explicit explanations.

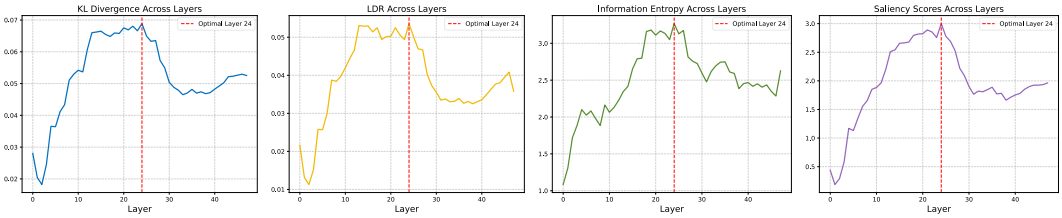


Figure 3: The plots of KL divergence, LDR, information entropy and saliency score across hidden states from different LLM layers.

revised. The complete metadata annotation process is illustrated in Figure 1(b). The explicit annotation process is described in Section 10 of supplementary material. Finally, we obtain 100K images with bounding-box annotations, judgment cues, and textual explanations.

4 ManipShield

In this section, we present ManipShield, a unified model for image manipulation detection, localization and explanation.

4.1 Overall architecture

As shown in Figure 2, ManipShield is built upon an MLLM architecture, *i.e.*, InternVL3.5 [70]. First, we apply **contrastive LoRA fine-tuning** to the vision encoder of a MLLM. Then, we perform **layer discrimination selection** using a part of samples to identify the optimal layer in the LLM that best separates positive and negative samples based on statistical distributions. The hidden state from this layer is fed into three specialized decoders: a **manipulation detection decoder** for classification, an **unrealistic attribute decoder** for cues extraction, and a **region localization decoder** for bounding boxes prediction. The outputs of these three decoders are then combined into a **structured prompt**, which ManipShield uses, together with the image, to generate explicit explanatory analysis.

4.2 Contrastive LoRA fine-tuning

Table 2 shows that the zero-shot performance of MLLMs on image manipulation detection is limited. Many studies have demonstrated that fine-tuning can significantly improve the visual capabilities of MLLMs [12, 78, 81, 56, 87], motivating us to fine-tune the pre-trained model. We adopt InternVL3.5 [70] as our backbone and apply the LoRA technique [21]. LoRA models the weight update of each layer $W \in \mathbb{R}^{d\times k}$ as a low-rank decomposition $\Delta W = AB$, where $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$ with $r \ll \min(d, k)$. The forward pass with LoRA is then given by

$$h = Wx + \Delta Wx = (W + AB)x. \tag{1}$$

To enhance the model's ability to discriminate between real and manipulated images, we construct positive and negative image pairs and employ contrastive learning. Let the feature embeddings of a positive pair (i, j) be denoted as f_i and f_j. The feature embeddings of all other images in the batch

are denoted as f_k ($k \neq i, j$) and serve as negative samples. The contrastive loss is formulated as:

$$\mathcal{L}_{\text{contrastive}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(\text{sim}(f_i, f_j)/\tau)}{\sum_{k\neq i,j}\exp(\text{sim}(f_i, f_k)/\tau)}, \tag{2}$$

where sim denotes the cosine similarity, τ is a temperature hyperparameter, and N is the batch size. All feature embeddings are ℓ_2-normalized before computing the similarity. To enable parameter-efficient adaptation with minimal modifications to the MLLM, we only fine-tuning the vision encoder, due to its pivotal role in visual feature extraction.

4.3 Layer discrimination selection

Recent studies have shown that the middle layers of MLLMs can reflect reasoning processes and yield better performance on visual tasks [16, 14]. Motivated by this, we systematically investigate the hidden state representations from different layers of MLLMs to identify the most informative layer for image manipulation detection. Specifically, we randomly select 1% of images from ManipBench and perform a single forward pass through the pre-trained MLLM, InternVL3.5 [70], whose vision encoder has been fine-tuned using our contrastive LoRA method. From each layer l, we extract the hidden state h_l of the first generation token. To quantify the information captured across layers statistically, we analyze properties of extracted hidden states h_l with following metrics:

Manipulation sensitivity via KL divergence: The Kullback–Leibler (KL) divergence quantifies the statistical distinguishability between features of real and manipulated images. For each feature dimension d at layer l, we assume that the hidden states of positive samples ($\mathbf{h}_l^N$) and negative samples ($\mathbf{h}_l^A$) follow Gaussian distributions $\mathcal{N}(\mu_{l,d}^N, (\sigma_{l,d}^N)^2)$ and $\mathcal{N}(\mu_{l,d}^A, (\sigma_{l,d}^A)^2)$, respectively. The KL divergence is computed between these two Gaussian distributions for each feature dimension, and the overall manipulation sensitivity for layer l is obtained by averaging across all feature dimensions D:

$$D_{\text{KL}}(l) = \frac{1}{D}\sum_{d=1}^{D} D_{\text{KL}}^{(d)}(l). \tag{3}$$

A higher $D_{\text{KL}}(l)$ indicates a greater distributional difference between features of real and manipulated images at layer l.

Class separability via local discriminant ratio: The Local Discriminant Ratio (LDR) measures the ability of features to linearly separate different classes. For each feature dimension d at layer l, we calculate a LDR as the ratio of the squared difference between the means of normal ($\mu_{l,d}^N$) and anomalous ($\mu_{l,d}^A$) features to the sum of their variances ($(\sigma_{l,d}^N)^2$ and $(\sigma_{l,d}^A)^2$), with a small constant ϵ added for numerical stability:

$$\text{LDR}^{(d)}(l) = \frac{(\mu_{l,d}^N - \mu_{l,d}^A)^2}{(\sigma_{l,d}^N)^2 + (\sigma_{l,d}^A)^2 + \epsilon}. \tag{4}$$

The overall class separability of layer l is the mean LDR across all D feature dimensions:

$$\text{LDR}(l) = \frac{1}{D}\sum_{d=1}^{D}\text{LDR}^{(d)}(l). \tag{5}$$

A higher $\text{LDR}(l)$ suggests stronger linear separability between the real and manipulation classes, implying more discriminative features at layer l.

Information concentration via feature entropy: To assess the information concentration within the feature representations, for each feature dimension d at layer l, we estimate the probability distribution by partitioning the feature values into a fixed number of B bins with evenly spaced boundaries determined by the range of feature values across all samples. The entropy for the d-th dimension is then calculated as:

$$H^{(d)}(l) = -\sum_{j=1}^{B} p(\mathbf{h}_l[d] \in \text{bin}_j)\log_2 p(\mathbf{h}_l[d] \in \text{bin}_j), \tag{6}$$

where $p(\mathbf{h}_l[d] \in \text{bin}_j)$ is the probability of the feature value falling into the j-th bin. The overall entropy for layer l is the average entropy across all D feature dimensions:

$$H(l) = \frac{1}{D}\sum_{d=1}^{D} H^{(d)}(l). \tag{7}$$

Table 2: Comparison results on different manipulation methods. ♠ standard CNN/Transformer baselines, ♡ image manipulation detection methods, ♣ AI-generated image detection methods, ♢ multimodal large language models. The fine-tuned results are marked with ∗. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Testing Subset	In-Distribution												Out-of-Distribution								Overall	
	SD1.4		SD1.5		SD3		SDXL		FLUX		DiT		FLUX-Kontext		NanoBanana		GPT-Image		SeedDream4			
Model/Metric	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
Random Choice	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00
♡HifiNet [18]	63.75	64.72	60.35	62.54	61.29	63.09	57.26	60.74	58.02	61.21	57.39	60.85	53.49	58.71	56.72	60.44	55.65	59.85	53.76	10.27	59.45	60.15
♡FakeShield [76]	73.11	74.43	67.81	70.66	67.34	70.33	60.48	66.21	61.34	66.82	60.21	66.11	54.57	63.02	54.84	63.16	54.84	63.16	53.40	11.73	64.45	66.81
♣Univ [52]	72.63	73.78	66.87	69.61	64.92	68.38	58.06	64.38	59.41	65.25	57.39	64.05	51.08	60.78	55.65	63.09	52.96	61.71	52.02	11.22	62.96	65.55
♣AIDE [79]	73.75	75.33	68.01	71.36	66.80	70.56	63.17	68.36	64.20	69.06	63.03	68.31	60.22	66.67	63.17	68.36	55.91	64.35	54.95	12.38	66.27	68.39
♢DeepSeekVL2 (small) [37]	53.05	66.47	51.81	65.87	51.88	65.91	52.96	66.41	52.06	65.99	50.27	65.16	49.73	64.92	48.12	64.19	48.66	64.43	53.36	17.63	50.20	63.52
♢InternVL3.5 (8B) [70]	69.89	71.32	64.78	67.67	66.80	68.95	57.53	63.43	60.08	64.96	67.61	69.62	62.10	66.02	64.25	67.32	54.84	61.99	55.96	11.80	64.10	65.44
♢Qwen3-VL (8B) [1]	69.53	70.68	64.78	67.18	67.20	68.73	56.18	62.18	58.83	63.79	68.28	69.68	59.95	64.27	61.83	65.37	53.76	60.91	56.06	11.20	63.52	64.65
♢Gemini2.5-Pro [8]	68.10	65.24	64.31	62.39	65.72	63.31	58.33	58.67	57.57	58.33	61.97	60.90	55.38	56.99	62.37	61.11	57.26	58.05	65.59	12.09	62.62	59.45
♢ChatGPT-4o [53]	71.23	71.98	71.10	71.83	68.81	70.26	70.16	71.17	67.38	69.32	66.66	68.92	66.94	69.02	67.47	69.37	68.01	69.72	65.01	14.41	68.99	68.29
♠ResNet50∗ [19]	88.80	80.11	88.98	80.15	88.84	80.12	88.71	80.10	88.62	80.08	86.56	79.70	65.32	74.77	58.06	72.48	56.99	72.11	68.01	75.52	84.33	79.05
♠Swin-T∗ [41]	90.59	82.80	90.52	82.79	90.59	82.80	89.78	82.67	90.10	82.72	89.52	82.63	62.63	76.90	66.13	77.85	62.37	76.82	66.94	78.06	86.17	81.90
♡MVSSNet∗ [10]	86.83	78.21	87.70	78.38	87.77	78.39	86.02	78.05	87.68	78.37	82.80	77.38	62.90	72.22	56.45	70.00	65.32	72.97	59.95	71.25	82.78	77.16
♡PSCCNet∗ [40]	90.99	83.68	91.06	83.69	90.99	83.68	91.13	83.70	91.08	83.70	89.11	83.39	66.40	78.91	59.41	77.00	57.26	76.34	63.71	78.22	86.19	82.70
♡HifiNet∗ [18]	92.34	87.29	93.08	87.38	92.74	87.34	92.20	87.28	92.88	87.36	89.65	86.96	72.04	84.28	66.94	83.28	61.02	81.95	69.89	83.87	88.42	86.67
♡FakeShield∗ [76]	92.97	87.37	93.01	87.37	92.88	87.36	92.74	87.34	93.10	87.38	89.38	86.93	75.54	84.89	68.28	83.55	66.94	83.28	63.44	82.52	88.80	86.73
♣CNNSpot∗ [68]	91.71	91.91	90.86	91.85	92.07	91.94	90.05	91.78	91.26	91.88	86.02	91.43	71.51	89.86	69.89	89.66	64.25	88.85	68.82	89.51	87.28	91.46
♣Lagrad∗ [64]	94.76	90.27	94.83	90.28	94.76	90.27	94.89	90.28	94.85	90.28	92.61	90.07	70.43	87.33	69.89	87.25	64.25	86.28	70.16	87.29	90.45	89.74
♣Univ∗ [52]	93.10	87.83	93.35	87.86	93.41	87.86	93.01	87.82	93.50	87.87	92.20	87.72	75.54	85.41	65.59	83.56	63.98	83.22	73.12	85.00	89.42	87.27
♣AIDE∗ [79]	**95.97**	**94.20**	**96.17**	94.21	**96.24**	94.21	94.62	94.12	**96.01**	94.20	**93.82**	**94.07**	**76.61**	**92.83**	72.85	92.49	75.00	**92.69**	75.81	92.76	92.46	**93.95**
♢DeepSeekVL2 (small)∗ [37]	94.65	93.21	95.16	93.27	91.68	93.74	93.25	94.50	92.16	92.51	91.37	94.26	73.85	88.94	73.28	89.87	71.52	85.70	75.06	86.96	88.54	89.30
♢InternVL3 (8B)∗ [88]	93.26	93.50	95.17	**94.33**	92.39	94.45	95.47	**94.62**	92.17	93.53	94.15	94.38	74.84	89.06	**77.17**	**92.62**	72.10	85.58	**76.49**	**93.62**	89.74	90.13
♢Qwen3-VL (8B)∗ [1]	95.14	93.26	94.17	94.29	95.27	**94.75**	**96.42**	92.38	94.53	**95.17**	93.28	93.51	76.24	91.18	76.68	91.88	**79.37**	90.13	76.48	92.25	**92.58**	93.66
ManipShield (Ours)∗	**97.27**	**95.26**	**97.51**	**95.27**	**97.18**	**95.26**	**97.31**	**95.26**	**97.36**	**95.27**	**96.91**	**95.24**	**81.45**	**94.39**	**80.91**	**94.36**	**82.57**	**94.27**	**81.18**	**94.38**	**94.66**	**95.12**

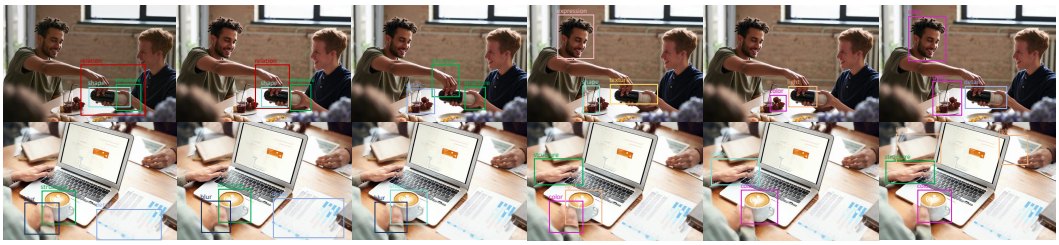


Figure 4: Examples of bounding box prediction results.

Higher values indicate more uniform distribution of feature values across bins and capturing more diverse information.

To effectively combine these metrics, we apply Z-score normalization across all layers for KL divergence, LDR, and Entropy. For a metric $M \in \{D_{KL}(l), LDR(l), H(l)\}$, the normalized score $\hat{M}(l) = \frac{M(l)-\mu_M}{\sigma_M}$. The saliency score $S(l)$ for each layer is then calculated by:

$$S(l) = \hat{D}_{KL}(l) + L\hat{D}R(l) + \hat{H}(l). \tag{8}$$

The optimal layer is determined as the one that maximizes the saliency score. As shown in Figure 12.2, the saliency score peaks at layer 24, where the KL divergence, LDR, and information entropy also attain their peak values, indicating that layer 24 provides the most distinctive features between real and manipulated images. A more detailed discussion of these metrics is provided in Section 12.2 of supplementary material.

4.4 Task-specific decoders

After selecting the optimal layer, three specialized decoders are introduced to interpret the hidden state of the first generated token from that layer. The manipulation detection decoder focuses on distinguishing real and manipulated images. The unrealistic attribute decoder captures diverse cues that reflect unrealistic characteristic, while the region localization decoder predicts the corresponding spatial regions through bounding-box regression. All three decoders are based on a similar multi-layer perception (MLP) and are optimized with task-specific objectives. In this process, LoRA module is applied solely to the LLM. The outputs are aggregated into a structured prompt. This prompt is returned to ManipShield along with the image, allowing it to generate explicit analysis. With these task-specific decoders, ManipShield operates as a unified model for manipulation detection, localization, and explanation.

Table 3: Comparison results (IoU↑) of bounding box prediction. ♣ object detection methods, ♢ multimodal large language models. The fine-tuned results are marked with ∗. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

	In-Distribution						Out-of-Distribution				
Model/Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Overall
♢DeepSeekVL2 (small) [37]	0.148	0.151	0.155	0.172	0.129	0.149	0.152	0.136	0.147	0.133	0.141
♢InternVL3.5 (8B) [70]	0.201	0.197	0.184	0.209	0.137	0.182	0.188	0.141	0.154	0.160	0.168
♢Qwen3-VL (8B) [1]	0.194	0.194	0.178	0.196	0.133	0.186	0.153	0.157	0.151	0.168	0.163
♢Gemini2.5-Pro [8]	0.257	0.287	0.264	0.272	0.198	0.252	0.234	0.228	0.213	0.231	0.234
♢ChatGPT-4o [53]	0.256	0.261	0.245	0.267	0.184	0.231	0.204	0.217	0.220	0.239	0.221
MVSSNet [10]	0.327	0.306	0.312	0.337	0.295	0.308	0.312	0.286	0.293	0.315	0.311
PSCCNet [40]	0.341	0.320	0.308	0.332	0.316	0.328	0.312	0.327	0.305	0.291	0.326
HifiNet [18]	0.387	0.374	0.368	0.379	0.358	0.336	0.341	0.320	0.322	0.324	0.361
FakeShield [76]	0.406	0.392	0.386	0.411	0.395	0.387	0.371	0.376	0.364	0.373	0.388
♣Fast R-CNN∗ [57]	0.435	0.451	0.413	0.456	0.413	0.437	0.349	0.310	0.312	0.358	0.378
♣DETR∗ [6]	0.661	0.696	0.633	0.645	0.574	0.638	0.521	0.491	0.478	0.510	0.573
♣DeformableDETR∗ [90]	0.701	0.737	0.716	0.740	**0.694**	0.718	0.677	0.617	0.621	0.629	0.674
♣GroundingDINO∗ [39]	**0.751**	**0.792**	**0.773**	**0.779**	0.688	0.706	0.661	0.658	0.675	**0.685**	0.708
♢DeepSeekVL2 (small)∗ [37]	0.673	0.719	0.694	0.702	0.655	0.650	0.655	**0.671**	0.653	0.654	0.697
♢InternVL3 (8B)∗ [88]	0.703	0.726	0.708	0.730	0.675	0.696	**0.683**	0.642	0.657	0.670	0.704
♢Qwen3-VL (8B)∗ [1]	0.743	0.775	0.750	0.764	0.680	**0.727**	0.682	0.651	**0.676**	0.684	**0.716**
ManipShield (Ours)∗	**0.778**	**0.798**	**0.792**	**0.787**	**0.725**	**0.753**	**0.711**	**0.701**	**0.709**	**0.716**	**0.749**

Table 4: Comparison results (CSS↑) of explanation capabilities. All embeddings are extracted using the InternVL3.5 [70]. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

	Testing Subset										
Model/Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Overall
LLaVA1.6 (7B) [38]	0.518	0.515	0.510	0.505	0.495	0.502	0.497	0.482	0.501	0.512	0.504
Ovis2.5 (9B) [44]	0.645	0.638	0.641	0.634	0.637	0.620	0.618	0.624	0.615	0.622	0.635
DeepSeekVL2 (small) [37]	0.512	0.508	0.506	0.471	0.477	0.482	0.482	0.478	0.455	0.464	0.484
Qwen3-VL (8B) [1]	0.718	0.720	0.706	**0.722**	0.713	**0.709**	0.697	0.702	**0.721**	0.715	0.714
InternVL3.5 (8B) [70]	0.685	0.672	0.682	0.653	0.659	0.662	0.672	0.661	0.653	0.676	0.665
ChatGPT-4o [53]	**0.739**	**0.732**	**0.736**	0.721	**0.715**	0.702	**0.714**	**0.715**	0.711	**0.717**	**0.721**
Gemini2.5-Pro [8]	0.663	0.658	0.592	0.671	0.624	0.604	0.629	0.625	0.631	0.637	0.632
ManipShield (Ours)	**0.826**	**0.814**	**0.837**	**0.831**	**0.823**	**0.818**	**0.826**	**0.806**	**0.814**	**0.829**	**0.815**

5 Experiment

5.1 Experiment setup

We report accuracy (Acc) and F1 for image manipulation detection, Intersection over Union (IoU) for localization, and Cosine Semantic Similarity (CSS) for explanation. A default threshold of 0.5 is applied for detection and localization. We employ a diverse set of methods for comparison. For detection, standard CNN and Transformer architectures are used as baselines, while image manipulation detection methods, AI-generated image detection methods, and MLLMs are also included in the evaluation. For localization, we compare our method with object detection approaches and MLLMs, while for explanation, we benchmark it against several advanced MLLMs. For all fine-tuned models, we adopt the same training, validation, and testing split (4:1:1), and fine-tune other MLLMs using the same strategy as our model. The detailed training configuration of our model is provided in Section 12 of supplementary material.

5.2 Comparison results on ManipBench

The **detection** results are shown in Table 2, where in-distribution images are further grouped based on the backbones of the editing models. We observe that both image manipulation and AI-generated image detection methods perform poorly in zero-shot settings due to unseen feature distributions. Advanced MLLMs also show limited detection ability. After fine-tuning on our dataset, however, all methods improve significantly, underscoring the importance of ManipBench. ManipShield achieves the best results, while other fine-tuned MLLMs remain competitive, demonstrating the adaptability and effectiveness of our approach. Table 2 also presents the results for the out-of-distribution subset. Most methods show a performance drop, highlighting the challenge of generalizing to unseen editing methods, while ManipShield still demonstrates strong performance.

Table 3 shows the **localization** results on both the in-distribution and out-of-distribution subsets. For those manipulation methods that output masks, we infer the bounding boxes from the masks and calculate the IoU results. We observe that the zero-shot performance of MLLMs is weak, indicating limited spatial reasoning ability and low sensitivity to manipulation regions. Although object detection methods perform well after training, they have difficulty predicting judgment cues as shown in Figure 4. Furthermore, the results on the out-of-distribution subset reveal poor generalization for these methods. In contrast, our method achieves the best performance across all subsets, demonstrating strong localization capability and robust generalization to unseen manipulation patterns.

Table 5: Explanation capabilities comparison based on human evaluation.

Model/Subset	Testing Subset										
	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Overall
LLaVA1.6 (7B) [38]	28.8	26.4	25.1	28.4	26.3	27.1	25.7	26.4	26.7	26.2	26.8
Ovis2.5 (9B) [44]	39.4	38.6	42.4	38.2	41.5	38.0	40.2	38.3	39.2	38.0	39.5
DeepSeekVL2 (small) [37]	45.8	46.2	45.7	52.5	46.0	52.4	44.6	45.7	45.2	50.6	48.6
Qwen3-VL (8B) [1]	66.6	64.8	65.7	64.8	66.2	61.4	62.8	64.2	65.5	64.2	65.3
InternVL3.5 (8B) [70]	68.0	67.5	66.5	64.3	68.6	64.2	65.2	65.4	67.4	63.6	65.8
ChatGPT-4o [53]	77.1	75.9	74.6	**74.5**	**77.3**	74.5	76.8	**78.4**	**76.3**	77.6	**77.5**
Gemini2.5-Pro [8]	**78.6**	**76.3**	**74.9**	70.6	72.8	**76.4**	**79.6**	75.2	75.6	**79.5**	77.2
ManipShield (Ours)	**84.7**	**85.2**	**84.5**	**85.6**	**82.9**	**84.8**	**83.2**	**85.0**	**84.3**	**83.9**	**84.6**

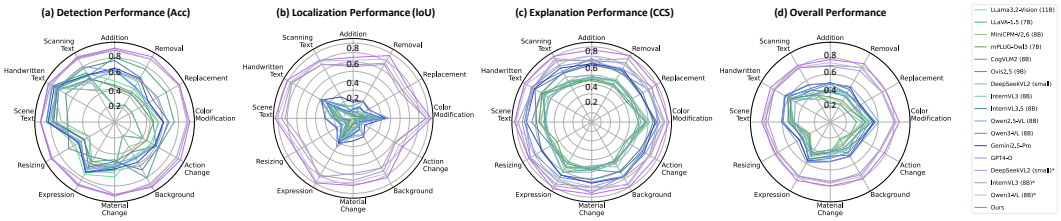


Figure 5: Comparison of performance of different MLLMs in terms of image manipulation detection, localization, and explanation, respectively.

Table 6: Ablation study on the different backbones, decoders and LoRA tuning strategy.

Backbone&Strategy					Metrics		
Backbone	Decoders	LoRA(vision)	LoRA(llm)	LDS	Acc↑	IoU↑	CSS↑
InternVL3.5 [70]		✓	✓		90.3	0.225	0.347
InternVL3.5 [70]	✓		✓		92.6	0.714	0.852
InternVL3.5 [70]	✓	✓			85.3	0.379	0.231
InternVL3.5 [70]	✓	✓	✓		93.5	0.732	0.859
InternVL3.5 [70]	✓	✓	✓	✓	**94.7**	**0.765**	**0.892**
InternVL3 [88]	✓	✓	✓	✓	92.4	0.729	0.846
DeepSeekVL2 [37]	✓	✓	✓	✓	88.5	0.693	0.812
Qwen3-VL [1]	✓	✓	✓	✓	92.8	0.745	0.861

To assess the quality of **textual explanation**, we compare the performance of pre-trained MLLMs with our ManipShield. We not only compare the CSS with the explanation texts in ManipBench, as shown in Table 4, but also conduct a subjective study to evaluate the generated explanations. Five experts are recruited to rate the explanations. After removing outliers, the final results were obtained by averaging the remaining scores, as reported in Table 5. The detailed procedure is described in Supplementary Section 7.4. Our approach consistently achieves the best performance across all subsets. While some MLLMs can identify manipulated regions, they often struggle to precisely specify the type of inconsistency.

Figure 5 further presents the image manipulation detection, localization, explanation, and overall performance of MLLMs, demonstrating that ManipBench also serves as a benchmark for evaluating MLLM capabilities. Without fine-tuning, MLLMs perform poorly on regional manipulations, such as addition, removal, replacement, and resizing, likely due to their limited sensitivity to local semantic features. After fine-tuning, their performance improves across all categories, with our ManipShield still achieving superior results.

5.3 Ablation results

To validate the effectiveness of each component in ManipShield, we perform comprehensive ablation studies, with the results summarized in Table 6. Rows 1 and 4 demonstrate the task-specific decoders in our model can dramatically improve the performance. As shown in Rows 2–4, applying LoRA to both the vision encoder and the LLM leads to the best overall performance. The layer discrimination selection (LDS) module further enhances performance, as evidenced in Row 5. Finally, Rows 5–8 compare different MLLM backbones of similar parameter scales, where the InternVL3.5 backbone used in our model achieves the best results.

5.4 Generalization ability

From the experimental results on ManipBench, we observe that existing image manipulation detection methods suffer from a clear performance drop when evaluated on images generated by unseen models. To further analyze this issue, we conduct experiments to assess their generalization ability across different generation backbones. Specifically, each model is trained on images from one backbone and tested on others.

Table 7: Accuracy results (Acc↑) of cross-validation on different training and testing subsets using ResNet50 [19] and our ManipShield. Proprietary subset includes images generated by FLUX-Kontext [30], NanoBanana [8], GPT-Image [54] and SeedDream4 [60].

Model	ResNet-50							ManipShield						
Train/Test on	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	Proprietary	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	Proprietary
SD1.4	**93.19**	85.75	72.04	86.69	68.77	62.50	58.80	**94.31**	90.59	83.33	85.89	74.10	73.79	67.74
SD1.5	88.84	**92.11**	66.40	70.16	70.39	64.25	60.28	91.85	**92.92**	69.35	69.09	79.12	70.43	64.31
SD3	76.39	72.31	**91.67**	71.10	62.99	61.02	59.81	80.33	79.21	**90.86**	81.59	67.38	65.59	69.42
SDXL	77.24	68.28	60.22	**92.47**	63.98	63.31	58.87	80.51	70.88	78.49	**92.47**	65.50	64.92	62.70
FLUX	71.06	74.73	70.97	77.42	**93.41**	77.15	63.04	85.39	85.75	72.85	83.47	**92.07**	79.03	73.86
DiT	69.80	66.13	66.13	73.66	71.33	**92.61**	63.51	76.48	70.34	75.54	76.88	81.32	**95.70**	76.55
Full Dataset	88.80	88.98	88.84	88.71	88.62	86.56	62.10	88.80	88.98	88.84	88.71	88.62	86.56	81.53

Table 8: Accuracy (Acc↑) on different testing subsets, averaged over models trained on *different training subsets*. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Model	Testing Subset										Avg
	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	
ResNet50 [19]	79.42	76.55	71.24	78.58	71.81	70.14	59.90	60.30	58.47	64.20	69.37
Swin-T [41]	78.00	79.02	71.51	77.42	69.85	70.83	58.15	62.19	58.38	63.04	68.84
MVSSNet [10]	76.15	74.34	65.19	73.70	66.05	67.18	59.59	56.90	57.30	61.29	65.77
HiFiNet [18]	84.01	76.67	74.64	79.41	75.11	69.83	65.10	61.07	58.11	64.87	70.88
CNNSpot [68]	76.40	72.25	71.33	75.16	71.28	71.82	57.84	56.50	57.89	64.34	67.48
AIDE [79]	**82.06**	**77.17**	**76.34**	**80.73**	**75.07**	**71.10**	**65.50**	**64.56**	**64.47**	**68.86**	**72.59**
ManipShield (Ours)	**84.81**	**81.62**	**78.41**	**81.56**	**76.58**	**74.91**	**71.01**	**66.85**	**68.32**	**70.21**	**75.43**

Table 9: Comparison of performance (Acc↑) on other image manipulation datasets. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Dataset	CASIA2 [11]		IMD2020 [51]		DDFT [9]		DF40 [80]	
Model/Metric	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
MVSSNet [10]	82.53	83.06	85.25	86.24	79.46	80.26	82.71	82.54
PSCCNet [40]	89.41	87.28	80.60	82.10	85.64	86.05	86.90	87.32
HifiNet [18]	87.16	86.50	86.38	87.15	84.32	85.21	85.25	86.00
FakeShield [76]	**91.65**	**92.11**	**87.43**	**86.25**	86.29	87.05	85.65	87.10
Univ [52]	86.32	87.23	84.34	86.20	**89.22**	**90.15**	85.24	85.87
AIDE [79]	88.47	90.05	86.76	87.56	85.18	88.03	**88.37**	**89.17**
ManipShield (Ours)	**93.71**	**93.08**	**92.53**	**92.88**	**91.62**	**93.14**	**92.46**	**93.25**

As shown in Table 7, although most methods perform well on test data from the same backbone, their accuracy decreases significantly when applied to unseen backbones. This suggests that these methods tend to overfit to backbone-specific artifacts rather than learning general manipulation cues. In contrast, ManipShield captures more high-level inconsistency features and generalizes well across different generation models. Table 8 further confirms this robustness, showing consistently strong performance across training-testing backbone shifts.

5.5 Comparison results on other datasets

We further evaluate the performance of ManipShield on several widely used manipulation datasets to assess its general applicability. CASIA2 [11] and IMD2020 [51] are well-known manually manipulated image datasets, while DDFT [9] and DF40 [80] are representative deepfake datasets focusing on identity alteration. We adopt the same training and testing split (4:1) for all datasets to ensure fair comparison. As shown in Table 9, ManipShield achieves best results, demonstrating its strong generalization ability and potential as a unified solution for general image manipulation detection.

6 Conclusion

In this paper, we introduce ManipBench, a large-scale benchmark for image manipulation detection focusing on AI-edited images, containing 450K+ manipulated samples and 100K+ fine-grained annotations. Based on ManipBench, we propose ManipShield, a unified model for image manipulation detection, localization, and explanation. Extensive experiments demonstrate that ManipShield overcomes existing methods, showing state-of-the-art performance and strong generalization ability.

7 Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grants 62401365, 62225112, 62522116, 62322302, 62273011, 62271312, 62132006, and U24A20220, and in part by the China Postdoctoral Science Foundation under Grants BX20250411 and 2025M773473, and in part by the STCSM under Grant 22DZ2229005, and in part by the National Key Research and Development Program of China under Grant 2024YFE0100500.

References

[1] Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., et al.: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025)

[2] Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., et al.: Qwen2.5-vl technical report. arXiv preprint arXiv:2502.13923 (2025)

[3] Banana, N.: Nano banana: Free online ai image editor (2025), `https://nanobanana.io`

[4] Bhattacharyya, C., Wang, H., Zhang, F., Kim, S., Zhu, X.: Diffusion deepfake. arXiv preprint arXiv:2404.01579 (2024)

[5] Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18392–18402 (2023)

[6] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision (ECCV). pp. 213–229 (2020)

[7] Chen, Y., Huang, X., Zhang, Q., Li, W., Zhu, M., Yan, Q., et al.: Gim: A million-scale benchmark for generative image manipulation detection and localization. In: Proceedings of the Conference on Association for the Advancement of Artificial Intelligence (AAAI). vol. 39, pp. 2311–2319 (2025)

[8] Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., et al.: Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv preprint arXiv:2507.06261 (2025)

[9] Dang, H., Liu, F., Stehouwer, J., Liu, X., Jain, A.K.: On the detection of digital face manipulation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5780–5789 (2020)

[10] Dong, C., Chen, X., Hu, R., Cao, J., Li, X.: Mvss-net: Multi-view multi-scale supervised networks for image manipulation detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **45**(3), 3539–3553 (2022)

[11] Dong, J., Wang, W., Tan, T.: Casia image tampering detection evaluation database. In: Proceedings of the IEEE China summit and international conference on Signal and Information Processing (ChinaSIP). pp. 422–426. IEEE (2013)

[12] Duan, H., Hu, Q., Wang, J., Yang, L., Xu, Z., Liu, L., et al.: Finevq: Fine-grained user generated content video quality assessment. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3206–3217 (2025)

[13] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., et al.: Scaling rectified flow transformers for high-resolution image synthesis. In: Proceedings of the International Conference on Machine Learning (ICML) (2024)

[14] Fu, S., Bonnen, T., Guillory, D., Darrell, T.: Hidden in plain sight: Vlms overlook their visual representations. arXiv preprint arXiv:2506.08008 (2025)

[15] Garibi, D., Patashnik, O., Voynov, A., Averbuch-Elor, H., Cohen-Or, D.: Renoise: Real image inversion through iterative noising. In: European Conference on Computer Vision (ECCV). pp. 395–413 (2024)

[16] Goloviznina, V., Kotelnikov, E.: I've got the "answer"! interpretation of llms hidden states in question answering. In: International Conference on Applications of Natural Language to Information Systems (NLDB). pp. 106–120 (2024)

[17] Guan, H., Kozak, M., Robertson, E., Lee, Y., Yates, A.N., Delgado, A., et al.: Mfc datasets: Large-scale benchmark datasets for media forensic challenge evaluation. In: Proceedings of the IEEE Conference on Winter Applications of Computer Vision Workshops (WACVW). pp. 63–72 (2019)

[18] Guo, X., Liu, X., Ren, Z., Grosz, S., Masi, I., Liu, X.: Hierarchical fine-grained image forgery detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3155–3165 (2023)

[19] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)

[20] He, Y., Gan, B., Chen, S., Zhou, Y., Yin, G., Song, L., et al.: Forgerynet: A versatile benchmark for comprehensive forgery analysis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR). pp. 4358–4367 (2021)

[21] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., et al.: Lora: Low-rank adaptation of large language models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2022)

[22] Huang, Y., Huang, J., Liu, Y., Yan, M., Lv, J., Liu, J., et al.: Diffusion model-based image editing: A survey. IEEE transactions on pattern analysis and machine intelligence (TPAMI) **47**(6), 4409–4437 (2025)

[23] Huang, Z., Hu, J., Li, X., He, Y., Zhao, X., et al.: Sida: Social media image deepfake detection, localization and explanation with large multimodal model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 28831–28841 (2025)

[24] Hui, M., Yang, S., Zhao, B., Shi, Y., Wang, H., Wang, P., et al.: Hq-edit: A high-quality dataset for instruction-based image editing. arXiv preprint arXiv:2404.09990 (2024)

[25] Islam, A., Long, C., Basharat, A., Hoogs, A.: Doa-gan: Dual-order attentive generative adversarial network for image copy-move forgery detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4675–4684 (2020)

[26] Ju, X., Liu, X., Wang, X., Bian, Y., Shan, Y., Xu, Q.: Brushnet: A plug-and-play image inpainting model with decomposed dual-branch diffusion. In: European Conference on Computer Vision (ECCV). pp. 150–168 (2024)

[27] Ju, X., Zeng, A., Bian, Y., Liu, S., Xu, Q.: Pnp inversion: Boosting diffusion-based editing with 3 lines of code. In: Proceedings of the International Conference on Learning Representations (ICLR) (2024)

[28] Kim, J., Park, J., Song, Y., Kwak, N., Rhee, W.: Reflex: Text-guided editing of real images in rectified flow via mid-step feature extraction and attention adaptation. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 15939–15948 (2025)

[29] Kulikov, V., Kleiner, M., Huberman-Spiegelglas, I., Michaeli, T.: Flowedit: Inversion-free text-based editing using pre-trained flow models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 19721–19730 (2025)

[30] Labs, B.F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., et al.: Flux.1 kontext: Flow matching for in-context image generation and editing in latent space. arXiv preprint arXiv:2506.15742 (2025)

[31] Li, H., Huang, J.: Localization of deep inpainting using high-pass fully convolutional network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8300–8309 (2019)

[32] Li, J., Xu, Y., Lv, T., Cui, L., Zhang, C., Wei, F.: Dit: Self-supervised pre-training for document image transformer. In: Proceedings of the ACM international conference on multimedia (ACM MM). pp. 3530–3539 (2022)

[33] Li, S., Zeng, B., Feng, Y., Gao, S., Liu, X., Liu, J., et al.: Zone: Zero-shot instruction-guided local editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6254–6263 (2024)

[34] Li, S., Singh, H., Grover, A.: Instructany2pix: Flexible visual editing via multimodal instruction following. arXiv preprint arXiv:2312.06738 (2023)

[35] Li, Y., Liu, X., Wang, X., Lee, B.S., Wang, S., Rocha, A., et al.: Fakebench: Probing explainable fake image detection via large multimodal models. IEEE Transactions on Information Forensics and Security (TIFS) **20**, 8730–8745 (2025)

[36] Li, Y., Zhou, J.: Fast and effective image copy-move forgery detection via hierarchical feature point matching. IEEE Transactions on Information Forensics and Security (TIFS) **14**(5), 1307–1322 (2018)

[37] Liu, A., Feng, B., Wang, B., Wang, B., Liu, B., Zhao, C., et al.: Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. arXiv preprint arXiv:2405.04434 (2024)

[38] Liu, H., Li, C., Li, Y., Lee, Y.J.: Improved baselines with visual instruction tuning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 26296–26306 (2024)

[39] Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., et al.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In: European Conference on Computer Vision (ECCV). pp. 38–55 (2024)

[40] Liu, X., Liu, Y., Chen, J., Liu, X.: Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization. IEEE Transactions on Circuits and Systems for Video Technology (TCSVT) **32**(11), 7505–7517 (2022)

[41] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., et al.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9992–10002 (2021)

[42] Long, Z., Zheng, M., Feng, K., Zhang, X., Liu, H., Yang, H., et al.: Follow-your-shape: Shape-aware image editing via trajectory-guided region control. In: International Conference on Learning Representations (ICLR) (2026)

[43] Lu, H., Liu, W., Zhang, B., Wang, B., Dong, K., Liu, B., et al.: Deepseek-vl: Towards real-world vision-language understanding. arXiv preprint arXiv:2403.05525 (2024)

[44] Lu, S., Li, Y., Xia, Y., Hu, Y., Zhao, S., Ma, Y., et al.: Ovis2.5 technical report. arXiv:2508.11737 (2025)

[45] Lupaşcu, M., Stupariu, M.S.: Optimal transport for rectified flow image editing: Unifying inversion-based and direct methods. In: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 6764–6774. IEEE (2026)

[46] Ma, X., Du, B., Jiang, Z., Du, X., Hammadi, A.Y.A., Zhou, J.: Iml-vit: Benchmarking image manipulation localization by vision transformer. arXiv preprint arXiv:2307.14863 (2023)

[47] Mao, C., Zhang, J., Pan, Y., Jiang, Z., Han, Z., Liu, Y., et al.: Ace++: Instruction-based image creation and editing via context-aware content filling. In: IEEE/CVF International Conference on Computer Vision Workshops (ICCVW). pp. 1979–1987 (2025)

[48] Mao, Q., Chen, L., Gu, Y., Fang, Z., Shou, M.Z.: Mag-edit: Localized image editing in complex scenarios via mask-based attention-adjusted guidance. In: Proceedings of the ACM International Conference on Multimedia (ACM MM). pp. 6842–6850 (2024)

[49] Meta, A.: Llama 3.2: Revolutionizing edge ai and vision with open, customizable models, meta ai blog, 2024

[50] Nam, H., Kwon, G., Park, G.Y., Ye, J.C.: Contrastive denoising score for text-guided latent diffusion image editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9192–9201 (2024)

[51] Novozamsky, A., Mahdian, B., Saic, S.: Imd2020: A large-scale annotated dataset tailored for detecting manipulated images. IEEE Winter Applications of Computer Vision Workshops (WACVW) pp. 71–80 (2020)

[52] Ojha, U., Li, Y., Lee, Y.J.: Towards universal fake image detectors that generalize across generative models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 24480–24489 (2023)

[53] OpenAI: Chatgpt-4o: Advanced multimodal chat model (2025), https://openai.com/chatgpt

[54] OpenAI: Gpt-image-1 model (2025), https://platform.openai.com/docs/models/gpt-image-1

[55] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., et al.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. In: International Conference on Learning Representations (ICLR) (2024)

[56] Qi, M., Fu, R., Feng, P., Li, X., Cai, Y., Guo, Y., et al.: Cacr: Reinforcing temporal answer grounding in instructional video via candidate-aware causal reasoning. arXiv preprint arXiv:2606.08436 (2026)

[57] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence (TPAMI) **39**(6), 1137–1149 (2016)

[58] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10674–10685 (2022)

[59] Salloum, R., Ren, Y., Kuo, C.C.J.: Image splicing localization using a multi-task fully convolutional network (mfcn). Journal of Visual Communication and Image Representation (JVCIR) **51**, 201–209 (2018)

[60] Seedream, T., Chen, Y., Gao, Y., Gong, L., Guo, M., Guo, Q., et al.: Seedream 4.0: Toward next-generation multimodal image generation. arXiv preprint arXiv:2509.20427 (2025)

[61] Song, H., Huang, S., Dong, Y., Tu, W.W.: Robustness and generalizability of deepfake detection: A study with diffusion models. arXiv preprint arXiv:2309.02218 (2023)

[62] Su, L., Ma, X., Zhu, X., Niu, C., Lei, Z., Zhou, J.Z.: Can we get rid of handcrafted feature extractors? sparsevit: Nonsemantics-centered, parameter-efficient image manipulation localization through spare-coding transformer. In: Proceedings of the Conference on Association for the Advancement of Artificial Intelligence (AAAI). vol. 39, pp. 7024–7032 (2025)

[63] Sun, Z., Zhang, Z., Luo, Z., Sha, Z., Cong, T., Li, Z., et al.: Fragfake: A dataset for fine-grained detection of edited images with vision language models. arXiv preprint arXiv:2505.15644 (2025)

[64] Tan, C., Zhao, Y., Wei, S., Gu, G., Wei, Y.: Learning on gradients: Generalized artifacts representation for gan-generated images detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12105–12114 (2023)

[65] Wang, J., Pu, J., Qi, Z., Guo, J., Ma, Y., Huang, N., et al.: Taming rectified flow for inversion and editing. arXiv preprint arXiv:2411.04746 (2024)

[66] Wang, J., Duan, H., Wang, J., Jia, Z., Yang, W.Y., Zhu, X., et al.: Dfbench: Benchmarking deepfake image detection capability of large multimodal models. In: Proceedings of the ACM International Conference on Multimedia (ACM MM). pp. 12666–12673 (2025)

[67] Wang, J., Lv, C., Li, X., Dong, S., Li, H., Yao, K., et al.: Forensics-bench: A comprehensive forgery detection benchmark suite for large vision language models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4233–4245 (2025)

[68] Wang, S.Y., Wang, O., Zhang, R., Owens, A., Efros, A.A.: Cnn-generated images are surprisingly easy to spot... for now. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8692–8701 (2020)

[69] Wang, W., Lv, Q., Yu, W., Hong, W., Qi, J., Wang, Y., et al.: Cogvlm: Visual expert for pretrained language models. In: Advances in Neural Information Processing Systems (NIPS). vol. 37, pp. 121475–121499 (2024)

[70] Wang, W., Gao, Z., Gu, L., Pu, H., Cui, L., Wei, X., et al.: Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. arXiv preprint arXiv:2508.18265 (2025)

[71] Wen, B., Zhu, Y., Subramanian, R., Ng, T.T., Shen, X., Winkler, S.: Coverage—a novel database for copy-move forgery detection. In: Proceedings of IEEE International Conference on Image Processing (ICIP). pp. 161–165 (2016)

[72] Wen, Q., Li, C.G.: Rethinking decoders for transformer-based semantic segmentation: A compression perspective. Advances in Neural Information Processing Systems (NIPS) **37**, 49806–49833 (2024)

[73] Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., et al.: Qwen-image technical report. arXiv preprint arXiv:2508.02324 (2025)

[74] Wu, C., Zheng, P., Yan, R., Xiao, S., Luo, X., Wang, Y., et al.: Omnigen2: Exploration to advanced multimodal generation. arXiv preprint arXiv:2506.18871 (2025)

[75] Xu, S., Huang, Y., Pan, J., Ma, Z., Chai, J.: Inversion-free image editing with natural language. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9452–9461 (2024)

[76] Xu, Z., Zhang, X., Li, R., Tang, Z., Huang, Q., Zhang, J.: Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2025)

[77] Xu, Z., Duan, H., Liu, B., Ma, G., Wang, J., Yang, L., et al.: Lmm4edit: Benchmarking and evaluating multimodal image editing with lmms. In: Proceedings of the ACM International Conference on Multimedia (ACM MM). pp. 6908–6917 (2025)

[78] Xu, Z., Duan, H., Ma, G., Yang, L., Wang, J., Wu, Q., et al.: Harmonyiqa: Pioneering benchmark and model for image harmonization quality assessment. In: IEEE International Conference on Multimedia and Expo (ICME) (2025)

[79] Yan, S., Li, O., Cai, J., Hao, Y., Jiang, X., Hu, Y., et al.: A sanity check for ai-generated image detection. In: International Conference on Learning Representations (ICLR) (2025)

[80] Yan, Z., Yao, T., Chen, S., Zhao, Y., Fu, X., Zhu, J., et al.: Df40: Toward next-generation deepfake detection. In: Advances in Neural Information Processing Systems (NIPS). vol. 37, pp. 29387–29434 (2024)

[81] Yang, D., Liu, J., Li, J.: Balancing privacy, utility, and safety in llm alignment through preference optimization. arXiv preprint arXiv:2608.30141 (2026)

[82] Yang, Z., Zhou, Y., Chen, H., Zhang, H., Zhao, D., Wei, B., Xu, Y.: Unipet: a universal network for high-quality pet image denoising across varied dose reduction factors. Medical Image Analysis p. 104059 (2026)

[83] Yao, Y., Yu, T., Zhang, A., Wang, C., Cui, J., Zhu, H., et al.: Minicpm-v: A gpt-4v level mllm on your phone. arXiv preprint arXiv:2408.01800 (2024)

[84] Ye, J., Xu, H., Liu, H., Hu, A., Yan, M., Qian, Q., et al.: mplug-owl3: Towards long image-sequence understanding in multimodal large language models. In: Proceedings of the International Conference on Learning Representations (ICLR) (2024)

[85] Yu, Z., Ni, J., Lin, Y., Deng, H., Li, B.: 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)

[86] Zhang, K., Mo, L., Chen, W., Sun, H., Su, Y.: Magicbrush: A manually annotated dataset for instruction-guided image editing. In: Advances in Neural Information Processing Systems (NIPS). pp. 31428–31449 (2023)

[87] Zheng, H., Chang, C., Li, K., Huang, J., Yang, Z., Liu, T.: Fingpt-agent: An advanced framework for multimodal research report generation with task-adaptive optimization and hierarchical attention. In: International Conference on Neural Networks, Information and Communication Engineering (NNICE). pp. 1597–1601 (2025)

[88] Zhu, J., Wang, W., Chen, Z., Liu, Z., Ye, S., Gu, L., et al.: Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. arXiv preprint arXiv:2504.10479 (2025)

[89] Zhu, X., Qian, Y., Zhao, X., Sun, B., Sun, Y.: A deep learning approach to patch-based image inpainting forensics. Signal Processing: Image Communication (SPIC) **67**, 90–99 (2018)

[90] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)

[91] Zhu, X., Ma, X., Su, L., Jiang, Z., Du, B., Wang, X., et al.: Mesoscopic insights: orchestrating multi-scale & hybrid architecture for image manipulation localization. In: Proceedings of the Conference on Association for the Advancement of Artificial Intelligence (AAAI). vol. 39, pp. 11022–11030 (2025)

[92] Zhuang, J., Zeng, Y., Liu, W., Yuan, C., Chen, K.: A task is worth one word: Learning with task prompts for high-quality versatile image inpainting. In: European Conference on Computer Vision (ECCV). pp. 195–211 (2024)

Supplementary Material

8 Overview

This supplementary material provides additional information on the data collection, methodology, experiments, and results discussed in the main paper. Section **??** presents the related work. Section 9 describes the image collection process, including the 12 distinct tasks for prompt and 25 image manipulation models for generation. Section 10 elaborates on the manual annotation process, including the annotation standards, interface design, and management strategy. Section 5 provides an in-depth analysis of the ManipBench dataset. Section 6 details the hyper-parameters and loss functions used in training the ManipShield model. Finally, Section 7 presents additional results and comparisons between models.

9 Image collection

9.1 Manipulation tasks define

In this study, we conducted a comprehensive investigation into various categories of image modification, which we systematically divided into three main dimensions: semantic, visual, and textual. Across these dimensions, a total of 12 distinct manipulation types were defined to capture diverse editing behaviors at different levels of abstraction.

Within the semantic dimension, modifications primarily involve changes to image content and meaning, including addition, removal, replacement, and action change, which reflect variations in object presence, relationships, and contextual semantics.

The visual dimension focuses on perceptual and appearance-level edits such as background, color, material, expression, and resize, encompassing adjustments to aesthetic or structural aspects of the image.

Finally, the textual dimension addresses manipulations related to embedded text within images, encompassing scene text, handwritten text, and scanning text, which involve recognition, alteration, or regeneration of text in different modalities. This structured categorization provides a fine-grained framework for analyzing and benchmarking image-editing models, enabling a deeper understanding of model performance across diverse manipulation types.

- **Addition**: This task involves inserting new objects or regions into an existing image. Detecting such modifications often relies on identifying inconsistent object boundaries, lighting disparities, or unnatural blending between the newly added region and its surrounding context.
- **Removal**: This task removes existing objects or regions from an image. Detection methods typically focus on recognizing unnatural background reconstruction patterns, texture repetitions, or inpainting artifacts in the removed area.
- **Replacement**: This task substitutes one object or region with another. Identifying replacements often involves detecting semantic mismatches between the object and its environment, such as inconsistent shading, geometry, or contextual incongruence.
- **Action Change**: This task alters the pose or activity of a subject (e.g., changing a standing person to a sitting one). Detection commonly depends on analyzing spatial or anatomical inconsistencies, unnatural body part alignments, or motion artifacts in regions of change.
- **Background**: This task replaces or edits the scene background while preserving the main subjects. Detection methods can utilize boundary consistency checks, depth discontinuities, or background–foreground semantic mismatches to localize the modification.
- **Color**: This task changes the color properties of objects or regions. Such modifications can be detected through abnormal color histograms, inconsistent illumination cues, or deviations from material-specific color distributions.
- **Material**: This task modifies an object's surface material (e.g., wood to metal). Detection focuses on identifying inconsistencies in reflectance, glossiness, or fine texture gradients that contradict the expected material properties.

Table 10: An overview of AI-editing models in our ManipBench

Models	Time	Prompt Type	Method	URL
IP2P [5]	2023.04	Instruction	SD1.4 [58]	https://github.com/timothybrooks/instruct-pix2pix
CDS [50]	2023.11	Description	SD1.4 [58]	https://github.com/HyelinNAM/ContrastiveDenoisingScore
Magicbrush [86]	2023.06	Instruction	SD1.4	https://github.com/OSU-NLP-Group/MagicBrush
PnP [27]	2023.10	Description	SD1.5 [58]	https://github.com/MichalGeyer/plug-and-play
Any2Pix [34]	2023.12	Instruction	SDXL [55]	https://github.com/jacklishufan/InstructAny2Pix
InfEdit [75]	2023.12	Description	SD1.4 [58]	https://github.com/sled-group/InfEdit
MAG [48]	2023.12	Description	SD1.4 [58]	https://github.com/HelenMao/MAG-Edit
ZONE [33]	2023.12	Instruction	SD1.5 [58]	https://github.com/lsl001006/ZONE
PowerPaint [92]	2023.12	Description	SD1.5 [58]	https://github.com/open-mmlab/PowerPaint
ReNoise [15]	2024.03	Description	SDXL [55]	https://github.com/garibida/ReNoise-Inversion
BrushNet [26]	2024.03	Description	SD1.5 [58]	https://github.com/TencentARC/BrushNet
HQEdit [24]	2024.04	Instruction	SD1.4 [58]	https://github.com/UCSC-VLAA/HQ-Edit
RFSE [65]	2024.11	Description	FLUX [13]	https://github.com/wangjiangshan0725/RF-Solver-Edit
FlowEdit (SD3) [29]	2024.12	Description	SD3 [58]	https://github.com/fallenshock/FlowEdit
FlowEdit (FLUX) [29]	2024.12	Description	FLUX [13]	https://github.com/fallenshock/FlowEdit
ACE++ [47]	2025.01	Instruction	FLUX [13]	https://github.com/ali-vilab/ACE_plus
OmniGen2 [74]	2025.01	Instruction	DiT [32]	https://github.com/VectorSpaceLab/OmniGen2
Reflex [28]	2025.07	Description	FLUX [13]	https://github.com/wlaud1001/ReFlex
OT-Inversion [45]	2025.08	Description	FLUX [13]	https://github.com/marianlupascu/OT-Inversion
Follow-Your-Shape [42]	2025.08	Description	FLUX [13]	https://github.com/mayuelala/FollowYourShape
Qwen-Image-Edit [73]	2025.08	Instruction	DiT [32]	https://github.com/QwenLM/Qwen-Image
GPT-Image [54]	2025.04	Instruction	N/A	https://chatgpt.com/
FLUX-Kontext [30]	2025.06	Instruction	FLUX [13]	https://github.com/black-forest-labs/flux
NanoBanana [3]	2025.07	Instruction	N/A	https://ainanobanana.io/
SeedDream4 [60]	2025.09	Instruction	N/A	https://dreamina.capcut.com/

- **Expression**: This task alters facial expressions while retaining identity. Detection often involves subtle analysis of facial muscle movements, local geometric deformations, and texture discontinuities in high-frequency facial regions.
- **Resizing**: This task resizes objects or regions in the image. Detection can rely on geometric distortion analysis, perspective inconsistency, or mismatched scaling ratios among related scene elements.
- **Scene Text**: This task edits or replaces text in natural scenes. Detection generally targets inconsistencies in font alignment, lighting adaptation, or perspective transformation of the rendered text relative to its surface.
- **Handwritten Text**: This task modifies handwritten content. Such edits can be detected through irregularities in stroke continuity, pen pressure distribution, or inconsistent texture blending between the handwriting and background.
- **Scanning Text**: This task edits text in scanned or printed documents. Detection methods focus on recognizing pixel-level noise patterns, font irregularities, misalignments, or scanning artifacts indicative of post-processing.

9.2 Details of manipulation methods

- **IP2P [5]** is a conditional diffusion model designed for image editing guided by human-written instructions. It is trained on a large synthetic dataset generated by combining GPT-3 and Stable Diffusion, enabling the model to generalize to real images and unseen prompts without per-example fine-tuning or inversion. By conditioning on both the input image and the editing instruction, IP2P enables fast, high-quality edits in seconds while preserving the original structure.
- **CDS [50]** introduces the Contrastive Denoising Score for text-guided latent diffusion image editing. Unlike Delta Denoising Score, CDS incorporates contrastive learning over intermediate self-attention features of latent diffusion models, preserving structural elements of the source image and enabling zero-shot, structure-consistent image-to-image translation.
- **MagicBrush [86]** is the first large-scale, manually annotated dataset for instruction-guided real image editing. It provides diverse editing pairs under single-turn, multi-turn, mask-provided, and mask-free conditions. Fine-tuning models such as IP2P on MagicBrush improves human-rated edit quality and realism.
- **PnP [27]** is a description-based image editing system built upon Stable Diffusion 1.5, enabling flexible global image transformation from descriptive textual inputs. It maintains high realism and structural consistency across varied global domains.

- **Any2Pix [34]** is a multi-modal instruction-following image editing framework supporting text, image, and audio inputs. It employs a unified multi-modal encoder, a diffusion decoder, and an LLM-based instruction interpreter to execute complex edits with high fidelity across global domains.
- **InfEdit [75]** is an inversion-free, text-guided diffusion editing method that overcomes the inefficiency of inversion-based pipelines. Using a Denoising Diffusion Consistent Model and unified attention control, it enables fast, faithful, and structure-preserving global edits without explicit inversion.
- **MAG [48]** is a diffusion-based inpainting model tailored for local edits guided by text descriptions. It ensures coherent blending between masked and unmasked regions, maintaining global context and high-quality restoration.
- **ZONE[33]** is a zero-shot, instruction-guided local editing framework that translates textual instructions into editable regions. Through segmentation-based region extraction and FFT-based edge smoothing, it achieves seamless, localized edits while preserving non-edited content.
- **PowerPaint[92]** is a diffusion-based inpainting system emphasizing semantic consistency and fine-grained texture recovery. It performs text-driven completion for masked areas, maintaining seamless integration within the global image structure.
- **ReNoise [15]** is a description-driven global editing framework that improves robustness to noisy inputs through refined noise modeling and enhanced denoising strategies, achieving high-fidelity, structure-consistent edits under degraded conditions.
- **BrushNet [26]** is a diffusion-based inpainting model with brush-like latent operations for masked region editing. It introduces hierarchical attention and spatially adaptive blending to maintain smooth transitions between edited and unedited areas.
- **HQEdit [24]** is an instruction-guided global editing model optimized for high-quality (HQ) outputs. It features enhanced instruction parsing and guided sampling strategies to produce professional-grade, photorealistic edits with strong structural fidelity.
- **RFSE [65]** is a description-based editing framework leveraging the FLUX backbone. By exploiting its dynamic latent representation, RFSE enables robust global image transformations with high controllability and semantic preservation.
- **FlowEdit (SD3)[29]** is a flow-based description-driven editing model using Stable Diffusion 3. It performs global edits through latent flow interpolation and adaptive guidance, maintaining structural coherence and semantic alignment.
- **FlowEdit (FLUX) [13]** is a flow-based description-driven editing model using FLUX.1, combining flow-based latent editing with FLUX's expressive diffusion space for precise, high-fidelity global edits.
- **ACE++ [47]** is an instruction-guided editing framework built on the FLUX backbone. It integrates a refined instruction encoder and fast latent adaptation mechanism to achieve real-time, high-quality global edits with strong semantic consistency.
- **OmniGen2 [74]** is an instruction-guided image editing model using a Diffusion Transformer (DiT) backbone. It unifies semantic reasoning and generation, supporting complex global edits driven by natural language instructions.
- **Reflex [28]** is a description-based global editing model that captures stylistic or emotional cues in textual prompts. By conditioning latent style embeddings on descriptions, it produces coherent mood-driven transformations while preserving content fidelity.
- **OT-Inversion [45]** introduces an optimal-transport-based inversion method for latent diffusion models. It computes latent flows between source and target semantics, enabling faithful reconstruction and description-guided global editing.
- **Follow-Your-Shape [42]** is a shape-conditional description-based editing model that integrates contour guidance with text-driven FLUX diffusion. It ensures shape alignment and structure preservation in globally consistent edits.
- **Qwen-Image-Edit [73]** is an instruction-based image editing system aligned with the Qwen multi-modal model family. Built on a DiT backbone, it interprets high-level natural language instructions for semantically consistent global edits.

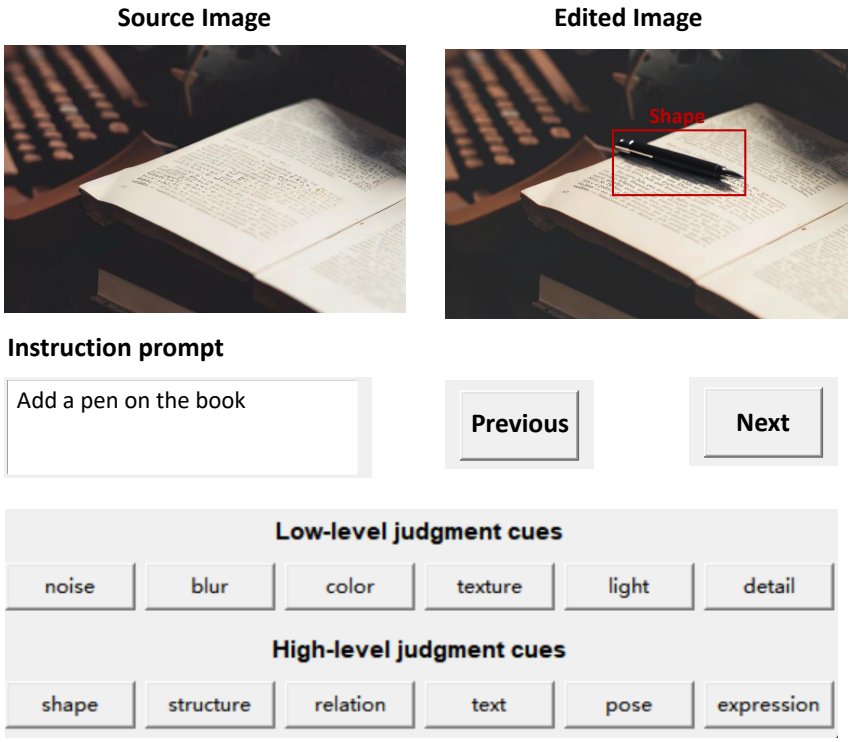


Figure 6: An example of the annotation interface.

- **GPT-Image [54]** is an instruction-based image editing interface that connects a large language model with a diffusion editing pipeline. It translates user intentions into image transformations, abstracting backbone specifics while ensuring global semantic coherence.
- **FLUX-Kontext [30]** is a context-aware instruction-guided editing model supporting multi-turn interactions. It tracks editing history and adapts latent states within the FLUX framework to maintain consistency across iterative global edits.
- **NanoBanana [8]** is a lightweight instruction-based diffusion model optimized for speed and interactivity. It performs fast global edits guided by simple textual commands while maintaining structural coherence.
- **SeedDream4 [60]** is an instruction-guided global editing model emphasizing seed-based control for reproducible edits. By coupling user instructions with seed conditioning, it achieves semantically faithful and consistent transformations across runs.

10 Details of manual annotation process

10.1 Localization annotation

To ensure an efficient manual annotation, we developed a custom annotation interface, as illustrated in Figure 6. The manual evaluation platform was implemented using the Python Tkinter package, featuring an intuitive and lightweight design to facilitate large-scale manipulation localization and cue labeling.

During the annotation process, human evaluators are presented with paired real and manipulated images and are instructed to draw bounding boxes around all perceptual semantically modified regions in the manipulated image. Here, semantic changes also include alterations in texture, color, or other visual attributes that differ from the original identity, even if the object category remains the same. For example, if a cat in the edited image is clearly not the same as the cat in the original image, it should still be considered a semantic change, even though it remains a cat. We emphasize perceptual changes because modifications that do not produce perceptible semantic differences are unlikely to be significant and generally do not pose safety or functional risks. The bounding box should encompass the entire semantic object and closely fit its boundaries. The overlap of different bounding boxes are allowed. For each annotated region, annotators also specify the judgment cues

that can help identify the manipulation. These cues are categorized into two main groups: high-level cues and low-level cues.

High-level judgment cues capture semantic and structural information of the scene, including object geometry, spatial relations, and human expressions. They reflect how AIGC models understand and reconstruct global semantics, and inconsistencies in these cues often reveal a lack of real-world reasoning or contextual coherence in edited images. Specifically, the following aspects represent typical high-level cues affected by AIGC editing:

- **Shape** cues describe the overall contour and geometry of objects in an image. AIGC editing may unintentionally distort object proportions—such as making limbs longer or altering the curvature of structures—leading to subtle yet detectable inconsistencies in global shape.
- **Structure** cues capture the spatial arrangement and connectivity among image components. Manipulations often disrupt these relationships, e.g., misaligned building elements or inconsistent object layouts caused by incomplete structural understanding of generative models.
- **Relation** cues focus on semantic and spatial relations among multiple objects. AIGC models sometimes generate objects that violate real-world logic—like inconsistent reflections or lighting directions—revealing manipulation through relational incoherence.
- **Text** cues are often difficult for generative models to render accurately. AIGC-edited images may show misspelled or distorted characters, irregular font spacing, or unnatural alignment, which serve as reliable traces of synthetic content.
- **Pose** cues reflect the geometric configuration of human or animal bodies. Manipulated or generated subjects may exhibit unnatural joint angles or inconsistent body alignment, as AIGC models occasionally fail to preserve plausible 3D pose relationships.
- **Expression** cues relate to fine-grained facial or emotional details. Editing may alter expressions inconsistently across regions—such as mismatched mouth and eye emotions—making emotional coherence an important signal for manipulation detection.

Low-level judgment cues focus on pixel-level visual statistics such as color, texture, noise, and lighting. They describe the appearance consistency between manipulated and authentic regions, where AIGC-generated content often exhibits deviations in fine-grained details or photometric properties. The following cues illustrate common low-level inconsistencies in AIGC-edited images:

- **Texture** cues describe surface granularity and local visual patterns. AIGC-edited regions often show over-smoothed or repetitive textures due to limited high-frequency detail synthesis, distinguishing them from authentic image regions.
- **Blur** cues reflect local focus and motion consistency. Manipulated content may exhibit inconsistent blur levels compared with surrounding areas, especially when compositing objects captured under different depth-of-field or motion conditions.
- **Noise** cues reveal sensor-level statistics that are hard for generative models to reproduce. Inconsistencies in noise distribution—such as missing camera noise or abnormal color noise patterns—can expose synthetic or edited areas.
- **Light** cues indicate illumination direction, color, and intensity. AIGC-based edits frequently fail to align lighting conditions between inserted and background regions, producing mismatched shadows or specular reflections.
- **Detail** cues correspond to fine local structures such as hair, texture edges, or reflections. AIGC editing may suppress or exaggerate these details, making local realism inconsistent with the overall image fidelity.
- **Color** cues capture chromatic harmony and tone consistency. Generative editing might produce unrealistic saturation, hue shifts, or mismatched white balance, especially when blending objects from different scenes.

In practical annotation, annotators can also select "other" to mark cues they consider not included in the predefined categories. However, upon review by expert groups, these annotations were found to also fit into the categories above.

All candidate annotators are required to undergo a 30-minute training session to become familiar with the annotation task and the marking rules. To evaluate their annotation quality, we have constructed a small image set consisting of 50 manipulated images that cover all types of judgment cues, each meticulously annotated by an expert group. Candidate annotators are required to complete annotations on this set, and only those whose annotations achieved an Intersection over Union (IoU) score above 0.7 with the expert annotations are qualified to participate in the main study. Based on this process, a total of 23 annotators were ultimately employed.

All annotations are conducted using a custom Python-based graphical interface displayed on a professionally calibrated LED monitor with a native resolution of 3840×2160. The monitor is color-calibrated to ensure consistent brightness, contrast, and color reproduction across sessions. Each image is presented in a randomized order to prevent potential ordering bias or learning effects during the annotation process. The annotation process is conducted under standardized laboratory conditions, with ambient illumination maintained at approximately 20 lux to minimize glare and reflections. Annotators are seated at a distance of approximately 2 feet from the display to ensure consistent visual perception and comfortable viewing. To mitigate visual fatigue and ensure annotation consistency, the entire process is divided into 15 sessions, each lasting no longer than 30 minutes, with mandatory short breaks between sessions. The order of the image sets assigned to each session is also randomized across annotators to further minimize potential contextual influence.

To ensure that the perceptual change is consistent with most people, we adopt a three-stage annotation protocol that integrates independent labeling, cross-verification, and expert voting.

- **Independent labeling:** Each manipulated image is annotated by two annotators. Each annotator independently labels the manipulated regions and corresponding judgment cues without reference to others' annotations, ensuring unbiased initial results.

- **Cross-verification:** For samples with IoU below 0.9, conflicting judgment cues or different bounding box number between annotators, a re-annotation is performed after discussion between the two person to ensure consistency and data quality. For all other samples, the final bounding box is obtained by averaging the annotations from the two annotators.

- **Expert Voting** Finally, all annotations were reviewed by 5 experts, and any items receiving fewer than 4 votes were discarded to ensure alignment with human perception. In this process, 27% of the annotations were removed.

The independent labeling and cross-verification required 1,800 working hours, while expert voting took an additional 650 working hours. This rigorous protocol guarantees that the resulting annotations are reliable, consistent, and perceptually meaningful, forming a high-quality foundation for subsequent model training and evaluation.

10.2 Explanation annotation

For explanation annotation, annotators are presented with the manipulated image, the corresponding annotated bounding box, judgment cues, and the textual explanation generated by ChatGPT-4o [53]. Ideal explanations should mention all regions within the annotated bounding boxes, cover accurate judgment cues, be linguistically fluent, logically coherent, and free of irrelevant information. Annotators can either accept the provided text, rewrite or delete some words, or seek a regeneration of the LLM draft.

Each image is reviewed by at least two annotators independently. If both annotators approve the draft, it is retained as the final explanation. If one annotator revise the text, the revised explanation is subsequently reviewed by three additional annotators. This process continued iteratively until at least three annotators agreed on the final version, ensuring consensus and alignment with human judgment.

All candidate annotators were required to undergo a training session to familiar with the annotation guidelines, criteria for judgment cues, and standards for textual explanations. The annotation process is conducted under the same controlled laboratory conditions as the localization annotation to minimize environmental variability. In total, this annotation procedure required approximately 750 hours.

11 More data analysis of ManipBench

As shown in Figure 11 and 12, we visualize the distributions of brightness, contrast, colorfulness, and spatial information (SI) of images generated by different editing models and real images to assess their visual characteristics and consistency. It is noteworthy that some source images inherently exhibit high brightness or low colorfulness, as certain text-related images are originally black-and-white, which may partially influence the overall distribution of the generated results. Nevertheless, several consistent trends can still be observed across different models.

Compared with real images, models such as IP2P [5], CDS [50], and MagicBrush [86] tend to generate outputs with brightness levels slightly higher than those of other generation models but still below those of real images. This suggests that these models more closely approximate the illumination characteristics of natural scenes, producing visually balanced and realistic outputs. In contrast, Qwen-Image-Edit [73] and several closed-source models yield smoother and more uniformly distributed brightness, reflecting enhanced tonal and chromatic consistency, albeit with reduced variability.

In terms of contrast, closed-source models generally achieve higher and more stable contrast, likely due to more advanced feature normalization or enhancement mechanisms. However, while excessive contrast can exaggerate textures and boundaries, moderate contrast contributes to more natural and perceptually pleasing results.

Regarding colorfulness, although some input images are intrinsically unsaturated, all models tend to produce more saturated outputs than real images. Open-source models such as PowerPaint [92], ReNoise [15], and InfEdit [75] exhibit larger variance, indicating less stable color calibration. In contrast, closed-source models generate higher yet smoother and more perceptually consistent colorfulness, suggesting better chromatic adjustment and adaptation to diverse input styles. Notably, the overall increase in colorfulness across generated images may serve as a potential statistical cue for distinguishing real from manipulated images.

Finally, spatial information (SI), which reflects the richness of structural and textural details, further differentiates model behaviors. Models such as ReNoise and InfEdit often produce smoother outputs with relatively low SI, whereas advanced approaches like Qwen-Image-Edit [73], FLUX-Kontext [30], and SeedDream4 [60] exhibit consistently higher SI values—sometimes even exceeding those of real images, indicating stronger preservation of fine details and local texture fidelity.

12 Details of training process

12.1 Details of hyperparameters

This configuration includes several key hyperparameters that substantially influence the model's performance during both training and evaluation. The model is trained in two stages. In the first stage, the vision encoder is trained using Contrastive LoRA Fine-tuning to enhance its ability to capture manipulation-related visual semantics while maintaining parameter efficiency. In the second stage, the vision encoder is frozen, and the large language model is fine-tuned together with the specified decoders and LoRA modules to align visual features with textual understanding and reasoning. The hyperparameters for both stages are summarized as follows:

- **Learning Rate (5e-6/1e-5):** The learning rate determines the step size at each iteration while moving toward the minimum of the loss function. A smaller learning rate allows the model to converge more slowly but with higher precision, whereas a larger learning rate can speed up convergence at the risk of overshooting the optimal solution. In this work, the learning rate is set to 1e-5, which is relatively small and suitable for fine-tuning pre-trained models.

- **Batch Size (32/16):** This hyperparameter defines how many samples are processed together in one forward and backward pass. A batch size of 1 means that each sample is processed individually, which trades off speed for lower memory usage and ensures stable training on GPUs with limited memory capacity.

- **Warm-up Ratio (0.05):** This parameter specifies the proportion of the total training steps during which the learning rate linearly increases from 0 to the initial value. It helps stabilize the optimization process, particularly during the early stages of training.

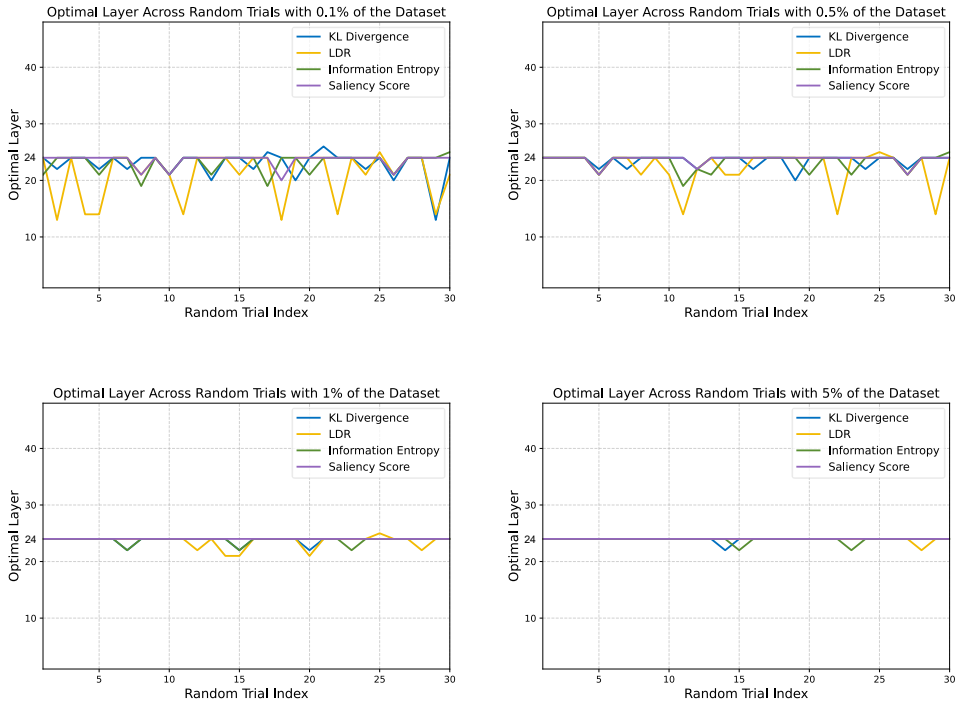


Figure 7: Optimal layer variation across random trials for KL divergence, LDR, information entropy, and saliency score under different sample scales.

- **Weight Decay (0.1):** Weight decay is a regularization technique that prevents overfitting by penalizing large parameter magnitudes. A value of 0.1 indicates that model weights are reduced by 10% during each update step, improving generalization on unseen data.

- **LoRA Rank (16):** The rank determines the dimension of the low-rank adaptation matrices in the LoRA module. A higher rank allows greater adaptation flexibility, while a lower rank reduces parameter overhead. A rank of 16 offers a balanced tradeoff between expressiveness and efficiency.

- **LoRA Alpha (32):** This scaling factor controls the update strength of the LoRA parameters relative to the base model weights. Setting LoRA alpha to 32 amplifies the contribution of the low-rank updates, enabling more effective fine-tuning without increasing the total number of trainable parameters significantly.

12.2 Details of layer discrimination selection

We combine KL divergence, LDR, and information entropy to select the optimal layer. Figure 7 shows that the optimal layer determined by individual metrics fluctuates when the sample size is low. With 0.1% and 0.5% of the dataset, both the individual metrics and the combined saliency score vary considerably, indicating that such few samples are insufficient for reliable layer selection. Using 1% of the dataset (around 4K samples), the saliency score stabilizes, although the individual metrics still fluctuate. With 5% of the dataset, all metrics are nearly stable, but at a higher computational cost. Therefore, combining the three metrics into a saliency score allows reliable identification of the optimal layer with fewer samples.

12.3 Details of loss functions

The training of our model is organized into two sequential stages to gradually build discriminative and task-aware representations. In the first stage, the focus is on learning robust and generalizable visual representations that can effectively distinguish real images from manipulated ones. In the second stage, the model is fine-tuned for multi-task objectives, including binary detection, bounding box regression, and judgment cue classification, thereby enhancing both global and local reasoning capabilities.

Stage 1: contrastive learning In the first training stage, a contrastive loss is employed to enforce the model to separate real and manipulated samples in the feature space. Let the feature embeddings of a positive pair (i, j) be denoted as f_i and f_j, where each pair consists of a real image and its manipulated counterpart. The embeddings of all other images within the same batch are denoted as f_k $(k \neq i, j)$ and serve as negative samples. The contrastive loss is formulated as:

$$\mathcal{L}_{\text{contrastive}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp(\text{sim}(f_i, f_j)/\tau)}{\sum_{k\neq i,j}\exp(\text{sim}(f_i, f_k)/\tau)}, \tag{9}$$

where $\text{sim}(\cdot)$ denotes the cosine similarity function, τ is a temperature hyperparameter controlling the concentration level of the distribution, and N is the batch size. All feature embeddings are ℓ_2-normalized before computing similarity to ensure numerical stability. This stage aims to cluster semantically similar features (i.e., real–manipulated pairs) closer in the latent space while pushing dissimilar pairs apart, encouraging the model to capture manipulation-sensitive patterns that are invariant to irrelevant appearance factors such as lighting or texture.

Stage 2: Multi-Task fine-Tuning After the representation space is pre-trained with contrastive supervision, the model is jointly optimized for three downstream objectives: (1) to determine whether an image is real or manipulated (binary detection), (2) to localize manipulated regions through bounding box regression, and (3) to identify the manipulation-related judgment cue category (e.g., light, color, texture).

The first task is formulated as a binary classification problem optimized using the Focal loss, which focuses training on hard or misclassified examples and mitigates class imbalance between real and manipulated samples. The second task is a regression problem optimized using the mean squared error (MSE) loss to ensure accurate localization of manipulated areas. The third task employs a standard cross-entropy loss to supervise the judgment cue classification. The overall multi-task loss function is defined as:

$$\mathcal{L}_{total} = \lambda_1\mathcal{L}_{binary} + \lambda_2\mathcal{L}_{bbox} + \lambda_3\mathcal{L}_{cue}, \tag{10}$$

where $\mathcal{L}_{binary}$, $\mathcal{L}_{bbox}$, and $\mathcal{L}_{cue}$ denote the binary judgment loss, bounding box regression loss, and cue classification loss, respectively. The weighting factors λ_1, λ_2, and λ_3 are empirically determined to balance the gradient magnitudes of different objectives.

The detailed formulations of these losses are as follows:

$$\mathcal{L}_{binary} = -\alpha(1-p_t)^{\gamma}\log(p_t), \tag{11}$$

$$\mathcal{L}_{bbox} = \frac{1}{N}\sum_{i=1}^{N}(B_i^{pred} - B_i^{gt})^2, \tag{12}$$

$$\mathcal{L}_{cue} = -\frac{1}{C}\sum_{c=1}^{C} y_c\log(p_c), \tag{13}$$

where p_t represents the predicted probability of the ground-truth class in binary detection, and α and γ are the balance and focusing parameters of the Focal loss. B_i^{pred} and B_i^{gt} denote the predicted and ground-truth bounding box coordinates, and N is the number of bounding boxes. For cue classification, C is the total number of cue categories, y_c is the one-hot ground-truth label, and p_c is the predicted probability of class c.

This multi-task optimization framework allows the model to learn both coarse-level manipulation detection and fine-grained reasoning cues simultaneously. The binary branch enhances global discrimination capability, the regression branch enforces spatial precision, and the cue classification branch strengthens interpretability by associating detection results with human-understandable manipulation causes.

13 Model comparison details

13.1 Details of AI-generated images detection methods

- **Univ [52]** introduces a unified representation learning framework for detecting AI-generated images across diverse generative models. It employs multi-scale frequency–spatial fusion and self-supervised domain adaptation, enabling strong generalization to unseen generation architectures (e.g., diffusion, GAN, DiT). Univ achieves robust detection even under post-processing operations such as compression and resizing, though its high complexity can limit real-time deployment.
- **AIDE [79]** proposes a cross-architecture forensic model based on hybrid vision transformers. By leveraging large-scale datasets of real and synthetic samples, AIDE learns source-invariant traces via contrastive fine-tuning. It provides state-of-the-art detection accuracy for diffusion-based fakes, while maintaining interpretability through attention-based saliency maps. However, its dependency on large-scale pretraining may hinder adaptation to niche domains.
- **CNNSpot [68]** is a CNN-based forensic detector that focuses on low-level texture and frequency inconsistencies. It is lightweight and efficient, making it suitable for fast fake image screening. It introduces frequency-aware convolution kernels and adaptive pooling, significantly improving robustness to image compression. Despite high sensitivity to pixel artifacts, CNNSpot remains vulnerable to adversarial perturbations and unseen generation pipelines.
- **Lagrad [64]** is a gradient-based forensic model designed to capture subtle distributional shifts between real and AI-generated images. It employs layer-wise gradient regularization and feature contrast alignment, enhancing robustness across generative families. It integrates multi-domain calibration, providing better resilience under cross-model evaluation. Nonetheless, gradient dependence can amplify noise in low-quality or heavily compressed images.

13.2 Details of image manipulation detection methods

- **HiFiNet [18]** introduces a high-fidelity manipulation detection network designed to identify subtle tampering traces in high-resolution images. It leverages hierarchical feature fusion between shallow forensic features and deep semantic representations, enabling accurate localization of manipulated regions. HiFiNet excels at detecting inpainting, splicing, and copy–move edits while maintaining low false-positive rates. However, its high-resolution backbone makes inference computationally expensive for large-scale applications.
- **FakeShield [76]** proposes a lightweight, transformer-enhanced forensic framework that integrates spatial and frequency cues to detect and localize image manipulations. It adopts a dual-branch architecture combining RGB-domain residual analysis and frequency-domain correlation learning, enabling effective generalization to various editing operations. While robust and efficient, FakeShield's performance may degrade under severe image compression or adversarial noise.
- **MVSSNet [10]** extends the original MVSSNet by incorporating multi-scale attention and structure-aware supervision. It models tampering traces as multi-view correlations, combining pixel-level cues with contextual priors. It introduces an adaptive refinement head that enhances boundary detection of small edited regions. MVSSNet achieves state-of-the-art manipulation localization performance but requires extensive training data for optimal generalization.
- **PSCCNet [40]** is a transformer-based manipulation detector that maintains consistency between pixel-level noise patterns and semantic features. It incorporates a correlation-consistency loss to enforce coherence across spatial scales, improving accuracy in detecting subtle photo composites and illumination edits. Despite its precision, PSCCNet demands high computational cost and memory during training.

13.3 Details of multimodal large language models

- **LLaMA-3.2-Vision (11B) [49]** is Meta's open multimodal variant of the LLaMA-3.2 family, integrating a visual encoder with the language backbone for image-text reasoning. It supports high-resolution visual grounding, object reasoning, and instruction following. In the context of fake image identification, its large-scale pretraining on paired vision-language data enables robust detection of subtle editing cues, though its performance can degrade under domain shifts or adversarial manipulations.

Table 11: Comparison results on different manipulation category subset. ♠ standard CNN/Transformer baselines, ♡ image manipulation detection methods, ♣ AI-generated image detection methods, ♢ multimodal large language models. The fine-tuned results are marked with ∗. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Testing Subset	Addition		Removal		Replacement		Color Modification		Action Change		Background	
Model/Metric	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♡HifiNet [18]	53.88	35.86	49.30	40.46	52.52	42.05	53.30	44.17	55.87	56.68	49.89	35.76
♡FakeShield [76]	55.24	40.28	54.34	46.91	49.86	44.58	56.76	50.00	59.86	62.75	50.79	39.89
♣Univ [52]	54.40	39.33	50.56	44.41	46.22	42.34	54.80	48.37	57.98	61.17	49.89	38.95
♣AIDE [79]	57.97	42.47	56.72	48.93	54.06	47.44	58.11	51.48	62.44	64.91	56.46	43.53
♢LLama3.2-Vision (11B) [49]	51.42	36.39	48.37	41.90	42.53	39.31	54.08	46.52	55.28	58.42	48.63	36.93
♢LLaVA-1.6 (7B) [38]	19.47	29.98	24.18	37.86	24.73	38.03	26.97	40.43	40.60	56.76	19.87	31.78
♢MiniCPM-V2.6 (8B) [83]	41.48	34.80	37.36	40.05	34.51	38.99	44.75	44.83	47.02	57.14	35.35	34.34
♢mPLUG-Owl3 (7B) [84]	46.55	29.64	43.21	34.69	44.57	35.24	48.98	38.81	46.10	48.58	42.81	29.88
♢CogVLM (17B) [69]	80.93	28.24	76.49	29.96	74.59	28.35	74.64	29.84	63.76	31.90	81.01	29.96
♢Ovis2.5 (9B) [44]	44.42	38.57	41.03	44.22	39.13	43.43	48.10	49.14	59.86	66.28	43.25	39.95
♢DeepSeekVL2 (small) [37]	19.57	30.38	25.14	38.57	24.05	38.23	26.82	40.80	41.51	57.57	19.98	32.19
♢InternVL3 (8B) [88]	63.49	35.71	58.29	39.45	44.57	32.89	55.69	39.68	52.75	49.26	53.79	32.21
♢InternVL3.5 (8B) [70]	77.08	43.78	64.40	40.18	55.57	34.99	68.08	44.56	63.07	52.23	69.26	38.60
♢Qwen2.5-VL (8B) [2]	65.82	44.30	50.27	42.27	46.33	40.42	60.64	49.81	57.80	59.29	51.04	37.54
♢Qwen3-VL (8B) [1]	64.88	44.99	51.40	44.12	49.30	43.08	60.66	51.12	59.86	61.57	52.15	39.37
♢Gemini2.5-Pro [8]	67.61	41.59	63.03	45.45	49.30	37.80	61.56	46.22	58.22	55.28	58.05	37.29
♢ChatGPT-4o [53]	64.47	44.70	61.62	50.00	62.32	50.46	66.37	55.02	64.55	64.47	63.72	46.13
♠ResNet50∗ [19]	90.14	94.68	89.86	94.47	91.08	95.14	90.96	95.13	90.38	94.74	88.73	93.79
♠Swin-T∗ [41]	89.90	94.49	89.86	94.43	90.38	94.74	91.35	95.31	90.77	94.96	89.26	94.09
♡MVSSNet∗ [10]	89.54	94.27	90.03	94.58	87.24	92.91	92.12	95.78	86.54	92.54	85.68	91.95
♡PSCCNet∗ [40]	90.26	94.69	90.56	94.89	89.51	94.23	91.92	95.62	88.85	93.84	87.53	93.09
♡HifiNet∗ [18]	90.38	94.76	90.03	94.55	88.99	93.92	90.38	94.75	89.62	94.29	88.99	93.93
♡FakeShield∗ [76]	90.02	94.53	92.13	95.73	88.46	93.62	92.50	95.95	90.77	94.96	90.85	95.00
♣CNNSpot∗ [68]	89.78	94.40	88.46	93.63	90.56	94.84	84.23	91.07	86.54	92.47	89.79	94.40
♣Lagrad∗ [64]	90.26	94.67	91.26	95.25	91.96	**95.64**	**92.88**	**96.17**	91.92	95.62	**92.84**	**96.13**
♣Univ∗ [52]	91.47	95.36	91.61	95.45	90.73	94.94	91.92	95.62	90.38	94.74	91.51	95.39
♣AIDE∗ [79]	**92.55**	**95.97**	**93.36**	**96.42**	91.43	95.34	91.35	95.29	**92.69**	**96.05**	92.71	96.06
♢DeepSeekVL2 (small)∗ [37]	89.90	94.46	90.91	95.04	**92.78**	94.36	92.31	95.83	91.92	95.62	92.71	96.06
♢InternVL3 (8B)∗ [88]	90.50	94.81	91.08	95.14	92.60	95.27	90.77	94.96	89.23	94.07	89.12	94.01
♢Qwen3-VL (8B)∗ [1]	91.35	95.29	91.08	95.14	92.43	95.18	91.35	95.29	88.85	93.84	90.98	95.08
ManipShield (Ours)∗	**92.67**	**96.04**	**94.41**	**97.00**	**93.71**	**96.62**	**94.04**	**96.81**	**95.38**	**97.54**	**94.16**	**96.87**

Testing Subset	Material Change		Expression		Resizing		Scene Text		Handwritten Text		Scanned Text	
Model/Metric	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♡HifiNet [18]	54.58	63.90	62.09	67.96	52.11	52.01	78.25	76.16	85.14	76.88	68.16	65.78
♡FakeShield [76]	64.38	72.54	75.49	79.34	54.01	56.92	83.62	83.24	88.55	83.48	73.13	72.73
♣Univ [52]	62.09	70.85	75.82	79.21	50.21	54.44	85.59	84.68	89.96	84.94	73.88	72.87
♣AIDE [79]	68.63	75.51	75.82	80.00	56.75	59.08	81.36	81.77	81.12	75.90	70.90	71.67
♢LLama3.2-Vision (11B) [49]	60.13	68.84	73.31	76.75	48.35	52.19	85.87	84.31	90.41	84.83	71.05	69.72
♢LLaVA-1.6 (7B) [38]	54.66	70.69	57.23	71.88	35.80	52.15	66.48	73.75	89.43	86.29	45.26	60.18
♢MiniCPM-V2.6 (8B) [83]	57.88	70.16	60.13	71.30	44.03	53.10	73.13	76.05	81.21	76.24	62.04	66.38
♢mPLUG-Owl3 (7B) [84]	47.27	57.51	57.23	62.54	44.65	45.21	77.29	73.03	84.54	73.75	66.67	61.84
♢CogVLM (17B) [69]	49.52	32.03	51.13	32.74	66.46	31.22	58.45	33.04	70.84	33.18	63.50	33.04
♢Ovis2.5 (9B) [44]	63.02	74.95	71.70	79.63	48.56	57.91	83.38	85.15	82.19	79.08	56.69	65.90
♢DeepSeekVL2 (small) [37]	55.95	71.64	62.06	74.57	36.42	52.82	78.12	81.41	83.95	80.84	46.72	61.24
♢InternVL3 (8B) [88]	49.52	56.02	65.27	64.94	43.42	42.11	76.18	69.93	83.17	69.93	77.86	68.73
♢InternVL3.5 (8B) [70]	52.41	54.32	67.20	63.31	53.29	43.67	72.85	64.23	80.82	64.23	60.10	51.76
Qwen2.5-VL (8B) [2]	56.59	66.50	72.03	75.49	50.82	52.86	85.32	83.49	89.82	83.75	67.64	66.83
♢Qwen3-VL (8B) [1]	58.82	68.50	72.55	76.54	51.90	54.58	83.33	82.28	87.35	81.31	68.16	68.16
♢Gemini2.5-Pro [8]	55.56	61.80	68.63	69.62	50.84	48.57	78.53	74.32	84.74	74.32	80.10	73.33
♢ChatGPT-4o [53]	66.67	72.87	70.26	75.07	60.55	59.44	83.05	82.04	88.55	82.78	77.11	74.86
♠ResNet50∗ [19]	91.54	95.40	83.08	90.60	88.14	93.43	92.31	95.95	87.28	92.92	94.87	97.37
♠Swin-T∗ [41]	88.46	93.62	85.38	92.05	89.10	93.99	92.86	96.30	95.15	97.76	94.02	96.92
♡MVSSNet∗ [10]	89.23	94.07	88.46	93.83	85.90	92.09	92.86	96.30	93.20	96.43	90.17	94.74
♡PSCCNet∗ [40]	89.23	94.12	89.23	94.31	89.74	94.39	92.86	96.30	91.42	95.33	88.03	93.64
♡HifiNet∗ [18]	92.31	95.83	91.54	95.55	91.35	95.29	92.31	96.00	91.42	95.33	92.31	96.00
♡FakeShield∗ [76]	93.85	96.69	90.00	94.74	**92.95**	**96.19**	91.76	95.70	90.24	94.72	89.32	94.33
♣CNNSpot∗ [68]	92.31	95.87	80.00	88.79	90.71	94.92	86.26	92.58	34.91	48.84	51.28	66.27
♣Lagrad∗ [64]	**93.85**	**96.69**	87.69	93.39	90.38	94.74	92.31	95.83	93.79	96.66	90.17	94.76
♣Univ∗ [52]	90.00	94.51	82.31	90.21	90.38	94.74	74.73	84.87	92.90	96.17	85.47	92.06
♣AIDE∗ [79]	90.77	94.96	83.85	91.14	91.99	95.65	**92.86**	**96.30**	**95.86**	**97.80**	91.45	95.54
♢DeepSeekVL2 (small)∗ [37]	93.08	96.27	90.08	96.27	90.38	94.74	92.86	96.14	92.90	96.17	92.74	95.07
♢InternVL3 (8B)∗ [88]	93.08	96.27	**92.15**	**95.96**	90.06	94.55	92.15	95.96	91.42	95.33	**94.97**	**97.46**
♢Qwen3-VL (8B)∗ [1]	91.54	95.40	91.54	95.40	91.35	95.29	92.60	95.66	95.56	97.64	93.59	96.55
ManipShield (Ours)∗	**94.62**	**97.12**	**92.31**	**96.00**	**93.59**	**96.55**	**93.41**	**96.59**	**97.04**	**98.47**	**95.30**	**97.59**

- **LLaVA-1.6 (7B) [38]** extends the LLaMA-2 architecture with a CLIP-based visual encoder aligned via instruction tuning. It achieves strong multimodal understanding, especially for localized reasoning such as identifying inconsistent shadows or textures in edited images. However, its reliance on pre-encoded CLIP embeddings limits fine-grained detection of low-level manipulations.

- **MiniCPM-V2.6 (8B) [83]** is a lightweight multimodal model from OpenBMB, designed for efficient visual-text reasoning with strong low-latency inference. It balances compactness and multimodal fidelity, making it suitable for deployment in real-time image authenticity detection. While efficient, its reduced parameter count can hinder nuanced detection of subtle forgeries.

- **mPLUG-Owl3 (7B) [84]** developed by Alibaba's X-PLUG team, introduces unified vision-language alignment using adaptive visual adapters. It excels in complex cross-modal question answering and reasoning about compositional visual semantics, which benefits fake image detection where multimodal contradictions exist. However, it may struggle with fine-scale pixel-level inconsistencies due to its high-level reasoning focus.
- **CogVLM (17B) [69]** is a large multimodal transformer that integrates visual tokens into a unified autoregressive architecture. Its massive capacity allows rich contextual reasoning about editing traces (e.g., lighting mismatch, semantic incongruence). Despite superior understanding, inference cost is high, and over-reliance on text priors may reduce pixel-level sensitivity.
- **Ovis 2.5 (9B) [44]** is a vision-language model emphasizing efficient architecture and high-resolution image understanding. It employs hierarchical cross-attention to enhance visual grounding, offering balanced accuracy and speed in identifying tampered or composited regions. Yet, its mid-sized configuration can miss micro-level editing artifacts.
- **DeepSeek-VL2 [43]** builds on the DeepSeek LLM series with a multi-stage visual alignment strategy, incorporating both coarse and fine-grained perception. It demonstrates superior generalization across diverse visual domains, performing well in fake image detection involving semantic or stylistic inconsistencies. However, closed-source training data may introduce unknown bias.
- **InternVL3 / 3.5 [88, 70]** developed by OpenGVLab, advances unified vision-language representation with scalable vision encoders and deep fusion layers. It supports multi-image reasoning, beneficial for comparative fake detection (e.g., before-after editing analysis). Its performance is state-of-the-art, though model size imposes heavy GPU requirements.
- **Qwen 2.5-VL / 3-VL [2, 1]** is part of Alibaba's Qwen multimodal series integrating DiT-style vision backbones with instruction-tuned LLMs. It provides strong text-grounded reasoning and fine-grained scene understanding, crucial for identifying forged object insertions. While highly accurate, hallucination from language priors may reduce precision in subtle artifact detection.
- **Gemini 2.5-Pro [8]** is Google's closed multimodal foundation model with integrated visual reasoning, OCR, and image understanding capabilities. It shows excellent zero-shot detection of composited or generated content due to broad multimodal pretraining. However, its proprietary nature prevents fine-tuning for domain-specific forensic tasks.
- **ChatGPT-4o [53]** integrates high-resolution vision and language reasoning in a unified architecture, supporting both spatial and semantic analysis of manipulated images. Its strong visual understanding enables general detection of synthetic artifacts, though closed weights and lack of forensic specialization limit interpretability and reproducibility.

13.4 Details of the human evaluation procedure for explanations

To evaluate the quality of the generated explanation texts, we conducted a controlled human study. We firstly collect the 16K output explanations from 7 MLLMs and our ManipShield based on testing set ManipBench. For each sample, the evaluators were provided with the original image, the manipulated image, the corresponding editing instruction, and the model-generated explanation, including both the detected manipulated region and the textual reasoning. The evaluators were asked to assess whether the explanation accurately identified the manipulated area, whether the reasoning was logically consistent with the visual evidence, and whether the description was clear and informative.

We recruited five domain experts with prior experience in image analysis and visual content interpretation. All evaluations were performed independently to avoid mutual influence. Each explanation was rated on a 0–5 Likert scale, where higher scores indicate better explanation quality in terms of correctness, consistency, and clarity.

To mitigate individual scoring bias and differences in scoring tendencies among evaluators, we first standardized the scores for each evaluator independently. This step ensures that variations caused by personal scoring scales do not disproportionately affect the aggregated results. After standardization, we performed outlier removal to reduce the influence of accidental misjudgments or annotation noise. For each sample, scores that deviated by more than two standard deviations from the mean of the standardized ratings were regarded as outliers and discarded.

Finally, the retained scores were linearly rescaled to a 0–100 range for better interpretability and reporting consistency. The final performance of the model is reported as the mean opinion score

averaged across all evaluated samples. All samples were presented in randomized order during evaluation to prevent order effects. The whole evaluation cost 30 working hours.

13.5 More comparison results

In this section, we present additional comparison results and analyses to further demonstrate the performance of different models, as well as key insights into image manipulation detection and localization.

Table 11 demonstrates the detection performance of various models across different manipulation tasks. We observe that general detection methods struggle with regional manipulations, such as addition, removal, and replacement, without fine-tuning, likely due to their limited ability to capture subtle local cues. In contrast, these methods perform relatively well on expression changes, as previous training datasets contain abundant facial content. They also achieve higher accuracy on text-related subsets, probably because AIGC-based editing models often generate unrealistic text manipulations. After fine-tuning, performance improves across all categories, while our method consistently surpasses all baseline approaches.

Table 12 shows the detection performance of various models across different manipulation methods. We observe that manipulation models based on stable diffusion backbones are generally easier to detect than those based on FLUX or DiT backbones, likely because FLUX- and DiT-based models tend to generate more realistic images. However, some diffusion-based models are also challenging to detect, indicating that detection difficulty depends not only on the backbone but also on the characteristics of the individual model itself. This observation highlights the importance of including a diverse set of manipulation models when constructing evaluation datasets.

Table 13 presents the detection accuracy of various models when trained on a subset of manipulation models and evaluated on different subsets. The results indicate that all detection models experience a notable drop in performance when tested on subsets containing backbones unseen during training. This decrease is likely due to the diverse characteristics of images generated by different manipulation models, which makes generalization challenging. In contrast, when trained on the full ManipBench dataset, model performance significantly improves even on previously unseen models such as Flux-Kontext, NanoBanana, GPT-Image, and SeedDream4. These findings highlight the critical importance of including a wide variety of manipulation models in the training dataset to capture diverse visual features and enhance generalization. Furthermore, our model consistently demonstrates the strongest generalization ability among all compared detection methods. This superior performance can be attributed to its design, which not only extracts low-level visual features but also effectively captures high-level semantic inconsistencies introduced by manipulations, enabling it to detect subtle and diverse manipulations across unseen models.

14 Limitations

14.1 Cost analysis

A limitation of our approach lies in its relatively higher computational cost compared to traditional image manipulation detection methods. Since our model is built on a large language model backbone, it inevitably requires more GPU memory and incurs higher inference cost than lightweight CNN-based methods such as MVSSNet [10] and AIDE [79]. As shown in Table 14, these traditional approaches achieve significantly lower resource consumption due to their compact architectures. However, compared to other explanation-oriented methods like FakeShield [76], our model is more efficient in terms of computation. In particular, ManipShield reduces both MACs and inference time while maintaining strong performance due to the designed light weight decoders. Overall, while our method introduces higher cost than conventional detectors, it offers a favorable trade-off between interpretability, generalization ability, and computational efficiency among LLM-based and explanation-capable frameworks.

15 Potential social impact

15.1 Positive impacts

ManipBench and ManipShield can substantially strengthen the reliability and interpretability of image manipulation detection in the era of advanced generative models. By covering a wide range

of state-of-the-art editing techniques and manipulation categories, the proposed benchmark better reflects real-world synthetic image distributions, enabling more robust model development and evaluation. This can help mitigate the spread of AI-generated misinformation, support digital forensics and media verification in journalism, and enhance the content moderation capabilities of online platforms. Moreover, the inclusion of localized evidence and textual explanations encourages the development of more transparent and human-understandable detection systems, which can improve user trust and facilitate responsible deployment of AI in safety-critical visual media applications.

15.2 Negative impacts

At the same time, the release of ManipBench and ManipShield may introduce certain risks associated with dual-use scenarios. Although designed for detection, the benchmark itself reveals detailed manipulation categories and model behaviors, which could potentially be exploited by malicious actors to study weaknesses in current detection systems and create more convincing forged images that evade detection. There is also a risk that downstream users or platforms may place excessive trust in automated predictions, leading to over-reliance on imperfect models and false confidence in content authenticity. In addition, despite its scale, the benchmark may not fully capture all emerging manipulation techniques in the rapidly evolving generative landscape, which could result in biased or incomplete generalization when deployed in uncontrolled real-world environments.

15.3 Safeguards

To mitigate these risks, careful release and usage governance are necessary. The dataset and models should ideally be distributed under a research-only license with clear usage restrictions and ethical guidelines to discourage malicious applications. It is also important to accompany the release with transparent documentation of known limitations and failure cases, enabling users to better understand model uncertainty rather than treating outputs as absolute judgments. In high-risk deployment scenarios, the system should be used as a decision-support tool rather than an autonomous authority, ideally combined with human review and complementary provenance-based techniques such as watermarking or metadata verification. Continuous evaluation against adaptive and adversarial manipulation methods should also be encouraged to improve robustness over time and reduce potential exploitation.

Table 12: Comparison results on different manipulation method subset. ♠ standard CNN/Transformer baselines, ♡ image manipulation detection methods, ♣ AI-generated image detection methods, ♢ multimodal large language models. The fine-tuned results are marked with ∗. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Model	IP2P [5]		CDS [50]		MagicBrush [86]		InfEdit [75]		MAG [48]		HQEdit [24]		PnP [27]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [49]	77.42	76.54	65.32	67.99	70.16	71.17	67.47	69.37	63.98	67.16	83.87	82.04	63.98	67.16
♢LLaVA-1.6 (7B) [38]	49.73	64.52	50.54	64.89	50.54	64.89	50.81	65.01	50.27	64.76	56.72	67.86	50.27	64.76
♢MiniCPM-V2.6 (8B) [83]	68.55	72.47	50.27	62.47	58.60	66.67	55.91	65.25	59.14	66.96	73.66	75.86	62.37	68.75
♢mPLUG-Owl3 (7B) [84]	62.90	61.67	52.69	55.78	55.91	57.51	53.76	56.35	56.18	57.66	67.47	64.72	52.69	55.78
♢CogVLM (17B) [69]	59.95	33.18	59.41	32.89	59.41	32.89	56.99	31.62	58.60	32.46	59.95	33.18	58.60	32.46
♢Ovis2.5 (9B) [44]	79.30	81.71	56.45	67.98	65.05	72.57	58.60	69.08	68.82	74.78	89.52	89.82	67.47	73.98
♢DeepSeekVL2 (small) [37]	55.11	67.45	51.08	65.53	52.96	66.41	50.54	65.28	51.88	65.90	56.72	68.24	51.88	65.90
♢InternVL3 (8B) [88]	70.70	64.72	55.91	54.95	62.90	59.17	59.95	57.31	60.48	57.64	75.00	68.26	60.48	57.64
♢InternVL3.5 (8B) [70]	76.08	75.48	60.75	65.24	69.89	70.98	66.67	68.84	64.52	67.49	81.45	79.88	65.05	67.82
♢Qwen2.5-VL (8B) [2]	70.97	61.97	63.98	56.77	68.28	59.86	65.32	57.70	66.13	58.28	72.85	63.54	66.13	58.28
♢Qwen3-VL (8B) [1]	76.61	75.49	59.14	63.81	68.28	69.43	66.13	68.02	65.05	67.34	81.99	80.00	65.05	67.34
♢Gemini2.5-Pro [8]	73.66	69.18	60.48	59.95	67.47	64.52	64.78	62.68	64.52	62.50	77.69	72.61	65.59	63.22
♢ChatGPT-4o [53]	74.73	74.46	69.89	70.98	68.55	70.08	68.01	69.72	69.89	70.98	76.34	75.69	69.35	70.62
♠ResNet50∗ [19]	88.98	80.15	88.17	80.00	88.98	80.15	88.98	80.15	88.98	80.15	88.71	80.10	88.98	80.15
♠Swin-T∗ [41]	90.59	82.80	90.59	82.80	90.59	82.80	90.59	82.80	90.59	82.80	90.59	82.80	90.59	82.80
♡MVSSNet∗ [10]	84.68	77.78	87.90	78.42	85.75	78.00	87.90	78.42	87.90	78.42	86.83	78.21	87.90	78.42
♡PSCCNet∗ [40]	90.59	83.62	91.13	83.70	91.13	83.70	91.13	83.70	91.13	83.70	90.86	83.66	91.13	83.70
♡HifiNet∗ [18]	90.59	87.08	93.28	87.41	91.40	87.18	93.28	87.41	93.01	87.37	92.47	87.31	93.01	87.37
♡FakeShield∗ [76]	92.74	87.34	93.28	87.41	92.47	87.31	93.28	87.41	93.28	87.41	92.74	87.34	93.28	87.41
♣CNNSpot∗ [68]	90.59	91.83	90.59	91.83	89.52	91.74	93.28	92.04	91.13	91.87	95.16	92.19	91.13	91.87
♣Lagrad∗ [64]	94.89	90.28	94.89	90.28	94.89	90.28	94.35	90.23	94.89	90.28	94.62	90.26	94.89	90.28
♣Univ∗ [52]	92.20	87.72	93.55	87.88	93.01	87.82	93.55	87.88	93.28	87.85	93.01	87.82	93.28	87.85
♣AIDE∗ [79]	**95.70**	**94.18**	**95.43**	**94.16**	**96.24**	**94.21**	**95.43**	**94.16**	**96.51**	**94.23**	**96.51**	**94.23**	**96.51**	**94.23**
ManipShield (Ours)∗	**97.31**	**95.26**	**97.04**	**95.25**	**97.31**	**95.26**	**97.04**	**95.25**	**97.58**	**95.28**	**97.31**	**95.26**	**97.58**	**95.28**

Model	ZONE [33]		PowerPaint [92]		BrushNet [26]		Any2Pix [34]		ReNoise [15]		FlowEdit(SD3) [29]		RFSE [65]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [49]	68.55	70.08	63.44	66.83	63.44	66.83	61.83	65.87	65.32	67.99	56.18	62.70	54.03	61.57
♢LLaVA-1.6 (7B) [38]	50.54	64.89	52.69	65.89	50.81	65.01	49.73	64.52	55.91	67.46	49.73	64.52	50.00	64.64
♢MiniCPM-V2.6 (8B) [83]	59.41	67.10	56.45	65.53	55.38	64.98	59.41	67.10	59.95	67.40	52.42	63.51	52.96	63.77
♢mPLUG-Owl3 (7B) [84]	53.23	56.06	56.72	57.96	54.84	56.92	53.23	56.06	57.53	58.42	51.88	55.36	51.34	55.09
♢CogVLM (17B) [69]	59.68	33.04	59.41	32.89	58.87	32.60	57.53	31.90	59.68	33.04	59.41	32.89	57.80	32.03
♢Ovis2.5 (9B) [44]	65.32	72.73	58.33	68.94	66.67	73.50	72.31	76.96	63.17	71.52	55.11	67.32	59.95	69.78
♢DeepSeekVL2 (small) [37]	52.69	66.28	51.34	65.65	51.34	65.65	51.61	65.78	52.15	66.03	52.96	66.41	52.15	66.03
♢InternVL3 (8B) [88]	61.56	58.31	55.38	54.64	59.68	57.14	62.37	58.82	61.83	58.48	53.49	53.62	48.66	51.15
♢InternVL3.5 (8B) [70]	67.20	69.19	62.63	66.34	64.25	67.32	68.55	70.08	65.05	67.82	57.53	63.43	59.68	64.62
♢Qwen2.5-VL (8B) [2]	64.78	57.33	63.17	56.23	64.25	56.96	62.10	55.52	66.13	58.28	59.68	53.99	59.14	53.66
♢Qwen3-VL (8B) [1]	66.94	68.54	62.90	66.01	64.25	66.83	68.28	69.43	66.13	68.02	56.18	62.18	57.26	62.76
♢Gemini2.5-Pro [8]	65.86	63.40	61.29	60.44	64.52	62.50	66.40	63.77	65.05	62.86	58.33	58.67	55.65	57.14
♢ChatGPT-4o [53]	70.97	71.73	72.04	72.49	72.04	72.49	67.74	69.54	69.89	70.98	70.16	71.17	68.55	70.08
♠ResNet50∗ [19]	88.98	80.15	88.98	80.15	88.98	80.15	88.71	80.10	88.98	80.15	88.71	80.10	87.63	79.90
♠Swin-T∗ [41]	90.59	82.80	90.59	82.80	90.32	82.76	90.59	82.80	90.59	82.80	89.78	82.67	90.32	82.76
♡MVSSNet∗ [10]	87.90	78.42	87.90	78.42	87.10	78.26	87.63	78.37	87.90	78.42	86.02	78.05	87.90	78.42
♡PSCCNet∗ [40]	91.13	83.70	91.13	83.70	90.86	83.66	90.86	83.66	91.13	83.70	91.13	83.70	91.13	83.70
♡HifiNet∗ [18]	93.28	87.41	93.01	87.37	93.01	87.37	92.20	87.28	93.28	87.41	92.20	87.28	93.01	87.37
♡FakeShield∗ [76]	93.01	87.37	93.01	87.37	92.74	87.34	92.47	87.31	93.28	87.41	92.74	87.34	93.01	87.37
♣CNNSpot∗ [68]	91.94	91.94	91.13	91.87	89.25	91.71	90.32	91.80	93.82	92.08	90.05	91.78	90.59	91.83
♣Lagrad∗ [64]	94.62	90.26	94.89	90.28	94.89	90.28	94.62	90.26	94.89	90.28	94.89	90.28	94.62	90.26
♣Univ∗ [52]	93.55	87.88	93.28	87.85	93.28	87.85	93.28	87.85	93.55	87.88	93.01	87.82	93.28	87.85
♣AIDE∗ [79]	**95.43**	**94.16**	**96.51**	**94.23**	**96.24**	**94.21**	**96.51**	**94.23**	**95.97**	**94.20**	**94.62**	**94.12**	**96.51**	**94.23**
ManipShield (Ours)∗	**97.58**	**95.28**	**97.58**	**95.28**	**97.31**	**95.26**	**97.31**	**95.26**	**97.04**	**95.25**	**97.31**	**95.26**	**97.31**	**95.26**

Model	FlowEdit(Flux) [29]		ACE [47]		Reflex [28]		OT [45]		Follow-Your-Shape [42]		OmniGen2 [74]		Qwen-Image-Edit [73]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [49]	50.54	59.83	66.13	68.50	62.90	66.50	55.91	62.56	54.57	61.85	52.42	60.75	59.14	64.32
♢LLaVA-1.6 (7B) [38]	48.92	64.15	51.61	65.38	50.54	64.89	49.73	64.52	50.27	64.76	49.19	64.27	49.73	64.52
♢MiniCPM-V2.6 (8B) [83]	49.73	62.22	57.80	66.24	59.95	67.40	54.84	64.71	52.96	63.77	52.15	63.37	58.33	66.52
♢mPLUG-Owl3 (7B) [84]	48.92	53.88	56.45	57.81	53.76	56.35	50.81	54.81	49.46	54.15	48.66	53.75	54.30	56.63
♢CogVLM (17B) [69]	58.33	32.31	58.87	32.60	59.41	32.89	59.41	32.89	58.87	32.60	57.53	31.90	55.65	30.96
♢Ovis2.5 (9B) [44]	52.15	65.90	64.52	72.27	71.77	76.61	62.37	71.07	56.18	67.85	56.45	67.98	61.83	70.78
♢DeepSeekVL2 (small) [37]	50.81	65.41	54.30	67.05	52.15	66.03	50.81	65.41	52.15	66.03	48.92	64.55	51.61	65.78
♢InternVL3 (8B) [88]	46.77	50.25	60.75	57.80	55.91	54.95	50.00	51.81	50.81	52.22	55.38	54.64	61.02	57.97
♢InternVL3.5 (8B) [70]	53.23	61.16	63.98	67.16	67.47	69.37	59.95	64.78	56.18	62.70	62.37	66.18	72.85	73.07
♢Qwen2.5-VL (8B) [2]	54.30	50.87	66.40	58.47	63.98	56.77	60.22	54.32	57.26	52.54	56.99	52.38	53.49	50.43
♢Qwen3-VL (8B) [1]	51.08	59.56	65.05	67.34	66.67	68.37	58.33	63.36	54.57	61.33	62.10	65.53	74.46	73.83
♢Gemini2.5-Pro [8]	52.15	55.28	65.32	63.04	60.22	59.78	56.18	57.44	55.91	57.29	59.41	59.30	64.52	62.50
♢ChatGPT-4o [53]	66.13	68.50	69.09	70.44	66.94	69.02	68.28	69.90	65.32	67.99	62.90	66.50	70.43	71.35
♠ResNet50∗ [19]	88.71	80.10	88.98	80.15	88.71	80.10	88.98	80.15	88.71	80.10	87.37	79.85	85.75	79.55
♠Swin-T∗ [41]	90.59	82.80	90.59	82.80	89.25	82.59	90.59	82.80	89.25	82.59	90.05	82.72	88.98	82.54
♡MVSSNet∗ [10]	87.63	78.37	87.90	78.42	87.90	78.42	87.63	78.37	87.10	78.26	81.72	77.16	83.87	77.61
♡PSCCNet∗ [40]	91.13	83.70	91.13	83.70	91.13	83.70	91.13	83.70	90.86	83.66	90.05	83.54	88.17	83.25
♡HifiNet∗ [18]	93.01	87.37	93.28	87.41	93.01	87.37	93.01	87.37	91.94	87.24	89.52	86.95	89.78	86.98
♡FakeShield∗ [76]	93.28	87.41	93.28	87.41	93.28	87.41	93.01	87.37	92.74	87.34	88.98	86.88	89.78	86.98
♣CNNSpot∗ [68]	90.86	91.85	95.16	92.19	91.67	91.91	89.52	91.74	89.78	91.76	86.29	91.45	85.75	91.40
♣Lagrad∗ [64]	94.89	90.28	94.89	90.28	94.89	90.28	94.89	90.28	94.89	90.28	92.20	90.03	93.01	90.10
♣Univ∗ [52]	93.55	87.88	93.55	87.88	93.55	87.88	93.55	87.88	93.55	87.88	92.47	87.76	91.94	87.69
♣AIDE∗ [79]	**95.70**	**94.18**	**96.24**	**94.21**	**96.77**	**94.24**	**95.97**	**94.20**	**94.89**	**94.13**	**95.16**	**94.15**	**92.47**	**93.99**
ManipShield (Ours)∗	**97.58**	**95.28**	**97.04**	**95.25**	**97.58**	**95.28**	**97.31**	**95.26**	**97.31**	**95.26**	**97.04**	**95.25**	**96.77**	**95.24**

Table 13: Accuracy results (Acc↑) of cross-validation on different training and testing subsets.

ResNet50 [19]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**93.19**	85.75	72.04	86.69	68.77	62.50	54.57	58.06	60.48	62.10	70.42
SD1.5	88.84	**92.11**	66.40	70.16	70.39	64.25	58.60	59.95	58.06	64.52	69.33
SD3	76.39	72.31	**91.67**	71.10	62.99	61.02	60.75	56.45	58.60	63.44	67.47
SDXL	77.24	68.28	60.22	**92.47**	63.98	63.31	56.99	58.60	59.41	60.48	66.10
FLUX	71.06	74.73	70.97	77.42	**93.41**	77.15	65.05	64.78	55.91	66.40	71.69
DiT	69.80	66.13	66.13	73.66	71.33	**92.61**	63.44	63.98	58.33	68.28	69.37
Full Dataset	88.80	88.98	88.84	88.71	88.62	86.56	65.32	58.06	56.99	68.01	84.33
Swin-T [41]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**90.37**	87.46	73.39	81.45	63.40	65.19	57.80	57.53	61.56	66.40	70.45
SD1.5	80.02	**90.95**	65.59	69.49	66.08	61.16	53.76	65.05	52.42	65.05	66.96
SD3	78.99	74.46	**92.20**	70.03	64.20	57.26	56.72	57.26	60.75	63.71	67.56
SDXL	70.47	73.92	62.37	**92.20**	61.56	69.09	52.42	63.71	55.38	56.45	65.76
FLUX	74.06	79.57	68.28	81.59	**92.43**	81.18	66.13	62.10	55.38	65.05	72.58
DiT	74.10	67.74	67.20	69.76	71.42	**91.13**	62.10	67.47	64.78	61.56	69.73
CNNSpot [68]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**95.21**	77.42	67.20	78.36	61.78	64.11	53.49	57.53	51.88	58.60	66.56
SD1.5	85.93	**92.20**	62.10	70.03	67.03	66.53	50.00	52.15	61.29	69.62	67.69
SD3	66.53	69.35	**95.43**	70.97	69.22	60.62	61.29	52.69	61.56	67.20	67.49
SDXL	79.35	61.11	62.37	**93.82**	68.06	65.73	55.91	56.45	58.33	61.83	66.29
FLUX	62.01	67.11	72.58	71.64	**95.83**	81.72	69.09	63.17	55.65	58.60	69.74
DiT	69.40	66.31	68.28	66.13	65.77	**92.20**	57.26	56.99	58.60	70.16	67.11
AIDE [79]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**96.06**	88.35	82.53	87.90	70.39	66.67	63.44	64.78	61.29	68.01	74.94
SD1.5	84.90	**93.37**	69.09	71.51	68.59	61.83	58.87	62.90	64.78	65.59	70.14
SD3	74.87	70.07	**95.70**	71.77	70.34	64.92	61.83	59.41	59.95	69.89	69.87
SDXL	83.60	73.21	71.77	**95.83**	67.47	63.04	58.87	58.87	61.83	64.78	69.93
FLUX	78.45	74.82	70.70	78.09	**95.61**	75.81	77.15	70.16	65.05	70.16	75.60
DiT	74.46	63.17	68.28	79.30	78.00	**94.35**	72.85	71.24	73.92	74.73	75.03
MVSSNet [10]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**90.68**	83.69	61.56	74.87	58.87	53.90	59.41	58.33	51.88	61.83	65.50
SD1.5	80.60	**92.83**	59.41	65.19	70.52	64.78	56.72	51.88	56.99	61.29	66.02
SD3	77.02	62.63	**86.83**	68.95	57.17	57.26	62.10	58.33	51.08	54.84	63.62
SDXL	72.09	70.16	62.10	**86.29**	54.75	59.01	56.99	50.27	59.14	54.30	62.51
FLUX	67.92	72.94	62.37	73.52	**85.08**	78.76	61.56	66.13	55.91	68.01	69.22
DiT	68.59	63.80	58.87	73.39	69.89	**89.38**	60.75	56.45	68.82	67.47	67.74
HiFiNet [18]	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**95.21**	81.72	79.84	94.89	76.08	60.22	65.59	62.10	55.91	59.14	73.07
SD1.5	95.61	**91.94**	67.74	70.16	70.25	68.95	61.29	56.18	55.65	66.13	70.39
SD3	83.87	72.31	**97.85**	74.33	62.99	61.16	66.94	58.87	54.57	59.95	69.28
SDXL	83.83	67.29	64.25	**90.19**	71.15	61.83	60.22	61.83	56.18	67.20	68.40
FLUX	72.58	73.30	76.34	73.39	**94.27**	75.27	68.28	67.20	58.87	69.09	72.86
DiT	72.94	73.48	61.83	73.52	75.94	**91.53**	68.28	60.22	67.47	67.74	71.29
ManipShield (Ours)	Testing Subset										
Training Subset	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
SD1.4	**94.31**	90.59	83.33	85.89	74.10	73.79	67.74	66.67	68.55	68.01	77.30
SD1.5	91.85	**92.92**	69.35	69.09	79.12	70.43	66.40	63.71	62.90	64.25	73.00
SD3	80.33	79.21	**90.86**	81.59	67.38	65.59	70.43	68.28	67.47	71.51	74.27
SDXL	80.51	70.88	78.49	**92.47**	65.50	64.92	66.94	57.80	63.17	62.90	70.36
FLUX	85.39	85.75	72.85	83.47	**92.07**	79.03	77.96	71.24	70.16	76.08	79.40
DiT	76.48	70.34	75.54	76.88	81.32	**95.70**	76.61	73.39	77.69	78.49	78.24

Method	GPU Memory	MACs(G)	Trainable Params(M)	Infer Time (s)
MVSSNet [10]	4GB	106.20	85.75	0.468
AIDE [79]	9GB	325.25	180.24	0.726
FakeShield [76]	42GB	5809.82	486.43	1.530
ManipShield	23GB	2980.08	122.26	0.825

Table 14: Cost analysis.



Figure 8: Example images generated by different manipulation models in ManipBench.

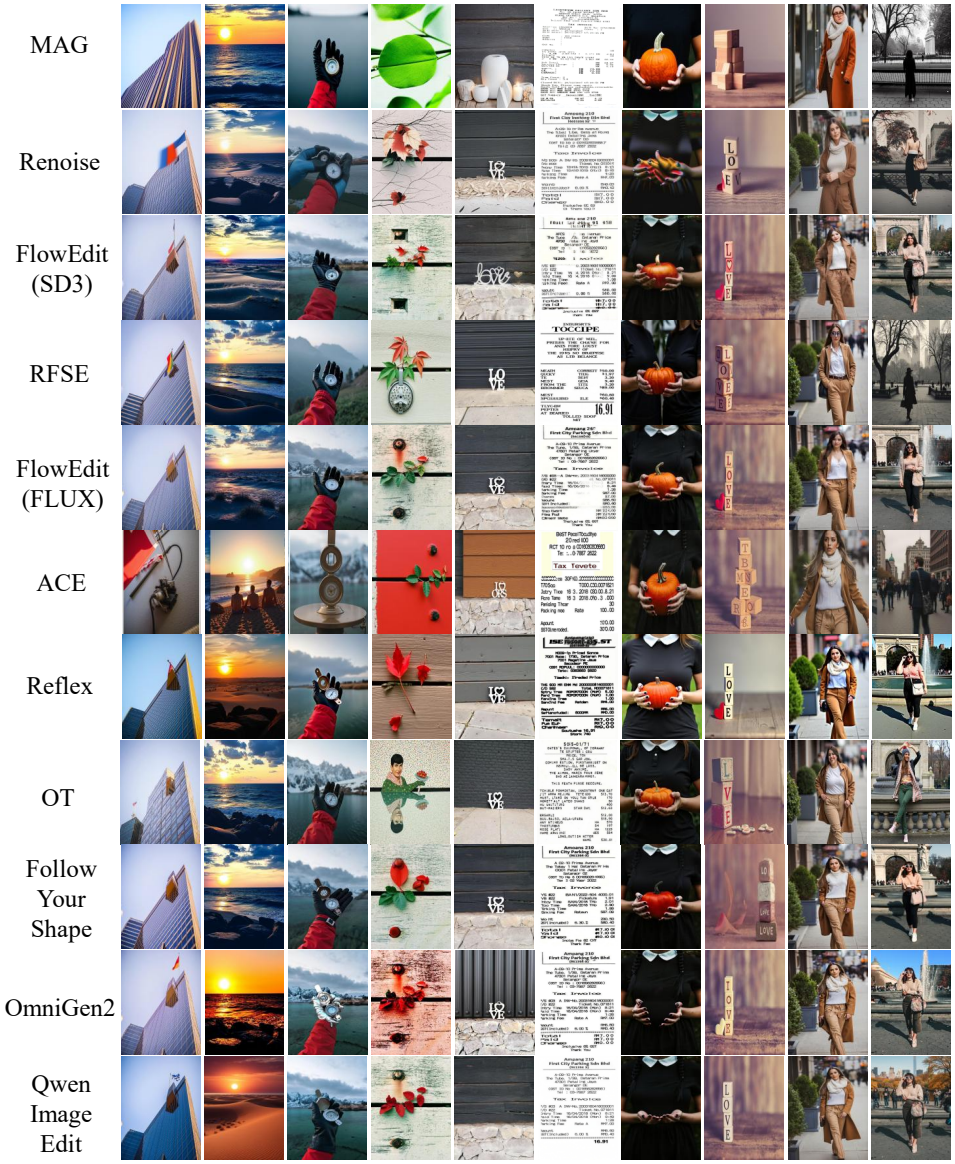


Figure 9: Example images generated by different manipulation models in ManipBench.

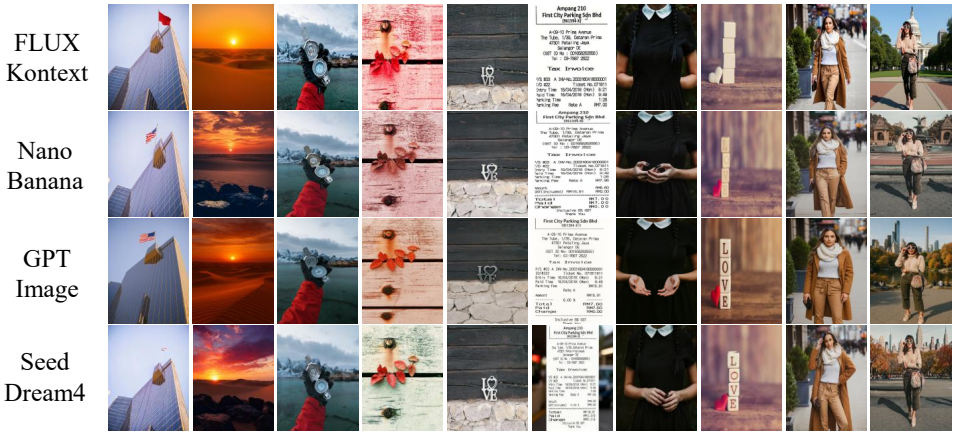


Figure 10: Example images generated by different manipulation models in ManipBench.

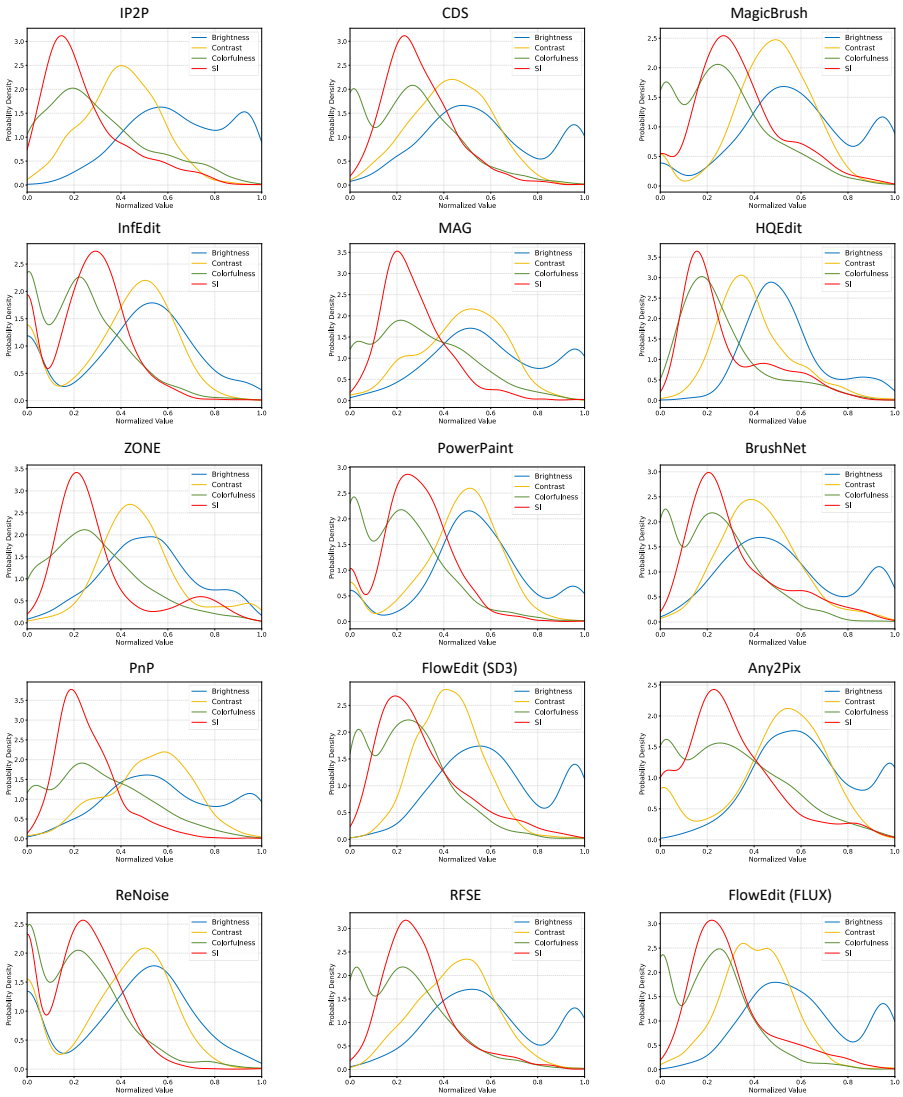


Figure 11: Feature distribution of images generated by different manipulation models.

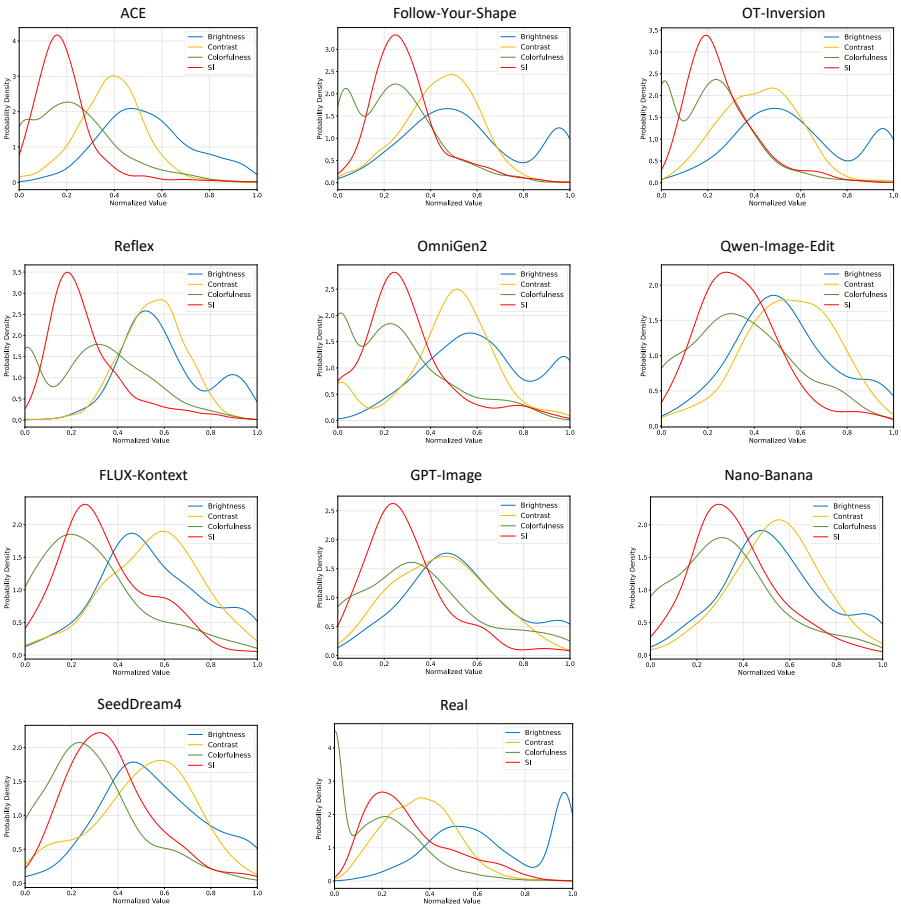


Figure 12: Feature distribution of images generated by different manipulation models and real images.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly and accurately reflect the paper's main contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We include a limitations section in the Section 14 of supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results such as theorems, formal assumptions, or mathematical proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides sufficient details on the dataset construction, model architecture, training procedure, and evaluation protocols to enable reproduction of the main experimental results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We plan to release the code and models upon paper acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides sufficient details on training configuration in Section 12 of supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We evaluate our model under different dataset splits to demonstrate its robustness and generalization ability. The corresponding analysis is provided in Section 5.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides information about the computational setup in Section 14.1 of supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics in all respects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses the potential social impacts in Section 15 of the supplementary material.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The paper describes responsible release considerations for the dataset and model Section 15.3 of the supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All code and datasets used in this work are sourced from publicly available open-source repositories on GitHub and are used in compliance with their respective licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper provides detailed descriptions of the newly introduced assets, including their construction process, annotations, and usage.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The paper includes human annotation. The instructions provided to annotators and the evaluation protocol are described in the Section 3.2 of main text and Section 10 in supplementary material.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: The study follows institutional ethical guidelines, and appropriate approval was obtained prior to conducting the human subject experiments. Participant risks were minimal and limited to standard image evaluation tasks.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper uses LLMs as a core component of the proposed multi-agent framework, where different agents are built upon LLM-based reasoning, perception, and evaluation modules.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.