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

Zitong Xu[1], Huiyu Duan[1†], Xiaoyu Wang[2], Zhaolin Cai[1], Kaiwei Zhang[1],
Qiang Hu[1], Jing Liu[3], Xiongkuo Min[1†] Guangtao Zhai[1†]
[1]Institute of Image Communication and Network Engineering, Shanghai Jiao Tong University
[2]University of Electronic and Science Technology of China, [3]Tianjin University
{xuzitong, huiyuduan, zhangkaiwei, qiang.hu, minxiongkuo, zhaiguangtao}@sjtu.edu.cn

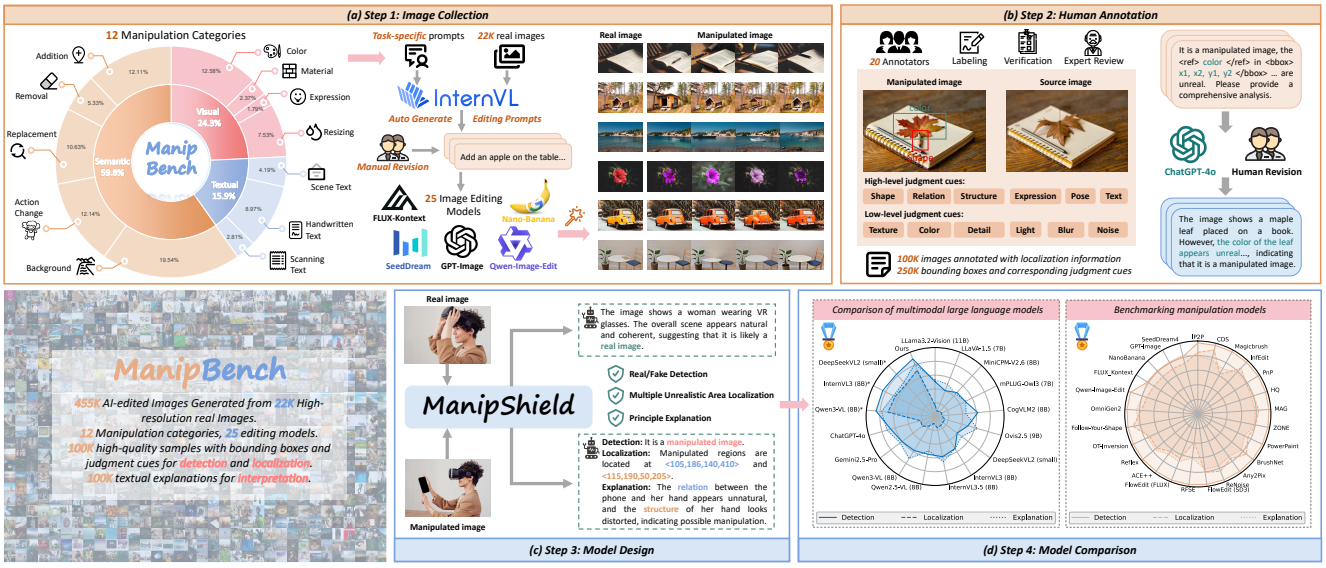

Figure 1. An overview of the constructed image manipulation database and the proposed image manipulation detection model, termed ManipBench and ManipShield, respectively. (a) We first collect 22K real-world images and produce numerous editing prompts using multimodal large language model across 12 manipulation categories. Then 25 latest image editing models are applied to generate 455K manipulated images. (b) 100K images are further annotated with localization and explanation information. (c) We design ManipShield for image manipulation detection, localization and explanation. (d) We perform model comparisons based on ManipBench.

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. Both ManipBench and ManipShield will be released upon publication.*

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 [12, 20, 41, 74]. With the rapid advancement of generative models [15, 34], numerous powerful artificial intelligence (AI) editing techniques have emerged, such as NanoBanana [1] and Qwen-Image-Edit [69]. Unlike early deepfake techniques, which are typically limited to straightforward, malicious facial or identity manipulations [12, 20, 41], 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 [59]. Furthermore, these editing methods are increasingly imperceptible, maintaining strong subject consistency while producing highly realistic images [31, 69, 70], which underscores the urgent need for more advanced image manipulation detection and localization approaches.

Existing IMDL datasets and benchmarks exhibit several critical limitations. **(i) Limited content diversity:** Existing datasets primarily focus on facial or identity forgeries [10, 13, 49, 76], whereas modern editing techniques allow personalized and localized modifications guided by flexible instructions [9, 46, 69]. 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 [8, 25, 59, 74], 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 [10, 38, 49, 76]. While some improved benchmarks include annotations of manipulated regions [8, 13], 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.

To address the limitations of existing datasets, we introduce ManipBench, 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 annotated with bounding boxes and corresponding judgment cues, along with textual explanations to support detection decisions. 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 tampered 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 (LLM) 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. Furthermore, by validating on manipulated images generated using several closed-source image editing models that are unseen in training set, we show that it achieves strong zero-shot performance on unseen generative models, highlighting its superior generalization capability. In summary, our main contributions are:

- We introduce ManipBench, a large-scale image manipulation detection dataset focusing on AI-edited images, comprising over 450K manipulated images produced by 25 state-of-the-art editing models, with fine-grained annotations including bounding boxes, judgment cues, and detailed 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 [55] and FLUX [15], numerous editing methods have emerged [24]. 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

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 Explanation
IMD2020 [49]	General	3	3	✗	35,000	✗	✗	✗
DDFT [10]	Face	4	7	✗	240,336	✗	✗	✗
DF40 [76]	Face	4	40	✗	1M+	✗	✗	✗
FakeBench [38]	General	6	10	✗	3,000	✗	✗	✓
FakeShield [74]	General	7	3	1	196,123	181,000	✓	✓
SID-Set [25]	General	2	1	1	200,000	200,000	✓	✓
GIM [8]	General	2	3	3	320,000	320,000	✓	✗
FragFake [59]	General	2	4	4	20,000	20,000	✗	✗
ManipBench (Ours)	**General**	**12**	**25**	**25**	**455,000**	**455,000**	✓	✓

cat") [24, 72]. Description-based methods primarily rely on inversion-based modification [17, 29, 31, 48, 71], 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 [6] and MagicBrush [81] generally utilize large-scale paired editing datasets to train editing models. Recent models [69, 70] further adopt the DiT [34] 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 [13, 19, 49, 68] mainly focus on manual manipulations such as splicing, removal, and copy-move. With the rise of generative models, datasets [5, 10, 22, 58, 76] use GANs and diffusion models to produce deepfakes, but most are limited to facial content. Some datasets [38, 62, 63] 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 [8, 25, 59, 74] have incorporated text-guided edited images, but typically cover only a limited range of models and manipulation types with only real/fake value annotations. Moreover, most datasets only adopt mask-guided editing methods [8, 25, 74], restricting to simple, localized edits. Recent advances in MLLMs [11, 67, 77] have driven some benchmarks [25, 38, 74] 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 [27, 33, 37, 56, 83]. However, since manipulation types vary widely in real-world scenarios, subsequent research has shifted toward universal manipulation localization methods [12, 20, 41, 80], which aim to identify artifacts and inconsistencies across diverse manipulation types. With the rapid emergence of advanced editing models, recent methods [8, 25, 59, 74] 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 artifacts, their generalization is limited. Some works [25, 38, 74] 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 [25, 74].

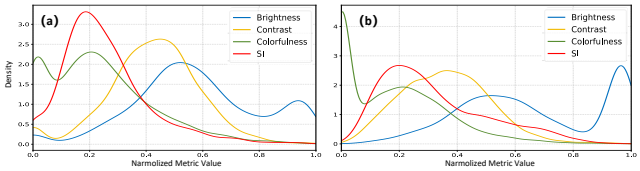

Figure 2. Feature distribution of the ManipBench. (a) Manipulated images. (b) Real images. Manipulated images exhibit decreased spatial information (SI) but increased colorfulness and contrast.

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 [67], 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 select 21 state-of-the-art open-source editing models with diverse backbones to generate images, including IP2P [6], CDS [48], MagicBrush [81], PnP [29], Any2Pix [35], InfEdit [71], MAG [47], ZONE [36], PowerPaint [85], ReNoise [17], BrushNet [28], HQEdit [26], RFSE [61], FlowEdit (SD3) [31], FlowEdit (FLUX) [31], ACE++ [46], OmniGen2 [70], Reflex [30], OT-Inversion [3], Follow-Your-Shape [43], and

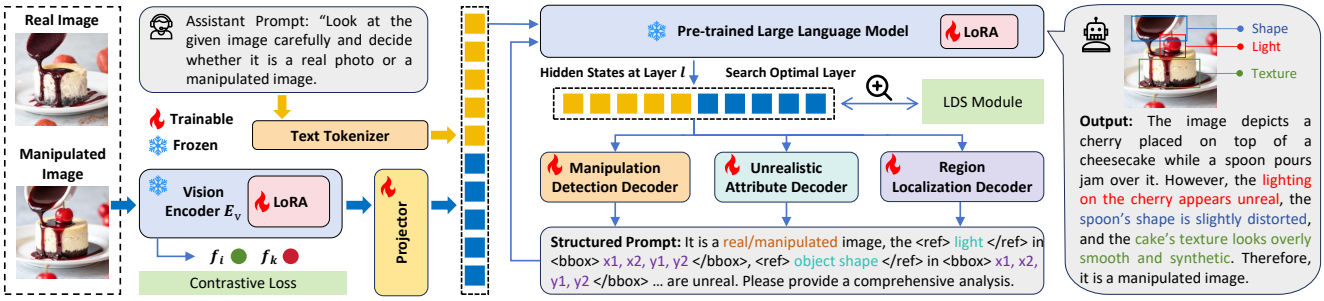

Figure 3. An overview of ManipShield. First, manipulated and real images are paired to train the vision encoder through **contrastive LoRA fine-tuning**. Then, we freeze the vision encoder, and feed the projected image features and an assistant prompt into the LLM. The **layer discrimination selection (LDS) module** then identifies the LLM layer that best separates positive and negative samples. The hidden state from this layer is passed through three decoders including: a **manipulation detection decoder** for classification, an **unrealistic attribute decoder** for judgment cues extraction, and a **region localization decoder** for bounding boxes prediction. The outputs are integrated into a **structured prompt**, which, together with the image, is used to generate explicit explanatory analysis.

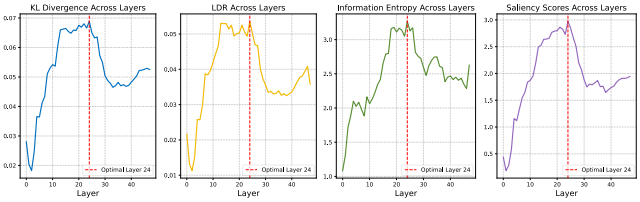

Figure 4. The plots of KL divergence, LDR, information entropy and saliency score across hidden states from different LLM layers.

Qwen-Image-Edit [69]. For generality verification, we further utilize four advanced closed-source editing models to produce manipulated images, including: GPT-Image [52], FLUX-Kontext [32], NanoBanana [9], and SeedDream 4 [57]. The details of these editing models are shown in *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 and diverse set suitable for benchmarking image manipulation detection. Figure 2 shows the feature distribution of images in ManipBench. Compared to real images, manipulated images exhibit lower spatial information (SI) but higher colorfulness and contrast.

3.2. Metadata Acquisition

Localizing edited regions is challenging because advanced editing methods often modify both the target and surrounding areas to maintain overall image consistency, resulting in multiple manipulated regions per image. To enable fine-grained localization analysis, we select 100K images for further annotation. Human annotators, presented with paired real and manipulated images, drew bounding boxes around each modified region and recorded judgment cues, 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 on a calibrated 3840 × 2160 LED monitor. Twenty trained 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 arbitration. For explanatory analysis, we first use ChatGPT-4o [51] to generate textual descriptions based on bounding boxes and cues, which are then manually reviewed and revised. The complete metadata annotation process is illustrated in Figure 1(b). 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 3, ManipShield is built upon an MLLM architecture, *i.e.*, InternVL3.5 [67]. 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 [14, 73], motivating us to fine-tune the pre-trained model. We adopt InternVL3.5 [67] as our backbone and apply the LoRA technique [23]. LoRA models the weight update of each layer $W \in \mathbb{R}^{d\times k}$

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 [20]	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 [74]	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 [50]	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 [75]	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
♢LLama3.2-Vision (11B) [2]	71.37	72.38	64.85	67.73	63.58	66.93	56.18	62.70	57.35	63.47	55.78	62.53	48.66	58.92	53.23	61.16	49.46	59.31	50.41	10.25	61.13	63.89
♢LLaVA-1.6 (7B) [39]	51.43	65.32	51.08	65.14	52.82	65.99	49.73	64.52	50.18	64.72	49.46	64.39	48.92	64.15	49.19	64.27	49.73	64.52	39.03	7.475	49.17	62.66
♢MiniCPM-V2.6 (8B) [78]	61.02	68.28	58.40	66.59	59.68	67.25	52.42	63.51	54.71	64.69	55.24	64.94	53.23	63.90	52.96	63.77	48.66	61.72	32.94	8.668	55.92	63.60
♢mPLUG-Owl3 (7B) [79]	58.15	58.95	54.37	56.68	55.38	57.24	51.88	55.36	51.79	55.35	51.48	55.19	48.12	53.49	48.92	53.88	50.27	54.55	48.35	8.162	53.54	54.51
♢CogVLM (17B) [66]	59.05	32.70	59.14	32.75	58.60	32.47	59.41	32.89	58.78	32.55	56.59	31.43	53.23	29.84	50.27	28.57	51.88	29.25	52.38	18.09	59.27	31.56
♢Ovis2.5 (9B) [45]	69.62	75.99	64.45	72.29	67.74	74.24	55.11	67.32	61.16	70.58	59.14	69.38	52.69	66.15	56.99	68.25	52.42	66.03	35.77	9.971	61.97	69.34
♢DeepSeekVL2 (small) [11]	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 (8B) [82]	64.16	60.34	59.27	56.93	62.10	58.65	53.49	53.62	52.15	53.03	58.20	56.31	48.39	51.02	57.80	56.02	51.34	52.49	61.48	19.69	57.92	54.43
♢InternVL3.5 (8B) [67]	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
♢Qwen2.5-VL (8B) [4]	67.92	59.69	64.58	57.20	64.11	56.90	59.68	53.99	60.22	54.44	55.24	51.41	51.08	49.16	49.73	48.48	47.31	47.31	74.48	12.48	61.93	53.66
♢Qwen3-VL (8B) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	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∗ [50]	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∗ [75]	**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)∗ [11]	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)∗ [82]	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)∗ [77]	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**

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

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

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, 18]. 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 [67], 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 key properties of the 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:

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

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
♢LLaVA-1.6 (7B) [39]	0.087	0.085	0.092	0.110	0.070	0.090	0.073	0.101	0.073	0.102	0.080
♢Ovis2.5 (9B) [45]	0.159	0.157	0.164	0.181	0.134	0.151	0.166	0.142	0.150	0.148	0.148
♢DeepSeekVL2 (small) [11]	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) [67]	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) [77]	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 [9]	0.257	0.287	0.264	0.272	0.198	0.252	0.234	0.228	0.213	0.231	0.234
♢ChatGPT-4o [51]	0.256	0.261	0.245	0.267	0.184	0.231	0.204	0.217	0.220	0.239	0.221
♣Fast R-CNN* [54]	0.435	0.451	0.413	0.456	0.413	0.437	0.349	0.310	0.312	0.358	0.378
♣DETR* [7]	0.661	0.696	0.633	0.645	0.574	0.638	0.521	0.491	0.478	0.510	0.573
♣DeformableDETR* [84]	0.701	0.737	0.716	0.740	**0.694**	0.718	0.677	0.617	0.621	0.629	0.674
♣GroundingDINO* [40]	**0.751**	**0.792**	**0.773**	**0.779**	0.688	0.706	0.661	0.658	0.675	**0.685**	0.708
♢DeepSeekVL2 (small)* [11]	0.673	0.719	0.694	0.702	0.655	0.650	0.655	**0.671**	0.653	0.654	0.697
♢InternVL3 (8B)* [82]	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)* [77]	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**

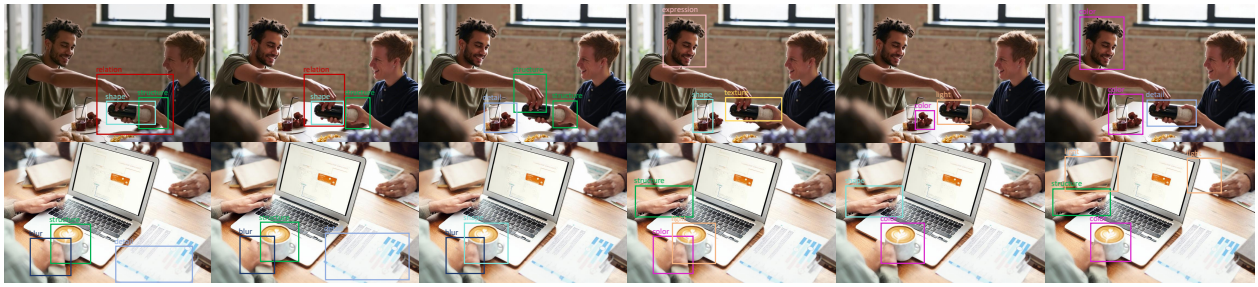

Figure 5. Examples of bounding box prediction results.

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:

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

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 4, 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 *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 4. Comparison results (CSS↑) of explanation capabilities. All embeddings are extracted using the InternVL3.5 [67]. The best results are highlighted in **red**, and the second-best results are highlighted in **blue**.

Model/Subset	Testing Subset										
	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Overall
LLaVA1.6 (7B) [39]	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) [45]	0.645	0.638	0.641	0.634	0.637	0.620	0.618	0.624	0.615	0.622	0.635
DeepSeekVL2 (small) [11]	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) [77]	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) [67]	0.685	0.672	0.682	0.653	0.659	0.662	0.672	0.661	0.653	0.676	0.665
ChatGPT-4o [51]	**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 [9]	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**

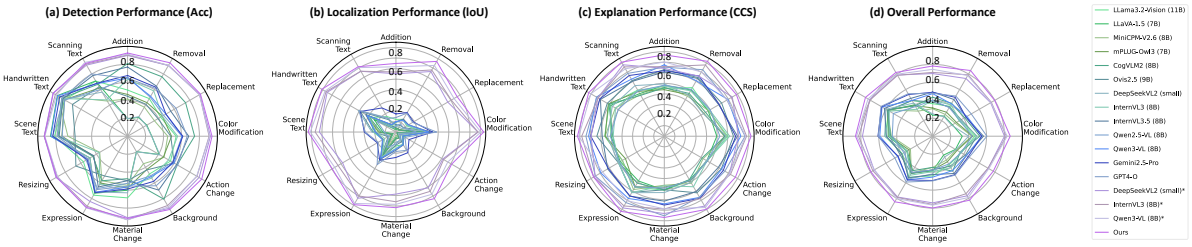

Figure 6. Comparison of performance of different MLLMs in terms of image manipulation detection, localization, and explanation, respectively.

Table 5. Ablation study on the different backbones, decoders and LoRA tuning strategy.

Backbone&Strategy					Metrics		
Backbone	Decoders	LoRA(vision)	LoRA(llm)	LDS	Acc↑	IoU↑	CCS↑
InternVL3.5 [67]		✓	✓		90.3	0.225	0.347
InternVL3.5 [67]	✓		✓		92.6	0.714	0.852
InternVL3.5 [67]	✓	✓			85.3	0.379	0.231
InternVL3.5 [67]	✓	✓	✓		93.5	0.732	0.859
InternVL3.5 [67]	✓	✓	✓	✓	**94.7**	**0.765**	**0.892**
InternVL3 [82]	✓	✓	✓	✓	92.4	0.729	0.846
DeepSeekVL2 [11]	✓	✓	✓	✓	88.5	0.693	0.812
Qwen3-VL [77]	✓	✓	✓	✓	92.8	0.745	0.861

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.

5.2. Comparison Results on ManipBench

We first divide ManipBench into two subsets: in-distribution and out-of-distribution. In-distribution subset contains images generated by open-source editing methods that also appear in the training set, while the out-of-distribution subset includes images produced by closed-source editing methods that are unseen during training.

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. 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 5. 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.

To assess the quality of **textual explanation**, we compare the performance of pre-trained MLLMs with our ManipShield. As shown in Table 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, such as lighting, color, or texture. Consequently, these models have limitations in fine-grained manipulation analysis. The unrealistic attribute decoder in our model helps overcome this limitation by accurately detecting and describing subtle

Table 6. Accuracy results (Acc↑) of cross-validation on different training and testing subsets using ResNet50 [21] and our ManipShield. Proprietary subset includes images generated by FLUX-Kontext [32], NanoBanana [9], GPT-Image [52] and SeedDream4 [57].

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	**93.19**	85.75	72.04	86.69	68.77	62.50	67.74
SD1.5	88.84	**92.11**	66.40	70.16	70.39	64.25	60.28	88.84	**92.11**	66.40	70.16	70.39	64.25	64.31
SD3	76.39	72.31	**91.67**	71.10	62.99	61.02	59.81	76.39	72.31	**91.67**	71.10	62.99	61.02	69.42
SDXL	77.24	68.28	60.22	**92.47**	63.98	63.31	58.87	77.24	68.28	60.22	**92.47**	63.98	63.31	62.70
FLUX	71.06	74.73	70.97	77.42	**93.41**	77.15	63.04	71.06	74.73	70.97	77.42	**93.41**	77.15	73.86
DiT	69.80	66.13	66.13	73.66	71.33	**92.61**	63.51	69.80	66.13	66.13	73.66	71.33	**92.61**	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 7. 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										
	SD1.4	SD1.5	SD3	SDXL	FLUX	DiT	FLUX-Kontext	NanoBanana	GPT-Image	SeedDream4	Avg
ResNet50 [21]	79.42	76.55	71.24	78.58	71.81	70.14	59.90	60.30	58.47	64.20	69.37
Swin-T [42]	78.00	79.02	71.51	77.42	69.85	70.83	58.15	62.19	58.38	63.04	68.84
MVSSNet [12]	76.15	74.34	65.19	73.70	66.05	67.18	59.59	56.90	57.30	61.29	65.77
HiFiNet [20]	84.01	76.67	74.64	79.41	75.11	69.83	65.10	61.07	58.11	64.87	70.88
CNNSpot [65]	76.40	72.25	71.33	75.16	71.28	71.82	57.84	56.50	57.89	64.34	67.48
AIDE [75]	**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 8. 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 [13]	IMD2020 [49]	DDFT [10]	DF40 [76]
MVSSNet [12]	82.53	85.25	79.46	82.71
PSCCNet [41]	89.41	80.60	85.64	86.90
HifiNet [20]	87.16	86.38	84.32	85.25
FakeShield [74]	**91.65**	**87.43**	86.29	85.65
Univ [50]	86.32	84.34	**89.22**	85.24
AIDE [75]	88.47	86.76	85.18	**88.37**
ManipShield (Ours)	**93.71**	**92.53**	**91.62**	**92.46**

inconsistencies.

Figure 6 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 5. 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 our ManipBench, we observe that the performance of existing image manipulation detection methods notably decreases when confronted with unseen generation models. To further investigate this issue, we conduct a series of experiments to comprehensively evaluate the generalization ability of different detection approaches and the influence of backbone architectures. Specifically, we train each detection model using images generated by one particular backbone and then test it on images produced by other backbones.

As presented in Tables 6, although detection methods achieve high detection accuracy when evaluated on test subsets generated by the same backbone, their performance degrades sharply when applied to subsets generated by unseen backbones. This phenomenon indicates that many existing models tend to overfit to the characteristics of specific generation models, rather than learning generalizable manipulation cues. Unlike previous methods, ManipShield can capture high-level unrealistic features, enabling strong generalization across diverse manipulation backbones. The results in Table 7, which report the average performance on the same testing subset when trained on different training subsets, further validating the robust generalization capability of our approach.

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 [13] and IMD2020 [49] are well-known manually manipulated image datasets, while DDFT [10] and DF40 [76] 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 8, 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, the first 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.

References

[1] Nano Banana AI. Nano banana ai: Advanced image editor, 2025. https://ainanobanana.io. 2, 3

[2] AI Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. *Meta AI Blog*, 2024. 5, 8, 9, 12

[3] Anonymous. Optimal transport for rectified flow image editing: Unifying inversion-based and direct methods. 3, 2, 12

[4] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 5, 9, 10, 12

[5] Chaitali Bhattacharyya, Hanxiao Wang, Feng Zhang, Sungho Kim, and Xiatian Zhu. Diffusion deepfake. *arXiv preprint arXiv:2404.01579*, 2024. 3

[6] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18392–18402, 2023. 3, 2, 5, 12

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229, 2020. 6

[8] Yirui Chen, Xudong Huang, Quan Zhang, Wei Li, Mingjian Zhu, and Qiangyu others Yan. 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)*, pages 2311–2319, 2025. 2, 3

[9] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, 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. 2, 4, 5, 6, 7, 8, 3, 9, 10, 12

[10] Hao Dang, Feng Liu, Joel Stehouwer, Xiaoming Liu, and Anil Jain. On the detection of digital face manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 8

[11] DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024. 3, 5, 6, 7, 9, 12

[12] Chengbo Dong, Xinru Chen, Ruohan Hu, Juan Cao, and Xirong Li. Mvss-net: Multi-view multi-scale supervised networks for image manipulation detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 45 (3):3539–3553, 2022. 2, 3, 5, 8, 9, 12, 13

[13] Jing Dong, Wei Wang, and Tieniu Tan. Casia image tampering detection evaluation database. In *Proceedings of the IEEE China summit and international conference on Signal and Information Processing (ChinaSIP)*, pages 422–426. IEEE, 2013. 2, 3, 8

[14] Huiyu Duan, Qiang Hu, Jiarui Wang, Liu Yang, Zitong Xu, Lu Liu, Xiongkuo Min, Chunlei Cai, Tianxiao Ye, Xiaoyun Zhang, et al. Finevq: Fine-grained user generated content video quality assessment. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3206–3217, 2025. 4

[15] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024. 2, 3

[16] S. Fu, T. Bonnen, D. Guillory, and T. Darrell. Hidden in plain sight: Vlms overlook their visual representations. *arXiv preprint arXiv:2506.08008*, 2025. 5

[17] Daniel Garibi, Or Patashnik, Andrey Voynov, Hadar Averbuch-Elor, and Daniel Cohen-Or. Renoise: Real image inversion through iterative noising. *arXiv preprint arXiv:2403.14602*, 2024. 3, 2, 6, 12

[18] V. Goloviznina and E. Kotelnikov. I've got the ⟨answer⟩! interpretation of llms hidden states in question answering. *arXiv preprint arXiv:2406.02060*, 2024. 5

[19] Haiying Guan, Mark Kozak, Eric Robertson, Yooyoung Lee, Amy N Yates, Andrew Delgado, 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)*, pages 63–72, 2019. 3

[20] Xiao Guo, Xiaohong Liu, Zhiyuan Ren, Steven Grosz, Iacopo Masi, and Xiaoming Liu. Hierarchical fine-grained image forgery detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3, 5, 8, 9, 12, 13

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5, 8, 9, 12, 13

[22] Yinan He, Bei Gan, Siyu Chen, Yichun Zhou, Guojun Yin, Luchuan Song, et al. Forgerynet: A versatile benchmark for comprehensive forgery analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 4360–4369, 2021. 3

[23] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, et al. Lora: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 4

[24] Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, et al. Diffusion model-based image editing: A survey. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, PP, 2024. 2, 3

[25] Zhenglin Huang, Jinwei Hu, Xiangtai Li, Yiwei He, Xingyu Zhao, et al. Sida: Social media image deepfake detection, lo-

calization and explanation with large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28831–28841, 2025. 2, 3

[26] Mude Hui, Siwei Yang, Bingchen Zhao, Yichun Shi, Heng Wang, Peng Wang, et al. Hq-edit: A high-quality dataset for instruction-based image editing. *arXiv preprint arXiv:2404.09990*, 2024. 3, 2, 12

[27] Ashraful Islam, Chengjiang Long, Arslan Basharat, and Anthony Hoogs. 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)*, 2020. 3

[28] Xuan Ju, Xian Liu, Xintao Wang, Yuxuan Bian, Ying Shan, and Qiang Xu. Brushnet: A plug-and-play image inpainting model with decomposed dual-branch diffusion. *arXiv preprint arXiv:2403.06976*, 2024. 3, 2, 12

[29] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. 3, 2, 12

[30] Jimyeong Kim, Jungwon Park, Yeji Song, Nojun Kwak, and Wonjong Rhee. Reflex: Text-guided editing of real images in rectified flow via mid-step feature extraction and attention adaptation. *arXiv preprint arXiv:2507.01496*. 3, 2, 12

[31] Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. *arXiv preprint arXiv:2412.08629*, 2024. 2, 3, 12

[32] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, et al. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint*, 2025. 4, 8, 3, 6

[33] Haodong Li and Jiwu Huang. Localization of deep inpainting using high-pass fully convolutional network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3

[34] Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. Dit: Self-supervised pre-training for document image transformer. *arXiv preprint arXiv:2203.02378*, 2022. 2, 3

[35] Shufan Li, Harkanwar Singh, and Aditya Grover. Instructany2pix: Flexible visual editing via multimodal instruction following. *arXiv preprint arXiv:2312.06738*, 2023. 3, 2, 12

[36] Shanglin Li, Bohan Zeng, Yutang Feng, Sicheng Gao, Xuhui Liu, Jiaming Liu, et al. Zone: Zero-shot instruction-guided local editing. *arXiv preprint arXiv:2312.16794*, 2023. 3, 2, 12

[37] Yuanman Li and Jiantao Zhou. 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. 3

[38] Yixuan Li, Xuelin Liu, Xiaoyang Wang, Bu Sung Lee, Shiqi Wang, Anderson Rocha, et al. Fakebench: Probing explainable fake image detection via large multimodal models. *IEEE Transactions on Information Forensics and Security (TIFS)*, 20:8730–8745, 2025. 2, 3

[39] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26296–26306, 2024. 5, 6, 7, 3, 9, 12

[40] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European Conference on Computer Vision (ECCV)*, pages 38–55, 2024. 6

[41] Xiaohong Liu, Yaojie Liu, Jun Chen, and Xiaoming Liu. Pscc-net: Progressive spatio-channel correlation network for image manipulation detection and localization. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 32(11):7505–7517, 2022. 2, 3, 5, 8, 9, 12

[42] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021. 5, 8, 9, 12, 13

[43] Zeqian Long, Mingzhe Zheng, Kunyu Feng, Xinhua Zhang, Hongyu Liu, Harry Yang, et al. Follow-your-shape: Shape-aware image editing via trajectory-guided region control. *arXiv preprint arXiv:2508.08134*, 2025. 3, 12

[44] Haoyu Lu, Wen Liu, Bo Zhang, Bingxuan Wang, Kai Dong, Bo Liu, et al. Deepseek-vl: Towards real-world vision-language understanding. *arXiv preprint arXiv:2403.05525*, 6, 2024. 3, 10

[45] Shiyin Lu, Yang Li, Yu Xia, Yuwei Hu, Shanshan Zhao, Yanqing Ma, et al. Ovis2.5 technical report. *arXiv:2508.11737*, 2025. 5, 6, 7, 3, 9, 10, 12

[46] Chaojie Mao, Jingfeng Zhang, Yulin Pan, Zeyinzi Jiang, Zhen Han, Yu Liu, et al. Ace++: Instruction-based image creation and editing via context-aware content filling. *arXiv preprint arXiv:2501.02487*, 2025. 2, 3, 12

[47] Qi Mao, Lan Chen, Yuchao Gu, Zhen Fang, and Mike Zheng Shou. 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)*, pages 6842–6850, 2024. 3, 2, 12

[48] Hyelin Nam, Gihyun Kwon, Geon Yeong Park, and Jong Chul Ye. Contrastive denoising score for text-guided latent diffusion image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9192–9201, 2024. 3, 2, 5, 12

[49] A. Novozámský, B. Mahdian, and S. Saic. Imd2020: A large-scale annotated dataset tailored for detecting manipulated images. *IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pages 71–80, 2020. 2, 3, 8

[50] Utkarsh Ojha, Yuheng Li, and Yong Jae Lee. Towards universal fake image detectors that generalize across generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24480–24489, 2023. 5, 8, 9, 12

[51] OpenAI. Chatgpt-4o: Advanced multimodal chat model, 2025. 4, 5, 6, 7, 9, 10, 12

[52] OpenAI. Gpt image 1: State-of-the-art image generation model, 2025. https://platform.openai.com/docs/models/gpt-image-1. 4, 8, 3

[53] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, et al. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 3

[54] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 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. 6

[55] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 2, 3

[56] Ronald Salloum, Yuzhuo Ren, and C.-C. Jay Kuo. Image splicing localization using a multi-task fully convolutional network (mfcn). *Journal of Visual Communication and Image Representation (JVCIR)*, 51:201–209, 2018. 3

[57] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, et al. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint*, 2025. 4, 8, 3, 6

[58] Haixu Song, Shiyu Huang, Yinpeng Dong, and Wei-Wei Tu. Robustness and generalizability of deepfake detection: A study with diffusion models. *arXiv preprint arXiv:2309.02218*, 2023. 3

[59] Zhen Sun, Ziyi Zhang, Zeren Luo, Zeyang Sha, Tianshuo Cong, Zheng Li, et al. Fragfake: A dataset for fine-grained detection of edited images with vision language models. *arXiv preprint arXiv:2505.15644*, 2025. 2, 3

[60] Chuangchuang Tan, Yao Zhao, Shikui Wei, Guanghua Gu, and Yunchao Wei. 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)*, pages 12105–12114, 2023. 5, 8, 9, 12

[61] Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming rectified flow for inversion and editing. *arXiv preprint arXiv:2411.04746*, 2024. 3, 2, 12

[62] Jiarui Wang, Huiyu Duan, Juntong Wang, Ziheng Jia, Woo Yi Yang, Xiaorong Zhu, Yu Zhao, Jiaying Qian, Yuke Xing, Guangtao Zhai, et al. Dfbench: Benchmarking deepfake image detection capability of large multimodal models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 12666–12673, 2025. 3

[63] Jin Wang, Chenghui Lv, Xian Li, Shichao Dong, Huadong Li, Kelu Yao, 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)*, pages 4233–4245, 2025. 3

[64] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. 3

[65] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. Cnn-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8695–8704, 2020. 5, 8, 9, 12, 13

[66] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, et al. Cogvlm: Visual expert for pretrained language models. 2023. 5, 9, 10, 12

[67] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. 3, 4, 5, 6, 7, 9, 10, 12

[68] Bihan Wen, Ye Zhu, Ramanathan Subramanian, Tian-Tsong Ng, Xuanjing Shen, and Stefan Winkler. Coverage—a novel database for copy-move forgery detection. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pages 161–165. IEEE, 2016. 3

[69] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025. 2, 3, 4, 5, 6, 12

[70] Chenyuan Wu, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, et al. Omnigen2: Exploration to advanced multimodal generation. *arXiv preprint arXiv:2506.18871*, 2025. 2, 3, 12

[71] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9192–9201, 2024. 3, 2, 6, 12

[72] Zitong Xu, Huiyu Duan, Bingnan Liu, Guangji Ma, Jiarui Wang, Liu Yang, Shiqi Gao, Xiaoyu Wang, Jia Wang, Xiongkuo Min, et al. Lmm4edit: Benchmarking and evaluating multimodal image editing with lmms. In *Proceedings of the 33rd ACM International Conference on Multimedia (ACM MM)*, pages 6908–6917, 2025. 3

[73] Zitong Xu, Huiyu Duan, Guangji Ma, Liu Yang, Jiarui Wang, Qingbo Wu, Xiongkuo Min, Guangtao Zhai, and Patrick Le Callet. Harmonyiqa: Pioneering benchmark and model for image harmonization quality assessment. *arXiv preprint arXiv:2501.01116*, 2025. 4

[74] Zhipei Xu, Xuanyu Zhang, Runyi Li, Zecheng Tang, Qing Huang, and Jian Zhang. Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. 2, 3, 5, 8, 9, 12

[75] Shilin Yan, Ouxiang Li, Jiayin Cai, Yanbin Hao, Xiaolong Jiang, Yao Hu, et al. A sanity check for ai-generated image detection. *arXiv preprint arXiv:2406.19435*, 2024. 5, 8, 9, 12, 13

[76] Zhiyuan Yan, Yong Zhang, Xinhang Yuan, Siwei Lyu, and Baoyuan Wu. Deepfakebench: A comprehensive benchmark of deepfake detection. In *Proceedings of the Advances*

in Neural Information Processing Systems (NeurIPS), pages 4534–4565, 2023. 2, 3, 8

[77] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 3, 5, 6, 7, 9, 10, 12

[78] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024. 5, 9, 12

[79] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, 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. 5, 9, 10, 12

[80] Zeqin Yu, Jiangqun Ni, Yuzhen Lin, Haoyi Deng, and Bin Li. Diffforensics: Leveraging diffusion prior to image forgery detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12765–12774, 2024. 3

[81] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3, 2, 5, 12

[82] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025. 5, 6, 7, 4, 9, 10, 12

[83] Xinshan Zhu, Yongjun Qian, Xianfeng Zhao, Biao Sun, and Ya Sun. A deep learning approach to patch-based image inpainting forensics. *Signal Processing: Image Communication (SPIC)*, 67:90–99, 2018. 3

[84] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 6

[85] Junhao Zhuang, Yanhong Zeng, Wenran Liu, Chun Yuan, and Kai Chen. A task is worth one word: Learning with task prompts for high-quality versatile image inpainting. In *European Conference on Computer Vision (ECCV)*, pages 195–211, Cham, 2024. Springer Nature Switzerland. 3, 2, 6, 12

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

Supplementary Material

1. Overview

This supplementary material provides additional information on the data collection, methodology, experiments, and results discussed in the main paper. Section 2 describes the 12 distinct tasks involved in the prompt and image collection process, while Section 3 presents an overview of the 25 image manipulation models. Section 4 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.

2. 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.
- **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.

3. Details of Manipulation Methods

- **IP2P [6]** 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 [48]** 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 [81]** 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 [29]** 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 [35]** 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 [71]** 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 [47]** 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[36]** 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[85]** 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 [17]** 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 [28]** 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 [26]** 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 [61]** 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 [31]** 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) [15]** extends FlowEdit to the FLUX backbone, combining flow-based latent editing with FLUX's expressive diffusion space for precise, high-fidelity global edits.
- **ACE++ [46]** 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 [70]** 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 [30]** 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 [3]** 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.

Table 9. An overview of AI-editing models in our ManipBench

Models	Time	Prompt Type	Method	URL
IP2P [6]	2023.04	Instruction	SD1.4 [55]	https://github.com/timothybrooks/instruct-pix2pix
CDS [48]	2023.11	Description	SD1.4 [55]	https://github.com/HyelinNAM/ContrastiveDenoisingScore
Magicbrush [81]	2023.06	Instruction	SD1.4	https://github.com/OSU-NLP-Group/MagicBrush
PnP [29]	2023.10	Description	SD1.5 [55]	https://github.com/MichalGeyer/plug-and-play
Any2Pix [35]	2023.12	Instruction	SDXL [53]	https://github.com/jacklishufan/InstructAny2Pix
InfEdit [71]	2023.12	Description	SD1.4 [55]	https://github.com/sled-group/InfEdit
MAG [47]	2023.12	Description	SD1.4 [55]	https://github.com/HelenMao/MAG-Edit
ZONE [36]	2023.12	Instruction	SD1.5 [55]	https://github.com/lsl001006/ZONE
PowerPaint [85]	2023.12	Description	SD1.5 [55]	https://github.com/open-mmlab/PowerPaint
ReNoise [17]	2024.03	Description	SDXL [53]	https://github.com/garibida/ReNoise-Inversion
BrushNet [28]	2024.03	Description	SD1.5 [55]	https://github.com/TencentARC/BrushNet
HQEdit [26]	2024.04	Instruction	SD1.4 [55]	https://github.com/UCSC-VLAA/HQ-Edit
RFSE [61]	2024.11	Description	FLUX [15]	https://github.com/wangjiangshan0725/RF-Solver-Edit
FlowEdit (SD3) [31]	2024.12	Description	SD3 [55]	https://github.com/fallenshock/FlowEdit
FlowEdit (FLUX) [31]	2024.12	Description	FLUX [15]	https://github.com/fallenshock/FlowEdit
ACE++ [46]	2025.01	Instruction	FLUX [15]	https://github.com/ali-vilab/ACE_plus
OmniGen2 [70]	2025.01	Instruction	DiT [34]	https://github.com/VectorSpaceLab/OmniGen2
Reflex [30]	2025.07	Description	FLUX [15]	https://github.com/wlaud1001/ReFlex
OT-Inversion [3]	2025.08	Description	FLUX [15]	https://github.com/marianlupascu/OT-Inversion
Follow-Your-Shape [43]	2025.08	Description	FLUX [15]	https://github.com/mayuelala/FollowYourShape
Qwen-Image-Edit [69]	2025.08	Instruction	DiT [34]	https://github.com/QwenLM/Qwen-Image
GPT-Image [52]	2025.04	Instruction	N/A	https://chatgpt.com/
FLUX-Kontext [32]	2025.06	Instruction	FLUX [15]	https://github.com/black-forest-labs/flux
NanoBanana [1]	2025.07	Instruction	N/A	https://ainanobanana.io/
SeedDream4 [57]	2025.09	Instruction	N/A	https://dreamina.capcut.com/

- **Follow-Your-Shape [43]** 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 [69]** 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.
- **GPT-Image [52]** 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 [32]** 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 [9]** 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 [57]** 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.
- **LLaVA-1.6 (7B) [39]** is a vision-language model that extends LLaMA-2 with a CLIP-based visual encoder and instruction-tuned alignment. In deepfake detection, it enables joint visual–textual reasoning to identify semantic inconsistencies (e.g., mismatched facial expressions or contextually implausible scenes). Its open-source nature allows fine-tuning for forensic detection, though its low-level visual sensitivity remains limited.
- **Ovis 2.5 (9B) [45]** integrates hierarchical cross-attention for enhanced visual grounding, making it suitable for detecting deepfakes involving object insertion or scene manipulation. Its balanced architecture yields high interpretability and moderate computational cost. However, its mid-scale capacity constrains its ability to detect extremely subtle diffusion artifacts.
- **DeepSeek-VL2 (small) [44]** is a compact multimodal variant trained for efficient image–text understanding. In the context of fake content detection, it can perform reasoning-based verification between textual descriptions and visual evidence. Its low parameter size provides speed advantages but leads to reduced robustness under visually complex manipulations.
- **InternVL3.5 (8B) [67]** developed by OpenGVLab, introduces deep fusion between visual and textual representations, enabling multi-image reasoning for consistency-based fake detection (e.g., comparing two versions of a portrait). It achieves high accuracy in detecting semantic tampering, though inference cost remains non-trivial.
- **Qwen3-VL (8B) [64]** is part of Alibaba's Qwen-VL fam-

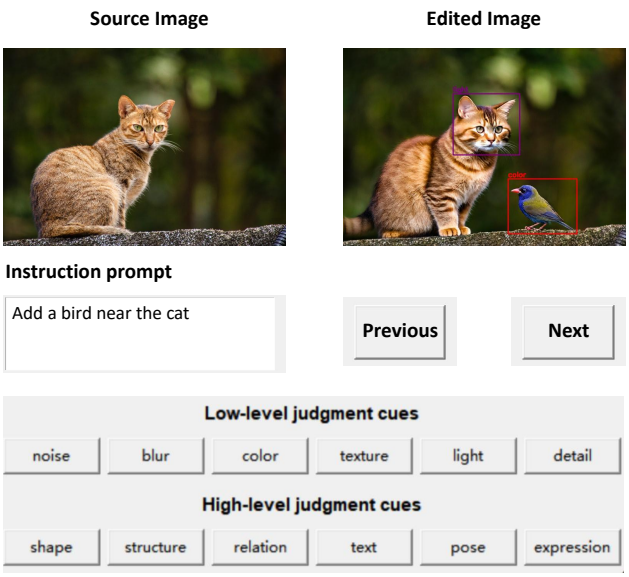

Figure 7. An example of the annotation interface.

ily that combines DiT-based visual encoding with an instruction-tuned LLM. It exhibits strong grounding and context reasoning, enabling detection of visual-textual mismatches often seen in AI-generated fakes. The model performs robustly in zero-shot forensic tasks but may hallucinate under ambiguous instructions.

- **Gemini 2.5-Pro [9]** is Google's proprietary multimodal foundation model integrating OCR, spatial reasoning, and world-knowledge grounding. In deepfake detection, it achieves exceptional generalization to unseen fake types via large-scale pretraining. However, its closed-source nature prevents controlled evaluation and domain-specific tuning.
- **ChatGPT-4o [51]** (OpenAI's multimodal foundation model) provides strong visual reasoning and linguistic grounding across a wide range of manipulation types. It effectively identifies semantic inconsistencies between text and images (e.g., mismatched lighting, impossible geometry). Yet, as a closed system, it lacks explainable outputs and forensic interpretability.
- **InternVL3 (8B) [82]** is an optimized version of InternVL3.5 with additional cross-modal attention supervision and image–text contradiction fine-tuning. Its variant enhances sensitivity to semantic forgeries and inconsistent composites but increases training complexity and memory requirements.

4. Details of Manual Annotation Process

To ensure a comprehensive and efficient image quality evaluation, we developed a custom annotation interface, as illustrated in Figure 7. 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 visually modified regions in the manipulated image. For each annotated region, annotators also specify the judgment cues they relied on to 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.

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.

A total of 20 trained annotators participate in the study 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. Before the formal annotation begins, all participants undergo a brief training phase, during which they are introduced to the evaluation criteria and perform several practice trials to familiarize themselves with the interface and scoring guidelines.

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. All annotation data are automatically logged and verified for completeness and response validity before statistical aggregation.

To ensure annotation reliability and consistency, we adopt a three-stage annotation protocol that integrates independent labeling, cross-verification, and expert arbitration.

- **Independent labeling:** Each annotator independently labels the manipulated regions and corresponding judgment cues without reference to others' annotations, ensuring unbiased initial results.
- **Cross-verification:** The annotations are then reviewed in pairs, where each annotator checks another's work to identify potential omissions, overlaps, or inconsistencies. This mutual verification enhances completeness and accuracy.
- **Independent labeling** Finally, discrepancies or ambiguous cases are resolved through expert arbitration conducted by senior researchers with experience in image processing studies. The experts consolidate the final annotations, ensuring the correctness across the dataset.

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.

5. More Data Analysis of ManipBench

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 [6], CDS [48], and MagicBrush [81] 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 [69] 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 [85], ReNoise [17], and InfEdit [71] 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 [69], FLUX-Kontext [32], and SeedDream4 [57] exhibit consistently higher SI values—sometimes even exceeding those of real images, indicating stronger preservation of fine details and local texture fidelity.

6. Details of Training Process

6.1. 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.
- **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.

6.2. Details of Layer Discrimination Selection

We combine KL divergence, LDR, and information entropy to select the optimal layer. Figure 8 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.

6.3. 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 for-

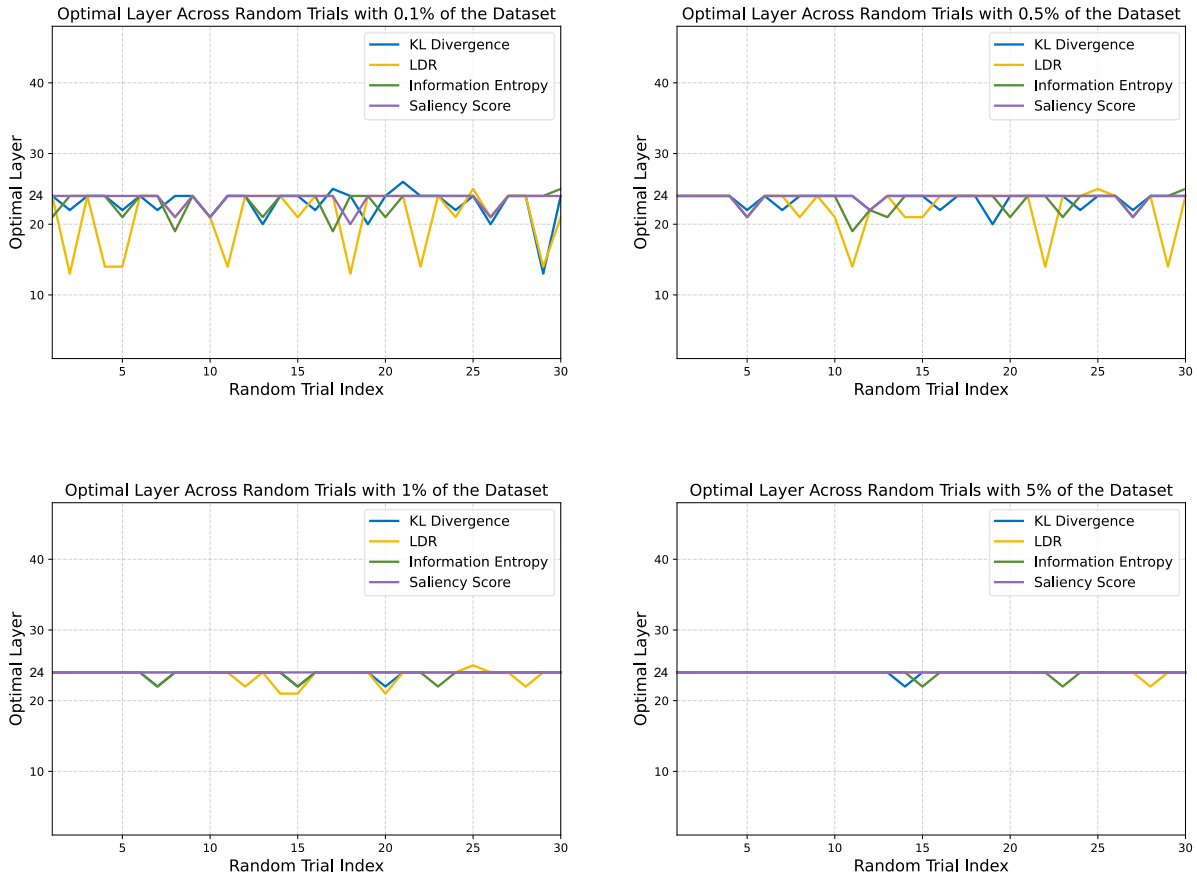

Figure 8. Optimal layer variation across random trials for KL divergence, LDR, information entropy, and saliency score under different sample scales.

mulated 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.

7. Model Comparison Details

7.1. Details of AI-generated Images Detection Methods

- **Univ [50]** 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 [75]** 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 [65]** 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 [60]** 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.

7.2. Details of Image Manipulation Detection Methods

- **HiFiNet [20]** 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 [74]** 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 [12]** 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 [41]** 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.

7.3. Details of Image Manipulation Localization Methods

7.4. Details of Multimodal Large Language Models

- **LLaMA-3.2-Vision (11B) [2]** 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

Table 10. 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 [20]	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 [74]	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 [50]	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 [75]	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) [2]	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) [39]	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) [78]	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) [79]	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) [66]	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) [45]	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) [11]	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) [82]	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) [67]	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) [4]	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) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	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∗ [50]	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∗ [75]	**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)∗ [11]	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)∗ [82]	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)∗ [77]	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 [20]	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 [74]	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 [50]	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 [75]	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) [2]	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) [39]	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) [78]	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) [79]	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) [66]	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) [45]	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) [11]	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) [82]	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) [67]	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) [4]	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) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	**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∗ [50]	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∗ [75]	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)∗ [11]	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)∗ [82]	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)∗ [77]	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**

subtle editing cues, though its performance can degrade under domain shifts or adversarial manipulations.

- **LLaVA-1.6 (7B) [39]** 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) [78]** 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) [79]** 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) [66]** 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) [45]** 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 [44]** 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 [67, 82]** 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 [4, 77]** 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 [9]** 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 [51]** 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.

7.5. 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 10 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 11 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 12 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.

Table 11. 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 [6]		CDS [48]		MagicBrush [81]		InfEdit [71]		MAG [47]		HQEdit [26]		PnP [29]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [2]	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) [39]	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) [78]	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) [79]	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) [66]	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) [45]	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) [11]	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) [82]	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) [67]	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) [4]	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) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	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∗ [50]	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∗ [75]	**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 [36]		PowerPaint [85]		BrushNet [28]		Any2Pix [35]		ReNoise [17]		FlowEdit(SD3) [31]		RFSE [61]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [2]	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) [39]	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) [78]	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) [79]	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) [66]	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) [45]	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) [11]	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) [82]	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) [67]	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) [4]	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) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	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∗ [50]	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∗ [75]	**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) [31]		ACE [46]		Reflex [30]		OT [3]		Follow-Your-Shape [43]		OmniGen2 [70]		Qwen-Image-Edit [69]	
	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑	Acc↑	F1↑
♢LLama3.2-Vision (11B) [2]	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) [39]	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) [78]	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) [79]	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) [66]	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) [45]	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) [11]	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) [82]	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) [67]	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) [4]	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) [77]	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 [9]	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 [51]	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∗ [21]	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∗ [42]	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∗ [12]	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∗ [41]	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∗ [20]	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∗ [74]	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∗ [65]	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∗ [60]	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∗ [50]	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∗ [75]	**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 12. Accuracy results (Acc↑) of cross-validation on different training and testing subsets.

ResNet50 [21]	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 [42]	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 [65]	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 [75]	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 [12]	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 [20]	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

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

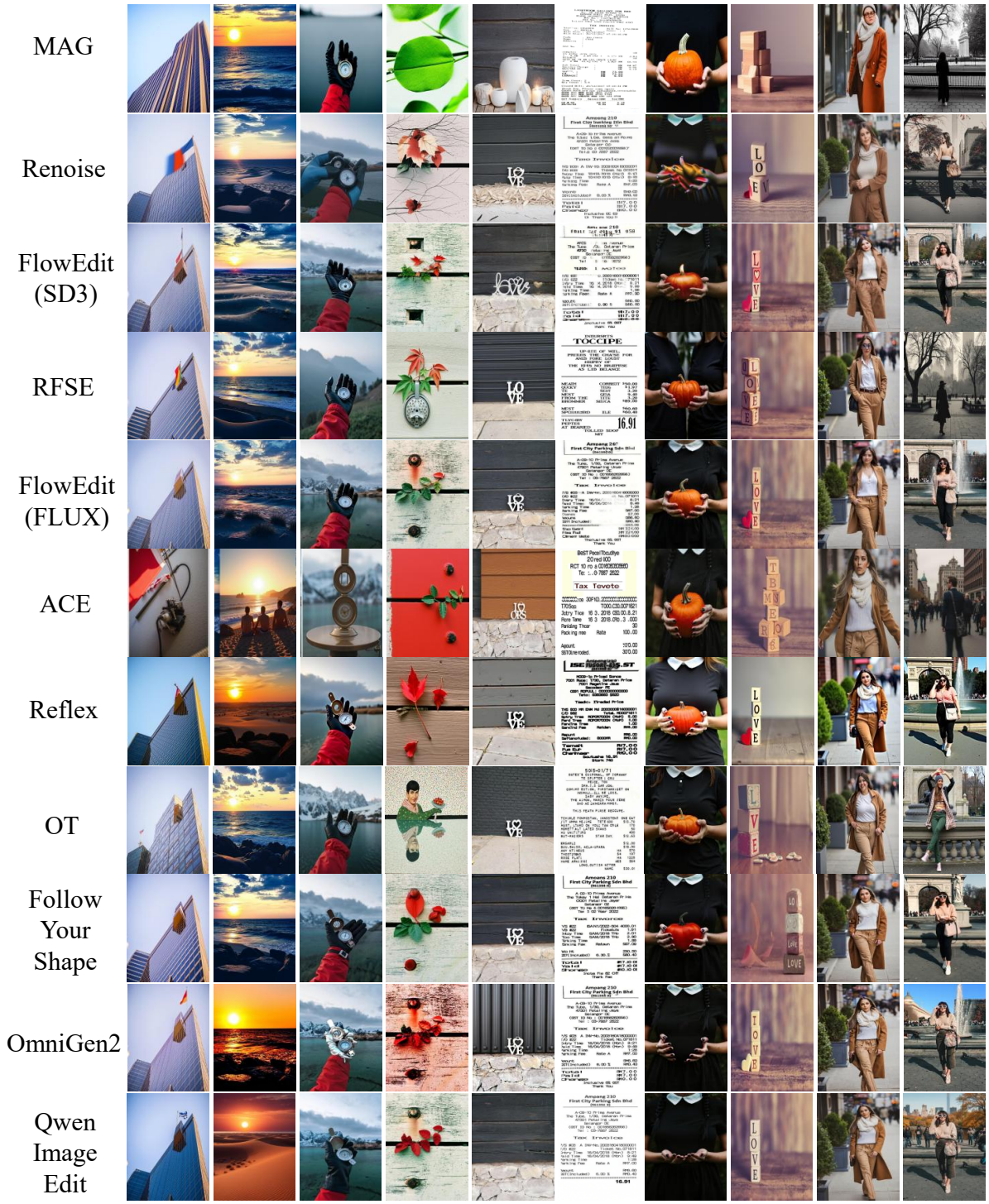

Figure 10. Example images generated by different manipulation models in ManipBench.

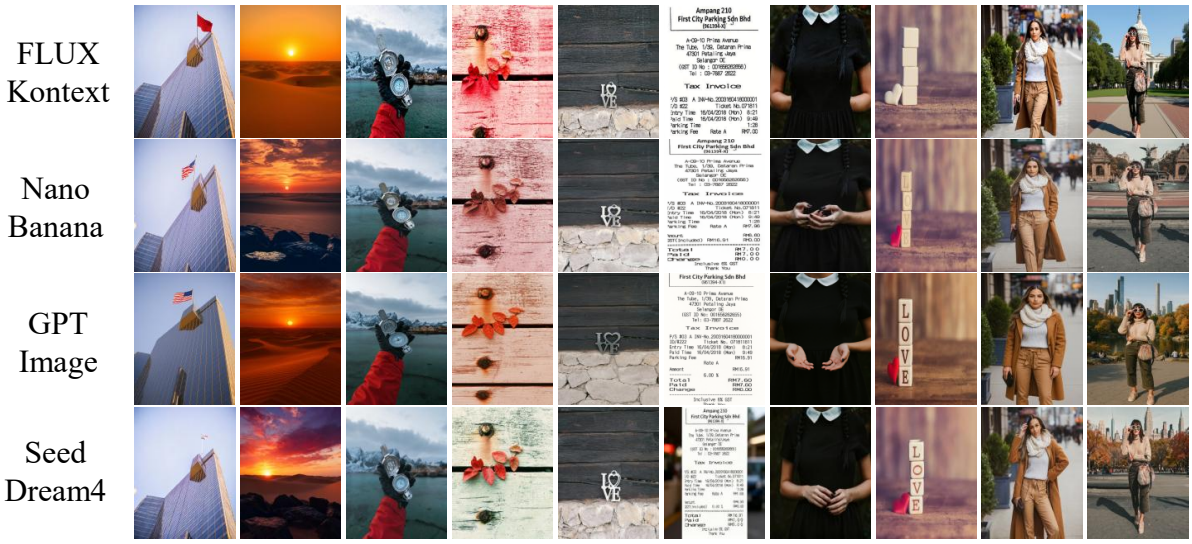

Figure 11. Example images generated by different manipulation models in ManipBench.

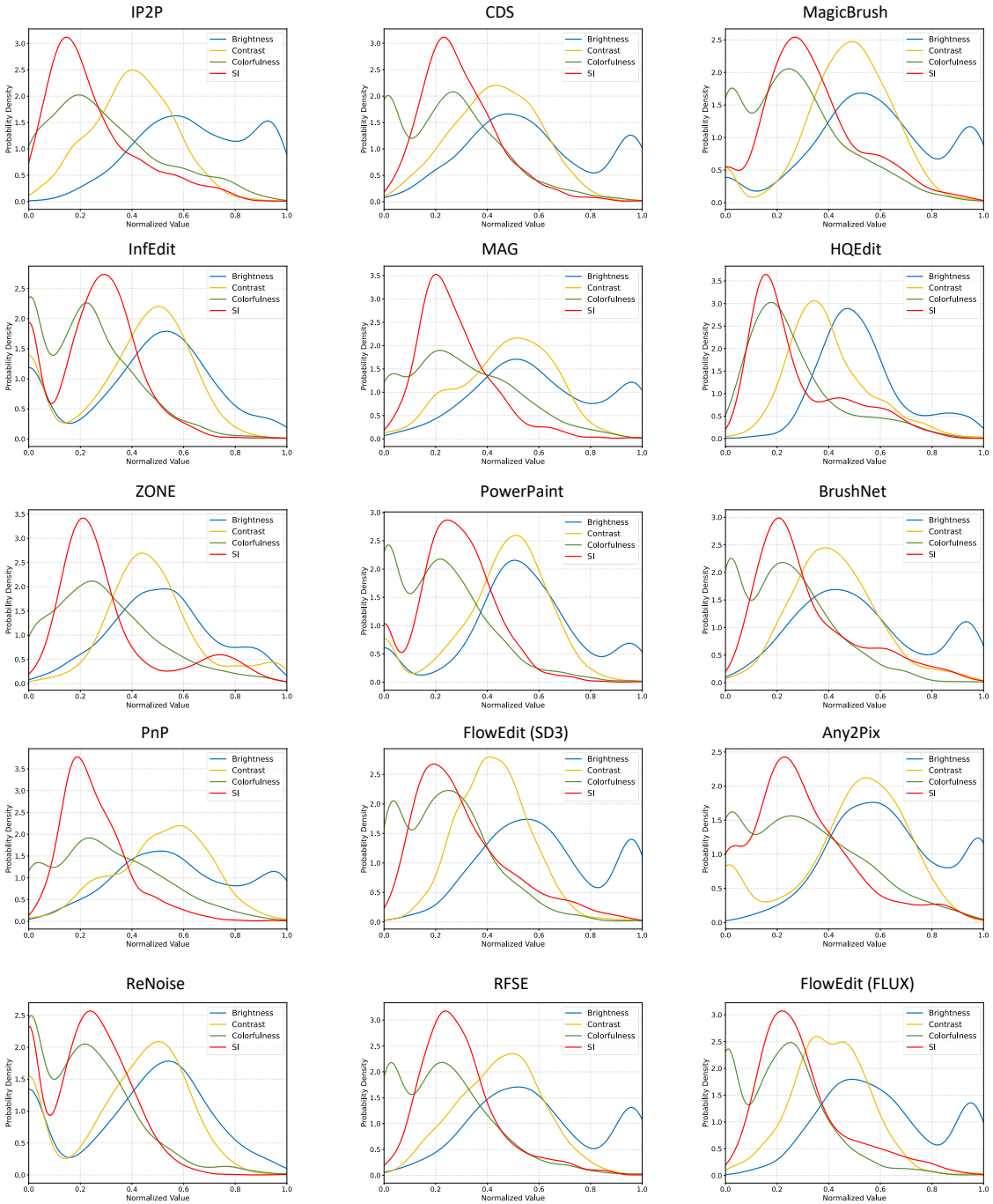

Figure 12. Feature distribution of images generated by different manipulation models.

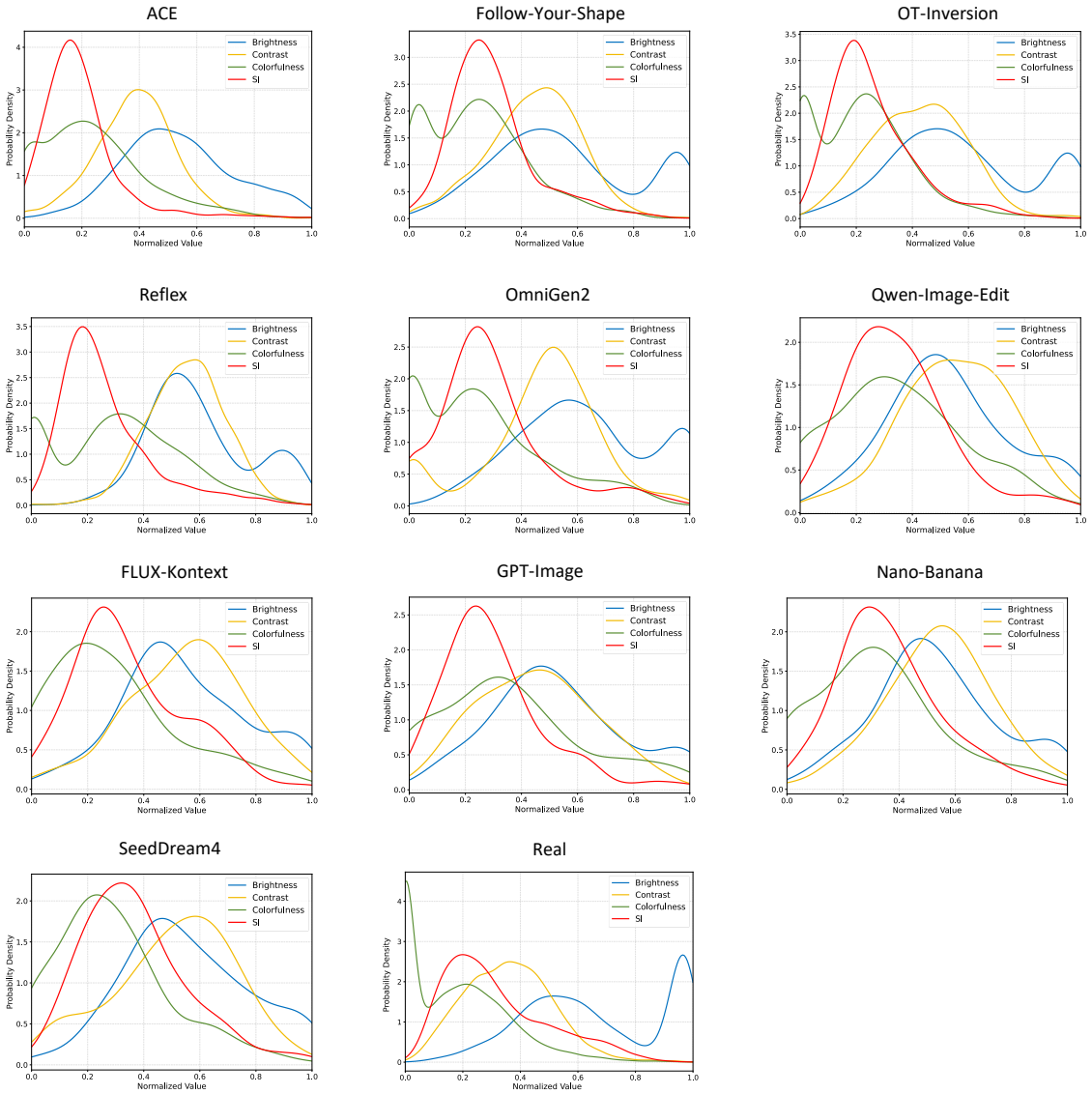

Figure 13. Feature distribution of images generated by different manipulation models and real images.