What Changed? Detecting and Evaluating Instruction-Guided Image Edits with Multimodal Large Language Models

Lorenzo Baraldi*[2], Davide Bucciarelli*[1,2], Federico Betti*[3],
Marcella Cornia[1], Lorenzo Baraldi[1], Nicu Sebe[3], Rita Cucchiara[1,4]
[1]University of Modena and Reggio Emilia, Italy [2]University of Pisa, Italy
[3]University of Trento, Italy [4]IIT-CNR, Italy
[1]{name.surname}@unimore.it, [2]{name.surname}@phd.unipi.it, [3]{name.surname}@unitn.it

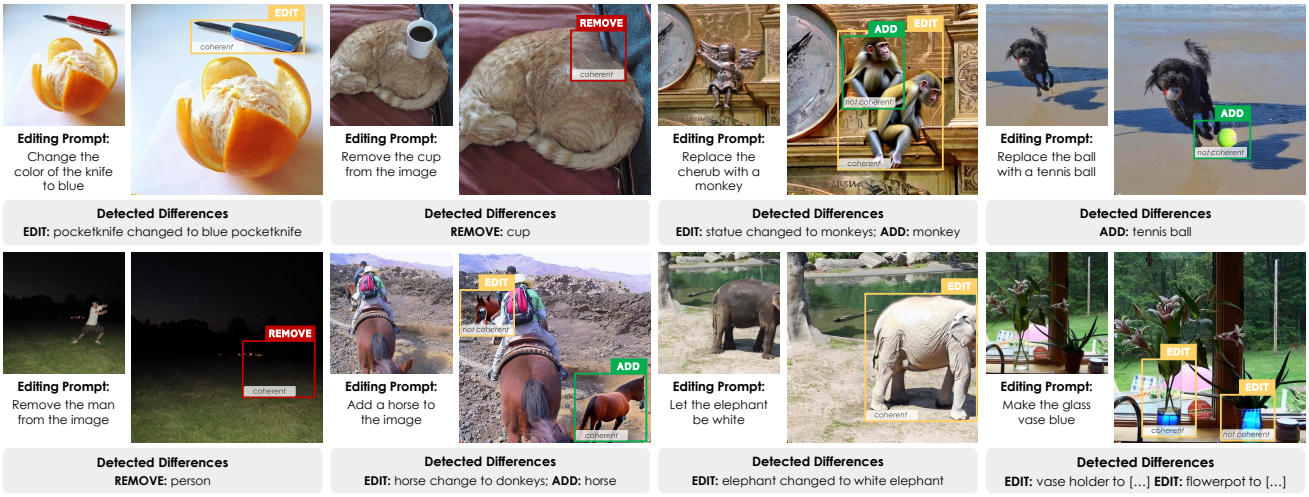

Figure 1. Qualitative examples from DICE. Our approach detects differences between an original image and an edited one, identifying the involved objects and the type of edit. Further, DICE evaluates each difference to determine its coherence with the editing prompt.

Abstract

Instruction-based image editing models offer increased personalization opportunities in generative tasks. However, properly evaluating their results is challenging, and most of the existing metrics lag in terms of alignment with human judgment and explainability. To tackle these issues, we introduce ***DICE*** *(**DI**fference **C**oherence **E**stimator), a model designed to detect localized differences between the original and the edited image and to assess their relevance to the given modification request. DICE consists of two key components: a difference detector and a coherence estimator, both built on an autoregressive Multimodal Large Language Model (MLLM) and trained using a strategy that leverages self-supervision, distillation from inpainting networks, and full supervision. Through extensive experiments, we evaluate each stage of our pipeline, comparing different MLLMs within the proposed framework. We demonstrate that DICE effectively identifies coherent edits, effectively evaluating images generated by different editing models with a strong correlation with human judgment. We publicly release our source code, models, and data at project page.*

1. Introduction

The field of Generative AI has recently gained significant attention in both industry and research, driven by the development of models capable of generating content that closely follows the distribution of natural language [1, 14, 21] and real images [15, 38, 41]. Thanks to the availability of cross-modal operators and architectures, multimodal generative approaches, which can seamlessly integrate both the vision and language modalities, are reaching increasing levels of accuracy [5, 32, 46, 49]. These advancements are driving the development of new applications and tasks while enhancing user control over the generation process.

Among these proposals, instruction-based image editing models [3, 18, 25, 51] enable the modification of an input image based on a free-form textual instruction from the user. This extends traditional text-to-image models, shifting the focus from generating an image from scratch to modifying an existing one while adhering to a given prompt. While

*Equal contribution.

such models promise to offer increased personalization levels, the understanding of the results they generate – and thus their proper evaluation – clearly becomes more challenging than in traditional text-to-image contexts.

Prior work has approached the evaluation of image editing models by developing annotated benchmarks with ground-truth edited images [43, 44, 51] or by proposing metrics based on pre-trained backbones such as CLIP [39] and DINO [7]. Other methods leverage pre-trained large language models to assess the quality of image edits [2, 27, 35]. While these proposals might offer a first viable solution to assess the performance of image editing models, they still do not reach a sufficient alignment with human evaluation. Further, being based on either global embedding vectors or replies from pre-trained (and often private) models, explaining their evaluations becomes cumbersome.

Drawing inspiration from these considerations, we take a novel approach and propose a learnable model for understanding image editing outputs, which can provide increased alignment with human preferences and more easily explainable results. Our approach starts from the hypothesis that modifications made by image editing models typically occur within localized regions of the input image. Accordingly, our model first identifies object-level differences between the original and the edited images and subsequently evaluates the coherence of each detected modification relative to the user's input instruction. We term our pipeline as **DICE**, short for **DI**fference **C**oherence **E**stimator. Sample results from our model are shown in Fig. 1.

Maintaining the goal of creating a fully explainable pipeline, we develop two distinct models, one for each stage, both built on an autoregressive Multimodal Large Language Model (MLLM) [5]. While both models leverage the understanding of RoIs, the first primarily focuses on their prediction, whereas the second emphasizes their interpretation to assess coherence with the given prompt. In particular, we first propose a difference detector that, given both images, estimates a set of localized differences which describe the change in content from one image to the other. The model also predicts the category of detected modifications, as either addition, removal, or editing of an object. From a technical standpoint, we train the model in a two-stage protocol, where we first teach the model to identify object-level differences in a self-supervised manner, leveraging pairs of similar images, and then fine-tune it for the specific task using inpainted images. A second model, based on the same architecture, is instead trained to predict the coherence of an object-level modification with respect to the modification request made by the user and provide a textual rationale for each decision made.

We experimentally assess the appropriateness of our strategy by carefully investigating the performance of both our difference detector and our coherence estimator, in comparison with existing solutions and when employing different MLLMs. Further, we conduct a dedicated user study to measure the alignment of the evaluations predicted by our proposal in terms of model ranking and evaluation. Experimental results demonstrate that our proposal achieves increased ranking capabilities while benefiting from an explainable-by-default approach. Further, when integrated in existing metrics, it acts as the basis for building model evaluation metrics with increased human alignment.

2. Related Work

Image Editing Models. Recent developments in Generative Adversarial Networks (GANs) [20, 28] and diffusion models [13, 15, 38, 41] have driven the rise of AI-generated content. However, users increasingly demand not only the creation of new visual data but also the modification of existing images, altering properties such as style [17, 28] and content [3, 16]. Within this domain, tasks vary based on the input used for conditioning. Image inpainting [10, 45], for instance, uses user-designed masks to guide edits in specific regions. In contrast, our work focuses on instruction-based image editing, where the model relies solely on textual instructions and the original image.

Within instruction-based editing, the most traditional approaches rely on diffusion models fine-tuned for instruction following. A fundamental contribution in this domain is InstructPix2Pix [3] which is trained on fully synthetic triplets composed of an input image, an editing instruction, and the resulting edited image. However, reliance on synthetic data may introduce noise into the predictions. To mitigate this limitation, MagicBrush [51] refines the InstructPix2Pix framework using a high-quality, curated dataset. Differently, InstructDiffusion [18] is a Stable Diffusion-based model that is trained to perform a wide variety of tasks including image editing, semantic segmentation, and object detection. Further, HIVE [52] proposes a reward-based training procedure that incorporates human feedback to enhance its instruction following capabilities.

A complementary research direction involves integrating an MLLM to improve language understanding within the editing process. In this regard, MGIE [16] employs a frozen MLLM to generate concise and representative edit instructions that guide the diffusion model during image editing, whereas SmartEdit [26] captures the interactions between image and text by extracting salient information from the image tokens produced by an MLLM.

Image Editing Evaluation and Benchmarks. Instruction-based image editing models have traditionally been evaluated along two dimensions: prompt adherence and background preservation. The former assesses the extent to which the output aligns with the given prompt instruction, whereas the latter evaluates the preservation of the unmodified contextual information. While benchmarks

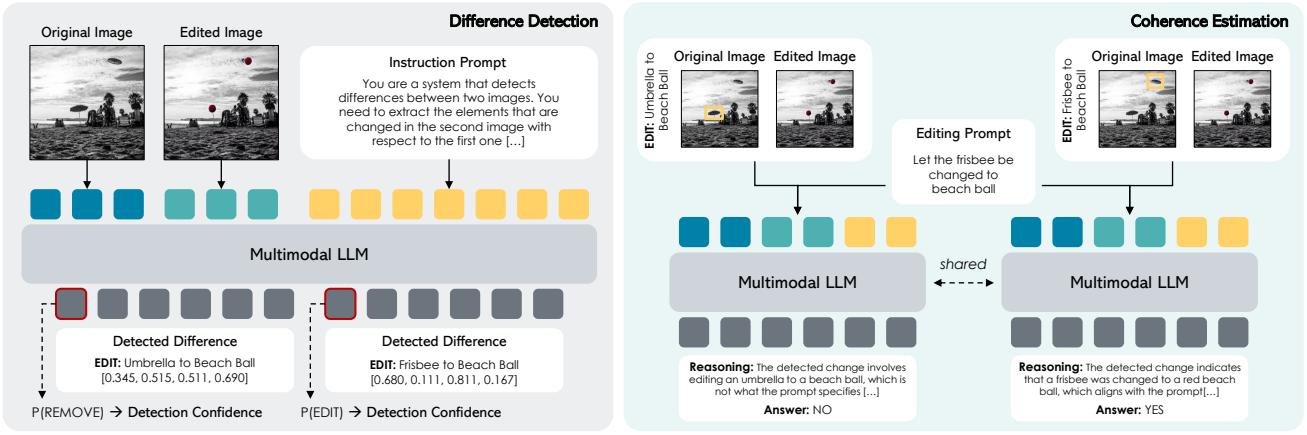

Figure 2. Illustration of DICE. We employ an MLLM and fine-tune it for two different tasks. In the first stage (difference detection), the MLLM is trained to detect semantic differences between the original image and the edited one. In the second stage (coherence estimation), the MLLM is instructed to analyze and assess the coherence of each detected difference with respect to the given user prompt.

featuring annotated ground-truth edited images are available [43, 44, 51], our work prioritizes a reference-free evaluation that eliminates the need for human-defined labels. In this setting, background preservation is typically evaluated by using either CLIP [39] or DINO [7] to compute cosine similarities between the visual tokens extracted from the source and edited images [3, 16, 52]. Likewise, prompt adherence is usually quantified using CLIP-based similarities between the target caption and the edited image [42, 51].

With the recent proposal of MLLMs, these methods have been employed for prompt adherence evaluation. For instance, in HQ-Edit [27], GPT-4V [1] is used to assess coherence and alignment metrics. Additionally, I^2E Bench [35] incorporates human annotation for certain queries, subsequently leveraging GPT-4V for question answering. However, reliance on GPT-based evaluations introduces potential variability and reproducibility issues, as model updates may alter the final scores. In contrast, we propose an interpretable solution based on an open-source MLLM, thereby increasing both the rationale behind our scoring mechanism and the reproducibility of the results.

Multimodal Large Language Models. With the introduction of LLMs, the research community has directed great effort towards connecting visual modalities with textual reasoning [4, 11, 32]. Building on these efforts, MLLMs have also been used to address visual grounding tasks. Among these, Shikra [9] leverages a novel training paradigm to give the MLLM the capacity to reason with bounding boxes, making it able to precisely locate objects and relations in the image without relying on external components. LION [8] improves spatial awareness through progressive integration of spatial knowledge and soft prompting. Groma [34] decomposes images into region tokens, enhancing localization via user instructions. In contrast, LISA [29] and GROUNDHOG [53] perform segmentation incorporating pixel-level representations for fine-grained visual understanding.

Compared to existing MLLMs for visual grounding, our model leverages multiple images to detect semantic differences, introducing a novel setting that requires an MLLM capable of multi-image understanding. In this context, Idefics3 [30] is able to comprehend multiple visual inputs interleaved with text, using a novel image encoding pipeline. Differently, Qwen [46] extends classical multimodal capabilities to video and multi-image understanding thanks to a novel encoding strategy and positional embedding technique for the visual inputs that takes into account their spatial dimensions. Finally, mPLUG-Owl3 [49] introduces a novel attention-based strategy to efficiently encode long image sequences. Building on these multi-image models, we extend the capabilities of MLLMs by introducing a novel framework for multi-image detection, enabling the identification of semantic differences across multiple images within a unified system.

3. Proposed Method

3.1. Preliminaries

The objective of our approach is to assess the quality of outputs generated by instruction-based image editing models [3, 16, 52]. Formally, given an original image $\boldsymbol{x}$ and a textual instruction $\boldsymbol{t}$, an image editing method f generates a new image $\boldsymbol{e} = f(\boldsymbol{x}, \boldsymbol{t})$, which incorporates the modifications specified by the textual prompt $\boldsymbol{t}$, while preserving the contextual integrity of $\boldsymbol{x}$.

While the modifications introduced by image editing models can be diverse – owing to the expressiveness of textual instructions and the flexibility of generative models – we focus specifically on object-level modifications. By means of this choice, we consider transformations that are clearly localizable inside of the input scene. In particular, we assume that the original image contains a finite set of

objects, denoted as $\{o_i\}_{i=1}^n$, and that the textual instruction $\boldsymbol{t}$ specifies an operation to be applied to it. Given a correct application of the specified operation, the expected edited image, denoted as $\tilde{\boldsymbol{e}}$, serves as the ground-truth for evaluating the intended modification.

Object-level Modifications. Without loss of generality, we assume that the alterations required in the textual prompt can be classified as either the addition, removal, or modification of objects inside the input image. Specifically, we consider three categories of alterations, namely:

1. **Addition of an object.** A new object instance o_{n+1} is introduced into the scene, yielding a new image representation. Treating images as a set of objects, and with a slight abuse of notation, this operation results in a new image that can be expressed as $\tilde{e} = (\bigcup_{i=1}^{n} o_i) \cup \{o_{n+1}\}$.
2. **Removal of an object.** The operation removes an object o_j which was present in the original image, which results in $\tilde{e} = \left(\bigcup_{i=1}^{j-1} o_i \cup \bigcup_{i=j+1}^{n} o_i\right)$.
3. **Edit of an object.** An existing object o_j is altered in at least one visual aspect. Let $\tilde{o}_j$ denote the modified object; the updated image is then represented as $\tilde{e} = (\bigcup_{i=1}^{n} o_i \setminus \{o_j\}) \cup \{\tilde{o}_j\}$.

3.2. Overview of the Approach

An image editing network can produce an image $\boldsymbol{e}$ that does not accurately respect all the object-level operations required to produce $\tilde{e}$. To quantify this discrepancy, we propose **DI**fference **C**oherence **E**stimator (**DICE**): a two-step approach in which (i) we detect the object-level differences between the edited and the original image and then (ii) we evaluate the coherence of each difference with the textual prompt to determine whether a discrepancy reflects the intended modifications or constitutes an unwanted alteration. An overview of the two stages of DICE is depicted in Fig. 2.

Difference Detection. In this stage, we extract the set of objects that are present in the original image $\boldsymbol{x}$ but absent or changed in the predicted image $\boldsymbol{e}$ or vice versa. We denote this collection of differences as *semantic difference*, which is formally defined as

$$\Delta(\boldsymbol{x}, \boldsymbol{e}) = \{o \mid o \in \{\boldsymbol{x} \setminus \boldsymbol{e}\} \cup \{\boldsymbol{e} \setminus \boldsymbol{x}\}\}. \tag{1}$$

The aforementioned set of differences is estimated through a learnable difference detector $D(\boldsymbol{x}, \boldsymbol{e})$ that, given the original and modified image, predicts a set of object-level differences which serves as an approximation of $\Delta(\boldsymbol{x}, \boldsymbol{e})$, expressed in terms of RoIs. Notably, this detection process is executed independently of the textual prompt $\boldsymbol{t}$ to ensure that the outcome is not biased by the intended edit.

Coherence Estimation. In the second step, we evaluate each detected difference individually to assess its coherence to the textual prompt. Specifically, a coherence estimator operates on each detected object-level change $o_i \in D(\boldsymbol{x}, \boldsymbol{e})$, determining whether the visual modifications align with the intended edit $\tilde{\boldsymbol{e}}$. Formally,

$$C(\boldsymbol{x}, \boldsymbol{e}, \boldsymbol{t}, o_i) = \begin{cases} 1 & \text{if } o_i \in \tilde{\boldsymbol{e}}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

This process enables the differentiation between intended and unintended edits, thereby facilitating a comprehensive and interpretable evaluation of the task.

3.3. Detecting Differences

To build the difference detector and the coherence estimator, we design two novel architectures building upon the general structure of autoregressive Multimodal Large Language Models (MLLMs), which offer a sufficiently general framework for encoding images, prompts, and RoIs.

For difference detection, we train an MLLM $D(\mathbf{x}, \mathbf{e})$ to predict object-level differences between two images in terms of (i) the type of modification (*i.e.*, addition, removal, edit), (ii) the RoI of the modified region, and (iii) the subjects involved in the modification. Formally, the difference detector operates as:

$$D(\boldsymbol{x}, \boldsymbol{e}) = [(c_i, S_i, b_i)]_{i=1}^{k}, \tag{3}$$

where c_i denotes a command from $\{\texttt{ADD}, \texttt{EDIT}, \texttt{REMOVE}\}$, S_i describes the modified object in free text, and $b_i \in \mathbb{R}^4$ defines the bounding box coordinates in the format $(x_{\text{min}}, y_{\text{min}}, x_{\text{max}}, y_{\text{max}})$.

Each tuple (c_i, S_i, b_i) is fully specified in the textual domain, and the model is optimized to output structured text representations of modifications, formatted as: `"COMMAND: object, [bb1, bb2, bb3, bb4]"`. This structured output ensures precise localization and identification of changes.

To estimate the confidence of a predicted RoI, we compute the probability of the predicted command c_i relative to the total probability mass of all possible commands. This confidence measure aligns with conventional object detection confidence estimation techniques [6, 19].

Training Procedure Overview. Our training pipeline consists of two phases: (i) a pre-training stage leveraging real image pairs and (ii) a fine-tuning stage using images modified with inpainting models. Notably, the model is intentionally not trained on instruction-based paired edited images due to two key considerations. First, such training may lead to overfitting to specific editing architectures while underperforming on unseen methods. Second, instruction-edited image pairs would necessitate extensive human annotations for commands and RoIs, introducing significant cost and time constraints.

Stage 1: Learning to Compare Similar Images. In this stage, we instruct the difference detector to identify object-level differences when considering pairs of similar real images. Specifically, the MLLM is trained to predict objects

that are not mutually present in both images, leveraging an existing object detection dataset.

In detail, we employ images and object-level annotations from the LVIS dataset [22]. We identify pairs of similar images by computing the cosine similarity in the DINOv2 [37] embedding space, and retaining only those with a cosine similarity greater than 0.6, ensuring high visual similarity. In addition, we make sure that the aforementioned pairs contain at least one common object class and differ by fewer than 15 object classes. This filtering results in a total of 118k image pairs. When encoding image pairs in this stage, objects present in the first image but absent in the second are labeled as `REMOVE`, while objects appearing in the second image but not in the first are labeled as `ADD`. Objects exhibiting an intersection-over-union above a predefined threshold across both images are classified as `EDIT`. Further details regarding dataset construction are provided in the supplementary material.

Stage 2: Learning to Detect Inpainted Areas. In the second stage, we refine the model by bridging the gap between pre-training on real image pairs and application to edited images. To accomplish this, we select images from the LVIS dataset and sample non-overlapping annotated objects. Starting from LVIS images, we employ inpainting to generate original and edited image pairs. For each object, an operation is randomly selected among $\{\texttt{ADD}, \texttt{EDIT}, \texttt{REMOVE}\}$. Inpainting is then performed using LaMa [45]: for the added objects, the corresponding region is inpainted on the original image, while for removed objects the inpaint is performed on the edited one. For the edit operation, 30% of the edited objects undergo a color change, while the remaining 70% are substituted with a different object. We rely on GPT-4o to generate target objects suitable for substitution by prompting it with a caption of the original image as well as the original object. To perform the edits, the Kandinsky 2.2 inpaint model [40] is employed to inpaint selected regions. This process leads to a training set of 97k elements and a test set of 19k.

3.4. Estimating the Coherence

Given the original and edited image, along with a detected difference, the coherence estimator produces a binary decision (*i.e.*, Yes/No) and a textual rationale indicating whether the modification is coherent with the intended edit. To induce the model to focus on a specific difference, the bounding box of the detected difference is visually depicted on the original and edited image before they are fed to the coherence estimator. Further, we also include the type of modification c_i and its textual description S_i as input to the model.

To train the model, we annotated 116 samples from the EmuEdit [42] test set, with the corresponding edited images generated using InstructDiffusion [18]. For each sample, we manually annotated both the observed differences and a binary coherence indicator. In addition, we employed GPT-4o to generate a rationale for each annotation, which was subsequently refined through human review to eliminate any inaccuracies. While the command c and the subjects S were provided in textual format, the bounding boxes were directly depicted on the images using a color-coded scheme: red for additions, green for edits, and blue for removals. Specifically, additions are superimposed on the edited image, whereas the bounding boxes corresponding to removals and edits are applied solely to the original image.

4. Experimental Results

4.1. Implementation and Training Details

In our experiments, we consider three recent MLLMs designed to support multi-image input tasks: Idefics3-8B [30], Qwen2-VL-7B [46], and mPLUG-Owl3-7B [49]. Following the training steps described in Sec. 3.3 and Sec. 3.4, we fine-tune each model using QLoRA [12]. Specifically, the vision encoder and projector of each model remain frozen, while the language model is quantized and augmented with LoRA [24] adapters. LoRA is applied to the self-attention and MLP layers across all 32 Transformer blocks.

During the second stage of the difference detector component, we observed that the model may be susceptible to artifacts introduced by the inpainting model, which can compromise its generalizability across different editing outputs. To remove the effect of these low-level imprints, we apply random JPEG compression at 15% and 50% magnitude, following an established practice in deepfake detection [47]. Further details regarding model configuration and training are provided in the supplementary material.

4.2. Evaluating Detected Differences

To evaluate the performance of our detection and coherence pipeline, we construct a new dataset, termed as DICE-D, specifically designed for the task. This dataset comprises 800 edited images extracted from the I^2EBench [35] benchmark. We select images and prompts from a subset of the edit categories that involve object-based modifications, in particular: color alteration, counting (only if involving a single object change), object removal, and object replacement. Each pair of images is manually annotated with detected differences represented as bounding boxes, editing commands (*i.e.*, `ADD`, `REMOVE`, `EDIT`), and textual descriptions. Also, we assign a coherence label for each bounding box to indicate its alignment with the prompt.

Difference Detection Results. We first evaluate the ability of the difference detection model to detect modifications between the original and edited images. Following object detection literature [6, 31], we compute the mean average precision (AP) between predicted and labeled bounding boxes from DICE-D. Specifically, we average results at 10 IoU

	Confidence	Training		Class-agnostic Detection					Class-aware Detection							
		Stage 1	Stage 2	AP	AP_{50}	AP_{75}	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_M	AP_L	AP_{ADD}	AP_{REM}	AP_{EDIT}
Alternative MLLMs																
Qwen2-VL-7B [46]	-	✓	✓	2.0	5.3	1.3	0.3	3.4	1.6	3.3	1.2	0.1	2.4	0.1	0.1	3.2
Qwen2-VL-7B [46]	✓	✓	✓	2.3	5.6	1.8	0.1	3.5	1.6	4.1	1.1	0	1.6	0.1	0.1	3.2
mPLUG-Owl3-7B [49]	-	✓	✓	3.8	10.3	2.4	0.5	5.1	2.5	6.7	1.7	0.3	3.3	1.1	1.5	5.1
mPLUG-Owl3-7B [49]	✓	✓	✓	5.2	13.7	3.2	0.5	7.3	4.4	10.6	3.1	0.3	5.5	0.1	7.3	4.8
Ablation Studies																
Idefics3-8B [30]	-	-	✓	15.1	30.1	13.4	8.5	18.6	9.4	18.1	9.3	5.5	11.0	7.3	7.5	13.4
Idefics3-8B [30]	-	✓	✓	16.7	30.7	16.0	10.4	20.3	10.6	17.9	10.9	6.6	12.5	7.4	7.7	16.5
Idefics3-8B [30]	$✓_{rand}$	✓	✓	16.8	31.2	16.2	10.3	20.5	9.7	16.5	10.0	6.7	11.2	7.0	7.9	14.2
Idefics3-8B [30]	✓	✓	-	2.9	5.1	2.8	1.9	4.2	1.4	2.2	1.4	0.9	1.8	2.0	0.4	1.8
Idefics3-8B [30]	✓	-	✓	18.2	36.8	16.0	9.8	22.7	12.2	22.2	12.1	7.3	15.3	7.2	13.4	15.8
DICE (Ours)																
Idefics3-8B [30]	✓	✓	✓	**22.3**	**40.1**	**21.9**	**13.5**	**36.8**	**15.5**	**24.8**	**16.5**	**11.4**	**15.4**	**19.6**	**14.8**	**16.4**

Table 1. Performance comparison of various MLLMs in the difference detection stage of our pipeline, evaluated under both class-agnostic and class-aware settings. Results are presented in terms of AP metrics across various training configurations.

	Confidence	Coherence over GT Areas	Coherence over Detected Areas		
		Accuracy	AP	AP_{50}	AP_{75}
Alternative MLLMs					
Qwen2-VL-7B [46]	-	76.6	1.4	3.8	0.6
Qwen2-VL-7B [46]	✓	76.6	1.8	4.9	0.9
mPLUG-Owl3-7B [49]	-	85.9	3.8	10.0	2.4
mPLUG-Owl3-7B [49]	✓	85.9	4.5	11.3	2.9
DICE (Ours)					
Idefics3-8B [30]	-	85.4	12.0	20.3	12.0
Idefics3-8B [30]	✓	85.4	**15.5**	**26.0**	**16.1**

Table 2. Performance comparison of various MLLMs in the coherence estimation stage of our pipeline. Coherence accuracy is measured using ground-truth differences, while coherence over detected areas employs the output of the first stage of our pipeline and is evaluated using AP metrics.

thresholds from 0.5 to 0.95. Further, we include AP_{50} (average precision at 50% IoU), AP_{75} (average precision at 75% IoU), AP_M (for medium-sized objects), and AP_L (for large-sized objects), ensuring a comprehensive evaluation across different object scales and IoU thresholds.

Results are reported in Table 1, where we consider both class-agnostic detection and class-aware detection. In class-agnostic detection, all predictions are treated as belonging to the same category, to evaluate only the bounding box location. Vice versa, in class-aware detection, both the bounding box location and the editing command are considered, evaluating the accuracy of both the detection and the associated command. As shown, using Idefics as the base model consistently achieves the best performance across all metrics, with an increase of 17.1 and 20.0 in AP in a class-agnostic setting compared to Qwen and mPLUG. Similarly, when considering the command labels, Idefics outperforms Qwen and mPLUG by 11.8 and 13.9. This superior performance may be attributed to the distinctive image encoding pipeline employed by Idefics, in which each image is resized to a larger dimension and segmented into a grid of crops, with each crop being encoded independently. Although this approach increases the amount of visual information processed, the grid-based encoding simultaneously introduces a localization factor to the captured visual details that can be leveraged in a task of detection.

When validating the contribution of the key components of our pipeline, it is worth noting that the initial pre-training stage significantly contributes to the final performance. Specifically, training Idefics in both stages results in improvements of 1.6 and 1.2 in class-agnostic and class-aware AP, respectively, compared to training solely on the second stage dataset. Notably, even training exclusively on the first stage enhances class-agnostic detection compared to Qwen (*i.e.*, +0.4 AP). This observation implies that predicting missing objects in pairs of similar real images not only facilitates the learning of prediction syntax but also enables the transfer of relevant knowledge to the difference detection task in edited images.

Moreover, incorporating the extracted confidence score enhances difference localization results, with improvements of 5.5 and 3.1 AP observed for Idefics-based models trained on both stages and solely on the second stage respectively, compared to using a fixed confidence value of 1. Additionally, using random confidence values results in worse performance, further indicating that the uncertainty in the command correlates with the actual prediction confidence.

Coherence Estimation Results. The coherence estimation model assesses whether each detected difference is correctly aligned with the prompt using a binary prediction (*i.e.*, Yes/No). Moreover, the coherence step incorporates explicit model reasoning [48], thereby enhancing the interpretability of the results. Under this setting, models are evaluated on DICE-D, in terms of coherence over ground-truth areas and coherence over detected areas. To measure the coherence accuracy, ground-truth differences are fed into the coherence estimator, and results are evaluated

Editing Models	Correct Edits		Unwanted Edits		No Visual Change	
	DICE (%) ↑	Humans (1-5) ↑	DICE (%) ↓	Humans (1-5) ↑	DICE (%) ↓	Humans (%) ↓
HIVE [52]	11.0	2.0	40.5	3.1	54.5	40.5
InstructPix2Pix [3]	15.5	2.3	32.1	3.3	44.0	22.0
MGIE [16]	23.0	2.7	21.6	3.8	11.5	10.0
MagicBrush [51]	24.5	2.9	23.1	3.9	8.5	5.5
InstructDiffusion [18]	30.0	3.0	19.1	4.0	13.5	10.5

Table 3. Benchmark comparison of model rankings generated by DICE and those derived from the user study. The first two columns contrast average human ratings with scores obtained using DICE. The final column compares the percentage of unchanged images in the user study – cases with maximal background preservation and minimal prompt adherence – to the corresponding one identified by DICE.

against the manually annotated coherence. Instead, for the coherence over detected areas, the performance of the entire DICE pipeline is measured through AP, considering the coherence label of the difference as the ground-truth category of the bounding box. This is done by extracting differences with the detector and subsequently predicting the coherence of each difference.

Results are summarized in Table 2, evaluating the effectiveness of various MLLMs as coherence estimators. Idefics consistently achieves the highest overall performance on AP, surpassing mPLUG and Qwen by 11.0 and 12.7 points, respectively, while maintaining comparable coherence accuracy. Additionally, incorporating extracted confidence into the AP computation leads to performance improvements across all evaluated MLLMs. This highlights the importance of confidence extraction in the difference detection phase and motivates further research on aligning token probability with confidence estimation in object detection. Based on these results, Idefics3-8B is selected as the base MLLM for DICE in all subsequent experiments.

4.3. Evaluation of Image Editing Models

User Study Details. We conduct a user study to evaluate our proposed framework within the context of instruction-based image editing. In detail, we collect 200 prompts along with their corresponding original images, 50% from MagicBrush [51] and 50% from I^2EBench [35]. For MagicBrush, we use GPT-4o to filter out prompts that do not involve object modification. Differently, for the I^2EBench subset, we employ the same selection criteria as in the construction of DICE-D (cf. Sec. 4.2). We select five state-of-the-art generators and generate 200 edited images per generator. Specifically, our model selection includes HIVE [52], InstructPix2Pix [3], MGIE [16], MagicBrush [51], and InstructDiffusion [18]. To assess the quality of the generated edits, we asked 10 human annotators to evaluate each sample on two dimensions: *prompt adherence* and *background preservation*. Prompt adherence measures how well the modifications in the edited image align with the instructions given in the prompt. Differently, background preservation assesses how well the elements that were not intended to be modified remained unchanged. Ratings for each dimension were recorded on a 5-point Likert scale. We refer the user to the supplementary materials for more detailed information about the user study.

Model Editing Ranking. To evaluate the performance of editing models, we establish three axes to rank the considered generators using DICE. *Correct edits* measure the percentage of images in which at least one detected difference is classified as coherent. Conversely, *unwanted edits* quantify the percentage of the total image area containing differences marked as non-coherent. Lastly, *no visual change* represents the percentage of images where DICE does not detect any visual alteration, potentially due to insufficient prompt clarity, lack of understanding, or inability of the model to execute the requested modification. Notably, *correct* and *unwanted edits* can be measured with the average prompt adherence and background preservation ratings from the user study. Similarly, from the user study, *no visual change* is assessed by identifying images rated 5 in background preservation and 1 in prompt following.

In Table 3, we compare the rankings generated by our evaluation with those obtained from the user study. As it can be seen, model rankings from DICE align with human evaluations. In particular, InstructDiffusion and MagicBrush emerge as the top-performing models both according to DICE and human ratings, excelling in performing correct modifications while effectively preserving the background. Likewise, the ranking order for *no visual change* remains consistent across DICE and the user study, further validating the reliability of our approach.

To qualitatively validate the predictions of DICE, Fig. 3 presents editing samples showcasing difference detection and coherence predictions across various prompts and models. Notably, DICE effectively identifies image differences and evaluates their coherence with the editing prompt.

Correlation with Human Judgment. To further validate the alignment between the detected differences and human judgment, we integrate DICE predictions within CLIP-based evaluation metrics measuring image similarity (CLIP-I) for background preservation and image-text similarity (CLIP-T) for prompt adherence [23]. In particular, CLIP-I measures the similarity between the original and the edited image. To compute CLIP-T, instead, we caption the

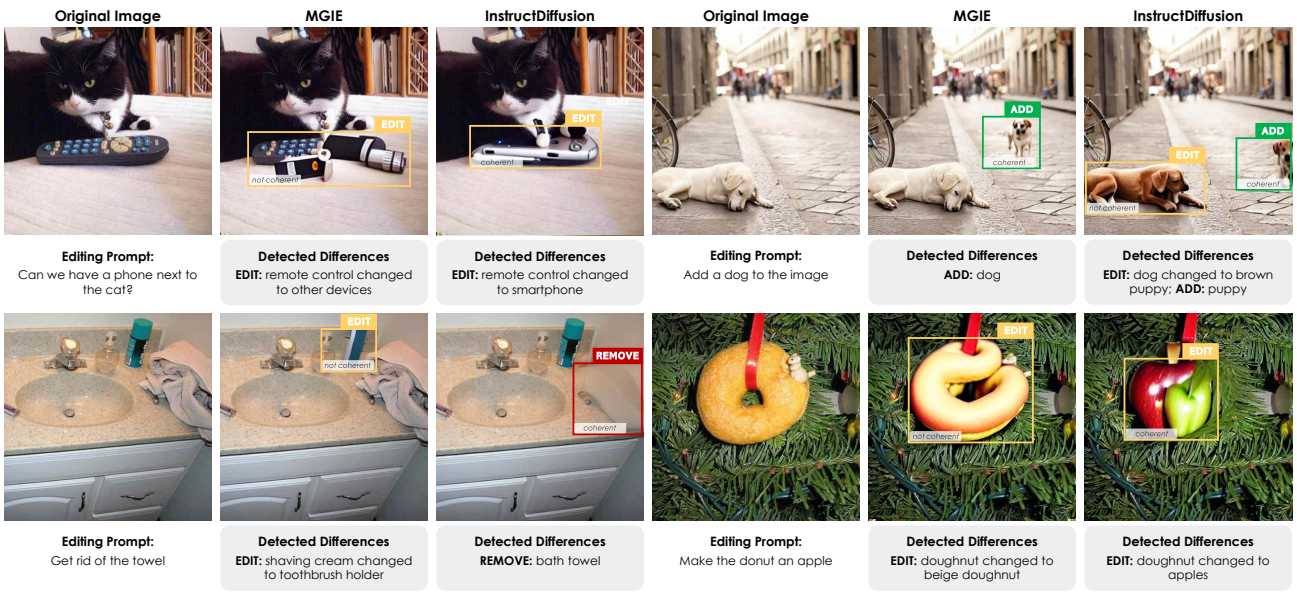

Figure 3. Qualitative samples of DICE applied on images edited by MGIE [16] and InstructDiffusion [18] models.

	ρ	ρ_s	τ
Background Preservation			
CLIP-I	51.1	42.9	33.3
w/ patch on random areas	34.2	25.6	19.4
w/ patch on all detected differences	32.7	15.6	11.6
w/ **DICE** (patch on coherent differences)	**54.5**	**45.4**	**35.1**
Prompt Adherence			
CLIP-T	21.5	23.1	17.1
w/ patch on random areas	17.8	19.5	14.3
w/ patch on all detected differences	13.3	14.6	10.7
w/ **DICE** (patch on non-coherent differences)	**24.6**	**26.6**	**19.6**

Table 4. Correlation analysis showing improved alignment with human judgment when integrating DICE in CLIP-I and CLIP-T.

original image using an MLLM (*i.e.*, Idefics3-8B in our experiments) and feed the extracted caption along with the editing prompt to GPT-4o to produce a caption of the ideal edited image (*i.e.*, target caption). Through CLIP-T we then measure the CLIP similarity between the target caption and the actual edited image. To integrate DICE in CLIP-I and CLIP-T, we modify both the original and edited images by selectively masking specific regions, taking into account DICE predictions. Specifically, for background preservation, we mask coherent differences, whereas, for prompt adherence assessment, non-coherent differences are patched.

For this experiment, we compare the correlation scores of standard CLIP-I and CLIP-T metrics with those obtained by masking original and edited images using DICE. Results are shown in Table 4, where we measure Spearman's ρ, Pearson's ρ_s, and Kendall's τ correlation coefficients between automated evaluation scores and human judgment. We also compare when randomly masking areas from the differences detected by DICE, or when masking all detected differences, including both coherent and non-coherent changes. As it can be seen, DICE effectively enhances the correlation of both CLIP-I and CLIP-T metrics with human ratings. Specifically, for background preservation, CLIP-I exhibits the strongest correlation with human judgments when masking only coherent differences, achieving a Pearson's score of 54.5. This is a significant improvement over the baseline score of 51.1 (no masking), and substantially higher than the scores obtained with random masking or masking all detected differences. This result suggests that excluding correctly modified regions allows CLIP-I to better focus on the actual background, producing a metric that more accurately aligns with human perception.

For prompt adherence, CLIP-T shows the strongest correlation when we patch non-coherent differences, achieving a Pearson's score of 24.6, compared to 21.5 without masking. Conversely, masking all detected differences leads to a drop to 13.3, likely due to indiscriminate removal of both erroneous and correct edits. This outlines that selectively removing only incorrect modifications helps CLIP-T focus on prompt-relevant changes, thereby improving alignment with human evaluations.

Overall, these results demonstrate that DICE enhances the reliability of CLIP-based metrics, making them more aligned with human perception of editing quality. In particular, distinguishing between coherent and non-coherent changes is crucial for obtaining meaningful evaluations.

5. Conclusions

In this work, we introduced DICE: a novel approach to evaluate instruction-guided image editing by detecting and assessing object-level differences between original and edited images. Our method leverages MLLMs to detect seman-

tic differences between images and determine their coherence with the given editing instructions. This dual-step process allows for a structured and interpretable evaluation, addressing both the accuracy of the detected modifications and their semantic alignment with user intent.

Acknowledgments

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high-performance computing resources. This work has been supported by the EU Horizon projects "ELIAS - European Lighthouse of AI for Sustainability" (No. 101120237) and "ELSA - European Lighthouse on Safe and Secure AI" (No. 101070617), and by the PNRRM4C2 project "FAIR - Future Artificial Intelligence Research" funded by the EU - NextGenerationEU.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 3

[2] Federico Betti, Jacopo Staiano, Lorenzo Baraldi, Lorenzo Baraldi, Rita Cucchiara, and Nicu Sebe. Let's ViCE! Mimicking Human Cognitive Behavior in Image Generation Evaluation. In *ACM Multimedia*, 2023. 2

[3] Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. In *CVPR*, 2023. 1, 2, 3, 7

[4] Davide Bucciarelli, Nicholas Moratelli, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Personalizing Multimodal Large Language Models for Image Captioning: An Experimental Analysis. In *ECCV*, 2024. 3

[5] Davide Caffagni, Federico Cocchi, Luca Barsellotti, Nicholas Moratelli, Sara Sarto, Lorenzo Baraldi, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. The Revolution of Multimodal Large Language Models: A Survey. In *ACL Findings*, 2024. 1, 2

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *ECCV*, 2020. 4, 5

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *ICCV*, 2021. 2, 3

[8] Gongwei Chen, Leyang Shen, Rui Shao, Xiang Deng, and Liqiang Nie. Lion: Empowering multimodal large language model with dual-level visual knowledge. In *CVPR*, 2024. 3

[9] Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing Multimodal LLM's Referential Dialogue Magic. *arXiv preprint arXiv:2306.15195*, 2023. 3

[10] Mang Tik Chiu, Yuqian Zhou, Lingzhi Zhang, Zhe Lin, Connelly Barnes, Sohrab Amirghodsi, Eli Shechtman, and Humphrey Shi. Brush2Prompt: Contextual Prompt Generator for Object Inpainting. In *CVPR*, 2024. 2

[11] Federico Cocchi, Nicholas Moratelli, Davide Caffagni, Sara Sarto, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. LLaVA-MORE: A Comparative Study of LLMs and Visual Backbones for Enhanced Visual Instruction Tuning. *arXiv preprint arXiv:2503.15621*, 2025. 3

[12] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient Finetuning of Quantized LLMs. In *NeurIPS*, 2023. 5

[13] Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat Gans on Image Synthesis. In *NeurIPS*, 2021. 2

[14] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024. 1

[15] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *ICML*, 2024. 1, 2

[16] Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Yang Wang, Yinfei Yang, and Zhe Gan. Guiding Instruction-based Image Editing via Multimodal Large Language Models. In *ICLR*, 2024. 2, 3, 7, 8

[17] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016. 2

[18] Zigang Geng, Binxin Yang, Tiankai Hang, Chen Li, Shuyang Gu, Ting Zhang, Jianmin Bao, Zheng Zhang, Houqiang Li, Han Hu, et al. InstructDiffusion: A Generalist Modeling Interface for Vision Tasks. In *CVPR*, 2024. 1, 2, 5, 7, 8

[19] Ross Girshick. Fast R-CNN. In *ICCV*, 2015. 4

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *NeurIPS*, 2014. 2

[21] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*, 2025. 1

[22] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A Dataset for Large Vocabulary Instance Segmentation. In *CVPR*, 2019. 5, 1

[23] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *EMNLP*, 2021. 7

[24] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022. 5

[25] Yi Huang, Jiancheng Huang, Yifan Liu, Mingfu Yan, Jiaxi Lv, Jianzhuang Liu, Wei Xiong, He Zhang, Shifeng Chen, and Liangliang Cao. Diffusion Model-Based Image Editing: A Survey. *arXiv preprint arXiv:2402.17525*, 2024. 1

[26] Yuzhou Huang, Liangbin Xie, Xintao Wang, Ziyang Yuan, Xiaodong Cun, Yixiao Ge, Jiantao Zhou, Chao Dong, Rui Huang, Ruimao Zhang, et al. SmartEdit: Exploring Complex Instruction-based Image Editing with Multimodal Large Language Models. In *CVPR*, 2024. 2

[27] Mude Hui, Siwei Yang, Bingchen Zhao, Yichun Shi, Heng Wang, Peng Wang, Yuyin Zhou, and Cihang Xie. HQ-Edit: A High-Quality Dataset for Instruction-based Image Editing. *arXiv preprint arXiv:2404.09990*, 2024. 2, 3

[28] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*, 2019. 2

[29] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning Segmentation via Large Language Model. In *CVPR*, 2024. 3

[30] Hugo Laurençon, Andrés Marafioti, Victor Sanh, and Léo Tronchon. Building and better understanding vision-language models: insights and future directions. In *NeurIPS Workshops*, 2024. 3, 5, 6

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 5

[32] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In *NeurIPS*, 2023. 1, 3

[33] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019. 1

[34] Chuofan Ma, Yi Jiang, Jiannan Wu, Zehuan Yuan, and Xiaojuan Qi. Groma: Localized Visual Tokenization for Grounding Multimodal Large Language Models. In *ECCV*, 2024. 3

[35] Yiwei Ma, Jiayi Ji, Ke Ye, Weihuang Lin, Zhibin Wang, Yonghan Zheng, Qiang Zhou, Xiaoshuai Sun, and Rongrong Ji. I2EBench: A Comprehensive Benchmark for Instruction-based Image Editing. In *NeurIPS*, 2024. 2, 3, 5, 7

[36] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple Open-Vocabulary Object Detection with Vision Transformers. In *ECCV*, 2022. 1

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning Robust Visual Features without Supervision. *arXiv preprint arXiv:2304.07193*, 2023. 5

[38] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 1, 2

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021. 2, 3

[40] Anton Razzhigaev, Arseniy Shakhmatov, Anastasia Maltseva, Vladimir Arkhipkin, Igor Pavlov, Ilya Ryabov, Angelina Kuts, Alexander Panchenko, Andrey Kuznetsov, and Denis Dimitrov. Kandinsky: an Improved Text-to-Image Synthesis with Image Prior and Latent Diffusion. *arXiv preprint arXiv:2310.03502*, 2023. 5, 1

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *CVPR*, 2022. 1, 2

[42] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu Edit: Precise Image Editing via Recognition and Generation Tasks. In *CVPR*, 2024. 3, 5

[43] Jing Shi, Ning Xu, Trung Bui, Franck Dernoncourt, Zheng Wen, and Chenliang Xu. A benchmark and baseline for language-driven image editing. In *ACCV*, 2020. 2, 3

[44] Jing Shi, Ning Xu, Yihang Xu, Trung Bui, Franck Dernoncourt, and Chenliang Xu. Learning by Planning: Language-Guided Global Image Editing. In *CVPR*, 2021. 2, 3

[45] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust Large Mask Inpainting with Fourier Convolutions. In *WACV*, 2022. 2, 5

[46] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution. *arXiv preprint arXiv:2409.12191*, 2024. 1, 3, 5, 6

[47] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, and Alexei A Efros. CNN-generated images are surprisingly easy to spot... for now. In *CVPR*, 2020. 5

[48] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*, 2022. 6

[49] Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mPLUG-Owl3: Towards Long Image-Sequence Understanding in Multi-Modal Large Language Models. *arXiv preprint arXiv:2408.04840*, 2024. 1, 3, 5, 6

[50] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training. In *ICCV*, 2023. 1

[51] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A Manually Annotated Dataset for Instruction-Guided Image Editing. In *NeurIPS*, 2023. 1, 2, 3, 7

[52] Shu Zhang, Xinyi Yang, Yihao Feng, Can Qin, Chia-Chih Chen, Ning Yu, Zeyuan Chen, Huan Wang, Silvio Savarese, Stefano Ermon, et al. HIVE: Harnessing Human Feedback for Instructional Visual Editing. In *CVPR*, 2024. 2, 3, 7, 1

[53] Yichi Zhang, Ziqiao Ma, Xiaofeng Gao, Suhaila Shakiah, Qiaozi Gao, and Joyce Chai. GROUNDHOG: Grounding Large Language Models to Holistic Segmentation. In *CVPR*, 2024. 3

What Changed? Detecting and Evaluating Instruction-Guided Image Edits with Multimodal Large Language Models

Supplementary Material

A. Difference Detection Training Dataset

Stage 1: Additional Details. To effectively train our difference detection model, we build our first stage on LVIS [22], which provides a broad range of annotated objects. Given that LVIS encompasses 1,723 distinct categories, inconsistencies may arise during annotation, whereby annotators might label the same or similar objects under different categories or overlook certain objects. To address our objective of predicting missing objects in each pair, we filter them using the open-vocabulary detector OWL-ViT [36]. OWL-ViT is then tasked with verifying whether objects annotated in the first image (but not in the second) are genuinely absent in the second image and vice versa, thereby preventing erroneous ground-truth labels.

Moreover, considering the density of annotations in LVIS, we further refine our collection by eliminating annotations that are overly small, specifically, those with at least one dimension measuring less than 16 pixels. The final dataset consists of 118k images with a total of 795k, 725k, and 94k `ADD`, `REMOVE` and `EDIT` operations respectively.

Stage 2: Additional Details. Building upon the previous stage, the second stage dataset is designed to refine our model for detecting differences in edited images. The dataset is constructed using LVIS annotations, applying specific filtering criteria to ensure data quality. In detail, we sample a maximum of 4 objects per image. For each selected object, we consider its segmentation mask and filter out records that cover less than 3% of the image area, while the maximum allowed overlap between selected object masks is 5%. As for the inpainting generation pipeline, the Kandinsky 2.2 inpaint model [40] is employed with 100 diffusion steps and a guidance scale of 4.

The final dataset comprises a total of 97k images for training and 19k images for testing. Within the training set, each of the three operations (`ADD`, `REMOVE`, and `EDIT`) is represented by approximately 33k instances to avoid unbalanced predictions. Additionally, 19k images remain unchanged. Similarly, in the test set, there are around 7k instances for each of the three operations, along with 5k images that remain unaltered.

B. Training Details

In each stage, the training of all the models is conducted within the same experimental setting. For instance, we apply QLoRA with rank 64, scaling factor 16, and a dropout rate of 0.1. Both difference detection and coherence estimation fine-tuning employ 4-bit quantization and the paged AdamW 8-bit optimizer [33].

For image encoding, images are always center-cropped based on the smaller dimension to maintain a square aspect ratio. When employing Idefics3, images are internally resized to 1,456 pixels and encoded in a list of 364-pixel crops, generating a grid of 16 crops separately encoded. For Qwen2-VL, the image is encoded using a bidimensional positional embedding. Image tokens are then compressed by an MLP layer that encodes 2×2 adjacent tokens into a single one. The input image resolution is kept at 1,456. For mPLUG-Owl3, the images are directly encoded through a SigLIP400m-384 [50].

Difference Detection. All stages of the difference detection training use 8 NVIDIA A100 64GB GPUs. Both training stages are conducted with a learning rate of 1×10^{-4} and a total batch size of 8. In the first training stage, Idefics converges after 9k training steps, with evaluations performed every 1k steps. Differently, in the second training stage, convergence is reached after 30k training steps, with evaluations conducted every 5k steps.

Coherence Estimation. Training is performed on a single NVIDIA A100 64GB GPU with a batch size of one and a learning rate of 1×10^{-5}. In this case, Idefics reaches convergence at 550 training steps, with evaluations every 50 steps.

C. User Study

To evaluate the performance of instruction-based image editing models, we conducted a user study in which participants assessed edited images based on prompt adherence and background preservation.

Data Generation. The editing models used for user study data generation are set according to their default configuration. In particular, for InstructDiffusion [18] the image guidance scale is set to 1.25 and the text guidance scale is set to 5, while for all the other models we use 1.5 and 7 respectively. For HIVE [52] and InstructDiffusion the number of inference steps is 100, while for all the other models we set the inference steps to 20.

Participants rated the degree to which the requested edit was applied on a scale from 1 to 5: *not applied at all*, *slightly applied*, *moderately applied*, *well applied*, and *perfectly applied*. Similarly, they evaluated the preservation of background elements using another scale from 1 to 5: *changed completely*, *moderately altered*, *mostly preserved*,

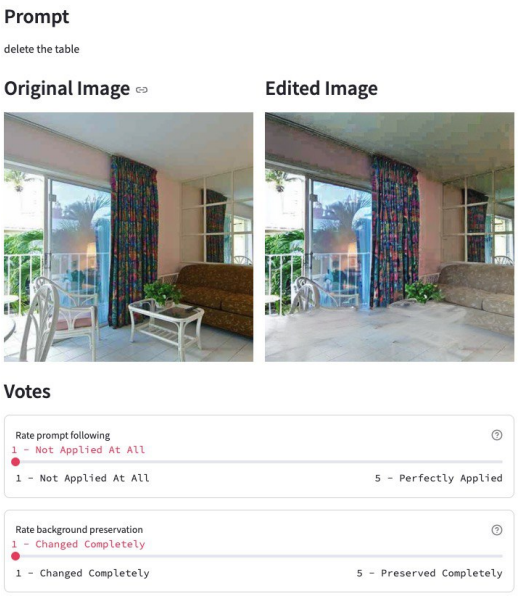

Figure 4. User study interface displaying the original and the edited image alongside the editing prompt.

nearly fully preserved, and *preserved completely*. The study was conducted through an interactive interface that allowed participants to compare the original and the edited image while reading the prompt. Fig. 4 illustrates the interface used in the study.

D. Additional Qualitative Results

Fig. 5 illustrates a wide range of successful edits handled by DICE across diverse scenarios, featuring highly precise bounding boxes around the modified elements. In several examples, the system accurately detects color changes (*e.g.*, altering the color of a bird or an umbrella) and clearly distinguishes added objects, such as a watermelon or a necktie. The pipeline also robustly identifies removed elements, whether it is a coffee table or a giraffe, and captures more subtle edit operations, for instance replacing a bird with a boy or a cat with a dog. These qualitative results highlight how DICE maintains spatial accuracy and class awareness when performing additions, removals, and edits, thereby demonstrating its adaptability to different editing requests.

Additionally, the coherence evaluator provides clear, step-by-step reasoning for each detected change, explaining why it is coherent or not. For example, as shown in Fig. 6, in the "change the color of the rose to blue" prompt, the system points out that "cyan" is still a valid shade of blue and correctly labels the edit as coherent. Likewise, when asked to "add a basketball to the top of the car", it confirms not only that the basketball has been introduced but also that it is located precisely where the prompt requires – on top of the car. These reasoned explanations highlight the transparency of our pipeline, helping users understand which aspects of the prompt were fulfilled and why certain edits might fall short of the requested modifications.

E. Limitations

DICE can occasionally exhibit inaccuracies in detected differences, as shown in Fig. 7, that can end up affecting the coherence estimation. For instance, in the first example, changing the left hat to white actually matches the prompt. However, the pipeline labels the detected edit as "fedora", which confuses the coherence evaluator into classifying it as non-coherent. A similar problem arises in the second image, where the correct recoloring of the dog to brown is labeled as "beige puppy", causing confusion – especially because the new color of the dog blends into the background.

Another failure can occur when the coherence evaluator is overly strict. In the "replace dog with watermelon" example, the editing model does replace the dog with a watermelon, yet the coherence evaluator rejects the result because it expects a total transformation. Similarly, in "replace boy with girl," the difference detector notes that a person has been edited but fails to recognize the new person as female, and the coherence evaluator does not correct this oversight, causing it to label the modification as incoherent even though it actually meets the prompt requirements.

Further ambiguities in the prompt can lead to inaccurate predictions. Indeed, sometimes, the prompt does not specify all the elements needed to produce a univocal decision, especially when it is unclear whether the prompt specifies an addition or a substitution of elements in the scene.

Custom System Prompt:

```
You are a system that detects differences between two images.

- Extract the elements that are changed in the second image with respect to the first one.
- Create a new entry for each distinct change.
- For each entry, use the following format:
"<CHANGE_COMMAND>:  <CHANGED_ELEMENT>, (<BOUNDING_BOX>)"

CHANGE_COMMAND:
- ADD: If a new element appears in the second image that was not present in the first.
- REMOVE: If an element from the first image is missing in the second.
- EDIT: If an element in the second image is different but in the same location as an element
in the first image.

CHANGED_ELEMENT: Describe the element that has changed.

BOUNDING_BOX: Use normalized coordinates [x0, y0, x1, y1] for the changed element position
in the second image, where (x0, y0) is the top-left corner, and (x1, y1) is the bottom-right
corner.  The coordinates should be scaled between 0 and 1, with 0 representing one edge of
the image and 1 representing the opposite edge.
```

Table 5. Prompting template for the difference detection stage of the DICE pipeline.

Custom System Prompt:

```
You are evaluating if a specific change detected by an AI vision model matches the request in
the original edit prompt.

## Task
Determine if the detected change, as described and bounded by the provided colored bbox,
matches the request in the original edit prompt.
A match is valid only if the localized detected change is 100% compatible with the requested
prompt.
Any unwanted modification of the original image (even small) should avoid a match.

## Context
- The original image and the edited image are provided, in this order.  The edited image is
the original with some changes applied.  Focus only on the area specified by the bbox in the
detected change.
- Another AI model has detected a change in the image, including its bbox.
- ADD: An object is only added in the edited image (on the background).
- EDIT: An object is substituted with another one in the edited image.
- REMOVE: An object is removed in the edited image.
- Be strict:  An EDIT means that an object has been removed and substituted with another one,
ensure nothing was removed unless explicitly stated in the prompt.  If an object has been
removed unexpectedly, then you should say NO.

## Example Response
- Reasoning:  <REASONING>
- Decision:  "YES" or "NO"
```

User Prompt:

```
## Instructions
1.  The original edit prompt is:  {SUBSTITUTE_PROMPT}
2.  The detected change to evaluate is:  {SUBSTITUTE_CHANGE}
3.  Use only the text and the observations from the specified bbox area (colored) in both the
original and edited images to decide if the specific detected change aligns with the original
edit prompt.
```

Table 6. Prompting template for the coherence estimation stage of the DICE pipeline.

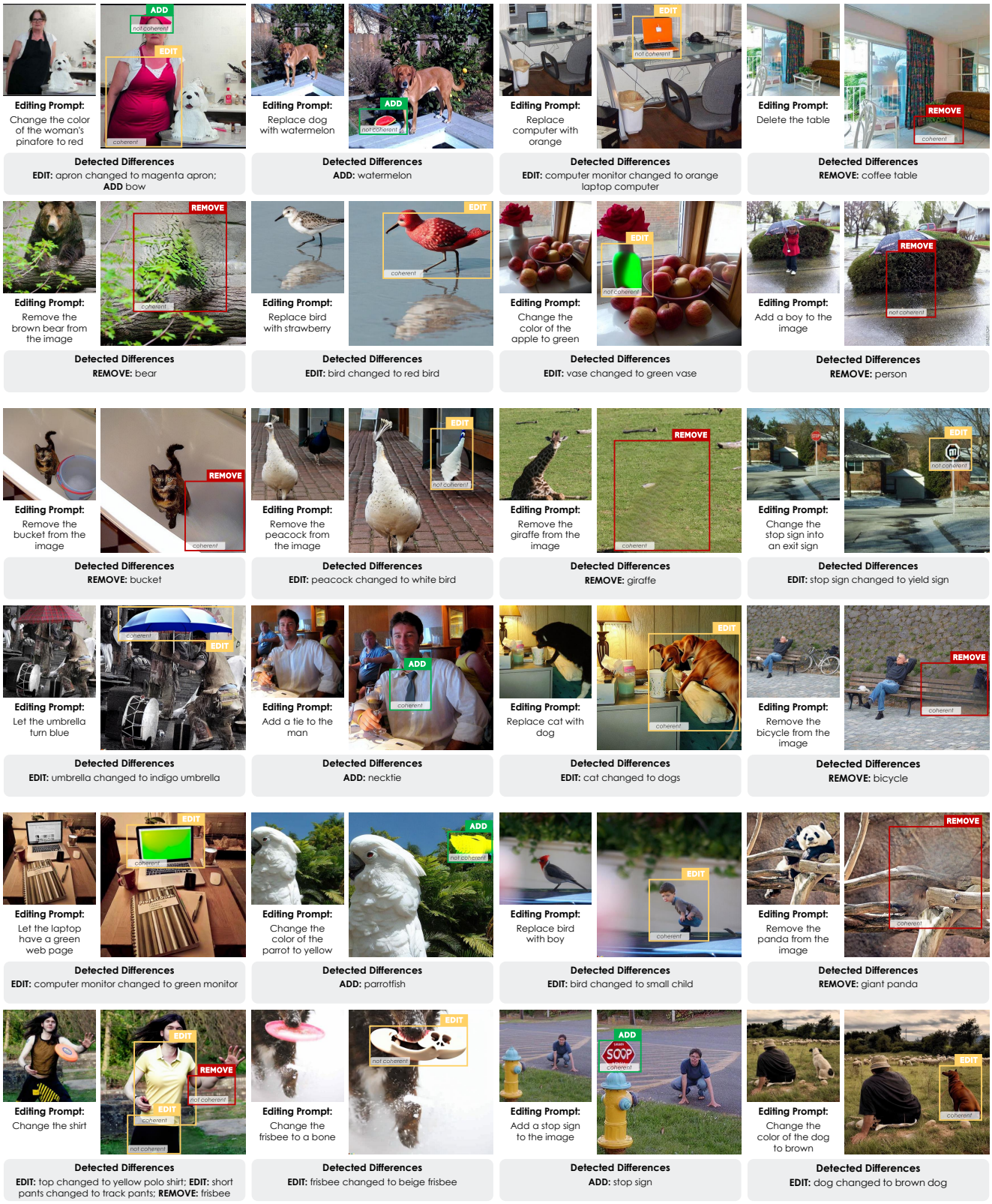

Figure 5. Additional qualitative results. Each instruction-based edit shows the original image (left) and the edited version (right), alongside the given prompt.

Figure 6. Examples illustrating the reasoning of the coherence evaluator that justifies its 'YES' or 'NO' decisions. Each box pairs a detected difference with an explanation of why the edit either fulfills or fails the user's request, highlighting the ability of the pipeline to handle both spatial and semantic context.

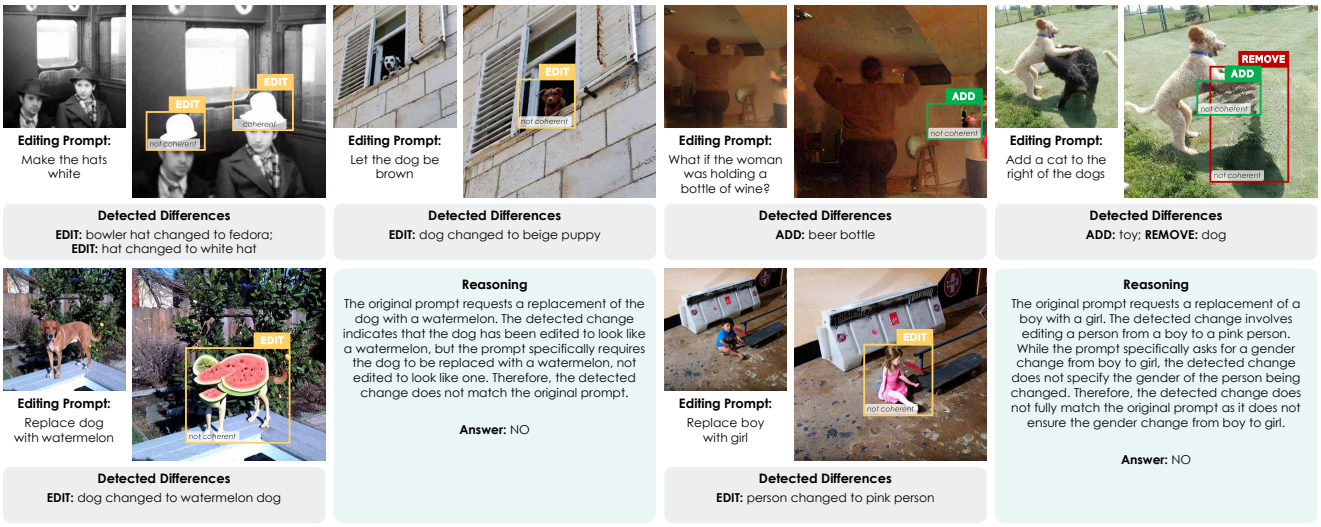

Figure 7. Failure cases where detection errors may impact coherence evaluation, potentially leading to misclassifications. Inaccurate identification of edits can introduce ambiguity for the coherence evaluator, while strict coherence criteria might occasionally reject valid changes, highlighting the interdependence of both stages.