Multimodal Inconsistency Reasoning (MMIR): A New Benchmark for Multimodal Reasoning Models

Anonymous ACL submission

Abstract

Existing Multimodal Large Language Models (MLLMs) are predominantly trained and tested on consistent visual-textual inputs, leaving open the question of whether they can handle inconsistencies in real-world, layout-rich content. To bridge this gap, we propose the Multimodal Inconsistency Reasoning (MMIR) benchmark to assess MLLMs' ability to detect and reason about semantic mismatches in artifacts such as webpages, presentation slides, and posters. MMIR comprises 534 challenging samples, each containing synthetically injected errors across five reasoning-heavy categories: Factual Contradiction, Identity Misattribution, Contextual Mismatch, Quantitative Discrepancy, and Temporal/Spatial Incoherence. We evaluate six state-of-the-art MLLMs, showing that models with dedicated multimodal reasoning capabilities, such as o1, substantially outperform their counterparts while open-source models remain particularly vulnerable to inconsistency errors. Detailed error analyses further show that models excel in detecting inconsistencies confined to a single modality, particularly in text, but struggle with cross-modal conflicts and complex layouts. Probing experiments reveal that single-modality prompting, including Chain-of-Thought (CoT) and Set-of-Mark (SoM) methods, yields marginal gains, revealing a key bottleneck in cross-modal reasoning. Our findings highlight the need for advanced multimodal reasoning and point to future research on multimodal inconsistency.

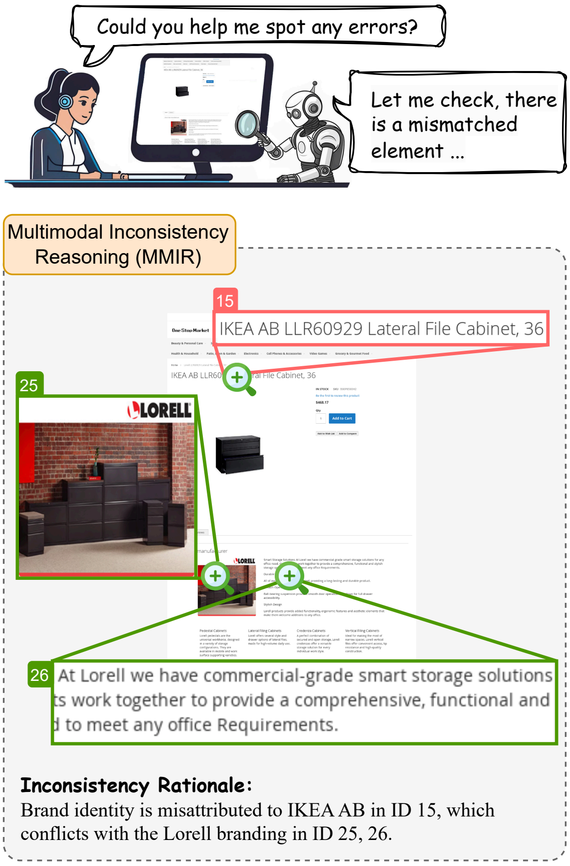


Figure 1: **An illustration of multimodal inconsistency reasoning on a webpage.** An agent examines a webpage where the brand "IKEA AB" is mentioned, but other elements clearly refer to "Lorell." Detecting this brand identity misattribution requires the ability to compare text fields across different sections of the page and reconcile them with accompanying images or context—an inherently multimodal reasoning task.

1 Introduction

Recent advances in Large Language Models (LLMs) have demonstrated impressive reasoning abilities across a variety of tasks (OpenAI, 2024b; Guo et al., 2025; Kojima et al., 2022; Wei et al., 2022). Building on pre-trained LLMs, Multimodal Large Language Models (MLLMs) are fast evolving. However, they usually face greater challenges as they need to reason across different modalities, especially when inconsistencies (i.e., mismatched or contradictory contents) exist. We find that, being primarily trained and evaluated on consistent visual-textual inputs, existing MLLMs are largely untested in scenarios where the input contains misaligned or contradictory information—a situation that is common in real-world scenarios. For example, in Figure 1, a user presents a web page containing conflicting visual and textual elements, asking the model to identify errors.

To comprehensively evaluate the ability of MLLMs in reasoning over multimodal inconsistency, we introduce the **Multimodal Inconsistency Reasoning Benchmark (MMIR)**. MMIR is the first framework dedicated to evaluating how effectively MLLMs can reason about and identify semantic mismatches within complex, layout-rich content with interleaved image and text components. Our benchmark is built on a diverse collection of real-world artifacts (e.g. websites, slides, posters) which have been augmented with **synthetic inconsistencies**—realistic inconsistency errors injected into their original structures. These inconsistency errors span a range of reasoning-heavy categories: *Factual Contradiction, Identity Misattribution, Contextual Mismatch, Quantitative Discrepancy*, and *Temporal/Spatial Incoherence*—posing a next-level reasoning challenge for models. For example, resolving a *Identity Misattribution* involves verifying entity alignment across modalities, while *Quantitative Discrepancy* requires cross-referencing chart data with textual claims. By challenging models to detect such inconsistencies, MMIR forces them to perform intricate reasoning that goes well beyond simple pattern recognition. This benchmark not only exposes the limitations of current MLLMs in handling real-world challenges of reasoning over multimodal content with inconsistency, but also provides a platform for developing more robust multimodal reasoning systems.

In our experiments, we evaluated the advanced multimodal reasoning model o1 (OpenAI, 2024b) and five other state-of-the-art MLLMs: GPT-4o (OpenAI, 2024a), Qwen2.5-VL (Team, 2025), LLaVA-NeXT (Liu et al., 2024b), InternVL2.5 (Chen et al., 2024) and Phi-3.5-Vision (Abdin et al., 2024) using MMIR's 534 test samples. The results overall underscore that current MLLM models struggle with multimodal inconsistency reasoning. Specifically, there is a stark contrast between proprietary and open-source models. The open-source models evaluated only reach less than 25% accuracy. o1 with strong reasoning capability achieves the overall best performance with over 50% accuracy.

To further understand the benchmarking results, we conduct analysis based on the inconsistency category, modality, and layout complexity of the artifact. We find the proprietary models excel in identifying factual contradiction and identity misattribute types of inconsistency and inconsistency within a single modality, either image or text. Last but not least, we investigate some approaches to enhance the model performance in our probing experiment. The results indicate that text-based Chain-of-Thought prompting and visual-based prompting (Set-of-Mark annotations) offer minimal and sometimes adverse effects, whereas an iterative multimodal interleaved reasoning strategy shows promising gains. Overall, these results highlight a critical bottleneck in the ability of MLLMs to perform robust, integrated reasoning—a key challenge for future research.

Our contributions are threefold:

- We introduce MMIR, a novel benchmark that targets the critical yet underexplored task of multimodal inconsistency reasoning in layout-rich content.
- We perform a comprehensive evaluation of one leading multimodal reasoning model and five state-of-the-art MLLMs, revealing significant gaps in their ability to detect inconsistency errors with detailed error analyses across multiple error types, modalities, and layout complexities.
- We provide detailed probing analyses that expose key challenges—from perceptual shortcomings to reasoning bottlenecks—and propose a framework that iteratively refines predictions by jointly leveraging visual and textual modalities.

2 Related Work

Multimodal Understanding and Reasoning Multimodal Large Language Models (MLLMs) process multimodal inputs by first processing visual inputs with pre-trained vision encoders such as CLIP (Radford et al., 2021) to extract features, and then projecting them into the textual representation space with adapters (Liu et al., 2024a; Li et al., 2023a). Significant efforts have been made to bridge the gap between vision and text modalities via integrating more cross-modality data such as interleaved image-text sequences and visual grounding data (Alayrac et al., 2022; Chen et al., 2023; Peng et al., 2023). Also, some recent works develop MLLMs with improved nuanced multimodal abilities, such as Optical Character Recognition (OCR) (Bai et al., 2023; Liu et al., 2024b), layout understanding (Feng et al., 2024; Fan et al., 2024a), Graphic User Interface (GUI) interpretation (Liu et al., 2024c; Team, 2025).

As MLLMs typically leverage pre-trained large language models (LLMs) as the backbone, they inherent strong textural reasoning abilities from the advanced LLMs(Floridi and Chiriatti, 2020; Touvron et al., 2023; Bai et al., 2023; Taori et al., 2023; Chowdhery et al., 2023; OpenAI, 2024a; Team, 2024). To further enhance the reasoning ability of MLLMs, increasing efforts have focused on improving MLLMs in multimodal reasoning. The proprietery model, o1 (OpenAI, 2024b) first realize strong multimodal reasoning with reasoning process similar to the Chain-of-Thought (Wei et al., 2022) and other following works have also explored the multimodal reasoning either through training (Wu and Xie, 2024; Qi et al., 2024; Shao et al., 2024) or prompting (Zhang et al., 2023, 2024b; Zheng et al., 2023).

Multimodal Reasoning Benchmarks To evaluate the reasoning capabilities of MLLMs, numerous benchmarks have been developed with various focuses. Broad-coverage benchmarks such as MM-Bench (Liu et al., 2024d), MMMU (Yue et al., 2024) and MM-Vet (Yu et al., 2024) cover comprehensive reasoning challenges in real life scenarios, offering holistic insights into model performance. Others are developed with focuses on specific perspectives, such as TextVQA (Singh et al., 2019), POPE (Li et al., 2023b) and MATHVERSE (Zhang et al., 2024a) respectively challenge models with tasks in domains of reasoning about text, objects, mathematics in multimodal contexts. Recently, additional benchmarks have emerged targeting artificially created multipanel images—such as posters and screenshots—that combine several subfigures in structured layouts (Fan et al., 2024b; Hsiao et al., 2025), which require models to analyze spatial relationships and hierarchical structures in complex visual contexts. However, current multi-modal benchmarks assume visual-text alignment, overlooking detecting critical errors of vision-language inconsistency in the input - a key challenge in real-world scenarios. Instead, we evaluate MLLMs' ability to detect and localize such inconsistency via the proposed MMIR benchmark.

Inconsistency Checking Existing works on tasks related to checking or verifying inconsistency in the input are primarily in the language domain. For example, fact-checking (Thorne et al., 2018) requires a model to first retrieve evidence and then decide if a claim is supported, where the model must reason if contradictive information existed in the retrieved corpus. One step further, summary inconsistency detection (Laban et al., 2022) focuses on flagging any errors in summaries that create contradictions regardless of correctness, including incorrect use or hallucination of entities. As modern language models prosper, inconsistencies are found existing within their outputs (Ravichander et al., 2020) and across different outputs of paraphrased queries (Elazar et al., 2021), and efforts have been made towards the evaluation of those inconsistencies (Fabbri et al., 2021; Wang et al., 2020; Lattimer et al., 2023). In our research, we lead efforts in detecting inconsistencies in the field of vision and language.

3 MMIR

The MMIR benchmark is designed to assess how effectively MLLMs can detect and localize semantic mismatches within complex, layout-rich artifacts. Unlike conventional benchmarks that assume coherent visual–textual inputs, MMIR challenges models with realistic errors that require deep, cross-modal reasoning. In MMIR, errors are defined and categorized along five semantic dimensions:

A. Factual Contradiction: Direct conflict between two elements (text–text, text–image, or image–image) within the modified content.

B. Identity Misattribution: Mislabeling of entities (objects, locations, brands, people) that conflict with other elements.

C. Contextual Mismatch: Tonal, thematic, or situational incompatibility between elements.

D. Quantitative Discrepancy: Numerical or statistical inconsistencies between elements.

E. Temporal/Spatial Incoherence: Implied timelines, dates, or spatial relationships that are impossible or conflicting.

Figure 2 provides one example from each error type across web, office, and poster artifacts, illustrating the diverse challenges MMIR poses.

3.1 Data Curation

MMIR's data is curated through a four-stage pipeline (Figure 3), ensuring high-quality, diverse, and challenging test cases.

Artifact Collection and Parsing We begin by manually selecting a total of 521 original artifacts from two domains: 349 webpages (sub-

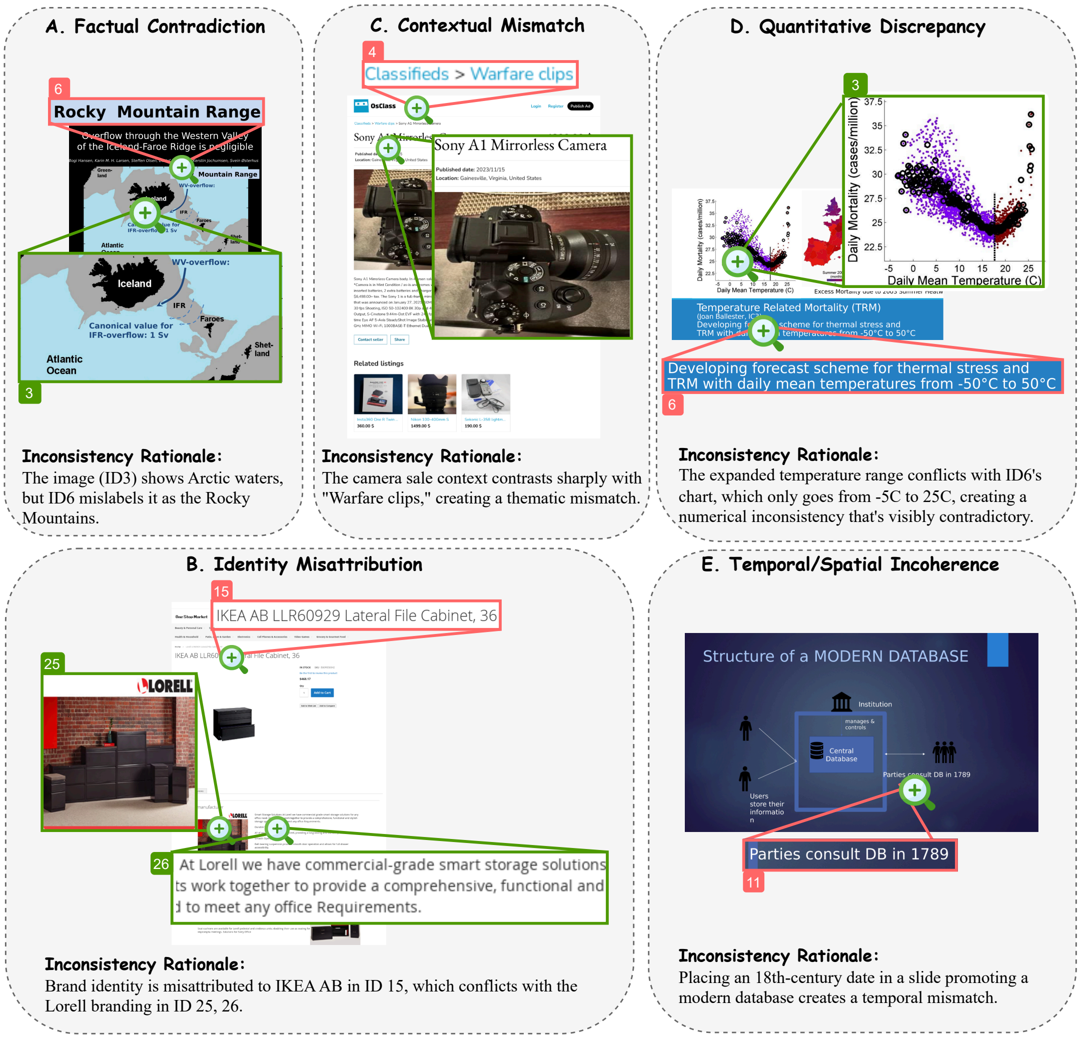


Figure 2: There are five inconsistency categories in the MMIR benchmark, posing diverse challenges.

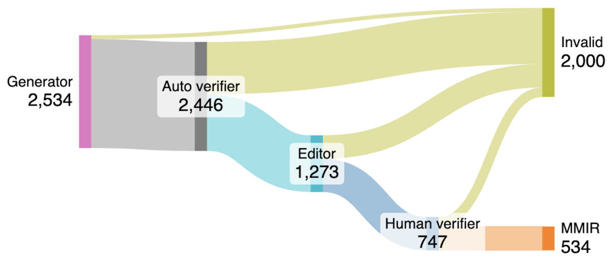


Figure 3: MMIR Data filtering process.

categories: shopping, classifieds, wiki) from VisualWebArena (Koh et al., 2024) and 172 presentations from Zenodo (European Organization For Nuclear Research and OpenAIRE, 2013), categorized into Office (sub-categories: slides, charts, diagrams) and Posters. Each artifact A_i is parsed using either using Document Object Model (DOM) or the `python-pptx` library to extract a set of elements $E_i = \{e_j\}_{j=1}^{n_i}$, where each element $e_j$is assigned a unique ID $\texttt{id}_j$ and labeled with its type, content, and a bounding box showing location information. Additionally, each artifact is paired with a Set-of-Marks (SoM) annotation A_i^{SoM} derived from E_i. This structured metadata forms the basis for subsequent error injection and question-answer curation.

Synthetic Inconsistency Generation To simulate real-world errors, we prompt an MLLM, o1-1217 (OpenAI, 2024b), as a generator with the annotated artifact and its element set $\{A_i^{\text{SoM}}, E_i\}$. The generator produces 2,534 proposals, each comprising a formatted edit instruction, the ground-truth element or element pair introducing the inconsistency:

$$\texttt{GT} \in \{\texttt{id}_j\} \cup \{(\texttt{id}_j, \texttt{id}_k) | j \neq k\},$$

the inconsistency error type, and the accompanying rationale. Following a self-evaluation loop (details in Appendix A.2), 2,446 valid proposals are retained.

Table 1: **MMIR Statistics.** Breakdown of the dataset by artifact category and error type.

Category	#Questions	Ave. #Elements
Artifact Categories		
Web	240	38.8
- Shopping	108	46.1
- Wiki	28	44.9
- Classifieds	104	29.5
Office	223	9.1
- Slides	102	9.4
- Tables/Charts	61	4.1
- Diagrams	60	13.9
Poster	71	27.6
Total	543	24.9
Error Categories		
Factual Contradiction	138	–
Identity Misattribution	84	–
Contextual Mismatch	141	–
Quantitative Discrepancy	76	–
Temporal/Spatial Incoherence	95	–
Total	543	–

Automated Editing and Human Verification An auto-verification process then filters these proposals based on format and backend constraints (e.g., ensuring the target elements are editable), reducing the candidate set to 1,273, and saves low-level edit details, such as the path of the new image for an image edit, as inputs to the editor.

An automated editor-implemented using the Chrome DevTools Protocol (CDP) for web pages and `python-pptx` for presentations-executes the approved edits, generating for each successful operation a modified pair: $\{A'_i, E'_i\}$ where A'_i represents the modified artifact and E'_i contains the updated element metadata after the edit. For each pair, a descriptive caption set C_i is generated, where each caption within C_j details the element ID, location, and content summary of e'_j. These captions serve as references for later evaluation. More details on the verifier and editor are provided in Appendix A.3.

Finally, human experts review 747 edited samples, resulting in a final dataset of 534 validated quintuples: $D_{MMIR} = \{S'_i, E'_i, \mathtt{GT}_i, \mathtt{category}_i, \mathtt{rationale}_i\}_{i=1}^{534}$, ensuring that only realistic and challenging samples remain. Table 1 provides a detailed breakdown by artifact type, subcategory, and error type. For example, webpages are further divided into shopping, wiki, and classifieds, each with its average number of elements, while errors are distributed across the five defined categories. Notably, the average word count in multiple-choice questions is 382.6, whereas open-ended responses are fixed at 59 words.

3.2 Evaluation

MMIR assesses a model's ability to *detect inconsistency*, i.e., identifying and localizing semantic mismatches where elements deviate from their expected roles within an artifact. To assess the model's performance comprehensively, each of the 534 test samples is provided to models under two distinct settings:

Open-Ended Setting Models receive the artifact A'_i with a fixed prompt $Q_{\text{open\_ended}}$ and generate a free-form response that identifies the semantic mismatch. This formulation evaluates the model's ability to detect inconsistencies without relying on predefined answer options, thereby testing its unsupervised perception and reasoning.

Multiple-Choice Setting Models receive the artifact A'_i, but now with a combined prompt $Q_{\text{MCQ}} = (Q_{\text{open\_ended}}, C_i)$. Each candidate in C_i is a textual description of an element. The model must select, from these options, the element(s) corresponding to the introduced inconsistency.

Evaluation Setup For the MCQ setting, we utilize regular expressions to compare the MLLM's predicted answers against the ground truth, using accuracy as our metric. For the open-ended setting, o1-mini (0912) is employed as an LLM judge (Hsu et al., 2023; Hackl et al., 2023; Liu et al., 2023) to map the model's free-form response back to the most likely ground-truth element IDs. The predicted IDs are then compared against $\mathtt{GT}_i$ to calculate accuracy.

4 Experiments and Analysis

We first evaluate the advanced multimodal reasoning model o1 (OpenAI, 2024b) and five other state-of-the-art MLLMs: GPT-4o (OpenAI, 2024a), Qwen2.5-VL (Team, 2025), LLaVA-NeXT (Liu et al., 2024b), InternVL2.5 (Chen et al., 2024) and Phi-3.5-Vision (Abdin et al., 2024) on the MMIR benchmark. We implement open-source models using their default settings and select the 1217 version of o1 and the 1120 version of GPT-4o for evaluation. Model implementation details are provided in Appendix B. We then examine error patterns across different inconsistency types and layout complexities and finally explore how prompting strategies affect multimodal reasoning under the open-ended setting.

Table 2: **The accuracy of six MLLMs under the two evaluation settings.** Proprietary models demonstrate higher performance as well as larger performance gain in the MCQ setting.

	Open-ended				Multiple-choice			
Models	Web	Office	Poster	Overall	Web	Office	Poster	Overall
Proprietary Models								
o1 (1217)	47.91	59.19	38.73	51.40	47.91	58.52	46.47	52.15
GPT-4o (1120)	25.00	42.60	30.98	33.14	37.29	58.96	47.88	47.75
Open-sourced Models								
Qwen2.5-VL-7B	8.54	29.14	11.97	17.60	14.37	33.18	16.90	22.56
LLaVA-NeXT-7B	10.20	21.97	7.04	14.70	11.45	25.33	5.63	16.47
InternVL2.5-8B	7.70	24.21	4.92	14.23	9.37	23.54	11.97	15.63
Phi-3.5-Vision-4B	6.87	24.43	7.04	14.23	1.66	8.52	0.00	4.30

4.1 Main Results

As shown in Table 2, proprietary models (o1 and GPT-4o) significantly outperform open-source alternatives, though all models exhibit substantial room for improvement. Appendix A.4 shows a qualitative example with question-answer and model response.

Performance Gap Between Reasoning, Proprietary and Open-Source Models. In both open-ended and MCQ settings, the reasoning o1 model substantially outperforms the rest, surpassing all open-source models by over 30%. The other proprietary model GPT-4o, although missing the explicit reasoning ability of o1, outperforms open-source alternatives, reflecting stronger multimodal alignment and reasoning capabilities.

Impact of Semantic Cues. GPT-4o sees a 14.61% accuracy boost in the MCQ setting with additional element descriptions as options, narrowing its gap with o1 from 18.26% to just 4.4%. This indicates that GPT-4o relies heavily on semantic context when available.

Inconsistent Gains for Open-Source Models. Most open-source models gain moderate or little accuracy when provided with MCQ-style prompts. Phi-3.5-Vision-4B experiences a 9.93% drop, suggesting weaker reasoning capacity and less effective use of textual cues. The gap between proprietary and open-source models widens further in MCQ (from 27.08% to 35.21%), highlighting the persistent challenge of integrating perceptual grounding with logical inference.

4.2 Error Analysis

4.2.1 Results Across Inconsistency Categories and Modalities

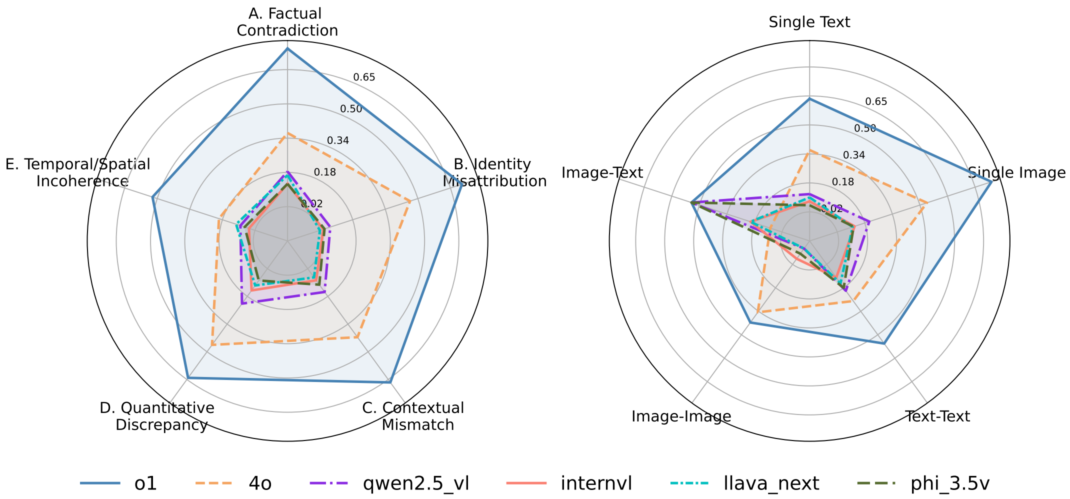
To investigate how different types of inconsistencies affect model performance, we show the results across the category and modality of inconsistency in Figure 4.

Inconsistency Categories Figure 4(a) breaks down accuracy by the five inconsistency error categories. Proprietary models (o1, GPT-4o) outperform open-source models across the board, but the gap is particularly pronounced for *Factual Contradictions* and *Identity Misattribution*, implying that high-capacity models may have stronger factual grounding and entity recognition. Interestingly, *Temporal/Spatial Incoherence* also poses a substantial challenge for all models, highlighting a limitation in reasoning about time and space coherence.

Inconsistency Modalities In Figure 4(b), we examine how accuracy varies by the modality of the inconsistency. Overall, single-modality errors (those involving only one text or image field) yield the highest performance, with text-text inconsistencies proving especially tractable—likely because these language-centric models excel at purely textual reasoning. Next in difficulty are inter-modality errors (image-text), which require partial cross-modal integration but can still leverage textual anchors. Finally, image-image inconsistencies pose the greatest challenge, as they demand more advanced visual understanding and the ability to reconcile two distinct visual elements without the benefit of textual cues. These findings highlight that while language-focused models cope relatively well with purely textual conflicts, their capacity for deep visual or cross-modal reasoning remains underdeveloped.

4.2.2 Impact of Layout Complexity

We further examine the relationship between model accuracy and the number of elements in an artifact. To ensure statistical significance, we only include data points where at least 10 samples share



(a) Model performance on error category. (b) Model performance on error modality.

Figure 4: Fine-grained analysis of model performance.

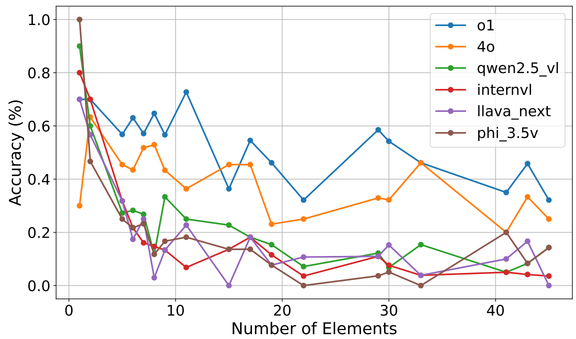


Figure 5: Model performance on layout complexity.

Table 3: **Probing results of different prompting methods.** Performance of each prompting method is directly compared with the vanilla setting. Gains are in blue and drops are in red.

Models	Vanilla	+ CoT	+ SoM	+ Both	MM-CoT
Proprietary Models					
o1 (1217)	51.40	–	-0.66	–	+0.09
GPT-4o (1120)	33.14	–	+5.34	–	+4.40
Open-sourced Models					
Qwen2.5-VL-7B	17.60	+0.28	+0.09	+0.28	+4.59
LLaVA-NeXT-7B	14.70	-1.78	-2.53	-0.47	+3.65
InternVL2.5-8B	14.23	+2.24	-0.66	-1.41	-0.85
Phi-3.5-Vision-4B	14.23	-0.38	+0.47	+0.84	+0.65

the same element count. As shown in Figure 7, the overall trend suggests that handling visually dense, information-rich artifacts remains a major challenge for current MLLMs. (1) Performance declines sharply as the number of elements increases, highlighting the difficulty in parsing cluttered layouts. (2) Proprietary models maintain higher accuracy in simpler layouts but degrade similarly in highly dense artifacts, indicating limitations in spatial reasoning. Open-source models struggle even in low-complexity settings, reinforcing the gap in perception and layout-aware inference.

4.3 Probing on Prompting Methods

We further investigate whether textual or visual prompts can alleviate the reasoning bottleneck. Table 3 compares *Chain-of-Thought (CoT)* prompting (Wei et al., 2022) and *Set-of-Mark (SoM)* visual augmentation (Yang et al., 2023), as well as their combination. We also explored an interleaved multimodal reasoning strategy, which we term *Multimodal Interleaved CoT (MM-CoT)* to further integrate and refine reasoning across both visual and textual modalities.

4.3.1 Chain-of-Thought (CoT) Prompting

To assess whether explicit reasoning instructions can enhance performance, we apply CoT prompting (Wei et al., 2022) to the four open-sourced models (benchmarked proprietary models have API guides to not include additional CoT prompting).

As shown in Table 3, CoT prompting yields negligible or even negative effects on accuracy. This suggests that simply injecting explicit reasoning steps is insufficient when the underlying model lacks strong cross-modal alignment or robust logical inference mechanisms.

4.3.2 Set-of-Mark (SoM) Prompting

We next examine the effect of SoM visual prompting (Yang et al., 2023). By overlaying bounding boxes onto the artifact screenshots (example in Figure 6), we aim to enhance the models' ability to perceive and localize elements.

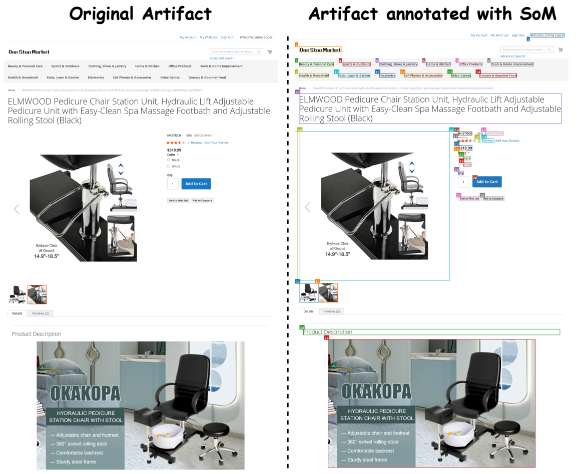


Figure 6: Example of original artifact in MMIR (left) and artifact annotated with Set-of-Mark in the probing analysis (right).

The result shows that these additional visual cues yield moderate improvements for GPT-4o (5.34%) yet confuse the rest of the models, leading to little or even slightly degraded performance, likely because the additional visual cues interfere with the model's initial perception.

When combined with CoT prompting, SoM provides little gains for some open-source models but remains largely inconsistent or even detrimental for others. This indicates that simply stacking CoT and SoM techniques does not guarantee improved performance, underscoring the need for more sophisticated strategies to unify visual cues with explicit reasoning steps.

4.3.3 Multimodal Interleaved CoT (MM-CoT)

Our previous analyses indicate that single-modality prompts (CoT or SoM) often yield minimal or even detrimental gains in the open-ended setting when models receive no textual hints about which elements might be inconsistent. We hypothesize that MMIR tasks demand *iterative* reasoning that tightly integrates both visual and textual modalities. To address this, we propose *Multimodal Interleaved CoT (MM-CoT)*, a two-stage approach explicitly designed to weave visual cues into a step-by-step reasoning process:

Stage 1: Initial Candidate Generation The model receives the same input in Stage 1 as in the open-ended setting, generating its top five predictions (along with associated reasoning). Using o1-mini (0912) to interpret these responses, we map each prediction back to one or a pair of element IDs from the artifact's metadata C_i. We then highlight the bounding boxes of those elements on the artifact image, producing an SoM-annotated version to be used in the next stage.

Stage 2: Multimodal Refinement The model is subsequently given the SoM-annotated artifact from Stage 1, alongside the textual reasoning it generated previously. This additional visual context helps the model refine its earlier predictions, integrating both the visual bounding-box annotations and the initial textual reasoning to arrive at a final answer.

Results As shown in Table 3, MM-CoT outperforms all other prompting methods. GPT-4o, for example, improves by 4.40% over its vanilla baseline, while open-source models gain an average of around 2% improvements. These findings underscore the importance of iterative cross-modal reasoning: once textual inferences guide which visual elements to focus on, SoM annotations become more informative, and the overall reasoning process becomes more accurate. Although the bounding boxes used for SoM are derived from ground-truth references, this probing experiment demonstrates that *interleaved* multimodal interaction is a promising direction for closing the reasoning gap in challenging, inconsistency-heavy scenarios.

5 Discussion and Conclusion

In this work, we introduce the *Multimodal Inconsistency Reasoning Benchmark (MMIR)* to evaluate how well MLLMs detect and localize semantic mismatches in complex real-world artifacts. MMIR challenges models across five error categories and two reasoning settings for a detailed assessment of multimodal reasoning. Our experiments show that even advanced proprietary models struggle with open-ended inconsistency detection. Although providing natural-language descriptions in a multiple-choice format offers modest gains, standard prompting techniques (e.g., Chain-of-Thought and Set-of-Mark) yield inconsistent or negative effects, while a proposed Multimodal Interleaved CoT (MM-CoT) method that iteratively refines reasoning by integrating visual and textual modalities, yielding greater performance improvements. Despite these advances, significant challenges remain, motivating further research on robust multimodal reasoning for real-world inconsistency detection.

Limitations

While MMIR provides a rigorous framework for evaluating multimodal inconsistency reasoning, it is not without its limitations. Annotating and verifying inconsistencies in layout-rich artifacts remains a labor-intensive process. Although MMIR's pipeline integrates automated editing and verification, the overall scale is still limited by the need for careful human review. Although these domains capture a range of layouts and content types, they do not encompass the full variety of real-world multimodal artifacts (e.g., multi-page documents, social media feeds, or mobile application interfaces). On the other hand, synthetic error generation—while effective for systematically introducing controlled inconsistencies—may not perfectly mirror the nuanced mistakes that occur in human-generated content. This could lead to discrepancies between model performance on MMIR and in truly open-ended, real-world scenarios. Scaling up the dataset to cover broader domains, more intricate layouts, and diverse error types would strengthen its ability to serve as a comprehensive benchmark for real-world multimodal inconsistency detection.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 1(2):3.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. 2023. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yiming Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Hui Deng, Jiaye Ge, Kaiming Chen, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahu Lin, Yunfeng Qiao, Jifeng Dai, and Wenhai Wang. 2024. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *ArXiv*, abs/2412.05271.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

European Organization For Nuclear Research and OpenAIRE. 2013. Zenodo.

Alexander R Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2021. Qafacteval: Improved qa-based factual consistency evaluation for summarization. *arXiv preprint arXiv:2112.08542*.

Yue Fan, Lei Ding, Ching-Chen Kuo, Shan Jiang, Yang Zhao, Xinze Guan, Jie Yang, Yi Zhang, and Xin Eric Wang. 2024a. Read anywhere pointed: Layout-aware gui screen reading with tree-of-lens grounding. *arXiv preprint arXiv:2406.19263*.

Yue Fan, Jing Gu, Kaiwen Zhou, Qianqi Yan, Shan Jiang, Ching-Chen Kuo, Yang Zhao, Xinze Guan, and Xin Wang. 2024b. Muffin or chihuahua? challenging multimodal large language models with multipanel vqa. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6845–6863.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. 2024. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36.

Luciano Floridi and Massimo Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Veronika Hackl, Alexandra Elena Müller, Michael Granitzer, and Maximilian Sailer. 2023. Is gpt-4 a reliable rater? evaluating consistency in gpt-4's text ratings. In *Frontiers in Education*, volume 8, page 1272229. Frontiers Media SA.

Yu-Chung Hsiao, Fedir Zubach, Gilles Baechler, Srinivas Sunkara, Victor Carbune, Jason Lin, Maria Wang, Yun Zhu, and Jindong Chen. 2025. Screenqa: Large-scale question-answer pairs over mobile app screenshots. *Preprint*, arXiv:2209.08199.

Ting-Yao Hsu, Chieh-Yang Huang, Ryan Rossi, Sungchul Kim, C Lee Giles, and Ting-Hao K Huang. 2023. Gpt-4 as an effective zero-shot evaluator for scientific figure captions. *arXiv preprint arXiv:2310.15405*.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. 2024. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Philippe Laban, Tobias Schnabel, Paul N Bennett, and Marti A Hearst. 2022. Summac: Re-visiting nli-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Barrett Martin Lattimer, Patrick Chen, Xinyuan Zhang, and Yi Yang. 2023. Fast and accurate factual inconsistency detection over long documents. *arXiv preprint arXiv:2310.13189*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating object hallucination in large vision-language models. *Preprint*, arXiv:2305.10355.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024b. Llava-next: Improved reasoning, ocr, and world knowledge.

Junpeng Liu, Tianyue Ou, Yifan Song, Yuxiao Qu, Wai Lam, Chenyan Xiong, Wenhu Chen, Graham Neubig, and Xiang Yue. 2024c. Harnessing webpage uis for text-rich visual understanding. *Preprint*, arXiv:2410.13824.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. 2024d. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer.

OpenAI. 2024a. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

OpenAI. 2024b. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. 2023. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*.

Ji Qi, Ming Ding, Weihan Wang, Yushi Bai, Qingsong Lv, Wenyi Hong, Bin Xu, Lei Hou, Juanzi Li, Yuxiao Dong, et al. 2024. Cogcom: Train large vision-language models diving into details through chain of manipulations. *arXiv preprint arXiv:2402.04236*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in bert. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. 2024. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Amanpreet Singh, Vivek Natarjan, Meet Shah, Yu Jiang, Xinlei Chen, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can read. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8317–8326.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Gemini Team. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Qwen Team. 2025. Qwen2.5-vl.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. *arXiv preprint arXiv:2004.04228*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Penghao Wu and Saining Xie. 2024. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyue Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *ArXiv*, abs/2310.11441.

Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2024. Mm-vet: Evaluating large multimodal models for integrated capabilities. *Preprint*, arXiv:2308.02490.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of CVPR*.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Peng Gao, and Hongsheng Li. 2024a. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? *Preprint*, arXiv:2403.14624.

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

Yuechen Zhang, Shengju Qian, Bohao Peng, Shu Liu, and Jiaya Jia. 2024b. Prompt highlighter: Interactive control for multi-modal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13215–13224.

Ge Zheng, Bin Yang, Jiajin Tang, Hong-Yu Zhou, and Sibei Yang. 2023. Ddcot: Duty-distinct chain-of-thought prompting for multimodal reasoning in language models. *Advances in Neural Information Processing Systems*, 36:5168–5191.

A Benchmark Details

This appendix provides a comprehensive overview of the MMIR benchmark. It details the dataset curation process, including error category definitions, the synthetic inconsistency generation mechanism, the auto-verification and human validation processes, and the task prompts for evaluation. These details are intended to facilitate reproducibility and provide clarity on the inner workings of MMIR.

A.1 Inconsistency Error Category Definitions

The MMIR benchmark employs five pre-defined error categories. These categories are designed based on semantic guidelines so that the generator model can propose diverse and generalizable inconsistencies without being tied to any specific artifact type.

- **A. Factual Contradiction**

  Direct conflict between two or more elements (text–text, text–image, or image–image) within the modified content.

  Example (Text–Text): The product title says "Caffeinated," while the description states "Caffeine-free."
  Example (Text–Image): The image shows a green tea bag, but the accompanying text describes a "fruit infusion."

- **B. Identity Misattribution**

  Mislabeling of entities (objects, locations, brands, people) that conflict with other elements.

  Example: A product lists "Country of Origin: China" while the manufacturer is described as "Elmwood Inn (USA)."

- **C. Contextual Mismatch**

  Tonal, thematic, or situational incompatibility between elements.

  Example: A celebratory image of diplomats shaking hands is paired with an article about violent clashes.

- **D. Quantitative Discrepancy**

  Numerical or statistical inconsistencies between elements.

  Example: A graph labeled "50% growth" shows flat bars.

- **E. Temporal/Spatial Incoherence**

  Implied timelines, dates, or spatial relationships that are impossible or conflicting.

  Example: A map labeled "North America" depicts landmarks from Europe.

These definitions serve as guidelines during the synthetic inconsistency generation process, ensuring that the proposed errors are semantically meaningful and cover a broad spectrum of potential real-world mistakes.

A.2 Generator Model and Self-Evaluation Loop

A.2.1 Generator Model Prompt

To create adversarial examples, the generator model (o1, 1217) is provided with rich context consisting of the annotated artifact A_i^{SOM} and its set of elements E_i. The task prompt includes detailed instructions regarding the types of modifications to propose, along with the following guidelines:

- **Modification Format:** Each modification must be expressed as:

  "'Modify [id] [original_content] [new_content]"'

For image fields, the original content includes the full details (e.g., URL), and the new content is a caption starting with "Image, description: ". For text fields, the new content should be of similar length to the original.

- **Error Categories:** The generator must propose one modification per error category. If it cannot propose an inconsistency for a given category, it may skip that category.

The generator output is structured as:

$$P_m = \{ \text{ edit }_m, \mathrm{GT}_m, \text{ category }_m, \text{ rationale }_m\}$$

where the ground-truth GT_m is defined as:

$$\mathrm{GT}_m \in \{\mathrm{id}_j\} \cup \{(\mathrm{id}_j, \mathrm{id}_k) \mid j \neq k\}$$

indicating either a single-element ID (for single-element inconsistencies) or a pair of distinct element IDs (for relational inconsistencies).

A.2.2 Self-Evaluation Loop

We follow a generator-evaluator loop that refines proposals through iterative self-assessment. A simplified Python snippet of the loop function is provided below:

```python
def loop(client, image_dir, frame_id, task: str, evaluator_prompt: str,
    generator_prompt: str) -> tuple[str, list[dict]]:
    """Keep generating and evaluating until requirements are met."""
    memory = []
    chain_of_thought = []

    thoughts, result = generate(client, image_dir, frame_id, generator_prompt, task)
    memory.append(result)
    chain_of_thought.append({"thoughts": thoughts, "result": result})

    loop_count = 1
    while True:
        all_pass = True
        evaluation, feedback = evaluate(client, image_dir, frame_id,
            evaluator_prompt, result, task)
        for eval_line in evaluation.split("\n"):
            if eval_line.strip() != "PASS":
                all_pass = False
                break
        if all_pass or loop_count == 2:
            return result, evaluation

        context = "\n".join([
            "Previous attempts:",
            *[f"- {m}" for m in memory],
            f"\nFeedback: {feedback}"
        ])
        thoughts, result = generate(client, image_dir, frame_id, generator_prompt,
            task, context)
        memory.append(result)
        chain_of_thought.append({"thoughts": thoughts, "result": result})
        loop_count += 1
```

In this loop, the generator produces proposals which are then evaluated against the following criteria (as specified in the evaluator prompt):

- **Category Compliance:** The edit must match the intended error category.

- **Atomic Modification:** Exactly one inconsistency should be introduced.

- **Visual Consistency:** The modified screenshot must visibly reflect the error without relying on external context.

- **Element Validity:** The referenced element IDs must exist in the artifact.

Only proposals receiving a "PASS" in the evaluation are retained. The loop iterates until either all criteria are met or a maximum of two iterations is reached.

A.2.3 Prompt details for generator-evaluator proposal generation framework

This is the task prompt as input to the o1 generator model.

```
    task_prompt = f"""
<user input>
Your task is to modify a {category_str} to create inconsistency. For each given
    category of inconsistency, you will propose a modification action that
    introduces the inconsistency in the modified {category_str}.

Here's the information you'll have:
Screenshot of the urrent {category_str}: This is a screenshot of the {category_str},
     with each editable element assigned a unique numerical id. Each bounding box
    and its respective id share the same color.
The Observation, which lists the IDs of all editable elements on the current {
    category_str} with their content, in the format [id] [tagType] [content],
    separated by "\n". Each id is mapped with the id in the screenshot. tagType is
    the type of the element, such as button, link, or textbox. For example, "[21] [
    SPAN] [Add to Wish List]" means that there is a span with id 21 and text content
     'Add to Wish List' on the current {category_str}. "[23] [IMG] [Image,
    description: a beige powder on a white background, url: http://localhost:7770/
    media/catalog/product/cache/829a59e57f886f8cf0598ffca4f8a940/B/0/B074DBMG66.0.
    jpg]" means that there is an image on the current screen with id 23, with a
    description of the image and its url specified.

Here are the categories of errors you can introduce:
A. Factual Contradiction -  Direct conflict between two or more elements (text-text,
     text-image, or image-image). For example, The product title says "Caffeinated,"
     while the description states "Caffeine-free." Another example: The image shows
    a green tea bag, but the text describes a "fruit infusion."
B. Identity Misattribution - Mislabeling of entities (objects, locations, brands,
    people) that conflict with other elements. Example: Product "Country of Origin:
    China" contradicts manufacturer info "Elmwood Inn (USA)."
C. Contextual Mismatch - Tonal, thematic, or situational incompatibility between
    elements. Example: A celebratory image of diplomats shaking hands paired with an
     article about violent clashes.
D. Quantitative Discrepancy - Numerical or statistical inconsistencies between
    elements. Example: A graph labeled "50%\ growth" shows flat bars.
E. Temporal/Spatial Incoherence - Implied timelines, dates, or spatial relationships
     that are impossible or conflicting. Example: A map labeled "North America"
    depicts landmarks from Europe

Here are the rules for the modification action:
The modification action you can propose to introduce inconsistency must be in the
    format of "Modify [id] [original_content] [new_content]": This action proposes
    to edit the orignal field assigned with the id to the new content to introduce
    inconsistency. If you propose to modify an image field, the [original_content]
    field should include the full content from observation including the url; the [
    new_content] field should be a caption describing the updated image, starting
    with "Image, description: ", no url needed. If you propose to modify a text
    field, the new content string should be about the same length as the original
    text field. For each inconsistency category, you should try to propose a
    modification action that introduces an inconsistency in that category. If you
    can't find a way to introduce an inconsistency in a category, you can skip it.
    Prioritize proposing edits on text fields over image fields.

Generate the response in the correct format. For each inconsistency, the format
    should be:
<proposal>
  <cat>[A-E]</cat> <-- Category letter
  <ele>[ID1,ID2]</ele> <-- Conflicting element IDs
  <mod>Modify [ID] [Original Content] [New Content]</mod> <-- Modification plan
  <rationale>Visible conflict explanation</rationale> <-- Visual verification
</proposal>
</user input>
```

```
"""
```

These are prompts for the generator and evaluator model.

```
    evaluator_prompt = """
Evaluate the following proposals one by one for:
1. Category Compliance: Introduced inconsistency matches the category definition (A-
   E)
2. Atomic Modification: Introduce EXACTLY ONE inconsistency without side effects
3. Visual Consistency: Conflict visible in the modified screenshot (with NO reliance
    on original page knowledge or external context)
4. Element Validity: Conflict IDs exist in observations

You should be evaluating only and not attemping to solve the task.
For each proposal, only output "PASS" if all criteria are met and you have no
   further suggestions for improvements.
Output your evaluation concisely in the following format.

<evaluation>
PASS, NEEDS_IMPROVEMENT, or FAIL <-- For each proposal
</evaluation>
<feedback>
What needs improvement and why. <-- For proposals that need improvement
</feedback>
"""

    generator_prompt = """
Your goal is to complete the task based on <user input>. If there are feedback
from your previous generations, you should reflect on them to improve proposals that
    NEEDS_IMPROVEMENT or FAIL. Leave the PASS proposals as they are.

Output your answer concisely in the following format:

<thoughts>
[Your understanding of the task and feedback and how you plan to improve]
</thoughts>

<response>
[Your response here]
</response>
"""
```

A.3 Auto-Verification and Editing Process

Following proposal generation, an auto-verification step filters the proposals based on format and backend constraints. Specifically:

- **Edit Format Verification:** The system uses a regular expression to ensure that each proposed edit adheres to the required format: "Modify [id] [old_content] [new_content]".
- **Element Matching:** For web-sourced artifacts, the proposal's element ID is used to locate the corresponding element and its bounding box in the metadata. The system checks that both the content and bounding box match an editable element in the HTML/PPTX structure. For image edits, the new content (a caption) is cross-referenced against an MSCOCO image database to verify its appropriateness.

  Proposals that pass these checks are automatically saved for further processing.

For web pages, we use the CDP to perform edit:

```
# text edit
client.send(
    "Runtime.callFunctionOn",
    {
        "objectId": object_id,
        "functionDeclaration": f"function() {{ this.nodeValue = '{new_content}'; }}"
            ,
```

```
        "arguments": [],
        "returnByValue": True
    }
)
# image edit
with open(new_content, "rb") as image_file:
    img = Image.open(image_file)
    new_image_width, new_image_height = img.size  # get original width and height for resizing
    aspect_ratio = new_image_width / new_image_height
    if w / h > aspect_ratio:
        w, h = w, int(w / aspect_ratio)
    else:
        w, h = int(h * aspect_ratio), h
    img = img.resize((w, h), Image.Resampling.LANCZOS)
    buffer = BytesIO()
    img.save(buffer, format="JPEG")
    buffer.seek(0)
    base64_image = base64.b64encode(buffer.read()).decode("utf-8")
    new_image = f"data:image/jpeg;base64,{base64_image}"
client.send(
    "Runtime.callFunctionOn",
    {
        "objectId": object_id,
        "functionDeclaration": f"""
            function() {{
                this.src = '{new_image}';
            }}
            """,
        "arguments": [],
        "returnByValue": True
    }
)
```

For Zenodo presentation, we use the python-pptx library:

```
# text edit
if target_shape.has_text_frame: # text edit
    text_frame = target_shape.text_frame
    for paragraph in text_frame.paragraphs:
        for run in paragraph.runs:
            if edit_info["old_content"] in run.text:
                try:
                    run.text = run.text.replace(edit_info["old_content"], edit_info["new_content"])
                    success = True
                    break
                except:
                    success = False
# image edit
left, top, orig_width, orig_height = target_shape.left, target_shape.top, target_shape.width, target_shape.height
pic = target_shape._element
pic.getparent().remove(pic)
new_image_path = f"{coco_image_dir}/{edit_info['new_img_path']}"
with Image.open(new_image_path) as img:
    new_width, new_height = img.size
new_aspect = new_width / new_height
orig_aspect = orig_width / orig_height
if new_aspect > orig_aspect:
    scaled_width = orig_width
    scaled_height = int(scaled_width / new_aspect)
else:
    scaled_height = orig_height
    scaled_width = int(scaled_height * new_aspect)
new_left = left + (orig_width - scaled_width) // 2
new_top = top + (orig_height - scaled_height) // 2
try:
    slide.shapes.add_picture(  # Add the new image in the same location and size
        new_image_path, new_left, new_top, scaled_width, scaled_height
```

```
    )
    success = True
except:
    success = False
```

A.4 A Qualitative Example

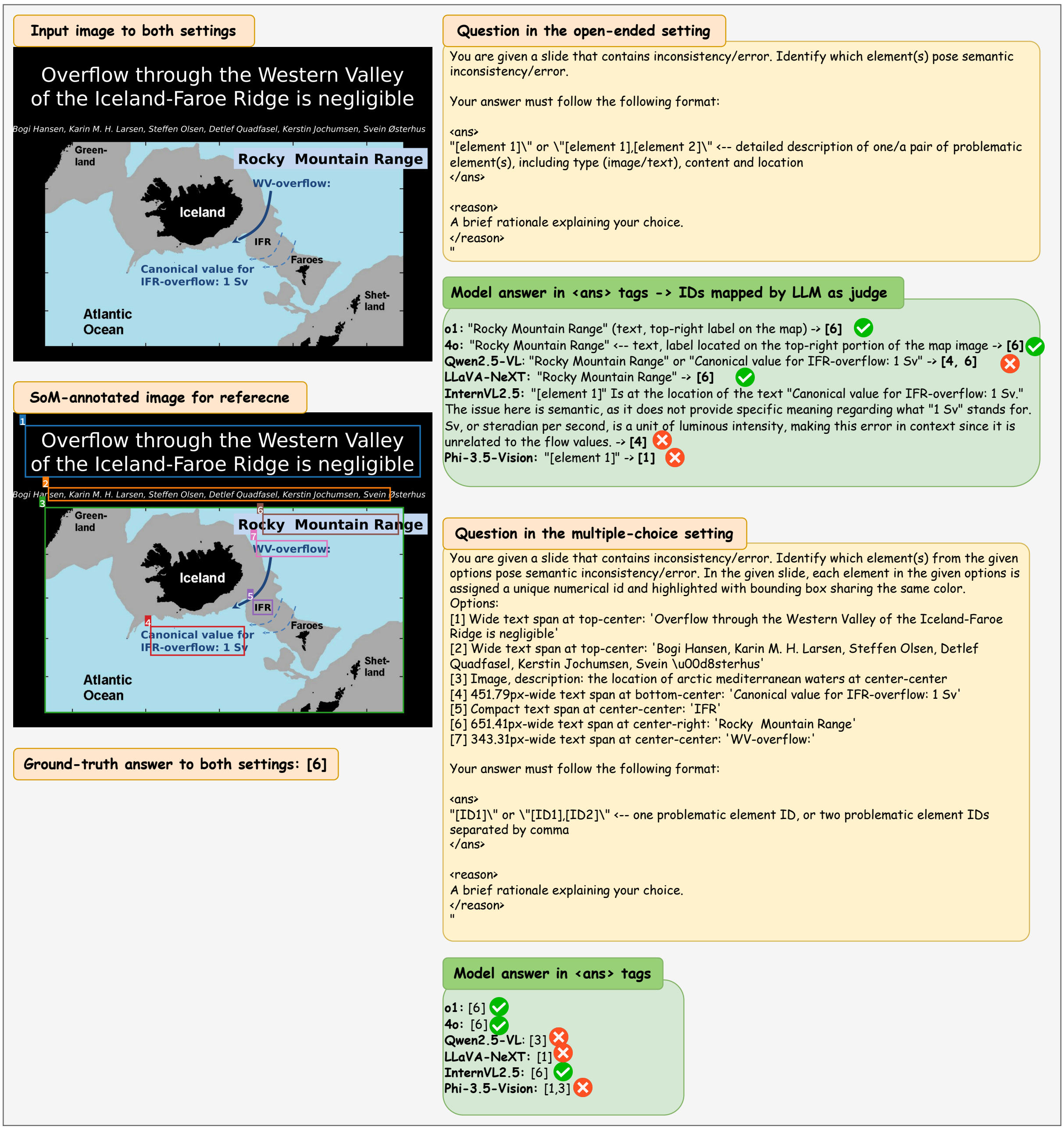


Figure 7: A test sample with model responses under the two main settings in MMIR: open-ended and multiple-choice.

B Model Application Details

Here are the generation methods for the open-sourced models.

For **o1** and **GPT-4o**, we utilized the API following API guidelines available at `https://platform.openai.com/docs/models#gpt-4o`.

For **Qwen2.5-VL**, we implemented the 7B version following the official repository: `https://github.com/QwenLM/Qwen2.5-VL`.

For **LLaVA-NeXT**, we followed the implementation from https://github.com/LLaVA-VL/LLaVA-NeXT.

For **InternVL2.5** we implemented the 8B version at https://github.com/OpenGVLab/InternVL.

For **Phi-3.5-Vision** we implemented the 4B version at https://github.com/instill-ai/models/tree/main/phi-3-5-vision.

C Data Release

We will publicly release a comprehensive dataset that includes the artifacts and question-answer pairs in both the open-ended and multiple-choice settings. The licensing terms for the artifacts will follow those set by the respective dataset creators, as referenced in this work, while the curated artifacts will be provided under the MIT License. Additionally, our release will include standardized evaluation protocols, and evaluation scripts to facilitate rigorous assessment. The entire project will be open-sourced, ensuring free access for research and academic purposes.