TRACER: Verifiable Generative Provenance for Multimodal Tool-Using Agents

Bihui Yu[1,2], **Caijun Jia**[1,2], **Jing Chi**[4], **Xiaohan Liu**[4], **Yining Wang**[3], **He Bai**[3], **Yuchen Liu**[3], **Jingxuan Wei**[1,2,4*], **Junnan Zhu**[3*]

[1] Shenyang Institute of Computing Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] MAIS, Institute of Automation, Chinese Academy of Sciences
[4] Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences)

yubihui@sict.ac.cn, junnan.zhu@nlpr.ia.ac.cn

Abstract

Multimodal large language models (MLLMs) increasingly solve vision-centric tasks by calling external tools for visual inspection, OCR, retrieval, calculation, and multi-step reasoning. Current tool-using agents usually expose the executed tool trajectory and the final answer, but they rarely specify which tool observation supports each generated claim. We call this missing claim-level dependency structure the *provenance gap*. The gap makes tool use hard to verify and hard to optimize, because useful evidence, redundant exploration, and unsupported reasoning are mixed in the same trajectory. We introduce TRACER, a framework for verifiable generative provenance in multimodal tool-using agents. Instead of adding citations after generation, TRACER generates each answer sentence together with a structured provenance record that identifies the supporting tool turn, evidence unit, and semantic support relation. Its relation space contains QUOTATION, COMPRESSION, and INFERENCE, covering direct reuse, faithful condensation, and grounded derivation. TRACER verifies each record through schema checking, tool-turn alignment, source authenticity, and relation rationality, and then converts verified provenance into traceability constraints and provenance-derived local credit for reinforcement learning. We further construct TRACE-BENCH, a benchmark for sentence-level provenance reconstruction from coarse multimodal tool trajectories. On TRACE-BENCH, simply adding tools often introduces noise. With Qwen3-VL-8B-Instruct, TRACER reaches 78.23% answer accuracy and 95.72% summary accuracy, outperforming the strongest closed-source tool-augmented baseline by 23.80 percentage points. Compared with tool-only supervised fine-tuning, it also reduces total test-set tool calls from 4,949 to 3,486. These results show that reliable multimodal tool reasoning depends on provenance-aware use of observations, not on more tool calls alone.

1 Introduction

MLLMs are increasingly used as tool-grounded agents for vision-centric tasks that require visual inspection, OCR, retrieval, calculation, and multi-step reasoning [1, 2, 3]. Recent multimodal

*Corresponding author.

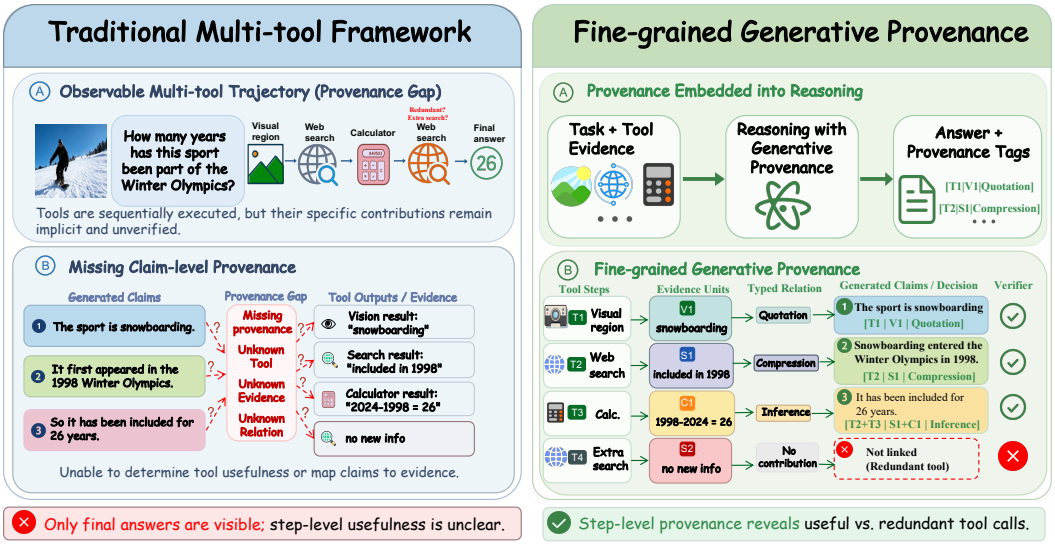


Figure 1: Overview of the provenance gap in multi-tool reasoning. Standard tool-using agents expose a tool trajectory and a final answer, but the claim-level dependency between tool observations and generated statements remains implicit. TRACER makes this dependency explicit by linking answer claims to tool turns, evidence units, and typed support relations, which separates provenance-bearing tool calls from redundant exploration.

search and deep-research agents extend this paradigm by training models to acquire and integrate information across modalities and tools [4, 5, 6, 7]. However, most tool-using MLLMs still provide only a trajectory-level view of reasoning: they record which tools were called and what final answer was produced. They do not make explicit how individual answer claims depend on specific tool observations. We refer to this missing claim-level dependency structure as the *provenance gap*.

As shown in Figure 1, the provenance gap weakens both verification and learning. A single trajectory can contain useful observations, failed or redundant calls, and unsupported reasoning steps. If the final answer does not state which observations support which claims, a correct response may come from faithful tool use, parametric memory, accidental guessing, or ungrounded inference. This ambiguity also creates a credit-assignment problem. Answer-level or trajectory-level supervision cannot identify which tool turns actually contributed evidence, nor can it distinguish useful observations from noisy exploration. The problem is especially acute in multimodal settings, where support may appear as image regions, OCR spans, retrieved snippets, visual descriptions, object counts, or computed values. Reliable tool reasoning therefore requires an explicit provenance layer that makes generated claims traceable to the tool observations from which they are derived.

To bridge this gap, we introduce TRACER, a framework for verifiable generative provenance in multimodal tool-using agents. TRACER treats provenance as part of generation rather than as a post-hoc explanation. For each generated sentence, it emits a structured provenance record that specifies the supporting tool turn, the support unit, and the semantic relation between the tool observation and the claim. The relation space contains QUOTATION, COMPRESSION, and INFERENCE, which respectively capture direct reuse, faithful condensation, and grounded derivation. Thus, a valid provenance record must identify not only *where* a claim is grounded, but also *how* the cited observation supports it.

TRACER further makes provenance an optimization interface. Generated records are checked for structural validity, tool-turn alignment, source authenticity, and relation rationality. Verified provenance is then used to identify provenance-bearing tool calls and assign local credit during group-relative reinforcement learning. The resulting objective rewards correct and traceable answers, penalizes fabricated or unsupported provenance, and discourages unnecessary tool exploration.

To train and evaluate this capability, we construct TRACE-BENCH for provenance-aware multimodal multi-tool reasoning. Starting from coarse tool-interaction trajectories, TRACE-BENCH reconstructs sentence-level reasoning paths with structured JSON provenance. Each answer sentence is linked to a supporting tool turn, an evidence unit, and a typed relation. The benchmark evaluates final answer

accuracy, summary accuracy (SummAcc.), traceability accuracy, provenance F1, tool errors, and total tool-call usage, thereby measuring both task success and the faithfulness of claim-level grounding.

Experiments show that tool access alone does not reliably improve strong models. Without provenance-aware training, additional tools can introduce noise and amplify planning errors. In contrast, a Qwen3-VL-8B-Instruct instantiation of TRACER reaches 78.23% accuracy and 95.72% SummAcc., outperforming the strongest closed-source tool-augmented baseline by 23.80 percentage points. Compared with tool-only supervised fine-tuning, TRACER improves accuracy while reducing total test-set tool calls from 4,949 to 3,486. Ablations and qualitative analyses show that these gains come from provenance-aware use of tool observations rather than from longer trajectories or more sampled tool calls.

Overall, this paper makes three contributions. (1) We formulate the *provenance gap* in multimodal tool-using agents, where final claims lack an explicit dependency structure over intermediate tool observations. (2) We introduce TRACER, a verifiable generative provenance framework that constructs sentence-level provenance during generation and converts verified provenance into local credit, traceability constraints, and efficient tool-use incentives. (3) We construct TRACE-BENCH, a benchmark for provenance-aware multimodal tool reasoning, and show that TRACER improves answer accuracy, provenance faithfulness, and tool-use efficiency.

2 Related Work

Multimodal tool-using agents. Recent multimodal agents move beyond single-turn perception toward long-horizon interaction, where models iteratively search, inspect, calculate, and integrate information across tools and modalities. Search-R1 [4] and MMSearch-R1 [5] integrate external search into model reasoning, first in text-dominant settings and then in multimodal search over images and text. WebWatcher [6] and Vision-DeepResearch [7] study deep-research-style interaction in visually grounded environments. DeepEyesV2 [8], Agent0-VL [9], and ReAgent-V [10] further improve tool-mediated reasoning, training stability, and video-oriented interaction. Other approaches address unnecessary tool use, context growth, and scalable trajectory collection [11, 12, 13, 14], while benchmarks stress multimodal search, sustained information integration, and workflow-level tool use [8, 15, 16, 17, 18]. These advances improve how agents acquire information and complete complex workflows. However, they often leave the provenance structure of the final response implicit: a trajectory may contain the right page, visual observation, or computed value, while the answer still lacks a claim-level account of which observations warrant which statements. TRACER targets this missing layer by requiring tool-grounded reasoning to produce verifiable generative provenance.

Provenance, citation, and process supervision. Citation and grounding methods aim to make model outputs more verifiable, but most of them attach references at a coarse granularity or after generation. Recent text-provenance work provides finer-grained accounts of how generated text originates from sources. TROVE [19] formulates provenance as sentence-level tracing with typed relations, including QUOTATION, COMPRESSION, and INFERENCE. GenProve [20] studies generation-time fine-grained provenance, where models generate answers together with structured provenance triples. We adopt this typed view of provenance, but study a different setting. Instead of tracing text answers to static source documents, TRACER links multimodal tool-grounded answers to tool turns, evidence units, and semantic support relations in dynamic interaction trajectories. It also converts verified provenance into local credit for tool use, which is not addressed by text-only provenance generation.

Process supervision and agent-alignment methods provide denser feedback for intermediate behavior [21, 22, 23, 24, 25]. Recent methods score progress, compare steps, propagate delayed feedback, or refine tool-integrated reinforcement learning [26, 27, 28, 29, 30, 31, 32]. These methods supervise actions, progress, or outcomes, but they do not directly model claim-level provenance over concrete multimodal tool observations. TRACER complements process supervision by making the evidential dependency structure explicit, verifiable, and usable for both evaluation and training.

3 Method

TRACER addresses the provenance gap by making the dependency between answer claims and tool evidence explicit, verifiable, and optimizable. Given a multimodal query, the model interacts

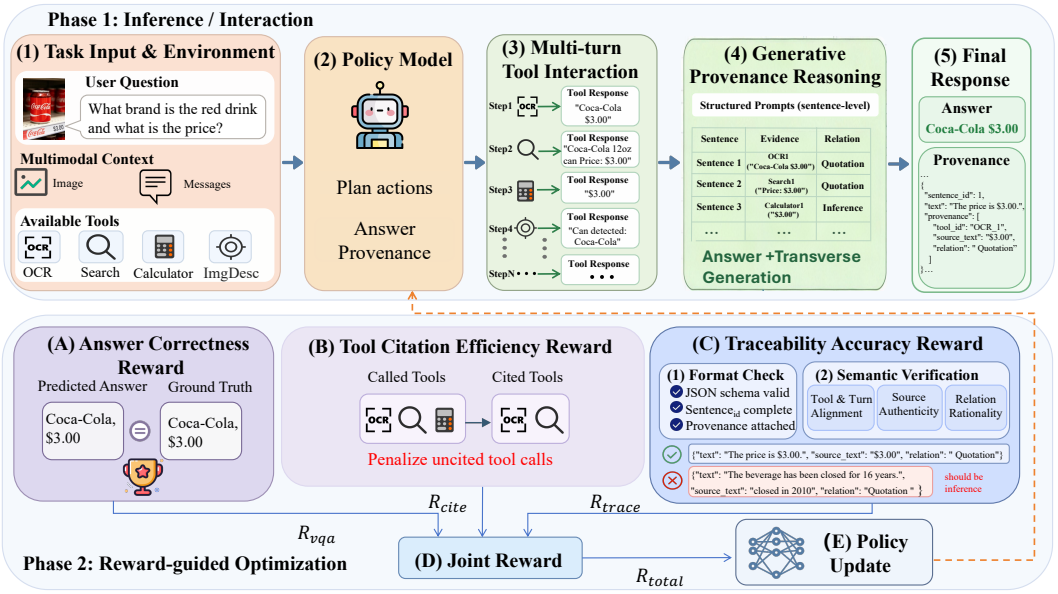


Figure 2: Illustration of TRACER. The policy model performs multi-turn tool-grounded reasoning, generates a final response with sentence-level provenance, and receives provenance-aware rewards for group-relative reinforcement learning.

with tools, generates sentence-level provenance for its final answer, and uses verified provenance to construct rewards and local credit signals for group-relative reinforcement learning.

3.1 Provenance-aware Multi-tool Reasoning

Each instance in TRACE-BENCH consists of a question q, a multimodal context $\mathcal{M}$, an optional history $\mathcal{H}_0$, a tool set $\mathcal{U}$, and a ground-truth answer $y^\star$. We denote the input as $x = (q, \mathcal{M}, \mathcal{H}_0, \mathcal{U})$. A policy model π_θ produces a tool-grounded trajectory $\tau = (a_1, e_1, \ldots, a_K, e_K)$, where $a_t = (u_t, \alpha_t)$ is the t-th tool query, $u_t \in \mathcal{U}$ is the selected tool, α_t is its argument, and e_t is the returned evidence. Each tool query has a unique turn identifier ℓ_t, such as `OCR_1` or `Calculator_2`.

A standard tool-using agent generates only a final answer y from (x, τ). TRACER instead generates both an answer and a provenance structure: $(y, \mathcal{G}) \sim \pi_\theta(\cdot \mid x, \tau)$. We decompose the answer into sentence-level claims $y = (s_1, \ldots, s_N)$. For each sentence s_i, TRACER emits a provenance set $\mathcal{P}_i = \{(\ell_{ij}, z_{ij}, r_{ij})\}_{j=1}^{m_i}$, where ℓ_{ij} refers to a tool turn, z_{ij} is the cited evidence span or localized multimodal evidence, and r_{ij} specifies the support relation: $r_{ij} \in \{\text{QUOTATION}, \text{COMPRESSION}, \text{INFERENCE}\}$. The complete provenance graph is $\mathcal{G} = \{(s_i, \mathcal{P}_i)\}_{i=1}^{N}$. This formulation makes the evidential dependency between final claims and intermediate tool evidence explicit, which a trajectory-level correctness reward cannot provide.

3.2 Generative Provenance and Provenance Rewards

TRACER generates each sentence together with its provenance:

$$p_\theta(y, \mathcal{G} \mid x, \tau) = \prod_{i=1}^{N} p_\theta(s_i, \mathcal{P}_i \mid x, \tau, s_{<i}, \mathcal{P}_{<i}). \tag{1}$$

The structured output follows a JSON schema that records the sentence text, sentence identifier, cited tool turn, source evidence, and relation type.

We verify provenance with a schema check and semantic traceability checks. The schema check $C_{\text{schema}} \in \{0, 1\}$ verifies that the output is valid, each sentence is indexed, and each sentence has at least one provenance item. For each provenance item, the traceability verifier returns v_{ij}^{turn}, v_{ij}^{src}, and $v_{ij}^{\text{rel}} \in \{0, 1\}$, which respectively check the existence of the cited tool turn, the authenticity of the

cited source, and whether the declared relation supports the sentence. The item-level validity is:

$$v_{ij} = v_{ij}^{\text{turn}} v_{ij}^{\text{src}} v_{ij}^{\text{rel}}. \tag{2}$$

TRACER uses three rewards. The answer reward is

$$R_{\text{vqa}} = \mathbb{I}\big[\text{Eval}(y) = y^\star\big]. \tag{3}$$

Let $\mathcal{T}_{\text{called}} = \{\ell_t\}_{t=1}^{K}$ be the called tool turns, and let $\mathcal{T}_{\text{cited}}$ be the subset of tool turns appearing in at least one valid provenance item. The citation-efficiency reward is

$$R_{\text{cite}} = \frac{|\mathcal{T}_{\text{cited}}|}{\max(|\mathcal{T}_{\text{called}}|, 1)}. \tag{4}$$

It penalizes tool calls that never support the final response.

The traceability reward is a hard faithfulness gate:

$$R_{\text{trace}} = C_{\text{schema}} \prod_{i=1}^{N} \prod_{j=1}^{m_i} v_{ij}. \tag{5}$$

It equals one only when every sentence-level provenance item passes all checks. The final reward is

$$R_{\text{total}} = R_{\text{vqa}} R_{\text{trace}} \left(w_0 + w_{\text{cite}} R_{\text{cite}}\right), \tag{6}$$

where w_0 preserves the reward for a correct and fully traceable answer, and w_{cite} encourages compact tool use. This design treats traceability as a strict requirement rather than a soft bonus: an answer with fabricated or unsupported provenance receives zero training reward.

3.3 Provenance-guided Group-relative Reinforcement

TRACER converts verified provenance into tool-turn credit. For the t-th tool turn, the local credit is

$$c_t = R_{\text{vqa}} R_{\text{trace}} \min\left(1, \sum_{i,j} \mathbb{I}[\ell_{ij} = \ell_t] v_{ij}\right). \tag{7}$$

Thus, a tool turn receives credit only when it contributes verified evidence to a correct and fully traceable answer. For each input x, the old policy samples a group of G rollouts:

$$\{(\tau^{(g)}, y^{(g)}, \mathcal{G}^{(g)})\}_{g=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x). \tag{8}$$

For token n in rollout g, we define a provenance-aware group-normalized advantage:

$$\widehat{A}_n^{(g)} = \text{Norm}_{g \in [G]}\left(R_{\text{total}}^{(g)} + \lambda \sum_t \mathbb{I}[n \in \mathcal{S}_t^{(g)}] c_t^{(g)}\right), \tag{9}$$

where $\mathcal{S}_t^{(g)}$ denotes the token span that generates the t-th tool query, and λ controls the strength of local provenance credit. Non-tool tokens receive only the trajectory-level reward term. This group-relative advantage does not require a learned value model. Let $\omega^{(g)}$ be the generated token sequence, $h_n^{(g)}$ be the prefix before token n, and

$$\rho_n^{(g)}(\theta) = \frac{\pi_\theta(\omega_n^{(g)} \mid h_n^{(g)})}{\pi_{\theta_{\text{old}}}(\omega_n^{(g)} \mid h_n^{(g)})}. \tag{10}$$

We optimize a clipped per-token objective:

$$\begin{aligned} \ell_n^{(g)}(\theta) = &\min\left(\rho_n^{(g)} \widehat{A}_n^{(g)},\ \text{clip}(\rho_n^{(g)}, 1-\epsilon, 1+\epsilon) \widehat{A}_n^{(g)}\right) \\ &- \beta D_{\text{KL}}\big(\pi_\theta(\cdot \mid h_n^{(g)}) \,\|\, \pi_{\text{ref}}(\cdot \mid h_n^{(g)})\big). \end{aligned} \tag{11}$$

The final optimization objective is

$$\mathcal{J}_{\text{TRACER}}(\theta) = \mathbb{E}_x \mathbb{E}_{\{\omega^{(g)}\}}\left[\frac{1}{G} \sum_{g=1}^{G} \frac{1}{|\omega^{(g)}|} \sum_{n=1}^{|\omega^{(g)}|} \ell_n^{(g)}(\theta)\right]. \tag{12}$$

The policy maximizes $\mathcal{J}_{\text{TRACER}}$. The group-normalized reward compares multiple reasoning trajectories for the same input, while provenance credit identifies which tool queries actually support the final answer. As a result, TRACER aligns optimization with the evidential structure of multi-tool reasoning rather than assigning the same trajectory-level reward to all intermediate decisions.

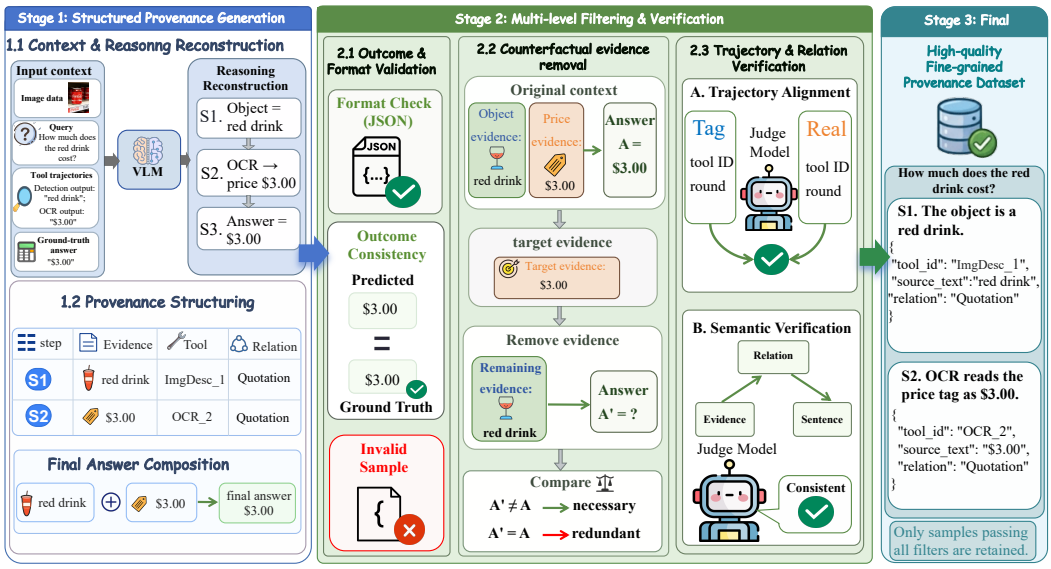


Figure 3: Overview of the TRACE-BENCH construction pipeline. Coarse tool-interaction trajectories are transformed into sentence-level provenance records and retained only after outcome, format, counterfactual, and provenance verification.

4 Dataset

TRACE-BENCH evaluates whether tool-augmented multimodal agents can produce correct answers together with verifiable claim-level provenance. We build it on ToolVQA [33], whose trajectories contain multimodal inputs, questions, tool calls, tool observations, and reference answers. The original trajectories do not specify which observation supports each final claim. We therefore transform coarse multi-turn trajectories into concise reasoning responses with structured provenance records that identify the exact tool observation supporting each answer sentence.

4.1 Provenance Construction

Each source instance consists of a question q, a multimodal context $\mathcal{M}$, a tool set $\mathcal{U}$, a ground-truth answer $y^\star$, and an ordered tool trajectory τ. We construct a provenance-aware output $(y, \mathcal{G})$, where y is decomposed into sentence-level claims and $\mathcal{G}$ records the supporting evidence for each claim. For each sentence, the provenance specifies the cited tool turn, the localized evidence span, and the semantic relation, chosen from QUOTATION, COMPRESSION, and INFERENCE. We instantiate this process with a VLM-based generator prompted with the full task context and the reference answer. The generator produces a coherent response and a JSON provenance record for every sentence, converting each original trajectory into an explicit claim, i.e., evidence graph. Details on the tool inventory and relation taxonomy are provided in Appendix C, and the generation and verification prompts are shown in Figures 6 and 7.

4.2 Multi-stage Filtering

Because generated provenance can contain answer inconsistencies, malformed JSON, unsupported citations, or incorrect relation labels, we apply a multi-stage filtering pipeline. First, we validate answer consistency against the reference answer and check the JSON schema with a rule-based parser. Second, we perform counterfactual evidence removal: for each cited evidence unit, we remove or mask that unit while preserving the rest of the trajectory and then re-evaluate the answer and provenance. If the answer and provenance remain unchanged, the citation is likely redundant, recoverable from other observations, or unsupported by the claimed source, so the record is filtered. If removing the evidence changes the answer or provenance, we treat the record as evidence-sensitive under the verifier and retain it for the next stage. Finally, an LLM-as-a-judge verifier checks trajectory alignment, source authenticity, and relation rationality, discarding records with nonexistent tool turns,

Table 1: Main results on TRACE-BENCH. Acc. denotes final task accuracy. Tool Call and Tool Error are total counts on the held-out test set. SummAcc. measures the quality of tool-trajectory summarization for the target question. Traceability Acc. and Prov. F1 evaluate generated claim-level provenance and are applicable only to models that output structured provenance records.

Model Setting	Acc. (%)	Tool Call ↓	Tool Error ↓	SummAcc. (%)	Traceability Acc. (%)	Prov. F1 (%)
Closed-source LFMs						
ChatGPT-4o-latest [34]	38.29	–	–	–	–	–
ChatGPT-4o-latest + tool	34.96	2003	1804	56.10	–	–
Claude-3-5-sonnet [35]	30.33	–	–	–	–	–
Claude-3-5-sonnet + tool	30.52	585	524	75.52	–	–
Gemini-2.5-Pro [36]	27.35	–	–	–	–	–
Gemini-2.5-Pro + tool	54.43	1569	438	70.27	–	–
Open-source LFMs						
Qwen2-VL-7B-Instruct [37]	3.64	–	–	–	–	–
Qwen2-VL-7B-Instruct + tool	5.86	1373	1051	38.18	–	–
LLaVA-v1.5-7B VLM [38]	8.57	–	–	–	–	–
LLaVA-v1.5-7B VLM + tool	1.17	9684	9684	0.01	–	–
Qwen2-VL-2B-Instruct [37]	7.71	–	–	–	–	–
Qwen2-VL-2B-Instruct + tool	2.10	4067	3915	21.82	–	–
Tuned LLaVA-7B [33]	7.21	–	–	–	–	–
Tuned LLaVA-7B + tool	18.80	4311	617	30.91	–	–
InternVL3.5-7B [39]	12.27	–	–	–	–	–
InternVL3.5-7B + tool	13.68	249	10	48.39	–	–
MiMo-VL-7B-RL [40]	22.66	–	–	–	–	–
MiMo-VL-7B-RL + tool	37.09	8491	1513	57.31	–	–
Qwen3-VL-8B-Instruct [41]	23.52	–	–	–	–	–
Qwen3-VL-8B-Instruct + tool	26.62	5580	3730	72.07	–	–
Qwen3-VL-8B-Instruct + tool-SFT	70.94	4949	590	84.39	–	–
TRACER-SFT	73.56	4034	654	93.29	77.53	70.70
TRACER-RL	**78.23**	**3486**	**573**	**95.72**	**93.61**	**90.52**

untraceable evidence, or implausible evidence–relation–claim triples. Additional details and common failure modes are discussed in Appendix C.

4.3 Split Construction and Statistics

After filtering, we retain 14,183 high-quality provenance-aware trajectories from 23,655 candidates, corresponding to a 59.96% retention rate. We preserve the official ToolVQA held-out test split of 2,550 examples and partition the remaining examples into 10,470 training and 1,163 validation instances. The final responses are compact, with an average of 2.4 reasoning sentences and 2.56 anchored provenance records per response. Comprehensive statistics on dataset scale, tool-call composition, text length, and relation distributions are presented in Appendix C.1.

5 Experimental

5.1 Main Results

Table 1 reports the main experimental results, and the detailed experimental configurations are provided in Appendix B. In addition to final task accuracy and tool-use statistics, we include provenance quality metrics in the main table, since the central goal of TRACER is not only to improve answer accuracy, but also to make generated answers verifiably grounded in external tool observations. Traceability Acc. and provenance F1 are reported for models that generate structured provenance records. Overall, the results show that simply equipping models with external tools does not consistently improve task performance. For example, ChatGPT-4o-latest reaches 38.29% accuracy without tools, but its tool-augmented variant drops to 34.96% while producing 2,003 tool calls and 1,804 tool errors. Several open-source models show the same pattern: they issue many tool calls but obtain limited or negative accuracy gains. LLaVA-v1.5-7B VLM + tool, for instance, produces 9,684 calls, all counted as tool errors under our evaluation protocol, and reaches only 0.01% SummAcc. These results indicate that tool use is not a simple interface extension; it requires planning, evidence selection, observation integration, and claim-level grounding.

Supervised training on tool trajectories substantially improves tool interaction. Qwen3-VL-8B-Instruct + tool-SFT reaches 70.94% accuracy, outperforming the untuned tool-augmented counterpart

by 44.32 percentage points. However, tool-only SFT still lacks explicit supervision over which observations support which final claims, and it continues to use a relatively large number of tools. Adding generative provenance supervision improves both task performance and grounding: TRACER-SFT reaches 73.56% accuracy, 93.29% SummAcc., 77.53% Traceability Acc., and 70.70% provenance F1. After reinforcement learning, TRACER-RL achieves the best results, with 78.23% accuracy, 95.72% SummAcc., 93.61% Traceability Acc., and 90.52% provenance F1. It outperforms the strongest closed-source tool-augmented baseline, Gemini-2.5-Pro + tool, by 23.80 percentage points. Compared with Qwen3-VL-8B-Instruct + tool-SFT, it also reduces total test-set tool calls from 4,949 to 3,486, a 29.56% reduction. Thus, the improvement comes from using fewer and more relevant observations, not from calling more tools.

5.2 Ablation Study

We ablate the two provenance-aware reward components to test whether the gains of TRACER-RL come from the proposed provenance mechanism. The citation-efficiency reward R_{cite} is designed to reduce tool calls that do not support the final response. The traceability reward R_{trace} enforces faithful alignment between generated provenance records and actual tool observations. We evaluate the effect of removing each reward on answer accuracy, traceability, provenance precision and recall, and tool-use efficiency.

Table 2: Ablation results of TRACER-RL. Provenance-aware rewards improve answer accuracy, traceability, provenance quality, and tool-use efficiency.

Model Setting	Acc. (%)	Tool Call	Tool Error	SummAcc. (%)	Traceability Acc. (%)	Prov. Precision (%)	Prov. Recall (%)	Prov. F1 (%)
TRACER-SFT	73.56	4034	654	93.29	77.53	68.42	73.15	70.70
TRACER-RL	**78.23**	**3486**	**573**	**95.72**	**93.61**	**89.24**	**91.85**	**90.52**
TRACER-RL w/o R_{cite}	75.13	4251	615	93.86	87.28	74.20	85.12	79.28
TRACER-RL w/o R_{trace}	74.58	3839	593	92.14	91.67	82.15	83.40	82.77

As shown in Table 2, the full TRACER-RL model consistently outperforms TRACER-SFT across both task-level and provenance-level metrics. Accuracy increases from 73.56% to 78.23%, Traceability Acc. improves from 77.53% to 93.61%, and provenance F1 increases substantially from 70.70% to 90.52%. This confirms that the RL stage does not simply improve answer selection, but teaches the model to generate answers whose claims are more faithfully grounded in the original tool trajectory. The two reward ablations reveal complementary failure modes. When R_{cite} is removed, the number of tool calls rises from 3486 to 4251, while provenance precision drops sharply from 89.24% to 74.20%. Although provenance recall remains relatively high at 85.12%, the precision degradation indicates that many additional tool observations do not provide reliable support for the final claims. This suggests that, without citation-efficiency constraints, the model tends to over-explore and becomes less selective about which observations should be treated as provenance-bearing evidence. When R_{trace} is removed, the model produces fewer tool calls than the variant without R_{cite}, but its accuracy drops to 74.58%, SummAcc. drops to 92.14%, and provenance F1 decreases to 82.77%. This shows that reducing redundant tool calls alone is insufficient for faithful reasoning, since the model may still generate plausible-looking provenance records that do not correctly match the gold evidence units or support relations.

Therefore, R_{cite} mainly improves tool-use efficiency by suppressing redundant exploration and increasing the density of useful evidence, while R_{trace} improves provenance faithfulness by enforcing valid alignment among claims, evidence units, tool turns, and relation types. Only when both rewards are used together does the model achieve high accuracy, low tool-call count, high traceability, and strong provenance F1. These results provide direct evidence that TRACER improves multimodal tool reasoning by learning a verifiable provenance structure, rather than merely benefiting from additional RL training or more sampled trajectories.

5.3 Case Study

We present a representative case to qualitatively examine how generative provenance affects multi-tool visual reasoning. The question asks when one of the manufacturers shown in the image first entered motorcycle racing. Solving it requires identifying the visual manufacturer and retrieving a fine-grained historical fact, making it difficult to answer reliably from parametric knowledge alone.

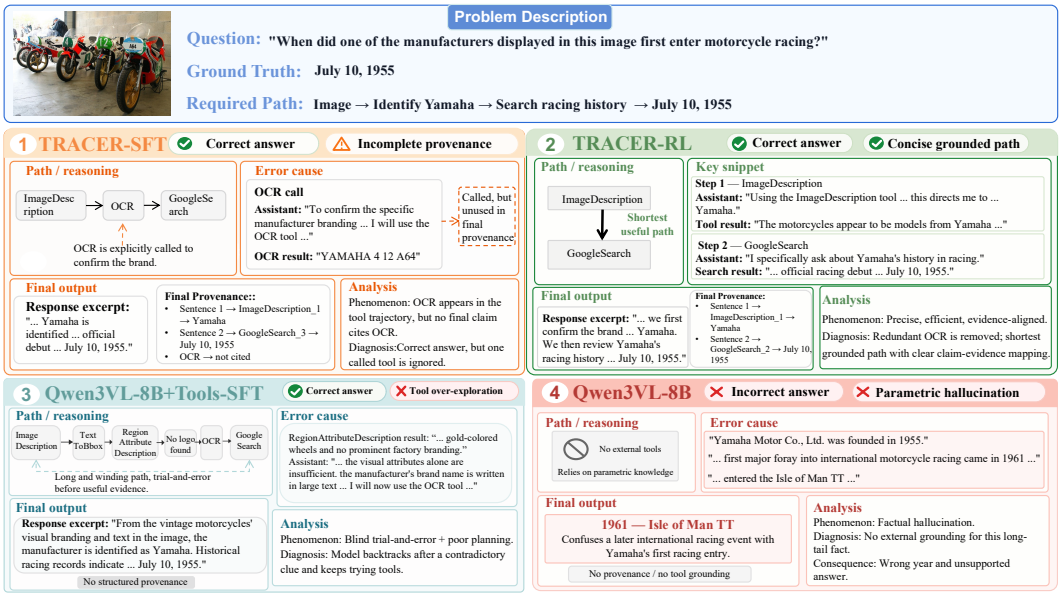


Figure 4: Comparison of reasoning trajectories. Unlike traditional methods that treat all tool calls as necessary evidence, **TRACER** exposes the actual dependency between observations and claims. Generative provenance reveals redundant steps (e.g., OCR in SFT) and enables **TRACER-RL** to learn more efficient, evidence-directed tool-use trajectories.

The tool-free model predicts an incorrect year, suggesting that memorized knowledge is insufficient for this long-tail multimodal factual query. The tool-augmented model without provenance training can access external tools, but follows an inefficient trajectory by invoking `TextToBbox`, `RegionAttributeDescription`, OCR, and `GoogleSearch`. Several early localization and regional-description calls fail to provide useful evidence, indicating that tool access alone does not ensure effective evidence acquisition. TRACER-SFT produces the correct answer with `ImageDescription`, OCR, and `GoogleSearch`. More importantly, its generated provenance exposes the actual dependency structure behind the answer: the manufacturer identification is supported by `ImageDescription_1`, while the racing-entry date is supported by `GoogleSearch_3`. Although OCR is called, none of the final claims cite its output. Thus, a conventional tool trajectory would make OCR appear to be part of the reasoning chain, whereas generative provenance reveals it as a redundant exploration step. This illustrates the provenance gap: standard trajectories record which tools are called, but do not reveal which observations actually support each generated claim. TRACER-RL further removes the redundant OCR step and directly uses `ImageDescription` for visual identification and `GoogleSearch` for factual retrieval. This shorter trajectory still yields the correct answer, showing that reinforcement learning based on provenance signals does not merely encourage more tool use, but instead promotes compact and evidence-directed tool use. By penalizing unused citations and assigning credit to useful tool observations, TRACER-RL learns to suppress low-value exploration while preserving the evidence needed for answer generation.

Overall, this case supports our main claim: reliable multi-tool visual reasoning requires not only access to external tools, but also explicit modeling of the dependency between tool observations and final claims. Generative provenance makes this dependency observable, enabling both better interpretability and more effective tool-use optimization.

6 Acknowledgments

The authors thank anonymous reviewers for their valuable comments and insightful feedback. The authors also thank Shenyang Institute of Computing Technology, Chinese Academy of Sciences, University of Chinese Academy of Sciences, and MAIS, Institute of Automation, Chinese Academy of Sciences for their support and resources. This research work has been supported by the National Natural Science Foundation of China under Grant No. 62676401.

7 Conclusion

We introduce TRACER, a framework for verifiable generative provenance in multimodal tool-using agents. TRACER addresses the provenance gap by pairing each generated sentence with structured provenance that specifies the supporting tool turn, evidence unit, and semantic support relation. This design turns provenance from a post-hoc explanation into a generation-time interface that can be verified and optimized. By using verified provenance to distinguish provenance-bearing tool calls from redundant exploration, TRACER enables finer-grained credit assignment and more efficient tool use. We also construct TRACE-BENCH, a benchmark for evaluating provenance-aware multimodal tool reasoning from coarse interaction trajectories. Experiments show that tool access alone can introduce noise, while provenance-aware training improves answer accuracy, summary accuracy, traceability, provenance F1, and tool-call efficiency. These results suggest that reliable multimodal tool reasoning should be driven by faithful and verifiable use of supporting observations, rather than by more tool calls alone.

References

[1] 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 *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13590–13618, 2024.

[2] Zijing Liang, Yanjie Xu, Yifan Hong, Penghui Shang, Qi Wang, Qiang Fu, and Ke Liu. A survey of multimodel large language models. In *Proceedings of the 3rd international conference on computer, artificial intelligence and control engineering*, pages 405–409, 2024.

[3] Shezheng Song, Xiaopeng Li, Shasha Li, Shan Zhao, Jie Yu, Jun Ma, Xiaoguang Mao, Weimin Zhang, and Meng Wang. How to bridge the gap between modalities: Survey on multimodal large language model. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 37(9):5311–5329, 2025.

[4] Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling (COLM)*, 2025.

[5] Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. Mmsearch-r1: Incentivizing lmms to search. *arXiv preprint arXiv:2506.20670*, 2025.

[6] Xinyu Geng, Peng Xia, Zhen Zhang, Xinyu Wang, Qiuchen Wang, Ruixue Ding, Chenxi Wang, Jialong Wu, Kuan Li, Yida Zhao, Huifeng Yin, Yong Jiang, Pengjun Xie, Fei Huang, Huaxiu Yao, Yi R. Fung, and Jingren Zhou. Webwatcher: Breaking new frontiers of vision-language deep research agent. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

[7] Wenxuan Huang, Yu Zeng, Qiuchen Wang, Zhen Fang, Shaosheng Cao, Zheng Chu, Qingyu Yin, Shuang Chen, Zhenfei Yin, Lin Chen, et al. Vision-deepresearch: Incentivizing deepresearch capability in multimodal large language models. *arXiv preprint arXiv:2601.22060*, 2026.

[8] Jack Hong, Chenxiao Zhao, ChengLIn Zhu, Weiheng Lu, Guohai Xu, and XingYu. Deepeyesv2: Toward agentic multimodal model. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

[9] Jiaqi Liu, Kaiwen Xiong, Peng Xia, Yiyang Zhou, Haonian Ji, Lu Feng, Siwei Han, Mingyu Ding, and Huaxiu Yao. Agent0-vl: Exploring self-evolving agent for tool-integrated vision-language reasoning. *arXiv preprint arXiv:2511.19900*, 2025.

[10] Yiyang Zhou, Yangfan He, Yaofeng Su, Siwei Han, Joel Jang, Gedas Bertasius, Mohit Bansal, and Huaxiu Yao. Reagent-v: A reward-driven multi-agent framework for video understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2026.

[11] Shilin Yan, Jintao Tong, Hongwei Xue, Xiaojun Tang, Yangyang Wang, Kunyu Shi, Guannan Zhang, Ruixuan Li, and Yixiong Zou. Act wisely: Cultivating meta-cognitive tool use in agentic multimodal models. *arXiv preprint arXiv:2604.08545*, 2026.

[12] Yifan Du, Zikang Liu, Jinbiao Peng, Jie Wu, Junyi Li, Jinyang Li, Wayne Xin Zhao, and Ji-Rong Wen. Towards long-horizon agentic multimodal search. *arXiv preprint arXiv:2604.12890*, 2026.

[13] Xiangyu Peng, Can Qin, An Yan, Xinyi Yang, Zeyuan Chen, Ran Xu, and Chien-Sheng Wu. Mta-agent: An open recipe for multimodal deep search agents. *arXiv preprint arXiv:2604.06376*, 2026.

[14] Yikun Liu, Yuan Liu, Le Tian, Xiao Zhou, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. Points-seeker: Towards training a multimodal agentic search model from scratch. *arXiv preprint arXiv:2604.14029*, 2026.

[15] Xuying Ning, Dongqi Fu, Tianxin Wei, Mengting Ai, Jiaru Zou, Ting-Wei Li, and Jingrui He. Search-MM: Benchmarking multimodal agentic RAG with structured reasoning chains. In *NeurIPS 2025 Workshop on Evaluating the Evolving LLM Lifecycle: Benchmarks, Emergent Abilities, and Scaling*, 2025.

[16] Yu Zeng, Wenxuan Huang, Zhen Fang, Shuang Chen, Yufan Shen, Yishuo Cai, Xiaoman Wang, Zhenfei Yin, Lin Chen, Zehui Chen, et al. Vision-deepresearch benchmark: Rethinking visual and textual search for multimodal large language models. *arXiv preprint arXiv:2602.02185*, 2026.

[17] Zhaochen Su, Jincheng Gao, Hangyu Guo, Zhenhua Liu, Lueyang Zhang, Xinyu Geng, Shijue Huang, Peng Xia, Guanyu Jiang, Cheng Wang, et al. Agentvista: Evaluating multimodal agents in ultra-challenging realistic visual scenarios. *arXiv preprint arXiv:2602.23166*, 2026.

[18] Xuan Dong, Huanyang Zheng, Tianhao Niu, Zhe Han, Pengzhan Li, Bofei Liu, Zhengyang Liu, Guancheng Li, Qingfu Zhu, and Wanxiang Che. Epibench: Benchmarking multi-turn research workflows for multimodal agents. *arXiv preprint arXiv:2604.05557*, 2026.

[19] Junnan Zhu, Min Xiao, Yining Wang, Feifei Zhai, Yu Zhou, and Chengqing Zong. TROVE: A challenge for fine-grained text provenance via source sentence tracing and relationship classification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 11755–11771, 2025.

[20] Jingxuan Wei, Xingyue Wang, Yanghaoyu Liao, Jie Dong, Yuchen Liu, Caijun Jia, Bihui Yu, and Junnan Zhu. Genprove: Learning to generate text with fine-grained provenance. *arXiv preprint arXiv:2601.04932*, 2026.

[21] Sanjiban Choudhury. Process reward models for llm agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*, 2025.

[22] Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. Agentprm: Process reward models for llm agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026 (WWW)*, page 4184–4195, 2026.

[23] Hanlin Wang, Jian Wang, Chak Tou Leong, and Wenjie Li. STeCa: Step-level trajectory calibration for LLM agent learning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11597–11614, 2025.

[24] Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. Spa-rl: Reinforcing llm agents via stepwise progress attribution. *arXiv preprint arXiv:2505.20732*, 2025.

[25] Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Forty-second International Conference on Machine Learning (ICML)*, 2025.

[26] Zhirui Deng, Zhicheng Dou, Yutao Zhu, Ji-Rong Wen, Ruibin Xiong, Mang Wang, and Weipeng Chen. From novice to expert: Llm agent policy optimization via step-wise reinforcement learning. *arXiv preprint arXiv:2411.03817*, 2024.

[27] Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon llm agents. *arXiv preprint arXiv:2603.08754*, 2026.

[28] Qiao Liang, Yuke Zhu, Chao Ge, Lei Yang, Ying Shen, Bo Zheng, and Sheng Guo. Learning from the irrecoverable: Error-localized policy optimization for tool-integrated llm reasoning. *arXiv preprint arXiv:2602.09598*, 2026.

[29] Yifeng Ding, Hung Le, Songyang Han, Kangrui Ruan, Zhenghui Jin, Varun Kumar, Zijian Wang, and Anoop Deoras. Empowering multi-turn tool-integrated reasoning with group turn policy optimization. *arXiv preprint arXiv:2511.14846*, 2025.

[30] Junzhe Wang, Zhiheng Xi, Hao Luo, Shihan Dou, Tao Gui, Qi Zhang, et al. Enhancing llm-based search agents via contribution weighted group relative policy optimization. *arXiv preprint arXiv:2604.14267*, 2026.

[31] Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun MA, and Bo An. SimpleTIR: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026.

[32] Weiyang Guo, Zesheng Shi, Liye Zhao, Jiayuan Ma, Zeen Zhu, Junxian He, Min Zhang, and Jing Li. E3-tir: Enhanced experience exploitation for tool-integrated reasoning. *arXiv preprint arXiv:2604.09455*, 2026.

[33] Shaofeng Yin, Ting Lei, and Yang Liu. Toolvqa: A dataset for multi-step reasoning vqa with external tools. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4424–4433, 2025.

[34] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[35] Anthropic. Claude 3.5 sonnet system card. Technical report, Anthropic PBC, 2024. Official technical report describing Claude 3.5 Sonnet capabilities and safety benchmarks. Available at: `https://www.anthropic.com/news/claude-3-5-sonnet`.

[36] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[37] 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.

[38] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024.

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

[40] LLM Xiaomi, Bingquan Xia, Bowen Shen, Dawei Zhu, Di Zhang, Gang Wang, Hailin Zhang, Huaqiu Liu, Jiebao Xiao, Jinhao Dong, et al. Mimo: Unlocking the reasoning potential of language model–from pretraining to posttraining. *arXiv preprint arXiv:2505.07608*, 2025.

[41] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

A Limitations and Broader Impacts

Limitations. TRACER improves verifiable generative provenance for multimodal tool-using agents, but several limitations remain. First, TRACE-BENCH relies on LLM-as-a-judge verification during data filtering and traceability evaluation. Although we combine schema checks, counterfactual evidence removal, trajectory alignment, and relation-rationality checks, evaluator bias and prompt sensitivity may still introduce noise. Second, the benchmark is built on ToolVQA and currently covers seven tools for global visual description, OCR, localization, region description, counting, retrieval, and calculation. Additional tools, modalities, and task domains are needed to test whether the same provenance interface generalizes to broader agentic workflows. Third, our trainable models are based on an 8B-scale MLLM. The scaling behavior of generative provenance, local tool credit, and citation-efficiency rewards for substantially larger models remains an important direction for future work. Finally, provenance verification checks whether cited evidence supports generated claims, but it does not prove that the model internally used the evidence causally. We therefore interpret provenance as a verifiable external dependency structure, not as a complete mechanistic explanation of the model's internal computation.

Broader impacts. The main positive impact of TRACER is improved accountability for tool-using multimodal agents. By requiring each claim to cite authentic tool observations and a rational support relation, the framework can help users audit generated answers, identify unsupported claims, and reduce redundant tool usage. Potential risks include over-trusting provenance records when verifiers are imperfect, using generated provenance to create misleading but plausible explanations, or applying tool-using agents in sensitive domains without domain-specific validation. Mitigations include reporting verifier limitations, retaining the original tool trajectories for audit, releasing data documentation with source and license information, and avoiding deployment in high-stakes settings without independent verification.

B Experimental Details

B.1 Experimental Setup

We evaluate on TRACE-BENCH using 10,470 training, 1,163 validation, and 2,550 held-out test examples in accordance with the ToolVQA protocol [33]. Operating under a tool-interactive environment equipped with visual perception, OCR, grounding, retrieval, and computation capabilities, models iteratively gather observations to produce final answers alongside structured claim-level provenance. All trainable models are initialized from Qwen3-VL-8B-Instruct [41]. We compare our provenance-aware TRACER-SFT and reward-optimized TRACER-RL against proprietary multimodal models, representative open-source baselines, and a standard tool-SFT baseline designed to isolate the impact of explicit provenance optimization. Model performance is comprehensively assessed across task-solving ability via answer and summary accuracy, provenance quality via traceability accuracy and provenance F1, and efficiency via the average number of tool calls per example.

B.2 Training Details

All trainable models are initialized from Qwen3-VL-8B-Instruct [41]. During validation and test inference, we use deterministic decoding with temperature set to 0.

Supervised fine-tuning. In the supervised fine-tuning stage, we perform full-parameter updates across the vision encoder, multi-modal projector, and language model backbone. The maximum input length is set to 8,192 tokens. We use a per-device batch size of 2 with gradient accumulation over 4 steps, a learning rate of 5×10^{-6}, and a 5% warmup ratio. Training is conducted for 2 epochs using bfloat16 precision and DeepSpeed ZeRO-2 optimization across 8 NVIDIA A100 GPUs. The SFT objective trains the model to follow the tool-interaction format and generate sentence-level provenance records for final answers.

Reinforcement learning. Following SFT, we apply reinforcement learning to further improve task performance and provenance faithfulness. The RL model is initialized from the SFT checkpoint

and optimized for 200 steps. For each prompt, we generate a rollout group of 8 candidate outputs and adopt rejection sampling to construct optimization signals. We further curate high-entropy training examples whose sampled candidates show substantial variance between correct and incorrect responses, focusing optimization on uncertain cases.

Reinforcement learning. The RL reward follows the provenance-aware reward defined in Section 3.2. It consists of three components: answer correctness, tool citation efficiency, and traceability accuracy. Answer correctness and traceability accuracy are used as hard gating signals. If the predicted answer is incorrect or the generated provenance fails the traceability verification, the overall reward is set to zero. Otherwise, the reward is computed as

$$R_{\text{total}} = R_{\text{vqa}} R_{\text{trace}} \left(w_0 + w_{\text{cite}} R_{\text{cite}} \right), \tag{13}$$

where we set $w_0 = 1.0$ and $w_{\text{cite}} = 0.5$ in all experiments. Here, R_{vqa} indicates whether the final answer is correct, R_{trace} indicates whether all sentence-level provenance items pass schema, source-alignment, and relation-rationality checks, and R_{cite} measures the fraction of called tool turns that are actually cited by valid provenance. This design rewards correct and fully traceable answers while encouraging compact tool use and penalizing unnecessary or uncited tool calls.

We further use provenance-derived local credit to guide optimization over tool-query tokens. Following Section 3.3, the local credit coefficient is set to $\lambda = 0.3$. The clipped group-relative objective uses a clipping threshold of $\epsilon = 0.2$. A KL penalty against the SFT reference model is applied during RL training, with coefficient $\beta = 0.02$. The reward weights, local credit coefficient, clipping threshold, and KL coefficient are selected on the validation set and kept fixed for all RL experiments. All training stages are implemented with `torchrun` and distributed across 8 NVIDIA A100 GPUs.

B.3 Baseline Details

We compare TRACER with a broad set of closed-source and open-source multimodal large foundation models.

Closed-source models. We include ChatGPT-4o-latest [34], Claude-3-5-Sonnet [35], and Gemini-2.5-Pro [36]. For each closed-source model, we evaluate both the direct-answer setting and the tool-augmented setting. In the direct-answer setting, the model directly predicts the answer from the input image and question. In the tool-augmented setting, the model is allowed to interact with the same tool environment before producing the final answer.

Open-source models. We evaluate representative open-source vision-language models, including Qwen2-VL-7B-Instruct [37], LLaVA-v1.5-7B [38], Qwen2-VL-2B-Instruct [37], Tuned LLaVA-7B [33], InternVL3.5-7B [39], MiMo-VL-7B-RL [40], and Qwen3-VL-8B-Instruct [41]. Whenever applicable, each model is evaluated in both direct-answer and tool-augmented modes. These baselines test whether existing open-source multimodal models can effectively plan tool calls, integrate multi-turn observations, and summarize intermediate evidence into the final answer.

Tool-use and provenance-aware baselines. We include Qwen3-VL-8B-Instruct+tool-SFT as a strong supervised tool-use baseline. This model is trained with the same base checkpoint and tool-interaction format as our method, but does not explicitly optimize claim-level generative provenance. We further report TRACER-SFT, which uses provenance-aware supervised fine-tuning, and TRACER-RL, which additionally applies reward-guided optimization with answer correctness, tool citation efficiency, and traceability accuracy rewards. For models that do not produce structured provenance records, Traceability Acc. and provenance F1 are not applicable.

B.4 Metric Details

We evaluate models using both task-solving and provenance-oriented metrics.

Answer accuracy. Answer accuracy (Acc.) measures whether the model produces the correct final answer for the original VQA task. It reflects end-to-end performance in the tool-interactive setting.

Summary accuracy. Summary accuracy (SummAcc.) measures whether the model can infer the correct final answer from a given tool trajectory without making additional tool calls. This metric isolates the model's ability to integrate and summarize already observed tool evidence.

Traceability accuracy. Traceability Acc. is computed using an LLM-as-a-judge verifier, where we use Gemini-2.5-Pro as the judge model. Given the generated provenance and the original tool trajectory, the verifier checks whether each cited tool identifier corresponds to a real tool call, whether the cited evidence can be traced back to the selected tool observation, and whether the predicted provenance relation is semantically reasonable. An example is counted as traceable only when its generated claim-level provenance passes all these checks.

Provenance precision, recall, and F1. Provenance precision, recall, and F1 are computed against the reference provenance annotations in the test set. We treat each provenance annotation as an evidence–claim link with a relation type. Precision measures the proportion of predicted provenance links that are correct, recall measures the proportion of reference provenance links recovered by the model, and F1 is their harmonic mean. These metrics evaluate whether the model can generate verifiable and complete claim-level provenance rather than merely producing a correct final answer.

Tool-call statistics. We additionally report the average number of tool calls per example to characterize tool-use behavior. This metric helps distinguish models that solve tasks through efficient evidence acquisition from those that rely on redundant or poorly grounded tool exploration.

C Additional Dataset Details and Statistics

C.1 Comprehensive Dataset Statistics

Table 3 summarizes the overall dataset scale, tool-call composition, text length statistics, and provenance annotation distribution. Furthermore, Figure 5 provides a compact statistical overview of the final benchmark, illustrating the data split, tool-call distribution, and provenance-relation distribution.

Statistic	Value
Dataset Overview	
Total examples	14,183
Total tool calls	40,491
Tool-call Distribution	
ImageDescription	12,594
GoogleSearch	9,666
OCR	5,801
Calculator	3,281
TextToBbox	3,094
RegionAttributeDescription	3,094
CountGivenObject	2,961
Length Statistics	
Average query length	15.46
Average response length	78.43
Provenance Statistics	
Total records	36,431
Avg records per example	2.57
Avg records per sentence	1.07
Relation Distribution	
QUOTATION	24,239
INFERENCE	8,364
COMPRESSION	3,828

Table 3: Summary of the TRACE-BENCH dataset.

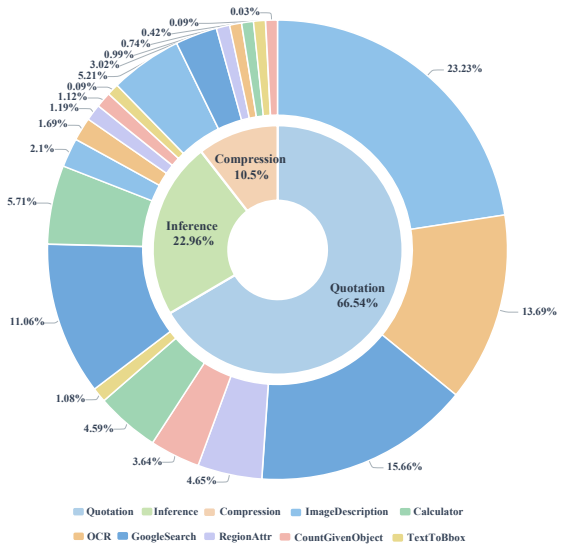


Figure 5: Statistical overview of TRACE-BENCH. The figure summarizes the data split, tool-call distribution, and provenance-relation distribution.

C.2 Detailed Cross-domain Affinity

As summarized in Table 3, the dataset categorizes evidence into quotation at 66.5%, inference at 22.9%, and compression relations at 10.5%, revealing a distinct cognitive division of labor. Further breakdown reveals distinct tool affinities. Among the 24,239 quotation relations, ImageDescription with 8,463 instances and OCR with 4,987 instances maintain absolute dominance. This aligns with the intuition of multimodal interaction: when processing complex images, visual models tend to prioritize quoting the extracted geometric and textual attributes of the scene as foundational factual anchors for reasoning.

Within the 8,364 inference relations, GoogleSearch with 4,029 instances and Calculator with 2,080 instances exhibit exceptionally strong task affinity. Singular visual quotation is often insufficient for resolving complex visual question answering problems. The model must leverage external retrieval to introduce background information of searched entities or input quoted numerical values into a calculator for symbolic deduction, thereby completing an evidence-based deep logical leap. Lastly, the compression relation is primarily concentrated in ImageDescription with 1,899 instances and GoogleSearch with 1,101 instances. Because the observation states returned by these tools often contain substantial redundant text, the model spontaneously exhibits the ability to distill crucial state information through summarization and noise reduction mechanisms.

C.3 Empirical Characteristics and Provenance Failures

TRACE-BENCH emphasizes verifiable generative provenance rather than merely adding structured metadata to existing trajectories. The benchmark requires a model to identify provenance-bearing observations, distinguish them from redundant tool calls, and assign rational support relations to generated claims. This design is especially important for multimodal tool-use agents, where a correct final answer may still be produced from unsupported reasoning, irrelevant observations, or redundant exploration.

The filtered examples reveal several common provenance failures. These include malformed provenance schemas, missing sentence-level evidence, invalid tool-turn references, evidence units that cannot be aligned to the cited observation, relation labels inconsistent with the evidence–claim transformation, and evidence-insensitive citations under counterfactual removal. Such failures correspond to realistic errors in tool-using agents. For instance, a model may call OCR but never use the OCR result in any final claim, or cite a retrieved snippet while actually relying on a previous visual description. Overall, TRACE-BENCH provides an empirical testbed for studying provenance-aware optimization, local credit assignment, and efficient tool use in multimodal agents.

C.4 Detailed Multi-stage Filtering Pipeline

As briefly introduced in Section 4.2, because provenance annotations are generated from complex, long-context multimodal tool trajectories, they may inevitably contain answer-level inconsistencies, malformed JSON structures, unsupported evidence citations, or incorrect relation labels. To mitigate these issues and retain only high-confidence provenance-aware examples, we design and apply a rigorous three-stage filtering process.

Stage 1: Outcome Consistency and Schema Validation. The first stage serves as a foundational filter for correctness and structural integrity. We first verify whether the reconstructed final answer is semantically consistent with the reference answer A_{gt}. Samples whose final conclusion contradicts the reference answer are strictly removed, as their provenance annotations might explain an incorrect reasoning path rather than the intended solution. Following outcome validation, we utilize a rule-based parser to validate the JSON provenance schema. Samples exhibiting parsing errors, malformed JSON structures, missing sentence-level provenance, invalid relation labels, nonexistent tool identifiers, or incomplete evidence fields are automatically discarded.

Stage 2: Counterfactual Evidence Removal. The second stage performs a rigorous evidence sensitivity check. For a claimed evidence unit e_{ij}, we construct a counterfactual context in which this specific evidence unit is removed or masked, while the remainder of the tool trajectory is perfectly preserved. The verifier then re-evaluates whether the generated answer and its supporting provenance remain stable under this modified context. If the prediction or the provenance structure becomes

unstable after removing the cited evidence, we treat the record as exhibiting behavioral reliance under the verifier. Conversely, if the answer remains unchanged, it suggests that the cited evidence might be redundant, recoverable from another observation, or already embedded within the model's parametric knowledge. Therefore, rather than interpreting counterfactual removal as absolute causal proof of necessity, we utilize it as a conservative filtering signal to identify explicitly evidence-sensitive provenance records.

Stage 3: LLM-as-a-Judge Provenance Verification. Finally, an LLM-as-a-judge verifier is deployed to perform a deep semantic check on trajectory alignment and relation rationality. Trajectory alignment strictly requires that every cited tool identifier corresponds to an actual tool call within the original trajectory, and that every cited evidence unit is authentically traceable to the selected observation. This effectively prevents the model from hallucinating citations or misattributing evidence. Relation rationality evaluates whether the declared relation type logically matches the semantic operation executed between the cited evidence and the generated claim. Records containing unsupported evidence, incorrect tool alignment, or implausible evidence–relation–claim triples are permanently removed.

C.5 Tool Inventory

Tool	Function	Observation Type
ImageDescription	Produces a global description of the input image, including salient objects, scene layout, visual attributes, and visible entities.	Global visual description
OCR	Extracts text visible in the image.	Recognized text
TextToBbox	Grounds a textual object or phrase to its corresponding image region.	Bounding box or grounded region
RegionAttributeDescription	Describes attributes of a specified image region, such as color, shape, texture, or local semantics.	Region-level visual description
CountGivenObject	Counts instances of a specified object category in the image.	Object count
GoogleSearch	Retrieves external textual knowledge for open-domain entity or fact verification.	Retrieved textual snippet
Calculator	Performs deterministic arithmetic or symbolic computation.	Computed value

Table 4: Tool inventory in TRACE-BENCH. The retained tools cover global visual perception, localized inspection, OCR, counting, retrieval, and deterministic computation.

Relation	Definition	Example
QUOTATION	The claim directly reuses or reports information explicitly stated in the tool observation, with minimal semantic transformation.	OCR returns "Yamaha", and the claim states that the visible text reads "Yamaha".
COMPRESSION	The claim faithfully condenses a longer or redundant observation into a shorter statement without introducing new reasoning.	An image description lists multiple visual details, and the claim summarizes that the image shows a Yamaha-branded motorcycle.
INFERENCE	The claim is derived by combining, comparing, or computing over one or more evidence units, while remaining grounded in the cited observations.	A search result gives the first racing date and the image identifies the manufacturer; the claim answers when that manufacturer first entered motorcycle racing.

Table 5: Relation taxonomy used in TRACE-BENCH. The taxonomy distinguishes direct reuse, faithful condensation, and grounded derivation.

Table 4 summarizes the seven tools retained from the source trajectories in TRACE-BENCH. These tools cover visual perception, localized image inspection, textual extraction, external knowledge retrieval, counting, and deterministic computation. For each tool, we report its functionality and the typical form of its observation returned to the agent.

C.6 Provenance Relation Taxonomy

Table 5 defines the three relation types used in TRACE-BENCH. The taxonomy describes how a cited evidence unit supports a generated claim, rather than simply identifying the source of the evidence.

D Provenance Generator Prompt

Multi-Tool VQA Provenance Generator Prompt

You are a "Multi-Tool VQA Provenance Answer Generator".

1. Input Data

You will receive structured input data containing:

- `image_path`
- `context`: A sequential multi-tool invocation chain; each step contains `name`, `thought`, `input`, `output`, etc.
- `question`: The question text.
- `answer`: The final correct answer.

2. Core Task

1. First, generate a complete user-facing **response**: It must reasonably and coherently answer the `question`, forming a closed explanatory loop.
2. Then, split the `response` into multiple continuous sentences/paragraphs (`sentence`) to form a `sentence` list; each `sentence` in the list must have verifiable `provenance`.

Note: You absolutely do not need, nor are you allowed, to use the following fields (even if they exist in the input):

- `thought_rethink`
- `thought_question`

3. Output Format (Must be strictly consistent)

You can only output the following JSON (no additional explanatory text is allowed):

```
{
  "role": "assistant",
  "content": {
    "response": "...",
    "sentence": [
      {
        "sentence_id": 1,
        "text": "...",
        "provenance": [
          {
            "tool_id": "...",
            "source_text": "...",
            "relation": "..."
          }
        ]
      }
    ]
  }
}
```

4. Response Generation Rules (Ensure completeness and rationality)

1. `response` **must answer the** `question`:
   - The `response` must directly address what the `question` asks (e.g., "How many years did the author live?" must explicitly state the number of years in the `response`).
   - The `response` must contain the literal content of the final `answer` (e.g., "55 years").
2. **Restrictions on information sources for the** `response` **(Mandatory)**:
   - The semantic content of the `response` can only come from two parts: A) `context[i].thought` (integrated and rewritten sequentially for i=1..n), and B) `answer` (used only as the final conclusion).

- It is not allowed to introduce new facts outside of `context.thought` and `answer`.
- It is not allowed to quote or paraphrase fields like `thought_rethink`, `thought_question`, `thought_choose`, or `thought_query`.

3. **Structural requirements for the `response` (Mandatory logical closed loop)**: The `response` must contain at least the following logical units (can be expressed in natural language, explicit numbering is not required):
   - Identify/determine the subject of the question (e.g., identify the author from the image, or determine key entities).
   - Obtain key supporting facts (e.g., get birth and death years via OCR/search).
   - (If needed) Deduce or calculate to get the answer.
   - Provide the final conclusion (containing the `answer`).

5. Sentence Generation Rules (Split from response)

1. `sentence` **must be split from the** `response`:
   - The `text` of all sentences in the `sentence` list, when concatenated by `sentence_id` from 1..k (allowed to be connected by spaces or line breaks), must strictly restore the content of the `response` (semantics and expression must be consistent).
   - Sentences not present in the `response` are not allowed to appear in the `sentence` list.
   - It is not allowed to have sentences in the `response` that do not appear in the `sentence` list.
2. **Quantity of** `sentences`:
   - Determined based on the answer length and the number of tool invocations (i.e., the length of the input `context`). If the answer is long, the quantity can appropriately exceed the number of tool invocations; if some tools are useless, it can be fewer. Ensure the reasoning chain is complete.
3. `sentence_id`:
   - Incrementing from 1, strictly increasing, no skipping numbers.
   - The order of `sentence_id` must be consistent with the order in which the sentences appear in the `response`.

6. Provenance Rules (Strong constraint, verifiable)
Each `sentence` must have at least 1 `provenance`.

1. `tool_id` **naming rules (Mandatory)**:
   - `tool_id` = `<context[i].name>_<i>`
   - `i` starts from 1 (the first element of `context` is 1).
   - Examples: `ImageDescription_1`, `OCR_2`, `GoogleSearch_3`.
2. `source_text` **rules (Most important, must be verifiable)**:
   - It must contain enough information to support the `text` of the `sentence`.
   - `provenance.source_text` must be an exact substring of the corresponding `context[i].output` (character-by-character match).
   - Fabricating `source_text` is not allowed.
   - For the "final answer sentence", its `provenance` must quote the tool output fragment that directly supports the `answer` (usually the last key tool output providing key numbers/facts).
3. `relation` **enumeration (Must choose one of four)**:
   - `Quotation`: Almost copied word-for-word from the original text.
   - `Compression`: Compressing and summarizing (still based on the original text).
   - `Inference`: Deduce/calculate based on the source (if `Inference` is used, `source_text` must contain the key facts required for deduction, such as years/numbers).

7. Consistency Hard Constraints & Prohibitions

- The `response` must look like a real assistant's answer to a user: natural, coherent, and directly answering the question.
- The `response` must contain a chain of explanation, not just list `thoughts`.

- The `sentence` list must be a breakdown of the `response`, not a mechanical splicing of `context.thought`.
- The `provenance` of each `sentence` must be directly related to the content of that `sentence`; do not provide irrelevant provenance.
- **Strictly Prohibited**: Outputting any text other than JSON; Using a `tool_id` that does not exist in `context`; Any sentence lacking `provenance`; `source_text` not found in the corresponding `output`; Using undefined categories for `relation`; Using information from prohibited fields like `thought_rethink`/`thought_question`; Introducing new facts.

Figure 6: Prompt used for the Multi-Tool VQA Provenance Answer Generator. The model generates a coherent response answering the question, splits it into verifiable sentences, and strictly maps each sentence to a provenance source within the tool execution chain.

Prompt for LLM-as-Verifier on Provenance Accuracy

1. Task Description You need to judge whether the provenance provided by the model is **completely correct** based on the **multi-round tool interaction history** and the **provenance information in the model's final output**, strictly following the 3 rules below.

2. Validation Rules

1. **Tool and Round Validation**
   - `tool_id` format: `ToolName_Number`
   - Number = the **N-th time this tool is called** (incrementing from 1)
   - Must be completely consistent with the `tool_call` order in `messages`
2. **Source Authenticity Validation**
   - `source_text` must **completely come from** the original `tool_response` result of that specific `tool_id` round
   - Fabricated, spliced, replaced, or non-existent content is not allowed
3. **Text and Relation Rationality Validation**
   For each `sentence_id`:
   - `Quotation`: The sentence is **directly copied/extracted** from `source_text`
   - `Compression`: The sentence is a **simplification, summary, or compression** of `source_text`
   - `Inference`: The sentence is derived through **reasonable deduction** based on `source_text`
   - Must judge: Whether `text` matches `source_text` + `relation`

3. Input Format
You will receive a complete JSON containing:

- `images`: Image paths.
- `tools`: List of available tools.
- `messages`: Complete multi-round dialogue of user / model / tool calls / tool returns.
- `solution`: The model's final output, with `sentence` and `provenance` tracing.

4. Output Requirements
Please **only output the structured judgment result**, in the following format:

```
{
  "overall_correct": true/false,
  "error_details": [
    "Error 1 description",
    "Error 2 description"
  ],
  "sentence_check": [
    {
      "sentence_id": number,
      "tool_id_correct": true/false,
      "source_text_correct": true/false,
      "relation_correct": true/false,
```

```
      "sentence_correct": true/false
    }
  ]
}
```

Figure 7: Prompt used to evaluate the correctness of the model's multi-tool provenance. The evaluator verifies tool invocation rounds, source text authenticity, and text-relation rationality, outputting a structured JSON validation result.