TOOLSCIVER: MULTIMODAL SCIENTIFIC CLAIM VERIFICATION WITH VISUAL TOOL AUGMENTED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal scientific claim verification (MSCV) checks claims against text, tables, and figures of scientific papers. It requires extracting multiple pieces of localized evidence and combining them through multi-step reasoning; vision-language models (VLMs) struggle with both. Generic crop-and-zoom tools only return pixels, and prompting VLMs to use tools can even hurt accuracy. We present ToolSciVer, which teaches VLMs when and how to acquire claim-relevant evidence. It pairs SciEviKit, a type-aware tool suite returning structured evidence (table rows or columns, chart values, zoomed regions), with two-stage training: On-Policy Self-Distillation (OPSD) bootstraps tool use from reference tool calls alone, and Group Relative Policy Optimization (GRPO) optimizes a reward coupling accuracy with tool-call validity and efficiency. We evaluate on SCIVER and MUSCICLAIMS, spanning diverse evidence modalities, domains, and reasoning structures, with three backbones (Qwen, InternVL, Gemma) and five baselines covering non-tool reasoning, training-free agents, and RL-based tool use. ToolSciVer improves over non-tool chain-of-thought by 8.16 points on SCIVER and 7.92 on MUSCICLAIMS, and over the strongest baselines by 3.70 and 2.23 points. Our analyses further show better evidence acquisition, tool selection, and evidence-based reasoning, and ablations confirm the value of OPSD warm-up.[1]

1 INTRODUCTION

Multimodal scientific claim verification (MSCV) is critical for building trustworthy scientific information systems (Wang et al., 2025a; Lal et al., 2025). In MSCV, a model determines whether a claim is supported or refuted by evidence that spans both textual descriptions and scientific visuals. Unlike traditional textual fact verification (Thorne et al., 2018; Wadden et al., 2020; 2022b;a; Alvarez et al., 2024), MSCV requires integrating heterogeneous modalities by addressing two key challenges: first extracting localized, structured visual evidence, and then performing multi-step reasoning over that evidence to verify the claim (Wang et al., 2025a).

Despite growing interest, existing approaches to MSCV remain limited in addressing these challenges. Recent studies highlight that frontier proprietary models, such as GPT, Claude, and Gemini, still struggle particularly when correct verification depends on precise interpretation of scientific tables, charts, and figures (Wang et al., 2025a; Lal et al., 2025; Ho et al., 2026a;b). They often fail to localize claim-relevant visual evidence and struggle to aggregate information across modalities via reliable grounded multi-step reasoning (Lal et al., 2025; Wang et al., 2025a; Ho et al., 2026b). Meanwhile, recent tool-augmented VLMs use external visual tools, including image-editing operations such as cropping, zooming, and highlighting, to support visual reasoning (Fu et al., 2025; Wu et al., 2025; Su et al., 2025). However, scientific verification often requires explicit table entries or chart values for comparison with a claim, in addition to closer visual inspection. Moreover, tool access alone does not ensure effective verification: our experiments show that prompting a VLM to use tools can even reduce accuracy. Table 1 compares representative tool-use methods.

In this paper, we introduce ToolSciVer, a type-aware tool-augmented framework that integrates visual evidence acquisition with two-stage policy training for MSCV. To address the above challenges,

[1] Our code is provided in supplementary material and will be released upon acceptance.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

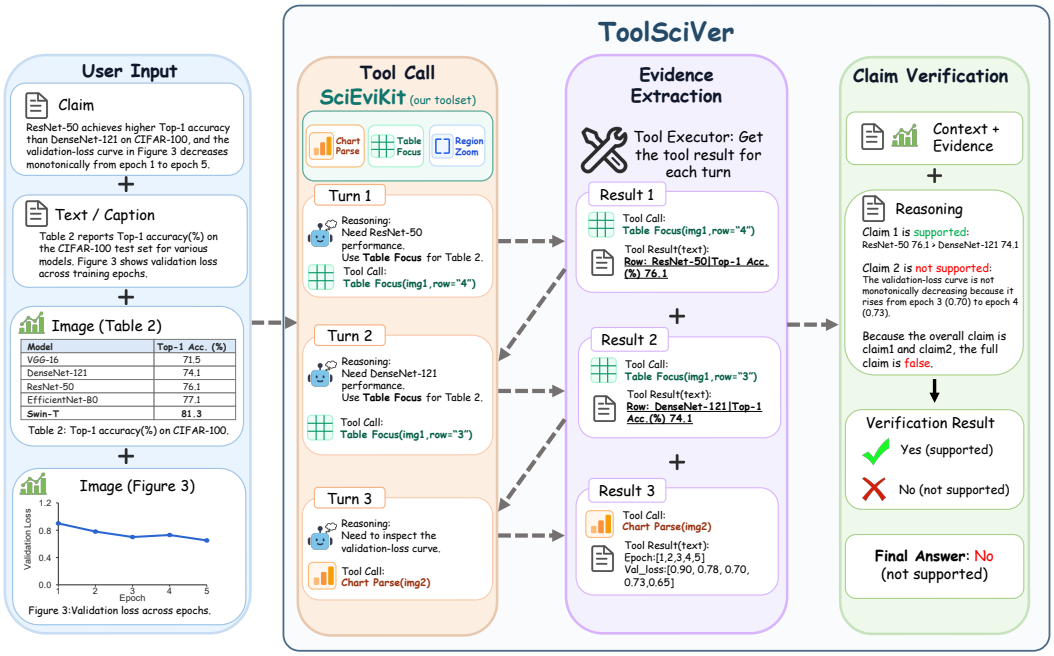


Figure 1: Overview of ToolSciVer framework for evidence-grounded tool-augmented scientific claim verification. Given a claim, textual context, and scientific visual evidence, the model can either answer directly or iteratively call tools from SciEviKit to acquire claim-relevant evidence. The collected tool observations are added to the augmented context and used for final claim verification.

ToolSciVer incorporates two main innovations: (1) SciEviKit, a suite of specialized tools for extracting claim-relevant visual evidence from scientific papers; (2) a two-stage training strategy that combines self-distillation-based warm-up with reinforcement learning under a tailored composite reward to promote efficient tool use and multi-step reasoning over the extracted evidence. Specifically, as shown in Figure 1, our framework equips a VLM with SciEviKit. It provides three tool families for different types of scientific visuals: a Table Focus tool that returns localized row or column evidence from tables, a Chart Parse tool that converts plots into structured textual observations, and a Region Zoom tool that returns high-resolution crops of selected regions in general or multi-panel figures. Furthermore, effective use of SciEviKit requires the model to determine whether a tool is needed, which tool matches the visual type, which arguments identify the relevant evidence, and how to integrate the returned observations into its verification decision. Directly learning tool use with reinforcement learning (RL) is challenging because an unreliable initial tool-calling policy can destabilize policy optimization. Conventional supervised fine-tuning (SFT) warm-up can establish an initial policy, but collecting complete tool-use and reasoning demonstrations typically requires stronger models or human experts to generate and verify these trajectories (Su et al., 2025; Geng et al., 2025). We therefore first warm up the model using On-Policy Self-Distillation (OPSD) (Zhao et al., 2026), which requires only reference tool-call annotations. We then apply Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to improve the effectiveness of tool use and strengthen multimodal reasoning. For the GRPO stage, we design a composite reward that combines final-answer correctness, output-format validity, length control, tool-use efficiency, and tool-call correctness, encouraging the model to acquire evidence only when needed.

To demonstrate the effectiveness of our method, we conduct a comprehensive set of experiments on two challenging MSCV benchmarks, SciVer (Wang et al., 2025a) and MuSciClaims (Lal et al., 2025), which contain complex claims to verify with diverse evidence modalities, scientific subdomains, and reasoning structures. We apply ToolSciVer to three backbone model families (Qwen, InternVL, and Gemma) and compare against five baselines spanning non-tool reasoning, training-free agents, and RL-based tool-use methods. Across the three 4B-scale backbones, ToolSciVer consistently improves overall accuracy over non-tool CoT, with average gains of 8.16 percentage points on SciVer and 7.92 points on MuSciClaims. It also outperforms the strongest baseline for each backbone and benchmark in Table 2, with average gains of 3.70 and 2.23 percentage points on

Table 1: Comparison with representative visual tool-use methods. For visual-tool columns, ✓ denotes dedicated evidence extraction, ◯ generic or partial support, and ✗ no explicit support. *Struct. Obs.* refers to structured observations directly usable as verification evidence, such as table rows/columns or chart values, rather than raw OCR text, bounding boxes, or edited images.

		Claim-Facing Visual Evidence Tools				Learning Strategy	
Method	**Target Task**	**Table Evid.**	**Chart Evid.**	**Region Insp.**	**Struct. Obs.**	**Training Pipeline**	**Eff. Reward**
VisProg (Gupta & Kembhavi, 2023)	Gen. visual reasoning	✗	✗	◯	✗	Training-free	✗
ViperGPT (Surís et al., 2023)	Gen. visual reasoning	✗	✗	◯	✗	Training-free	✗
MM-ReAct (Yang et al., 2023)	Gen. multimodal reasoning	✗	✗	◯	✗	Training-free	✗
Visual Sketchpad (Hu et al., 2024)	Visual/math reasoning	✗	✗	◯	✗	Training-free	✗
ReFocus (Fu et al., 2025)	Struct. img. underst.	◯	◯	◯	✗	Prompt/SFT	✗
DePlot/MatCha (Liu et al., 2023a;b)	Chart understanding	✗	✓	✗	✓	Sup./Pretrain	✗
VTool-R1 (Wu et al., 2025)	Struct. visual reasoning	◯	◯	◯	✗	GRPO	✗
OpenThinkIMG (Su et al., 2025)	Chart/tool reasoning	✗	◯	◯	✗	SFT → GRPO	✗
ToolSciVer (Ours)	**MSCV**	✓	✓	✓	✓	**OPSD → GRPO**	✓

SciVer and MuSciClaims, respectively. Further analyses show improvements in visual evidence acquisition, tool selection, and reasoning over extracted evidence across tables, charts, and general scientific figures. The ablation study further highlights the importance of the OPSD warm-up stage, as removing this stage reduces overall performance.

Our contributions are summarized as follows:

- We propose ToolSciVer, a visual tool-augmented post-training framework for MSCV. It equips a VLM with SciEviKit for acquiring claim-relevant scientific evidence and trains the model through OPSD warm-up followed by GRPO-based reinforcement learning to enable selective, valid, and efficient tool use.
- We empirically validate the effectiveness of ToolSciVer across challenging MSCV benchmarks in multiple scientific domains and reasoning types, with further analyses of evidence acquisition, tool selection, efficiency, and evidence-based reasoning.

2 Related Work

Scientific and multimodal claim verification. Scientific verification has progressed from text-based evidence retrieval and prediction (Wadden et al., 2020; 2022b;a; Alvarez et al., 2024) to table-grounded and multimodal settings (Lu et al., 2023; Wang et al., 2025a; Lal et al., 2025; Wang et al., 2025b; Zhou et al., 2025; Kishore et al., 2025). Recent evaluations expose sensitivity to table and chart formats and persistent difficulty with figure-based verification (Ho et al., 2026a;b). We focus on acquiring the claim-relevant visual evidence needed for a reliable verification decision.

Tool-augmented VLMs. Early visual agents combined language models with programs, external modules, and image manipulation (Gupta & Kembhavi, 2023; Surís et al., 2023; Yang et al., 2023; Hu et al., 2024; Fu et al., 2025). More recently, ChartAgent uses chart-specific visual actions for question answering (Kaur et al., 2026), while DocLens localizes evidence within long visual documents (Zhu et al., 2026). VTool-R1 and OpenThinkIMG learn visual tool use through reinforcement learning (Wu et al., 2025; Su et al., 2025); Mags-RL further trains region selection and super-resolution-assisted inspection (Dong et al., 2026b). As summarized in Table 1, ToolSciVer specializes visual tool use to MSCV by learning when and how to acquire evidence through the table, chart, and region tools in SciEviKit.

On-policy distillation and reinforcement learning. On-policy distillation (OPD) supervises student-generated trajectories with teacher distributions, reducing the mismatch between training and inference (Agarwal et al., 2024). OPSD obtains this guidance from the same model conditioned on privileged context (Zhao et al., 2026), and privileged information distillation transfers agent behavior using action-only traces (Penaloza et al., 2026). Recent work couples distillation with verifiable rewards (Lin et al., 2026) and studies OPD as initialization for subsequent RL (Dong et al., 2026a). ToolSciVer adapts this line of work to multi-turn visual verification: reference tool calls guide OPSD warm-up without complete reasoning demonstrations, and GRPO then optimizes verification accuracy together with tool-call validity and efficiency.

3 METHOD

We study MSCV, where a model verifies a scientific claim using textual context and scientific visual evidence, as illustrated in the "User Input" column of Figure 1. An instance is $x = (q, \mathcal{V}, t)$, comprising a claim q, scientific visuals $\mathcal{V} = \{v_i\}_{i=1}^{N}$, and optional textual context t. The model predicts $\hat{y} \in \mathcal{Y} = \{\text{supported}, \text{refuted}\}$.

A key challenge is that decisive evidence is often visually localized and structurally constrained, requiring the model to read specific table entries, recover chart trends, or inspect local regions of dense figures. To address this challenge, ToolSciVer uses SciEviKit to acquire claim-relevant evidence for the final verification decision (Figure 1). At each turn, the policy either answers directly or requests a tool; the scheduler executes valid calls and adds the returned observations to the context for further reasoning. Conventional SFT-based tool-use warm-up typically requires stronger models or human experts to produce complete tool-call and reasoning trajectories, making data construction costly. To reduce this supervision burden, we first use OPSD to establish tool-use competence only with reference tool calls as privileged guidance. We then apply GRPO to optimize verification accuracy and selective tool use through outcome-based feedback.

3.1 SCIEVIKIT: A TYPE-AWARE VISUAL TOOL SUITE

Generic visual tools such as cropping and zooming improve local visibility but still leave small VLMs to recover table structure and chart values from pixels. We therefore design SciEviKit, a compact tool suite for three common evidence structures in MSCV: tables, charts, and general scientific figures. Each tool call targets one image through an `image_path` argument, and the policy decides which tool to call and what arguments to provide.

Table Focus. Scientific claims often depend on localized tabular evidence, such as a method's performance on a specific dataset or the value of a particular metric. For table images, we first convert the image into a structured CSV-like representation using OCR. We then expose `focus_row` and `focus_column`, which return only the selected row or column. This reduces irrelevant context and makes the returned evidence easier to compare against the claim.

Chart Parse. Claims over charts or plots often require recovering trends, relative ordering, thresholds, or numerical values. For chart images, `parse_content` uses a stronger VLM to convert visual chart content into structured textual observations, such as JSON-like entries of chart elements and values.

Region Zoom. Many scientific figures are dense, multi-panel, or diagrammatic, and the decisive evidence may occupy only a small local region. For these cases, `image_zoom_in` crops and enlarges a model-selected region, returning a zoomed-in image. Coordinates are normalized to a $[0, 1000]$ scale, making the interface resolution-invariant.

Detailed tool interfaces are provided in Appendix A.1.

3.2 TRAINING STAGE I: TOOL-USE WARM-UP VIA OPSD

Learning tool use directly with RL is challenging: the initial VLM lacks a stable tool-calling policy, leading to unreliable interactions that can destabilize policy optimization. We therefore introduce a warm-up stage to establish a more reliable initial SciEviKit tool-use policy before RL.

SFT warm-up provides rich supervision through complete tool-use and reasoning demonstrations (Su et al., 2025; Geng et al., 2025), but collecting them typically requires stronger teacher models or human experts to generate and verify tool calls, intermediate reasoning, and final responses. We instead use OPSD (Zhao et al., 2026) for warm-up, requiring only reference tool-call annotations rather than complete demonstration targets. A teacher conditioned on these reference calls provides token-level supervision along the student's own sampled trajectories.

Let π_0 be the initial VLM and π_θ the student initialized from it. Following prior work on action-based privileged information (PI) (Penaloza et al., 2026), we provide the ordered reference tool calls, including their arguments, as PI z for each warm-up instance. The construction of this *tool-PI* is described in Appendix B.1. A frozen copy of π_0 serves as the teacher, conditioned on z in addition to the student's interaction history; z is never provided to the student.

The student samples a multi-turn trajectory $\tau \sim \pi_\theta(\cdot \mid x)$ by interacting with the tools. The teacher evaluates the same sampled assistant tokens, using the same preceding actions, online observations, and derived images. Let $\mathcal{A}(\tau)$ index all assistant-generated tokens, and let h_ℓ contain the original input x and the student's interaction history before token ℓ. At each such position, the teacher and student distributions are $p_\ell^T = \pi_0(\cdot \mid h_\ell, z)$ and $p_\ell^S = \pi_\theta(\cdot \mid h_\ell)$. We minimize

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = \mathbb{E}_{(x,z),\,\tau\sim\pi_\theta(\cdot|x)}\left[\frac{1}{|\mathcal{A}(\tau)|}\sum_{\ell\in\mathcal{A}(\tau)}\mathcal{D}_\delta(p_\ell^T, p_\ell^S)\right], \tag{1}$$

where $\mathcal{D}_\delta$ is a clipped forward-KL loss, defined in Appendix A.3. Teacher probabilities are treated as fixed targets. The loss covers all assistant turns, including tool calls and the final response; user inputs and tool observations provide context but receive no loss. Thus, reference tool calls guide the teacher's predictions, while the student learns from feedback on its own sampled trajectories. The resulting policy π_{warm} initializes the second stage.

3.3 Training Stage II: Verification Policy Optimization via GRPO

Starting from π_{warm}, we apply GRPO (Shao et al., 2024) to optimize complete verification trajectories. The warm-up supplies an initial tool-use policy, while this stage couples final-answer correctness with the validity and efficiency of evidence acquisition. We keep a frozen copy of π_{warm} as the reference policy π_{ref}. For each input x, we sample a group of G trajectories $\{\tau_i\}_{i=1}^G$ from the current rollout policy π_{old}, then score and compare them within the group.

Composite reward. For rollout τ, we combine answer correctness, output format, length control, tool efficiency, and tool-call validity:

$$R(\tau) = s_{\mathrm{eff}}(\tau)\big(r_{\mathrm{ans}}(\tau) + \lambda_{\mathrm{fmt}} r_{\mathrm{fmt}}(\tau)\big) + \lambda_{\mathrm{len}} r_{\mathrm{len}}(\tau) + \lambda_{\mathrm{tool}} r_{\mathrm{tool}}(\tau). \tag{2}$$

Here, $r_{\mathrm{ans}} = \mathbb{I}[\hat{y} = y]$ measures correctness against the ground-truth label y; $r_{\mathrm{fmt}} \in \{0, 0.5, 1\}$ checks the required reasoning and normalized-answer format; and $r_{\mathrm{len}} \in [-1, 0]$ softly penalizes overlong trajectories (Yu et al., 2025). The default coefficients are $\lambda_{\mathrm{fmt}} = \lambda_{\mathrm{len}} = \lambda_{\mathrm{tool}} = 0.1$.

For K tool attempts, the tool-call reward is $r_{\mathrm{tool}} = K^{-1}\sum_{j=1}^K q_j$ when $K > 0$, and zero for a no-call trajectory. A valid, type-compatible, successfully executed model call receives $q_j = +1$; an invalid, incompatible, failed, or over-budget attempt receives $q_j = -1$. Per-image annotations specify which tools are compatible with each image, without supplying an answer or requiring a call. This reward assesses tool-call validity; the answer reward assesses verification accuracy.

Following OTC (Wang et al., 2025c), s_{eff} compares the call count $m(\tau)$ with $n^\star(x) = \min_{\tau_i\in\mathcal{C}(x)} m(\tau_i)$, where $\mathcal{C}(x)$ contains the group's rollouts with a correct label and valid final-answer format. It favors direct answers when a successful no-call trajectory exists, and otherwise favors call counts near $n^\star$. If $\mathcal{C}(x)$ is empty, $s_{\mathrm{eff}} = 1$. A detailed description of the RL reward design is provided in Appendix A.4.

Policy optimization. After rollout completion, we assign each trajectory a scalar reward $R_i = R(\tau_i)$ using Eq. 2 and compute the group-relative advantage $A_i = (R_i - \mathrm{mean}_j\, R_j)/(\mathrm{std}_j\, R_j + \epsilon_A)$, with a small stabilizer ϵ_A. We then maximize the GRPO objective estimated from this group, favoring higher-reward trajectories while keeping the policy close to π_{ref}:

$$\widehat{\mathcal{J}}_{\mathrm{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{\ell=1}^{T_i}\min\left(\rho_{i,\ell}(\theta)A_i,\ \mathrm{clip}(\rho_{i,\ell}(\theta), 1-\epsilon, 1+\epsilon)A_i\right) - \beta\overline{D}_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}}). \tag{3}$$

Here, $T_i = |\mathcal{A}(\tau_i)|$ counts assistant-generated tokens in τ_i, and ℓ indexes these tokens. The token-level ratio $\rho_{i,\ell}(\theta)$ compares π_θ with π_{old} under the same interaction history. The KL term $\overline{D}_{\mathrm{KL}}$ averages the conditional token-level divergence within each trajectory and then across the group. The parameters ϵ and β control clipping and the KL penalty, respectively. Tool observations remain in the conditioning history but receive no policy loss. Further optimization details are provided in Appendix A.4. At inference, the trained policy verifies claims using only the task input and observations returned by its tool call.

4 EXPERIMENTAL SETUP

Datasets. We evaluate on two MSCV benchmarks: SCIVER (Wang et al., 2025a) and MUSCICLAIMS (Lal et al., 2025). SCIVER contains scientific claims paired with multimodal evidence from scientific documents and covers four reasoning subsets: *Analytic*, *Direct*, *Parallel*, and *Sequential*. MUSCICLAIMS contains figure-centric scientific claims from biology, physics, and chemistry. We use the binary verification setting for both datasets and report subset/domain accuracy together with sample-weighted overall accuracy. Appendix B.1 details the train/test splits and the shared warm-up data for OPSD and SFT.

Models. We evaluate three open-source VLM backbones: Qwen3.5-4B (Qwen Team, 2026), InternVL3.5-4B (Wang et al., 2025d), and Gemma4-E4B (Farabet & Lacombe, 2026). All trainable methods are initialized from the same backbone and trained on the same MSCV training splits using a fixed training seed. We evaluate checkpoints every 50 GRPO training steps and report results for the checkpoint with the highest average accuracy for each trainable method. For all open-source methods, we report mean $\pm$ standard deviation over three evaluation runs with different inference seeds. Additional training and evaluation details are provided in Appendix B.

Baselines. We compare ToolSciVer with three training-free baselines: non-tool CoT (Wei et al., 2022), CHARTAGENT (Kaur et al., 2026), and DOCLENS (Zhu et al., 2026). The trained baselines are VTOOL-R1 (Wu et al., 2025) and OPENTHINKIMG (Su et al., 2025), both adapted to MSCV. We also report proprietary VLMs (GPT-4o, GPT-5.4, GPT-5.6 Sol, Sonnet 4.6, Opus 5, and Opus 5.5) as external references under the non-tool CoT setting. For GPT-5.6 Sol and Opus 5.5, we additionally evaluate CoT with access to SciEviKit. Implementation details, prompts, and proprietary-model evaluation settings are provided in Appendix B.4.

5 RESULTS AND ANALYSIS

We evaluate the effectiveness of ToolSciVer on MSCV through comparisons with existing methods and proprietary VLMs across datasets and model families (Section 5.1), ablations of training strategies and the OTC efficiency term (Section 5.2), and analyses of tool selection, evidence acquisition, and reasoning over acquired evidence (Section 5.3).

5.1 MAIN RESULTS

Table 2 shows two main trends. First, ToolSciVer consistently improves over non-tool CoT on both benchmarks across all three open-source backbones. Second, it outperforms the training-free agent frameworks and the trained visual tool-use baselines VTOOL-R1 and OPENTHINKIMG, achieving the highest overall accuracy among the compared existing methods within each backbone. On Qwen3.5-4B, ToolSciVer achieves 84.07% on SCIVER, exceeding the strongest baseline by 2.89 percentage points, and 80.79% on MUSCICLAIMS, a gain of 4.01 points. Its gains extend to all four reasoning categories in SCIVER and all three scientific domains in MUSCICLAIMS.

Comparisons with proprietary VLMs show that ToolSciVer enables small open-source VLMs to achieve competitive MSCV performance. ToolSciVer with Qwen3.5-4B outperforms all proprietary references on SCIVER and approaches Opus 5 on MUSCICLAIMS. Some advanced proprietary VLMs perform surprisingly poorly on SCIVER. We analyze their errors in Appendix C.6.

5.2 ABLATION STUDY

Training for tool use. Prompt-only CoT gives the model access to SciEviKit and interleaves reasoning with tool calls (Yao et al., 2023). It improves Qwen3.5-4B over non-tool CoT but remains below the trained variants in Table 3. With the same tools and no warm-up, GRPO raises accuracy from 73.13% to 82.40% on SCIVER and from 71.68% to 77.23% on MUSCICLAIMS. Prompt-only tool access is also inconsistent across backbones: it reduces SCIVER accuracy for InternVL3.5-4B and Gemma4-E4B (Appendix C.2). These results support learning a verification policy beyond simply making tools available through prompting.

Warm-up strategy. Adding OPSD warm-up before tool-enabled GRPO improves accuracy by 1.67 points on SCIVER and 3.56 points on MUSCICLAIMS, surpassing no-tool GRPO on both bench-

Method / Model	SciVer					MuSciClaims			
	Analytic	Direct	Parallel	Sequential	*Overall*	Bio	Phy	Chem	*Overall*
`Qwen3.5-4B`									
CoT w/o SciEviKit	72.92±0.90	75.00±1.40	68.57±0.41	71.32±5.22	71.97±1.76	65.58±4.78	55.55±4.21	71.52±2.44	64.88±3.61
ChartAgent	73.83±1.03	78.36±2.29	71.97±2.32	75.43±1.51	74.90±0.89	80.28±0.68	66.32±2.17	73.79±0.97	76.30±0.98
DocLens	62.89±3.40	66.94±2.52	59.05±2.09	57.37±1.99	61.57±2.32	74.18±0.56	69.10±1.20	70.55±4.04	72.48±0.86
VTool-R1	84.40±0.40	84.77±1.11	75.99±1.03	79.38±0.87	81.18±0.03	77.34±2.13	71.15±1.80	80.37±1.10	76.78±0.75
OpenThinkIMG	74.20±0.68	76.49±0.82	73.82±0.89	79.27±1.07	75.94±0.23	77.27±0.49	67.94±2.82	77.79±1.06	75.60±0.94
ToolSciVer	**88.02**±1.58	**90.32**±2.64	**76.46**±1.18	**81.27**±2.39	**84.07**±1.27	**80.72**±1.18	**81.25**±2.76	**80.58**±0.97	**80.79**±0.52
`InternVL3.5-4B`									
CoT w/o SciEviKit	71.91±2.29	68.96±3.04	62.89±0.76	64.61±3.52	67.12±0.68	70.91±0.98	56.95±3.66	69.26±6.32	67.92±2.33
ChartAgent	60.16±2.82	53.76±3.24	52.79±1.55	60.16±2.39	56.77±1.92	63.73±3.00	57.29±5.21	70.23±2.44	63.83±3.28
DocLens	66.80±2.07	66.40±2.22	54.96±1.65	59.76±1.05	62.03±0.55	67.43±3.11	60.76±5.74	69.90±4.45	66.67±0.75
VTool-R1	74.98±1.00	**69.18**±2.13	66.29±0.91	70.23±1.94	70.22±1.49	73.44±3.48	66.37±1.94	72.05±2.17	71.81±2.14
OpenThinkIMG	71.32±0.19	66.56±1.12	64.79±2.30	68.12±1.19	67.74±0.96	73.97±1.73	**70.16**±1.24	73.20±0.55	73.09±1.02
ToolSciVer	**78.39**±1.80	69.09±2.68	**69.93**±3.32	**73.71**±1.38	**72.83**±1.16	**74.95**±2.62	65.62±4.77	**77.99**±3.12	**73.80**±1.77
`Gemma4-E4B`									
CoT w/o SciEviKit	78.78±0.81	75.27±1.91	71.02±1.78	72.91±1.38	74.53±0.57	**74.18**±1.82	67.01±2.17	73.46±3.68	72.67±0.86
ChartAgent	77.60±2.15	73.39±0.70	74.15±3.43	77.16±1.66	75.60±1.42	73.31±1.05	65.28±0.60	**75.40**±3.12	72.21±0.12
DocLens	75.13±2.00	72.04±2.07	62.04±4.02	65.87±1.51	68.83±0.32	72.66±3.03	66.32±2.62	70.55±0.56	71.02±2.03
VTool-R1	79.17±0.22	74.46±3.43	69.12±1.65	73.97±1.22	74.23±1.16	69.61±0.87	62.15±5.35	72.49±5.35	68.78±0.80
OpenThinkIMG	80.99±0.45	80.11±3.38	63.13±2.25	65.07±2.00	72.40±1.61	71.79±0.50	68.40±3.35	71.85±1.94	71.16±0.98
ToolSciVer	**82.29**±0.23	**84.01**±1.01	**78.10**±1.25	**80.35**±1.66	**81.20**±0.95	73.20±0.57	**78.47**±4.21	**75.40**±3.41	**74.65**±0.59
Proprietary models *CoT w/o SciEviKit*									
`GPT-4o`	71.20	77.00	73.60	73.80	73.89	75.49	67.71	71.84	73.27
`GPT-5.4`	53.52	75.00	66.53	56.97	62.90	78.76	68.75	70.87	75.25
`GPT-5.6 Sol`	54.69	77.82	64.08	61.35	64.40	82.68	72.92	76.70	79.60
`Sonnet 4.6`	71.48	79.44	65.31	65.74	70.50	83.01	72.92	74.76	79.41
`Opus 5`	65.23	79.44	67.76	64.14	69.10	86.27	69.79	78.64	81.58
`Opus 5.5`†	61.81	78.14	65.57	64.54	67.47	88.74	76.04	89.00	85.30
Proprietary models *CoT w/ SciEviKit*									
`GPT-5.6 Sol`	62.50	80.65	68.16	67.73	69.70	85.62	72.92	87.38	83.56
`Opus 5.5`†	63.39	78.95	65.16	64.14	67.87	91.22	77.08	89.00	86.63

Table 2: Main results on SciVer and MuSciClaims. Accuracy (%) is reported for each subset and as sample-weighted *Overall*. Open-source results are mean ± standard deviation over three evaluation runs; proprietary results are single-evaluation CoT references with or without SciEviKit. **Bold** and underline indicate the best and second-best means within each open-source backbone. †Opus 5.5 accuracy is computed after excluding refused examples.

Configuration	SciEviKit	GRPO	Warm-up	SciVer	MuSciClaims
CoT w/o Tool	-	-	-	71.97 ± 1.76	64.88 ± 3.61
CoT w/ Tool	✓	-	-	73.13 ± 2.67	71.68 ± 1.69
GRPO	-	✓	-	80.57 ± 0.74	77.89 ± 0.94
GRPO	✓	✓	-	82.40 ± 1.08	77.23 ± 2.23
SFT+GRPO	✓	✓	SFT	**84.53** ± 1.59	**82.51** ± 2.35
ToolSciVer	✓	✓	OPSD	84.07 ± 1.27	80.79 ± 0.52

Table 3: Ablation study on Qwen3.5-4B. Overall accuracy (%), mean ± standard deviation over three evaluations. **Bold** and underline indicate the best and second-best means. SFT uses complete demonstrations generated by a stronger model and reviewed by humans.

marks. Warm-up also benefits the other backbones (Appendix C.2), suggesting that it can help ToolSciVer establish an initial tool-use policy and achieve better verification performance.

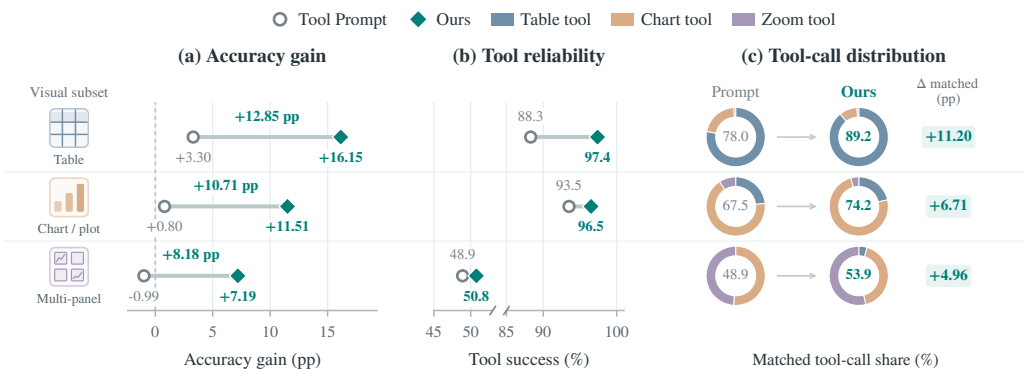


Figure 2: Tool-use behavior: (a) accuracy gain over non-tool CoT, (b) execution success (broken axis), and (c) tool-call distribution. Donut centers show the share (%) of calls to the tool family matched to each visual subset; the right column shows gains over Tool Prompt (pp). In (a,b), circles denote Tool Prompt and diamonds denote ToolSciVer.

SFT warm-up achieves slightly higher scores, with OPSD trailing by 0.46 points on SciVer and 1.72 points on MuSciClaims. We choose OPSD for its less demanding data requirements (Section 3.2) while retaining competitive verification accuracy.

Efficient tool use. We ablate the OTC efficiency term s_{eff} while keeping all other reward components and evaluation settings fixed. Both variants use a tool-encouraging test-time prompt. Table 4 shows that adding s_{eff} improves accuracy by 4.24 points on SciVer and 5.14 points on MuSciClaims. Calls per example decrease from 1.05 to 0.74, and average response length drops from 2,760 to 1,362 tokens. These results suggest that s_{eff} regularizes the policy toward shorter, more economical verification trajectories. The call-depth distribution (Appendix C.4) shows that the direct-answer rate increases from 5.2% to 28.8%, while 3.3% of examples still use two calls. One example uses separate calls to acquire complementary evidence for different clauses of a claim (Appendix C.7).

Table 4: OTC ablation. Accuracies are percentages; calls and tokens are averaged over both datasets.

s_{eff}	SciVer	MuSciClaims	Calls/Ex.	Tokens
-	76.96	73.47	1.05	2760.42
✓	**81.20**	**78.61**	**0.74**	**1361.58**

5.3 Analysis of MSCV

We analyze tool selection and reasoning over acquired evidence using Qwen3.5-4B. The diagnostic set contains 100 examples for each evidence type: tables, charts/plots, and multi-panel figures. Within each type, examples are sampled in proportion to category frequencies in the source pool, and the claim-relevant gold evidence is manually annotated from the original scientific visuals and claims. The sampling, annotation, and scoring protocols are detailed in Appendices B.5–B.7.

Type-aware tool selection. We compare Tool Prompt (CoT w/ Tool) and ToolSciVer by accuracy change over non-tool CoT, tool-execution success, and tool-family distribution.

Figure 2 shows that ToolSciVer learns a more type-aware tool-use policy than Tool Prompt. On table and chart/plot subsets, ToolSciVer routes most calls to the corresponding tool family and obtains larger accuracy gains, with an additional 12.85 points on tables and 10.71 points on charts/plots over Tool Prompt. For multi-panel evidence, ToolSciVer makes `image_zoom_in` the largest call family and improves accuracy by 7.19 points over non-tool CoT, while Tool Prompt splits calls between chart parsing and zooming and reduces accuracy by 0.99 points. Tool-execution success also improves across all three evidence types, reaching 97.39% for tables and 96.51% for charts/plots. Thus, policy training improves both tool-call executability and evidence-aligned tool routing.

Since a successful call may still return irrelevant information, we also report *Relevant Evidence Acquisition Rate* (REAR), the fraction of examples where the model's reasoning trace or tool observations contain the required visual evidence. This metric separates evidence access from the correctness of the final verdict. Table 5 shows that ToolSciVer achieves the highest REAR across

Method	Table	Chart	M-panel	Avg.
CoT w/o Tool	93.00	87.00	69.00	83.00
CoT w/ Tool	87.00	88.00	75.00	83.33
ToolSciVer	**98.00**	**93.00**	**79.00**	**90.00**

Table 5: Relevant Evidence Acquisition Rate (REAR, %).

Model	Table	Chart	M-panel	Overall
Qwen3.5-4B	60.00	75.00	46.00	60.33
ToolSciVer	**71.00**	**84.00**	**60.00**	**71.67**
Δ	+11.00	+9.00	+14.00	+11.33

Table 6: Evidence-conditioned verification accuracy (%).

all evidence types, raising the average from 83.00% for non-tool CoT and 83.33% for Tool Prompt to 90.00%. The largest gain over non-tool CoT appears on multi-panel figures, where evidence is often localized and visually dense, while the table subset suggests that prompt-only tool use can retrieve irrelevant evidence. Together with the qualitative example in Appendix C.5, these results indicate that learned tool use helps the model acquire claim-relevant visual evidence. Despite these gains, REAR remains lowest on multi-panel figures (79.00%), indicating that acquiring localized and cross-panel evidence remains challenging.

Reasoning over acquired evidence. Beyond visual evidence acquisition and type-aware tool selection, we examine whether ToolSciVer reasons more reliably given relevant evidence. To isolate evidence-based reasoning, we use a text-only evidence-conditioned setting where each input contains the claim, original textual context/caption, and the manually annotated gold visual-evidence statement, but excludes images, tool observations, tool schemas, and tool-use instructions. Table 6 shows that ToolSciVer improves overall accuracy from 60.33% to 71.67%, yielding an absolute gain of 11.33 percentage points under the same gold evidence. The gains are consistent across evidence types, with improvements of 11.00 points on tables, 9.00 points on charts, and 14.00 points on multi-panel examples. A paired comparison further supports this improvement: ToolSciVer corrects 59 examples missed by the base model, while the base model corrects 25 examples missed by ToolSciVer, yielding a McNemar exact test result of $p = 2.66 \times 10^{-4}$. Because both models receive the same gold evidence, this gain points to improved evidence use during verification, complementing the acquisition improvements measured by REAR. Even with gold evidence, multi-panel accuracy remains 60.00%, leaving substantial room to improve verification after evidence acquisition.

General chart and figure reasoning. We further evaluate whether the improvement transfers beyond MSCV on ChartQA (Masry et al., 2022), DVQA (Kafle et al., 2018), and FigureQA (Kahou et al., 2018), using 1,000 examples from each benchmark with only the original chart/figure image and question, without gold evidence or tool-use instructions.

Table 7 shows that ToolSciVer consistently improves over the backbone, raising average accuracy from 66.37% to 71.50% with the largest gains on DVQA and FigureQA. A paired comparison over all 3,000 examples confirms that the improvement is statistically significant ($p = 4.66 \times 10^{-16}$), suggesting that our training also strengthens general chart and figure reasoning beyond the MSCV task.

Table 7: Accuracy (%) on external chart and figure benchmarks.

Benchmark	Qwen3.5	ToolSciVer	Δ
ChartQA	63.70	**65.40**	+1.70
DVQA	84.70	**90.10**	+5.40
FigureQA	50.70	**59.00**	+8.30
Avg.	66.37	**71.50**	+5.13

6 Conclusion

We presented ToolSciVer, a framework for multimodal scientific claim verification with small, open VLMs. It combines SciEviKit for tables, charts, and scientific figures with OPSD warm-up from reference tool calls and GRPO, enabling selective tool use without complete reasoning demonstrations. Experiments on SciVer and MuSciClaims across three backbones show consistent improvements over non-tool and prior visual tool-use methods. Further analyses show more efficient tool use, better evidence acquisition, and stronger reasoning over acquired evidence, with gains extending to general chart and figure reasoning. These findings highlight the value of combining specialized evidence tools with learned verification policies to improve scientific claim verification.

AI USE STATEMENT

We used generative AI tools to improve the language and readability of the manuscript and to assist with data annotation, including the generation of SFT tool-use and reasoning trajectories (Appendix B.1). The SFT annotations underwent human review. The authors take responsibility for the final manuscript, data, analyses, and conclusions.

REPRODUCIBILITY STATEMENT

We provide the training and evaluation code for ToolSciVer in the supplementary material. Section 3 and Appendices A.1–A.4 describe the tool interfaces, execution scheduler, and training objectives. Section 4 and Appendix B document the data splits and annotation procedures, training configurations and random seeds, checkpoint selection, evaluation protocols, and baseline prompts. Diagnostic subset construction and metric definitions are detailed in Appendices B.6–B.7.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.13649.

Carlos Alvarez, Maxwell Bennett, and Lucy Lu Wang. Zero-shot scientific claim verification using LLMs and citation text. In *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*, pp. 269–276, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.sdp-1.25/.

Shuai Dong, Yongfu Zhu, Yuqi Xu, Weichu Xie, Liuwenpu, Ziyue Wang, Kaiwen Tuo, Congcong Wang, Siyuan Wang, Wenqi Shao, Shuai Yang, Ji Zhao, Caoyuan Ma, Wenzheng Chang, Taiqiang Wu, Xinlei Yu, Hongrui Wu, Xiaoxuan He, Fangke Chen, Dianyi Wang, Kanghui Tian, Sirry Chen, Xingyu Liu, Xiangnan Wu, Jiawei Guo, Haowen Hou, LingHan Chen, Zhongyu Wei, and Jiaqi Wang. RL starts before RL: On policy distillation for better reinforcement learning. *arXiv preprint arXiv:2609.28145*, 2026a. doi: 10.48550/arXiv.2609.28145. URL https://arxiv.org/abs/2609.28145.

Xuanzhao Dong, Wenhui Zhu, Peijie Qiu, Xiwen Chen, Xiaobing Yu, Xin Li, Zhipeng Wang, Shao Tang, Gen Li, Yujian Xiong, Hao Wang, Yanxi Chen, Prayag Tiwari, and Yalin Wang. Mags-RL: Wearing multimodal LLMs a magnifying glass via agentic reinforcement learning for complex scene reasoning. *arXiv preprint arXiv:2605.27960*, 2026b. doi: 10.48550/arXiv.2605.27960. URL https://arxiv.org/abs/2605.27960.

Clement Farabet and Olivier Lacombe. Gemma 4: Byte for byte, the most capable open models, April 2026. URL https://blog.google/innovation-and-ai/technology/developers-tools/gemma-4/.

Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Richard Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei Florencio, and Cha Zhang. Refocus: Visual editing as a chain of thought for structured image understanding. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 17783–17805. PMLR, 2025. URL https://proceedings.mlr.press/v267/fu25d.html.

Xinyu Geng, Peng Xia, Zhen Zhang, Xinyu Wang, Qiuchen Wang, Ruixue Ding, Chenxi Wang, Jialong Wu, Yida Zhao, Kuan Li, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. Webwatcher: Breaking new frontier of vision-language deep research agent. *arXiv preprint arXiv:2508.05748*, 2025. doi: 10.48550/arXiv.2508.05748. URL https://arxiv.org/abs/2508.05748.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14953–14962, 2023.

Xanh Ho, Yun-Ang Wu, Sunisth Kumar, Florian Boudin, Atsuhiro Takasu, and Akiko Aizawa. Format matters: The robustness of multimodal LLMs in reviewing evidence from tables and charts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31014–31022, 2026a. doi: 10.1609/aaai.v40i37.40361. URL https://arxiv.org/abs/2511.10075.

Xanh Ho, Yun-Ang Wu, Sunisth Kumar, Tian Cheng Xia, Florian Boudin, André Greiner-Petter, and Akiko Aizawa. SciClaimEval: Cross-modal claim verification in scientific papers. In *Proceedings of the 15th Language Resources and Evaluation Conference*, Palma de Mallorca, Spain, 2026b. ELRA Language Resources Association. doi: 10.48550/arXiv.2602.07621. URL https://arxiv.org/abs/2602.07621.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4423.

Kushal Kafle, Brian Price, Scott Cohen, and Christopher Kanan. DVQA: Understanding data visualizations via question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5648–5656, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Kafle_DVQA_Understanding_Data_CVPR_2018_paper.html.

Samira Ebrahimi Kahou, Vincent Michalski, Adam Atkinson, Akos Kadar, Adam Trischler, and Yoshua Bengio. FigureQA: An annotated figure dataset for visual reasoning. In *ICLR 2018 Workshop*, 2018. doi: 10.48550/arXiv.1710.07300. URL https://openreview.net/forum?id=H1mz0OyDz.

Rachneet Kaur, Nishan Srishankar, Zhen Zeng, and Sumitra Ganesh. ChartAgent: A multimodal agent for visually grounded reasoning in complex chart question answering. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18488–18555. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.843. URL https://aclanthology.org/2026.acl-long.843/.

Aditya Kishore, Gaurav Kumar, and Jasabanta Patro. Multicheck: Strengthening web trust with unified multimodal fact verification. *arXiv preprint arXiv:2508.05097*, 2025. doi: 10.48550/arXiv.2508.05097. URL https://arxiv.org/abs/2508.05097.

Yash Kumar Lal, Manikanta Bandham, Mohammad Saqib Hasan, Apoorva Kashi, Mahnaz Koupaee, and Niranjan Balasubramanian. MuSciClaims: Multimodal scientific claim verification. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 3285–3307, Mumbai, India, 2025. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics. doi: 10.18653/v1/2025.ijcnlp-long.175. URL https://aclanthology.org/2025.ijcnlp-long.175/.

Wenze Lin, Jiale Zhao, Xitai Jiang, Songde Rao, Yining Li, Shenzhi Wang, Bingxiang He, and Gao Huang. On-policy distillation with verifiable reward. *arXiv preprint arXiv:2608.24696*, 2026. doi: 10.48550/arXiv.2608.24696. URL https://arxiv.org/abs/2608.24696.

Fangyu Liu, Julian Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. DePlot: One-shot visual language reasoning by plot-to-table translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 10381–10399, Toronto, Canada, 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.660. URL https://aclanthology.org/2023.findings-acl.660/.

Fangyu Liu, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Yasemin Altun, Nigel Collier, and Julian Eisenschlos. MatCha: Enhancing visual language pretraining with math reasoning and chart derendering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12756–12770, Toronto, Canada, 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.714. URL https://aclanthology.org/2023.acl-long.714/.

Xinyuan Lu, Liangming Pan, Qian Liu, Preslav Nakov, and Min-Yen Kan. SCITAB: A challenging benchmark for compositional reasoning and claim verification on scientific tables. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7787–7813, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.483. URL https://aclanthology.org/2023.emnlp-main.483/.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.177. URL https://aclanthology.org/2022.findings-acl.177/.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026. doi: 10.48550/arXiv.2602.04942. URL https://arxiv.org/abs/2602.04942.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. doi: 10.48550/arXiv.2402.03300. URL https://arxiv.org/abs/2402.03300.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, and Yu Cheng. Openthinkimg: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025. doi: 10.48550/arXiv.2505.08617. URL https://arxiv.org/abs/2505.08617.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. FEVER: A large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 809–819, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1074. URL https://aclanthology.org/N18-1074/.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7534–7550, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.609. URL https://aclanthology.org/2020.emnlp-main.609/.

David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Iz Beltagy, Lucy Lu Wang, and Hannaneh Hajishirzi. SciFact-open: Towards open-domain scientific claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4719–4734, Abu Dhabi, United Arab Emirates, 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.347. URL https://aclanthology.org/2022.findings-emnlp.347/.

David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 61–76, Seattle, United States, 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.6. URL https://aclanthology.org/2022.findings-naacl.6/.

Chengye Wang, Yifei Shen, Zexi Kuang, Arman Cohan, and Yilun Zhao. SciVer: Evaluating foundation models for multimodal scientific claim verification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8562–8579, Vienna, Austria, 2025a. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.420. URL https://aclanthology.org/2025.acl-long.420/.

Haoran Wang, Aman Rangapur, Xiongxiao Xu, Yueqing Liang, Haroon Gharwi, Carl Yang, and Kai Shu. Piecing it all together: Verifying multi-hop multimodal claims. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 7453–7469, Abu Dhabi, UAE, 2025b. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.498/.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently. *arXiv preprint arXiv:2504.14870*, 2025c. doi: 10.48550/arXiv.2504.14870. URL https://arxiv.org/abs/2504.14870.

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*, 2025d. doi: 10.48550/arXiv.2508.18265. URL https://arxiv.org/abs/2508.18265.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Mingyuan Wu, Jingcheng Yang, Jize Jiang, Meitang Li, Kaizhuo Yan, Hanchao Yu, Minjia Zhang, Chengxiang Zhai, and Klara Nahrstedt. Vtool-r1: Vlms learn to think with images via reinforcement learning on multimodal tool use. *arXiv preprint arXiv:2505.19255*, 2025. doi: 10.48550/arXiv.2505.19255. URL https://arxiv.org/abs/2505.19255.

Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. MM-REACT: Prompting ChatGPT for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023. doi: 10.48550/arXiv.2303.11381. URL https://arxiv.org/abs/2303.11381.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Qiying Yu, Zheng Zhang, Ruofei Zhu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. doi: 10.48550/arXiv.2503.14476. URL https://arxiv.org/abs/2503.14476.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models, 2026. URL https://arxiv.org/abs/2601.18734.

Mingyang Zhou, Lingyu Zhang, Sophia Horng, Maximillian Chen, Kung-Hsiang Huang, and Shih-Fu Chang. M^2-TabFact: Multi-document multi-modal fact verification with visual and textual representations of tabular data. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 26239–26256, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1345. URL https://aclanthology.org/2025.findings-acl.1345/.

Dawei Zhu, Rui Meng, Jiefeng Chen, Sujian Li, Tomas Pfister, and Jinsung Yoon. DocLens: A tool-augmented multi-agent framework for long visual document understanding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26804–26829. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1234. URL https://aclanthology.org/2026.acl-long.1234/.

A Method Details

A.1 SciEviKit Interfaces

Each tool call targets a single image through the argument `image_path`, such as `img1`, `img2`, or a crop returned by a previous zoom operation. The scheduler checks that each assistant turn contains either exactly one valid tool call or a final answer. Malformed calls, unsupported tool names, illegal image references, missing or invalid arguments, and execution failures are rejected and recorded as negative tool-call events.

Table Focus. For table images, we first convert the original image into a structured CSV-like representation using OCR. We then expose two focused tools:

`focus_row(image_path, row_number)`

and

`focus_column(image_path, column_number).`

The row tool returns the content of the selected row, and the column tool returns the content of the selected column. This design is intended to expose localized tabular evidence, such as a specific method row, dataset row, metric column, or numerical entry, without returning the entire table.

Chart Parse. For chart and plot images, we expose:

`parse_content(image_path).`

As described in Section 3, Chart Parse uses a stronger VLM to convert the selected chart into structured textual observations. In our experiments, this extraction backend is GPT-6-Astra with reasoning effort set to `max`. The tool returns parsed chart content as structured text when possible. For example, the returned observation may contain chart elements, axis labels, series names, and associated values in a JSON-like format. When strict structural parsing is unavailable, the tool falls back to raw extracted content rather than failing silently. This allows the verifier to compare a claim against explicit chart evidence such as trends, thresholds, relative ordering, or numerical values.

Region Zoom. For general scientific figures, diagrams, dense images, or multi-panel figures, we expose:

`image_zoom_in(image_path,` x_1`,` y_1`,` x_2`,` y_2`).`

The tool crops and enlarges the selected region and returns the zoomed-in image. The bounding box uses normalized coordinates on a $[0, 1000]$ scale with respect to the selected image. This makes the interface resolution-invariant and allows the model to inspect small local regions that may contain decisive visual evidence.

A.2 Tool Scheduler and Execution Logs

Both training stages of ToolSciVer interact with tools from SciEviKit through the same scheduler. The scheduler parses model outputs, validates tool-call syntax, executes valid calls, and returns either a tool observation or an execution error. Each assistant turn is constrained to contain either one valid tool call or the final answer. If a turn contains multiple tool calls, an unsupported tool name, missing arguments, invalid argument types, coordinates outside the valid range, or an unparsable call format, the scheduler records it as a malformed tool turn. The scheduler enforces a configured budget of model tool-call turns; valid, malformed, invalid-argument, and failed attempts all consume this budget. An attempt beyond the budget is not executed, receives a negative tool-call event, and is followed by a final-answer-only turn. Both training stages allow the policy to answer directly without a tool call.

The scheduler records the following metadata for each rollout: tool name, image slot, tool arguments, format and argument validity, execution outcome, tool/type compatibility, per-attempt score, number of tool calls, response length, and final-answer validity. These logs are used for both training and analysis. During GRPO, the per-attempt scores are aggregated into the trajectory-level tool-call reward. During OPSD, observations enter the shared teacher/student history, but event scores do not enter the distillation objective. During evaluation, the logs are used to compute tool success rate, tool-family routing statistics, tool calls per example, and tool calls per correct example.

A.3 OPSD Warm-up Details

The warm-up examples and reference tool-call annotations are described in Appendix B.1.

Student and teacher views. The student starts with the original claim, context, and source images. It generates each assistant turn and receives the resulting online tool feedback through the scheduler. Before generation, reference labels, solutions, privileged prompts, and reference-trajectory metadata are removed from the student request. To construct the teacher view, we retain the student's complete sampled interaction and augment only the original task message with the privileged action sequence. All later assistant turns, tool observations, and crop images are identical in the two views. In particular, the reference action sequence does not replace the student's actions or determine which tools the environment executes. Any failed student call remains part of the history seen by both models.

Teacher parameters and gradient flow. The student is initialized from the base VLM π_0 and updated with LoRA (Hu et al., 2022) on the language-model linear layers; the vision encoder and multimodal connector remain frozen. The teacher uses the same frozen base parameters, implemented by disabling the student adapters during teacher evaluation. These base parameters stay fixed throughout warm-up, and teacher outputs are computed without gradients. The student performs fully on-policy sampling, with no mixture of reference-response training and no additional supervised cross-entropy term.

Masked distribution matching. Let $\mathcal{W}$ be the token vocabulary, $\mathcal{A}(\tau)$ the assistant-token positions in a sampled trajectory, and h_ℓ the original task and all preceding online tokens and images at position ℓ. The reference calls z enter only the teacher distribution. At distillation temperature T_d, define

$$p_\ell^T(v) = \mathrm{softmax}(f_0(h_\ell, z)/T_d)_v\,, \qquad p_\ell^S(v) = \mathrm{softmax}(f_\theta(h_\ell)/T_d)_v\,, \quad v \in \mathcal{W},$$

where f_0 and f_θ denote teacher and student logits. Following the pointwise clipping used by OPSD (Zhao et al., 2026), the forward-KL objective in Eq. 1 is

$$\mathcal{D}_\delta(p_\ell^T, p_\ell^S) = \sum_{v\in\mathcal{W}} \min\left\{ p_\ell^T(v) \log \frac{p_\ell^T(v)}{p_\ell^S(v)}, \delta \right\}. \tag{4}$$

Clipping is applied to each vocabulary contribution before summation, rather than to the summed token KL. We use $T_d = 1$ and $\delta = 0.05$. Removing the clipping recovers $D_{\mathrm{KL}}(p_\ell^T \| p_\ell^S)$; with clipping, the aggregate is not constrained to be nonnegative. The loss is averaged over assistant-token positions, including intermediate reasoning, tool calls, and the final response. System/user messages and environment observations are masked from the loss but remain available as context. Separate masks account for the different prompt lengths, and the implementation verifies that the two views have exactly the same supervised token IDs in the same order. Sampled histories and teacher probabilities are held fixed when taking the student gradient.

Transition to GRPO. After warm-up, we merge the learned adapters into the base model to obtain π_{warm}. This model initializes the GRPO policy and supplies a frozen reference $\pi_{\mathrm{ref}} = \pi_{\mathrm{warm}}$ for the second-stage KL penalty. Teacher-only reference actions are discarded; GRPO rollouts and inference use the original task and their own online observations. The OPSD action references and the per-image annotations used by the GRPO reward serve different purposes: the former provide task-specific reference actions to the teacher, whereas the latter only assess tool/type compatibility.

A.4 GRPO Objective and Reward Details

GRPO objective. For each input instance x, we sample a group of G rollouts

$$\mathcal{G}(x) = \{\tau_i\}_{i=1}^{G}$$

from the rollout policy π_{old}, initialized from π_{warm}. Each rollout may contain zero or more tool calls before the final answer. After assigning rollout-level rewards R_i, we compute the group-relative advantage:

$$A_i = \frac{R_i - \text{mean}_{j=1}^{G} R_j}{\text{std}_{j=1}^{G} R_j + \epsilon_A},$$

where $\epsilon_A > 0$ is a small numerical stabilizer; equal-reward groups have zero advantages. Let $a_{i,\ell}$ be the sampled token at assistant position ℓ of trajectory τ_i, and define

$$\rho_{i,\ell}(\theta) = \frac{\pi_\theta(a_{i,\ell} \mid h_{i,\ell})}{\pi_{\text{old}}(a_{i,\ell} \mid h_{i,\ell})}, \qquad d_{i,\ell}(\theta) = D_{\text{KL}}(\pi_\theta(\cdot \mid h_{i,\ell}) \| \, \pi_{\text{ref}}(\cdot \mid h_{i,\ell})) \, .$$

The policy maximizes the clipped GRPO objective (Shao et al., 2024)

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x,\, \{\tau_i\} \sim \pi_{\text{old}}} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|\mathcal{A}(\tau_i)|} \sum_{\ell \in \mathcal{A}(\tau_i)} g_{i,\ell}(\theta) \right], \tag{5}$$

$$g_{i,\ell}(\theta) = \min(\rho_{i,\ell}(\theta) A_i, \, \text{clip}(\rho_{i,\ell}(\theta), 1-\epsilon, 1+\epsilon) A_i) - \beta d_{i,\ell}(\theta).$$

Here, ϵ is the policy-ratio clipping threshold and β controls the KL penalty to the fixed reference $\pi_{\text{ref}} = \pi_{\text{warm}}$. The trajectory advantage is shared across its assistant-generated tokens. Tool observations are excluded from the optimized token positions while remaining in the conditioning histories.

Overall reward. For a rollout τ, the final reward is

$$R(\tau) = s_{\text{eff}}(\tau) \, (r_{\text{ans}}(\tau) + \lambda_{\text{fmt}} r_{\text{fmt}}(\tau)) + \lambda_{\text{len}} r_{\text{len}}(\tau) + \lambda_{\text{tool}} r_{\text{tool}}(\tau).$$

The default auxiliary reward coefficients are

$$\lambda_{\text{fmt}} = \lambda_{\text{len}} = \lambda_{\text{tool}} = 0.1.$$

The answer and format rewards form the task-critical part of the objective, while the length term discourages overlong generations and the tool-call term rewards valid type-compatible interactions while penalizing incorrect attempts. The coefficient s_{eff} encourages efficient tool use relative to the best correct trajectories in the sampled GRPO group.

Answer reward. The answer reward measures final task correctness:

$$r_{\text{ans}}(\tau) = \mathbb{I}[\hat{y} = y],$$

where y is the ground-truth label and $\hat{y}$ is the normalized model prediction.

Format reward. The format reward encourages the model to produce a parseable final response. We define

$$r_{\text{fmt}}(\tau) = 0.5 \, I_{\text{reason}}(\tau) + 0.5 \, I_{\text{answer}}(\tau),$$

where $I_{\text{reason}}(\tau)$ indicates whether the final response contains the required reasoning segment under the prompt template, and $I_{\text{answer}}(\tau)$ indicates whether the response contains exactly one valid normalized final-answer marker. This term stabilizes training and ensures that final predictions can be reliably extracted.

Tool-efficiency coefficient. To encourage selective rather than excessive tool use, we adapt the OTC tool-efficiency coefficient (Wang et al., 2025c). Let $m(\tau)$ be the number of tool calls in rollout τ. Let $\mathcal{C}(x)$ be the set of strict-correct rollouts in the GRPO group, where a rollout is strict-correct if it predicts the correct label and ends with a valid final answer. If $\mathcal{C}(x)$ is non-empty, we estimate the group-local optimal tool budget as

$$n^\star(x) = \min_{\tau_i \in \mathcal{C}(x)} m(\tau_i).$$

If no strict-correct rollout exists, we set $s_{\text{eff}}(\tau) = 1$. Otherwise, with the OTC scaling parameter c_{OTC}, we define

$$s_{\text{eff}}(\tau) = \begin{cases} 1, & n^\star = 0,\ m = 0, \\ \cos\left(\frac{m\pi}{2m+c_{\text{OTC}}}\right), & n^\star = 0,\ m > 0, \\ 0, & n^\star > 0,\ m = 0, \\ \sin\left(\frac{m\pi}{m+n^\star}\right), & n^\star > 0,\ m > 0. \end{cases}$$

This coefficient rewards correct trajectories that use close to the group-local minimal number of tool calls, while penalizing unnecessary tool use when a direct answer is sufficient. The default coefficient uses $c_{\text{OTC}} = 5$. This reward-scaling parameter is distinct from the scheduler's interaction budget: the experiments allow at most three model tool-call rounds, as specified in Appendix B.2.

Length reward. We use the soft overlong penalty from DAPO (Yu et al., 2025) to discourage unnecessarily long generations. Let $L(\tau)$ denote the total number of assistant tokens across the trajectory, let $L_{\max}$ be the maximum completion length, and let C denote the soft-cache length. We define the soft threshold as

$$L_{\text{soft}} = L_{\max} - C.$$

The length reward is

$$r_{\text{len}}(\tau) = \text{clip}_{[-1,0]}\left(-\frac{\max(L(\tau) - L_{\text{soft}}, 0)}{C}\right).$$

Thus, the penalty is zero below the soft threshold and decreases linearly as the response approaches the hard length limit. The default values are $C = 768$ and $L_{\max} = 4096$; the backbone-specific completion-length setting is given in Appendix B.3.

Tool-call reward. For the j-th audited tool attempt, we assign

$$q_j = \begin{cases} +1, & \text{allowed tool, legal source image and arguments, successful execution,} \\ -1, & \text{wrong tool/image, malformed or invalid call, failed execution, or excess turn.} \end{cases}$$

The positive score requires all four conditions simultaneously. A positive event establishes execution and type compatibility, not that the returned evidence alone entails or refutes the claim. If one response contains multiple tool tags, the turn consumes one model-call round and each attempted tag contributes a separate $q_j = -1$ event.

For K audited attempts, the trajectory-level reward is

$$r_{\text{tool}}(\tau) = \begin{cases} 0, & K = 0, \\ \frac{1}{K}\sum_{j=1}^{K} q_j, & K > 0. \end{cases}$$

Hence $r_{\text{tool}} \in [-1, 1]$, no-call trajectories remain neutral, and averaging prevents reward from growing merely by increasing the number of calls.

Tool-type compatibility. Each of the 4,014 source-image slots in the 3,010-example training set has manually double-reviewed `visual_types` and `allowed_tools` fields. The mapping is: table → `focus_row`/`focus_column`; chart → `parse_content`; table–chart composite → the union of those structured tools; and other or multi-panel visual → `image_zoom_in`. For a multi-image example, the selected legal `imgN` slot is judged against its own annotations. These annotations contain no claim, gold label, solution, or answer and do not indicate that a tool is required; they only define which tools are valid if the policy elects to call one.

B Experiment Details

B.1 Training and Evaluation Data

Table 8 and Table 9 summarize the exact split sizes used in our experiments. GRPO uses a mixed training set with 3,010 examples in total. Following Section 4, both datasets are evaluated in the

binary verification setting. The evaluation reported in the main text uses 1,505 instances in total: 1,000 from the SCIVER evaluation split and 505 from the MUSCICLAIMS test split. The refusal exclusions for the Opus 5.5 reference are detailed in Appendix B.4.

These counts are also the reason we report *sample-weighted* overall accuracy in the main results table. SCIVER is nearly balanced across its four reasoning categories, whereas the MUSCICLAIMS evaluation split is not balanced across biology, physics, and chemistry. A simple macro average would therefore over-weight the smaller physics and chemistry subsets.

Shared warm-up examples. OPSD and SFT warm-up use the same 200 examples sampled from the MSCV training set, with the number drawn from each category proportional to its size in the training set.

OPSD annotations. For each selected claim, we manually annotate the complete tool calls needed for verification, taking the types of the associated images into account. Each annotation records an ordered sequence of concrete calls, including the tool name, target source image, and full arguments. We serialize these annotated calls as teacher-only privileged information z, using `<no_tool_call/>` when no call is needed. The annotations contain no reference reasoning, final answers, tool observations, or derived crops.

SFT annotations. For the same examples, complete tool-use and reasoning trajectories are annotated by GPT-6-Astra with reasoning effort set to `max`, followed by a second round of human review.

Table 8: Category distribution of the mixed training set used for the GRPO runs.

Split	Category	# Examples
SCIVER train mixture	Analytic	494
	Direct	502
	Parallel	505
	Sequential	499
	Subtotal	2,000
MUSCICLAIMS train mixture	Biology	612
	Physics	192
	Chemistry	206
	Subtotal	1,010
Total training mixture	–	**3,010**

Table 9: Category distribution of the evaluation splits used in the main results. The "Overall" values use these sample counts as weights, except for the Opus 5.5 reference, whose weights are recomputed after the refusal exclusions described in Appendix B.4.

Split	Category	# Examples
SCIVER evaluation	Analytic	256
	Direct	248
	Parallel	245
	Sequential	251
	Subtotal	1,000
MUSCICLAIMS test	Biology	306
	Physics	96
	Chemistry	103
	Subtotal	505
Total evaluation set	–	**1,505**

B.2 Training and Evaluation Protocol

For each trainable method, we fine-tune the corresponding backbone on the mixed MSCV training set. For methods equipped with SciEviKit, the model may either produce a final answer directly or issue tool calls through the scheduler during training and evaluation. Each assistant turn contains at most one tool call, and the model tool-call budget is capped at $c_{\max} = 3$. The scheduler validates tool-call syntax, executes valid calls, returns tool observations or execution errors, and records tool-use metadata for later analysis.

All models are evaluated in a closed-context setting using only the benchmark-provided claim, visual input, caption, and textual context. We do not allow open-web retrieval or external corpora at evaluation time. Free-form generations are normalized to the benchmark label space with rule-based answer extraction, and accuracy is the primary metric. For diagnostic analyses, we additionally report Relevant Evidence Acquisition Rate (REAR), type-aware tool routing statistics, tool calls per example, tool calls per correct example, and response length; metric definitions are provided in Appendix B.7.

B.3 Training Configuration

Table 10 summarizes the default training and rollout configuration for the GRPO-trained methods. Training runs for one epoch over the 3,010-example mixed MSCV set, corresponding to at most 1,505 GRPO optimization steps.

Table 10: Default training and rollout configuration for GRPO-trained methods.

Setting	**Default value**
Training duration	1 epoch; at most 1,505 GRPO steps
Optimizer	AdamW
Learning-rate schedule	1×10^{-6}; cosine decay; 3 warmup steps
Weight decay	0.1
GRPO group size	8 rollouts per prompt
KL coefficient β	0.005
Clipping ratio ϵ	0.2
Per-GPU batch size	1
Gradient accumulation	8 steps
Maximum completion length	4,096 tokens
Rollout sampling	Temperature 1.0; top-p 0.99
Tool-interaction budget	At most 3 model tool-call rounds
Numerical precision	bfloat16
Distributed/memory settings	DeepSpeed ZeRO-3; gradient checkpointing
Canonical random seed	42

For Gemma4-E4B, memory constraints require a GRPO group size of 4, gradient accumulation of 4, and a 2,048-token maximum completion length; all other settings remain unchanged. Each trainable method uses a single fixed training seed of 42. Following Section 4, checkpoints are evaluated every 50 GRPO steps, and the checkpoint with the highest average evaluation accuracy is selected for each trainable method. Open-source scores are reported as mean $\pm$ sample standard deviation ($\mathrm{ddof} = 1$) over three evaluations with different inference seeds. For each method, the model checkpoint, evaluation examples, prompt, decoding parameters (including temperature), and answer-extraction rule remain fixed across these evaluations; only the inference seed changes. These statistics describe inference-seed variability for a fixed model and are not estimates across independently trained models. The reported warm-up checkpoints and inference seeds for Table 3 are specified in Appendix C.1.

B.4 Baseline and Prompting Details

Training-free evaluation and answer normalization. The training-free comparisons in Tables 2 and 3 use the 1,000-example SciVer and 505-example MuSciClaims evaluation sets. Generated

responses are mapped to binary predictions, and empty responses or runtime failures count as incorrect. CoT responses are normalized to `yes`/`no` with a fixed rule-based answer extractor, while CHARTAGENT and DOCLENS use their native binary-answer parsers. For each method, the same answer-extraction rule and decoding settings are used in all three evaluations, which differ only in inference seed, as specified in Appendix B.3.

Non-tool CoT inference. The non-tool CoT baseline receives the claim, context, caption, and image, but has no access to external visual tools. It is prompted to reason step by step and then produce a normalized yes/no answer. The prompt template is:

```
Claim: $claim
Context: $context
Caption: $caption

Your task is to critically evaluate the claim based on the image
and the caption. Carefully examine whether the information in the
caption truly supports the claim. Be skeptical and cautious: if
there is any inconsistency, missing evidence, or ambiguity,
consider the claim incorrect.

Start by explaining your reasoning process clearly, focusing on
identifying potential contradictions, lack of support, or misleading
interpretations. If the claim is unsupported or contradicted by the
caption and image, respond with 'no'. Only respond with 'yes' if the
claim is fully and clearly supported.

Conclude your analysis by stating: 'Therefore, the final answer is:
Answer: $$ANSWER', where $$ANSWER is your final answer. Think step by
step before answering.
```

Prompt-only tool-use CoT inference. The prompt-only tool-use baseline receives the same MSCV input and the same SciEviKit APIs as our method. However, the model is not trained with tool-use trajectories or reward feedback. It must decide from prompting alone whether a tool is needed, which tool to call, and what arguments to provide. This baseline isolates the effect of exposing tools without learning a tool-use policy.

No-tool GRPO. This baseline is initialized from the same backbone and fine-tuned on the same mixed MSCV training set with the same GRPO optimization framework as our method. It receives the non-tool input and cannot invoke any visual tool. The comparison therefore isolates the benefit of task-specific RL adaptation from the additional benefit of claim-conditioned visual evidence acquisition.

Training-free agentic baselines. We adapt CHARTAGENT (Kaur et al., 2026) and DOCLENS (Zhu et al., 2026) as training-free agentic scaffolds around each evaluated backbone, without parameter updates on MSCV. CHARTAGENT provides chart-specific visual interaction and localized chart reasoning, whereas DOCLENS provides hierarchical visual-evidence navigation and region-level inspection. Each method receives the same benchmark-provided claim, images, captions, and textual context as the other baselines; no web retrieval or external document collection is available. When DOCLENS expects a document collection, the benchmark-provided images form the complete closed collection for that example.

Dataset-adapted trained tool-use baselines. For the VTOOL-R1 and OPENTHINKIMG baselines, we retain their original tool suites and reward designs. We train each baseline on the same MSCV training splits used by our method. This comparison tests whether general visual tool-use training recipes are sufficient for MSCV, or whether task-aligned evidence extraction tools are needed.

Proprietary model references. All proprietary VLMs are evaluated under non-tool CoT; GPT-5.6 Sol and Claude Opus 5.5 are additionally evaluated with access to SciEviKit. They are included

as external references rather than controlled training baselines, since their training data, model parameters, and tool-use training procedures are not available. GPT-5.6 Sol, Claude Opus 5, and Claude Opus 5.5 use native reasoning at `medium` effort with an 8,192-token combined reasoning-and-answer cap; their single-evaluation scores use original labels and tail-50 answer parsing.[2] With SciEviKit, the 8,192-token cap covers all generated tokens across the entire trajectory, including reasoning and tool arguments. Each example allows at most three native tool-call rounds, with at most one call per round, followed by a final answer with tools disabled; native reasoning history is preserved across rounds. The tool-enabled evaluations use the same SciEviKit backend as the open-source tool-use evaluations and are submitted through Batch APIs. For Opus 5.5 under non-tool CoT, responses with API `stop_reason=refusal` are excluded from both subset and overall accuracy: 4/1,000 SCIVER and 158/505 MUSCICLAIMS examples. All 162 refusals have the API safety category `bio`. Its non-tool overall accuracies are therefore computed over 996 and 347 non-refused examples, respectively, with subset weights recomputed using the retained sample counts. For its tool-enabled evaluation, these 162 prior refusals are excluded from the evaluation population, and three additional MUSCICLAIMS biology examples trigger explicit API refusals and are also excluded from accuracy. The retained denominators are consequently 996 for SCIVER and 344 for MUSCICLAIMS, including 148 biology, 96 physics, and 100 chemistry examples. GPT-5.6 Sol retains all 1,000 SCIVER and 505 MUSCICLAIMS examples in both settings; Opus 5.5 scores therefore use different retained populations.

B.5 GOLD-EVIDENCE ANNOTATION AND QUALITY CONTROL

The REAR and evidence-conditioned diagnostics require a reference description of the visual evidence needed to verify each claim. We manually annotate these references from the original scientific visuals, claims, captions, and textual context. For each sampled example, we record the claim-relevant visual facts, such as a table entry and its row or column label, a chart series and trend, or a relation between panels. We check that the annotated elements and values are grounded in the original visual and sufficient to assess the claim, retaining the benchmark's original claim and label. The resulting gold-evidence statements are used to score evidence acquisition and are supplied identically to the base model and ToolSciVer in the evidence-conditioned reasoning experiment. Diagnostic sample selection and reference annotation are independent of the evaluated models' predictions and tool traces. The category-level candidate pools and stratified sampling procedure are described in Appendix B.6.

B.6 DIAGNOSTIC SUBSET CONSTRUCTION

We construct diagnostic subsets to analyze whether the model acquires the correct type of visual evidence. These subsets are used for the REAR, tool-selection, and evidence-conditioned reasoning analyses in Section 5.3. We sample 100 examples for each of three evidence types—tables, charts/plots, and multi-panel figures—for a total of 300 examples. Within each evidence type, category-proportional stratified sampling approximately preserves the category distribution of its qualifying pool, using random seed 5169. The strata are reasoning categories for SCIVER and scientific domains for MUSCICLAIMS. The qualifying pools, sampling strata, and resulting coverage are summarized in Table 11.

Table subset. The table subset is drawn from a qualifying pool of 225 SCIVER examples whose evidence images contain table indicators in the file name or metadata. We sample 100 examples in proportion to the reasoning-category frequencies in this pool. These examples are used to evaluate whether the model can acquire localized tabular evidence, such as a target row, column, or cell value.

Chart/plot subset. The chart/plot subset is drawn from a qualifying pool of 265 SCIVER examples for which chart pre-computation metadata indicates that chart parsing is applicable. We sample 100 examples in proportion to the reasoning-category frequencies in this pool. These examples test whether the model can acquire chart evidence such as trends, thresholds, relative ordering, or numerical values.

[2] Seven GPT-5.6 Sol non-tool pilot requests retain an earlier black-background image rendering; Claude Opus 5 and Opus 5.5, and both models' tool-enabled evaluations, use the corrected white-background rendering for those inputs.

Multi-panel subset. The multi-panel subset is drawn from the complete 505-example MUSCICLAIMS evaluation split, whose examples are centered on figure-centric scientific claims. We sample 100 examples in proportion to the frequencies of biology, physics, and chemistry examples in this split. This subset evaluates whether the model can acquire local and cross-panel evidence from dense, diagrammatic, or multi-panel scientific figures.

Table 11: Diagnostic sampling protocol. All samples are drawn with random seed 5169.

Evidence type	Qualifying pool	Category-proportional sample	Coverage
Table	225 SCIVER	100 examples, proportional to reasoning-category frequencies	$100/225 = 44.44\%$
Chart/plot	265 SCIVER	100 examples, proportional to reasoning-category frequencies	$100/265 = 37.74\%$
Multi-panel	505 MUSCICLAIMS	100 examples, proportional to scientific-domain frequencies	$100/505 = 19.80\%$

The same sampled instances are evaluated by every method, and selection is completed without reference to any evaluated model's predictions or tool traces.

B.7 EVALUATION METRICS FOR DIAGNOSTIC ANALYSES

In addition to final verification accuracy, we compute diagnostic metrics for the analyses in Section 5. These metrics are designed to test whether the model improves because it acquires claim-relevant visual evidence, selects tools according to the visual evidence type, and solves the task efficiently.

Sample-weighted overall accuracy. For each benchmark, the overall accuracy is computed by weighting each subset by its number of test examples:

$$\mathrm{Acc}_{\text{overall}} = \frac{\sum_k n_k \,\mathrm{Acc}_k}{\sum_k n_k},$$

where k indexes benchmark subsets, n_k is the number of examples in subset k, and Acc_k is the accuracy on that subset.

Relevant Evidence Acquisition Rate. Following Section 5.3, Relevant Evidence Acquisition Rate (REAR) measures whether the model's reasoning trace or returned tool observations contain the visual evidence required to verify the claim. For each example, the available reasoning and observations are checked against the manually annotated gold-evidence statement described in Appendix B.5. The same evidence criterion applies to non-tool CoT, prompt-only tool use, and ToolSciVer; a direct answer can satisfy it when the reasoning explicitly identifies the required visual facts. Successful tool execution alone is insufficient, and evidence acquisition is scored separately from final-answer correctness. Formally, for a diagnostic subset $\mathcal{D}$,

$$\mathrm{REAR} = \frac{1}{|\mathcal{D}|} \sum_{x_i \in \mathcal{D}} \mathbb{I}[\text{claim-relevant visual evidence is acquired or identified}].$$

We report REAR separately for table, chart/plot, and multi-panel evidence subsets.

Evidence-conditioned verification. For Table 6, both Qwen3.5-4B and ToolSciVer receive the claim, original textual context/caption, and the same manually annotated gold visual-evidence statement. Images, tool observations, tool schemas, and tool-use instructions are excluded. Accuracy is computed on the same 300 diagnostic examples, and the paired comparison uses the McNemar exact test reported in Section 5.3. This setting measures verification given relevant evidence, complementing the evidence-acquisition analysis.

General chart and figure reasoning. The external evaluation in Table 7 uses 1,000 examples each from ChartQA, DVQA, and FigureQA. Each model receives only the original chart or figure image and its question, without gold-evidence annotations or tool-use instructions. We report accuracy for each benchmark and the average across the three equally sized evaluation sets; the main text also reports a paired comparison over all 3,000 examples.

Tool success rate. Tool success rate measures the fraction of attempted tool calls that execute successfully:

$$\text{ToolSucc} = \frac{\#\text{ successfully executed tool calls}}{\#\text{ attempted tool calls}}.$$

A low tool success rate indicates malformed calls, invalid arguments, unsupported tool names, or execution failures.

Tool-family routing share. For a tool family f, the routing share is

$$\rho_f = \frac{\#\text{ calls to tool family } f}{\#\text{ all valid tool calls}}.$$

In the main analysis, we aggregate `focus_row` and `focus_column` into the table-tool family, `parse_content` into the chart-tool family, and `image_zoom_in` into the zoom-tool family.

Tool-use efficiency. We measure tool-use efficiency using the average number of tool calls per example and the average number of tool calls per correct example:

$$\text{Calls/Ex} = \frac{1}{N}\sum_{i=1}^{N} m(\tau_i),$$

$$\text{Calls/Correct} = \frac{\sum_{i=1}^{N} m(\tau_i)\,\mathbb{I}[\hat{y}_i = y_i]}{\sum_{i=1}^{N} \mathbb{I}[\hat{y}_i = y_i]}.$$

We also report average response length in tokens. Together, these metrics test whether the learned policy solves MSCV with fewer unnecessary actions and shorter responses while preserving accuracy.

C Additional Results and Analysis

C.1 Warm-up Comparison Details

Table 3 compares Qwen3.5-4B with no warm-up, SFT warm-up, and OPSD warm-up before tool-enabled GRPO. As discussed in Section 5.2, SFT achieves slightly higher accuracy, while OPSD requires only reference tool-call annotations and retains competitive verification performance. Both warm-up methods use the same 200 training examples; their supervision differs as described in Appendix B.1.

Reported checkpoints and evaluation settings. The warm-up comparison follows the checkpoint-selection and repeated-evaluation protocol in Appendix B.3. The SFT+GRPO row uses SFT checkpoint 60 followed by GRPO checkpoint 350. ToolSciVer uses OPSD checkpoint 15 followed by GRPO checkpoint 150, while the tool-enabled GRPO run without warm-up uses checkpoint 1400. For each reported configuration, the three evaluations use inference seeds 42, 14561, and 56782 while keeping the checkpoint, decoding parameters (including temperature), and answer-extraction rule fixed. The mean and sample standard deviation therefore summarize inference-seed variability, not independent training runs. Training configurations and update budgets differ across the compared training variants.

The comparison supports the annotation-requirement tradeoff discussed in the main text; it does not quantify data-construction cost or total training cost.

C.2 Additional Backbone Ablations

Table 12 expands the tool-use and warm-up comparison in Section 5.2 with complete subset results for all three backbones. Prompt-only tool access is unstable: on SciVer, CoT w/ Tool underperforms CoT w/o Tool for two of three backbones, showing that simply exposing tools does not reliably improve verification. The reported OPSD warm-up variants have higher overall scores than the corresponding tool-enabled GRPO variants without warm-up on both benchmarks for all three backbones.

Table 12: Full subset results for tool-use and warm-up variants across three backbones. Scores retain the reported mean ± standard deviation. GRPO w/ Tool denotes training without warm-up; ToolSciVer uses OPSD warm-up.

	SciVer					MuSciClaims			
Method	**Analytic**	**Direct**	**Parallel**	**Sequential**	**Overall**	**Bio**	**Phy**	**Chem**	**Overall**
				`Qwen3.5-4B`					
CoT w/o Tool	72.92 ± 0.90	75.00 ± 1.40	68.57 ± 0.41	71.32 ± 5.22	71.97 ± 1.76	65.58 ± 4.78	55.55 ± 4.21	71.52 ± 2.44	64.88 ± 3.61
CoT w/ Tool	76.56 ± 5.43	75.81 ± 3.45	67.62 ± 2.09	72.38 ± 2.99	73.13 ± 2.67	72.99 ± 2.18	66.32 ± 3.94	72.81 ± 2.57	71.68 ± 1.69
GRPO w/o Tool	80.73 ± 1.26	83.33 ± 2.22	78.91 ± 2.95	79.28 ± 1.74	80.57 ± 0.74	80.28 ± 0.82	69.10 ± 1.59	78.96 ± 1.48	77.89 ± 0.94
GRPO w/ Tool	85.55 ± 1.41	81.99 ± 1.23	79.05 ± 1.70	82.87 ± 1.20	82.40 ± 1.08	78.54 ± 2.97	67.71 ± 3.13	82.20 ± 3.68	77.23 ± 2.23
ToolSciVer	88.02 ± 1.58	90.32 ± 2.64	76.46 ± 1.18	81.27 ± 2.39	84.07 ± 1.27	80.72 ± 1.18	81.25 ± 2.76	80.58 ± 0.97	80.79 ± 0.52
				`InternVL3.5-4B`					
CoT w/o Tool	71.91 ± 2.29	68.96 ± 3.04	62.89 ± 0.76	64.61 ± 3.52	67.12 ± 0.68	70.91 ± 0.98	56.95 ± 3.66	69.26 ± 6.32	67.92 ± 2.33
CoT w/ Tool	70.18 ± 1.76	66.40 ± 3.23	60.14 ± 0.47	66.00 ± 0.92	65.73 ± 0.42	68.52 ± 0.82	64.58 ± 2.76	74.76 ± 3.50	69.04 ± 1.49
GRPO w/o Tool	75.65 ± 2.15	72.72 ± 1.01	66.94 ± 1.41	67.20 ± 3.62	70.67 ± 1.53	75.27 ± 2.18	63.89 ± 2.62	69.58 ± 2.97	71.95 ± 1.59
GRPO w/ Tool	75.00 ± 0.68	70.16 ± 3.04	67.35 ± 3.19	70.65 ± 2.53	70.83 ± 0.42	72.11 ± 1.05	67.71 ± 4.17	76.70 ± 1.68	72.21 ± 1.73
ToolSciVer	78.39 ± 1.80	69.09 ± 2.68	69.93 ± 3.32	73.71 ± 1.38	72.83 ± 1.16	74.95 ± 2.62	65.62 ± 4.77	77.99 ± 3.12	73.80 ± 1.77
				`Gemma4-E4B`					
CoT w/o Tool	78.78 ± 0.81	75.27 ± 1.91	71.02 ± 1.78	72.91 ± 1.38	74.53 ± 0.57	74.18 ± 1.82	67.01 ± 2.17	73.46 ± 3.68	72.67 ± 0.86
CoT w/ Tool	73.51 ± 7.04	75.34 ± 2.94	67.14 ± 2.52	69.59 ± 6.93	71.42 ± 4.82	68.45 ± 0.42	68.86 ± 3.94	71.65 ± 5.39	69.18 ± 0.45
GRPO w/o Tool	76.17 ± 1.70	71.64 ± 1.30	73.06 ± 2.27	79.42 ± 0.83	75.10 ± 0.85	73.20 ± 1.70	71.88 ± 2.76	71.84 ± 1.68	72.67 ± 1.73
GRPO w/ Tool	77.86 ± 1.80	75.40 ± 2.13	71.84 ± 3.19	77.82 ± 2.80	75.77 ± 2.31	73.64 ± 1.05	73.26 ± 0.60	76.70 ± 0.97	74.19 ± 0.41
ToolSciVer	82.29 ± 0.23	84.01 ± 1.01	78.10 ± 1.25	80.35 ± 1.66	81.20 ± 0.95	73.20 ± 0.57	78.47 ± 4.21	75.40 ± 3.41	74.65 ± 0.59

C.3 Type-Aware Tool Selection

We next examine whether, when encouraged to solve MSCV with appropriate tools, the learned policy selects tools according to the visual evidence type. Using the diagnostic subsets described in Appendix B.6, we compare Tool Prompt (CoT w/ Tool) and ToolSciVer by accuracy change over non-tool CoT, tool-execution success, and tool-family distribution.

Figure 2 shows that ToolSciVer learns a more accurate, reliable, and type-aware tool-use policy than Tool Prompt. Across table, chart/plot, and multi-panel evidence, ToolSciVer increases accuracy by an additional 12.85, 10.71, and 8.18 points, respectively, over prompt-only tool access (Figure 2a). The improvement is especially notable for multi-panel evidence: Tool Prompt falls below non-tool CoT (-0.99 points), whereas ToolSciVer yields a $+7.19$-point gain. Tool execution also becomes more reliable on every subset, reaching 97.39% on tables and 96.51% on charts/plots (Figure 2b). Finally, ToolSciVer increases the share of calls routed to the matched tool family from 78.05% to 89.25% for tables, from 67.53% to 74.24% for charts/plots, and from 48.89% to 53.85% for multi-panel figures (Figure 2c). Together, these results indicate that the trained policy improves not only call executability, but also the alignment between visual evidence type and tool choice, which is accompanied by consistent downstream accuracy gains.

C.4 Efficient Tool Use

Table 4 in Section 5.2 reports the comparison with and without the OTC efficiency term s_{eff}. Both variants use the tool-encouraging test-time prompt described in the main text, with the other reward components and evaluation settings held fixed. We further examine the resulting distribution of tool-call depth. Figure 3 shows that OTC raises the direct-answer rate from 5.2% to 28.8% and lowers the one- and two-call rates from 85.1% to 67.9% and from 9.7% to 3.3%, respectively. Thus, s_{eff} favors economical trajectories rather than indiscriminately suppressing tool use; a complementary two-call example appears in Appendix C.7.

C.5 Qualitative Evidence Acquisition

Figure 4 illustrates how structured chart observations help acquire claim-relevant visual evidence. The non-tool CoT baseline fails to recover the country labels and values, while ToolSciVer obtains the required evidence through Chart Parse.

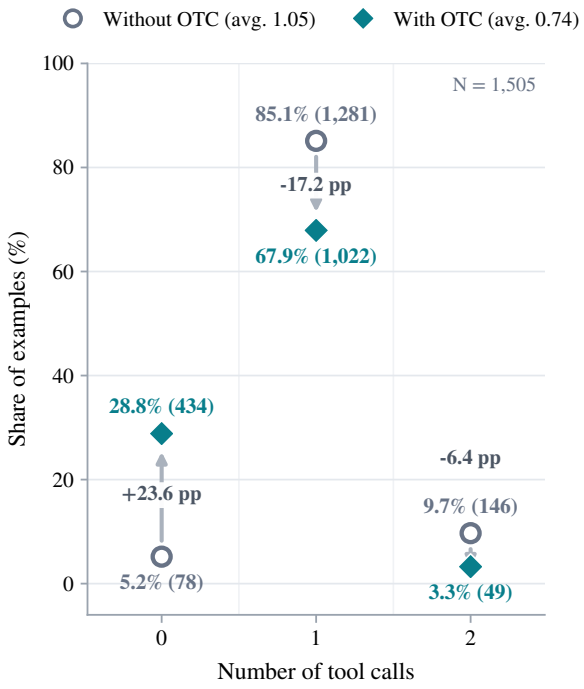


Figure 3: Tool-call depth with and without OTC. Labels show percentage (count); $N = 1{,}505$.

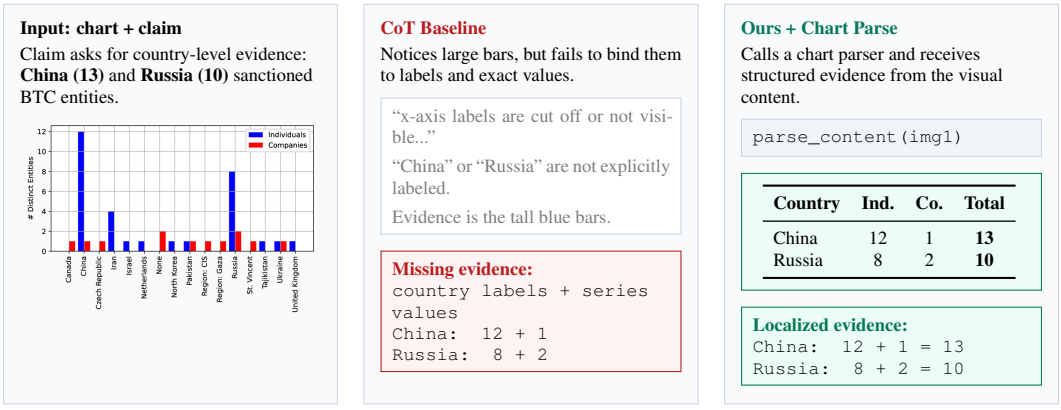


Figure 4: Case study of chart evidence acquisition. The CoT baseline identifies salient bars but fails to recover the corresponding labels and values. ToolSciVer calls `parse_content` and obtains structured country-level evidence needed for verification.

C.6 Error Analysis of Proprietary VLMs

We analyze GPT-5.4, Claude Sonnet 4.6, GPT-5.6 Sol, and Claude Opus 5 under the non-tool CoT setting, using the original benchmark labels as in Table 2. The 1,000-example SciVer split contains 505 positive and 495 negative claims, yet all four models strongly favor *No* predictions. Table 13 reports their confusion matrices; Figure 5 summarizes prediction bias and attributed error stages.

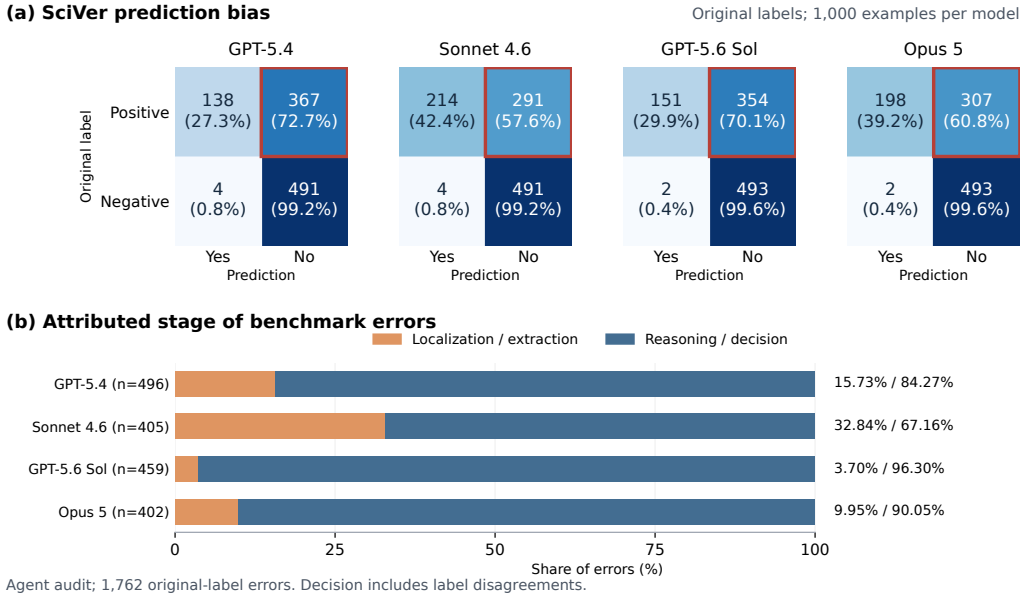


Figure 5: Error analysis of proprietary VLMs. (a) Row-normalized confusion matrices on the 1,000-example SciVer evaluation set; cells show counts and within-label percentages, with false negatives outlined. (b) Structured agent-audit distributions over 1,762 model-error records on SciVer and MuSciClaims: 901 from GPT-5.4 and Sonnet 4.6, and 861 from GPT-5.6 Sol and Opus 5. The reasoning/decision category includes disagreements about evidence sufficiency and benchmark labels.

Table 13: Confusion counts and positive-prediction rates on SciVer, using original labels.

Model	TP	FN	FP	TN	Yes rate
GPT-5.4	138	367	4	491	14.2%
Claude Sonnet 4.6	214	291	4	491	21.8%
GPT-5.6 Sol	151	354	2	493	15.3%
Claude Opus 5	198	307	2	493	20.0%

False negatives constitute 98.92%, 98.64%, 99.44%, and 99.35% of SciVer errors for GPT-5.4, Sonnet 4.6, GPT-5.6 Sol, and Opus 5, respectively. For GPT-5.6 Sol and Opus 5, false-negative rates are 70.10% and 60.79%, while both false-positive rates are only 0.40%; balanced accuracies are 64.75% and 69.40%. Their accuracy therefore reflects a conservative decision policy on this nearly balanced split. On MuSciClaims, false negatives also dominate, accounting for 84/103 errors for GPT-5.6 Sol and 71/93 for Opus 5 (81.55% and 76.34%).

Subgroup results further localize the difficulty. On SciVer, GPT-5.6 Sol reaches 71.13% on table-only evidence versus 61.64% on chart evidence; Opus 5 reaches 78.69% versus 65.16%. Accuracy decreases from one-image to two-image examples: 66.07% to 62.70% for Sol and 72.22% to 65.93% for Opus. Direct claims are easier (77.82%/79.44%) than Analytic (54.69%/65.23%), Parallel (64.08%/67.76%), or Sequential claims (61.35%/64.14%). These descriptive gaps parallel the table/chart gaps for GPT-5.4 (70.10%/58.83%) and Sonnet (82.82%/65.44%), but subgroup accuracy alone does not identify the failed stage.

Case-level error taxonomy. The structured agent audit covers 901 error records from GPT-5.4 and Sonnet 4.6 and all 861 original-label errors from GPT-5.6 Sol (459) and Opus 5 (402); the latter cover 501 distinct examples. Each returned explanation is examined separately; we reuse documented visual evidence and directly inspect all 32 previously unaudited examples plus identified conflicts. These annotations are not independently human-adjudicated and concern observable explanations, not hidden native reasoning traces.

We assign one category according to the earliest attributable failure stage. *Visual localization/extraction* covers a decisive panel, value, label, trend, or relation that is misread or unavailable from the supplied visual input. *Reasoning/decision* covers subsequent comparison, calculation, integration, interpretation, or answer selection after the relevant observations are recovered. The latter also retains evidence-sufficiency and label disagreements under the forced binary taxonomy; it is not a count of proven reasoning defects. Ambiguous attributions are marked low confidence (390 of the 861 GPT-5.6 Sol and Opus 5 records); input insufficiency can also receive a low-confidence extraction attribution.

Table 14: Structured agent attribution over all original-label errors from both benchmarks. Percentages are within each model's error set.

Model	Localization/extraction	Reasoning/decision	Total
GPT-5.4	78 (15.73%)	418 (84.27%)	496
Claude Sonnet 4.6	133 (32.84%)	272 (67.16%)	405
GPT-5.6 Sol	17 (3.70%)	442 (96.30%)	459
Claude Opus 5	40 (9.95%)	362 (90.05%)	402
Combined	268 (15.21%)	1,494 (84.79%)	1,762

For GPT-5.6 Sol, reasoning/decision accounts for 341/356 SCIVER errors (95.79%) and 101/103 MUSCICLAIMS errors (98.06%); for Opus 5, the corresponding counts are 276/309 (89.32%) and 86/93 (92.47%). The pattern persists among the 222 high-confidence GPT-5.6 Sol and Opus 5 records, of which 195 (87.84%) are assigned to reasoning/decision. Responses frequently recover the plotted trend but demand explicit significance tests, exact agreement for approximate values, or additional causal evidence; the skeptical evaluation prompt can contribute to this rejection policy. There are also concrete extraction and decision failures: Opus reverses the direction of a Li 1s peak shift on a descending energy axis (`musci-0366`), while Sol correctly reads an overlapping, nonsignificant ascites comparison but accepts a claim of different onset times (`musci-0289`).

Label sensitivity and comparison limits. The main results and the error counts above use the original benchmark labels. As a separate sensitivity analysis, under the reviewed-label export, 141 of Sol's 459 original-label errors and 147 of Opus's 402 become correct. Removing those cases leaves reasoning/decision shares of 311/318 (97.80%) and 227/255 (89.02%); this is a sensitivity analysis of the audited subset, not a complete re-audit of reviewed-label errors. For this alternative label set only, full-dataset accuracies are 76.70%/80.59% for Sol and 81.60%/84.55% for Opus on SCIVER/MUSCICLAIMS; these are distinct from the original-label results in Table 2. Label corrections include mismatched captions or figures. Different token caps (8,192 for GPT-5.6 Sol and Opus 5; 1,536 for the earlier GPT-5.4 and Sonnet 4.6 evaluations) and the rendering exceptions noted in Appendix B.4 further limit controlled comparisons across these proprietary references.

C.7 COMPLEMENTARY TWO-CALL EVIDENCE

Although Figure 3 shows a predominantly single-call policy, OTC retains 49 two-call trajectories (3.3%) and therefore does not restrict the model to one-shot routing. Figure 6 illustrates a retained two-call trajectory in which the two calls are necessary for different clauses of the same claim. The first chart supports the CNN-Momentum accuracy change but contains no evidence about CNN2-Momentum loss; the model therefore inspects the second chart, integrates the two observations, and returns the gold answer *Yes*.

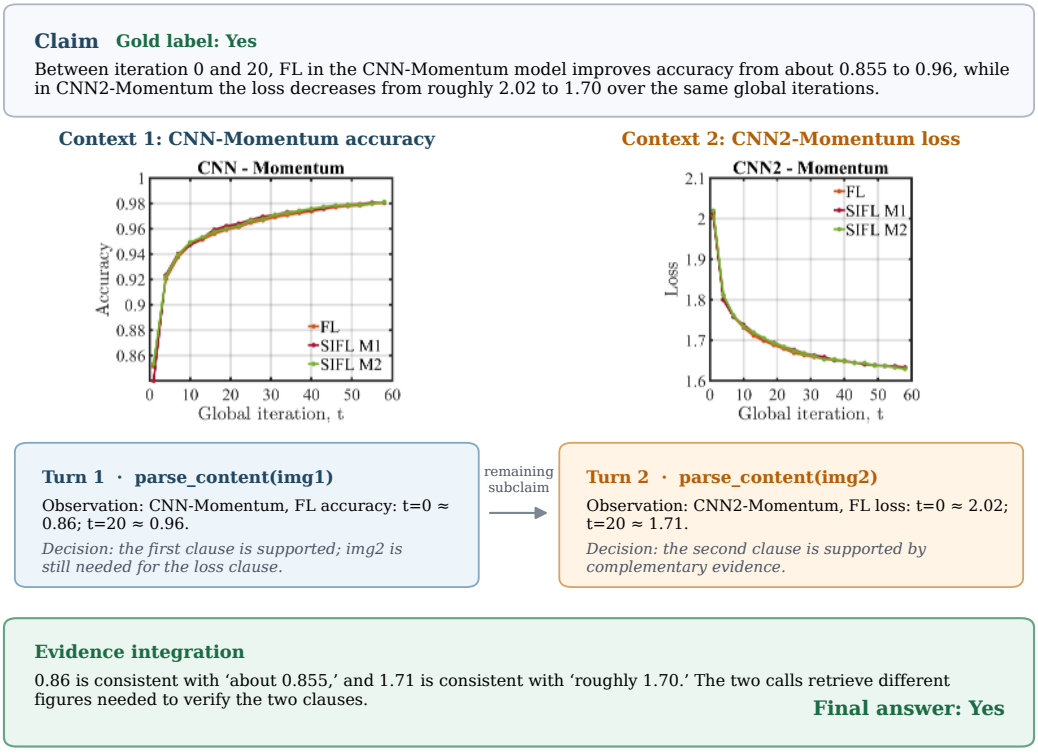


Figure 6: Two-turn complementary-evidence acquisition. The first call verifies the CNN-Momentum accuracy clause, and the second retrieves a different figure needed for the CNN2-Momentum loss clause. The final decision integrates both observations rather than repeating the first retrieval.