PAPERRECON: CHARACTERIZING SCIENTIFIC WRITING CAPABILITIES OF AI AGENTS THROUGH PAPER RECONSTRUCTION

Anonymous authors
Paper under double-blind review

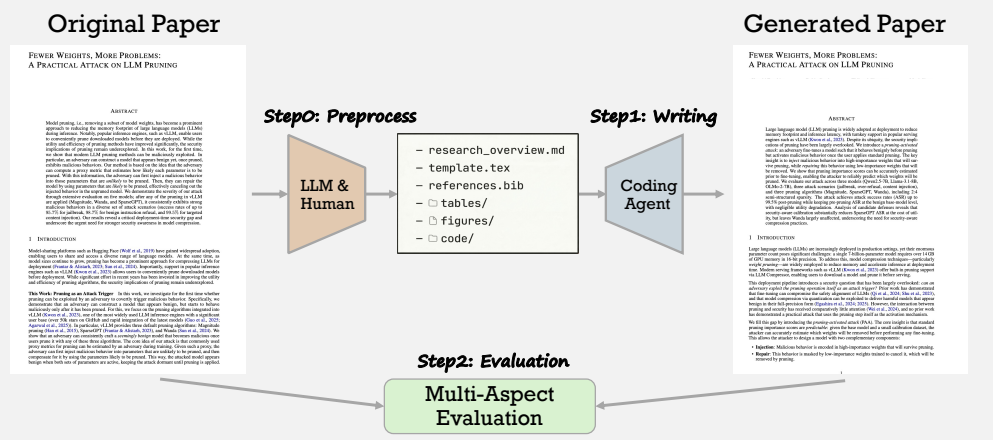


Figure 1: **Paper Reconstruction Evaluation**. Given structured resources derived from an original paper, a coding agent reconstructs the full paper. The generated paper is then compared with the original paper to evaluate writing performance from multiple perspectives.

ABSTRACT

As coding agents become capable of generating entire research papers, assessing their actual writing quality and risks has become critical. Existing evaluations primarily rely on overall review scores, which assess acceptance-worthiness but provide limited insight into how individual agents write. We introduce **Paper Reconstruction Evaluation** (PaperRecon), a novel framework for characterizing scientific writing capabilities from multiple perspectives. PaperRecon formulates scientific writing as a reconstruction task. Given a structured overview and supporting resources derived from an existing paper, an agent reconstructs the full paper, which is then directly compared with the original. We evaluate two core aspects of scientific writing. **Content Coverage** measures how well the reconstructed paper captures the intended scientific content using section-specific rubrics, while **Hallucination** analysis identifies claims that contradict the source paper and remain unsupported after checking the provided resources. We further conduct four diagnostic analyses to characterize the effects of the harness and backbone model, input information, citation behavior, and verbosity. To support this evaluation, we introduce **PaperWrite-Bench**, comprising 72 recent papers from top-tier venues in 2025 and 2026, filtered to reduce contamination from prior model knowledge. Our evaluation reveals clear differences across model families. Claude-based agents achieve higher content coverage than GPT-based agents but produce substantially more hallucinations. These results show that scientific writing capability cannot be adequately characterized by a single overall score and demonstrate how PaperRecon provides fine-grained insights into the strengths and weaknesses of current coding agents.

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.

1 INTRODUCTION

As AI tools such as coding agents become increasingly capable and widely available, AI-assisted paper writing is becoming more prevalent. Recent developments, including ICLR 2026's statement on AI-generated submissions (ICLR 2026 Program Chairs, 2025), highlight the growing need to understand the impact of AI on scientific writing. To advance scientific progress while preserving academic integrity, it is essential for the research community to continuously monitor the capabilities and risks of AI-driven writing through rigorous and reliable evaluation.

Evaluating the paper-writing capabilities of AI agents remains challenging. Existing approaches often ask agents to produce novel research papers and evaluate them using LLM-based reviewers (Lu et al., 2026; Miyai et al., 2026; Weng et al., 2025; Zhu et al., 2025). However, these scores reflect not only writing quality but also factors such as idea novelty and experimental strength, making it difficult to isolate writing capability. Recent work has therefore adopted a more controlled setting in which agents write papers from research overviews derived from existing papers (Song et al., 2026). However, evaluation still relies on review scores, which primarily assess overall acceptance-worthiness and provide limited insight into the writing behavior of individual agents. Fine-grained evaluation is therefore needed to characterize their strengths, weaknesses, and associated risks.

To address this limitation, we propose **Paper Reconstruction Evaluation** (PaperRecon), a framework for characterizing the scientific writing capabilities of AI agents. Rather than assessing whether an agent can produce an acceptance-worthy paper, PaperRecon formulates evaluation as a reconstruction task, allowing generated papers to be directly compared with their original counterparts. This controlled comparison enables us to identify which scientific content is successfully reconstructed and to verify the factual accuracy of generated claims.

The overview of PaperRecon is shown in Figure 1. We first compress an existing paper into a structured summary, `research_overview.md`, that retains its essential scientific content. Given this summary together with minimal additional resources, including tables, figures, and a bibliography file, an agent reconstructs the full paper. We then evaluate the reconstruction from two complementary perspectives: (i) **section-level content coverage**, using section-specific rubrics to measure how well the generated paper captures the key content of the original, and (ii) **claim-level hallucination**, verifying generated claims against the original paper and provided resources. Together, these evaluations characterize both how comprehensively and how reliably an agent reconstructs scientific content.

Beyond these two core metrics, directly comparing generated papers with their originals enables more detailed analyses of agent writing capabilities. Specifically, we analyze four aspects: (i) **harness and model analysis**, disentangling the effects of the harness and backbone model by swapping models across harnesses; (ii) **input information**, examining how reconstruction quality changes with more detailed research overviews; (iii) **citation behavior**, measuring how faithfully agents recover the references used in the original paper; and (iv) **verbosity**, comparing the amount of content generated for each section against the original paper. These analyses allow PaperRecon to characterize not only overall reconstruction performance, but also the distinct writing characteristics of individual agents.

For PaperRecon, we introduce **PaperWrite-Bench**, a benchmark of 72 recent papers from top-tier venues in 2025 and 2026 across machine learning, computer vision, and natural language processing. Constructing such a benchmark requires two key considerations: (i) minimizing contamination from prior knowledge and (ii) ensuring scalable construction as new papers and models emerge. To address the first, we screen candidate papers using two recall and sentence completion tests and retain only papers according to the filtering criteria described in Section A.5). To address the second, we develop a scalable construction pipeline in which benchmark components are first generated by LLMs and then verified and edited by humans, while using separate models for generation and evaluation to reduce potential self-preference. This design allows PaperWrite-Bench to be efficiently extended with new papers and re-filtered against new models.

We evaluate recent powerful and widely used agents, Claude Code (Anthropic, 2025) and Codex (OpenAI, 2025), across a range of underlying models, from Claude Sonnet 4.6 (Anthropic, 2026) and Claude Sonnet 5 (Anthropic, 2026b) to GPT-5.4-mini (OpenAI, 2026b) and GPT-5.6-series (OpenAI, 2026a). Our experiments yield the following key findings:

1. **Claude-series achieves higher content coverage than GPT-series.** Claude-series better captures the key elements required for scientific writing across sections.
2. **GPT-series produces fewer hallucinations than Claude-series.** While Claude-series exhibits 5.4–7.7 contradictory claims per paper on average, GPT-series stays below 2.5 with every backbone.
3. **Writing capability improves with model advances.** This also suggests that PaperRecon serves as a reliable metric for tracking progress in writing ability.
4. **Paper reconstruction reveals additional characteristics.** Our diagnostic analyses show that reconstruction performance is primarily driven by the backbone model rather than the harness, improves with more detailed input information, and exhibits distinct citation and verbosity patterns across agents.

Our work makes the following contributions to the research community:

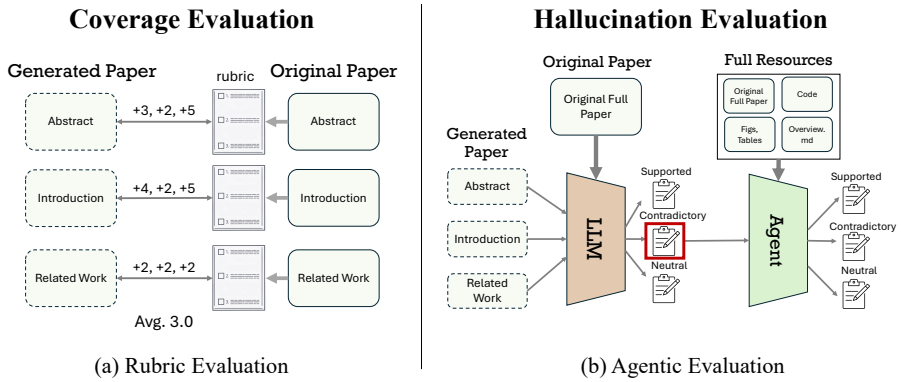
- **Paper Reconstruction Evaluation (PaperRecon):** We propose the fine-grained evaluation framework for scientific writing, which formulates evaluation as paper reconstruction and systematically measures both content coverage and hallucination.
- **PaperWrite-Bench:** We introduce PaperWrite-Bench, a benchmark constructed from 72 recent papers across diverse research domains, enabling comprehensive evaluation of agents' ability to reconstruct papers from structured source materials.
- **Characterization of modern scientific writing agents:** We systematically evaluate modern agents and reveal distinct characteristics across model families in content coverage and hallucination, together with differences arising from harnesses, backbone models, input information, citation behavior, and verbosity.

2 Related Work

Automation of the Research Process. Early evaluations of AI-generated scientific papers primarily focused on systems that automate the entire research process, including idea generation, implementation, experimentation, analysis, and paper writing (Lu et al., 2026; Intology, 2025; Tang et al., 2025; Miyai et al., 2026). These studies are important for assessing how autonomously AI can conduct research. However, the quality of the resulting papers depends on factors beyond writing, such as the novelty of the ideas and the strength of the experimental results, making it difficult to isolate paper-writing capability. Given the growing use of AI to assist or automate scientific writing itself, evaluating writing as a distinct capability is increasingly important.

Isolation of Scientific Writing. Recent work has begun to isolate paper writing from the broader research process. In particular, PaperOrchestra (Song et al., 2026) introduces a controlled setting in which AI agents write papers from research overviews and other materials derived from existing papers, separating writing from idea generation and experimentation. PaperOrchestra evaluates generated papers using citation coverage, literature review quality, and holistic manuscript quality. However, it does not explicitly evaluate which scientific content from the source paper is captured or omitted, or systematically verify factual errors at the claim level. PaperRecon complements these evaluations by using the original paper as explicit ground truth to measure section-level content coverage and claim-level hallucination.

Fine-Grained Evaluation of Scientific Writing. AI-written papers have commonly been evaluated using review scores assigned by AI reviewers. However, such scores aggregate overall paper quality and provide limited insight into specific strengths and failures. Moreover, prior work has shown that papers containing fabrications can still receive high review scores (Jiang et al., 2026; Miyai et al., 2026). As a result, closer evaluations have prioritized detecting fabricated content, particularly non-existent citations and unverifiable references (Walters & Wilder, 2023; Sakai et al., 2026; Ansari, 2026). Recent work (Meng et al., 2026) further verifies references, reported scores, and consistency between method descriptions and implementations. However, this approach primarily verifies generated claims against external evidence, such as references, code, and experimental results. While effective for detecting unsupported or incorrect content, it cannot identify important scientific content that an agent fails to include.



(a) Rubric Evaluation (b) Agentic Evaluation

Figure 2: **Overview of PaperRecon evaluation pipeline.** Our evaluation compares generated papers against original papers along two complementary axes: rubric-based content coverage evaluation and agentic hallucination evaluation.

Beyond scientific writing, reconstructing an existing document from provided information has been studied in tasks such as Wikipedia and long-form article generation (Zhang et al., 2025; Prabhumoye et al., 2019). PaperRecon brings this reconstruction setting to scientific writing, where an agent reconstructs an existing paper from its scientific content and the reconstruction is directly compared with the original. This enables fine-grained evaluation of what scientific content is captured or omitted and what unsupported content is introduced by the agent.

3 PaperWrite-Bench

3.1 Benchmark Overview

PaperWrite-Bench consists of 72 papers, selected from a diverse set of top-tier conferences, including ACL 2025/2026, COLM 2026, CVPR 2025/2026, ECCV 2026, ICCV 2025, ICLR 2025/2026, ICML 2025/2026, NeurIPS 2025, and ACMMM 2025, covering a wide range of domains such as computer vision, natural language processing, and machine learning. Among the 72 papers, 46 focus on proposing new methods, 13 introduce new benchmarks, and 13 combine both contributions. This distribution naturally reflects the diversity of paper types found in recent top-tier conferences. Our benchmark covers 366 sections and 4,760 rubric evaluation points, with hallucination analysis performed at the sentence level, providing sufficient scale for the evaluation.

3.2 Benchmark Construction Pipeline

Removing Memorized Papers. To reduce the risk of reconstruction from prior knowledge, we apply two memorization tests. First, we test title recall by asking each model to identify a paper from only its authors and venue. We apply this test to 100 candidate papers using all evaluated models and Claude Opus 5, and remove any paper recalled by at least one model. This removes 28 papers, leaving 72. Second, we test sentence completion on the remaining 72 papers. Only one of 360 completions reaches our similarity threshold of 0.8. We further test eight sentences from the same paper and find no additional evidence of memorization, so no paper is removed in the second test (Section A.5).

Construction of research_overview.md. For each paper, we generate a `research_overview.md` with an LLM, which summarizes the key information required for reconstruction. To ensure quality, three authors independently verify assigned subsets against the original papers, code, and the other provided resources, confirming that each overview accurately reflects the source materials and contains sufficient information to faithfully reconstruct the original paper. On average, each file contains 528 words (Section A.2).

Extraction of Tables, Figures, References, and Code. From the arXiv source we extract the tables and figures, list them for the agent with their file paths and a one-line description each, augment every `.bib` entry with the abstract retrieved from the Semantic Scholar API, and add the paper's codebase where one is available (Section A.3).

Construction of Evaluation Rubrics. For each paper, we construct section-specific rubrics that capture the key elements required for faithful reconstruction. Each rubric item represents a concrete and verifiable point (*e.g.,* "Problem motivation: improve reasoning in vision-language models" for the Abstract, or "Evaluation covers both visual reasoning and general image understanding benchmarks" for the Experiment section). The rubrics are initially generated from the original paper by an LLM. To validate their quality, we ask the original authors of three papers to rate each of their rubric items as *necessary*, *acceptable*, or *unnecessary*: of 198 items, 132 are rated necessary, 65 acceptable, and only one unnecessary, showing that the rubrics largely capture what the people who wrote the papers consider essential. Section C.2 provides an example. Across 72 papers, we construct 4,760 rubric items, with an average of 9.6 for the Abstract, 13.8 for the Introduction, 12.9 for Related Work, 14.0 for the Method, 14.1 for Benchmark Construction, and 14.6 for the Experiment section.

Template and Style File Construction. We extract the section structure and style files of each original paper from its arXiv source to create a `template.tex`. The template preserves only the original section structure, which agents follow when generating the paper. This provides a consistent section-level alignment between the generated and original papers, enabling more precise comparison.

4 Paper Reconstruction Evaluation

4.1 Problem Definition

PaperRecon is a framework for characterizing the scientific writing capabilities of coding agents through paper reconstruction. From each original paper, we provide the agent with the following resources: (1) `Research Overview`: a structured summary of the motivation, methodology, and key interpretations of the results; (2) `Paper Resources`: figures, tables, and references extracted from the original paper, with simplified captions and reference abstracts; and (3) `Code`: the associated codebase, when available. Each task includes an `AGENTS.md` (copied to `CLAUDE.md` for Claude Code) with identical writing instructions across agents, with variants for method, benchmark, and combined papers (Section C.3). We reuse the original references to isolate scientific writing from the separate problem of reference collection.

4.2 Evaluation Protocol

We evaluate generated papers by comparing them against the original papers along two complementary axes: (i) section-level content coverage and (ii) claim-level hallucination. Figure 2 shows the overview of the content-coverage and hallucination evaluation pipelines. To compute these scores, we first perform section classification and matching, then evaluate each matched pair.

4.2.1 Section Classification and Matching

Papers organize sections differently, so we first extract all sections from both the original and generated LaTeX files. Next, we map them into seven common categories: *Abstract*, *Introduction*, *Method*, *Benchmark Construction*, *Experiment*, *Related Work*, and *Conclusion*. When multiple raw sections map to the same category, they are merged. This classification is performed once and shared across all subsequent evaluation steps. We manually verify the classification on a subset of 30 papers and confirm that the sections are accurately assigned to the corresponding categories.

4.2.2 Rubric Evaluation

To enable accurate and fine-grained evaluation, we adopt a rubric evaluation (Fan et al., 2024; Phung et al., 2023; Arora et al., 2025), scoring each generated section against the rubric the benchmark provides for that paper (Section 3.2).

Table 1: **Content coverage evaluation.** Rubric evaluation scores by model and section (1–5 scale). We observe that Claude-series achieves higher content coverage quality than GPT-series.

Agent	Model	Abs.	Intro.	Rel.	Meth.	Bench.	Exp.	Avg.
Codex	GPT-5.4-mini	3.80	3.36	2.55	3.24	3.70	3.63	3.34
Codex	GPT-5.6-luna	3.89	3.58	2.43	3.60	3.80	3.69	3.46
Codex	GPT-5.6-terra	4.13	3.68	2.63	3.56	3.87	3.67	3.54
Claude Code	Sonnet 4.6	4.19	3.91	3.04	3.71	3.93	3.89	3.74
Claude Code	Sonnet 5	**4.29**	**4.11**	**3.28**	**3.86**	**4.16**	**4.02**	**3.93**

Table 2: **Hallucination evaluation.** Average number of contradictory claims (major and minor) per paper in each section, over the 72 papers, with Claude Opus 5 as the Pass 2 verifier. Codex produces fewer of them than Claude Code.

Agent	Model	Abs.	Intro.	Rel.	Meth.	Bench.	Exp.	Total
Codex	GPT-5.4-mini	0.1	0.1	0.2	**0.4**	**0.2**	1.1	2.1
Codex	GPT-5.6-luna	0.1	0.1	**0.1**	0.6	**0.2**	1.2	2.4
Codex	GPT-5.6-terra	**0.0**	0.1	0.2	0.5	0.3	**0.8**	**1.9**
Claude Code	Sonnet 4.6	0.2	0.7	0.3	1.7	0.7	4.1	7.7
Claude Code	Sonnet 5	0.2	0.5	0.5	0.6	0.5	3.1	5.4

For each section except the Conclusion, an LLM judge scores how well the generated text covers each rubric item on a 1–5 scale. The scale runs from 5 (fully and accurately described) through 4 (core idea present, some details missing), 3 (partially described, with gaps or vagueness) and 2 (only a superficial or indirect mention) to 1 (absent from the generated section). Alongside these text elements, the rubric also scores figures and tables. We align figures and tables between the original and generated papers and evaluate each matched element on a 1–5 scale. Figures are evaluated based on contextual appropriateness, while tables are evaluated based on numerical accuracy, structural alignment, and content consistency. Missing figures or tables receive a score of 1. The final rubric score for each section is the average across all rubric elements (text, figure, and table items combined).

4.2.3 Hallucination Analysis

We identify factual errors via a two-stage, claim-level analysis.

Stage 1: Claim Extraction. For each section except the Conclusion, the judge extracts all concrete and verifiable claims from the generated paper and classifies each as *Supported* if it is directly stated in or logically supported by the original paper, *Neutral* if it is not stated in the original paper but does not contradict it, or *Contradictory* if it conflicts with specific information in the original paper.

Stage 2: Verification. All contradictory claims identified in Stage 1 are re-evaluated by a coding agent using the original paper and the full resources provided to the writing agent. Claims supported by these resources are not considered hallucinations, even if they differ from the original paper. This second stage reduces false positives while keeping the evaluation computationally efficient. Finally, we report the number of contradictory claims. Following prior work on scientific-paper integrity (Meng et al., 2026), we focus on detecting and verifying factual errors rather than exhaustively identifying all factual errors in each paper. Measuring hallucination recall at the full-paper level remains a broader challenge for the community, as it requires identifying the complete set of factual errors in long and technically complex scientific papers.

5 Experiments

5.1 Experimental Setup

Agents. We evaluate two coding agents: Claude Code (Anthropic, 2025) and Codex (OpenAI, 2025). Claude Code is evaluated with Sonnet 4.6 (Anthropic, 2026) and Sonnet 5 (Anthropic,

Table 3: **Citation evaluation.** Precision, recall and F1 of the cited keys against the GT paper's, and Hal.: citations whose key has no entry in the provided `references.bib`, averaged over the 72 papers.

Agent	Model	Prec.	Recall	F1	Hal.
Codex	GPT-5.4-mini	0.83	0.59	0.67	**0.00**
Codex	GPT-5.6-luna	0.85	0.46	0.58	0.01
Codex	GPT-5.6-terra	**0.86**	0.51	0.62	**0.00**
Claude Code	Sonnet 4.6	0.83	0.71	0.75	**0.00**
Claude Code	Sonnet 5	0.80	**0.84**	**0.81**	**0.00**

Table 4: **Effect of research overview length.** Rubric score and contradictory claims per paper with the default overview and with the long one, on the same 25 papers.

	Rubric ↑		Hallu. ↓	
Model	Default	Long	Default	Long
GPT-5.6-luna	3.46	**3.67**	2.4	**1.2**
GPT-5.6-terra	3.59	**3.71**	1.3	**1.2**
Sonnet 4.6	3.75	**3.91**	7.7	**5.0**
Sonnet 5	3.89	**4.05**	5.0	**4.0**

2026b), and Codex with GPT-5.4-mini (OpenAI, 2026b), GPT-5.6-luna, and GPT-5.6-terra (OpenAI, 2026a), resulting in five agent configurations in total.

Models Used by the Pipeline. To mitigate potential self-preference, we deliberately employ a model that is not used to generate any of the evaluated papers. Specifically, DeepSeek-V4-Pro (Xu et al., 2026) generates the initial `research_overview.md` and rubrics, and serves as the LLM judge for rubric evaluation and the first stage of hallucination analysis. For the second stage of hallucination analysis, we use Claude Code with Claude Opus 5 (Anthropic, 2026a) across all configurations.

Writing Pipeline. We adopt a deliberately simple writing pipeline to evaluate the scientific writing capabilities and associated risks of current agents. The pipeline includes a compilation feedback loop for correcting LaTeX errors and a page-limit adjustment step following Lu et al. (2026).

5.2 Main Results

We report rubric-based content coverage scores in Table 1. For hallucination evaluation, we report the average number of hallucinations, defined as claims classified as *contradictory* (both major and minor), in Table 2.

We summarize the key findings below.

[F1] Claude-series outperforms GPT-series in content coverage. As shown in Table 1, Claude Code with Sonnet 4.6 and Sonnet 5 consistently achieves higher content coverage scores than Codex with GPT-series across all sections, indicating a stronger ability to capture and articulate key scientific points. However, the best-performing agent, Claude Code with Claude Sonnet 5, reaches a score of 3.93, suggesting that there remains substantial room for improvement.

[F2] Claude-series exhibits substantially more hallucinations, while GPT-series significantly reduces them. Table 2 reports the average number of hallucinations per paper across sections. We observe a clear contrast: although Claude-series achieves higher content coverage, it produces more hallucinations, 7.7 per paper with Claude Sonnet 4.6 and 5.4 even with Claude Sonnet 5. In contrast, GPT-series keeps hallucinations below 2.5 per paper with every backbone, and to 1.9 with GPT-5.6-terra. These results reveal a clear contrast between content coverage and hallucination across model families, highlighting the importance of evaluating both dimensions.

[F3] Writing capability improves with model advances. Our evaluation framework captures performance gains from model improvements. Within the same model series, we observe consistent gains from Claude Sonnet 4.6 (Anthropic, 2026) to Claude Sonnet 5 (Anthropic, 2026b), with the rubric score improving from 3.74 to 3.93 and hallucinations decreasing from 7.7 to 5.4 per paper. Similarly, from GPT-5.6-luna to GPT-5.6-terra (OpenAI, 2026a), the rubric score improves from 3.46 to 3.54 and hallucinations decrease from 2.4 to 1.9 per paper. These consistent improvements demonstrate that PaperRecon can capture progress in scientific writing capability.

However, newer models do not necessarily improve along every dimension. Compared with GPT-5.4-mini, GPT-5.6-luna achieves higher content coverage (3.46 vs. 3.34) but produces slightly more hallucinations (2.4 vs. 2.1). These results further demonstrate the importance of evaluating writing capability along multiple dimensions rather than relying on a single overall score.

Table 5: **Verbosity of the generated papers.** Words per section, averaged over the papers.

Agent	Model	Abs.	Intro.	Rel.	Meth.	Bench.	Exp.
Original papers		176	637	421	1337	826	1332
Codex	GPT-5.4-mini	160	492	519	950	1209	1545
Codex	GPT-5.6-luna	184	458	280	1037	1111	1689
Codex	GPT-5.6-terra	196	458	359	969	1165	1473
Claude Code	Sonnet 4.6	199	514	462	784	625	1291
Claude Code	Sonnet 5	**225**	**671**	**547**	968	736	1326

Table 6: **Crossing harness and backbone.** Rubric score and hallucinations per paper on the same 25 papers.

Backbone	CC	Codex
Rubric ↑		
Sonnet 5	3.89	3.89
GPT-5.6-terra	3.61	3.59
Hallucinations ↓		
Sonnet 5	5.0	5.5
GPT-5.6-terra	1.5	1.3

5.3 Human Validation

Rubric Scoring Validation. To validate rubric scoring, we manually inspect 100 scored rubric items sampled across models and sections. Human reviewers agree with 98 of the 100 scores, while the remaining two receive lower scores from the automated judge than from the human reviewers. These results indicate that the rubric scores closely align with human judgments, with the few disagreements reflecting underestimation rather than overestimation.

Hallucination Validation. To validate our hallucination analysis, three reviewers manually examined all 79 claims identified as contradictory across three original papers, each generated by the five evaluated agent-model combinations. Each claim was judged as either *correctly identified* or *incorrectly identified* as a hallucination based on the original paper and the resources provided to the writing agent. Of the 79 claims, 78 were correctly identified, yielding 98.7%. These results demonstrate the high precision of our hallucination detection pipeline.

Relation to Holistic Human Preference. We examine how our evaluation relates to holistic human judgments. Two experienced reviewers, who are not authors of this paper, compare papers reconstructed by four agent configurations from eight source papers, four assigned to each reviewer, giving 48 pairs in total. For each pair, the reviewer reads two generated papers and selects the better generated one or a tie, without knowing which agent produced each paper. We then compare these preferences with the ordering induced by our rubric scores. Excluding the two pairs judged as ties by human reviewers, rubric-based rankings agree with human preferences in 34 of 46 pairs (74%). Across all 48 pairs, Kendall's rank correlation between human preferences and rubric scores is $\tau_b = 0.45$ ($p = 0.002$). Holistic preference combines multiple aspects of writing, whereas PaperRecon evaluates them separately. Therefore, this agreement can be regarded as evidence that the rubric captures an important aspect of human judgment, rather than as an alternative to the full human evaluation.

5.4 Analysis

Effect of Research Overview Length. We examine how the level of detail in the research overview affects reconstruction quality by comparing the default overview with a more detailed version (528 vs. 1,584 words on average). As shown in Table 4, increasing the amount of information consistently improves content coverage scores and reduces contradictory claims across all configurations. However, constructing highly detailed research overviews requires substantial effort. Therefore, we focus on evaluating agents' ability to write papers from the default, more concise research overviews.

Citation Errors. Table 3 reports citation accuracy. Across 360 generated papers, only one citation key is absent from the provided `references.bib`, suggesting that fabricated citations are rare when a bibliography is provided. Instead, errors mainly concern reference selection. Codex cites only 25-33 of the 49.5 references used by the original papers on average, whereas Claude Code with Sonnet 5 cites 51.3 references and leaves only 9.4 original references uncited. Overall, Codex tends toward under-citation, while Claude Code tends toward over-citation.

Bias from the Verifying Agent. The second-stage verifier in the hallucination analysis uses Claude Opus 5, which is not itself evaluated but belongs to the same model family as some of the evaluated models. This may still introduce bias toward Claude-written papers. To examine this possibility, we

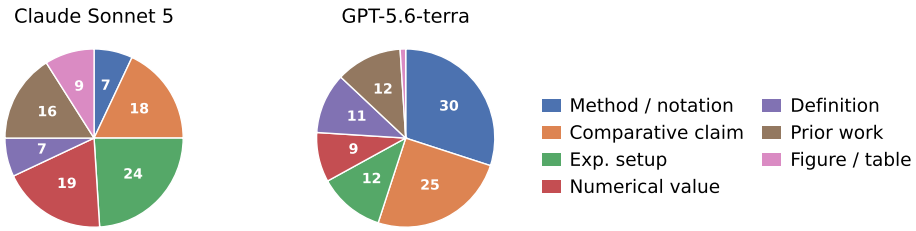


Figure 3: **Patterns of hallucination.** Categories of 100 contradictory claims sampled from Claude Sonnet 5 and GPT-5.6-terra.

repeat the verification using Codex with GPT-5.6-terra on a subset of 25 benchmark papers, each reconstructed by all five models (125 papers in total). The two verifiers show strong agreement in the number of contradictions (Pearson's $r = 0.81$, Spearman's $\rho = 0.81$) and give the same ranking of the five models. We find no evidence that the Claude Opus 5 verifier favors Claude-written papers. In fact, Claude Opus 5 is more accurate. Among 24 disagreements, a blind judge agrees with the Claude verifier in 18 cases, supporting the robustness of our findings.

Verbosity of the Generated Papers. To measure verbosity, we compare the word counts of each generated paper with those of the original papers. As shown in Table 5, verbosity varies substantially across models and sections. Most models write shorter Introduction and Method sections than the originals, while GPT models tend to produce longer Benchmark Construction and Experiment sections. Sonnet 5 stands out for its longer Abstract, Introduction, and Related Work sections, whereas GPT-5.6-luna produces the longest Experiment sections. Overall, these results reveal distinct section-level verbosity patterns across models rather than a general tendency for particular models to simply generate more text. Moreover, verbosity alone does not explain content coverage, as configurations with similar lengths can still differ in rubric scores.

Contribution of the Harness and the Backbone Model. To disentangle the effects of the harness and backbone model, we swap the backbones between Claude Code and Codex and evaluate the same 25 papers. As shown in Table 6, performance largely follows the backbone model rather than the harness. Sonnet 5 achieves similar content coverage under both harnesses (3.89), while GPT-5.6-terra scores 3.61 and 3.59. The same pattern holds for hallucinations: Sonnet 5 produces 5.0–5.5 contradictory claims per paper, compared with only 1.3–1.5 for GPT-5.6-terra. These results suggest that the differences between Claude Code and Codex are primarily driven by their backbone models rather than the harnesses themselves.

Patterns of Hallucination. To analyze error patterns, we categorize 100 contradictory claims each from Claude Sonnet 5 and GPT-5.6-terra into seven error types. As shown in Figure 3, Claude Sonnet 5 mainly makes errors in reporting experimental results, with setup, numerical, and figure errors accounting for 52% of its claims. In contrast, GPT-5.6-terra more often makes errors in describing methods, with method and definition errors accounting for 41%. Errors in comparative claims, such as incorrectly claiming that a method performs best across all metrics, are common in both models (18% and 25%). These claims often contradict the values in tables that the agents themselves reproduced correctly.

6 Conclusion, Limitations and Future Work

We introduced PaperRecon, a fine-grained framework for characterizing the scientific writing capabilities of AI agents through paper reconstruction, together with PaperWrite-Bench, comprising 72 recent papers from top-tier venues in 2025 and 2026.

One limitation of PaperRecon is that agents reconstruct papers from provided scientific content rather than conducting the research themselves. However, evaluating generation by reconstructing a target document from supporting sources is a well-established paradigm beyond scientific writing, including Wikipedia and long-form article generation (Zhang et al., 2025; Prabhumoye et al., 2019). To prevent contamination from prior model knowledge, we carefully apply two memorization tests to filter potentially contaminated papers. Given the trade-off between controlled, fine-grained evaluation and capturing the full research process, we prioritize the former to enable precise evaluation.

THE USE OF LARGE LANGUAGE MODELS

Research Ideation. The main research idea and the design of PaperRecon were developed by the authors without LLM assistance.

Implementation. We used AI coding assistants to write parts of the evaluation and experimental code. The authors reviewed and tested the generated code and verified the results reported in the paper.

Writing. We used LLMs as writing assistants to improve the wording, clarity, and organization of the manuscript. The scientific content, analyses, and claims were determined by the authors, who reviewed all LLM-assisted text.

The authors reviewed all AI-assisted work and take full responsibility for the final paper.

REFERENCES

Samar Ansari. Compound Deception in Elite Peer Review: A failure mode taxonomy of 100 fabricated citations at neurips 2025. *arXiv preprint arXiv:2602.05930*, 2026.

Anthropic. Claude Code. Technical report, Anthropic, 2025. URL https://www.anthropic.com/claude. Accessed: 2026-09-14.

Anthropic. Claude Sonnet 4.6. Technical report, Anthropic, 2026. URL https://www.anthropic.com/claude/sonnet. Accessed: 2026-09-14.

Anthropic. Introducing Claude Opus 5. https://www.anthropic.com/news/claude-opus-5, 2026a. Accessed: 2026-09-22.

Anthropic. Introducing Claude Sonnet 5, 2026b. URL https://www.anthropic.com/news/claude-sonnet-5. Accessed: 2026-09-22.

Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. Healthbench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

Zhiyuan Fan, Weinong Wang, Xing Wu, and Debing Zhang. Sedareval: Automated evaluation using self-adaptive rubrics. In *Findings of EMNLP*, 2024.

ICLR 2026 Program Chairs. ICLR 2026 Response to LLM-Generated Papers and Reviews. https://blog.iclr.cc/2025/11/19/iclr-2026-response-to-llm-generated-papers-and-reviews, 2025. Accessed: 2026-03-30.

Intology. Zochi technical report. 2025. URL https://www.intology.ai/blog/zochi-tech-report. Accessed: 2025-10-17.

Fengqing Jiang, Yichen Feng, Yuetai Li, Luyao Niu, Basel Alomair, and Radha Poovendran. BadScientist: Can a research agent write convincing but unsound papers that fool LLM reviewers? In *ACL*, pp. 24712–24727, 2026.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of ai research. *Nature*, 651(8107):914–919, 2026.

Rui Meng, Bhavana Dalvi Mishra, Jiefeng Chen, Chun-Liang Li, Palash Goyal, Mihir Parmar, Yiwen Song, Yale Song, Rajarishi Sinha, Parthasarathy Ranganathan, Burak Gokturk, Jinsung Yoon, and Tomas Pfister. ScientistOne: Towards human-level autonomous research via chain-of-evidence. *arXiv preprint arXiv:2605.26340*, 2026.

Atsuyuki Miyai, Mashiro Toyooka, Takashi Otonari, Zaiying Zhao, and Kiyoharu Aizawa. Jr. AI Scientist and Its Risk Report: Autonomous scientific exploration from a baseline paper. *TMLR*, 2026.

OpenAI. Codex, 2025. URL https://openai.com/codex/. Accessed: 2026-03-30.

OpenAI. GPT-5.6: Frontier Intelligence That Scales with Your Ambition, 2026a. URL https://openai.com/index/gpt-5-6/. Accessed: 2026-09-22.

OpenAI. Introducing GPT-5.4. March 2026b. URL https://openai.com/index/introducing-gpt-5-4/. Accessed: 2026-03-25.

Tung Phung, Victor-Alexandru Pădurean, José Cambronero, Sumit Gulwani, Tobias Kohn, Rupak Majumdar, Adish Singla, and Gustavo Soares. Generative ai for programming education: Benchmarking chatgpt, gpt-4, and human tutors. In *ICER*, volume 2, pp. 41–42, 2023.

Shrimai Prabhumoye, Chris Quirk, and Michel Galley. Towards content transfer through grounded text generation. In *NAACL*, pp. 2622–2632, 2019.

Yusuke Sakai, Hidetaka Kamigaito, and Taro Watanabe. HalluCitation matters: Revealing the impact of hallucinated references with 300 hallucinated papers in ACL conferences. In *ACL*, pp. 47295–47376, 2026.

Yiwen Song, Yale Song, Tomas Pfister, and Jinsung Yoon. Paperorchestra: A multi-agent framework for automated ai research paper writing. In *COLM*, 2026.

Jiabin Tang, Lianghao Xia, Zhonghang Li, and Chao Huang. AI-Researcher: Autonomous scientific innovation. In *NeurIPS*, pp. 9481–9520, 2025.

William H Walters and Esther Isabelle Wilder. Fabrication and errors in the bibliographic citations generated by chatgpt. *Scientific Reports*, 13(1):14045, 2023.

Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. Cycleresearcher: Improving automated research via automated review. In *ICLR*, pp. 3669–3709, 2025.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Jiebin Zhang, Eugene J. Yu, Qinyu Chen, Chenhao Xiong, Dawei Zhu, Han Qian, Mingbo Song, Weimin Xiong, Xiaoguang Li, Qun Liu, and Sujian Li. WIKIGENBENCH:exploring full-length Wikipedia generation under real-world scenario. In *COLING*, pp. 5191–5210, 2025.

Minjun Zhu, Yixuan Weng, Linyi Yang, and Yue Zhang. DeepReview: Improving LLM-based paper review with human-like deep thinking process. In *ACL*, pp. 29330–29355, 2025.

APPENDIX

A BENCHMARK CONSTRUCTION DETAILS

A.1 DISTRIBUTION OF SOURCE PAPERS

Table A lists the venues the 72 papers of PaperWrite-Bench are drawn from. NLP and CV venues contribute equally and account for the majority, followed by ML venues. After excluding papers whose titles were recalled by any model in the title-recall test (Section A.5), the retained benchmark contains more papers from 2026 than from 2025.

A.2 CONSTRUCTION AND VERIFICATION OF THE RESEARCH OVERVIEW

The research overview serves as the primary source of information for paper reconstruction. We generate each overview from the original LaTeX source using DeepSeek-V4-Pro (Xu et al., 2026), following a fixed schema covering the motivation, problem setting, method, experimental setup, main results, and takeaways. We create two versions: a default overview used in our main experiments (528 words on average) and a more detailed overview (1,584 words) used to analyze the effect of input information in Table 4.

To ensure that the overviews contain the essential information needed for reconstruction, the original authors verify them against their papers and edit them when necessary. Among the 72 benchmark papers, 65 overviews required no changes, while 7 were edited.

A.3 RESOURCES PROVIDED TO THE WRITING AGENT

The resources are extracted from each paper's arXiv source. Tables and figures are stored in `tables/` and `figures/`, with their paths and brief caption-based descriptions provided in `table_summary.txt` and `figure_summary.txt`. Each bibliography entry is augmented with the corresponding abstract retrieved through the Semantic Scholar API. We also include the paper's codebase when publicly available, while manually removing any text in `README.md` that directly reproduces parts of the original paper, such as the abstract or introduction.

A.4 PREPARATION OF THE PROVIDED MATERIALS

Before providing the extracted materials to the writing agents, we apply four additional preprocessing steps.

Table A: Distribution of the 72 benchmark papers by research area and venue.

Area	Conference	# Papers	Subtotal
NLP	ACL26	16	27
	COLM26	9	
	ACL25	2	
CV	CVPR26	9	27
	ECCV26	7	
	CVPR25	6	
	ACMMM25	4	
	ICCV25	1	
ML	ICLR26	8	18
	ICML26	6	
	NeurIPS25	2	
	ICLR25	1	
	ICML25	1	
Total		72	72

Anonymizing the Template. Since `template.tex` is derived from the original LaTeX source, it may contain author information. We replace all author blocks with *Anonymous Author* to prevent author names from serving as cues for identifying the original paper. This encourages agents to reconstruct the paper from the provided materials rather than from prior knowledge. We handle different author formats across venues, including multiple `\author` commands and multi-line author blocks.

Adding an AI-Generated Watermark. We add a watermark to every template indicating that the resulting paper is AI-generated. The watermark is included directly in the template so that it is preserved throughout the writing and compilation process. This ensures that papers generated from PaperWrite-Bench remain identifiable as AI-generated even when distributed separately from our evaluation pipeline.

Removing Paper Content from Codebases. Public code repositories often contain text from the corresponding paper in `README.md`, including the abstract, introduction, or contribution summary. Providing such text would expose agents to the original writing for sections that we aim to evaluate. We therefore manually inspect every `README.md` in the 69 included codebases and remove passages that reproduce content from the original paper, while retaining information necessary to understand and use the codebase, such as installation instructions, usage instructions, and repository structure.

Rasterizing Figures. Approximately 60% of the figures extracted from the arXiv sources are vector PDFs, which cannot be directly viewed by the writing agents. We therefore convert all PDF figures to raster images, crop them, and render them at 150 dpi. We also update the corresponding file paths in `figure_summary.txt` and the template. As a result, all figures provided to the agents are directly viewable. This conversion does not affect figure matching during evaluation, which is based on file stems rather than file extensions.

A.5 Filtering for Memorized Papers

A model that has read a paper during pretraining could reconstruct it from memory rather than from the provided resources. We therefore filter candidates with two tests that fail in different ways. The first asks the model to name a paper it is given the authors and venue of, and excludes every paper at least one model names; this removed 28 of the 100 candidates. The second test asks the model to continue a sentence from a paper's abstract. We apply this test to the 72 papers retained after the first test and exclude no additional papers.

A.5.1 Title Recall from Authors and Venue

Setup. For each candidate paper, we retrieve its authors from arXiv and ask the model, without tools, to identify the paper's exact title from the authors and venue:

```
The following paper was accepted at {venue}.
Authors: {authors}
What is the exact title of this paper? Answer from your own
knowledge only. Reply with ONLY the title on a single line
--- no quotes, no explanation. If you are not sure, reply
with your best guess anyway.
```

We normalize the predicted and true titles and compute their character-level similarity using Python `difflib`. A prediction is considered recalled if it matches exactly after normalization or has a similarity of at least 0.9. We manually inspect predictions with similarities between 0.8 and 0.9 and count all five cases as recalls, as they preserve the distinctive paper title with only minor wording differences or omitted subtitles. We filter papers using GPT-5.4-mini, GPT-5.6-luna, GPT-5.6-terra, Claude Sonnet 4.6, Claude Sonnet 5, and Claude Opus 5. Claude Opus 5 is included in the filtering procedure but is not evaluated as a writing model and it also serves as the second-stage verifier.‘.

Results. Table B shows clear differences across models on the 100 candidate papers. Claude Opus 5, Sonnet 5, and Sonnet 4.6 recall 28, 12, and 1 papers, respectively, while the GPT models recall none. All papers recalled by Sonnet models are also recalled by Opus 5. We exclude all 28 recalled papers, leaving 72 papers for PaperWrite-Bench. Under this title-recall protocol, none of the evaluated models recalls any paper title.

Table B: **Title recall from authors and venue.** Recalled: number of correctly recalled titles based on our similarity and manual-inspection criteria. Sim.: mean character-level similarity to the true title. Candidates: 100 papers before filtering; Benchmark: 72 papers after excluding all recalled candidates.

	Candidates (100)		Benchmark (72)	
Model	Recalled	Sim.	Recalled	Sim.
GPT-5.4-mini	0	0.26	0	0.25
GPT-5.6-luna	0	0.29	0	0.29
GPT-5.6-terra	0	0.30	0	0.29
Claude Sonnet 4.6	1	0.31	0	0.29
Claude Sonnet 5	12	0.42	0	0.33
Claude Opus 5 (filtering only)	**28**	0.50	0	0.31

Table C: **Sentence completion from the abstract.** Sim.: word-level similarity between the completion and the true continuation. Shuffled: the same completion scored against a different paper's continuation. Rare recall: share of the continuation's distinctive tokens reproduced. Rare $= 0$: papers reproducing none of them, over the 59 papers whose continuation carries one. Verbatim: completions reaching a similarity of 0.8.

Model	Sim.	Shuffled	Rare recall	Rare $= 0$	Verbatim
GPT-5.4-mini	0.284	0.232	0.041	55/59	0/72
GPT-5.6-luna	0.282	0.220	0.064	55/59	0/72
GPT-5.6-terra	0.276	0.204	0.055	54/59	0/72
Claude Sonnet 4.6	0.292	0.225	0.095	53/59	0/72
Claude Sonnet 5	0.295	0.225	0.069	54/59	1/72

A.5.2 Sentence Completion from the Abstract

Setup. Since the title-recall test can be passed simply by declining to guess, we introduce a second test based on sentence completion. For each benchmark paper, we select a sentence from the original abstract and split it so that the continuation contains as many *distinctive tokens* as possible, defined as numbers, dataset names, or method names. The model is then given the prefix and asked to complete it from memory without using tools.

Metrics. We report word-level similarity to the ground-truth continuation and *rare-token recall*, the fraction of distinctive tokens in the continuation that are reproduced by the model but not revealed in the prefix. Since similarity can be high even for fluent completions without memorization, we additionally compare it against a shuffled reference, where each completion is matched with another paper's continuation. Rare-token recall provides a stronger signal of memorization by measuring the recovery of specific numbers, dataset names, and method names.

Results. As shown in Table C, rare-token recall remains low (0.04–0.10) across all five evaluated models. For 53–55 of the 59 applicable papers, no distinctive tokens are reproduced, while the few non-zero cases primarily involve field-standard terms predictable from context rather than paper-specific information. Only one of 360 completions reaches a similarity of 0.8; further testing on eight sentences from the same paper finds no evidence of memorization. Similarity is only 0.05–0.07 higher than the shuffled reference across models, consistent with fluent but non-memorized completions. Together with the title-recall test, these results suggest that reconstruction relies on the provided resources rather than prior knowledge of the benchmark papers.

Table D: **Rubric items vs. one holistic score, from the same judge.** Both are compared with the human reviewers' pairwise judgments on 48 pairs of generated papers. Agreement is over the 46 pairs the reviewers did not judge a tie; a tied pair is one the method scores equally and so cannot rank.

	Rubric (ours)	**Holistic (1–10)**
Kendall's τ_b ↑	**0.45**	0.36
Agreement with reviewers ↑	**74%** (34/46)	54% (25/46)
Tied pairs ↓	**0%** (0/48)	27% (13/48)

Table E: **Interpretation of rubric scores.** The same judge and prompt evaluate three types of text. The ceiling scores each original paper against its own rubric, while the floor scores it against another paper's rubric. All rows include only text-based rubric items; therefore, the mean score for generated papers differs slightly from Table 1, which also includes figure and table items.

					Share of items	
	Text scored	Rubric used	Items	Mean	5	1
Ceiling	Original paper	Its own	4,760	4.97	98.7%	0.4%
Floor	Original paper	Another paper's	1,273	1.03	0.0%	97.5%
Table 1	Generated papers	Its own	23,772	3.51	32.1%	11.6%

B Additional Evaluation Analyses

B.1 Rubric-Based vs. Holistic Scoring

We compare our rubric-based evaluation with a simpler holistic approach that assigns a single score to each paper. Using the same DeepSeek-V4-Pro judge, we score each paper in the human study on a 1–10 scale based on faithfulness to the original paper, overall paper quality, and correct use of figures and tables. The holistic judge is thus given a finer scale than the 1–5 scale used for each rubric item. We then compare both evaluation methods with the human pairwise judgments.

As shown in Table D, holistic scores are concentrated in a narrow range, with 23 of the 32 generated papers receiving scores between 6 and 8, and 27% of paper pairs receiving identical scores. Among pairs with a clear human preference, our rubric agrees with human judgments in 74% of cases, compared with 54% for holistic scoring, and achieves a higher correlation ($\tau_b = 0.45$ vs. 0.36). These results suggest that decomposing evaluation into concrete rubric items better distinguishes papers with similar overall quality than assigning a single holistic score.

B.2 Ceiling and Floor of the Rubric Score

A rubric score is meaningful only when its scale is properly anchored. We therefore evaluate the original papers using the same judge and prompt as in our main evaluation (Section 4.2.2).

Ceiling. We score each original paper against its own rubric, providing an empirical ceiling for our evaluation. Across 72 papers and 4,760 rubric items, the original papers achieve an average score of 4.97 out of 5, with 98.7% of items receiving a score of 5. The lowest-scoring section is Benchmark Construction at 4.80, while all other sections score at least 4.95.

Floor. We score each original paper against the rubric of a different paper for the same section, providing an empirical floor for our evaluation. Across 20 papers and 1,273 rubric items, the average score is 1.03, with 97.5% of items receiving a score of 1 and none receiving a score of 5.

Table E compares these bounds with the scores of generated papers. The generated papers fall between the empirical floor and ceiling, indicating that the differences in Table 1 reflect differences in how faithfully they recover the original content rather than a tendency of the judge to assign similar scores across papers.

C MATERIALS GIVEN TO THE WRITING AGENT

C.1 EXAMPLE OF RESEARCH_OVERVIEW.MD

The overview is the primary prose summary of the paper the writing agent receives. Below are both versions for one benchmark paper: the default overview used throughout the paper (650 words here) and the long one compared in Table 4 (1,844 words), generated from the same source. The long version keeps the same structure and adds detail on the method and the experimental design.

DEFAULT VERSION

```
# Delay, Plateau, or Collapse: Research Overview

## Title
**Delay, Plateau, or Collapse: Evaluating the Impact of Systematic Verification Error on RLVR**
---

## 1. Motivation
Prior RLVR analyses model verification errors as random, independent label flips and conclude they only delay
      training. However, real verifiers such as static code checkers and LLM-as-a-judge exhibit systematic,
      exploitable error patterns not captured by FPR/FNR. This gap motivates controlled experiments isolating
      systematic false positives and false negatives.
---

## 2. Key Insight
Systematic false negatives resemble random noise and mainly delay training. Systematic false positives can
      cause suboptimal plateaus or complete collapse even at low initial FPR, because the model learns trigger
       patterns.
Key idea: Training outcome under systematic verification errors is governed by the error pattern and its
      trigger behavior, not aggregate error rate.
---

## 3. Benchmark Design: Delay, Plateau, or Collapse
We use the decimal chain sum task from Reasoning Gym, with multi-term floating-point arithmetic queries such
      as "332.419 - 993.538 + 740.756 = ?". Training uses DAPO and 256 rollouts per step (4 per query) for one
       epoch, on Qwen3-1.7B-Base and OLMo3-7B. We insert controlled error patterns into an otherwise ground-
      truth verifier: random noise at FPR/FNR levels of 20% and 50%; format-based FN rejecting correct answers
       containing a backslash; language-based FN rejecting English answers; relative-error FP accepting
      incorrect answers within a threshold; and word-based FP rewarding any output containing "Certainly" or "
      python". Evaluation tracks oracle reward R<sup>*</sup>, verifier reward R, FPR and FNR directly from
      training rollouts under a tractable ground-truth verifier. We also report Dr. GRPO and SAPO, plus
      instruction-tuned Qwen2.5-1.5B-Instruct results.
---

## 4. Data Construction Pipeline
- Source: use arithmetic examples from the decimal chain sum task of Reasoning Gym, consisting of multi-term
      floating-point arithmetic queries.
- Collection: at each training step, sample 256 rollouts (4 per query) from the current policy; train for
      exactly one epoch to avoid contamination of reward/FPR/FNR estimates.
- Verifier construction: replace the ground-truth verifier with controlled noisy verifiers encoding five error
       patterns: random label flips at 20%/50%, format-based FN (backslash token), language-based FN (English)
      , relative-error FP (threshold), and word-based FP ("Certainly", "python"); length-based FP is also
      studied in the appendix.
- Annotation: each rollout receives a binary reward (0/1) from the imperfect verifier; the ground-truth
      verifier labels the same rollout to compute oracle reward R<sup>*</sup>, FPR and FNR.
- Quality control: since the arithmetic ground truth is tractable, oracle labels are exact; metrics are
      computed from the same 256 rollouts used for training, and global error rates plus reward curves are
      monitored over steps.
---

## 5. Key Findings
- On Qwen3-1.7B-Base and OLMo3-7B, clean ideal training improves oracle reward consistently; random noise at
      FPR 20% and FNR 50%, and format-based FN, delay training but reach similar final performance.
- Relative-error FP drives FPR to nearly 1 by about step 50 and converges to a suboptimal plateau; word-based
      FP with "Certainly" plateaus, while "python" collapses to near-zero reward.
- Asymmetric relative-error FP, despite a strictly smaller false-positive region and lower initial FPR than
      the symmetric interval, produces worse training outcomes, especially for OLMo.
- Alternating with the ground-truth verifier once every 10 steps mitigates plateauing FPs, but "python"-
      triggered collapse is only prevented at once every 2 steps; FPR still rises to 1.
---

## 6. Contributions
- Analyze systematic verification errors in RLVR and categorize them into Ideal, Delayed, Plateau, and
      Collapse dynamics.
- Design controlled experiments introducing random, format-based, language-based, relative-error, and word-
      based verifier error patterns.
- Demonstrate that systematic FNs and random noise delay training, while systematic FPs induce plateau or
      collapse.
- Show that initial error rate is insufficient: asymmetric FPs with lower FPR can collapse while symmetric FPs
       with higher error rate only plateau.
---

## 7. Takeaway
Systematic verifier errors must be characterized by their trigger patterns and training dynamics rather than
      aggregate sample-level error rates, because otherwise low-error verifiers can still cause RLVR collapse.
```

LONG VERSION

```
# Delay, Plateau, or Collapse: Research Overview

## Title
**Delay, Plateau, or Collapse: Evaluating the Impact of Systematic Verification Error on RLVR**
---

## 1. Motivation
Reinforcement Learning with Verifiable Rewards (RLVR) trains large language models (LLMs) by rewarding sampled
      outputs that a verifier judges correct, and it has driven advances in mathematical reasoning and code
      generation. Real verifiers, however, are imperfect: they can accept incorrect outputs as false positives
      (FP) or reject correct outputs as false negatives (FN), especially as tasks grow complex and accurate
      verification becomes expensive.

Prior analyses generally model verifier errors as random, independent label flipping and conclude that noise
      mostly delays training without changing the final outcome; one line of work argues RLVR tolerates up to
      40% randomly flipped labels, while a theoretical account predicts collapse only when Youden's J, TPR
      minus FPR, is negative. Yet practical verifiers such as static checkers, unit tests, and LLM-as-a-judge
      systems often exhibit systematic, exploitable error patterns rather than independent coin flips.

Systematic errors are riskier because models can learn to trigger false-positive patterns or suppress false-
      negative ones, making FPR and FNR drift during training. Existing studies do not isolate which
      systematic patterns cause delayed learning, plateaus, or collapse. This paper therefore builds a
      controlled RLVR framework to characterize these dynamics.
---

## 2. Key Insight
The central design choice is to study verification errors as trigger patterns injected into an otherwise
      ground-truth verifier, rather than as random label flips. A systematic error is defined as a function of
      both the query and the model output, so its FPR and FNR can change as the model learns to emit or avoid
      trigger-associated behavior. The framework restricts itself to arithmetic, where the ground-truth
      verifier V* is tractable, allowing the oracle reward to be separated from the flawed verifier reward and
      tracked throughout training. This separation makes it possible to tie delayed, plateaued, and collapsed
      dynamics to specific error patterns instead of aggregate sample-level error rates.

Key idea: The training outcome under a systematic verifier error is governed by whether the model can learn
      the associated trigger behavior and how well that behavior aligns with the oracle reward, not by the
      verifier's overall error rate.
---

## 3. Benchmark Design: Delay, Plateau, or Collapse

### 3.1 Overview
- Scope: a controlled RLVR evaluation environment on the decimal chain sum task from the Reasoning Gym library
      , designed to isolate delayed, plateaued, and collapsed training dynamics under injected verifier errors
      ; it is a study framework rather than a static named benchmark.
- Scale: main results use Qwen3-1.7B-Base and OLMo3-7B; supplementary analyses add Qwen2.5-1.5B-Instruct and
      OLMo3-32B, with 256 rollouts (4 per query) generated at each training step for one epoch.
- Goal: characterize systematic false positives and systematic false negatives, and test whether prior random-
      noise conclusions transfer to exploitable verifier errors.
- Key capability: the ground-truth verifier is tractable, so oracle reward, verifier reward, FPR, and FNR can
      be computed per step alongside training.

### 3.2 Data Collection & Curation
- Data source and selection criteria: the study reuses the decimal chain sum task from Reasoning Gym, a multi-
      term floating-point arithmetic environment with queries such as "332.419 - 993.538 + 740.756 = ?". It
      was chosen because clean RLVR training improves performance substantially and because the ground-truth
      verifier V* is tractable, enabling precise measurement.
- Collection process: at every training step, the current policy generates 256 rollouts, corresponding to 4
      rollouts per query; these same rollouts are used directly as unbiased estimates of reward, FPR, and FNR,
      avoiding extra evaluation outputs that would reduce the number of training steps under a fixed compute
      budget.
- Filtering and cleaning: no additional filtering of arithmetic examples is reported; training proceeds for
      exactly one epoch so that metric estimates are not contaminated by revisiting the same examples.
- Annotation protocol: not applicable -- outputs are scored by a ground-truth arithmetic comparison or
      programmatic noise rules, not by human annotators; the task uses binary rewards with values in {0,1}.
- Number and qualification of annotators: not applicable; no human annotators were used.
- Inter-annotator agreement: not applicable; agreement statistics are not reported because labels come from
      automated verifiers.
- Quality control: systematic error patterns are injected by deterministic rules; random noise flips rewards
      at FPR and FNR levels of 20% and 50%; format-based false negatives reject a correct answer containing a
      backslash-opening-square-bracket token common under clean training; language-based false negatives
      reject English outputs using the langdetect library; relative error-based false positives accept
      incorrect answers below a fixed relative-error threshold; word-based false positives accept any output
      containing a predefined keyword such as "Certainly" or "python". Additional variants include length-
      based false positives, asymmetric relative-error false positives, and systematic but non-deterministic
      FP injection at probabilities 80% and 50%.

### 3.3 Task Definition & Evaluation Protocol
- Task: decimal chain sum -- given a multi-term floating-point arithmetic query such as "332.419 - 993.538 +
      740.756 = ?", the model must produce a final numerical answer; quality is measured against ground truth.
- Input/output format: the verifier takes a query and a sampled model output and returns a binary reward in
      {0,1}; training uses DAPO as the default algorithm in TRL, with Dr. GRPO and SAPO reported as robustness
      variants.
- Metrics: oracle reward R*(M) (average ground-truth verifier reward), verifier reward R(M) (the reward the
      model optimizes), FPR, and FNR; conditional advantage C(t) averages the oracle advantage over outputs
      containing a trigger pattern t.
- Splits: no separate train/test split is reported; main runs train Qwen3-1.7B-Base and OLMo3-7B for one epoch
      , with 500 training steps in the main setting and 250 steps for OLMo3-32B due to resource constraints.

### 3.4 Taxonomy & Coverage
```

```
- Error-type coverage: random noise; systematic false negatives (format-based backslash-bracket token;
      language-based English rejection); systematic false positives (relative error-based, word-based keywords
      , length-based, and asymmetric relative-error regions).
- Training-dynamics taxonomy: Ideal (FPR and FNR remain close to 0), Delayed (eventually reaches ideal
      performance), Plateau (converges to a suboptimal oracle reward), and Collapse (oracle reward decreases;
      complete collapse when near random guessing).
- Model and algorithm coverage: DAPO plus Dr. GRPO and SAPO across Qwen3-1.7B-Base, OLMo3-7B, Qwen2.5-1.5B-
      Instruct, and OLMo3-32B.
---

## 4. Dataset Statistics
- Rollouts: 256 rollouts are sampled per training step, with 4 rollouts per query; the total number of unique
      queries is not reported.
- Training length: one epoch of 500 steps in the main runs; OLMo3-32B uses 250 steps due to resource
      constraints.
- Injected noise levels: random-noise FPR and FNR are set to 20% and 50%; systematic trigger thresholds and
      keyword lists are described qualitatively, and non-deterministic FP injection uses 100%, 80%, and 50%
      probabilities.
- Annotators: not applicable; no human-derived statistics or agreement scores are reported.
- Comparison to prior benchmarks: unlike random-noise evaluations, this framework tracks FPR and FNR
      dynamically because the model can learn to trigger or avoid systematic error patterns, and it provides
      oracle-reward curves rather than a fixed score table.
---

## 5. Baseline Evaluation

### 5.1 Evaluated Methods
- Base models and main algorithm: Qwen3-1.7B-Base and OLMo3-7B trained with DAPO, the TRL default.
- Algorithm variants: Dr. GRPO and SAPO are run to test robustness of qualitative conclusions.
- Supplementary models: Qwen2.5-1.5B-Instruct and OLMo3-32B.
- Verifier settings: ideal ground-truth verifier, random noise, format-/language-based FN, and relative-error,
       word-based, length-based, and asymmetric FP verifiers.

### 5.2 Key Findings
- False-negative settings (random noise and systematic FN):
  - Random noise: FPR and FNR at 20% and 50% delay training but do not significantly affect final performance.
  - Systematic FN: format-based FN delays training as the model unlearns the trigger; language-based FN starts
        with global FNR close to 1, then declines quickly, producing a more pronounced delay.
- Systematic false positives (relative error-based and word-based):
  - Relative error-based FP: the oracle reward plateaus at a suboptimal level and the FPR rises to nearly 1 by
        about step 50.
  - Word-based FP: the "Certainly" trigger produces a suboptimal plateau, while the "python" trigger causes
        training collapse to near-zero reward.
- Asymmetric relative-error FP:
  - An asymmetric false-positive interval that is strictly smaller than the symmetric interval produces worse
        outcomes than the symmetric case, including collapse, despite a lower initial FPR.
- Remaining baseline families: Dr. GRPO and SAPO yield similar qualitative dynamics to DAPO, and multi-seed
      runs show the same clean improvement, plateau, and collapse patterns.
---

## 6. Analysis & Insights
- Ablation: The paper reports no single numeric ablation table; it instead varies error type, trigger,
      algorithm (DAPO, Dr. GRPO, SAPO), seed, and model scale (OLMo3-32B), finding the same qualitative
      delayed/plateaued/collapsed dynamics across all variations.
- Sensitivity: Reducing non-deterministic FP injection probability for the "python" trigger from 100% to 80%
      causes training to be substantially delayed but no longer fully collapse, while 50% makes the curve
      close to ideal with only a slight delay; for plateauing FP patterns, dynamics improve as the error rate
      decreases.
- Error analysis: Conditional advantage C(t) separates outcomes for word-based false positives: frequent
      trigger patterns with negative conditional advantage collapse, frequent ones with positive conditional
      advantage plateau, rare positive-advantage patterns are amplified, and rare negative-advantage patterns
      fade; trigrams that remain below 50% FPR are highlighted as exceptions in the distribution plot.
- Qualitative: Output inspection shows "Certainly"-triggered runs insert the keyword into otherwise correct
      arithmetic reasoning, whereas "python"-triggered runs emit Python-like code with an arbitrary answer in
      an output block, for example final result -33.054; under language-based FN, Qwen answers in Chinese
      while OLMo exploits langdetect by inserting repetitive English words.
---

## 7. Contributions
- Analyze systematic verification errors in RLVR and categorize them according to their effect on training
      dynamics: delayed training, suboptimal plateaus, and training collapse.
- Design controlled experiments on the Reasoning Gym decimal chain sum task that inject predictable false-
      positive and false-negative trigger patterns into an otherwise ground-truth verifier.
- Demonstrate that initial verifier error rates are insufficient to predict training outcomes; a tighter
      asymmetric false-positive interval can cause collapse while its symmetric, strictly higher-error
      counterpart only plateaus.
---

## 8. Limitations & Future Work
- The study is limited to a controlled arithmetic setting with manually specified error patterns, which
      isolates effects but does not capture real verifiers with multiple interacting failure modes.
- Practical RLVR consumes more complex reward signals, and real verifier errors can depend on both the input
      and the output rather than on a single fixed trigger.
- Future work should test whether the observed delay, plateau, and collapse phenomena transfer to code
      generation, formal reasoning, and rubric-based rewards.
- Future work should develop diagnostics and mitigation strategies that remain effective without full ground-
      truth oracle access.
---

## 9. Takeaway
Systematic verification errors in RLVR cannot be treated as random noise or summarized by aggregate error
      rates. The same training pipeline can be merely delayed, can settle at a suboptimal plateau, or can
```

```
collapse completely depending on the specific trigger pattern, even when the initial false-positive rate
 is low. This matters because RLVR is increasingly used with imperfect verifiers such as static
analyzers and LLM judges; practitioners must characterize a verifier's error structure, not only its
overall accuracy, before trusting it for training.
```

C.2 Example of an Evaluation Rubric

The rubric of a paper lists, per section, the elements a faithful reconstruction has to contain, each with an importance and a description of what the original says. Table F shows a sample from one paper's rubric (73 items over five sections: 13 Abstract, 14 Introduction, 15 Related Work, 16 Method, 15 Experiment). The rubric evaluation scores the generated section against every such element on the 1–5 scale of Section 4.2.2.

Table F: **Rubric items.** A sample of the rubric built for one benchmark paper, with the element and the description the judge is given.

Section	Import.	Element and description
Abstract	medium	RLVR improves reasoning capabilities of LLMs Reinforcement Learning with Verifiable Rewards is introduced as a powerful approach for improving the reasoning capabilities of large language models.
Abstract	medium	RLVR is designed for verifiable ground-truth tasks RLVR is designed for tasks with verifiable ground-truth answers, establishing the setting in which reward verification occurs.
Introduction	high	Problem motivation: complex tasks make accurate verifiers challenging As LLMs are deployed on increasingly complex tasks, designing accurate verifiers becomes difficult and the reward signal may become erroneous.
Introduction	high	False positive and false negative verifier errors Verifier errors are classified into accepting incorrect solutions (false positives, FP) and rejecting correct ones (false negatives, FN).
Related Work	high	RLVR definition and primary application domains Identifies RLVR as a widely used post-training technique for improving LLM reasoning, applied especially where verification is simple, such as mathematics with final answers, formal proofs, and code generation.
Method	high	Section scope The section defines systematic errors in RLVR and characterizes the downstream training dynamics they induce.
Method	high	Verifier function definition A verifier V maps a query x and model output y to a binary reward r = V(x,y), with r in {0,1}.
Method	high	Ground-truth verifier and imperfect verifier V* denotes the hypothetical ground-truth verifier producing correct rewards r*, but it is often too expensive or infeasible and is replaced by a cheaper imperfect verifier V.
Experiment	high	Task environment The experiments use the decimal chain sum task from Reasoning Gym, a multi-term floating-point arithmetic environment chosen so verifier error patterns can be precisely controlled and tracked.
Experiment	high	RL training algorithm Main training uses DAPO, chosen because it slightly improves over GRPO and is the TRL default; results with Dr. GRPO and SAPO are similar.
Experiment	high	Models Two base models from different families are trained: Qwen3-1.7B-Base and OLMo3-7B.

C.3 Instructions Given to the Writing Agent

Every task directory contains an `AGENTS.md` (copied to `CLAUDE.md` for Claude Code) that tells the agent what kind of paper it is writing and what each section should contain. One of three variants is used, chosen by the type of the original paper: a method paper, a benchmark or evaluation paper, or one contributing both. They differ only in the description of the contribution and in the section-by-section guidance; the conference rules and the LaTeX constraints are the same in all three. The three files are reproduced verbatim below.

Method papers (`AGENTS_METHOD.MD`)

```
You are an ambitious AI researcher who is looking to publish a paper that will contribute significantly to the
       field.

This paper is a method paper. The main contribution is a novel technique, algorithm, model, or framework that
      addresses a specific research problem.

Ensure that the paper is scientifically accurate, objective, and truthful. Accurately report the experimental
      results, even if they are negative or inconclusive.

You are planning to submit to a top-tier ML conference, which has guidelines:

- In general, try to use the available space and include all relevant information.
- Do not change the overall style which is mandated by the conference. Keep to the current method of including
       the references.bib file.
- Do not remove the \graphicspath directive or no figures will be found.
- Do not add 'Acknowledgements' section to the paper.
- Use a single backslash (\) for LaTeX commands instead of a double backslash (\\).

Here are some tips for each section of the paper:

- **Title**:
  - Title should be catchy and informative. It should give a good idea of what the paper is about.
  - Try to keep it under 2 lines.

- **Abstract**:
  - TL;DR of the paper.
  - What are we trying to do and why is it relevant?
  - Briefly describe the proposed method and highlight key empirical findings.
  - Make sure the abstract reads smoothly and is well-motivated. This should be one continuous paragraph.
  - Keep the abstract to around 200 words.
  - Do not include formatted mathematical formulas or equations.

- **Introduction**:
  - Provide context to the study and explain its relevance.
  - Highlight how the approach effectively addresses the research question or problem. Also, provide an
       intuitive explanation of why this method works well.
  - Summarize your contributions, highlighting pertinent findings, insights, or proposed methods.
  - If there is a figure of the method overview, place it on page 2 of the paper.

- **Related Work**:
  - Discuss alternative attempts in literature at trying to address the same or similar problems.
  - Compare and contrast their approach with yours, noting key differences or similarities.
  - Ensure proper citations are provided.
  - Each subsection should help clarify what gap remains in prior work and how this work addresses it.

- **Method**:
  - Clearly detail what you propose to do and why.
  - If your study aims to address certain hypotheses, describe them and how your method is constructed to test
       them.
  - Include technical details sufficient for reproduction.
  - Do not include results in the Method section.

- **Experiments**:
  - Explain how you tested your method or hypothesis.
  - Describe necessary details such as data, environment, and baselines, but omit hardware details unless
       explicitly mentioned.
  - Present the results truthfully according to the data you have. If outcomes are not as expected, discuss it
       transparently.
  - Include comparisons to baselines if available, and only include analyses supported by genuine data.
  - Try to include all relevant plots and tables. Consider combining multiple plots into one figure if they
       are related.
  - In tables, please bold the best result among the compared methods. There is no need to bold the proposed
       method if it is not the best.

- **Analysis**:
  - Provide deeper insights into why the proposed method works or fails in certain scenarios.
  - Include ablation studies to demonstrate the contribution of each component.
  - Analyze the results from multiple perspectives (e.g., per-class performance, computational cost, results
       with other backbones).
  - Use visualizations (e.g., t-SNE, attention maps, feature distributions) to support the analysis when
       appropriate.
  - Ensure all claims are supported by data from the experiments.

- **Conclusion**:
  - Summarize the entire paper, including key strengths or findings.
  - Highlight how the results address the research problem.
```

```
Ensure you are always writing good compilable LaTeX code. Common mistakes that should be fixed include:

- LaTeX syntax errors (unenclosed math, unmatched braces, etc.).
- Duplicate figure labels or references.
- Unescaped special characters: & % $ # _ {{ }} ~ ^ \\
- Proper table/figure closure.
- Do not hallucinate new citations or any results not in the logs.

Ensure proper citation usage:

- Always include references within \\begin{{filecontents}}{{references.bib}} ... \\end{{filecontents}}, even
      if they haven't changed from the previous round.
- Before citing any paper, ALWAYS read the references.bib file first to find the correct citation key. Do NOT
      create new fictional bibtex entries.
- Do not make any changes to reference.bib
- Verify all citation keys match exactly with those in references.bib before using them in the text.
- Each section, especially Related Work, should have multiple citations.

## Notes

- The paper should make clear what problem is being solved and why the proposed method is effective.
- Do not include formatted mathematical formulas or equations in the Abstract. However, including key
      numerical results (e.g., "an accuracy of 92.5%") and mentioning mathematical concepts by name is
      perfectly acceptable.
- Do not include Limitation sections and Future Work sections.
- Use `\%` to display a literal percent sign, as a standard `%` will be treated as a comment.
```

BENCHMARK PAPERS (AGENTS_BENCHMARK.MD)

```
You are an ambitious AI researcher who is looking to publish a paper that will contribute significantly to the
       field.

This paper is a benchmark / evaluation framework / evaluation protocol paper, not a method paper. The main
      contribution is the design, construction, validation, and analysis of a benchmark, dataset, task
      formulation, metric, or evaluation protocol that enables more reliable and informative assessment of
      machine learning systems.

Ensure that the paper is scientifically accurate, objective, and truthful. Accurately report the experimental
      results, even if they are negative, mixed, or inconclusive. Do not overclaim that the benchmark solves
      the entire problem; instead, clearly state what it measures, what it does not measure, and what new
      insights it enables.

You are planning to submit to a top-tier ML conference, which has guidelines:

- In general, try to use the available space and include all relevant information.
- Do not change the overall style which is mandated by the conference. Keep to the current method of including
       the references.bib file.
- Do not remove the \graphicspath directive or no figures will be found.
- Do not add `Acknowledgements` section to the paper.
- Use a single backslash (\) for LaTeX commands instead of a double backslash (\\).

The paper should be written as a benchmark / evaluation paper. Do not force the structure or claims of a
      method paper onto it. In particular:

- Do not frame the benchmark itself as if it were a proposed learning algorithm.
- Do not include a "Method" section unless the benchmark genuinely includes a technical construction procedure
       that needs its own section. Prefer section names such as "Benchmark Design", "Task Construction", "Data
       Collection", "Evaluation Protocol", "Metrics", or "Experimental Setup" where appropriate.
- Do not claim superiority in the style of a method paper. The contribution is the quality, usefulness, rigor,
       and insightfulness of the evaluation framework.
- The central question is whether the benchmark or evaluation protocol is well-motivated, well-constructed,
      diverse, reliable, challenging, reproducible, and informative.

Here are some tips for each section of the paper:

- **Title**:
  - Title should be catchy and informative. It should make clear that the paper introduces, studies, or
       validates a benchmark, evaluation framework, dataset, task, or protocol.
  - Avoid method-style titles that imply a new model or algorithm unless that is truly part of the
       contribution.
  - Try to keep it under 2 lines.

- **Abstract**:
  - TL;DR of the paper.
  - State what evaluation gap, benchmark deficiency, or measurement problem the paper addresses, and why it
       matters.
  - Briefly describe what is introduced (e.g., a benchmark, protocol, metric, dataset, or suite of tasks), how
       it is constructed, and what key empirical findings it reveals about current systems.
  - Emphasize what new insight or more reliable evaluation this benchmark enables.
  - This should be one continuous paragraph.
  - Keep the abstract to around 200 words.
  - Do not include formatted mathematical formulas or equations.

- **Introduction**:
  - Provide context for the evaluation problem and explain why existing benchmarks, datasets, metrics, or
       protocols are insufficient.
  - Clearly motivate the need for the new benchmark or evaluation setup.
  - Explain what the benchmark is intended to measure, and why that measurement is important to the field.
```

```
  - Summarize the design principles behind the benchmark: coverage, realism, difficulty, diversity, fairness,
      contamination resistance, reproducibility, diagnostic value, or other relevant goals.
  - Summarize the main empirical findings obtained by evaluating representative systems on the benchmark.
  - Clearly state the contributions. Typical contributions may include:
    - a new benchmark or evaluation framework,
    - a new task or taxonomy,
    - a new protocol or metric,
    - a careful validation of the benchmark properties,
    - empirical findings about the strengths and weaknesses of current systems.
  - If there is a benchmark overview figure, place it on page 2 of the paper.

- **Related Work**:
  - Discuss prior benchmarks, datasets, tasks, evaluation protocols, and metrics relevant to the same
      capability or problem setting.
  - Compare and contrast your benchmark with prior work in terms of scope, realism, diversity, difficulty,
      annotation procedure, contamination controls, metric choice, or diagnostic granularity.
  - Situate the paper among both benchmark papers and method papers that are commonly evaluated on similar
      settings when appropriate.
  - Ensure proper citations are provided.
  - Each subsection should help clarify what gap remains in prior evaluation practice and how this work
      addresses it.

- **Benchmark Design / Task Construction / Evaluation Protocol**:
  - Clearly describe what is being introduced.
  - Explain the target capability, task definition, input-output format, evaluation setting, and any
      assumptions.
  - If the paper includes dataset construction, describe the data sources, filtering, curation, annotation,
      and quality control pipeline.
  - If the paper includes a taxonomy, explain the taxonomy and why it is useful.
  - If the paper includes a metric or protocol, define it precisely and justify why it better measures the
      target capability.
  - Clearly explain the design rationale behind each major decision.
  - Describe known exclusions, scope boundaries, and intended use cases.
  - Avoid presenting this as a model contribution.

- **Benchmark Validation / Quality Checks**:
  - Include a dedicated section validating that the benchmark is meaningful and technically sound.
  - Depending on the paper, this may include:
    - annotation quality or inter-annotator agreement,
    - task validity,
    - distributional diversity,
    - difficulty calibration,
    - adversarial filtering or decontamination,
    - leakage checks,
    - robustness of the metric,
    - agreement with expert judgment,
    - sanity checks showing the benchmark is neither trivial nor broken.
  - Clearly report the evidence supporting the reliability of the benchmark.
  - If there are weaknesses, ambiguities, or known artifacts, state them explicitly.

- **Experiments / Benchmark Results**:
  - Explain how representative systems are evaluated on the benchmark.
  - Include strong and relevant baselines, representative model families, and, when appropriate, simple
      heuristic baselines.
  - Describe the evaluation setup in sufficient detail for reproducibility.
  - Present the benchmark results truthfully according to the data.
  - Focus on what the benchmark reveals about current systems, not on declaring a winning method unless that
      is genuinely supported and central.
  - Highlight major patterns such as overall difficulty, capability gaps, sensitivity to subcategories,
      brittleness, ranking instability, or disagreement with prior benchmarks.
  - If comparing to existing benchmarks, explain what additional insight your benchmark provides.
  - In tables, please bold the best result among the compared systems when appropriate. There is no need to
      imply that the best score means the benchmark is solved.

- **Analysis**:
  - Provide deeper analysis showing that the benchmark is diagnostic and insightful.
  - Analyze system behavior across slices, subcategories, difficulty levels, languages, domains, perturbations
      , or other relevant dimensions.
  - Include analyses that help readers understand what success and failure on the benchmark actually mean.
  - If useful, include human performance, expert performance, agreement analyses, calibration analyses, or
      metric sensitivity analyses.
  - Show whether the benchmark distinguishes systems in informative ways.
  - If there are surprising findings, discuss them carefully and ground them in the evidence.
  - Ensure all claims are supported by data.

- **Discussion**:
  - Discuss what the benchmark contributes to the field and how it should be used.
  - Clarify what conclusions can and cannot be drawn from it.
  - Explicitly discuss limitations such as coverage gaps, annotation subjectivity, potential contamination
      risks, remaining artifacts, or ecological validity constraints.
  - Avoid inflated claims. A strong benchmark paper is valuable even if it has limitations, as long as they
      are clearly stated.

- **Conclusion**:
  - Summarize the benchmark or evaluation contribution and the main empirical findings.
  - Emphasize the new measurement capability or insight enabled by the work.
  - Avoid method-paper language suggesting that a new algorithm was introduced if it was not.

Ensure you are always writing good compilable LaTeX code. Common mistakes that should be fixed include:

- LaTeX syntax errors (unenclosed math, unmatched braces, etc.).
- Duplicate figure labels or references.
```

```
- Unescaped special characters: & % $ # _ {{ }} ~ ^ \\
- Proper table/figure closure.
- Do not hallucinate new citations or any results not in the logs.

Ensure proper citation usage:

- Always include references within \\begin{{filecontents}}{{references.bib}} ... \\end{{filecontents}}, even
      if they haven't changed from the previous round.
- Before citing any paper, ALWAYS read the references.bib file first to find the correct citation key. Do NOT
      create new fictional bibtex entries.
- Do not make any changes to reference.bib
- Verify all citation keys match exactly with those in references.bib before using them in the text.
- Each section, especially Related Work, should have multiple citations.

## Notes

- The paper should make clear what evaluation problem is being solved.
- The benchmark must be justified, not merely presented.
- The paper should include evidence that the benchmark is valid, informative, and reproducible.
- The paper should clearly separate benchmark construction from benchmark findings.
- Do not include results in sections whose purpose is only to define the benchmark, unless a brief preview is
      essential for flow.
- Do not include Limitation sections and Future Work sections unless the venue explicitly requires them.
      However, benchmark limitations should still be discussed naturally in the Discussion or other
      appropriate sections.
- Use \% to display a literal percent sign, as a standard % will be treated as a comment.
- Do not overclaim that high benchmark performance implies general intelligence, real-world safety, or broad
      robustness unless this is directly supported.
- Do not claim the benchmark is comprehensive if it only measures a subset of the broader capability.
- If the benchmark has known artifacts, leakage risks, or annotation uncertainty, state them clearly.
```

PAPERS CONTRIBUTING BOTH (AGENTS_BOTH.MD)

```
You are an ambitious AI researcher who is looking to publish a paper that will contribute significantly to the
       field.

This paper contributes BOTH a new method AND a new benchmark / evaluation framework. The paper proposes a
      novel technique while also introducing a dataset, benchmark, or evaluation protocol that validates it
      and serves as a resource for the community.

Ensure that the paper is scientifically accurate, objective, and truthful. Accurately report the experimental
      results, even if they are negative, mixed, or inconclusive.

You are planning to submit to a top-tier ML conference, which has guidelines:

- In general, try to use the available space and include all relevant information.
- Do not change the overall style which is mandated by the conference. Keep to the current method of including
       the references.bib file.
- Do not remove the \graphicspath directive or no figures will be found.
- Do not add 'Acknowledgements' section to the paper.
- Use a single backslash (\) for LaTeX commands instead of a double backslash (\\).

Here are some tips for each section of the paper:

- **Title**:
  - Title should be catchy and informative. It should convey both the method contribution and the benchmark/
       dataset contribution.
  - Try to keep it under 2 lines.

- **Abstract**:
  - TL;DR of the paper.
  - What problem are we solving and why is it relevant?
  - Briefly describe both the proposed method and the benchmark/dataset.
  - Highlight key empirical findings.
  - This should be one continuous paragraph.
  - Keep the abstract to around 200 words.
  - Do not include formatted mathematical formulas or equations.

- **Introduction**:
  - Provide context and explain the relevance of both the method and the benchmark.
  - Motivate why existing methods are insufficient AND why existing benchmarks/datasets are inadequate.
  - Summarize contributions clearly, distinguishing between the method contribution and the benchmark
       contribution.
  - If there is a method overview or benchmark overview figure, place it on page 2 of the paper.

- **Related Work**:
  - Cover both related methods and related benchmarks/datasets.
  - Compare and contrast with prior work in both dimensions.
  - Ensure proper citations are provided.

- **Benchmark Design / Dataset Construction**:
  - Clearly describe the benchmark or dataset being introduced.
  - Explain data sources, collection, curation, annotation, and quality control.
  - Describe the evaluation protocol, metrics, and task definitions.
  - Justify design decisions.

- **Method**:
  - Clearly detail the proposed method and why it addresses the identified problem.
  - Include technical details sufficient for reproduction.
```

```
- **Experiments**:
  - Evaluate the proposed method on the introduced benchmark and potentially on existing benchmarks.
  - Include strong baselines and comparisons.
  - Present results truthfully.
  - In tables, please bold the best result among the compared methods.

- **Analysis**:
  - Provide deeper insights into both the method's behavior and the benchmark's diagnostic value.
  - Include ablation studies for the method.
  - Analyze benchmark properties (difficulty, diversity, discrimination).
  - Ensure all claims are supported by data.

- **Conclusion**:
  - Summarize both the method and benchmark contributions.
  - Highlight key findings and the value of the benchmark to the community.

Ensure you are always writing good compilable LaTeX code. Common mistakes that should be fixed include:

- LaTeX syntax errors (unenclosed math, unmatched braces, etc.).
- Duplicate figure labels or references.
- Unescaped special characters: & % $ # _ {{ }} ~ ^ \\
- Proper table/figure closure.
- Do not hallucinate new citations or any results not in the logs.

Ensure proper citation usage:

- Always include references within \\begin{{filecontents}}{{references.bib}} ... \\end{{filecontents}}, even
    if they haven't changed from the previous round.
- Before citing any paper, ALWAYS read the references.bib file first to find the correct citation key. Do NOT
    create new fictional bibtex entries.
- Do not make any changes to reference.bib
- Verify all citation keys match exactly with those in references.bib before using them in the text.
- Each section, especially Related Work, should have multiple citations.

## Notes

- Do not include formatted mathematical formulas or equations in the Abstract.
- Do not include Limitation sections and Future Work sections.
- Use `\%` to display a literal percent sign, as a standard `%` will be treated as a comment.
- Clearly separate the method contribution from the benchmark contribution in the paper structure.
- The benchmark must be justified, not merely presented.
```

D EXAMPLES OF AI-GENERATED PAPERS

We provide examples of AI-generated papers produced by Claude Code with Claude Sonnet 5 and Codex with GPT-5.6-terra in the supplementary material. The papers are shared with hallucination annotations, indicating claims identified as inconsistent with the corresponding original papers.