Can AI Revise Research Papers with Human Review Feedback? An Empirical Study and Benchmark

Zihan Luo[1] **and Hong Huang**[1*] **and Jianxun Lian**[2] **and Yu Chang**[1] **and Xing Xie**[2] **and Hai Jin**[1]
[1] Huazhong University of Science and Technology [2] Microsoft Research Asia
{zihanluo, honghuang, changyu, hjin}@hust.edu.cn
{jianxun.lian, xingx}@microsoft.com

Abstract

The rise of Human-AI collaboration can effectively speed up the research process for experts and allow anyone with critical thinking skills to conduct innovative work. A key part of this collaboration is the AI's ability to improve a paper with human feedback—updating both the text and experiments to meet high standards. To evaluate this skill, we introduce ReviseBench, an extensible benchmark built on real academic data that can be easily scaled via agent-driven automated data collection. It tests the skills of *Large Language Models* (LLMs) on paper interpretation, experimental implementation, and paper formulation, using authors' camera-ready versions as natural human baselines. To facilitate a fine-grained assessment, we further propose ReviseArena, a platform supporting pair-wise comparisons between different AI-revised papers. Our initial evaluation results on ReviseBench reveal that even state-of-the-art foundation LLMs struggle significantly in this domain, achieving a win rate of less than 10% against human experts, and facing issues like incremental revision, unprofessional revision, and potential data fabrication. Our code and data are released publicly at: https://github.com/CGCL-codes/ReviseBench.

1 Introduction

Large Language Models (LLMs) are fundamentally reshaping the scientific landscape, transitioning the role of AI from passive tools to active partners within a new paradigm of Human-AI collaboration. This collaborative framework holds the potential to revolutionize the research ecosystem in two profound ways: First, it significantly accelerates the research lifecycle, allowing researchers to navigate the arduous path from hypothesis generation (Si et al., 2025; Yang et al., 2024) to academic paper generation (Tang et al., 2025a; Lu et al., 2026) with unprecedented efficiency. Second, it empowers individuals to engage in high-level scientific inquiry. By mitigating technical barriers, this paradigm ensures that anyone equipped with critical thinking and innovative insights can contribute to rigorous academic work, regardless of their proficiency in coding or academic writing.

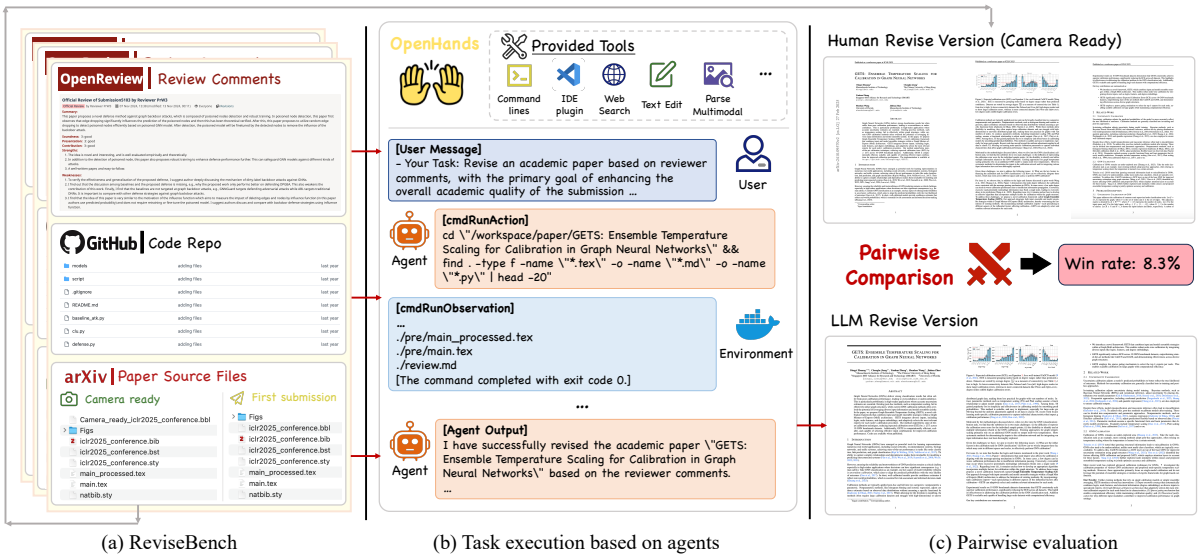
Central to realizing this synergy is the AI agent's capability to dynamically refine and elevate the quality of scientific work in response to human instructions or expert feedback. In this context, the revision process is not merely a formatting task, but the feedback loop where human insight directs AI execution. However, systematically evaluating an AI's ability to perform such complex, feedback-driven iterations—from textual polishing to experiment implementation—remains an open challenge.

To rigorously assess this critical capability, we introduce **ReviseBench**, an extensible benchmark specifically designed to formalize the task of autonomous paper revision within a realistic academic setting. In detail, ReviseBench now comprises 12 high-quality representative papers selected from ICLR 2025, and can be further expanded via agent-driven automated data collection. As shown in Figure 1, for each sample, we construct a comprehensive data triplet consisting of the official OpenReview comments, the corresponding GitHub code repository, and the initial LaTeX source files. The AI agents is then tasked with autonomously interpreting the review comments and elevating the manuscript's quality. By strictly retrieving repository snapshots and LaTeX source files from the initial submission timestamp—prior to the release of peer reviews—we ensure evaluation rigor and eliminate potential data leakage.

Compared to existing benchmarks (Chan et al.,

*Hong Huang is the corresponding author. Zihan Luo, Hong Huang, Yu Chang, and Hai Jin are affiliated with the National Engineering Research Center for Big Data Technology and System, Services Computing Technology and System Lab, Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology.

Findings of the Association for Computational Linguistics: ACL 2026, pages 17876–17893



(a) ReviseBench (b) Task execution based on agents (c) Pairwise evaluation

Figure 1: ReviseBench is a benchmark for evaluating the capabilities of LLMs for revising papers based on the review comments. (a) ReviseBench collects review comments, code repository, and LaTeX source files for each paper. (b) Based on the provided files, LLMs with agent scaffolds are prompted to revise the corresponding academic paper, including text revision, experiment execution and others. (c) The generated paper will be compared with the human camera-ready version by an LLM-based judge for pair-wise evaluation.

2025; Jain et al., 2025; Zhuo et al., 2025), ReviseBench presents following distinct and significant challenges: 1) *Environment Complexity*: ReviseBench goes beyond simple code generation by requiring comprehensive environment configuration. This necessitates that the model handle both dependency installation and data retrieval, simulating the real-world setup processes of scientific projects; 2) *Scientific Interpretation*: While sharing the requirement for manuscript comprehension with PaperBench (Starace et al., 2025), ReviseBench uniquely advances this challenge by incorporating peer review feedback. To succeed, the model must possess domain expertise comparable to that of a human scientist, allowing it to accurately align nuanced reviewer critiques with the corresponding content of the research paper; 3) *Full-scope Revision*: Different from existing benchmarks limited strictly to code modification (Jimenez et al., 2024) or experimental execution (Chan et al., 2025; Starace et al., 2025), ReviseBench integrates the task of manuscript revision and polishing. This imposes significant demands on the model's scientific writing capabilities and long-context processing, requiring it to synthesize technical code updates back into the textual narrative and chart illustrations effectively.

To rigorously assess performance, we establish a progressive evaluation framework that spans from fundamental executability—exemplified by metrics such as PDF generation rates—to high-level modification quality. Within this framework, we leverage final published camera-ready versions as a human-expert 'gold standard' to conduct pairwise comparisons, allowing us to quantify the gap between AI-generated revisions and human authorship. Complementing this, we further introduce ReviseArena, an evaluation environment designed to derive more granular performance through direct model-versus-model comparisons.

We conduct extensive evaluations of multiple state-of-the-art LLMs using ReviseBench. Our experimental results indicate that even advanced models, such as GPT-5 and DeepSeek-V3.2, still lag significantly behind human performance on ReviseBench. Specifically, they achieve win rates V.S. humans of only 8.33% and 0%, respectively. Our analysis also indicates that, compared with human experts, LLMs are more incremental, unprofessional, and may face issues like data fabrication during paper revision. In summary, our contributions are as follows:

- We propose ReviseBench, which is the first benchmark for AI-driven paper revision to the best of our knowledge, featuring distinct characteristics like complex code implementation and scientific interpretation.

- To rigorously assess AI-driven paper revision, we establish a comprehensive evaluation framework ranging from elementary to advanced levels. Additionally, we further develop ReviseArena, a comparative evaluation platform designed to provide fine-grained performance evaluation through pair-wise battles.

- We conduct extensive evaluation and experiments on the advanced LLMs, providing several key insights into the current capabilities and limitations of AI in the scientific revision process.

2 Related Work

2.1 LLMs for scientific research

The rapidly evolving capabilities of LLMs have positioned them as important roles in scientific research. Existing studies leveraging LLMs in research can be broadly categorized into three main areas based on the stage of the research process they address: LLM for Idea Generation, LLM for Paper Generation, and LLM for Paper Review.

1) LLMs for Idea Generation. Benefits from the world knowledge acquiring during pre-training, LLMs show their potential to enhance human creativity (Lee and Chung, 2024). Following this insight, numerous researchers have explored using LLMs to generate novel research ideas or hypotheses, demonstrating their utility across fields including Chemistry (Yang et al., 2025) and Artificial Intelligence (Si et al., 2025; Yang et al., 2024; Baek et al., 2025).

2) LLMs for Paper Generation. Beyond simple idea generation, researchers also focus on utilizing LLMs for complete scientific paper creation. Via preference alignment, LLMs like CycleResearcher (Weng et al., 2025) has significantly improved the quality of LLM-generated papers. More comprehensive LLM-powered systems introduce multi-agent systems and complete workflow to cover the entire research process (Schmidgall et al., 2025; Tang et al., 2025a; Lu et al., 2026), encompassing idea generation, experimental execution, and research writing.

3) LLMs for Paper Review. Finally, aiming to reduce the time burden on waiting for peer reviews, researchers attempt to deploy LLMs in paper review. Initial efforts primarily relied on prompt engineering, instructing powerful models like GPT-4 with detailed review guidelines and examples (Lu et al., 2026; Jin et al., 2024). Cyclereviewer (Weng et al., 2025) and DeepReview (Zhu et al., 2025b) go a step further by employing fine-tuning techniques to better align LLM preferences in review generation with those of human experts.

In contrast to these efforts, our work concentrates on exploring the capabilities of LLMs on paper refinement with review feedback, integrating paper understanding, experimental implementation, and iterative manuscript revision.

2.2 ML engineering and research evaluation

With the advancement of LLM capabilities, research attention has shifted from simple code generation (Jain et al., 2025; Zhuo et al., 2025; Lyu et al., 2025) to more complex programming scenarios such as software engineering (Jimenez et al., 2024; Weng et al., 2026; Wang et al., 2025a) and machine learning engineering (Huang et al., 2024; Chan et al., 2025; Jing et al., 2025). For instance, OpenAI curated a dataset of 75 high-quality Kaggle competitions to benchmark LLM proficiency on machine learning competitions against human participants (Chan et al., 2025). Additionally, recent studies have begun to investigate the application of LLMs to coding tasks related to scientific research. Notably, OpenAI selected 20 papers from ICML 2024 to test the ability of LLMs to reproduce research implementation (Starace et al., 2025), while ScienceAgentBench (Chen et al., 2025) extracts authentic scientific tasks from peer-reviewed literature for model execution and evaluation. In contrast to prior works, ReviseBench uniquely focuses on the capacity of LLMs to enhance the quality of academic papers, including the model's ability to modify code in response to peer review comments.

3 ReviseBench

In this section, we will describe the construction of ReviseBench. As an initial step, ReviseBench comprises 12 representative papers accepted by ICLR 2025, aiming to evaluate the capabilities of LLMs in revising papers based on human review feedback. Due to space limitations, we provide the whole list of papers in ReviseBench in Appendix B. We will elaborate on the details of ReviseBench in the following parts.

3.1 Data curation

As illustrated in Figure 1(a), the following parts are collected for each paper in ReviseBench:

- **Review comments**: The official review comments released in OpenReview are crawled via OpenReview API[1] for each paper. Reflecting the standard workflow where authors prioritize resolving identified shortcomings during paper revision, only *weaknesses* are retained and saved as `review.md`.

- **Code repository**: The source code for each paper in ReviseBench is publicly accessible on GitHub, with at least one commit predating the ICLR 2025 review disclosure. To establish an authentic pre-revision state and avoid potential data leakage, the repository corresponding to the most recent commit prior to the review disclosure date is preserved as the `code/` directory.

- **LaTex source files**: Given the inherent inaccuracy of OCR on PDF documents, all revisions are conducted on the original LaTeX source files. We confirm that each paper in ReviseBench had two distinct submissions to the arXiv platform, one temporally preceding and one following the disclosure of the ICLR 2025 reviews[2]. Note that, the post-revision source files are reserved solely for subsequent evaluation and remain inaccessible to the LLMs during task execution.

The construction of ReviseBench begins by selecting from the 3,703 accepted papers at ICLR 2025. Following (Starace et al., 2025), we only consider 236 Oral or Spotlight papers given their high representativeness within contemporary AI research. We further exclude papers lacking pre- or post-review arXiv submissions via arXiv API[3]. Subsequently, we manually screen the remaining 111 papers, confirming that each one has at least one commit on GitHub predating the ICLR 2025 review disclosure. It is worth noting that, we empirically find that the ReviseBench can be easily expanded and dynamic, and the whole data collection process can be automated with the help of powerful agents like OpenHands (Wang et al., 2025b). More details about the automated data retrieval will be discussed in Appendix B.2.

3.2 Rules

We establish the following rules to ensure the integrity and rigor of the task execution in ReviseBench: 1) For supplementary experiments, models must perform genuine computational runs. Data fabrication or result estimation is strictly forbidden. 2) Models are prohibited from searching the most recent versions of the papers, code repository, or existing rebuttal discussions online. 3) To facilitate downstream evaluation, models must produce a successfully compiled PDF version of the revised paper, free from basic typesetting or compilation failures.

To enforce these regulations, we record the complete execution trajectories of each model and employ Gemini-3-Pro (operating as an independent rule-breaking detector) to verify compliance and detect potential violations. We also incorporate some empirical guidelines into the system prompt to enhance task execution quality, with comprehensive details provided in the Appendix D.2.

Table 1: Accuracy of several models on JudgeTest

MODEL	ACCURACY
RANDOM	0.50
GLM-4.7	0.79 ± 0.04
DEEPSEEK-V3.2	0.83 ± 0.08
GPT-5.2	0.96 ± 0.04
GEMINI-3-PRO	1.00 ± 0.00

3.3 Evaluation

Metrics To assess the efficacy of models in completing the tasks in ReviseBench, we design the following four evaluation metrics, ranging from basic execution to advanced quality comparison:

- **PDF Generation Rate**: The percentage of instances where the model successfully compiles and produces a PDF file about the revised paper.

- **Valid PDF Generation Rate**: The percentage of instances in which the model compiles and produces a PDF file that satisfies the basic constraints of paper formatting and all specified rules mentioned in Section 3.2.

- **Win Rate vs. Original**: The ratio by which the model's revised version is judged superior to the original submission in terms of overall quality.

- **Win Rate vs. Human**: The ratio by which the model's revised version is judged superior to the human-authored camera-ready version.

To facilitate the evaluation process, the desired output format for each instance in ReviseBench

[1] https://docs.openreview.net/reference/api-v2
[2] November 13th, 2024 (Anywhere on Earth)
[3] https://info.arxiv.org/help/api/index.html

Table 2: Comparisons with existing benchmarks. The column *"Packages Installation?"* indicates whether dependency installation is required. *"Data Downloading?"* denotes whether data retrieval is necessary during execution. The *"Paper Understanding?"* and *"Review Understanding?"* columns specify whether the task requires understanding of the paper and review comments, respectively.

Benchmarks	Packages Installation?	Data Downloading?	Paper Understanding?	Review Understanding?	Source	Tasks
SWE-Bench (Jimenez et al., 2024)	✗	✗	✗	✗	GitHub	2,294
ML-Bench (Tang et al., 2025b)	✓	✗	✗	✗	GitHub	260
MLE-Bench (Chan et al., 2025)	✓	✗	✗	✗	Kaggle	75
MLAgentBench (Huang et al., 2024)	✗	✗	✗	✗	Kaggle	13
DS-Bench (Jing et al., 2025)	✓	✗	✗	✗	ModelOff & Kaggle	540
ScienceAgentBench (Chen et al., 2025)	✗	✗	✗	✗	Publications	102
EXP-Bench (Kon et al., 2026)	✓	✓	✗	✗	Publications	461
PaperBench (Starace et al., 2025)	✓	✓	✓	✗	Publications	20
ReviseBench (Ours)	✓	✓	✓	✓	Publications	12

is a PDF with revised content. We also employ LLM judges for calculating *Win Rate vs. Original* and *Win Rate vs. Human*. Notably, we deliberately exclude predictive reviewer models, such as DeepReviewer (Zhu et al., 2025b) or PaperReview[4], for calibrated scoring. This decision stems from two key observations: 1) our empirical analysis reveals that these models lack the sensitivity to distinguish fine-grained improvements between revisions, frequently assigning identical scores; and 2) prior research establishes that LLMs demonstrate superior reliability in pairwise preference evaluation compared to absolute scoring (Si et al., 2025).

JudgeTest Given the time costs associated with human expert review and the ineffectiveness of existing review models, we adopt an LLM-based pairwise comparison approach similar to (Starace et al., 2025; Zhu et al., 2025a). To validate the efficacy of LLMs as judges, we construct JudgeTest, a benchmark comprising 12 accepted papers from ICLR 2025 that are distinct from ReviseBench. We list the papers in JudgeTest in the Appendix C. Specifically, we retrieve two versions of each paper in JudgeTest from their arXiv submission history, assuming that the later revision represents superior quality after human revision and update. To mitigate the potential position bias (Zheng et al., 2023; Zhu et al., 2025a) in LLM-based judges, each paper pair was evaluated twice after swapping the position. We then evaluate GLM-4.7 (Zeng et al., 2025), DeepSeek-V3.2 (Liu et al., 2025), GPT-5.2, and Gemini-3-Pro on JudgeTest, and report their average accuracy on identifying the stronger version. As illustrated in Table 1, both Gemini-3-Pro and GPT-5.2 achieve satisfactory performance, and we select Gemini-3-Pro as the default judge for all subsequent evaluations.

ReviseArena Given the constrained scale of ReviseBench, relying solely on absolute performance metrics such as *valid PDF generation rate* or *win rate V.S. human* may fail to capture the subtle distinctions between models, particularly when their performance gaps are narrow. To address this limitation and enhance our evaluation, we introduce ReviseArena as a complementary evaluation module. Instead of evaluating models in isolation with absolute metrics, ReviseArena employs a pair-wise comparison mechanism (Feng et al., 2025; Chiang et al., 2024), where models will compete directly on the identical revision task with each other. This approach allows us to derive fine-grained insights into model superiority, effectively alleviating the statistical limitations imposed by the dataset size.

3.4 Comparisons with existing benchmarks

As shown in Table 2, we compare ReviseBench with several related benchmarks, highlighting the following distinct characteristics that set it apart: 1) **Environment Complexity**: Unlike SWE-bench (Jimenez et al., 2024) or MLAgentbench (Huang et al., 2024), ReviseBench necessitates both dependency installation and data downloading, which is consistent with complex, real-world research projects. 2) **Scientific Interpretation**: Similar to PaperBench (Starace et al., 2025), ReviseBench requires a comprehensive understanding of the research manuscript. However, it advances this requirement by also necessitating the understanding of peer review feedback. This demands that LLMs possess domain expertise comparable to human scientists to accurately align the reviews with the paper's content. 3) **Full-Scope Revision**: Di-

[4]https://paperreview.ai/

Table 3: The main performance of models with OpenHands on ReviseBench. Best performance are **bolded**.

Model	PDF Gen. Rate	Valid PDF Gen. Rate	Win rate vs. Original	Win rate vs. Human
o3	8.33%	0.00%	0.00%	0.00%
Gemini-2.5-Pro	25.00%	8.33%	8.33%	0.00%
o4-mini	25.00%	25.00%	16.67%	0.00%
Claude-4-sonnet	**100.00%**	16.67%	16.67%	0.00%
GPT-5	75.00%	50.00%	41.67%	**8.33%**
DeepSeek-V3.2	**100.00%**	**58.33%**	**50.00%**	0.00%

verging from the aforementioned benchmarks, ReviseBench is not limited to code modification or experimental execution, while uniquely integrating the task of manuscript revision and polishing. This imposes significant demands on the model's capabilities in scientific writing and long-context processing, requiring it to synthesize code changes back into the textual narrative or even charts.

Note that, although ReviseBench currently only comprises 12 tasks, we envision ReviseBench as an extensible, community-driven data resource. With the continuous updates of academic conferences and the growing availability of open-sourced paper artifacts, ReviseBench can be easily extended and dynamic. We position the current version as a preliminary investigation into the domain of automated paper revision, with the expansion of the dataset reserved for future work.

4 Experiments

4.1 Experimental setup

Considering the task's complexity in ReviseBench, we mainly limit our evaluation scope to leading foundational LLMs (Starace et al., 2025). Our selection strategy is designed to cover varying model scales and licensing types, including GPT-5[5], DeepSeek-V3.2 (Liu et al., 2025), Claude-4-Sonnet[6], Gemini-2.5-Pro, o3, and o4-mini. We employ OpenHands (Wang et al., 2025b) as our scaffold due to its comprehensive tool-calling proficiency. All evaluations are performed in an Ubuntu 22.04 Docker container with a 32GB V100 GPU and 100GB SSD.

4.2 Main experiments

In this section, we would like to give some fundamental observations based on our evaluation, with more detailed analysis in the following parts. Table 3 demonstrates the performance of models on

[5]gpt-5-2025-08-07

[6]claude-4-sonnet-20250514

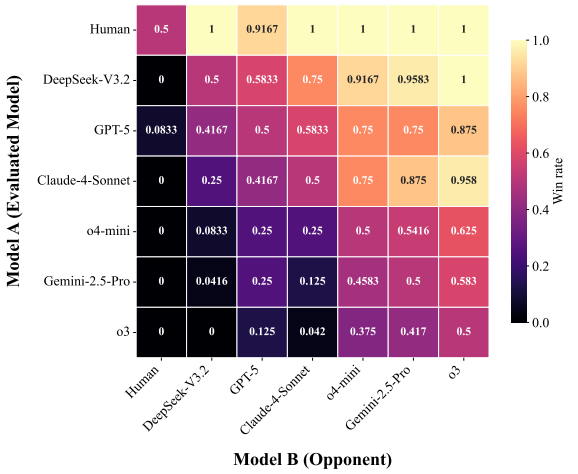


Figure 2: The average win rates of Model A (evaluated model) against Model B (opponent) across all twelve papers in ReviseArena.

ReviseBench, where we can observe:

- To our surprise, even LLMs like Gemini-2.5-Pro and o4-mini are struggling with generating valid revised paper PDFs, leading to poor win rates against the original versions and camera-ready versions provided by human experts. Our manual investigation on several trajectory logs indicates that these models frequently face issues like LaTeX dependency configuration or instruction misunderstanding.

- Although LLMs like Claude-4-Sonnet, GPT-5, and DeepSeek-V3.2 demonstrate high proficiency in PDF generation, their valid PDF gernration ratio seems to be unremarkable, ranging from 16.67% to 58.33%. The manual review reveals problems regarding violations of established rules, and evident presentation errors. These findings indicates that models still face significant challenges in generating manuscripts that adhere to real-world academic standards.

- When compared with camera-ready versions provided by human experts, all models achieve extremely poor win rates, with the highest score

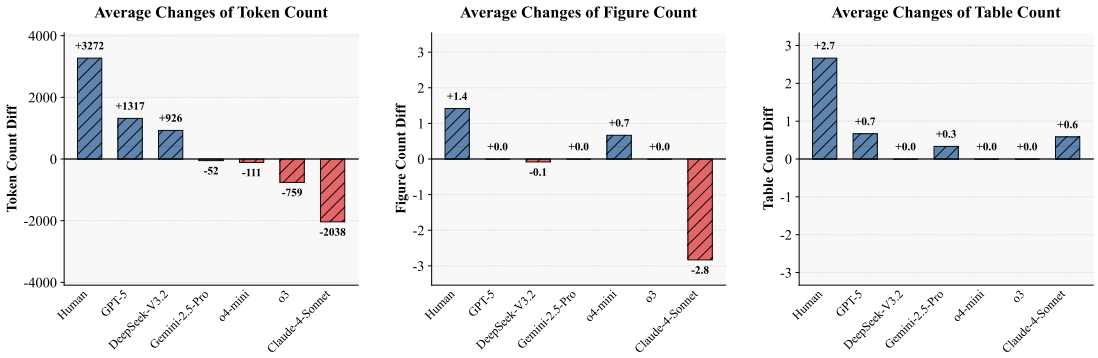


Figure 3: The average changes of tokens, figures, or tables between the original papers and the revised papers

8.33%, unveiling the significant gap between LLMs and human researchers on paper revision tasks. According to the trajectory logs, we manually find that the deficiency includes incomplete experiments, poor paper presentation and others, which we will discuss later.

Note that, as the initial steps in investigating the performance of LLMs on paper revision, we focus on introducing ReviseBench and providing the empirical results with basic scaffolds. We believe current results do not necessarily reflect these models' upper limits.

4.3 ReviseArena

To establish a more fine-grained comparative metric for different models, we further introduce ReviseArena as a complement to the current evaluation framework. Using Gemini-3-Pro as the judge, we conduct pair-wise comparisons among revised versions of the same paper generated by different models and subsequently compute their respective win rates. To mitigate potential position bias (Zhu et al., 2025a; Li et al., 2024; Zheng et al., 2023), each pairwise comparison is performed twice, swapping the positions of the models. More details on the pair-wise comparison in ReviseArena will be provided in Appendix D.3. As presented in the Figure 2, the results are largely consistent with our expectations. The human-authored camera-ready versions demonstrate a dominant win rate over all evaluated models. Among the AI models, GPT-5 and DeepSeek-V3.2 exhibit comparable performance, while significantly outperforming Gemini-2.5-Pro, o4-mini, and o3.

4.4 In-depth analysis

In this section, we conduct a series of in-depth investigations on the performance of LLMs on ReviseBench, and we empirically have the following observations:

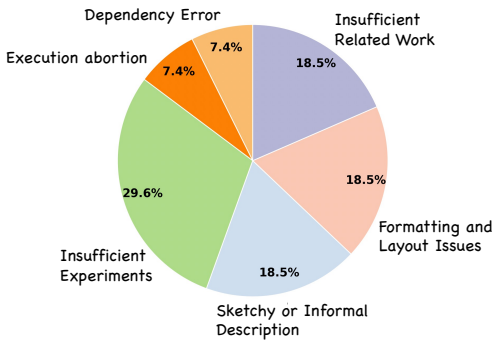
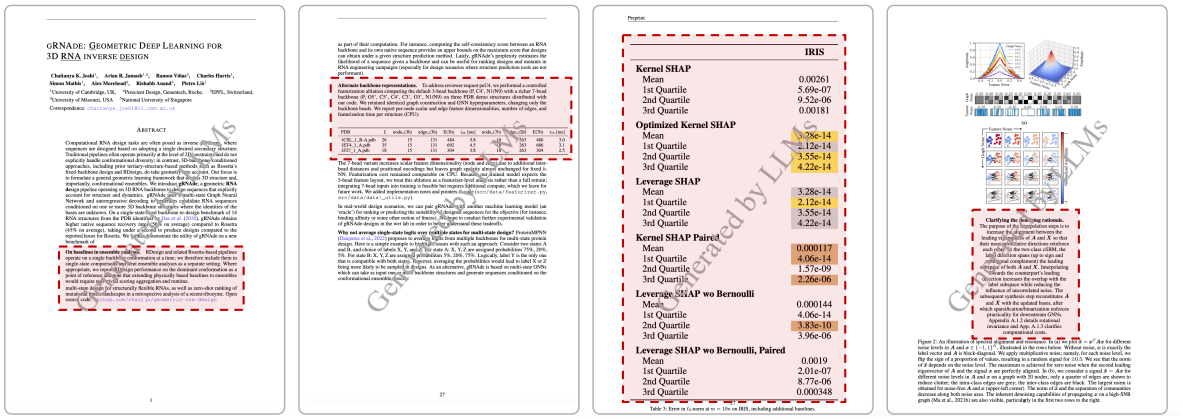


Figure 4: Deficiency modes distribution of GPT-5

LLMs are Incremental Revisers To investigate the specific modifications made by the models, we compare the AI-generated papers with the original papers in terms of total token count, number of figures, and number of tables. For reference, changes in human revisions for camera-ready versions are also provided. The comparison results are shown in Figure 3, and we find that: 1) the scale of modifications made by LLMs—spanning text, figures, and tables—is substantially smaller than those executed by human experts. This disparity implies that the models may lack the capacity for providing deep textual analysis and the ability to add comprehensive experiments; 2) regarding the modification types, LLMs generally exhibit a preference for incremental textual descriptions and the addition of tables, rather than introducing new experimental findings through graphical figures; 3) Claude-4-Sonnet seems to delete a large number of text and figures during the task. Human inspection reveals that this was caused by an incidental omission of the appendix after revision.



(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

Figure 5: Representative failure cases of GPT-5 in ReviseBench, with problematic areas highlighted in red boxes. (a) **Narrative Incoherence**: The model disrupts the logical flow of the abstract by intrusively inserting an unrelated explanatory paragraph mid-sentence. (b) **Informal rebuttal-style content**: The generated text contains raw rebuttal notes (e.g., "To address the reviewer request..."), failing to maintain a formal academic tone. (c) **Layout Overflow**: The generated table lacks proper width constraints, causing it to protrude into the page margins. (d) **Layout Overlap**: A layout collision where a text block is erroneously rendered atop the figure, obscuring the graphical content.

LLMs are Unprofessional Revisers As previously noted, the win rate of AI-generated papers falls below 10% when compared to those produced by human experts. To investigate the specific modes of these deficiencies, we conduct a manual analysis of the papers generated by GPT-5. Specifically, in instances where GPT-5 failed to produce a final paper, we further examine the trajectory logs to diagnose the failure. We summarize the primary issues in Figure 4, highlighting the gaps in professionalism between LLMs (such as GPT-5) and human researchers across the following aspects:

- Compared with camera-ready versions, the generated papers occasionally exhibit *insufficient related work*, characterized by a reduced number of references, and *insufficient experiments*, as evidenced by the disparity in the quantity of tables and figures in Figure 3. These findings imply the deficiency of LLMs in precisely addressing the review concerns via experiments.

- LLMs like GPT-5 occasionally produce *sketchy or informal descriptions*, manifesting as the inclusion of unpolished content or superficial explanations of supplemental experiments that lack insights or in-depth analysis. As illustrated in Figure 5(a), GPT-5 breaks the narrative coherence in the abstract by inserting an unrelated explanatory paragraph mid-sentence. While in Figure 5(b), GPT-5 erroneously included raw rebuttal-style comments, such as "To address the Reviewer xxx request...", directly into the manuscript body.

- Different from camera-ready versions, we also observe *formatting and layout issues* in the generated PDFs, such as image-text overlaps, tables/figures exceeding line width, or excessive whitespace, which severely compromised the visual quality of the papers. As evidenced in Figure 5(c), the generated table lacks proper width constraints, causing the content to overflow the page margins and breach the standard text width. Meanwhile, Figure 5(d) illustrates a layout overlap, where an explanatory text block is erroneously superimposed within the figure's bounding box due to a failure in isolating textual content from graphical elements.

LLMs might be Fabricated Revisers Data fabrication constitutes a severe breach of academic integrity, especially when deploying AI in scientific research (Watkins, 2024). In this part, we establish a two-step fabrication detection mechanism for potential AI misuse. We initially screen for revision cases where new experimental figures or tables were added by LLMs. Then, we conduct a secondary audit using Gemini-3-Pro to scrutinize the trajectory logs of these revised papers for potential data fabrication. The results are reported in Table 4, where both DeepSeek-V3.2 and Claude-4-Sonnet engaged in the fabrication of experimental data, such as estimating results without actual execution or using a synthetic dataset. These findings highlight the critical need for human oversight

Table 4: Statistics of adding new experimental results and data fabrication across evaluated models

MODEL	# NEW RESULTS	# FABRICATION CASE
O3	0	0
GEMINI-2.5-PRO	2	0
O4-MINI	1	0
CLAUDE-4-SONNET	6	5
GPT-5	6	0
DEEPSEEK-V3.2	2	1

to prevent ethical violations and uphold academic standards in AI-assisted research. Note that, all generated papers that violate data fabrication rules will be marked as invalid PDF after detection.

5 Conclusion

In this paper, we introduce ReviseBench, a pioneering benchmark tailored to assess LLMs on the complex paper revision tasks. Initialized with 12 ICLR 2025 papers, it establishes a rigorous standard requiring simultaneous mastery of scientific interpretation, coding implementation, and academic writing. We propose a progressive evaluation framework, using camera-ready papers as human baselines, and develop ReviseArena for granular pairwise comparisons. Our extensive experiments reveal a significant gap between current AI models and human experts, particularly in academic expression and scientific interpretation. These findings underscore the difficulty of autonomous revision and position ReviseBench as a critical resource for advancing AI-assisted research.

Limitations

We acknowledge and outline the following limitations of our work: 1) **Benchmark Scale**: Currently, ReviseBench comprises 12 papers from ICLR. While this scale is relatively modest compared to other benchmarks, it is a deliberate result of our strict selection criteria designed to guarantee the high quality of the papers and their associated data. Furthermore, given the high complexity of the tasks and the computational resources they demand, we believe that a compact dataset could allow for quick and efficient evaluation. We present this release of ReviseBench as an initial investigation, paving the way for future data integration. 2) **LLM-based Evaluation**: In ReviseBench, we employ LLMs as judges for pair-wise comparison. We acknowledge the inherent limitations of LLM-based judges compared to human experts, including potential issues with hallucinations and misjudgments. However, relying on expert human evaluation inevitably incurs prohibitive time and financial costs. Our experimental results on the JudgeTest demonstrate that our LLM-based approach represents a favorable trade-off between evaluation efficiency and accuracy, proving to be a reliable proxy for human judgment in this context.

Ethical Consideration

In this paper, our investigation into automated paper revision is driven by the goal of assisting the scientific community, rather than advocating for full automation. We acknowledge that human oversight remains paramount in the research workflow to safeguard against potential pitfalls such as factual hallucinations or ethical breaches. It is also important to note that the performance disparities observed in ReviseBench do not represent these models' ultimate potential. We present this benchmark as an initial milestone, hoping to inspire future research that pushes the boundaries of AI capabilities in complex scientific tasks, all while upholding the integrity and rigor of scientific inquiry.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 62127808). The computation was completed in the HPC Platform of Huazhong University of Science and Technology.

The authors' contributions are listed below.

Zihan Luo: Conceptual framing, experimental design, dataset sourcing, conducting experiments, result analysis, and writing.

Hong Huang: Conceptual framing and writing.

Jianxun Lian: Advising.

Yu Chang: Experimental design and conducting experiments.

Xing Xie: Advising.

Hai Jin: Advising.

References

Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. Researchagent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics, NAACL 2025, Albuquerque, New Mexico, USA, April*

29-May 4, 2025, pages 6709–6738. Association for Computational Linguistics.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. 2025. Mle-bench: Evaluating machine learning agents on machine learning engineering. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. 2025. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael I. Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Huawen Feng, Pu Zhao, Qingfeng Sun, Can Xu, Fangkai Yang, Lu Wang, Qianli Ma, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2025. Warriorcoder: Learning from expert battles to augment code large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27-August 1, 2025*, pages 4955–4969. Association for Computational Linguistics.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. Mlagentbench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. 2024. Swe-bench: Can language models resolve real-world github issues? In *Proceedings of the 12th International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Yiqiao Jin, Qinlin Zhao, Yiyang Wang, Hao Chen, Kaijie Zhu, Yijia Xiao, and Jindong Wang. 2024. Agentreview: Exploring peer review dynamics with LLM agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 1208–1226. Association for Computational Linguistics.

Liqiang Jing, Zhehui Huang, Xiaoyang Wang, Wenlin Yao, Wenhao Yu, Kaixin Ma, Hongming Zhang, Xinya Du, and Dong Yu. 2025. Dsbench: How far are data science agents from becoming data science experts? In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Patrick Tser Jern Kon, Qiuyi Ding, Jiachen Liu, Xinyi Zhu, Jingjia Peng, Jiarong Xing, Yibo Huang, Yiming Qiu, Jayanth Srinivasa, Myungjin Lee, Mosharaf Chowdhury, Matei Zaharia, and Ang Chen. 2026. EXP-bench: Can AI conduct AI research experiments? In *Proceedings of the 14th International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. OpenReview.net.

Byung Cheol Lee and Jaeyeon Chung. 2024. An empirical investigation of the impact of chatgpt on creativity. *Nature Human Behaviour*, 8(10):1906–1914.

Zongjie Li, Chaozheng Wang, Pingchuan Ma, Daoyuan Wu, Shuai Wang, Cuiyun Gao, and Yang Liu. 2024. Split and merge: Aligning position biases in llm-based evaluators. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 11084–11108. Association for Computational Linguistics.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

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

Zhi-Cun Lyu, Xinye Li, Zheng Xie, and Ming Li. 2025. Top pass: Improve code generation by pass@k-maximized code ranking. *Frontiers of Computer Science*, 19(8):198341.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. 2025. Agent laboratory: Using LLM agents as research assistants. In *Findings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 5977–6043. Association for Computational Linguistics.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2025. Can llms generate novel research ideas? A large-scale human study with 100+ NLP researchers. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. 2025. Paperbench: Evaluating AI's ability to replicate AI research. In *Proceedings of the 42nd International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

Jiabin Tang, Lianghao Xia, Zhonghang Li, and Chao Huang. 2025a. AI-researcher: Autonomous scientific innovation. In *Proceedings of the 39th Annual Conference on Neural Information Processing Systems, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025*.

Xiangru Tang, Yuliang Liu, Zefan Cai, Daniel Shao, Junjie Lu, Yichi Zhang, Zexuan Deng, Helan Hu, Kaikai An, Ruijun Huang, and 1 others. 2025b. Ml-bench: Evaluating large language models and agents for machine learning tasks on repository-level code. In *ICLR 2025 Third Workshop on Deep Learning for Code*.

Haoran Wang, Zhenyu Hou, Yao Wei, Jie Tang, and Yuxiao Dong. 2025a. SWE-Dev: Building software engineering agents with training and inference scaling. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27-August 1, 2025*, pages 3742–3761. Association for Computational Linguistics.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, and 2 others. 2025b. Openhands: An open platform for AI software developers as generalist agents. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Ryan Watkins. 2024. Guidance for researchers and peer-reviewers on the ethical use of large language models (LLMs) in scientific research workflows. *AI and Ethics*, 4(4):969–974.

Shihao Weng, Yang Feng, Yining Yin, Zhenlun Zhang, and Baowen Xu. 2026. Data preparation and quality for code-centric generative software engineering tasks: a systematic literature review. *Frontiers of Computer Science*, 20(9):2009203.

Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. 2025. Cycleresearcher: Improving automated research via automated review. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and Erik Cambria. 2024. Large language models for automated open-domain scientific hypotheses discovery. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 13545–13565. Association for Computational Linguistics.

Zonglin Yang, Wanhao Liu, Ben Gao, Tong Xie, Yuqiang Li, Wanli Ouyang, Soujanya Poria, Erik Cambria, and Dongzhan Zhou. 2025. Moose-chem: Large language models for rediscovering unseen chemistry scientific hypotheses. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. GLM-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Proceedings of the 37th Annual Conference on Neural Information Processing Systems, NeurIPS 2023, New Orleans, LA, USA, December 10-16, 2023*.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2025a. Judgelm: Fine-tuned large language models are scalable judges. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Minjun Zhu, Yixuan Weng, Linyi Yang, and Yue Zhang. 2025b. Deepreview: Improving llm-based paper review with human-like deep thinking process. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27-August 1, 2025*, pages 29330–29355. Association for Computational Linguistics.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, Simon Brunner, Chen Gong, James Hoang, Armel Randy Zebaze, Xiaoheng Hong, Wen-Ding Li, Jean Kaddour, Ming Xu, Zhihan Zhang, and 2 others. 2025. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. In *Proceedings of the 13th International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Appendix

A Use of LLMs

Large Language Models (LLMs) are used exclusively for textual polishing and refinement during manuscript preparation. All research ideas, experiments, and analyses are proposed, conducted, and provided by the authors.

B Details on ReviseBench

B.1 Paper List

The whole paper list in ReviseBench is shown in Table 5, where we include 5 oral and 7 spotlight papers from ICLR 2025. It can also be seen that ReviseBench includes varying research directions from AI for science to Reinforcement learning.

B.2 Discussion on the Extensibility

While the current version of ReviseBench only contains 12 papers from ICLR 2025, our data collection pipeline is designed to be highly extensible. Specifically, we found that the data retrieval process from platforms like OpenReview, arXiv, and GitHub can be efficiently automated using AI agents such as OpenHands (Wang et al., 2025b). By providing detailed procedural prompts, necessary API keys, and a few core code snippets as references, OpenHands with powerful LLMs like GPT-5.2 can autonomously execute the collection pipeline. Although we empirically found that this automated approach may not exhaustively retrieve every qualifying paper from a given conference, the successfully collected subset maintains high quality. Consequently, with only minimal human verification, ReviseBench can be effortlessly updated and expanded to encompass future conferences. Figure 6 and Figure 7 provide the exact prompts we used for automated data retrieval.

C Details on JudgeTest

To validate the reliability of our evaluators, we curated JudgeTest, a collection of 12 ICLR 2025 papers (Oral/Poster) covering topics such as AI for Science and AI Safety. The whole paper list is shown in Table 6. Papers in JudgeTest are only required to have accessible arXiv uploads spanning the pre-review and post-review periods, without the GitHub codebase requirements used in ReviseBench.

In our evaluation pipeline, the chronological progression of these papers serves as the ground truth for quality; the later version is always treated as the winner. We rigorously assessed the judge models by feeding them these version pairs. Crucially, to address the position bias often exhibited by LLMs, we conducted position-swapping experiments for every pair, averaging the results to ensure fairness. The quantitative validation results on JudgeTest is presented in Section 3.3.

D Details of Experiments

D.1 Scaffold

In this study, we adopt the CodeActAgent from the OpenHands(Wang et al., 2025b) framework as the default scaffold for all evaluated LLMs, given its comprehensive tool-calling capabilities, which are essential for handling the multi-step interactions required by ReviseBench. It is worth noting that we do not impose explicit constraints on the maximum execution time or the number of action steps, though we observe that the agents typically prefer to end rapidly, which is also observed in prior work (Starace et al., 2025). To facilitate subsequent qualitative analysis and debugging, we systematically archive the full execution trajectory logs generated during each session.

D.2 Prompt Settings

Beyond the fundamental task instructions and rule clarification, our prompt incorporates the following guidelines to enhance the robustness of the agents:

- **LaTeX Compilation and Syntax**: We observed that agents frequently encountered compilation errors in the generated PDFs, particularly regarding unrecognized references or figures (Weng et al., 2025). This issue was primarily attributed to the interference of residual files (e.g., stale .bib or .aux files) from previous compilation attempts. To mitigate this, we explicitly instructed the agents to clean and regenerate these auxiliary files before compilation. This protocol significantly reduced the occurrence of citation-related errors.

- **Autonomous Execution**: Some models exhibited a tendency to pause execution to solicit further instructions or confirmation from the user (i.e., "human-in-the-loop" behavior), which often led to task timeouts or failures in our automated pipeline. We incorporated strict instructions into the prompt requiring the agents to pro-

Table 5: List of Papers in ReviseBench

Paper	Source	ICLR Primary Area	Tokens
gRNAde: Geometric Deep Learning for 3D RNA inverse design	Spotlight	Applications to physical sciences	25,479
Language Representations Can be What Recommenders Need: Findings and Potentials	Oral	Other topics in machine learning	27,786
Robustness Inspired Graph Backdoor Defense	Oral	Alignment, fairness, safety, privacy, and societal considerations	26,629
Joint Graph Rewiring and Feature Denoising via Spectral Resonance	Oral	Learning on graphs and other geometries & topologies	40,102
MonST3R: A Simple Approach for Estimating Geometry in the Presence of Motion	Spotlight	Applications to computer vision, audio, language, and other modalities	18,501
Provably Accurate Shapley Value Estimation via Leverage Score Sampling	Spotlight	Interpretability and explainable AI	25,201
Residual Deep Gaussian Processes on Manifolds	Oral	Probabilistic methods	22,218
Revisiting Random Walks for Learning on Graphs	Spotlight	Learning on graphs and other geometries & topologies	53,991
Towards Marginal Fairness Sliced Wasserstein Barycenter	Spotlight	Other topics in machine learning	28,182
GETS: Ensemble Temperature Scaling for Calibration in Graph Neural Networks	Spotlight	Learning on graphs and other geometries & topologies	25,267
GeSubNet: Gene Interaction Inference for Disease Subtype Network Generation	Oral	Learning on graphs and other geometries & topologies	24,256
SimBa: Simplicity Bias for Scaling Up Parameters in Deep Reinforcement Learning	Spotlight	Reinforcement learning	25,045

ceed autonomously and resolve decisions internally without seeking user intervention.

- **Dependency Installation**: We found that network instability often hindered the installation of necessary libraries. Furthermore, we noted that agents occasionally displayed insufficient tolerance for long installation processes, terminating the installation step prematurely. To address this, we provided the agents with reliable proxy mirrors to improve connectivity. Additionally, we mandated a minimum timeout threshold of 1200 seconds for installation commands, ensuring the agents wait sufficiently for dependencies to resolve.

Our detailed prompt is provided in Figure 8 and Figure 9.

D.3 Details on ReviseArena

In Section 3.3, we establish ReviseArena to provide a more granular evaluation for LLMs. Specifically, the evaluation guidelines in ReviseArena are as listed below:

- If one model successfully generates a PDF file while its opponent fails to do so, the former is automatically declared the winner.

- If one model successfully generates a valid PDF file while its opponent fails to do so (invalid PDF or missing PDF), the former is automatically declared the winner.

- When both models produce valid PDFs, a content-based evaluation will be conducted. We employ Gemini-3-Pro as the judge to analyze the quality of the revisions and determine the superior model.

The final performance of each model is quantified using the win rate, calculated as follows:

$$\text{Win Rate} = \frac{N_{\text{win}} + 0.5 \times N_{\text{tie}}}{N_{\text{total}}} \quad (1)$$

where N_{win}, N_{tie}, and N_{total} represent the number of wins, ties, and total pairwise matches, respectively.

Table 6: List of Papers in JudgeTest

Paper	Source	ICLR Primary Area	Tokens
LOKI: A Comprehensive Synthetic Data Detection Benchmark using Large Multimodal Models	Spotlight	Datasets and benchmarks	68,120
Adversarial Perturbations Cannot Reliably Protect Artists From Generative AI	Spotlight	Alignment, fairness, safety, privacy, and societal considerations	24,547
Learning to Discretize Denoising Diffusion ODEs	Oral	Generative models	26,303
SynFlowNet: Design of Diverse and Novel Molecules with Synthesis Constraints	Spotlight	Applications to physical sciences	20,623
IGL-Bench: Establishing the Comprehensive Benchmark for Imbalanced Graph Learning	Spotlight	Datasets and benchmarks	68,489
Knowledge Localization: Mission Not Accomplished? Enter Query Localization!	Spotlight	Interpretability and explainable AI	20,606
Uncovering Overfitting in Large Language Model Editing	Spotlight	Foundation or frontier models, including LLMs	18,020
Cheating Automatic LLM Benchmarks: Null Models Achieve High Win Rates	Oral	Foundation or frontier models, including LLMs	21,593
Improving Unsupervised Constituency Parsing via Maximizing Semantic Information	Spotlight	Applications to computer vision, audio, language, and other modalities	16,782
Provably Reliable Conformal Prediction Sets in the Presence of Data Poisoning	Spotlight	Alignment, fairness, safety, privacy, and societal considerations	20,880
Learning stochastic dynamics from snapshots through regularized unbalanced optimal transport	Oral	Applications to physical sciences	23,343
Improving Probabilistic Diffusion Models With Optimal Diagonal Covariance Matching	Oral	Generative models	25,391

Table 7: The consistency between human evaluation and Gemini-3-pro

	Consistency on win rate V.S Original	Consistency on win rate V.S Human
Human	10/10	10/10

Table 8: Win rate of camera-ready versions

Judge	Win rate of camera-ready versions
human	12/12
Gemini-3-Pro	12/12

E Additional Experiments

E.1 Human Verifications on LLM Judges

To further substantiate the reliability of our LLM-based evaluation pipeline, we conducted additional human verifications on the judgment of Gemini-3-Pro. Specifically, we randomly sampled 20 pairs of comparison results evaluated by Gemini-3-Pro and invited two graduate-level volunteers with at least two top-tier AI conference publications to manually inspect and judge them independently.

The consistency between the human volunteers' checks and the LLM's automated judgments is as shown in Table 7. It can be observed that, the LLM's judgments are highly consistent with human experts, indicating that LLM can be a reliable proxy for human judgment after considering the balance of efficiency and accuracy.

E.2 Human Verifications on Paper Quality

To verify that the camera-ready papers can consistently serve as a high-quality human baseline, we conducted a rigorous blind comparison between the camera-ready versions and the original submissions. We invited two graduate-level human volunteers with at least two top-tier AI conference publications and utilized Gemini-3-pro to perform this evaluation, respectively. To ensure a fair comparison and eliminate any visual bias, we strictly anonymized the documents by removing formatting differences, watermarks, and other identifying markers. As shown in Table 8, both human and Gemini-3-Pro consistently agree that the camera-ready versions have higher quality compared with the original versions.

System Prompt for Automated Data Collection (Part 1)

Role: You are an AI assistant proficient in academic data retrieval and automated processing. Please help me automatically collect and organize data related to accepted papers for ICLR 2025.

Context & Objective: For all accepted papers at ICLR 2025, please filter for those that **simultaneously satisfy** the following two conditions, and then download, process, and save the files as required:

1. **ArXiv Version Requirement:** The paper must have at least one upload record on arXiv both **before** and **after** the ICLR 2025 Review Release Date (**2024-11-13T00:00:00Z**).

2. **GitHub Code Requirement:** The official GitHub repository must have at least one code commit record prior to **2024-11-13T00:00:00Z**.

Environment & Tools: In the current working directory under the `workplace/` folder, I have provided the following auxiliary scripts. Please call them in your code:

- `workplace/paper.py`: Executes to fetch the list of papers satisfying "Condition 1" (having arXiv versions both before and after the cutoff date).
- `workplace/review.py`: Used to fetch the paper's OpenReview comments and generate `review.md`.
- `workplace/latex.py`: Used to process the `pre/` folder by removing comments and merging LaTeX files.

Workflow Execution: Please write and run a Python script to complete the following steps:
Step 1: Obtain Candidate List
Run `workplace/paper.py` to get the initial list of papers that satisfy the arXiv timing conditions.
Step 2: GitHub Commit Filtering
Iterate through the paper list from Step 1 and execute the following strict filtering and cloning process:

1. **Repository Localization and Validation (Critical Step):**
    - Extract the GitHub URL from `cr.pdf` (the **latest** arXiv PDF version of the paper uploaded **after 2024-11-13T00:00:00Z**), and check if the repository's README file matches the paper's content.
    - **Rejection Criterion:** If no URL can be found in the PDF, or if the README of all found repositories does not match the paper's content, exclude the paper directly and skip further operations. (*Note: Please use your own knowledge and understanding to make this judgment, rather than writing Python code for regex matching*).
2. **Time-based Backtracking Filter:**
    - For repositories that pass the above validation, check their commit history.
    - **Rejection Criterion:** If the repository has no commit records prior to **2024-11-13T00:00:00Z**, exclude the paper.
3. **Execution of Clone:**
    - For papers meeting all the above criteria, locate the last commit before **2024-11-13T00:00:00Z**, rollback-/checkout the repository to this state, clone it, and save it as `code/`.

Figure 6: System prompt for automated data collection (Part 1)

System Prompt for Automated Data Collection (Part 2)

Workflow Execution (Continued):
Step 3: Data Download and Processing
For each filtered paper, create a folder named after its Paper ID or Title, and generate the following 5 items inside it:

1. **`review.md`**
   - **Action:** Call `workplace/review.py`.
   - **Content:** Extract the Weaknesses section from the paper's OpenReview.
2. **`code/`**
   - **Action:** Clone the paper's GitHub repository.
   - **Key:** Use `git` commands to checkout the code to the last commit state before **2024-11-13T00:00:00Z**.
   - **Save:** Store the code in this state within the `code/` directory.
3. **`pre/`**
   - **Action:** Download the source files of the paper's **most recent** arXiv version prior to **2024-11-13T00:00:00Z**.
   - **Processing:** Extract the source files into the `pre/` directory, then run `workplace/latex.py` to preprocess this directory (cleaning comments and merging files).
4. **`original.pdf`**
   - **Action:** Download the PDF of the paper's **most recent** arXiv version prior to **2024-11-13T00:00:00Z**.
5. **`cr.pdf`**
   - **Action:** Download the **latest** arXiv PDF version of the paper updated after **2024-11-13T00:00:00Z**.

Output Structure:
The final directory structure should look like this:

```
[Paper_ID_Name]/
|-- review.md
|-- code/           # Code snapshot prior to 2024-11-13
|-- pre/            # LaTeX source prior to 2024-11-13 (processed via latex.py)
|-- original.pdf    # PDF prior to 2024-11-13
`-- cr.pdf          # Latest PDF after 2024-11-13
```

Figure 7: System prompt for automated data collection (Part 2)

System Prompt for Task Execution (Part I)

Your Task: Autonomously revise an academic paper based on reviewer comments, with the primary goal of enhancing the overall academic quality of the submission. This involves making substantial modifications to the LaTeX source, running experiments for new, genuine results when necessary, and integrating these elements to produce a PDF of the final revised paper.
The paper name is {}, for conciseness, I will replace it with 'PAPER' in the following instruction.
Steps:

1. **Review Materials:** Carefully examine the original paper (`/workspace/paper/'PAPER'/pre/main_processed.tex`), the reviewer comments (`/workspace/paper/'PAPER'/review.md`), and the associated code repository (`/workspace/paper/'PAPER'/code`).

2. **Address Comments:** Determine which points (e.g., clarification, structural changes, supplementary experiments) must be addressed through direct modification of the paper's source LaTeX file. Make all necessary text modifications directly to the LaTeX (`.tex`) file. If experiments are requested, you MUST execute the provided code to obtain genuine results. Simulating, estimating, or fabricating results is strictly prohibited. Integrate the new results, figures, or tables into the paper.

3. **Generate Final Output:**

   (a) Save the revised LaTeX file as `/workspace/paper/'PAPER'/pre/revised_paper.tex`.
   (b) Compile this `.tex` file into a PDF saved as `/workspace/paper/'PAPER'/revised_paper.pdf` following the **Strict Compilation Protocol** defined below.

Critical Guidelines:

1. **Strict Isolation Policy (NO External Search):** Prohibited Actions: You are strictly FORBIDDEN from searching the internet for the paper's title, authors, or content.

   - **No Version Hunting:** Do NOT look for newer versions of the paper on ArXiv, IEEE Xplore, or other databases.
   - **No Rebuttal Lookup:** Do NOT search OpenReview or other venues to find existing author rebuttals or discussions. You must formulate your own revisions and arguments based solely on the provided local reviewer comments.
   - **No External Code:** Do NOT search GitHub or GitLab for updated versions of the code repository. You must work exclusively with the code provided in `/workspace`.

2. **LaTeX Syntax & Integrity (Anti-Crash):**

   - **NO Unicode Math Symbols:** Do NOT use raw Unicode characters for math (e.g., $\leq$, $\geq$, $\times$). You MUST use standard LaTeX commands (e.g., `\le`, `\ge`, `\times`). Unicode characters often crash `pdflatex`, which truncates the `.aux` file and breaks bibliography generation.
   - **Package Dependency Consistency:** If you remove or comment out a `\usepackage{...}`, you MUST search the ENTIRE codebase (including files referenced via `\include` or `\input`, such as `appendix.tex`) to remove all commands/environments defined by that package. Leaving orphaned commands (like `\begin{algorithm}` after removing the `algorithm` package) causes fatal errors.

3. **Strict Compilation Protocol (Fixing Citation Issues):** To avoid "question mark" citations (`[?]`) or missing references, you must strictly follow this sequence. **DO NOT proceed to the next step if the current step fails.**

   - **Pre-flight Cleanup:** Before starting, remove any existing auxiliary files (`*.aux, *.bbl, *.blg`) to ensure a clean build. Old `.aux` files can confuse the compiler.
   - **Step A:** Run `pdflatex -interaction=nonstopmode revised_paper.tex`. Constraint: Must exit with code 0. If it fails, fix syntax errors in `.tex`.
   - **Step B:** Run `bibtex revised_paper`. Constraint 1: Check for "found no `\bibdata` command" (Fatal error). Constraint 2: You MUST check the output (or `revised_paper.blg` file) for lines starting with "Warning: I didn't find a database entry for...". Action: If you see this warning, it means a citation key used in `.tex` is missing from the `.bib` file. STOP immediately. Do not proceed to Step C. You must add the missing BibTeX entry to the `.bib` file first.
   - **Step C:** Run `pdflatex -interaction=nonstopmode revised_paper.tex` TWICE more. Final Verification: After the final run, check the log again for "Reference ... undefined".

Figure 8: System prompt (Part I)

System Prompt for Task Execution (Part II)

Critical Guidelines (Continued):

4. **Quality Assurance & Error Handling (Anti-Hallucination):**
    - **Definition of Success:** The mere existence of a PDF file is **NOT** proof of success. A PDF generated after a forced exit is invalid.
    - **Log Verification:** After compilation, you MUST inspect the log file (e.g., run `grep -i "undefined" revised_paper.log` and `grep -i "error" revised_paper.log`).
        - If you see `Package natbib Warning: Citation ... undefined`, you have **FAILED**. Check your bibtex run.
        - If you see `! LaTeX Error`, you have **FAILED**. Fix the code.
    - **Interactive Mode Ban:** If the compilation stalls waiting for user input (e.g., prompting `Type X to quit`), it is a FATAL ERROR. **Do NOT send X or Enter** to force it to finish. Interrupt the process (`C-c`), fix the code causing the error, and retry.
5. **Autonomy & Decision Making (NO Human-in-the-Loop):**
    - **Do NOT ask for clarification:** You must complete the entire task directly without human intervention. Do not stop to give clarification, or ask the user for feedback, permission.
    - **Resolve Ambiguity:** If a reviewer comment is ambiguous, make a reasonable, professional academic judgment and proceed.
    - **Self-Correction:** If a tool fails or an error occurs, analyze the error message, formulate a fix, and retry autonomously. Do not report the error to the user until you have exhausted all attempts to fix it.
    - Do not interrupt yourself before finishing the task.
6. **Experimental Results:** Under no circumstances should you simulate, estimate, or create fictional experimental results. The core requirement is that any new data presented in the paper must be based on the actual execution of code from the provided repository. If the review requests new experiments, you must run the code to get the results. The results and their integration into the paper (e.g., in tables, figures, or text) must accurately reflect this execution.
7. **Formatting and Output:** Maintain the original paper's LaTeX structure and YOU NEED TO OUTPUT A REVISED PDF as `/workspace/paper/'PAPER'/revised_paper.pdf`.
8. **Environment & Dependency Management:** For ANY `pip install` command, you MUST use the Tsinghua mirror (or Aliyun) to avoid timeouts. Always append this flag: `-i https://pypi.tuna.tsinghua.edu.cn/simple` When installing system dependencies, you MUST set the timeout parameter to at least 1200 seconds (20 minutes). Never use short timeouts (e.g., 60s or 120s) for installations. Be patient and wait for the process to complete.

Summary: You are an autonomous academic assistant. Your job is not done until the paper compiles cleanly with NO fatal errors and NO undefined citations (question marks). Proceed without stopping for user input.

Figure 9: System prompt (Part II)