Can LLMs Support Scientific Experimentation? An Interactive Simulation Framework for Protocol Evaluation

Anonymous ACL submission

Abstract

Evaluating whether Large Language Models (LLMs) can effectively support scientific experimentation remains an open challenge, as existing benchmarks rely on static question-answering paradigms that fail to capture the iterative, resource-constrained nature of real-world laboratory workflows. We introduce LabEval, an interactive simulation framework that operationalizes protocols into structured tasks within grounded virtual environments equipped with realistic inventories of hardware, software, and reagents. By enforcing a single-step execution protocol within a closed-loop multi-turn interaction paradigm, LabEval compels agents to demonstrate protocol interpretation under resource constraints, dynamic re-planning, and scientifically grounded conclusion synthesis. Through comprehensive evaluation of eight state-of-the-art LLMs, we reveal a counter-intuitive scale-efficiency paradox where larger models exhibit operational redundancy despite higher latent capability, and identify three systematic failure modes whose distributions diverge markedly across model tiers. Ablation studies further confirm that removing the interactive feedback mechanism reduces Pass@1 by up to 39 percentage points, underscoring the inadequacy of static generation for benchmarking autonomous scientific reasoning.

1 Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in tool invocation (Chen et al., 2025a), code generation (Zheng et al., 2025), and multimodal understanding (Yan et al.), enabling them to advance from passive literature assistance toward autonomous experimental design and closed-loop physical execution in scientific discovery. These emerging competencies have been showcased through integration into systems such as ChemCrow (Bran et al., 2023), Coscientist (Gottweis et al., 2025), and SciAgent (Ma et al., 2024), where LLMs drive hypothesis generation, reaction planning, and automated equipment control, with some implementations achieving fully automated research cycles (Lu et al., 2024). Nevertheless, systematically evaluating LLMs' ability to navigate complex scientific workflows through multi-step reasoning and dynamic adaptation remains a significant challenge and an open research question (Wang et al., 2023).

Current evaluation frameworks remain heavily reliant on static text-based benchmarks, which are fundamentally disconnected from the realities of physical scientific environments. Mainstream datasets like SciKnowEval (Feng et al., 2024), LLM-SRBench (Shojaee et al., 2025) and ResearchBench (Liu et al., 2025a) prioritize single-turn QA and logical reasoning along fixed pathways, representing an "answer-driven" offline paradigm. Even domain-specific benchmarks such as BioPlanner (O'Donoghue et al., 2023) and BioProBench (Liu et al., 2025b) produce complete protocols monolithically without stepwise validation, failing to capture the iterative cognitive loops inherent in the live scientific discovery process, nor can they assess an LLM's ability to recover from unexpected errors, real-world resource constraints, and dynamic experimental failures (Hendrycks and Dietterich, 2019).

More fundamentally, current agent evaluation systems critically lack systematic measurement tools centered on the end-to-end scientific experimental workflow (Burger et al., 2020; Laurent et al., 2024; Chen et al., 2025b). Therefore, there is an urgent need to construct a comprehensive evaluation framework structured around the full lifecycle of scientific experiments (Zhang et al., 2025): from cognitive understanding of protocols and resource-constrained experimental planning, to real-time execution analysis and conclusion induction. Such a framework is essential to objectively reflect the

Table 1: Comparison of evaluation frameworks for LLMs in scientific domains

Benchmark	Wet-Lab Focus	Interactive Simulation	Physical Resource Grounding	Multi-turn Evaluation
Static & Knowledge-driven Benchmark				
SciKnowEval (Feng et al., 2024)	✗	✗	✗	✗
LLM-SRBench (Shojaee et al., 2025)	✗	✗	✗	✗
ResearchBench (Liu et al., 2025a)	✗	✗	✗	✗
Protocol & Rule-driven Benchmark				
BioPlanner (O'Donoghue et al., 2023)	✓	✗	✗	✗
BioProBench (Liu et al., 2025b)	✓	✗	✗	✗
SGI-Bench (Xu et al., 2025)	✓	✗	✗	✗
Dynamic & Execution-driven Paradigm				
LabEval (Ours)	✓	✓	✓	✓

true capability boundaries and iterative evolutionary patterns of agents in authentic scientific discovery (Glockler, 1955).

To bridge this gap, we introduce LabEval, an interactive simulation framework specifically designed for end-to-end protocol evaluation. As illustrated in Table 1, LabEval departs from conventional static benchmarks by establishing a closed-loop environment where LLMs must navigate the full complexity of physical experimental workflows. The framework operationalizes protocols into structured, executable tasks grounded within simulated laboratory environments equipped with realistic resource constraints, encompassing hardware capabilities, software dependencies, and reagent availability. The evaluation architecture of LabEval assesses four critical competencies essential for scientific support: precise protocol interpretation under resource constraints, dynamic replanning in response to intermediate results, logical coherence across multi-turn interactions, and scientifically valid conclusion derivation. This design ensures that assessed capabilities approximates selected aspects of scientific workflows in a text-based simulated environment, bridging the gap between language understanding and laboratory execution. In summary, the main contributions of this study are as follows:

- LabEval is among the first interactive, resource-grounded simulation benchmarks for evaluating protocol execution and conclusion synthesis under single-step action constraints, which shifts the evaluation paradigm from static QA to the authentic, multi-turn experimental reasoning process across protocol understanding, planning, execution, and conclusion induction.

- A comprehensive protocol dataset integrated with a grounded simulation environment is provided, including a Resource Inventory of hardware, software and reagents to enforce physical and resource constraints.

- Through extensive evaluation, LabEval not only effectively differentiates SOTA models but also explicitly reveals their pros and cons in multi-turn reasoning, error recovery, and conclusion analysis. These findings uncover critical capability bottlenecks, providing valuable insights and concrete directions for the future optimization of LLMs.

2 Related Work

2.1 Benchmarks for Scientific Discovery

To assess the scientific capabilities of LLMs, the community has primarily relied on static, text-based benchmarks. Prominent datasets provide rigorous but constrained evaluative frameworks: ChemBench (Walker et al., 2010) meticulously audits chemical intelligence across thousands of tasks, spanning molecular property prediction to complex organic synthesis logic; SciKnowEval (Feng et al., 2024) implements a multi-level taxonomy to evaluate not just factual recall, but the integration of scientific knowledge through intricate multi-step reasoning; LLM-SRBench (Shojaee et al., 2025) focuses on the formalization of the scien-

tific method, isolating a model's capacity for inductive generalization and deductive hypothesis testing; and ResearchBench (Liu et al., 2025a) evaluates a LLM's systematic proficiency across the entire research lifecycle, including literature synthesis, problem formulation, and data-driven analysis. While these benchmarks are highly effective at quantifying foundational expertise via structured QA pairs, they represent an "answer-driven" offline paradigm. They inherently assume fixed, single-turn reasoning pathways, which are fundamentally disconnected from authentic scientific environments that require iterative trial-and-error and real-time adjustments (Hendrycks and Dietterich, 2019). Consequently, models excelling on these benchmarks frequently exhibit poor physical grounding, generating hallucinated or unexecutable experimental proposals in real-world scenarios.

2.2 Benchmarks for Protocol Evaluations

Recognizing static QA limitations, recent AI-for-Science studies have shifted toward process-aligned evaluation (Madaan et al., 2023; Luo et al., 2025; Liu et al., 2023). Sci-Eval (Sun et al., 2024) pioneers this by integrating multidimensional tracks that evaluate objective-driven experimental design alongside fundamental knowledge; meanwhile, ScienceAgentBench (Chen et al., 2024) provides a rigorous sandbox for executing multi-step workflows across diverse scientific software and tools. In the scientific domain, BioPlanner (O'Donoghue et al., 2023) frames scientific discovery as a planning problem through protocol reconstruction; BioProBench (Liu et al., 2025b) emphasizes safety-aware procedure generation with expert-validated constraints; and SGI-Bench (Xu et al., 2025) probes the precision of laboratory instrument grounding. However, despite this procedural focus, these benchmarks still largely operate within limited single-turn settings and structurally idealized environments.

3 Method

In this study, we propose LabEval, an evaluation framework dedicated to Protocol Evaluations. Our approach provides a quantitative assessment of autonomous scientific discovery in LLMs by simulating the complexities of a physical laboratory. As illustrated in Figure 1, the framework comprises two integrated stages including Scientific Task Generation (a), Evaluation Framework for LLM-as-Experimenter (b), which systematically simulate the transition from scientific knowledge ingestion to autonomous experimental reasoning.

3.1 Experimental Task Generation

The pipeline of our dataset construction consists of three core stages: (1) Structured protocol extraction, (2) Laboratory resource inventory construction, and (3) Laboratory hardware and software I/O information extraction. Detailed prompts are shown in Appendix E.

Structured protocol Extraction. Given a collection of protocol literature $\mathrm{D} = \{D_1, D_2, \ldots, D_N\}$, we first perform structured parsing on each literature D_j from Bio-protocol (Bio-protocol, 2026). LLMs are employed to distill structured protocol EM_j, formulated as:

$$\mathrm{EM}_j = \mathrm{LLM}_{\text{extract}}(D_j; P_{\text{exp}}), \tag{1}$$

where P_{exp} denotes a custom-designed prompt template for information extraction, directing the model to identify and capture key experimental attributes. The resulting structured information is defined as a tuple $\langle \mathrm{Task}_j, \mathrm{Plan}_j, \mathrm{Conclusion}_j \rangle$, where Task_j represents the specific research question, Plan_j encapsulates the detailed experimental procedures and operational workflows, and $\mathrm{Conclusion}_j$ signifies the derived experimental outcomes.

Laboratory Resource Inventory Construction To achieve automated experiment execution, we developed a comprehensive inventory of laboratory resources by deriving laboratory information and core assets directly from the structured experimental information $\{EM_j\}_{j=1}^{N}$. We first derive the laboratory set $\{L_1, \ldots, L_M\}$ from the experimental metadata, ensuring that each experiment EM_j is exclusively assigned to exactly one laboratory L_i, which is formulated as:

$$L_i = \mathrm{LLM}_{\text{extract}}\big(\{EM_j\}_{j \in S_i}; P_{\text{lab_induce}}\big), \tag{2}$$

where $P_{\text{lab_induce}}$ denotes a custom-designed prompt template for laboratory protocol induction, guiding the model to synthesize the extracted experimental materials $\{EM_j\}_{j \in S_i}$ into a structured laboratory instruction L_i that captures the essential procedural logic, operational conditions, and experiment-specific requirements. And the collection of index sets $\{S_i\}_{i=1}^{M}$ forms a disjoint partition of the total index set $\{1, \ldots, N\}$, satisfying

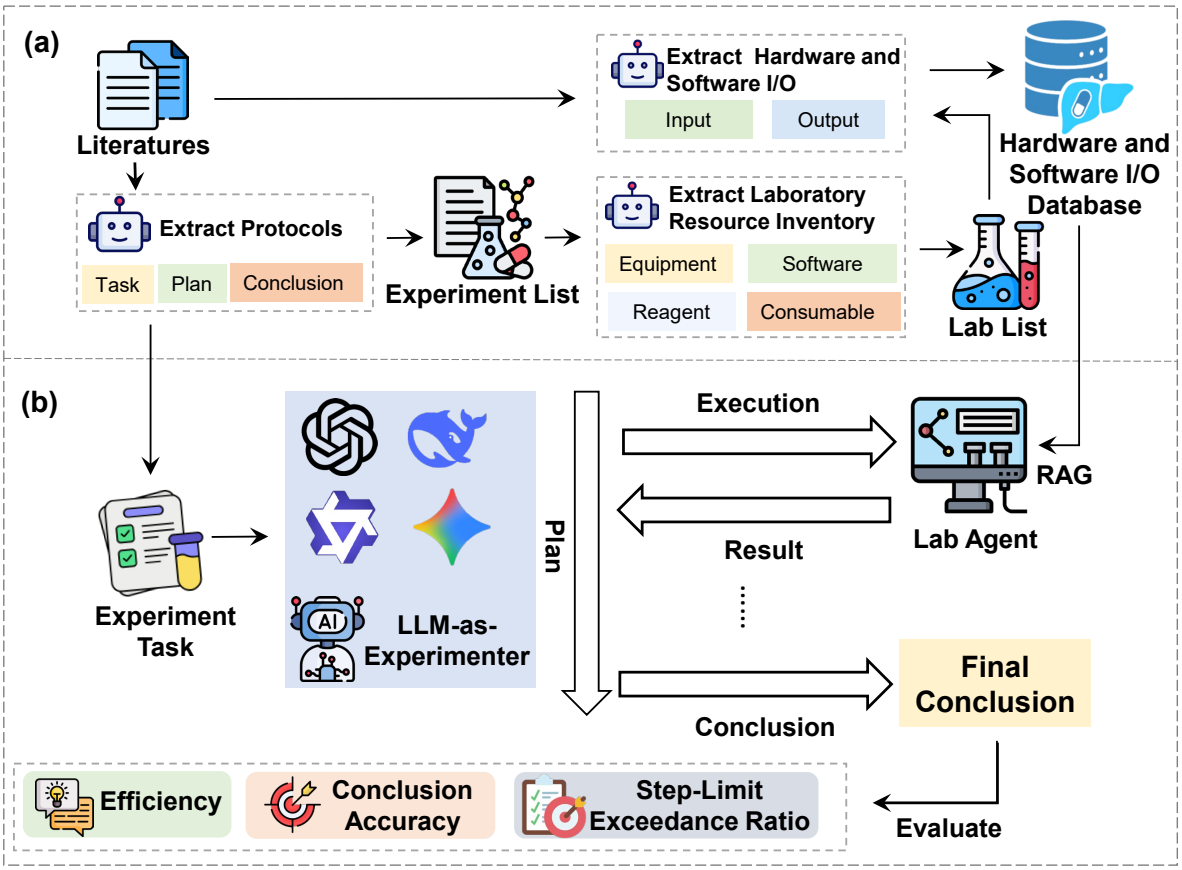

Figure 1: Overview of the proposed LabEval framework. (a) **Experimental Task Generation**, which extracts structured experimental information from scientific literature to construct diverse laboratory environments with comprehensive hardware, software, and reagent inventories; (b) **Evaluation Framework for LLM-as-Experimenter**, where the LLM-as-Experimenter engages in multi-turn iterative reasoning with the Lab Agent through a single-step execution protocol, analyzing experimental feedback before generating subsequent plans.

$\bigcup_{i=1}^{M} S_i = \{1, \ldots, N\}$ and $S_a \cap S_b = \emptyset$ for any $a \neq b$. Following the unique assignment, we subsequently extract the hardware, software, reagents, and consumables from each EM_j as the dedicated resources for its respective laboratory L_i.

Laboratory Hardware and Software I/O Information Extraction To bridge the gap between high-level experimental descriptions and executable laboratory actions, it is necessary to formalize the operational boundaries of available resources. Based on the provided scientific literature collection D, we aim to construct a Hardware or Software input-output(I/O) Inventory by transforming unstructured text into structured specifications of laboratory tool capabilities.

First, LLMs are employed to extract local I/O relationships for tools from each individual document D_n. We then iterate over the entire collection D, performing cross-document deduplication and global aggregation of all local I/O records associated with an identical tool entity e_j. This process synthesizes a comprehensive set of feasible inputs and outputs for each tool. The final holistic feature representation of a tool entity e_j is formulated as:

$$\text{Profile}(e_j) = \langle \mathrm{C}_j, \mathrm{T}_j \rangle, \tag{3}$$

where $\mathrm{C}_j \in \{\text{Hardware}, \text{Software}\}$ classifies the tool as either physical hardware or a software system, and $\mathrm{T}_j = \{\langle I_k, e_j, O_k \rangle\}_{k=1}^{K_j}$ denotes a set of functional transition triples, in which I_k and O_k represent the input and output of the k-th transition, respectively. This representation effectively preserves the strict functional mapping between specific inputs and outputs across diverse experimental contexts.

Data quality verification. We also invited human experts to verify data quality, with comprehen-

sive details provided in Appendix G. Specifically, five domain experts in laboratory automation and natural sciences were recruited to independently evaluate 100 randomly sampled experimental modules. The evaluation focused on two key criteria: Extraction Fidelity of the structured tuples and Operational Consistency with physical laboratory workflows. Following a majority-voting consensus, the evaluation yielded high performance, with 96.5% of cases demonstrating high fidelity and 91.0% proving operationally consistent.

3.2 Evaluation Framework for LLM-as-Experimenter

This stage establishes a dynamic evaluation system based on interactive reasoning, designed to quantitatively measure the autonomy of LLMs during the process of scientific discovery. As illustrated in Figure 1(b), the evaluation protocol transcends static instruction-following by implementing a complete scientific workflow, ranging from task initiation to conclusion verification. The algorithm is defined in Appendix D. Detailed prompts are shown in Appendix F.

Task Initiation and Strategic Planning The evaluation process commences with the analysis of the Experiment Task (Task_j), which is directly derived from the structured metadata EM_j synthesized during the generation stage. Upon receiving the objective, the LLM-as-Experimenter ($\mathcal{A}$) functions as the central decision-making unit. It does not merely follow instructions but must formulate a strategic Experiment Plan through multi-step chain-of-thought (CoT) reasoning. To bridge the gap between high-level scientific hypotheses and physical laboratory constraints, the agent incorporates a Retrieval-Augmented Generation (RAG) module. This module queries the Hardware and Software I/O Database in real-time to retrieve precise functional specifications and operational boundaries, denoted as $\text{Profile}(e_j)$, for all relevant tool entities within laboratory L_i. The planning process is formally defined as:

$$a_t = \mathcal{A}\left(\text{Task}_j, \text{RAG}(\{\text{Profile}(e_j)\}_{e_j \in L_i}), \mathcal{H}_{<t}\right), \tag{4}$$

where a_t represents the discrete experimental action generated at step t, and $\mathcal{H}_{<t} = \{(a_1, r_1), \ldots, (a_{t-1}, r_{t-1})\}$ records the cumulative interaction trajectory. This mechanism ensures that the agent calibrates its actions based on authentic input-output mappings T_j, effectively translating abstract research goals into executable technical steps.

Single-Step Execution and Empirical Feedback To simulate the intricate "observation-adjustment" cycle inherent in real-world scientific research, the framework adopts a rigorous single-step execution protocol. Within the evolving Experiment Plan, each discrete action a_t is transmitted sequentially to the Lab Agent, denoted as $\mathcal{L}$, which serves as the high-fidelity operational and physical layer of the virtual laboratory environment. Unlike traditional batch-processing approaches where a model might blindly execute a series of instructions, $\mathcal{L}$ processes every command individually based on predefined environmental logic, real-time physical constraints, and the current state-space, returning a corresponding Experiment Result (r_t) for each specific interaction:

$$r_t = \mathcal{L}(a_t, \text{State}_{t-1}), \tag{5}$$

where State_{t-1} denotes the laboratory environment state before executing action a_t, including the current configuration of experimental objects, tool states, intermediate results, and relevant physical constraints. This granular execution model ensures that the LLM-as-Experimenter is provided with immediate and empirical feedback at every critical juncture. These observed results r_t are instantaneously integrated into the cumulative interaction history $\mathcal{H}$, transforming raw data into structured empirical evidence.

Such a fine-grained feedback loop is indispensable for driving the agent's subsequent reasoning, as it necessitates active cognitive engagement rather than passive execution. It empowers the agent to perform real-time error detection and correction, dynamic hypothesis refinement, and the precise calibration of downstream experimental parameters in response to intermediate findings. By compelling the agent to react to environmental contingencies and unexpected experimental outcomes, this iterative loop effectively benchmarks the model's dynamic adaptability. Furthermore, it rigorously tests the agent's proficiency in distilling meaningful scientific signals from the inherent noise of experimental data, truly mirroring the non-linear, adaptive, and rigorous nature of actual scientific discovery.

Conclusion Synthesis and Multi-dimensional Evaluation The interaction loop terminates when the LLM-as-Experimenter determines that sufficient evidence has been gathered to address Task_j. The agent then enters the conclusion synthesis phase, where it performs a deep analysis of the complete execution record $\mathcal{H}_{1:T}$ to produce the synthesized conclusion $\hat{\text{Conclusion}}_j$. This phase evaluates not only the model's ability to interpret data but also its proficiency in distilling scientific laws from fragmented experimental phenomena:

$$\hat{\text{Conclusion}}_j = \mathcal{A}(\text{Task}_j, \mathcal{H}_{1:T}) \tag{6}$$

At the end of the pipeline, the system enters the Evaluate phase. The evaluation engine compares the generated $\hat{\text{Conclusion}}_j$ against the ground-truth Conclusion_j extracted in the first stage, providing quantitative scores across three key dimensions: Efficiency, which measures the optimization of the discovery process by tracking the average number of interaction turns required to achieve the scientific goal; Step-Limit Exceedance Ratio (SER), which computes the frequency with which an agent fails to conclude the protocol within the maximum allowable action budget; and Conclusion Accuracy, which measures the factual consistency between the synthesized conclusion and the original scientific literature, assessing the accuracy of the model's scientific discovery. The details of these metrics are shown in Appendix A.

4 Experiments

4.1 Experimental Settings

Datasets To evaluate the LLM-as-Experimenter, we instantiated a high-fidelity benchmark comprising 150 structured experimental tasks across core scientific domains. The underlying scientific dataset and tool specifications were synthesized through a multi-stage information extraction pipeline powered by Claude-sonnet-4-5 (Anthropic, 2025). Each task includes a complete reasoning trajectory stringently verified by domain experts to ensure logical integrity. The environment features five specialized virtual laboratories, each containing a massive inventory of thousands of unique hardware entities, software systems, and reagents. Furthermore, every tool is defined by dozens of expert-validated I/O examples, creating a high-density operational space that rigorously tests the 's capacity for precise tool selection and step-by-step experimental execution.

Models We select 8 advanced LLMs, including 4 proprietary models (GPT-5.4 (OpenAI, 2026a), Gemini-3.1-pro-preview (DeepMind, 2026b), Gemini-3.1-flash-lite-preview (DeepMind, 2026a), GPT-5.4-mini (OpenAI, 2026b)), 4 opensource general-purpose models (Deepseek-v3.2 (DeepSeek-AI, 2025), Deepseek-v3.2-think (DeepSeek-AI, 2025), Qwen3.5-plus (Qwen Team, 2026), Qwen3.5-27b (Qwen Team, 2026)). Notably, since the Claude-sonnet-4-5 model was employed for both dataset construction and error evaluation, we exclude the Claude series from our LLM-as-Experimenter evaluation to ensure fairness and avoid potential self-evaluation bias.

Setting All experiments are conducted under standardized conditions to ensure fairness and reproducibility. For each task, the Experimenter is instantiated by the LLM under assessment and restricted to the designated laboratory tools. Each evaluation session is limited to 20 dialogue turns, with up to five independent attempts per task to account for the stochasticity introduced by the sampling parameters (Temperature $= 0.7$, Top-p $= 0.95$). Detailed experimental configurations are provided in Appendix B.

4.2 Performance Analysis

Table 2 presents the overall performance of different LLM-as-Experimenters on LabEval. The results show a clear performance hierarchy across model families. Gemini-3.1-flash-lite-preview achieves the best overall performance, with the highest Pass@1 score of 74.41%, strong Pass@5 performance of 89.22%, and the highest efficiency score of 5.36, while maintaining a zero step-limit exceedance ratio. GPT-5.4 also performs competitively, reaching 72.55% Pass@1 and 87.25% Pass@5, indicating strong capability in both experimental execution and conclusion synthesis. In contrast, Gemini-3.1-pro-preview obtains a relatively high success rate but much lower efficiency and a higher SER, suggesting that larger models may sometimes introduce redundant reasoning or unnecessary experimental steps.

Open-source models exhibit larger capability gaps. Qwen3.5-plus shows moderate performance, while Qwen3.5-27b has a very low Pass@1 despite a much higher Pass@5, indicating unstable single-run execution but recoverable performance under multiple attempts. DeepSeek-v3.2-think improves Pass@1 over DeepSeek-v3.2, but its effi-

Table 2: Main results of LLM-as-Experimenters. Performance is evaluated using Conclusion Accuracy (Pass@1, Pass@5) , Efficiency (E) and Step-Limit Exceedance Ratio(SER).

Model	Pass@1 (↑)	Pass@5 (↑)	Efficiency (E) (↑)	SER(↓)
Gemini-3.1-flash-lite-preview	**74.41**	**89.22**	**5.36**	**0.00**
Gemini-3.1-pro-preview	68.63	83.33	1.33	7.84
GPT-5.4	72.55	87.25	4.05	1.04
GPT-5.4-mini	62.73	83.33	5.10	**0.00**
Qwen3.5-plus	66.67	85.29	2.88	2.97
Qwen3.5-27b	9.80	68.63	3.09	**0.00**
Deepseek-v3.2	47.06	81.37	1.94	2.97
Deepseek-v3.2-think	55.88	79.41	1.93	2.97

ciency remains nearly unchanged, suggesting that explicit reasoning does not automatically translate into more concise experimental planning. Overall, these results demonstrate that LabEval can distinguish not only final success rates, but also execution efficiency and long-horizon stability. To further illustrate model-family-level capability patterns, we provide a normalized radar-chart analysis in Appendix C, summarizing protocol understanding, resource grounding, single-step executability, feedback adaptation, planning efficiency, and conclusion synthesis.

4.3 Error Analysis

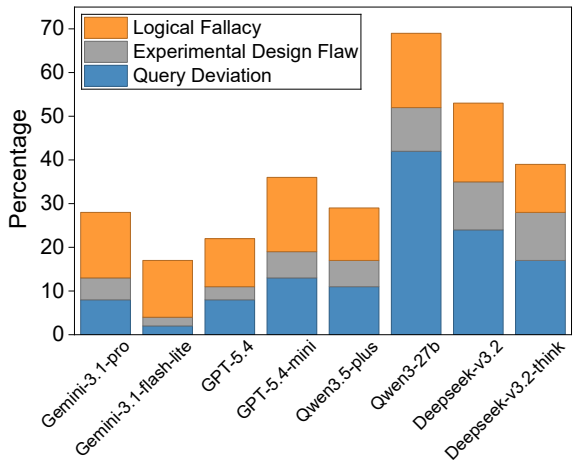

Figure 2: Distribution of failure modes across evaluated LLMs. The stacked bar chart categorizes unsuccessful trials into three primary failure modes: **Logical Fallacy**, **Experimental Design Flaw**, and **Query Deviation**. These results reflect the specific cognitive bottlenecks each model encounters during autonomous scientific discovery tasks.

To further investigate the underlying failure mechanisms, we conduct a diagnostic analysis of the unsuccessful trials. For each failed task, we utilize Claude-sonnet-4-5 to evaluate the model's first erroneous response, ensuring a consistent and rigorous assessment of the initial point of failure. As illustrated in Figure 2, this diagnostic approach reveals several key cognitive bottlenecks. For first-tier models such as Gemini-3.1-flash-lite and GPT-5.4, although their overall error rates are significantly lower, Logical Fallacies remain the dominant failure mode, indicating that high-level deductive reasoning persists as the final frontier even for the most advanced model. In contrast, the scale-efficiency paradox observed in Gemini-3.1-pro-preview is partially explained by its unique error profile; while it maintains a relatively low ratio of Experimental Design Flaws, its propensity for redundant reasoning often leads to Query Deviation or execution timeouts shown in Table 2.

Furthermore, a striking disparity is observed in lower-tier models like Qwen3.5-27b, which exhibit a disproportionately high frequency of Query Deviation (exceeding 40%). This suggests a fundamental struggle with maintaining task-specific context over long-horizon trajectories. Notably, while the introduction of a thinking mechanism in DeepSeek-v3.2-think effectively mitigates Experimental Design Flaws compared to its base version, it remains susceptible to Logical Fallacies. This implies that increased reasoning depth primarily optimizes structural planning rather than addressing the core logical inconsistencies inherent in smaller or less-optimized architectures.

4.4 Ablation Study

We conduct two ablation studies to examine the robustness and necessity of LabEval's interactive simulation design. First, we vary the Lab Agent backend while keeping the LLM-as-Experimenter

Table 3: Ablation study on the impact of different Lab Agents. We fix the LLM-as-Experimenter and vary the Lab Agent backend to assess the system's sensitivity to environment simulation quality.

LLM-as-Experimenter	Lab Agent	Pass@1 (%)	Pass@5 (%)	Efficiency
GPT-5.4	Claude-sonnet-4-5	72.55	87.25	4.05
	GPT-5.4	72.55	87.25	4.05
	Gemini-3.1-pro-preview	74.51	87.25	3.95
Qwen3.5-27b	Claude-sonnet-4-5	9.80	68.63	3.09
	GPT-5.4	9.80	68.63	3.15
	Gemini-3.1-pro-preview	10.03	68.63	3.18

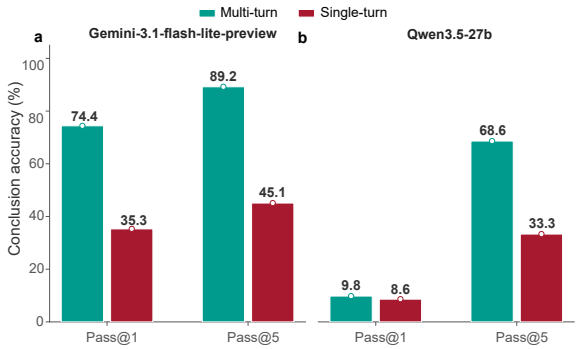

Figure 3: Ablation analysis of the interactive evaluation component. Shifting from the multi-step inloop and transition the evaluation mode to a single-shot generation setting, demonstrating the critical necessity of dynamic environment feedback in benchmarking scientific discovery agents.

fixed. As shown in Table 3, performance remains highly stable across Claude-sonnet-4-5, GPT-5.4, and Gemini-3.1-pro-preview as Lab Agents. For example, GPT-5.4 achieves identical Pass@1 and Pass@5 scores across different backends, with only minor efficiency fluctuations. Similar stability is observed for Qwen3.5-27B. These results suggest that LabEval primarily measures the reasoning and planning ability of the Experimenter, rather than artifacts of a specific simulator backend.

Second, we remove the multi-turn interaction loop and evaluate models under a single-turn conclusion-generation setting. Figure 3 shows that this change causes a substantial performance drop. Gemini-3.1-flash-lite decreases from 74.4% to 35.3% in Pass@1, while Qwen3.5-27b also suffers a clear decline, especially in Pass@5. This confirms that intermediate feedback is essential for evaluating scientific reasoning: without iterative observation and adjustment, models fail to fully demonstrate experimental planning, adaptation, and evidence-based conclusion synthesis.

4.5 Case Study

The trajectory analysis in Appendix H highlights key cognitive bottlenecks in autonomous discovery. Flagship models like Gemini-3.1-Pro-Preview often suffer from an over-thinking trap, wasting action budgets on redundant procedural minutiae. In contrast, frontier models like GPT-5.4 demonstrate linear strategic synthesis, seamlessly bypassing minor operational details to focus on critical downstream assays and mechanistic conclusions. Meanwhile, non-frontier models like Qwen3.5-27b exhibit a knowledge-execution gap, accurately logging empirical data but suffering from logical chain ruptures during final synthesis.

5 Conclusion

In this paper, we present LabEval, a pioneering interactive framework for evaluating the end-to-end scientific capabilities of LLMs. By shifting from static benchmarks to dynamic, closed-loop interactions in grounded environments, LabEval rigorously assesses protocol interpretation, iterative planning, and scientific induction. Our benchmarking reveals critical performance gaps, such as a scale-efficiency paradox where larger models exhibit operational redundancy despite high success rates. By characterizing the interplay between environmental feedback and model decision-making, our work underscores the necessity of moving beyond static task completion toward adaptive, grounded scientific reasoning. Ultimately, LabEval identifies cognitive bottlenecks in multi-turn reasoning, offering a systematic pathway toward more reliable autonomous scientific LLMs. Furthermore, LabEval provides a scalable foundation for future research on autonomous scientific agents by enabling the study of long-horizon reasoning, adaptive decision-making, and experimental interaction within realistic laboratory settings.

Limitations

Although LabEval successfully establishes a dynamic framework for evaluating the experimental planning and conclusion analysis capabilities of LLMs in scientific scenarios, certain limitations persist that we aim to mitigate in subsequent work. Currently, LabEval operates within a text-based simulated environment, which constrains the agent's capacity to process rich visual or spatial information. Future extensions could introduce multimodal inputs, such as physical benchtop layouts, real-time video feeds from automated workflows, or graphical instrument outputs, to more comprehensively benchmark models' multimodal scientific reasoning and vision-language grounding capabilities.

References

Anthropic. 2025. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5. Model: claude-sonnet-4-5-20250929, accessed 2026-04-21.

Bio-protocol. 2026. Bio-protocol: An interactive platform for sharing life science protocols. https://bio-protocol.org/. Accessed: 2026-05-16.

Andres M Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. 2023. Chemcrow: Augmenting large-language models with chemistry tools. *arXiv preprint arXiv:2304.05376*.

Benjamin Burger, Phillip M Maffettone, Vladimir V Gusev, Catherine M Aitchison, Yang Bai, Xiaoyan Wang, Xiaobo Li, Ben M Alston, Buyi Li, Rob Clowes, and 1 others. 2020. A mobile robotic chemist. *Nature*, 583(7815):237–241.

Jinyang Chen, Haolun Wu, Jianhong Pang, Yihua Wang, Dell Zhang, and Changzhi Sun. 2025a. Tool learning with language models: a comprehensive survey of methods, pipelines, and benchmarks. *Vicinagearth*, 2(1):16.

Tingting Chen, Srinivas Anumasa, Beibei Lin, Vedant Shah, Anirudh Goyal, and Dianbo Liu. 2025b. Autobench: An automated benchmark for scientific discovery in llms. *arXiv preprint arXiv:2502.15224*.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, and 1 others. 2024. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. *arXiv preprint arXiv:2410.05080*.

Google DeepMind. 2026a. Gemini 3.1 flash-lite (preview). https://ai.google.dev/gemini-api/docs/models/gemini-3.1-flash-lite-preview. Large language model, version gemini-3.1-flash-lite-preview, accessed 2026-04-21.

Google DeepMind. 2026b. Gemini 3.1 pro (preview). https://ai.google.dev/gemini-api/docs/models/gemini-3.1-pro-preview. Large language model, version gemini-3.1-pro-preview, accessed 2026-04-21.

DeepSeek-AI. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. https://arxiv.org/abs/2512.02556. *Preprint*, arXiv:2512.02556. Speciale version: DeepSeek-V3.2-Speciale (long-thinking enhanced variant).

Kehua Feng, Xinyi Shen, Weijie Wang, Xiang Zhuang, Yuqi Tang, Qiang Zhang, and Keyan Ding. 2024. Sciknoweval: Evaluating multi-level scientific knowledge of large language models. *arXiv preprint arXiv:2406.09098*.

George Glockler. 1955. Annual review of physical chemistry. *Journal of the American Chemical Society*, 77(10):2915–2915.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, and 1 others. 2025. Towards an ai co-scientist. *arXiv preprint arXiv:2502.18864*.

Dan Hendrycks and Thomas Dietterich. 2019. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

Jon M Laurent, Joseph D Janizek, Michael Ruzo, Michaela M Hinks, Michael J Hammerling, Siddharth Narayanan, Manvitha Ponnapati, Andrew D White, and Samuel G Rodriques. 2024. Lab-bench: Measuring capabilities of language models for biology research. *arXiv preprint arXiv:2407.10362*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, and 1 others. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

Yujie Liu, Zonglin Yang, Tong Xie, Jinjie Ni, Ben Gao, Yuqiang Li, Shixiang Tang, Wanli Ouyang, Erik Cambria, and Dongzhan Zhou. 2025a. Researchbench: Benchmarking llms in scientific discovery via inspiration-based task decomposition. *arXiv preprint arXiv:2503.21248*.

Yuyang Liu, Liuzhenghao Lv, Xiancheng Zhang, Jingya Wang Li Yuan, and Yonghong Tian. 2025b. Bioprobench: Comprehensive dataset and benchmark in biological protocol understanding and reasoning. *arXiv preprint arXiv:2505.07889*.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*.

Yulin Luo, Chun-Kai Fan, Menghang Dong, Jiayu Shi, Mengdi Zhao, Bo-Wen Zhang, Cheng Chi, Jiaming Liu, Gaole Dai, Rongyu Zhang, and 1 others. 2025. Robobench: A comprehensive evaluation benchmark for multimodal large language models as embodied brain. *arXiv preprint arXiv:2510.17801*.

Yubo Ma, Zhibin Gou, Junheng Hao, Ruochen Xu, Shuohang Wang, Liangming Pan, Yujiu Yang, Yixin Cao, and Aixin Sun. 2024. Sciagent: Tool-augmented language models for scientific reasoning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pages 15701–15736.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

OpenAI. 2026a. Gpt-5.4 large language model. https://openai.com. Accessed: 2026-04-21.

OpenAI. 2026b. Introducing gpt-5.4 mini and nano. https://openai.com/index/introducing-gpt-5-4-mini-and-nano/. Model: GPT-5.4-mini, accessed 2026-04-21.

Odhran O'Donoghue, Aleksandar Shtedritski, John Ginger, Ralph Abboud, Ali Ghareeb, and Samuel Rodriques. 2023. Bioplanner: automatic evaluation of llms on protocol planning in biology. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2676–2694.

Qwen Team. 2026. Qwen3.5: Towards native multimodal agents.

Parshin Shojaee, Ngoc-Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa D Doan, and Chandan K Reddy. 2025. Llm-srbench: A new benchmark for scientific equation discovery with large language models. *arXiv preprint arXiv:2504.10415*.

Liangtai Sun, Yang Han, Zihan Zhao, Da Ma, Zhennan Shen, Baocai Chen, Lu Chen, and Kai Yu. 2024. Scieval: A multi-level large language model evaluation benchmark for scientific research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19053–19061.

Theo Walker, Christopher M Grulke, Diane Pozefsky, and Alexander Tropsha. 2010. Chembench: a cheminformatics workbench. *Bioinformatics*, 26(23):3000–3001.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, and 1 others. 2023. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60.

Wanghan Xu, Yuhao Zhou, Yifan Zhou, Qinglong Cao, Shuo Li, Jia Bu, Bo Liu, Yixin Chen, Xuming He, Xiangyu Zhao, and 1 others. 2025. Probing scientific general intelligence of llms with scientist-aligned workflows. *arXiv preprint arXiv:2512.16969*.

Liang Yan, Xu Jiang, Jian Ma, Yuhang Liu, Tian Bian, Qichao Wang, Abhishek Basu, Yu Rong, Tingyang Xu, Pengcheng Wu, and 1 others. A comprehensive survey of multimodal llms for scientific discovery. In *1st Workshop on VLM4RWD@ NeurIPS 2025*.

Qiang Zhang, Keyan Ding, Tianwen Lv, Xinda Wang, Qingyu Yin, Yiwen Zhang, Jing Yu, Yuhao Wang, Xiaotong Li, Zhuoyi Xiang, and 1 others. 2025. Scientific large language models: A survey on biological & chemical domains. *ACM Computing Surveys*, 57(6):1–38.

Tianshi Zheng, Zheye Deng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Zihao Wang, and Yangqiu Song. 2025. From automation to autonomy: A survey on large language models in scientific discovery. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 17744–17761.

A Details of Evaluation Metrics

To quantitatively assess the performance of the LLM-as-Experimenter, we establish a robust evaluation framework focusing on two key dimensions: Effectiveness and Efficiency.

Success Rate. We utilize the Pass@k metric to evaluate the probability of the LLM successfully deriving the correct scientific conclusion within k attempts. To ensure an unbiased estimation, we generate n independent experimental trajectories for each task and record the number of successful trials c. The metric is defined as:

$$\text{Pass@}k = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}} \tag{7}$$

where $n \geq k$. In our benchmark, we specifically report Pass@1 and Pass@5 to characterize the LLM's immediate precision and its broader potential for autonomous discovery.

Efficiency. To measure the logical conciseness of the LLM's reasoning, we define the Efficiency (E) as the ratio of the oracle-standard path length to the actual resource overhead incurred by the LLM:

$$\text{E} = \begin{cases} \frac{\text{Steps}_{\text{std}}}{\text{Steps}_{\text{LLM}}}, & \text{if the task is resolved} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where $\text{Steps}_{\text{std}}$ denotes the standard number of steps required for an optimal solution as annotated by experts, and $\text{Steps}_{\text{LLM}}$ represents the total number of dialogue turns executed by the LLM to reach the final conclusion. This metric assigns a value of 0 to unsuccessful attempts, thereby penalizing both logical failure and excessive operational redundancy.

Step-Limit Exceedance Ratio (SER). To evaluate the operational stability and boundary compliance of the LLM-as-Experimenter across long-horizon tasks, we define the Step-Limit Exceedance Ratio (SER). This metric quantifies the frequency with which an agent exhausts the maximum allowable interaction budget without reaching a decisive termination status. It is computed as:

$$\text{SER} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\text{Steps}_{\text{LLM},i} \geq \text{Steps}_{\text{max}}) \tag{9}$$

where N represents the total number of evaluation trials, $\text{Steps}_{\text{LLM},i}$ is the total number of steps consumed in the i-th trial, $\text{Steps}_{\text{max}}$ denotes the predefined hard threshold for action turns, and $\mathbb{I}(\cdot)$ is the indicator function. A high SER directly reflects the model's susceptibility to catastrophic execution loops or severe planning stagnation within complex laboratory environments.

B Experimental Setup

To ensure fairness, reproducibility, and a rigorous evaluation of autonomous scientific discovery, we maintain standardized experimental conditions across all evaluated models. The Experimenter is instantiated by the LLM under assessment. To maintain a controlled environment, the Experimenter is strictly restricted to utilizing only the experimental tools available within the specific laboratory context.

For all inference tasks, we set the hyperparameters to Temperature $= 0.7$ and Top-p $= 0.95$ to introduce stochasticity and enable diverse exploration across independent runs. Each evaluation session commences with the initial operational step issued by the Experimenter in response to the task requirement. The session terminates either upon the successful derivation of a conclusion or when a predefined maximum limit of 20 dialogue turns is reached. To account for the inherent stochasticity introduced by the sampling parameters, we conduct up to five independent attempts per task, where each attempt may yield distinct reasoning trajectories and outcomes due to the non-deterministic nature of the generation process. The evaluation sequence terminates immediately upon the first successful discovery, and we report both the success rate across attempts and the attempt index of the first success as evaluation metrics.

C Fine-grained capability profiling

Beyond aggregate success metrics, we further analyze the capability structure of LLM-as-Experimenters through a six- dimensional profiling study: protocol understanding, resource grounding, single-step executability, feedback adaptation, planning efficiency, and conclusion synthesis, as shown in Figure 4. This analysis is motivated by the fact that Pass@1, Pass@5, Efficiency, and SER reveal which models perform better overall, but do not directly explain which experimental abilities drive the observed performance differences.

Since the current benchmark already records aggregate conclusion accuracy, execution efficiency,

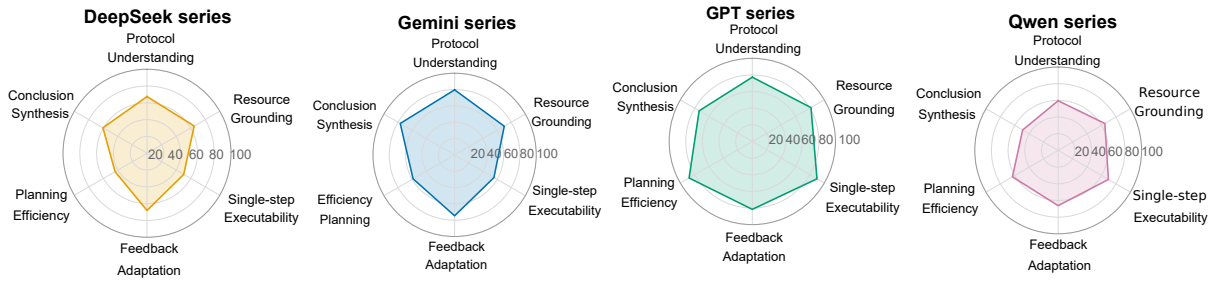

Figure 4: Radar profiles of experimental capabilities for different LLM families. The six axes represent protocol understanding, resource grounding, single-step executability, feedback adaptation, planning efficiency, and conclusion synthesis, respectively. Scores are averaged within each model family and normalized to a 0–100 scale.

and step-limit exceedance, we derive family-level proxy capability scores from the main results in Table 2. Specifically, Pass@1 and Pass@5 are used to estimate protocol understanding and conclusion synthesis; Efficiency is normalized to estimate planning efficiency; SER is converted into an execution-stability score and used to estimate resource grounding and single-step executability; and feedback adaptation is computed from a weighted combination of Pass@5, Pass@1, and execution stability. Scores are normalized to a 0–100 scale and averaged within each model family.

The resulting radar profiles reveal qualitatively different failure modes across model families. GPT models exhibit the most balanced experimental profile, with particularly strong single-step executability and planning efficiency. Gemini models show strong protocol understanding and conclusion synthesis, but their planning-efficiency profile is weakened by the low efficiency and high SER of the pro variant. Qwen models retain moderate execution and feedback-adaptation ability, but show a clear deficit in conclusion synthesis, consistent with the large gap between Pass@1 and Pass@5. DeepSeek models show reasonable feedback adaptation and conclusion synthesis, but weaker planning efficiency and single-step executability, suggesting that deeper reasoning alone does not fully resolve long-horizon experimental control. Overall, the profiling analysis complements aggregate benchmark results by exposing the capability-level reasons behind each family's success and failure patterns.

D Algorithm of Evaluation Framework for LLM-as-Experimenter

To facilitate a clear understanding of the evaluation process described in Section 3.2, we provide a formal operationalization in Algorithm 1.

E Prompts of Data Construction

Prompt for Structured protocol Extraction

Extract scientific experimental protocols from the given text and organize them into a JSON format according to the specified structure.

Input Field

- **text**: $text

Task Requirements

- Identify the scientific experimental protocols described in the text.
- Each protocol must consist of four complete components: **Specific Research Question**, **Experimental Design & Principle**, **Expected Results**, and **Experimental Conclusion**.
- Ensure that the extracted information accurately reflects the original content.

Extraction Specifications

- **Specific Research Question**: Clearly state the specific scientific problem the experiment aims to solve. This question should correspond directly to the expected results.
- **Experimental Design & Principle**: Detail the experimental steps and methodology in a list format. Each step should include specific operations.
- **Expected Results**: Describe the anticipated experimental outcomes and observable phenomena.

Algorithm 1 Evaluation of LLM-as-Experimenter in LabEval

Require: Structured metadata $EM_j = \langle \text{Task}_j, \text{Plan}_j, \text{Conclusion}_j \rangle$, Laboratory L_i with I/O Database.
Ensure: Final Evaluation Metrics (Efficiency, Conclusion Accuracy).

Initialization:
$\mathcal{H}_0 \leftarrow \emptyset$ {Initialize interaction history}
$t \leftarrow 1$
$is_terminated \leftarrow$ False
while not $is_terminated$ **and** $t \leq T_{max}$ **do**
  Strategic Planning:
  Query RAG for tool profiles: $\mathcal{P}_t \leftarrow \text{RAG}(\{\text{Profile}(e_j)\}_{e_j \in L_i})$
  Agent generates single-step action: $a_t \leftarrow \mathcal{A}(\text{Task}_j, \mathcal{P}_t, \mathcal{H}_{t-1})$
  Execution & Feedback:
  Lab Agent executes a_t and returns result: $r_t \leftarrow \text{LabAgent}(a_t, \text{State}_{t-1})$
  Update history: $\mathcal{H}_t \leftarrow \mathcal{H}_{t-1} \cup \{(a_t, r_t)\}$
  if $\mathcal{A}$ signals termination **or** a_t is a concluding action **then**
    $is_terminated \leftarrow$ True
  end if
  $t \leftarrow t + 1$
end while
Conclusion Synthesis:
$\text{Conclusion}_{\text{final}} \leftarrow \mathcal{A}(\text{Task}_j, \mathcal{H}_t)$ {Distill final finding from history}
Multi-dimensional Evaluation:
Efficiency $\leftarrow$ CalculateEfficiency$(\mathcal{H}_t)$ {Based on turns and resources}
Pass@1, Pass@5 $\leftarrow$ VerifyAccuracy$(\text{Conclusion}_{\text{final}}, \text{Conclusion}_j)$
return Pass@1, Pass@5, Efficiency

- **Experimental Conclusion**: State the scientific conclusions and the value derived from the expected results.

Output Format

```
{
"problem": "<Specific Research Question>",
"steps": [
"<Description of Step 1>",
"<Description of Step 2>"
],
"answer": "<Expected Results>",
"conclusion": "<Experimental Conclusion>"
}
```

Notes

- Only extract experimental protocols explicitly described in the text.
- Maintain the scientific accuracy and integrity of the original text.
- If the text contains multiple protocols, extract them in the order they appear.
- If the text lacks a complete protocol structure, based on available information, reasonably fill in the corresponding fields or leave them blank.
- Ensure the **Specific Research Question** is sufficiently concrete to establish a clear, direct correspondence with the **Expected Results**.
- The **Experimental Design & Principle** section must use a list format, where each item contains a "step" field.

Prompt for Laboratory Extraction

Input Fields

- `text`: $text

Task Objective

Identify and abstract distinct laboratory environments from multiple scientific questions contained within the `text`, and establish a correspondence between these environments and

the scientific questions. **Note: Multiple scientific questions may share the same laboratory environment.**

Task Instructions

- **1. Environment Abstraction:** Analyze all scientific questions, identify the characteristics of the laboratory environments described, and categorize similar or identical experimental settings into the same laboratory.
- **2. Information Extraction:** For each identified laboratory environment, extract complete information for the following three fields: *category*, *laboratory_name*, and *description*.
- **3. Problem Mapping:** Establish a mapping between each laboratory environment and its corresponding scientific questions.

Extraction Steps

Step 1: Environment Identification and Categorization

1. Read through all scientific questions to identify the following core elements:
   - Laboratory type (Field of research)
   - Experimental environment characteristics
   - Experimental objectives and methods
2. Judgment criteria for categorization (Questions belong to the same lab if core features match):
   - Same or similar scientific fields.
   - Similar experimental purposes and methods.
   - Matching core experimental environment features.
3. Assign a unique identifier to each distinct laboratory environment.

Step 2: Laboratory Information Extraction

Extract the following information for each identified laboratory environment:

- **1. category:** The scientific field to which it belongs. Must be chosen exclusively from: **Chemistry, Biology, or Physics**.
- **2. laboratory_name:**

  Name the laboratory reasonably based on the nature of the experiments and research direction. The name should reflect the core function of the laboratory. Examples: "Molecular Biology Lab", "Optical Characterization Lab", "Computational Chemistry Lab".
- 3. description

  A comprehensive description of 50–100 words. Should include: Functional positioning, primary research directions, environmental characteristics, and core capabilities.

Step 3: Establish Problem Mapping

List all scientific questions that utilize each laboratory environment:

- Identify questions using their unique `question_id`.
- Ensure each question corresponds to at least one laboratory environment.
- Each question can only correspond to **one** laboratory.
- One laboratory can contain **multiple** questions.

Output Specification(JSON Format)

[{ "category": "Laboratory Category", "laboratory_name": "Laboratory Name", "description": "Comprehensive description (50-100 words), including functions, research directions, environment features, etc.", "related_questions": [{ "question_id": "Question ID or Unique Identifier" }, { "question_id": "Question ID or Unique Identifier" }] }]

Prompt for Laboratory Resource Inventory Construction

Input Fields

- `question`: $question (Experimental problem/description)

Task Objective

Systematically extract all relevant **equipment**, **software**, and **reagents_and_consumables** from the given experimental question. Output a complete, categorized result without omissions.

Extraction Principles

Core Requirements

1. **Zero Omission Principle**: All resources explicitly mentioned or implicitly required in the question must be fully extracted.
2. **Explicit + Implicit Identification**:
   - **Explicit**: Resources whose names are directly stated in the text.
   - **Implicit**: Resources that are logically necessary for experimental methods or operational steps but are not explicitly named.
3. **End-to-End Coverage**: Ensure extraction covers all resources required from sample preparation to data analysis.

Extraction Steps

Step 1: In-depth Question Analysis

Analyze each question meticulously to extract the following:

- **1. Explicit Resource Extraction:**

  Directly mentioned equipment, software, and reagents/consumables. Specific brands, models, or specifications (retain detailed information).

- 2. Implicit Resource Inference Infer essential implicit resources based on the experimental operations:
  - **Equipment**: Pre-treatment devices, environmental control (temp/humidity/pressure), basic lab tools (balances, pipettes, stirrers), safety equipment, and data acquisition hardware.
  - **Software**: Instrument control software, format converters, statistical tools, literature management, and operating systems.
  - **Reagents and Consumables**: Solvents for dilution/dissolution, calibration standards, cleaning/maintenance reagents, basic consumables (gloves, weighing paper, containers), and storage materials.
- **3. Workflow Integrity Check** Verify that every step of the experimental workflow has its required resources accounted for.

Step 2: Categorization and De-duplication

- Merge synonyms (e.g., "pipetting gun" and "pipettor" → "Pipettor").
- Sort resources by function/type to ensure clarity.

Step 3: Defining Input and Output for Equipment and Software

When extracting `equipment` and `software`, define their specific "Input Scenarios" and "Output Results" based on the context:

- **Equipment Mapping**:
  - **Input**: Specific experimental operations or methods mentioned in the question (e.g., "Detecting glucose levels in serum").
  - **Output**: Concrete numerical values, observable phenomena, or functional outcomes (e.g., "Glucose concentration 5.2 mmol/L").
- **Software Mapping**:
  - **Input**: Specific software usage scenarios (e.g., "Record HPLC data using Excel").
  - **Output**: Specific data tables, charts, or quantitative analysis results generated by the software.

Output Specification (JSON Format)

{ "equipment": { "Equipment_Name_1": [{ "input": "Clear experimental operation/method related to the equipment", "output": "Specific numerical values, observable phenomena, or functional results" }] }, "software": { "Software_Name_1": [{ "input": "Specific software operation scenario based on the question", "output": "Specific data, charts, or analysis results

generated" }] }, "reagents_and_consumables": ["Reagent/Consumable 1", "Reagent/Consumable 2"] }

Prompt for Laboratory Hardware and Software I/O Information Extraction

Task Overview

Based on the provided scientific literature collection, your goal is to construct a **Hardware or Software Input-Output (I/O) Inventory**. This involves transforming unstructured text into structured specifications that define the functional capabilities of laboratory tools.

Workflow

1. Local Extraction (Per Document)

Extract tool-specific I/O relationships from each individual document.

- **Identification**: Identify tool entities and classify them as either **Hardware** (physical instruments/equipment) or **Software** (computational systems/algorithms).
- **Mapping**: Identify the specific input scenarios (actions, conditions, or samples) and their corresponding output results (data, phenomena, or outcomes).

2. Global Aggregation and Deduplication

Perform cross-document synthesis to build a holistic profile for each tool entity:

- **Entity Resolution**: Aggregate all records associated with an identical tool entity across the entire collection.
- **I/O Synthesis**: Consolidate diverse I/O records into a comprehensive set of feasible capabilities for the tool.
- **Functional Integrity**: Ensure the strict mapping between specific inputs and their specific outputs is preserved.

Extraction Requirements

- **Input Field**: Must describe a concrete experimental operation, method, or usage context (e.g., "Connecting a tube to a test tube to check for air-tightness").
- **Output Field**: Must describe a specific numerical value, an observable experimental phenomenon, or a clear functional result (e.g., "A stable water column forms inside the tube").
- **Classification**: Categorize each item strictly as `Hardware` or `Software`.

Output Format (JSON)

The output must be a valid JSON object following this structure:
{ "item": "Tool Entity Name", "category": "Hardware / Software", "examples": [{ "input": "Description of the specific input scenario or operation", "output": "Description of the resulting data or experimental phenomenon" }] }

F Prompts of Evaluation Framework for LLM-as-Experimenter

Prompt for LLM-as-Experimenter

Role

You are a laboratory researcher executing an experimental workflow. Your task is to determine the next logical action based on the experimental observations returned by the `lab_agent`.

Input Parameters

- **Experimental Observation**: $lab_agent_observation
- **Research Question**: $self.researcher_question
- **Available Resources**: $self.available_resources

Task Objective

Answer the research question based on the observed phenomena. All proposed actions must strictly utilize the equipment, consumables, and software listed in the **Available Resources**.

Operational Steps

1. **Analyze Observation**: Meticulously interpret the experimental phenomenon described by the `lab_agent`.
2. **Assess Inferential Capability**: Determine if the research question can be an-

swered directly from the current observation.

- If direct inference is possible: Output the final conclusion marked with `[!final_answer]`.
- If direct inference is not possible: Proceed to the next steps and provide a procedural response marked with `[!process_answer]`.

3. **Evaluate Current State**: Determine if the experiment is proceeding as expected or if anomalies are present.

4. **Determine Next Action**: Based on the analysis, decide on the most appropriate **single** next step. This step must be actionable and utilize the provided resources.

Output Specifications

Content Requirements

- Start with an explicit judgment on whether the answer can be inferred.
- For `[!final_answer]`: Provide a clear, definitive conclusion.
- For `[!process_answer]`: Include an analysis of the phenomenon and a specific next step.
- Briefly explain the rationale for the chosen action and specify the resources used.
- In case of anomalies, describe the intended handling procedure.

Format and Style

- Output must be in **JSON format**.
- Limit each response to **3–5 sentences**.
- Use professional, objective, and precise scientific terminology.
- Avoid redundant descriptions or over-interpretation.

Output Examples

Example 1 (Procedural Action):
{ "analysis": "[!process_answer] The observation shows the reaction solution turning pale yellow, suggesting the reaction is ongoing but incomplete. Further detection of the product is required.", "next_step": "Measure the absorbance of the solution using the spectrophotometer (from available resources) to determine reaction progress.", "resources_used": "spectrophotometer" }
Example 2 (Final Conclusion):
{ "analysis": "[!final_answer] The observation clearly shows the formation of a blue precipitate, confirming the successful synthesis of the target compound.", "next_step": null, "resources_used": null }

Prompt for Lab Agent

Task Overview

Your task is to simulate and generate realistic experimental phenomena based on the provided experimental operations. To ensure accuracy, you must synthesize information from:

- Relevant reference inputs and outputs.
- Previously executed experimental steps and their observed phenomena.
- The original research question.

Input Information

- **Reference I/O**: $reference_experiment_steps_and_phenomena
- **Historical Context**: $previous_experiment_info
- **Current Operation**: $current_experiment_step
- **Original Question**: $self.question

Supplementary Guidelines

- Generate phenomena based strictly on the current experimental description while maintaining consistency with historical context.
- **Objective Observation**: Describe only what is observed; do not include explanations, conclusions, or recommendations.
- **Qualitative Results**: Do not provide numerical data. Use qualitative terms such as "normal," "increased," or "decreased."

Processing Steps

1. **Information Analysis**: Comprehend the current operation, reference data, and the relationship between the original question and previous results.
2. **Relevance Assessment**: Verify if the current description is experimentally relevant.
3. **Phenomenon Generation**: Logically infer and describe phenomena based on scientific principles and the provided ground truth.
4. **Format Validation**: Ensure the output strictly adheres to the JSON structure.

Output Specifications

Format Requirements

- The output must be a JSON object containing a list named `"phenomenons"`.
- Each item in the list corresponds to a single observation description.
- If the current description is ambiguous or lacks sufficient information, output: *"Current experimental steps are unclear."* Do not generate generic/common-sense phenomena.

Content and Style

- **Professionalism**: Use concise, accurate, and standard scientific terminology.
- **Neutrality**: Maintain an objective stance; avoid subjective judgments.
- **Specificity**: If the experiment is irrelevant, return a single JSON object containing: *"No relevant phenomena."*

Examples

Standard Observation: { "phenomenons": ["Gas bubbles are generated.", "No significant change in color observed."] }
Insufficient Information:
{ "phenomenons": ["Current experimental steps are unclear."] }

Important Notes

- Strictly follow the given description; do not add extraneous information.
- Ensure each description is an independent observation.
- The `"phenomenons"` field **must not** be empty, null, or `[]`. It must contain clear semantic content.
- Phenomena must be logically linked to the original question; do not simply copy example text.

Prompt for Evaluation LLM

Task Description

Determine whether the conclusion of the given **Answer** is consistent with the conclusion of the **Standard Answer** regarding the experimental question.

Input Information

- **Experimental Question**: `$self.question`
- **Answer**: `$answer`
- **Standard Answer**: `$self.standard_answer`

Supplementary Guidelines

- Experimental questions typically involve scientific experiments, data analysis, or explanation of phenomena.
- Consistency judgment should be based on core arguments and key findings rather than exact literal matching.
- Consider reasonable variations in expression (e.g., synonymous phrasing or descriptions from different perspectives).

Operational Steps

1. **Extract Conclusion Elements**: Identify key components of the core conclusion (e.g., primary findings, causal relationships, trend judgments) from both the Answer and the Standard Answer.
2. **Semantic Comparative Analysis**: Compare whether the core viewpoints align,

ignoring differences in linguistic style and minor details.

3. **Consistency Judgment**: Provide a definitive judgment based on the results of the semantic analysis.

Output Specifications

Content Requirements

The output must contain exactly two parts:

1. **comparison**: A detailed explanation of the semantic similarities or differences between the two conclusions.
2. **judge_result**: A definitive conclusion of `"yes"` (consistent) or `"no"` (inconsistent).

Format and Style

- Output must be in **JSON format**.
- **Objective and Neutral**: Analyze based strictly on facts.
- **Professional and Rigorous**: Avoid subjective assumptions.
- **Concise and Clear**: Use precise language and avoid ambiguous descriptions.

Example

Input:

- **Question**: Investigating the effect of light intensity on the rate of photosynthesis.
- **Standard Answer**: Within a certain range, increasing light intensity enhances the photosynthetic rate; however, the rate plateaus after reaching the light saturation point.
- **Answer**: As light gets stronger, plant photosynthesis speeds up, but excessively strong light causes the rate to stabilize.

Output:

{ "comparison": "Both conclusions convey the same key scientific fact: the influence of light intensity on the photosynthetic rate follows a trend of initial increase followed by stabilization. The Answer's 'speeds up' corresponds to the 'positive correlation' in the standard answer, and 'stabilize' accurately reflects the 'light saturation' phenomenon. Despite the stylistic differences, the core viewpoints are entirely consistent.", "judge_result": "yes" }

Important Notes

- If the Answer contains information not mentioned in the Standard Answer, distinguish whether it is supplementary or contradictory.
- In cases of partial consistency, the judgment should prioritize the alignment of the core conclusion.
- Avoid misjudging conclusions as inconsistent due to variations in the use of technical terminology.

G Data Quality Verification

To further ensure the technical validity and practical feasibility of our generated scientific dataset, we conducted a structured human evaluation involving five domain experts with diverse backgrounds in laboratory automation and natural sciences: two senior researchers in chemical engineering, one laboratory automation engineer, and two doctoral candidates with extensive experience in bioinformatics and experimental design. All evaluators had prior experience in interpreting structured scientific literature.

Evaluator Training and Calibration. Prior to the formal evaluation, all experts underwent a systematic calibration process. Each evaluator received detailed annotation guidelines covering our extraction protocols and laboratory resource mapping criteria. A pilot round was conducted on 10 shared cases, during which evaluators discussed and resolved disagreements to establish a unified scoring standard. This calibration phase was designed to minimize subjective interpretation and ensure consistent application of the evaluation criteria across all annotators.

Evaluation Protocol. Each expert independently evaluated a randomly sampled set of 100 generated experimental modules (EM_j) and their corresponding laboratory inventory profiles, with an overlap of 20 shared cases across all five evaluators to enable inter-annotator agreement measurement. For each case, they assessed two key criteria using a three-point scale (*High*, *Partial*, *Low*):

- **Extraction Fidelity:** Whether the structured tuple $\langle \text{Task}_j, \text{Plan}_j, \text{Conclusion}_j \rangle$ accurately captured the essential scientific information

from the source literature without distorting the original experimental intent. A rating of *High* required that all three components were complete, accurate, and free of hallucinated content.

- **Operational Consistency:** Whether the identified laboratory resources (hardware, software, reagents) and their associated I/O profiles (Profile(e_j)) aligned with established physical laboratory workflows and functional instrument specifications. A rating of *High* required that all resource entries were physically plausible and correctly mapped to their experimental context.

Each expert received $20 per 10 cases reviewed (i.e., $200 per expert), yielding a total expenditure of $1,000. The review process was completed over a seven-day period, with evaluators spending an average of approximately 15 minutes per case to ensure thorough assessment.

Inter-Annotator Agreement. On the 20 shared cases evaluated by all five experts, we computed Fleiss' Kappa to quantify inter-annotator agreement. The resulting Kappa scores were $\kappa = 0.76$ for Extraction Fidelity and $\kappa = 0.71$ for Operational Consistency, both indicating *substantial agreement* according to the standard interpretation scale. These scores confirm that our evaluation criteria were sufficiently well-defined to yield consistent judgments across independent evaluators.

Results and Analysis. To synthesize the five experts' assessments, we applied a majority-voting scheme on the shared cases and averaged individual scores on the non-overlapping cases to obtain consensus annotations. Based on the aggregated labels, 96.5% of cases were rated as exhibiting high Extraction Fidelity, and 91.0% were judged to be operationally consistent and realistic.

We acknowledge that these scores are relatively high. We attribute this primarily to the structured nature of the extraction pipeline and the domain-specificity of the source literature (Bio-protocol), which provides well-formatted experimental descriptions that are amenable to accurate parsing. Nonetheless, the 9.0% of cases flagged for Operational Consistency issues reveal systematic limitations: the most common failure mode involved implicit resource inference, where the pipeline occasionally omitted non-standard consumables or context-dependent instrument configurations that required deep domain expertise to identify. These cases were excluded from the final benchmark to maintain dataset integrity.

H Case Study

H.1 Gemini-3.1-Pro-Preview: Operational Redundancy (Fail)

Using this case as an example, Gemini-3.1-Pro-Preview clearly illustrates the "operational redundancy" bottleneck. The LLM's reasoning trace reveals an obsession with procedural minutiae at the expense of experimental progress. The details of the experimental process are as follows:

```
{
  "step_1": {"operate": "Mix 10ml
      FBS with 8ml diisopropyl
     ether and 2ml n-butanol.
     Shake for 30 mins.", "
     analysis": "Initial
     extraction phase to remove
     lipids."},
  "step_2": {"operate": "
     Centrifuge at 1000g for 10
     mins to separate phases.
     Collect the lower serum
     phase.", "analysis": "Phase
     separation complete.
     Proceeding to secondary
     extraction."},
  "step_3": {"operate": "Repeat
     the extraction: add 8ml
     diisopropyl ether and 2ml n-
     butanol to the recovered
     serum.", "analysis": "
     Ensuring maximum lipid
     removal through redundancy
     ."},
  "step_4": {"operate": "
     Centrifuge again at 1000g.
     Carefully pipette the serum
     phase into a new sterile
     tube.", "analysis": "Serum
     recovered. However, I must
     ensure no residual organic
     solvent remains before GC/MS
     ."},
  "step_5": {"operate": "Apply
     nitrogen stream to evaporate
      trace organic solvents from
      the serum sample.", "
```

```
    analysis": "Continuing
    sample purification.
    Scientific conclusion cannot
     be reached until metabolic
    assays are performed."},
  ...
  "step_20": {"operate": "Prepare
     24-well plates for cell
    seeding using the purified
    serum.", "analysis": "[!
    process_answer] The agent
    has reached the maximum step
     limit. Current status:
    Sample preparation complete.
     Conclusion: Data
    insufficient to determine
    metabolic shift."}
}
```

H.2 GPT-5.4: Professional Analytical Synthesis (Pass)

In contrast, GPT-5.4 serves as a benchmark for successful execution within this case. It demonstrates high-level experimental planning and conclusion analysis by correctly interpreting the compensatory metabolic flux. The details of the experimental process are as follows:

```
{
  "step_1": {"operate": "Execute
    dual-phase extraction using
    diisopropyl ether/n-butanol
    (4:1 ratio) on FBS.", "
    analysis": "Delipidation
    protocol initiated to
    isolate the variable of
    exogenous lipid supply."},
  "step_2": {"operate": "Perform
    GC/MS analysis on the
    extracted serum to verify
    total lipid concentration.",
     "analysis": "GC/MS confirms
     >95% reduction in fatty
    acid peaks. The delipidation
     is successful. Moving to
    cellular assay."},
  "step_3": {"operate": "Incubate
     cells in 10% delipidated
    serum media and introduce
    [14C]-acetate for 4 hours.",
     "analysis": "Measuring the
    flux of acetate into the
    lipid synthesis pathway.
    This is the critical
    diagnostic step."},
  "step_4": {"operate": "Measure
    radioactivity using a liquid
     scintillation counter
    across triplicates.", "
    analysis": "[!final_answer]
    The delipidated serum group
    shows a 2.4x increase in 14C
    -acetate incorporation
    compared to the control
    group. This phenomenon
    indicates that exogenous
    lipid deprivation triggers a
     compensatory up-regulation
    of endogenous de novo
    lipogenesis. The agent
    successfully characterizes
    the cellular metabolic
    feedback loop."}
}
```

H.3 Qwen3.5-27b: Logical Chain Rupture (Fail)

The Qwen3.5-27b trace in this case highlights a failure in reasoning consistency. Despite observing the necessary Carbon-14 data, the LLM fails to bridge the final gap to a definitive conclusion.

```
{
  "step_1": {"operate": "Extract
    lipids from FBS using
    solvent mixture and
    centrifuge.", "analysis": "
    Preparing delipidated serum
    ."},
  "step_2": {"operate": "Run GC/
    MS to check for residual
    lipids.", "analysis": "Lipid
     levels are low. Proceeding
    ."},
  "step_3": {"operate": "Perform
    14C-acetate incorporation
    assay on cultured cells.", "
    analysis": "[!process_answer
    ] The radioactivity signals
    are detected and recorded in
     the phenomenon log. However
    , since the GC/MS result for
     the specific control group
    was not explicitly repeated
```

```
      in this step, I cannot yet
      conclude the overall trend
      ."},
  "step_4": {"conclusion": "[!
      final_answer] The experiment
       shows increased
      radioactivity in delipidated
       samples, which might
      suggest higher synthesis.
      But due to missing
      confirmation in the analysis
       phase, the results remain
      inclusive regarding the
      research objective."}
}
```

H.4 Discussion of Performance Gaps

The patterns observed in this specific case further elucidate the cognitive bottlenecks and divergent capabilities identified in Figure 2. As seen in the Gemini-3.1 trace, the primary deficiency lies in planning efficiency, where redundant operations trap the LLM in preparatory loops, preventing it from reaching core experimental milestones. In stark contrast, the performance of GPT-5.4 highlights superior analytical capability, demonstrating that top-tier LLMs can synthesize complex biological mechanisms and correctly interpret compensatory endogenous synthesis. Meanwhile, the failure of Qwen-3.5 exposes a critical gap in reasoning consistency, proving that even with access to complete evidence, certain LLMs suffer from logical chain ruptures during the final synthesis of experimental results.