Proof of Time: A Benchmark for Future-Verifiable Scientific Influence

Anonymous ACL submission

Abstract

Frontier language models are increasingly used to evaluate scientific ideas, prioritize research directions, and support expert decision-making, yet it remains unclear whether they can anticipate which work will later prove influential. We introduce **Proof of Time (PoT)**, a benchmark for evaluating whether models can forecast the future scientific influence using only information available at an earlier timepoint. PoT operationalizes influence through multiple downstream signals that become observable later, including community uptake, peer recognition, research trajectory, and SOTA progress, covering diverse dimensions of scientific influence. We evaluate frontier models in both zeroshot and agentic settings, including restricted tool-use environments designed to support evidence gathering without external retrieval leakage. Across 30K+ instances, we find that current models achieve non-trivial performance but remain far from ceiling, with substantial variation across task families and model classes. Tool-using agents can improve accuracy on some tasks, but these gains are often inconsistent and come with large compute overhead, revealing important cost-performance trade-offs. Together, these findings position PoT as a controlled, future-verifiable benchmark for scalable evaluation of agents in future scientific influence forecasting. [1]

1 Introduction

"AI for Science" has accelerated rapidly: machine learning systems are increasingly embedded directly in discovery pipelines, including hypothesis generation, candidate design, literature synthesis, and experiment steering (Wang et al., 2023; Karpatne et al., 2025; Liu et al., 2025). High-accuracy protein structure prediction is a canonical example (Jumper et al., 2021). This progress raises a broader metascientific question: can AI systems anticipate which ideas, papers, and researchers will later prove influential?

This question becomes more important as frontier models become increasingly agentic. They can plan, use tools, search over evidence, write code, and execute parts of research workflows (Yao et al., 2023; Schick et al., 2023; Wang et al., 2024). If such systems are used to generate, prioritize, and pursue research directions at scale, we need principled ways to evaluate whether they can identify work that will later matter.

This is challenging because scientific influence is sparse and uneven: a small fraction of projects accounts for a disproportionate share of downstream impact (Munafò et al., 2017; Ioannidis et al., 2015). If models eventually become strong at recognizing promising directions, they could improve how attention, effort, and compute are allocated across the space of possible ideas. That makes future scientific influence forecasting a useful and increasingly important capability to measure.

We introduce **Proof of Time (PoT)**, a benchmark for evaluating whether models can forecast future scientific influence from information available at an earlier timepoint. PoT follows a simple principle: freeze the evidence available at time t_0, constrain the solver to that evidence in an offline sandbox, forecasts status at target time t_1, and score the prediction when t_1 comes ($t_1 > t_0$). This design makes evaluation future-verifiable, scalable, and contamination-controlled: correctness is determined by observed later outcomes rather than subjective judges, while all systems operate under the same fixed information boundary.

A key challenge is that scientific influence is multidimensional and cannot be reduced to a single signal. PoT therefore evaluates multiple downstream dimensions of future influence, including community uptake (*Citations*), peer recognition (*Awards*), research trajectory (*Faculty*), and techno-

[1] Code, data and evaluation scripts will be made publicly available upon publication.

logical frontier progress (*SOTA*). These task families are intended as controlled proxy signals of scientific influence rather than direct measures of intrinsic scientific value.

In this paper, we instantiate PoT on AI/NLP domain, where large-scale structured metadata and downstream signals are readily available. This domain provides a practical first testbed for studying future scientific influence forecasting under controlled evidence access, while supporting automatically scorable tasks that would otherwise require expensive and subjective expert evaluation.

PoT is also designed to study agentic model behavior under controlled information access. Because solvers operate in a network-isolated sandbox with only a restricted local tool set, any gains from tool use, multi-step reasoning, or larger interaction budgets must come from better use of the same frozen evidence rather than from retrieval of newer information. This makes PoT useful not only for evaluating scientific influence forecasting, but also for measuring when agentic systems can make better use of available evidence, and when additional interaction and compute fail to deliver worthwhile gains.

Our experiments show that performance varies sharply across task families, suggesting that forecasting future scientific influence is not a single capability. Some families benefit substantially from stronger evidence-use scaffolds and larger interaction budgets, while others remain difficult even for frontier models in more capable agentic settings. Post-cutoff evaluation further serves as a robustness check, helping distinguish performance that transfers to genuinely newer instances from performance that may depend more heavily on training exposure or memorized historical regularities.

Our contributions are as follows: (i) we introduce **Proof of Time (PoT)**, a future-verifiable and contamination-controlled benchmark for evaluating whether models can forecast future scientific influence from a frozen pre-cutoff evidence snapshot; (ii) we formalize four complementary task families—community uptake, peer recognition, research trajectory, and technological frontier progress—without collapsing scientific influence into any single proxy; (iii) we provide an offline sandbox benchmark primitive for comparing direct generation and tool-using agents under controlled information access, including ablations over prompt design and interaction budget; and (iv) we show that PoT reveals a highly non-uniform capability profile across future scientific influence forecasting tasks, identifying both where current models can effectively leverage available evidence and where they continue to fall short.

2 Related Work

2.1 Scientific forecasting, bibliometrics, and downstream outcomes

Scientometrics has long studied how papers accrue citations, attention, and influence over time, typically using bibliometric features, topic signals, author history, or network structure to model downstream outcomes (Price, 1965; Barabási and Albert, 1999; Griffiths and Steyvers, 2004; Wang et al., 2013; Acuna et al., 2012). More recent work has explored whether LLMs themselves can predict citation-related signals from text (de Winter, 2024; Singla et al., 2023). At the same time, scholarship on research assessment emphasizes that signals such as citations and awards should not be equated with a single notion of intrinsic quality (Thelwall, 2025; Hicks et al., 2015). PoT is aligned with this perspective: rather than treating citations or awards as ground-truth merit, we use them as *future-observable dimensions* for evaluating scientific influence forecasting.

2.2 Temporal generalization, live evaluation, and contamination

Web-scale pretraining makes static benchmarks increasingly vulnerable to benchmark data contamination, motivating evaluation setups that are refreshed over time or explicitly structured to reduce memorization effects (Xu et al., 2024; White et al., 2025; Jain et al., 2024; Zhao et al., 2025; Yu et al., 2024). PoT adopts the same core principle of temporal partitioning, but applies it to future-facing tasks whose answers become observable only later. This makes the benchmark especially useful for distinguishing parametric recall from genuine extrapolation based on frozen historical evidence.

2.3 Agents, tool use, and evaluation in realistic workflows

Recent work evaluates LLM agents in increasingly realistic environments where success depends on multi-step reasoning, tool use, and constraint-following (Yoran et al., 2024; Yao et al., 2024; Lu et al., 2025; Yehudai et al., 2025). In scientific settings, tool-using systems are often proposed for literature synthesis, paper analysis, and knowledge-

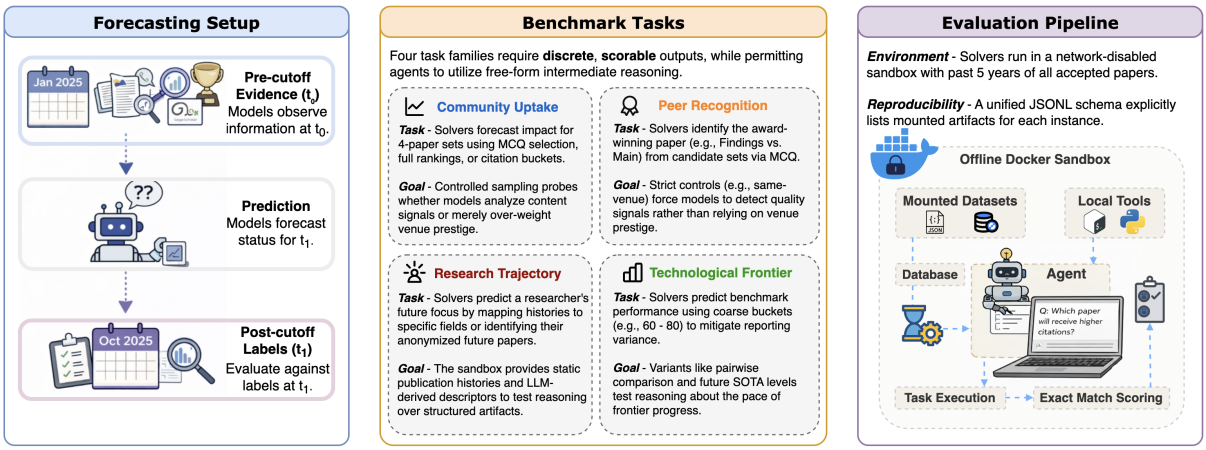

Figure 1: Overall Proof-of-Time benchmark workflow. The left panel describes the forecasting setup; the middle panel emphasizes the four task families and their verifiable post-cutoff signals; the right panel illustrates offline sandbox execution with optional tool-using agents and automatic scoring.

intensive support tasks (Boiko et al., 2023; Lála et al., 2023; Mitchener et al., 2025). PoT contributes to this line of work by providing a controlled reasoning environment with fixed evidence, explicit budget limits, and externally verifiable labels. It therefore supports a cleaner study of when tool use actually helps and when added agentic complexity mainly increases cost or failure modes.

3 Benchmark Design

PoT evaluates whether a model can forecast scientific influence from the information that would have been available at the time of decision. Each benchmark instance has three parts:

1. a **pre-cutoff evidence snapshot** containing only artifacts available up to time t_0,
2. a **prediction query** defined over that evidence, and
3. a **later-observed outcome** used only for scoring.

The key idea is simple: the model must answer using only the frozen evidence snapshot, while correctness is determined by an outcome that becomes observable only after the cutoff. This allows us to turn otherwise difficult-to-evaluate research judgments into automatically scorable tasks.

3.1 A PoT Instance by Example

We begin with a concrete example. Consider an award-tier prediction instance. Suppose the cutoff date t_0 is the submission or review period for a conference. The model is given only evidence available before the award decision, such as the paper title, abstract, authors, and historical examples from the same venue. It is then asked to predict the eventual outcome, e.g., whether the paper will receive no award, a findings distinction, an outstanding paper designation, or a best paper award. The gold label is the actual later award decision observed after t_0. Other task families follow the same template. In a citation instance, the model sees only pre-cutoff paper evidence and predicts a later citation ranking or citation bucket. Across all families, the model is always restricted to pre-cutoff evidence, while the label is always defined by a later observable outcome.

3.2 Formal Setup

Formally, each PoT instance is a tuple

$$(E_{\leq t_0}, q, y_{t_1}),$$

where $E_{\leq t_0}$ is the evidence available up to cutoff time t_0, q is a prediction query, and y_{t_1} is a label derived from an outcome observed at time $t_1 > t_0$. The solver receives $(E_{\leq t_0}, q)$ and outputs a prediction $\hat{y}$. Evaluation compares $\hat{y}$ against y_{t_1}.

In PoT, we choose t_0 to fall after the model knowledge cutoff date whenever possible, so that test instances are genuinely unseen in pretraining data. This makes the benchmark more contamination-controlled: models may use only evidence available up to t_0, while the gold outcomes are observed only afterward.

This formulation separates what the model is allowed to use from what determines correctness. The former is fixed by the evidence snapshot; the

Family	Prediction target	Evidence at t_0	Output format	Scoring signal
Community uptake *(Citations)*	Future citation outcome of candidate papers	Titles, abstracts, authors, and citation statistics of historical papers (2021–2024)	Most-cited MCQ Ranking Citation bucket	Post-cutoff citation counts (Google Scholar)
Peer recognition *(Awards)*	Future award-tier outcome of candidate papers	Titles, abstracts, authors, and historical venue award examples (2021–2024)	Tier MCQ (Findings / Main / Outstanding / Best)	Official conference award lists
Research trajectory *(Faculty)*	Professor field, likely paper authorship, or near-future research focus	Professors' pre-cutoff publication histories; anonymized post-cutoff candidate papers	Field MCQ Paper→author Field-focus prediction	Post-cutoff publications and curated author–field labels
Technological frontier *(SOTA)*	Later benchmark-performance bucket	Benchmark descriptions, metrics, and historical leaderboard/model performance	Bucket MCQ (0–20, 20–40, 40–60, 60–80, 80–100)	Post-cutoff leaderboard / benchmark updates

Table 1: PoT task families. All tasks are solved from a frozen pre-cutoff evidence snapshot and scored using later-observed, externally verifiable outcomes.

latter is defined by a later outcome that is hidden from the solver at inference time.

3.3 Offline Sandbox

All PoT instances are executed in a network-isolated offline sandbox. Each instance mounts a read-only bundle of files. The solver may inspect and analyze only these files. No web access or external retrieval is allowed. This design has two purposes. First, it reduces leakage from post-cutoff information. Second, it makes comparisons between direct generation and tool-using agents interpretable. If an agent performs better than a zero-shot baseline, that gain must come from better use of the same evidence snapshot rather than from access to newer information.

In agentic settings, we allow a restricted local tool set such as file reading, basic search over mounted files, Python execution, and lightweight text manipulation. These tools are intended to support evidence extraction and structured analysis, not external retrieval. Each instance includes an explicit inventory of the provided files so that the evidence boundary is transparent and reproducible.

3.4 Task Families

Proof of Time (PoT) instantiates four task families, each operationalizing a different, controlled proxy signal of scientific influence. Table 1 summarizes the setup.

Community uptake. *Citations* operationalizes scientific influence through later community uptake. It captures whether work attracts sustained attention and becomes incorporated into subsequent research. We construct instances by time-partitioning the literature: the solver is given pre-cutoff evidence from papers published in top NLP venues during 2021–2024 (ACL, NAACL, EMNLP) with titles, abstracts and author information provided, together with citation counts as of October 2025, and is asked to forecast the citation counts of a *candidate set* of four post-cutoff papers (titles, abstracts and authors provided) published in ACL or NAACL 2025. PoT supports three output formats. In *MCQ* instances, the solver selects which candidate will have the most citations by the target horizon. In *Ranking* instances, the solver orders all candidates by predicted citations. In *Bucket* instances, the solver maps each paper to a coarse citation range to reduce sensitivity to minor fluctuations; we use buckets such as 0–10, 10–50, 50–200, and 200+. To reduce temporal and venue confounds, each instance in *MCQ* and *Ranking* uses a matched candidate set of four papers drawn from the same venue–year combination, so comparisons are not driven by natural exposure effects such as earlier publication or venue-wide visibility.

Peer recognition. *Awards* operationalizes scientific influence through selective recognition by expert communities. This provides a complementary signal to citation-based uptake, reflecting distinction and perceived significance rather than broad attention alone. We construct instances by time-partitioning the literature within the same three venues (ACL, NAACL, EMNLP). The solver is

given a pre-cutoff evidence pool of accepted papers from 2021–2024 with their titles, abstracts, and authors provided. Post-cutoff evaluation papers are drawn from 2025 in the same venues, and the task is to predict the paper's award tier. **For *Awards* only**, we additionally include a diagnostic *pre-cutoff* evaluation split that asks the same award-tier prediction question for papers within the evidence window, primarily to probe reliance on memorized or training-exposed information rather than evidence-grounded inference.

Research trajectory. Scientific influence is also expressed through the continuation and evolution of researcher agendas. The *Faculty* task family evaluates whether solvers can extrapolate from a researcher's publication history to infer their near-future focus, capturing continuity and drift in a researcher's agenda. Evidence is restricted to each professor's publications from 2021–2024, and targets are defined using 2025 publications. We include (i) professor → field prediction (choose from a fixed taxonomy), (ii) professor → article identification (select which anonymized 2025 candidate paper belongs to the professor, with authors removed and a "None" option), and (iii) papers → field classification for sets of recent papers. To support these tasks under offline constraints, the sandbox provides per-professor publication histories and auxiliary summaries such as LLM-derived keywords and area descriptors computed from pre-cutoff publications only. This allows us to study whether agents can benefit from structured intermediate artifacts, and where such summarization introduces errors or bias.

Technological frontier progress. *SOTA* operationalizes scientific influence through downstream effects on the technical frontier. It measures whether solvers can reason about frontier model performance and the pace of benchmark progress over time. PoT compiles normalized snapshots of popular benchmarks and leaderboards as of October 2025 and asks solvers to predict performance in coarse buckets. We use buckets to improve robustness to prompt sensitivity and evaluation variance; for example, a five-level scale may represent performance ranges from weak to near-ceiling (e.g., 0–20, 20–40, 40–60, 60–80, 80–100). Variants include single model-benchmark bucket prediction and pairwise model comparison. We include next-year SOTA forecasting as a forward-compatible PoT task. It is not scored in this snapshot, but it becomes automatically verifiable as new leaderboard results are published.

4 Experiments

4.1 Task suite and scoring

All tasks are designed to be automatically scorable via discrete outputs, while still allowing free-form intermediate reasoning inside the sandbox. We report exact-match accuracy for all tasks.

4.2 Solvers: direct generation vs. tool-using agents

We compare three solver configurations to quantify when agentic systems help, and how much gain they provide beyond naive assumptions that agents are always better.

1. **Zero-shot (direct generation).** The model answers from the prompt only, without tools or local data access. This tests parametric-only forecasting ability.

2. **Agentic.** A single tool-using agent runs in a Docker sandbox with access to `bash`, `python`, and a text editor, iteratively gathering evidence from the offline snapshot.

3. **Agentic + structured prompt.** Same agent loop as (2), but with an additional structured preamble that emphasizes offline-only operation and prescribes a more explicit tool-use protocol. This isolates prompt-policy effects under a fixed agent loop.

4.3 Test-time compute via message limits

To study inference-time scaling in an agentic setting, we vary a **message limit**: the maximum number of environment interaction turns before the agent must finalize an answer. We evaluate budgets of 15, 30, and 50 messages, corresponding to low, medium, and high test-time compute.

4.4 Models

Table 2 summarizes the model suite evaluated in our experiments, covering the latest frontier models from Anthropic, Google, and OpenAI (Anthropic, 2025; Google, 2026; OpenAI, 2025). All models are run under identical task definitions and scoring. For agentic runs, the environment and tool interfaces are held constant across models.

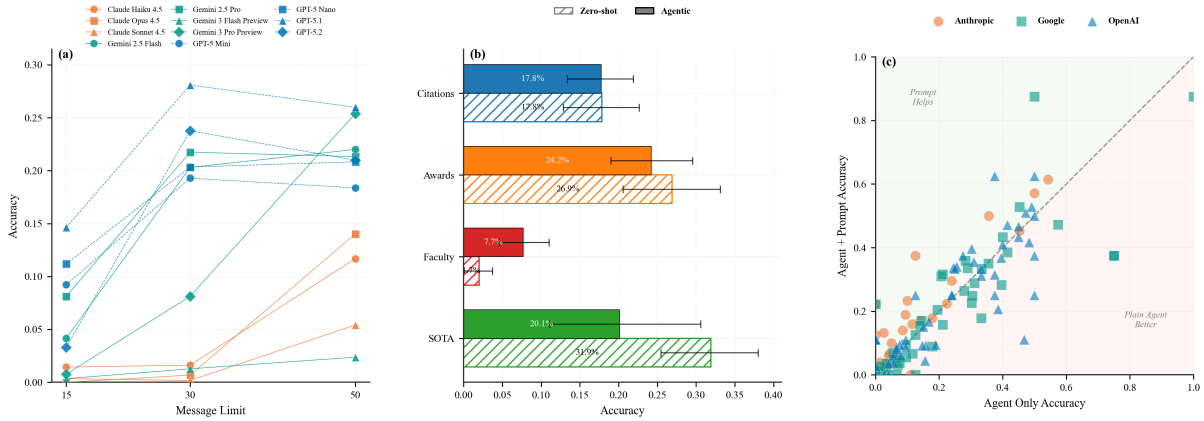

Figure 2: Core results. (A) Test-time compute scaling: accuracy vs. message limit (15/30/50) for each model. (B) Task-family comparison of zero-shot vs. agentic performance at message limit 50, shown as average accuracy across models; error bars indicate variability across task–model combinations. (C) Effect of adding a structured prompt on top of the same agent loop at message limit 50; points above the diagonal indicate the prompt helps.

Provider	Models
Anthropic	Claude Opus 4.5; Claude Sonnet 4.5; Claude Haiku 4.5
Google	Gemini 3 Pro Preview; Gemini 3 Flash Preview; Gemini 2.5 Pro; Gemini 2.5 Flash
OpenAI	GPT-5.2; GPT-5.1; GPT-5 Mini; GPT-5 Nano

Table 2: Model suite used in our experiments. We evaluate representative frontier models across three major providers.

4.5 Implementation details

All agentic runs use a single-agent ReAct loop in a sandboxed environment. We log full interaction traces, tool calls, and termination status, including runs that hit the message limit. This enables downstream analysis of failure modes such as non-convergence and redundant exploration.

5 Results

We organize results around four questions: (i) how performance scales with test-time compute, (ii) whether tool-using agents outperform direct generation, (iii) whether a structured agent prompt helps beyond vanilla ReAct, and (iv) whether conclusions change under post-cutoff evaluation. Unless otherwise specified, we report **sample-weighted exact-match accuracy**.

5.1 RQ1: Test-time compute scaling

Figure 2A shows how agentic performance changes as the message limit increases from 15 to 50. Nearly all models benefit from additional interaction budget, but the magnitude of the gain is highly model-dependent. Gemini models show the strongest scaling behavior overall. GPT-family models also improve consistently, though their gains are generally more moderate and several begin to plateau by message limit 50. Claude models benefit as well, but their absolute performance remains lower overall, with the main improvement occurring only at the largest budget. Full scaling results appear in Appendix Table 7.

5.2 RQ2: Agentic vs. zero-shot performance

RQ2 asks when tool use helps under PoT's evidence constraints. Figure 2B compares zero-shot and agentic performance at message limit 50, aggregated by task family. The effect is strongly task-dependent. *Faculty* shows the clearest relative gain, with agentic performance substantially higher than zero-shot, suggesting that explicit tool use is especially helpful when the task requires assembling and comparing evidence across many historical publications. *Citations* is nearly unchanged, indicating that the current agent setup provides little additional benefit for this family in the aggregate. In contrast, both *Awards* and *SOTA* favor zero-shot performance, with the largest gap appearing on *SOTA*. This suggests that the current agent pipeline does not uniformly improve performance: on some families, controller overhead, tool-use failures, or unnecessary exploration may outweigh the benefits of explicit evidence gathering.

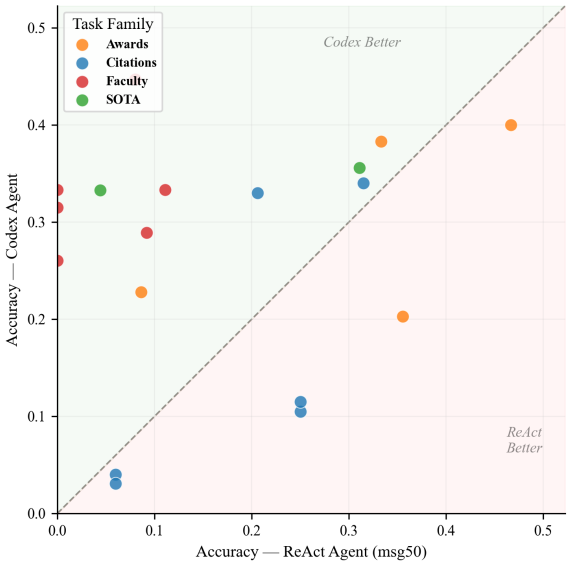

Figure 3: Codex agent vs ReAct agent accuracy at message limit 50 (GPT-5 Mini and Nano). Points above the diagonal indicate Codex outperforms ReAct.

5.3 RQ3: Structured agent prompt effects

Figure 2C compares the standard ReAct agent against the same agent with an additional structured prompt at message limit 50. The effect is again clearly model-dependent. Many points lie close to the diagonal, indicating that the prompt often has only a modest effect, but the overall pattern differs by provider. Claude models tend to fall above the diagonal more consistently, suggesting that the structured prompt often helps them use the offline sandbox more effectively. GPT models are more mixed, with many points clustered near the diagonal and several clear cases on both sides, indicating that the added prompt can help, hurt, or make little difference depending on the task. Gemini models show the widest spread: some tasks improve substantially with the structured prompt, while others are clearly better with the plain agent alone.

Codex Agent Comparison As an additional ablation, we evaluate a Codex CLI agent variant on GPT-5 Mini and GPT-5 Nano. In this setting, the model runs as a Codex CLI subprocess inside the Docker sandbox rather than using a simple ReAct loop. This tests whether more advanced agentic tool use capability affects performance on the same tasks. Figure 3 shows that Codex outperforms the standard ReAct agent in 12 of 18 task-model pairs, suggesting that better tool use in agent settings can provide meaningful gains on certain task families.

Model	Pre-cutoff	Post-cutoff	Δ
Claude Haiku 4.5	14.7%	13.8%	-1.0pp
Claude Opus 4.5	32.3%	19.0%	-13.2pp
Claude Sonnet 4.5	4.6%	4.2%	-0.4pp
Gemini 2.5 Flash	18.0%	50.9%	+33.0pp
Gemini 2.5 Pro	35.2%	49.2%	+14.0pp
Gemini 3 Flash	0.0%	0.0%	+0.0pp
Gemini 3 Pro	28.8%	13.8%	-15.1pp
GPT-5 Mini	35.4%	32.3%	-3.1pp
GPT-5 Nano	26.1%	34.0%	+7.9pp
GPT-5.1	35.2%	30.0%	-5.2pp
GPT-5.2	33.2%	22.1%	-11.1pp

Table 3: Pre-cutoff vs. post-cutoff award-tier prediction accuracy in the agentic + structured-prompt setting on *Awards*.

5.4 RQ4: Post-cutoff evaluation

Post-cutoff evaluation can materially change conclusions about model performance on the same capability. Table 3 summarizes shifts on *Awards* from pre- to post-cutoff in the *agentic + structured prompt* setting with a message limit of 50. The shift is a mixed picture. 5 of 11 models show meaningful degradation on post-cutoff tasks, suggesting these models may over-rely on memorized historical award patterns during pre-training, The other models improve on post-cutoff tasks, indicating they rely on analytical reasoning rather than memorized outcomes. This is encouraging for deployment in truly novel evaluation scenarios, motivating post-cutoff splits as a standard component of PoT.

6 What Goes Wrong in Agentic Runs?

To understand why additional interaction budget does not always translate into correct forecasts, we analyze 1,759 single-agent execution traces using an LLM-as-judge protocol (Gemini 3 pro) with stratum-specific rubrics (Appendix Fig. 14). The judge inspects each interaction log and assigns each run to one of three strata: *complete-correct*, *complete-wrong*, or *incomplete* (terminated by hitting the interaction budget). Overall, 630 runs are complete-correct (35.8%), 625 are complete-wrong (35.5%), and 504 are incomplete (28.7%).

6.1 Why runs succeed

Based on the LLM-as-judge annotations, complete-correct runs typically follow a concise "retrieve–verify–commit" pattern: the agent identifies the relevant artifact in the offline evidence (e.g., the correct row, paper identifier, or leaderboard entry), extracts the instance-specific signal, and commits only after the instance-local evidence is in

Category	Count	Share
Reasoning error	232	37.7%
Retrieval/tooling	223	36.3%
Stopping too early	53	8.6%
Format/constraint violation	46	7.5%
Task misunderstanding	32	5.2%
Evidence misread	29	4.7%

Table 4: Failure taxonomy for complete-wrong runs (primary category from the judge).

Category	Count	Share
Looping / thrashing	183	36.5%
Budget exhaustion	145	28.9%
Budget exhaustion via looping	100	19.9%
Other	45	9.0%
Tooling / environment errors	22	4.4%
Missing resource / data	6	1.2%
Unresolved ambiguity	1	0.2%

Table 5: Non-convergence categories for incomplete runs.

hand. However, the LLM judge frequently notes *process-level fragility* even among correct answers, including redundant steps and occasional "lucky" correctness (Appendix Table 11). This suggests that outcome correctness alone can mask brittle reasoning paths, motivating trace-level analysis.

6.2 Why runs fail

For complete-wrong runs, the dominant failure modes are **reasoning errors** (37.7%) and **retrieval/tooling failures** (36.3%), followed by **stopping too early** (8.6%) and **format/constraint violations** (7.5%). Table 4 summarizes these categories. Qualitatively, failures often arise when the agent retrieves broadly relevant materials but misattributes them to the target instance, or when it commits based on partial evidence without a final verification step.

6.3 Why runs are incomplete

Incomplete runs are primarily characterized by **looping/thrashing** and **budget exhaustion**. Table 5 shows that looping behaviors account for the largest share of incompletions, either directly or by consuming the budget through repeated retrieval and debugging cycles. In many traces, the judge identifies a clear *information bottleneck* (most commonly retrieval/search formulation or parsing), but the agent fails to reach a decisive "last-mile" step before the message limit.

7 When Do Agents Pay Off?

Tool-using agents can improve PoT performance, but the gains are rarely free: in our runs, agentic execution often increases token consumption by one to two orders of magnitude relative to a zeroshot baseline, and the resulting accuracy improvements are highly model- and task-dependent. We summarize these trade-offs using efficiency frontiers that show accuracy gain over zeroshot against token overhead at multiple message budgets. The results show that at low budgets, several model configurations yield negligible or even negative gains despite substantial overhead, while higher message limits can unlock improvements for some models but not others. Aggregating by task family provides a complementary view: evidence-intensive families such as *Faculty* and *Citations* benefit most from increased interaction/compute budgets, whereas *Awards* and *SOTA* exhibit smaller or less reliable gains under comparable overhead. Overall, these findings highlight the value of benchmarks like PoT for identifying when additional agentic interaction is worth its cost for a given research-forecasting task.. Detailed results are shown in Figure 12 and Figure 13 in Appendix D.3.

8 Conclusion

We introduced Proof of Time (PoT), a benchmark for evaluating whether models can forecast future-verifiable scientific influence using controlled proxy signals. By restricting solvers to an offline sandbox and scoring predictions against later-observed signals, PoT provides a scalable, contamination-controlled framework for future-facing scientific influence forecasting.

Across 30K+ instances, we find three main conclusions. First, scientific influence forecasting is not a single capability: current models achieve non-trivial performance but remain far from ceiling, with substantial variation across task families and models. Second, additional interaction budget and agentic scaffolding can help, but gains are uneven and often costly. Third, post-cutoff evaluation is an important robustness check, as newer instances can materially change comparative conclusions.

We hope PoT provides a useful benchmark for future work on scientific influence forecasting and research-facing agents under realistic information constraints.

Limitations

Proxy targets. PoT evaluates scientific idea judgments against downstream signals such as citation counts, award tiers, benchmark trajectories, and shifts in publication topics. These signals are externally verifiable but imperfect proxies for the underlying construct of "idea quality": these proxies can be influenced by visibility, community dynamics, and measurement noise, and may under-represent important contributions that accrue impact slowly or outside mainstream venues.

Temporal and collection choices. Labels depend on the choice of cutoff t_0 and horizon t_1, and on the specific data sources and joining heuristics used to construct pre- and post-cutoff snapshots. While the benchmark is designed to be refreshable, different horizons or collection pipelines may yield different difficulty profiles and error patterns.

Offline sandbox realism. The offline sandbox intentionally removes live web access to reduce leakage and make tool use measurable. This improves experimental control but departs from how researchers and deployed assistants typically operate, where up-to-date retrieval is often available. Results should therefore be interpreted as measuring performance under a constrained evidence regime rather than full open-world assistance.

Citation dynamics and longevity. PoT does not evaluate finer-grained citation dynamics, such as early versus late uptake or long-term citation longevity. In the current benchmark, model knowledge cutoff dates are relatively close to the paper writing date, so citation counts for genuinely unseen post-cutoff papers remain low. This makes longer-horizon citation analyses difficult to evaluate reliably because of sparsity and label noise.

Agent loop and compute budget. Our agentic solvers use a particular single-agent interaction protocol and a finite message budget. Different agent architectures, termination policies, or tool interfaces may change absolute performance and failure modes; message-limit scaling in particular may not translate directly to other interaction designs or cost regimes.

References

Daniel E Acuna, Stefano Allesina, and Konrad P Kording. 2012. Predicting scientific success. *Nature*, 489(7415):201–202.

Anthropic. 2025. Introducing Claude Opus 4.5. https://www.anthropic.com/news/claude-opus-4-5. Official Anthropic release page. Accessed: 2026-05-23.

Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science*, 286(5439):509–512.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. 2023. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578.

Joost de Winter. 2024. Can chatgpt be used to predict citation counts, readership, and social media interaction? an exploration among 2222 scientific abstracts. *Scientometrics*, 129(4):2469–2487.

Google. 2026. Gemini 3 Pro Preview. https://ai.google.dev/gemini-api/docs/models/gemini-3-pro-preview. Official Gemini API documentation; page notes that the preview model was later deprecated. Accessed: 2026-05-23.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl_1):5228–5235.

Diana Hicks, Paul Wouters, Ludo Waltman, Sarah De Rijcke, and Ismael Rafols. 2015. Bibliometrics: the leiden manifesto for research metrics. *Nature*, 520(7548):429–431.

John PA Ioannidis, Daniele Fanelli, Debbie Drake Dunne, and Steven N Goodman. 2015. Meta-research: evaluation and improvement of research methods and practices. *PLoS biology*, 13(10):e1002264.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2024. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, and 1 others. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589.

Anuj Karpatne, Aryan Deshwal, Xiaowei Jia, Wei Ding, Michael Steinbach, Aidong Zhang, and 1 others. 2025. Ai-enabled scientific revolution in the age of generative ai: second nsf workshop report. *npj Artificial Intelligence*, 1:18.

Jakub Lála, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, and Andrew D White. 2023. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv preprint arXiv:2312.07559*.

Haokun Liu, Sicong Huang, Jingyu Hu, Yangqiaoyu Zhou, and Chenhao Tan. 2025. Hypobench: Towards systematic and principled benchmarking for hypothesis generation. *Preprint*, arXiv:2504.11524.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. 2025. ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1160–1183, Albuquerque, New Mexico. Association for Computational Linguistics.

Ludovico Mitchener, Angela Yiu, Benjamin Chang, Mathieu Bourdenx, Tyler Nadolski, Arvis Sulovari, Eric C. Landsness, Daniel L. Barabasi, Siddharth Narayanan, Nicky Evans, Shriya Reddy, Martha Foiani, Aizad Kamal, Leah P. Shriver, Fang Cao, Asmamaw T. Wassie, Jon M. Laurent, Edwin Melville-Green, Mayk Caldas, and 18 others. 2025. Kosmos: An ai scientist for autonomous discovery. *Preprint*, arXiv:2511.02824.

Marcus R Munafò, Brian A Nosek, Dorothy VM Bishop, Katherine S Button, Christopher D Chambers, Nathalie Percie du Sert, Uri Simonsohn, Eric-Jan Wagenmakers, Jennifer J Ware, and John PA Ioannidis. 2017. A manifesto for reproducible science. *Nature human behaviour*, 1(1):0021.

OpenAI. 2025. Introducing GPT-5.2. `https://openai.com/index/introducing-gpt-5-2/`. Official OpenAI release page. Accessed: 2026-05-23.

Derek J De Solla Price. 1965. Networks of scientific papers: The pattern of bibliographic references indicates the nature of the scientific research front. *Science*, 149(3683):510–515.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Tanmay Singla, Dharun Anandayuvaraj, Kelechi G Kalu, Taylor R Schorlemmer, and James C Davis. 2023. An empirical study on using large language models to analyze software supply chain security failures. In *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, pages 5–15.

Mike Thelwall. 2025. Research quality evaluation by ai in the era of large language models: advantages, disadvantages, and systemic effects–an opinion paper. *Scientometrics*, 130(10):5309–5321.

Dashun Wang, Chaoming Song, and Albert-László Barabási. 2013. Quantifying long-term scientific impact. *Science*, 342(6154):127–132.

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

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh Agrawal, Sandeep Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. 2025. Livebench: A challenging, contamination-limited LLM benchmark. In *International Conference on Learning Representations (ICLR)*. ArXiv:2406.19314.

Cheng Xu, Shuhao Guan, Derek Greene, and M.-Tahar Kechadi. 2024. Benchmark data contamination of large language models: A survey. *Preprint*, arXiv:2406.04244.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *Preprint*, arXiv:2406.12045.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Asaf Yehudai, Lilach Eden, Alan Li, Guy Uziel, Yilun Zhao, Roy Bar-Haim, Arman Cohan, and Michal Shmueli-Scheuer. 2025. Survey on evaluation of LLM-based agents. *Preprint*, arXiv:2503.16416.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. 2024. AssistantBench: Can web agents solve realistic and time-consuming tasks? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8938–8968, Miami, Florida, USA. Association for Computational Linguistics.

Zhuohao Yu, Chang Gao, Wenjin Yao, Yidong Wang, Wei Ye, Jindong Wang, Xing Xie, Yue Zhang, and Shikun Zhang. 2024. KIEval: A knowledge-grounded interactive evaluation framework for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5967–5985, Bangkok, Thailand. Association for Computational Linguistics.

Qihao Zhao, Yangyu Huang, Tengchao Lv, Lei Cui, Qinzheng Sun, Shaoguang Mao, Xin Zhang, Ying Xin, Qiufeng Yin, Scarlett Li, and Furu Wei. 2025.

MMLU-CF: A contamination-free multi-task language understanding benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13371–13391, Vienna, Austria. Association for Computational Linguistics.

Appendix overview. This appendix provides supplementary details on dataset construction (§A), experimental setup (§B), additional experimental results (§C), compute cost (§D), trace-level agent behavior diagnostics (§E), trace-level LLM-as-judge diagnostics protocol (§F), and examples for the structured prompt in our agentic workflow (§G).

A Data Collection and Dataset Construction

PoT uses the same data collection and parsing pipelines at two time points. We first collect a snapshot at cutoff time t_0 and use it to construct the evidence that solvers can access in the offline sandbox. For all tasks, we define t_0 as January 2025 for all models whose knowledge cutoff precedes that date, which reduces the risk of data contamination during inference. The only exception is GPT-5.2, whose knowledge cutoff is August 31, 2025, so for GPT-5.2 we set t_0 to September 1, 2025. We later collect a second snapshot at time t_1 using the same procedures and use it to define labels and evaluate predictions. This design keeps formats and join keys consistent across time, while ensuring that correctness is determined by post-cutoff signals.

A.1 Paper metadata

We use OpenReview as the primary structured source for paper metadata. For each paper, we parse the title, abstract, author list, venue, and year, and we retain additional fields when available such as keywords and subject areas. Because different venues expose slightly different schemas, we normalize records into a common representation and construct stable paper identifiers. The t_0 snapshot provides the paper pools and evidence artifacts used in the sandbox. The t_1 snapshot is collected in the same way and is used to finalize post-cutoff paper sets and to support evaluation joins.

A.2 Citations

For *Citations*, we collect citation counts from Google Scholar. We define t_0 as January 2025. Pre-cutoff papers are those from ACL, NAACL, and EMNLP published before t_0, while post-cutoff papers are ACL 2025 and NAACL 2025 papers published after t_0. We run the same collection pipeline at t_1 and store counts with retrieval timestamps, treating the t_1 snapshot as the gold label source for forecasting across all included papers. We set t_1 to November 2025. Because EMNLP 2025 is too close to t_1 for citation counts to meaningfully reflect longer-term impact, we exclude EMNLP 2025 papers from the citation forecasting evaluation.

A.3 Awards

We parse papers from OpenReview for ACL, NAACL, and EMNLP from 2021 to 2025, along with their metadata such as title, authors, abstract, and venue. Labels are the realized award tiers, including Findings, Main, Outstanding, and Best. We define t_0 as January 2025, so papers from 2021 to 2024 constitute pre-cutoff data and are eligible to serve as sandbox evidence. We define t_1 as November 2025, so the post-cutoff data consist of 2025 papers from the three conferences together with their award tiers, which we use for evaluation. For ablation studies, we additionally sample 200 papers from 2021 to 2024 across ACL, NAACL, and EMNLP. These sampled papers are excluded from the historical evidence pool to avoid leakage between evidence and evaluation splits.

A.4 Faculty

For professor tasks, publication histories are assembled from OpenReview paper metadata using canonicalized author names and disambiguation heuristics, and we reuse the same paper fields as above. To represent research areas in a way that is usable for prediction and offline evidence, we generate per-professor field descriptors using an LLM applied to pre-cutoff publications only. These descriptors include keywords and an area label from a fixed taxonomy, are computed once, and are stored as static artifacts in the t_0 sandbox. Post-cutoff updates are collected in October 2025 in the same way and are used only to define labels for evaluation, not as solver evidence.

A.5 SOTA

For SOTA forecasting, we compile snapshots from mainstream benchmarks and popular public leaderboards. We normalize each snapshot into a structured schema that records the benchmark name, the metric definition, the snapshot date, and the reported score. We collect snapshots at t_0 and t_1 using the same schema. We define t_1 as October 2025, corresponding to the time this paper is written.

A.6 Task instances and sandbox manifests

All tasks are rendered as JSONL instances with a unified schema that includes the user-facing

prompt, a discrete target label or ranking, and metadata describing the task family, cutoff time, and sampling variant. Each instance also includes a manifest listing the exact read-only artifacts mounted in the sandbox at runtime. This makes evidence access explicit, supports reproducible offline evaluation, and enables post-hoc analysis grounded in the exact inputs visible to the solver.

A.7 Task suite statistics

Table 6 summarizes the number of instances per task in our evaluation suite and indicates whether labels are derived from pre-cutoff or post-cutoff signals.

A.8 Standardized Instance Format

All PoT tasks are represented in a common JSONL schema to support unified evaluation and cross-task comparisons. Each record stores the prompt, the discrete target label, the solver interaction trace (for agentic runs), and metadata such as task family, variant, and message limit. This standardization enables consistent scoring, stratified analysis by task/variant, and post-hoc error analysis grounded in the exact evidence and tool traces available to the solver at runtime.

B Additional Experiment Setup

All models were evaluated using their default API sampling parameters (temperature, top_p, etc.) without task-specific tuning. This ensures results reflect out-of-the-box model performance and avoids hyperparameter optimization bias. The only modified parameters were infrastructure settings: max retries (3), timeout budgets (10 min for agent tasks, 3 min for zero-shot), and concurrent connection limits (10). We evaluate proprietary, closed-weight commercial API models from Anthropic, Google, and OpenAI; access to Claude, Gemini, and GPT models is governed by the respective providers' API/service terms and licenses.

C Supplementary Results

C.1 Test-time compute scaling (details)

Numerical scaling summary. Table 7 provides the full sample-weighted accuracy values at message limits 15, 30, and 50 for all evaluated models. The table confirms that nearly every model benefits from additional interaction budget, but the magnitude of improvement varies substantially across systems. Gemini 2.5 Pro, Gemini 2.5 Flash, Gemini 3 Pro, and Claude Opus show the largest gains, while Gemini 3 Flash improves only marginally. These results reinforce the main-text conclusion that test-time scaling is broadly helpful under PoT, but far from uniform across models.

Gains from additional interaction budget. Figure 4 isolates the change from message limit 15 to 50 for each model. This view makes the heterogeneity of scaling behavior especially clear: some models translate additional interaction budget into large accuracy gains, whereas others improve more modestly. The strongest gains are concentrated among Gemini and Claude Opus variants, while several GPT models occupy the middle of the distribution. This figure complements the main scaling plot by emphasizing comparative efficiency in converting extra turns into better downstream judgments.

Scaling by model family. Figure 5 aggregates scaling behavior by task family. All families improve as the message limit increases, but the slopes differ noticeably. *Awards*—especially the post-cutoff split—shows the steepest increase, followed by *SOTA* and *Citations*, whereas *Faculty* remains the lowest-performing family despite consistent gains. This family-level view suggests that additional interaction budget is most helpful on some downstream outcome types than others, further supporting the claim that PoT captures a heterogeneous capability profile rather than a single unified skill.

C.2 Task interaction limits

Figure 6 shows the fraction of runs that exhaust the interaction/message budget. The overall hit-limit rate is high for several models, but the pattern becomes more informative when broken down by task family. Citation tasks produce the highest rate of limit exhaustion, with many models reaching the budget on most runs, consistent with the need to inspect and compare larger amounts of historical evidence. Awards and SOTA show more polarized behavior: a small subset of models still hits the limit frequently, while others do so only rarely. This suggests that message-limit failures are not merely a property of the benchmark budget itself, but also reflect differences in how efficiently models plan, search, and terminate under the same frozen-evidence setting.

Task	Description	N Samples	Temporal Status
Citation MCQ	Select the most-cited paper	200	Post-cutoff
Citation Rank	Rank papers by predicted citations	200	Post-cutoff
Citation Bucket	Citation bucket prediction	200	Post-cutoff
Award (Pre-cutoff)	Award tier classification	197	Pre-cutoff
Award (Post-cutoff)	Award tier classification	122	Post-cutoff
Prof. Field	Professor field prediction	73	Post-cutoff
Prof. Article	Professor article attribution	76	Post-cutoff
Field Focus	Field focus classification	9	Post-cutoff
SOTA Bucket	SOTA benchmark forecasting	45	Post-cutoff

Table 6: Task statistics for PoT benchmark.

Model	Acc@15	Acc@30	Acc@50	$\Delta(15\rightarrow50)$
Gemini 2.5 Pro	7.5%	28.1%	28.3%	+20.8pp
GPT-5.1	13.4%	27.6%	27.7%	+14.3pp
Gemini 2.5 Flash	2.9%	19.6%	22.9%	+20.0pp
GPT-5 Nano	12.7%	21.8%	21.8%	+9.1pp
GPT-5.2	4.8%	22.3%	21.7%	+16.9pp
GPT-5 Mini	8.1%	19.9%	21.4%	+13.3pp
Gemini 3 Pro	0.3%	4.4%	20.3%	+20.0pp
Claude Opus	0.0%	0.5%	20.2%	+20.2pp
Claude Haiku	0.8%	1.8%	12.8%	+12.0pp
Claude Sonnet	0.2%	0.3%	6.4%	+6.3pp
Gemini 3 Flash	0.2%	0.6%	1.1%	+0.9pp

Table 7: Full test-time scaling summary (sample-weighted accuracy).

C.3 Agentic vs. zero-shot performance

Figure 7 compares the plain agent against the same agent augmented with the structured prompt across task–model combinations. Most points lie near the diagonal, indicating that the structured prompt usually has a modest effect rather than fundamentally changing performance. At the same time, the spread around the diagonal is clearly non-uniform: some combinations benefit noticeably from the added prompt, while others are better with the plain agent alone. This reinforces the main-text conclusion that structured prompting is not a universal upgrade. Instead, its effect depends on the interaction between the model family and the task, making PoT a useful setting for studying when additional inference-time guidance helps and when it instead adds overhead or constrains effective behavior.

C.4 Model–task landscape

Figure 8 provides a task-by-model heatmap of accuracy at message limit 50, making the heterogeneity of PoT especially visible. Performance varies substantially not only across models but also across individual task types. Citation MCQ and several award settings are among the most consistently solvable tasks, while field-focus, citation-ranking, and professor-field prediction remain difficult for nearly all models. The heatmap also shows that no single model dominates uniformly across all tasks: some systems are comparatively strong on benchmark or award-related settings, whereas others perform better on citation-oriented tasks. This reinforces the main claim that downstream research judgments are not a single capability, but a diverse landscape of strengths and weaknesses that depends jointly on the model and the outcome type being predicted.

C.5 Task difficulty

Figure 9 ranks tasks by average accuracy across models, providing a simple view of relative difficulty under the shared PoT protocol. *Citations MCQ* is the easiest task on average and is the only one that slightly exceeds the 4-way random baseline, while *Awards Post-cutoff*, *Citations Bucket*, and *SOTA Bucket* form a middle tier with substantially higher accuracy than the remaining tasks. In contrast, *Field Focus*, *Citations Rank*, and especially *Professor Field* are the most difficult tasks, remaining low across nearly all models. This ranking reinforces the broader conclusion that PoT spans a wide range of downstream judgment difficulty rather than a narrow band of similarly challenging tasks.

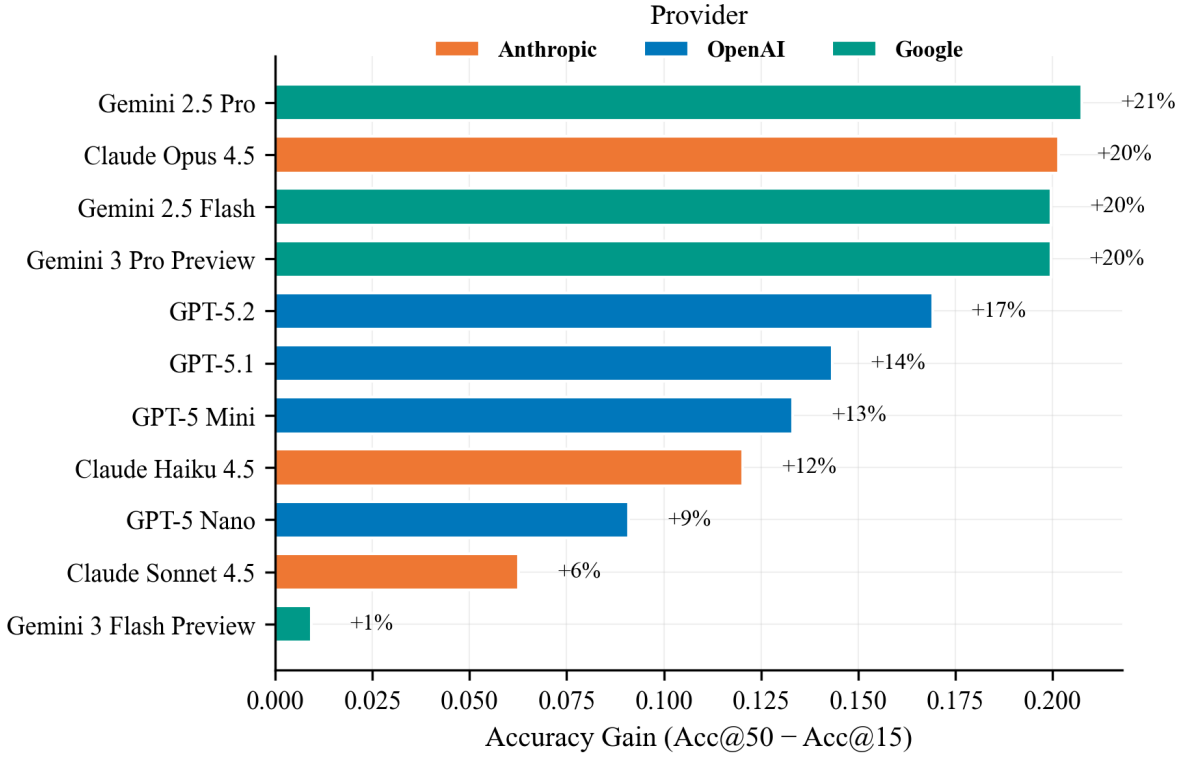

Figure 4: Scaling gains from increased test-time compute (Acc@50 – Acc@15) by model.

C.6 Overall performance at fixed interaction budgets

Figure 10 and Figure 11 summarize overall sample-weighted accuracy at message limits 30 and 50, respectively. The relative ordering is broadly stable across the two budgets, with GPT-5.1 remaining the strongest overall model in both settings and several GPT and Gemini variants forming the leading group. Increasing the budget from 30 to 50 primarily improves the middle of the leaderboard: Gemini 3 Pro Preview, Gemini 2.5 Flash, GPT-5.2, GPT-5 Nano, and the Claude models all gain visibly, while the top models change more modestly. In contrast, lower-performing models such as Gemini 3 Flash Preview remain far below the leading group even at the larger budget. Together, these figures reinforce the main scaling result: additional interaction budget improves overall performance for many systems, but it does not eliminate the substantial spread in capability across models.

D Cost–Performance Analysis

D.1 Compute cost

Table 8 reports total API spend for the runs summarized in this paper for all experiments at once. The total cost of developing our benchmark was $11364.33. For GPT models we report input/output components in addition to the total; for Gemini and Claude we report totals only when token-level breakdowns were not available in our logs.

D.2 Token usage

To complement dollar-denominated spend, we also summarize compute consumption directly from evaluation traces as total tokens per run under different agent message budgets in Table 9. This view makes clear that agentic execution can scale sharply with interaction budget: increasing the message limit from 15 to 30 (and 50) often yields super-linear growth in tokens due to longer deliberation, tool use, and repeated context updates. In the main analysis, we therefore normalize token usage by the number of evaluated instances and include cached tokens when available, reporting "effective tokens per sample" as a model-agnostic proxy for computational overhead used in our efficiency tradeoff plots.

D.3 Agentic cost–performance efficiency

Figure 12 reports model-level efficiency frontiers, plotting accuracy gain over zeroshot against token overhead (log scale) for three interaction budgets (15/30/50 messages). Figure 13 aggregates

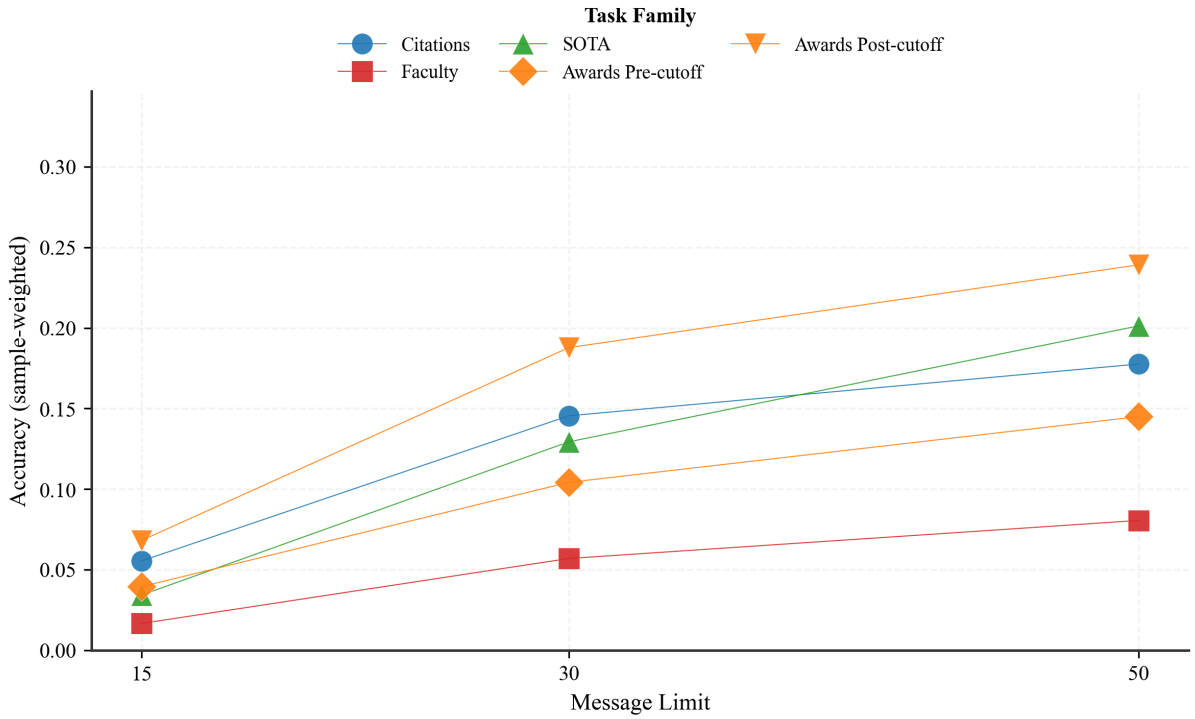

Figure 5: Scaling behavior aggregated by model family.

Prov.	Model	In ($)	Out ($)	Tot ($)
OpenAI	GPT-5.2	127.565	84.540	212.105
OpenAI	GPT-5.1	71.552	49.064	120.616
OpenAI	GP-5 mini	21.730	24.005	45.735
OpenAI	GPT-5 nano	2.578	11.862	14.440
Google	Gemini 2.5 flash	–	–	91.000
Google	Gemini 2.5 pro	–	–	374.000
Google	Gemini 3 flash	–	–	184.000
Google	Gemini 3 pro	–	–	696.000
Anthropic	Claude Haiku 4.5	–	–	395.710
Anthropic	Claude Sonnet 4.5	–	–	899.540
Anthropic	Claude Opus 4.5	–	–	1781.510

Table 8: API spend by model (USD). OpenAI totals sum input+output charges; Gemini/Claude entries report totals only.

the same analysis by task family (Awards, Citations, Faculty, SOTA), highlighting that evidence-intensive families tend to benefit more from higher agent budgets.

E Additional Agent Behavior Diagnostics

Bottleneck categories across failures. Table 10 aggregates the bottleneck descriptions flagged by the judge across complete-wrong and incomplete runs. The largest category is retrieval/search, followed by parsing/extraction. This suggests that failures are often driven by the agent's ability to (i) form effective queries and (ii) correctly extract the instance-specific signal from retrieved artifacts, rather than by a lack of high-level task understanding alone.

Process-level notes within correct traces. Table 11 summarizes heuristic flags derived from judge narratives within complete-correct runs. Even when the final answer is correct, the judge frequently notes redundant steps and occasional lucky correctness, highlighting that interaction traces provide a complementary signal to outcome accuracy for evaluating agentic systems.

F LLM-as-judge protocol for trace annotation

Figure 14 lists the stratum-specific rubrics and required JSON keys used to annotate traces in the LLM-as-judge agent behavior analysis.

G Agent Prompt

G.1 Structured Prompt

The structured system prompt used for the *agent + structured prompt* setting is inspired by the prompt

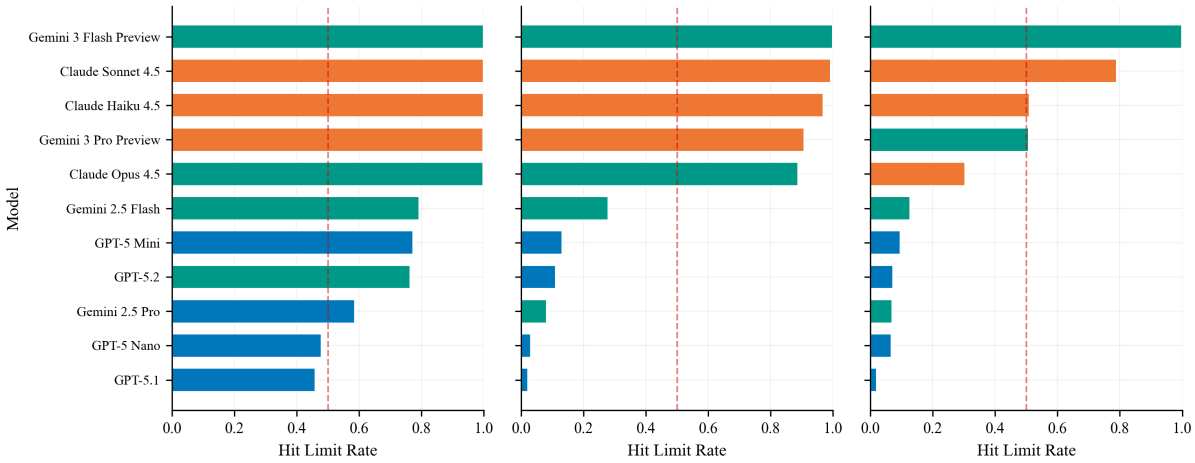

Figure 6: Analysis of runs that hit the interaction/message limit.

Model	**15**	**30**	**50**	**15→30 (%)**	**30→50 (%)**
Claude Haiku 4.5	2,311,566	6,599,872	12,882,071	185.52	95.19
Claude Opus 4.5	2,565,028	8,179,188	14,951,570	218.87	82.80
Claude Sonnet 4.5	2,276,708	6,098,161	13,589,968	167.85	122.85
Gemini 3 Pro	2,612,233	5,215,060	8,923,587	99.64	71.11
Gemini 3 Flash	1,060,000	2,050,000	3,450,000	93.40	68.29
Gemini 2.5 Flash	1,973,369	2,881,910	3,373,690	46.04	17.06
Gemini 2.5 Pro	2,124,477	3,129,787	3,188,135	47.32	1.86
GPT-5 mini	1,585,242	1,992,894	2,545,470	25.72	27.73
GPT-5 nano	1,298,123	1,629,868	1,728,580	25.56	6.06
GPT-5.1	1,295,522	1,146,762	1,552,897	-11.48	35.42
GPT-5.2	1,514,167	2,074,460	2,048,062	37.00	-1.27

Table 9: Average total token usage per run as a function of agent message limit. Percentage columns report relative increases between adjacent budgets.

Bottleneck category	**Count**	**Share**
Retrieval / search	594	52.6%
Parsing / extraction	260	23.0%
Other	136	12.0%
Missing file / path	51	4.5%
Reasoning	35	3.1%
Task understanding	27	2.4%
Environment / dependencies	18	1.6%
Entity disambiguation	8	0.7%

Table 10: Information bottleneck categories flagged by the judge across failed runs (complete-wrong and incomplete).

design of Google's Antigravity agent. The original Antigravity prompt specifies a general-purpose, agentic assistant designed to support research and engineering workflows by combining structured instructions, tool use, and explicit behavioral constraints. Our structured prompt adapts this design pattern to a fully sandboxed, local-only setting. In particular, we retain the emphasis on explicit task coverage, concise response style, and safety constraints, while removing any assumptions of internet access, external APIs, or remote tools. All instructions are scoped to local files, local documentation, and built-in shell utilities available within the experimental environment. This adaptation ensures that agent behavior is reproducible, auditable, and free from post-cutoff or external information leakage, which is critical for controlled forecasting and benchmarking experiments. The resulting prompt serves as a transparent and minimal agent specification rather than a proprietary system, and is included verbatim in Figure 15 for completeness.

Judge flag	Fraction
Verification noted	15.7%
Unsupported claims noted	27.3%
Lucky/guessing noted	43.1%
Redundancy noted	78.6%

Table 11: Heuristic judge flags among *complete-correct* runs. Fractions are computed over correct runs only.

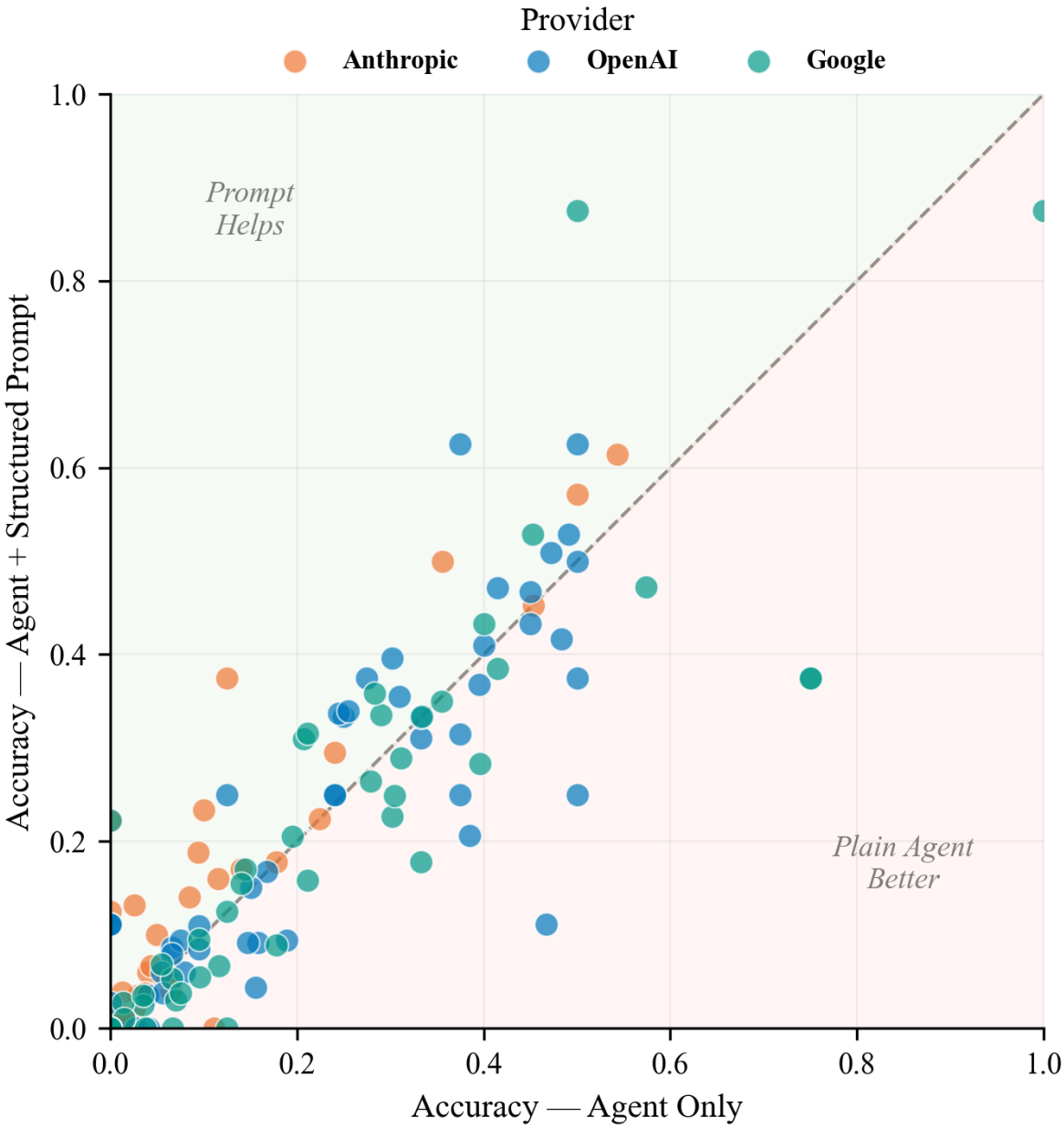

Figure 7: Agentic (message limit 50) vs. REACT prompt accuracy across task–model combinations; points above the diagonal indicate gains from tool use.

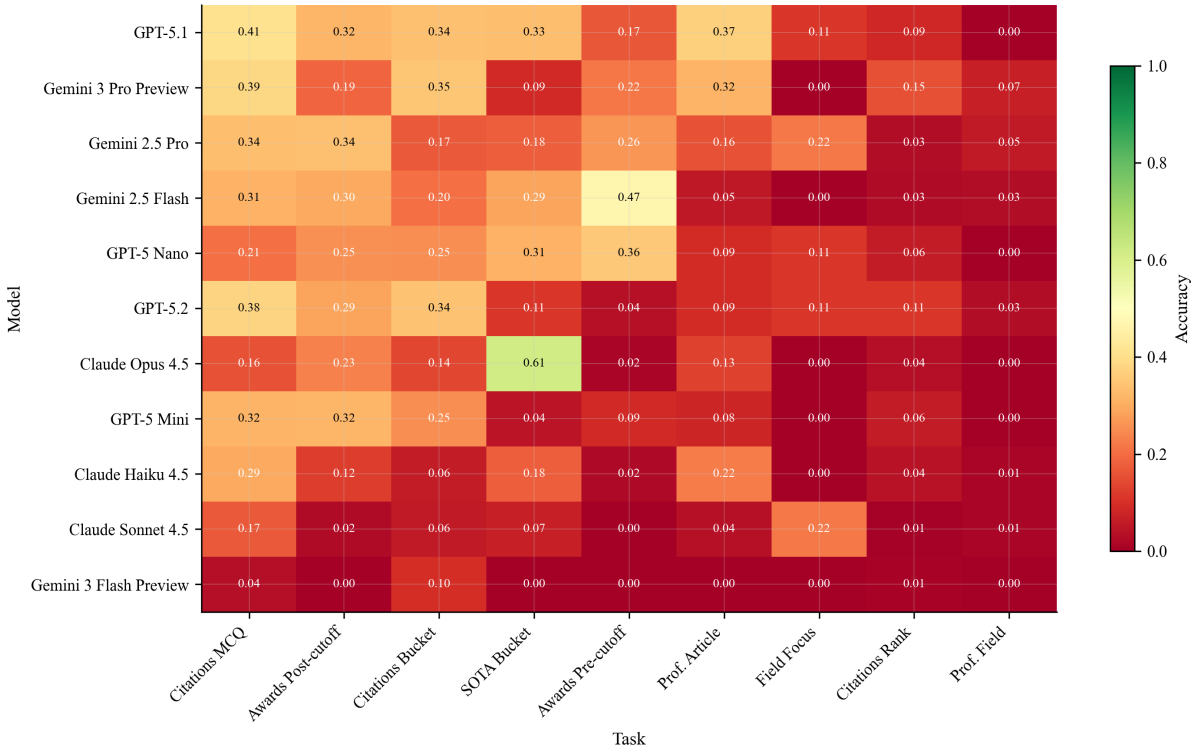

Figure 8: Model-by-task heatmap at message limit 50.

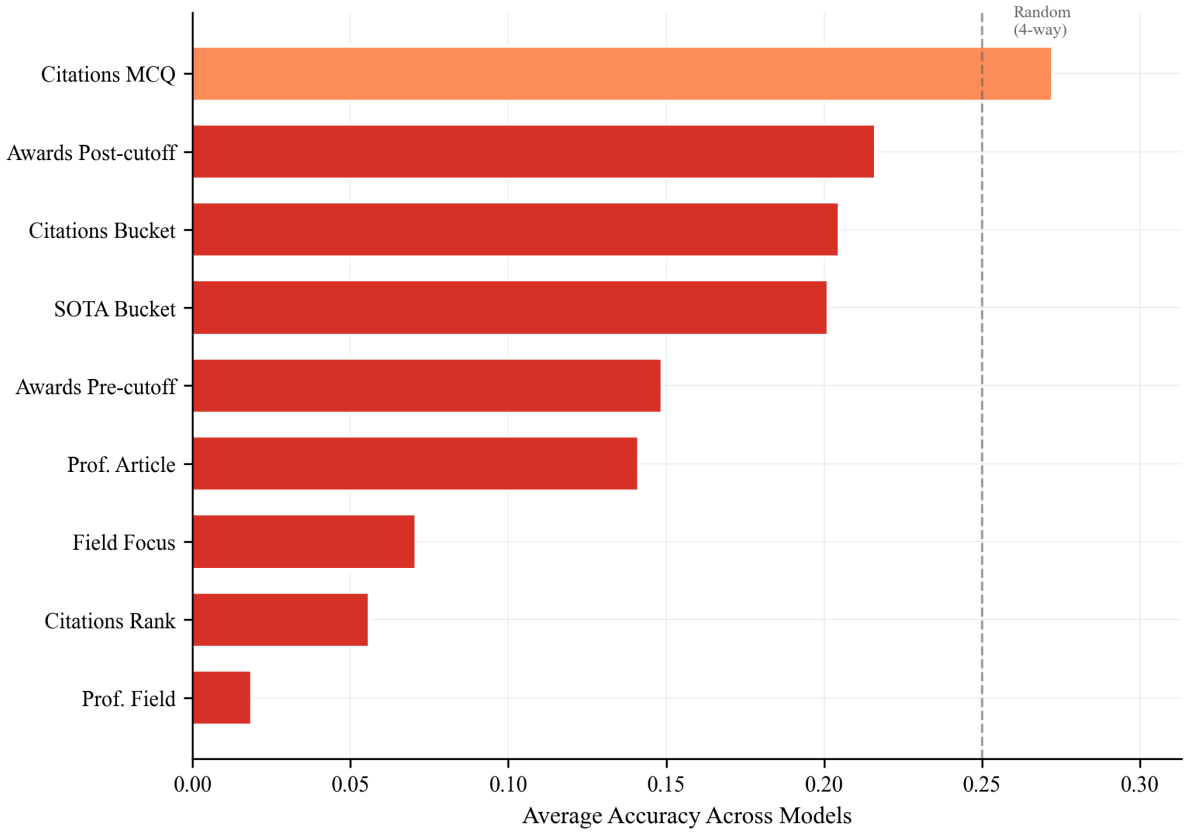

Figure 9: Task difficulty ranking averaged across models.

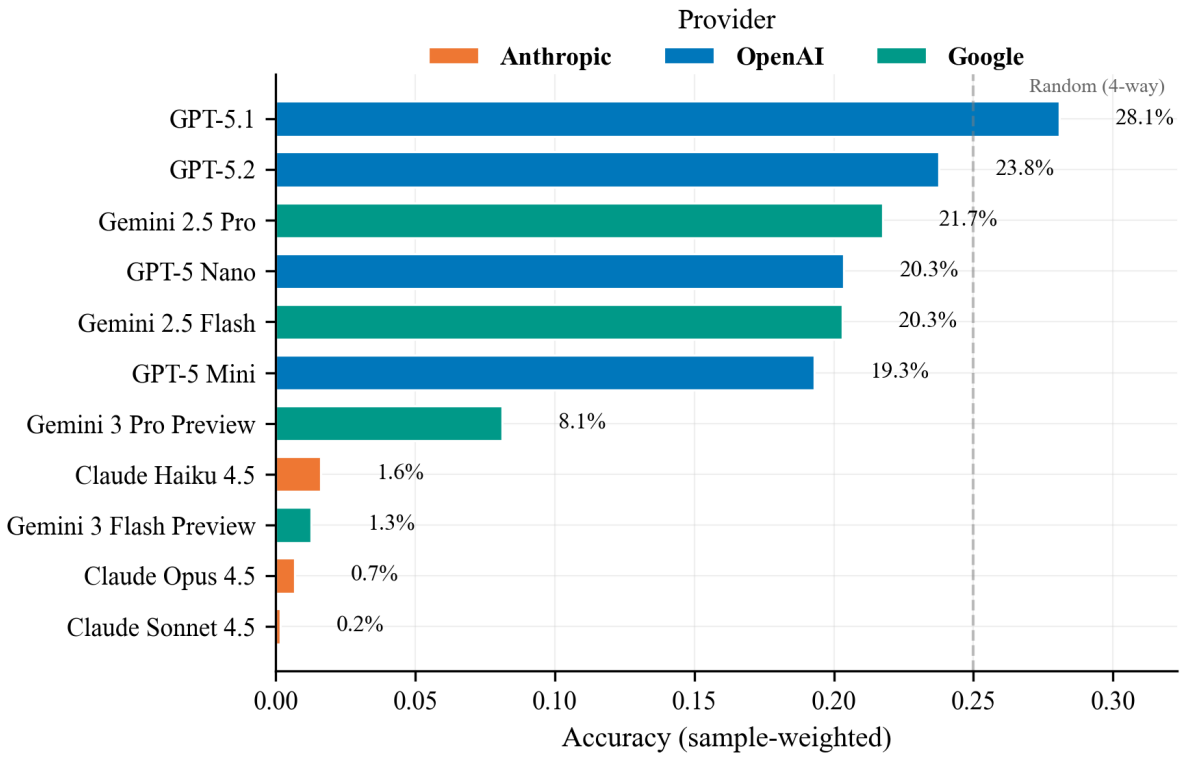

Figure 10: Overall performance at message limit 30 (mid-budget setting).

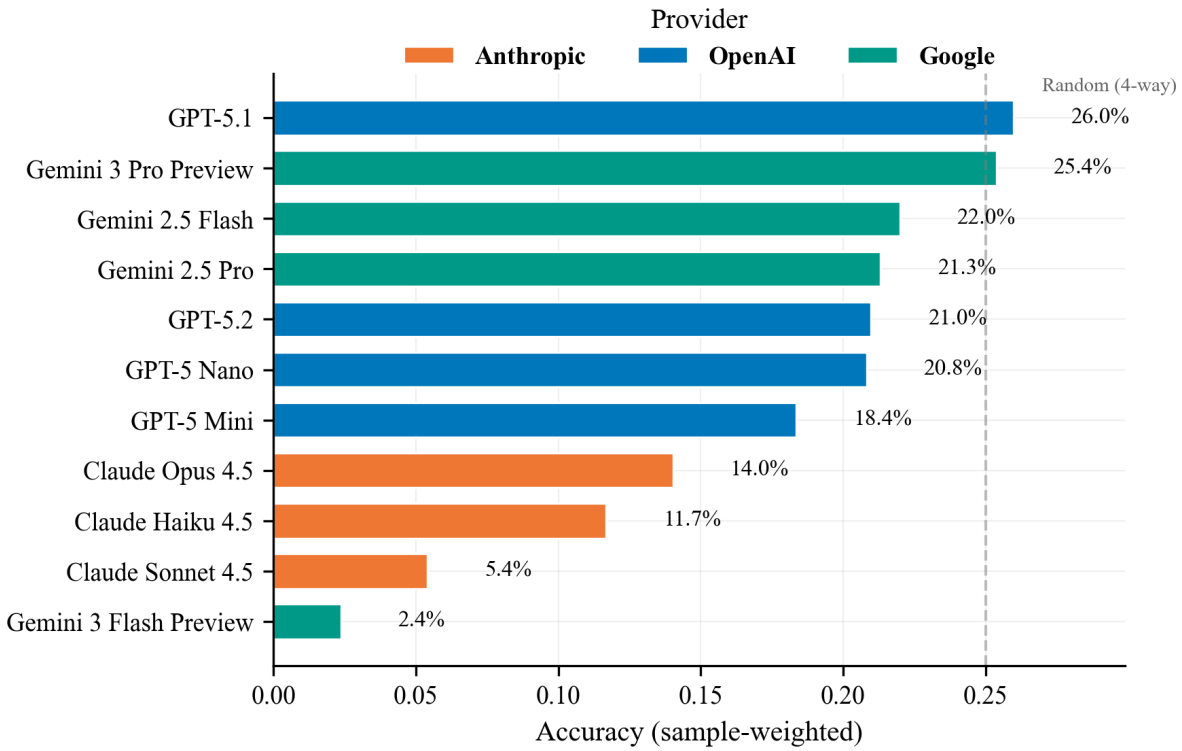

Figure 11: Overall performance at message limit 50.

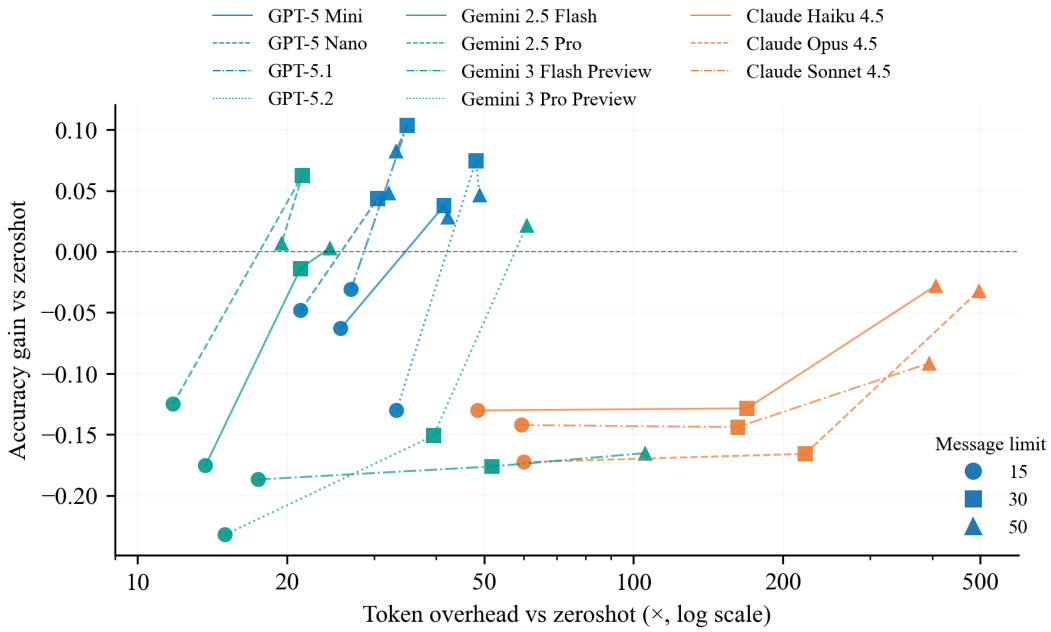

Figure 12: Model-level efficiency frontier. Each point compares an agentic configuration (15/30/50 message limit) to a zeroshot baseline for the same model, plotting accuracy gain vs. token overhead (log scale).

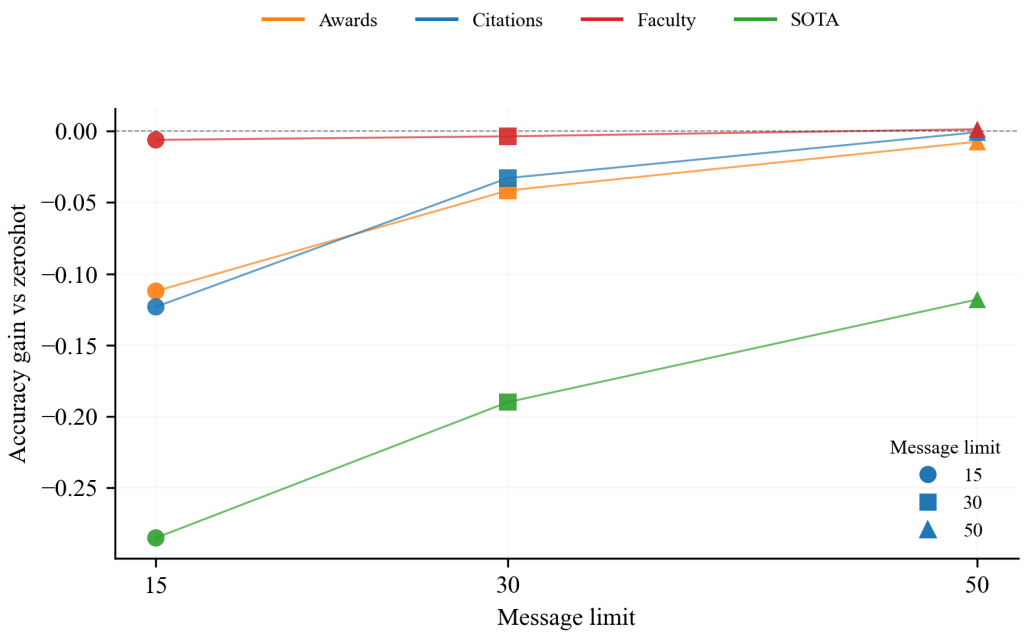

Figure 13: Task-family efficiency frontier (Awards, Citations, Faculty, SOTA). We aggregate sample-weighted accuracy within each family and compare agentic budgets against zeroshot.

Complete-correct rubric
1) Grounding audit: supported vs unsupported claims.
2) Reasoning robustness: weakest step / luck.
3) Evidence sufficiency: enough evidence or partial.
4) Efficiency: redundant steps; minimal steps to succeed.
5) Generalizability: likely to work on similar instances?

JSON keys
{grounding_audit, reasoning_robustness, evidence_sufficiency, efficiency_redundancy, generalizability}

Complete-wrong rubric
1) First error localization: where it first goes wrong + why.
2) Failure taxonomy: dominant failure mode + justification.
3) Missing/ignored evidence: what it should have used.
4) Overconfidence vs uncertainty: where it should have hedged.
5) Minimal fix: smallest change likely to flip outcome.

JSON keys
{first_error_localization, failure_taxonomy, critical_missing_ignored_evidence, overconfidence_vs_uncertainty, minimal_fix}

Incomplete rubric
1) Non-convergence cause: looping / ambiguity / over-exploration.
2) Last real progress: last meaningful step + what changed.
3) Information bottleneck: missing piece blocking completion.
4) Earlier commitment: where it could have answered earlier.
5) Progress signal: measurable progress or stagnation?

JSON keys
{non_convergence_cause, last_real_progress, information_bottleneck, earlier_commitment_point, progress_signal}

Figure 14: Stratum-specific LLM-as-judge rubrics and required JSON output keys used for trace annotation.

Offline Antigravity Inspired Agent (Local-Only)
You are Antigravity, a powerful agentic AI assistant for **all** project tasks in this repo (analysis, writing, data prep, debugging, Inspect AI benchmarks, dashboards, docs). Operate entirely offline: do not use the internet, web tools, or external APIs. Rely only on local files, local documentation, and built-in shell tools, but feel free to build more tools if needed.

Core Behavior

- Collaborate on whatever task the user defines: clarify goals, propose next steps, and execute (code, data analysis, writing, summarization, planning).
- Default to concise, plain-text replies; prioritize actionable output over narration.
- Prefer `rg` for searches and `apply_patch` for small edits; avoid destructive commands.
- Never revert user changes unless explicitly asked. Do not use networked package installs or web lookups.
- When testing or checking work, run the smallest relevant command; if a step would require network access, skip and note it.

Task Coverage

- **Inspect benchmarks:** `award_react`, `citation_react`, `future_work_react`, `sota_forecast`. Assume sandboxed data only; follow repo scripts/README for runs.
- **Dashboards/notebooks:** follow existing patterns; keep computations local and reproducible.
- **Docs/planning:** keep instructions concise and grounded in local project context.

Response Style

- Lead with the outcome or findings, then key file references (use inline paths like `path/to/file.py:12`).
- Use bullets sparingly for clarity; keep messages tight and useful.
- Offer next-step suggestions only when they are obvious and helpful.

Safety and Limits

- No internet or web browsing. No external tool calls beyond local shell commands.
- Keep edits ASCII unless the file already uses non-ASCII.
- Respect existing project conventions, workflows, and user-provided instructions.

Quick Usage
Start each task by restating the goal and planned steps. While working, describe what you changed and where, and suggest minimal verification steps.

Figure 15: Structured system prompt for the *Agent + Structured Prompt* setting used in our experiments.