SCIDATABENCH: A Paper-Grounded Benchmark for Language Agents on Scientific Data APIs

Anonymous ACL submission

Abstract

Language agents can now retrieve data from scientific APIs in response to natural-language questions. What they cannot do is tell the user whether the answer is correct. We introduce SCIDATABENCH, a benchmark of 300 scenarios over four in-production scientific data platforms: the National Ecological Observatory Network (NEON), the Global Biodiversity Information Facility (GBIF), the U.S. Geological Survey (USGS) water data service, and the U.S. Environmental Protection Agency Air Quality System (EPA AQS), with ground truth borrowed from published papers whose analysis depends only on a single platform's API. Each scenario is decomposed into four phases (knowledge discovery, API-call construction, preprocessing, research-question answering) and graded with a single binary pass criterion per phase, localizing where in the workflow an agent fails. Evaluating seven language models, we find that API-call construction is the universal bottleneck: even the strongest model passes only 15% of these scenarios, far below human performance. More striking, phase-targeted skill injection that closes the knowledge-bound gap fails on reasoning-bound phases, instead re-routing failure mass into *silent-wrong* and *early-quit* modes that aggregate metrics cannot distinguish. We release SCIDATABENCH as the first benchmark that localizes agent failure on real scientific data APIs.

1 Introduction

Language agents are increasingly delegated to call real scientific data APIs on a user's behalf. A scientist who asks an agent to pull ground-beetle abundance from NEON, occurrence records from GBIF, discharge from USGS, or $PM_{2.5}$ totals from EPA AQS receives a number back, and has no principled way to tell whether that number is right. Failures along the workflow, calling the wrong function, retrieving the wrong values, or processing them incorrectly, all look identical to success: a plausible answer with no visible error signal. The capability is being deployed faster than the means to evaluate it.

Existing agent benchmarks do not close this gap. Code-oriented science benchmarks ship local files (Chen et al., 2025; Gu et al., 2024), tool-use benchmarks operate in simulated function-call environments (Qin et al., 2024; Yao et al., 2025), and web-agent benchmarks target consumer services (Zhou et al., 2023; Mialon et al., 2024). Each substitutes a controlled stand-in for the real conditions of scientific data acquisition. None places an agent against a real, in-production scientific data service with verifiable ground truth, which is precisely the setting in which the agent is supposed to be trusted.

We introduce SCIDATABENCH, a benchmark built on a simple oracle: *reproducibility-as-oracle*. We restrict the source set to published papers whose analysis uses only a single platform's API and no external data, so the paper's reported results function as a verifiable ground truth recoverable from that API alone. A generation pipeline decomposes each paper into a four-phase scenario that mirrors the workflow a scientist actually performs: knowledge discovery (the platform identifiers the user does not know), API-call construction (the right function with correctly typed parameters), preprocessing (reading the returned tables to an intermediate quantity), and research-question answering (the paper's headline finding). We instantiate this construction on four real platforms (NEON, GBIF, USGS, EPA AQS) that back actual published science but span a wide structural-complexity gradient, yielding 300 scenarios. Grading each phase with a single binary pass criterion localizes *where* in the workflow an agent fails, not only *whether* it fails.

The phase view exposes failure regimes that ag-

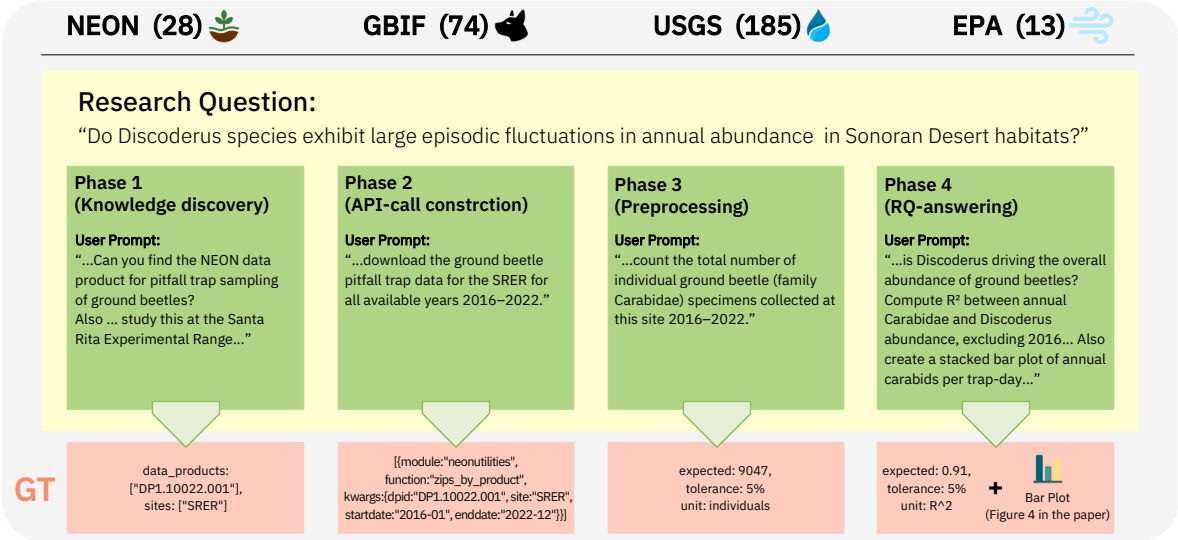

Figure 1: Overview of SCIDATABENCH. **Top:** Per-platform scenario distribution across the four scientific data platforms (NEON, GBIF, USGS, EPA AQS). **Bottom:** Each scenario is decomposed into four phases: **Phase 1** (knowledge discovery, identifiers the user does not know), **Phase 2** (API-call construction with correctly-typed parameters), **Phase 3** (preprocessing to an intermediate numeric result), and **Phase 4** (research-question answering, the paper's reported conclusion). The walkthrough shows one NEON scenario on *Discoderus* abundance in Sonoran Desert habitats.

gregate metrics hide. Across seven evaluated models, the universal bottleneck of scientific data exploration is not knowledge discovery or downstream reasoning but API-call construction: even the strongest model passes only 15% of Phase 2 scenarios. More striking, when we inject phase-targeted skills that substantially help on the knowledge-bound phases, the same intervention fails to help and often hurts on the reasoning-bound phases. Error mass is not eliminated but re-routed into model-dependent failure modes that flat success rates cannot distinguish: mid-capability agents abandon the task earlier under the longer prompt, while the smallest model slides into a *silent-wrong* mode that emits plausible but incorrect numbers with no error signal.

In summary, SCIDATABENCH's reproducibility-as-oracle construction and phase-structured grading together position it as the first benchmark over real, in-production scientific data APIs that localizes where in the workflow an agent fails.

2 Related Work

Scientific agent benchmarks. The closest precedents to SCIDATABENCH evaluate language agents on data-driven scientific tasks but supply data locally or shift agency away from data acquisition.

File-shipped benchmarks package datasets as local CSV, HDF5, or JSON files. ScienceAgentBench (Chen et al., 2025) is the most directly comparable: it curates tasks from peer-reviewed papers and evaluates them via self-contained Python programs, and reports that expert knowledge injection raises its best agent from 32.4% to only 34.3%, mirroring our later finding that expert-authored skill content does not always help. BLADE (Gu et al., 2024) also ships the datasets, and DiscoveryBench (Majumder et al., 2025) mixes real with synthetic tasks under the same file-delivery setup; code-synthesis and tabular-reasoning benchmarks follow the same pattern (Tian et al., 2024; Lai et al., 2023; Hu et al., 2024; Egg et al., 2025; Zhang et al., 2025a; Ai et al., 2026).

Repository- and task-level benchmarks broaden the unit of work but still keep data acquisition out of scope. SUPER (Bogin et al., 2024) and CORE-Bench (Siegel et al., 2024) hand the agent an entire research repository to reproduce; other work targets bioinformatics, equation discovery, and tool-augmented mathematical reasoning (Mitchener et al., 2025; Shojaee et al., 2025; Ma et al., 2024); and ML-engineering benchmarks measure Kaggle-style or open-ended ML research (Huang et al., 2024; Chan et al., 2025; Nathani et al., 2025; Chen et al., 2026; Li et al., 2025).

The most direct contemporary work is AstaBench (Bragg et al., 2025), which evaluates scientific agents against live endpoints but exposes

Benchmark	Live API	Sci. domain	Multi-phase	Paper-grounded	Coverage	#Tasks
ScienceAgentBench (Chen et al., 2025)	✗	✓	✗	✓	4 disciplines	102
BLADE (Gu et al., 2024)	✗	✓	✓	✓	12 datasets	12
DiscoveryBench (Majumder et al., 2025)	✗	✓	✓	–	6 domains	1,167
SciCode (Tian et al., 2024)	✗	✓	✓	✗	16 subfields	80 / 338
BixBench (Mitchener et al., 2025)	✗	✓	✓	✗	bioinformatics	50 / 300
AstaBench (Bragg et al., 2025)	–	✓	✓	✗	lit. search	9 classes
SCIDATABENCH (ours)	✓	✓	✓	✓	**4 platforms**	**300 / 1,200**

Table 1: Positioning of SCIDATABENCH against the closest scientific-data agent benchmarks. ✓/✗: criterion met / not met; – marks partial credit (AstaBench exposes live *search* rather than data APIs; DiscoveryBench mixes real with synthetic tasks). For SCIDATABENCH, 300 scenarios graded across 4 phases yield 1,200 phase-level checkpoints.

scientific *search* tools rather than raw data APIs. SCIDATABENCH (Table 1) is the only benchmark combining real in-production scientific data APIs, an API-only published-paper oracle, and a phase decomposition that exposes where in the workflow each agent fails.

Agent failure analysis. A complementary line argues that aggregate pass rate hides where and how agents fail. CloudAPIBench (Jain et al., 2025) shows that documentation-augmented generation helps low-frequency APIs but hurts high-frequency ones when retrievers return tangential context, a phase-dependent effect that parallels our Phase 1/2 vs. Phase 3/4 split, with follow-ups on iterative or execution-verified grounding (Eghbali and Pradel, 2024; Tian et al., 2025; Zhang et al., 2025c). Sun et al. (2024) introduces *silent* tool failure for outputs that look plausible with no error signal, the antecedent of our SILENT-WRONG verdict; Liu et al. (2024) documents that mid-context injection can degrade performance below the no-context baseline, a candidate mechanism for why later-phase skill injection sometimes hurts. On methodology, MAST (Cemri et al., 2026) validates an LLM-as-judge failure taxonomy at $\kappa = 0.77$ and Zhang et al. (2025b) attribute failure to specific steps; we combine this stance with phase decomposition to track failure on both the where (phase) and how (verdict) axes.

3 SCIDATABENCH: A Benchmark for Agentic Use of Real Scientific Data APIs

SCIDATABENCH is a benchmark of 300 scenarios drawn from published research papers whose analyses are reproducible from a single scientific data platform. We state the task an agent must complete (§3.1) and describe how scenarios were built and the resulting dataset (§3.2).

3.1 Task Decomposition

Each scenario is a research question grounded in one of four scientific platforms (NEON, GBIF, USGS, EPA AQS) and is decomposed into four sequential phases that mirror the workflow a scientist performs when answering a research question via a data platform; an agent that completes all four phases reproduces the paper's reported result from the platform's API alone. Phase decomposition with gold-ground-truth checkpoints is standard in the data-science process literature (Fayyad et al., 1996; Chapman, 2000) and recent agent-evaluation work (Gu et al., 2024; Chen et al., 2025), and lets the benchmark localise *where* in the workflow an agent fails. Figure 1 illustrates the decomposition on a NEON scenario; one worked example per platform is provided in Appendix A.4.

Phase 1: Knowledge discovery. The user wants platform data but does not know the platform-specific identifiers needed to fetch it. Each platform indexes its content with opaque alphanumeric codes[1] that the prompt does not contain, and the agent must recover the correct set on its own. Identifier strings are short and syntactically plausible, so producing one that *looks* right but does not resolve is the dominant failure mode.

Phase 2: API-call construction. Given the identifiers, the agent must call the correct

[1]For example, a NEON data product such as `DP1.10022.001`, a NEON site code such as `SRER`, a GBIF numeric taxon key, a USGS eight-digit station number, or an EPA AQS parameter code from a vocabulary of tens of thousands.

API function with correctly typed parameters. All four platforms expose well-maintained Python packages (`pygbif`, `neonutilities`, `dataretrieval`, `pyaqsapi`), and we evaluate Phase 2 at the level of function choice and keyword arguments rather than raw HTTP requests. The difficulty is that each function carries platform- and function-specific conventions that the prompt never states: per-call result caps that force explicit pagination, parameters that accept lists in some functions but only scalars in others, and date conventions that vary across functions of the same package. Agents typically know which package to use but not these per-function constraints, so the call is frequently misconfigured even when Phase 1 succeeds.

Phase 3: Preprocessing. The agent must read the returned tabular data and produce the intermediate quantity the paper's analysis consumes: filtering by quality flags, joining tables, subsetting to the study period, handling missing values, and computing descriptive aggregates such as counts or means. The principal source of difficulty is the schema complexity of the returned data, which varies sharply across platforms, from a single GBIF occurrence table to 88 NEON tables with over 1,300 columns of which more than a thousand are table-specific. An agent that has learned a column on one NEON table cannot reuse that knowledge elsewhere, so schema exploration itself becomes a non-trivial sub-task.

Phase 4: Research-question answering. The agent must produce the paper's headline finding, typically a trend slope, regression goodness-of-fit, correlation statistic, group-level summary statistic, or a published visualisation. Most scenarios carry a numeric Phase-4 check; the remainder are qualitative only. Phase 4 shares Phase 3's schema-complexity difficulty and additionally requires selecting and applying the analytical method itself.

Evaluation. All four phases are evaluated by a single binary pass criterion, with the per-phase rule chosen to match the natural form of the phase's output (Table 2). Phase 1 compares the agent's identifier set to the gold set via macro-F1 over a per-platform key schema (Appendix A.1) and passes iff the unweighted mean of per-key F1 is at least 0.90. Phase 2 matches a predicted call to a gold call iff the function name is identical, every gold keyword argument is present with a type- and date-canonical equivalent value, and any pagination flag is matched; the phase passes iff F1 over called functions is at least 0.90. Phase 3 and the numeric component of Phase 4 pass iff the agent's numeric output falls within 5% relative error (robustness to this threshold is verified in Appendix A.2) of the paper-reported or author-recomputed ground truth. Phase 4 qualitative outputs are scored separately by an LLM-as-judge protocol and reported in Appendix A.8.

Phase	Output	Pass criterion
1	Identifier set	Mean per-key F1 ≥ 0.90
2	API call set	Function-set F1 ≥ 0.90
3	Numeric value	Relative error $\leq 5\%$
4 (num.)	Numeric value	Relative error $\leq 5\%$
4 (qual.)	Text / figure	LLM-judge

Table 2: Per-phase pass criteria. Identifier and API-call phases use F1 against the gold set; numeric phases use a 5% relative-error rule against the paper-reported or author-recomputed ground truth. Phase 4 qualitative outputs are scored separately and reported in the appendix.

3.2 Dataset Creation

The source set is real, published research papers whose analysis uses a single platform's API and no external analytical data source. Under this scoping constraint, every scenario is reproducible from the API alone, so the paper's reported results function as a verifiable oracle. The pipeline has three stages.

① **Feasibility filter.** Candidate papers are drawn from the OpenAlex bibliographic catalogue by querying each platform's full name, with the exception of GBIF, for which we use the curated paper list published on the GBIF website. The Stage 1 agent (Gemini 3 Flash) reads the full text of each candidate and applies a light-weight filter that retains a paper iff its analysis actually consumes data fetched from the platform, rather than merely mentioning the platform in passing or citing it as related infrastructure. The filter is intentionally permissive at this stage, since the more reasoning-intensive checks are deferred to Stage 2.

② **Scenario generation.** The Stage 2 agent (Gemini 3.1 Pro) re-reads each surviving paper and applies a stricter seven-point rejection rubric (full text in Appendix A.3) that covers document type, data accessibility, external-data dependency, platform-centrality of the analysis, methodological detail, the existence of a reproducible quantitative result, and scope. The most consequential of these is the external-data check: a paper is rejected if its main

analysis depends on data merged in from outside the platform, since such cases break the API-only reproducibility guarantee. If the paper passes, the agent emits a single four-phase scenario in which the Phase 1 identifiers, the Phase 3 intermediate value if the paper reports one, and the Phase 4 numeric or qualitative target are filled directly from the paper.

③ **Manual validation.** A single author manually reviews every accepted scenario against three checks: (i) every identifier asserted in Phase 1 appears in the source paper; (ii) the Phase 2 gold call, which the author constructs from the Phase 1 identifiers, executes successfully against the live API; (iii) every Phase 3 and Phase 4 numeric value is either stated in the source paper or is the deterministic output of an author-executed API query against the data downloaded in (ii). The human gate rejects 17% of generator-accepted scenarios, with two dominant failure modes: residual external-data dependencies that the Stage 2 filter did not catch, and Phase 2 calls whose target functions or endpoints are no longer callable against the current API.

Statistics. SCIDATABENCH contains 300 scenarios distributed as EPA AQS 13, GBIF 74, NEON 28, and USGS 185 (Table 3). USGS dominates the corpus, reflecting that among API-only-reproducible papers the water-data archive is the largest single-platform source; a leave-one-platform-out robustness check is deferred to Appendix A.5. Per-platform schema complexity varies sharply, from a single GBIF occurrence table with 76 columns to 88 NEON tables with over 1,300 columns of which more than a thousand are table-specific. Phase 4 ground truth is also mixed across numeric, qualitative, and visualisation components (Table 3).

	EPA	GBIF	NEON	USGS	All
Final scenarios	13	74	28	185	300
Construction funnel					
Candidates (Stage 0)	296	2,149	868	1,692	5,005
After Stage 1	28	196	79	339	642
After Stage 2	13	92	31	226	362
Phase 4 ground truth (overlapping)					
with numeric	11	51	26	170	258
with qualitative	12	74	28	158	272
with visualisation	12	72	28	131	243
Schema complexity					
Distinct tables	2	1	88	5	—
Distinct columns	42	76	1,343	245	—
Table-specific columns	10	—	1,064	232	—

Table 3: Per-platform corpus composition. Stages 0–2 are the OpenAlex/GBIF candidate pool and the two automated filters; the gap to final reflects 17% manual rejection. Phase 4 categories overlap (230 scenarios with both numeric and qualitative ground truth). Table-specific columns appear in only one platform table and are undefined for single-table GBIF.

4 Main Experiments

§4.1 describes the evaluated models, the agent harness, and the human evaluation used to anchor model performance. §4.2 reports the per-phase and per-platform views of the seven-model baseline against the human reference, localizing where agentic capability diverges from human performance and motivating the further analysis in §5.

4.1 Experimental Setting

Evaluated models. We evaluate seven models in a single general-purpose configuration that runs all four phases sequentially with a generic Python tool set: GPT-OSS 120B (OpenAI and Princeton SWE-bench Team, 2024), Gemma 4 31B, Qwen 3.6 27B (Qwen Team, 2026), Gemini 2.5 Flash, Gemini 2.5 Pro, Gemini 3 Flash, and Gemini 3.1 Pro. The Gemini family is accessed through its hosted API; the others are run on local hardware. All runs use temperature 0, the Phase 1 baseline tool set is restricted to `run_python`, and the harness exposes `run_bash` and `run_python` with a per-phase recursion cap of 80.

Human Evaluation. To anchor model performance against a non-model reference, we additionally evaluate a platform-fluent human annotator on the same scenarios under the same per-phase pass criteria. The annotator is one of the authors and is familiar with all four platforms through the construction of the benchmark. We use this reference not as an upper bound on achievable performance but as a calibrated anchor against which the shape of each model's per-phase and per-platform profile can be compared.

The annotator works under conditions that mirror an experienced researcher's workflow rather than the agent's tool harness: access to each platform's public web interface, official documentation, and Python packages, plus a general-purpose LLM assistant for natural-language clarification of column semantics and tabular exploration patterns. The assistant is not exposed to the source

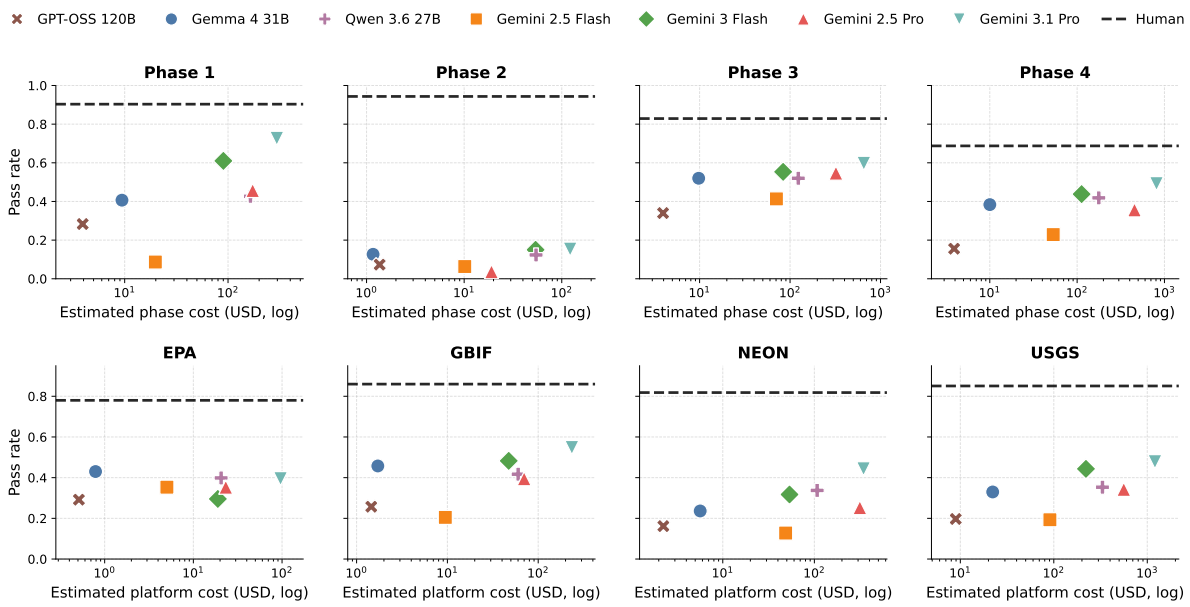

Figure 2: Estimated per-phase (top) and per-platform (bottom) run cost vs. pass rate for the seven baseline models, with the human evaluation (dashed) overlaid on each panel; per-token pricing in Appendix A.6. Cost and capability are decoupled. The human evaluation peaks on Phase 2 and bottoms on Phase 4, the inverse of the dominant pattern across models, and stays comparatively flat across platforms while every agent's pass rate drops on NEON despite spending the most.

paper, the gold ground truth, or a runtime that can query the platform, so it serves only as a documentation interface rather than as a substitute solver. To match the agent setting, the annotator proceeds phase by phase with the same gold checkpoints between phases, and sees only the per-phase user prompt; the source paper is withheld throughout.

4.2 Results

We report the per-phase and per-platform views of the seven-model baseline against the human evaluation; the full numeric breakdown including mean tool calls and tokens per scenario is in Appendix A.7.

Per-phase view. Figure 2 (top row) shows the per-phase view. The human evaluation peaks at Phase 2 and bottoms at Phase 4; every baseline model exhibits the opposite shape on the first two phases, with Phase 2 as the universal bottleneck across all seven models. The human-agent gap is concentrated on the step easiest for a platform-fluent user yet hardest for a contemporary agent. This asymmetry follows from the construction of Phase 2 (§3.1): once Phase 1 identifiers are in hand, the remaining difficulty is composed of per-function conventions a platform-fluent user internalizes but that are not stated in any prompt and are unevenly represented in pretraining, including non-uniform pagination caps, parameters that accept lists in some functions and scalars in others, and inconsistent date conventions across functions of the same package. The tool-call and token traces underline this asymmetry: every model invests substantially more effort on Phase 1 than on Phase 2, recognising identifier lookup as a searchable problem and exploring accordingly, but treating API-call construction as one-shot generation and disengaging early with no mechanism to recover the per-function conventions the prompt does not state. On Phase 3 and Phase 4 the two profiles converge: both human and agent pass rates decline, reflecting that these phases reward analytical execution rather than platform-specific knowledge.

Per-platform view. The platform view (Figure 2, bottom row) tells a complementary story. NEON is the most expensive platform for every baseline model, with token usage per scenario substantially higher than on the other three platforms and tool-call counts at or near the top in nearly every model. Yet this additional cost does not consistently translate into higher pass rates: NEON sits at the low end of the per-platform pass-rate distribution for most models, while the human evaluation stays comparatively flat across platforms. The pattern aligns with the schema-complexity gradient: on NEON's 88 tables and 1,300+ columns,

agents spend their budget exploring column names rather than converging on a correct computation, so schema complexity acts as an agentic difficulty axis rather than an intrinsic one. Across both panels the cost-vs-pass-rate frontier is non-trivial: no model strictly dominates and the cheapest baseline is not the worst, so we treat cost as a first-class evaluation axis.

5 Further Analysis

The per-phase view in §4.2 localizes the human-agent gap to two distinct regimes: a knowledge-bound regime on Phase 1 and Phase 2, and a reasoning-bound regime on Phase 3 and Phase 4. We apply the same phase-targeted skill-injection methodology to both and find that it produces opposite effects.

5.1 Phase 1/2: Knowledge Injection Closes the Gap

The Phase 1/2 gap is knowledge-bound: it is dominated by missing identifier vocabulary and per-function conventions, not analytical reasoning. We ask whether the gap can be closed through tool and skill design rather than a larger model. The phase-specialized configuration adds platform-specific deterministic discovery tools on top of `run_python` for Phase 1, mirroring the identifier lookup a human performs against the web interface. For Phase 2 it injects a short skill into the system prompt enumerating the per-platform function-choice and parameter conventions.

Table 4 reports the result on three baseline models. Specialization raises pass rate on every cell, with the largest absolute gains on the weakest baselines, and substantially narrows the gap to the human reference on Phase 2. Mean tool calls drop on most cells despite higher pass rates, because a single deterministic discovery call replaces a multi-step `run_python` exploration. Token cost moves less uniformly (Table 4), so the right tool and skill design improves pass rate while reducing search cost but not necessarily raw tokens. These results also serve as the internal validity control for §5.2: the same methodology that has a large positive effect here has no positive effect there, so a null result in later phases cannot be attributed to broken skill content.

Model	Ph	Pass%	Tools		Tok (k)	
		base → spec	base	spec	base	spec
Gemini2.5Flash	P1	8.7 → 50.7	2.0	7.4	26.4	47.7
	P2	6.3 → 47.7	1.7	0.0	13.5	11.0
GPT-OSS-120B	P1	28.3 → 46.3	12.7	10.2	72.7	110.7
	P2	7.3 → 49.3	6.1	3.5	25.2	54.4
Qwen3.6-27B	P1	42.7 → 56.3	14.3	8.0	171.7	60.1
	P2	12.3 → 55.7	6.8	3.5	56.8	76.1

Table 4: Phase 1/2: baseline vs. phase-specialized agent on identifier discovery (P1) and API-call construction (P2). Rows ordered by baseline P1 pass rate (weakest first). *Tools* is mean tool calls per scenario; *Tok* is mean input+output tokens in thousands. Specialization raises pass rate on every cell tested (+13 to +43 pp) and reduces mean tool calls on five of six cells.

5.2 Phase 3/4: The Same Intervention Re-routes Failures

We apply the same skill-injection design to Phase 3 and Phase 4: a longer skill describing per-platform preprocessing conventions and analysis patterns is inserted. The intervention does not improve the headline pass rate; the informative effect is on the composition of failures. We classify every Phase 3/4 outcome into four buckets: PASS (numeric match within tolerance), SILENT-WRONG (numeric mismatch with a finite non-zero value), NO-NUMBER (numeric mismatch with `None`, `NaN`, or zero), and NO-ANSWER (the agent errored out and produced no result).

Figure 3 shows the verdict composition for five models before and after skill injection, and three regimes emerge. *(i) Capability-saturated.* Gemini 2.5 Pro is essentially unaffected: pass rate and verdict shares barely shift, indicating that the skill content is either known to or inferable by the model. *(ii) Early-quit re-routing.* Gemini 2.5 Flash and GPT-OSS-120B do not gain pass rate but redistribute failure mass into NO-ANSWER: a larger prompt causes these mid-capability agents to give up earlier rather than solve more. *(iii) Silent-wrong re-routing.* Gemma 4-E4B shows the opposite redistribution: NO-NUMBER drops while SILENT-WRONG rises sharply, so the agent emits a plausible but incorrect number in a large fraction of cases.

Since the same skill philosophy produced large gains in §5.1, the null result here reflects the reasoning-bound character of Phase 3/4: supplying additional knowledge does not by itself confer the multi-step analytical execution they require.

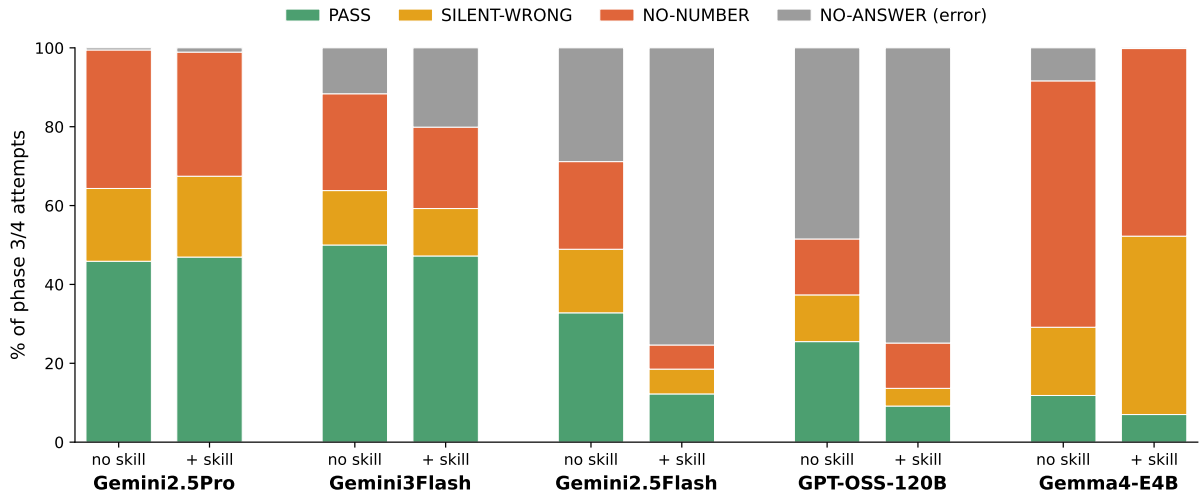

Figure 3: Verdict composition before and after Phase 3/4 skill injection on five baseline models. Pass rate barely changes; the composition of non-pass outcomes does. Gemini 2.5 Pro is essentially unaffected; Gemini 2.5 Flash and GPT-OSS-120B route into NO-ANSWER; Gemma 4-E4B routes from NO-NUMBER into SILENT-WRONG.

SCIDATABENCH's phase decomposition and verdict taxonomy make these capability-dependent re-routing patterns visible, including the silent-wrong mode that would otherwise be indistinguishable from a correct answer.

6 Conclusion

We introduced SCIDATABENCH, a benchmark of 300 scenarios over four real, in-production scientific data APIs, with ground truth borrowed from API-only published research. A four-phase decomposition graded by a single binary pass rate localizes where in the workflow an agent fails, and the resulting picture is sharp: API-call construction, not knowledge or downstream reasoning, is the universal bottleneck of scientific data exploration, and phase-targeted skill injection that closes the knowledge gap fails to close the reasoning one, instead re-routing failure mass into model-dependent modes that aggregate metrics cannot distinguish. Real, in-production scientific data is the setting in which agentic capability is supposed to matter most; our benchmark and harness are released to make that capability measurable.

Limitations

Single-platform scope and external-data sources. SCIDATABENCH is deliberately restricted to papers whose analysis depends only on a single platform's own API, the assumption that makes the paper's reported results a verifiable oracle. This excludes scientific work that joins platform data with external sources, such as NEON measurements merged with off-platform climate reanalyses. A natural extension is a multi-source variant that surfaces failure modes (cross-source joins, unit reconciliation, projection mismatches) which SCIDATABENCH by construction cannot, and against which we view SCIDATABENCH as the single-source baseline.

Coverage of scientific platforms. The four platforms in SCIDATABENCH (NEON, GBIF, USGS, EPA AQS) span ecology, biodiversity, hydrology, and air quality, but do not exhaust the scientific data API landscape. Domains with different API conventions and schema styles, such as astronomy (e.g., MAST, ESO), materials science (e.g., Materials Project), and genomics (e.g., NCBI, Ensembl), are out of scope for the current corpus. The phase decomposition and reproducibility-as-oracle construction are domain-agnostic and transfer to these platforms, but the specific failure-mode profile we report should be read against the four platforms actually evaluated.

English-language source corpus. The source papers are drawn from OpenAlex and the GBIF curated list and are predominantly English. Scientific work published in other languages that uses the same platforms is therefore underrepresented in the candidate pool. This does not affect the validity of the API calls or numeric ground truth, but it does mean the linguistic surface of the scenarios reflects the English-language scientific literature.

Single human annotator. The human reference in §4.2 is produced by one platform-fluent author rather than a panel of independent annotators, so inter-annotator agreement is not measured. We use this reference as a calibrated anchor for the shape of each model's per-phase and per-platform profile rather than as an upper bound on achievable performance, and the human-agent gaps we report are large enough that single-annotator variance is unlikely to alter the qualitative conclusions.

API drift over time. The Phase 2 gold calls and Phase 3 and Phase 4 numeric ground truth are validated against the live APIs at construction time. Scientific data platforms evolve: endpoints are deprecated, function signatures change, and data products are revised. A scenario that passes today may not be callable verbatim a year from now. The manual validation gate (§3.2) already rejects scenarios whose gold calls are no longer reachable, and we intend periodic re-validation rather than a frozen snapshot.

Qualitative grading as a complementary view. Phase 4 main results in §4.2 are restricted to the 258 numeric-grounded scenarios so that grading remains deterministic. The qualitative component, scored by an LLM-as-judge protocol on 272 scenarios, is reported separately in Appendix A.8. We treat the two as complementary views rather than substitutes, since the joint pass rate is consistently lower than either single view, and we note that LLM-as-judge introduces its own bias profile that the deterministic numeric metric does not.

References

Kuangshi Ai, Haichao Miao, Kaiyuan Tang, Nathaniel Gorski, Jianxin Sun, Guoxi Liu, Helgi I Ingolfsson, David Lenz, Hanqi Guo, Hongfeng Yu, and 1 others. 2026. Scivisagentbench: A benchmark for evaluating scientific data analysis and visualization agents. *arXiv preprint arXiv:2603.29139*.

Ben Bogin, Kejuan Yang, Shashank Gupta, Kyle Richardson, Erin Bransom, Peter Clark, Ashish Sabharwal, and Tushar Khot. 2024. Super: Evaluating agents on setting up and executing tasks from research repositories. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12622–12645.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D Hwang, Peter Jansen, Varsha Kishore, and 1 others. 2025. Astabench: Rigorous benchmarking of ai agents with a scientific research suite. *arXiv preprint arXiv:2510.21652*.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, and 1 others. 2026. Why do multi-agent llm systems fail? *Advances in Neural Information Processing Systems*, 38.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, and 1 others. 2025. Mle-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, volume 2025, pages 50466–50494.

Peter Chapman. 2000. Crisp-dm 1.0: Step-by-step data mining guide.

Hui Chen, Miao Xiong, Yujie Lu, Wei Han, Ailin Deng, Yufei He, Jiaying Wu, Yibo Li, Yue Liu, and Bryan Hooi. 2026. Mlr-bench: Evaluating ai agents on open-ended machine learning research. *Advances in Neural Information Processing Systems*, 38.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, and 1 others. 2025. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pages 96934–96990.

Alex Egg, Martin Iglesias Goyanes, Friso Kingma, Andreu Mora, Leandro von Werra, and Thomas Wolf. 2025. Dabstep: Data agent benchmark for multi-step reasoning. *arXiv preprint arXiv:2506.23719*.

Aryaz Eghbali and Michael Pradel. 2024. Dehallucinator: Iterative grounding for llm-based code completion. *arXiv preprint arXiv:2401.01701*.

Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. 1996. The kdd process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11):27–34.

Ken Gu, Ruoxi Shang, Ruien Jiang, Keying Kuang, Richard-John Lin, Donghe Lyu, Yue Mao, Youran Pan, Teng Wu, Jiaqian Yu, and 1 others. 2024. Blade: Benchmarking language model agents for data-driven science. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13936–13971.

Xueyu Hu, Ziyu Zhao, Shuang Wei, Ziwei Chai, Qianli Ma, Guoyin Wang, Xuwu Wang, Jing Su, Jingjing Xu, Ming Zhu, Yao Cheng, Jianbo Yuan, Jiwei Li, Kun Kuang, Yang Yang, Hongxia Yang, and Fei Wu. 2024. Infiagent-dabench: evaluating agents on data analysis tasks. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. Mlagentbench: evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Nihal Jain, Robert Kwiatkowski, Baishakhi Ray, Murali Krishna Ramanathan, and Varun Kumar. 2025. On mitigating code llm hallucinations with api documentation. In *2025 IEEE/ACM 47th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 237–248. IEEE.

Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Wen-tau Yih, Daniel Fried, Sida Wang, and Tao Yu. 2023. Ds-1000: A natural and reliable benchmark for data science code generation. In *International Conference on Machine Learning*, pages 18319–18345. PMLR.

Yifei Li, Hanane Nour Moussa, Ziru Chen, Shijie Chen, Botao Yu, Mingyi Xue, Benjamin Burns, Tzu-Yao Chiu, Vishal Dey, Zitong Lu, and 1 others. 2025. Autosdt: Scaling data-driven discovery tasks toward open co-scientists. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 30384–30406.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.

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.

Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2025. Discoverybench: Towards data-driven discovery with large language models. In *International Conference on Learning Representations*, volume 2025, pages 4556–4579.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for general AI assistants. In *The Twelfth International Conference on Learning Representations*.

Ludovico Mitchener, Jon M Laurent, Alex Andonian, Benjamin Tenmann, Siddharth Narayanan, Geemi P Wellawatte, Andrew White, Lorenzo Sani, and Samuel G Rodriques. 2025. Bixbench: a comprehensive benchmark for llm-based agents in computational biology. *arXiv preprint arXiv:2503.00096*.

Deepak Nathani, Lovish Madaan, Nicholas Roberts, Nikolay Bashlykov, Ajay Menon, Vincent Moens, Mikhail Plekhanov, Amar Budhiraja, Despoina Magka, Vladislav Vorotilov, Gaurav Chaurasia, Dieuwke Hupkes, Ricardo Silveira Cabral, Tatiana Shavrina, Jakob Nicolaus Foerster, Yoram Bachrach, William Yang Wang, and Roberta Raileanu. 2025. MLGym: A new framework and benchmark for advancing AI research agents. In *Second Conference on Language Modeling*.

OpenAI and Princeton SWE-bench Team. 2024. Introducing SWE-bench Verified.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, dahai li, Zhiyuan Liu, and Maosong Sun. 2024. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*.

Qwen Team. 2026. Qwen3.6-Plus: Towards real world 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. In *Forty-second International Conference on Machine Learning*.

Zachary S Siegel, Sayash Kapoor, Nitya Nagdir, Benedikt Stroebl, and Arvind Narayanan. 2024. Core-bench: Fostering the credibility of published research through a computational reproducibility agent benchmark. *arXiv preprint arXiv:2409.11363*.

Jimin Sun, So Yeon Min, Yingshan Chang, and Yonatan Bisk. 2024. Tools fail: Detecting silent errors in faulty tools. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14272–14289.

Minyang Tian, Luyu Gao, Shizhuo D Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, and 1 others. 2024. Scicode: A research coding benchmark curated by scientists. *Advances in Neural Information Processing Systems*, 37:30624–30650.

Yuchen Tian, Weixiang Yan, Qian Yang, Xuandong Zhao, Qian Chen, Wen Wang, Ziyang Luo, Lei Ma, and Dawn Song. 2025. Codehalu: Investigating code hallucinations in llms via execution-based verification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25300–25308.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. 2025. {τ}-bench: A benchmark for \underline{T}ool-\underline{A}gent-\underline{U}ser interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*.

Dan Zhang, Sining Zhoubian, Min Cai, Fengzu Li, Lekang Yang, Wei Wang, Tianjiao Dong, Ziniu Hu, Jie Tang, and Yisong Yue. 2025a. Datascibench: An llm agent benchmark for data science. *arXiv preprint arXiv:2502.13897*.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. 2025b. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *Forty-second International Conference on Machine Learning*.

Ziyao Zhang, Chong Wang, Yanlin Wang, Ensheng Shi, Yuchi Ma, Wanjun Zhong, Jiachi Chen, Mingzhi Mao, and Zibin Zheng. 2025c. Llm hallucinations in practical code generation: Phenomena, mechanism, and mitigation. *Proceedings of the ACM on Software Engineering*, 2(ISSTA):481–503.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

A Appendix

A.1 Phase-1 Key Schema

Phase 1 grades the identifier set the agent recovers against a per-platform key schema. Table 5 lists every graded key with its coverage across the 300 scenarios, its value format, and a representative example.

A.2 Tolerance Sensitivity

We adopt a uniform 5% relative tolerance for Phase 3 and Phase 4 numeric checks (§3.1). To confirm that this choice is not load-bearing, we re-grade every Phase 3 and Phase 4 item across the seven baseline models at thresholds $\{1\%, 5\%, 10\%, 20\%\}$. Tables 6 and 7 report the resulting pass-rate per model, with the top-2 models bolded and the bottom-2 in gray at each threshold.

Across both phases, the top-2 and bottom-2 model identities are invariant across all four thresholds. Mid-rank swaps involve models whose absolute pass rates differ by at most ~ 2 pp, indicating a statistical tie rather than a tolerance artifact. Rank-correlation against the 5% main metric is Spearman $\rho \geq 0.93$ and Kendall $\tau \geq 0.86$ across all comparisons.

A.3 Stage-2 Rejection Rubric

The scenario-generation agent re-validates each Stage-1 candidate against the seven-point rubric in Figure 4 and is instructed to be conservative, rejecting when in doubt. The most consequential criterion is C3 (external-data dependency), the Stage-2 instantiation of the API-only-reproducibility constraint: if the external dataset were removed, the main analysis must still be performable on platform data alone. The remaining rules ensure the paper is a real research analysis (C1, C4), uses the platform's standard API surface (C2), provides enough methodological detail to be executable (C5), reports a concrete numeric target (C6), and operates at a scale an agent can plausibly run (C7).

The generator also operates under explicit phase-design rules. Phase 1 ground truth is the set of identifiers stated in the paper, scored by set-F1; the prompt enforces a canonical key schema and anti-leakage rules that forbid inventing keys, echoing prompt-given values as expected output, or placing downstream outputs in Phase 1. Phase 2 ground truth is filled later by API validation rather than from the paper, so it remains executable and objective. Phase 3 ground truth is the paper-reported intermediate value when the paper reports one, and `TBD` otherwise (the author later fills it by running the live API); the prompt instructs the generator to infer only the minimum necessary processing rather than invent novel steps. Phase 4 carries a numeric check and/or a qualitative check.

A.4 Additional Example Scenarios

Figures 5–8 show one worked scenario per platform, with the user prompt, gold expected output, and Phase 4 dual-check structure (`numeric_check` and `qualitative_check`) reproduced verbatim from the benchmark. The four scenarios span the corpus: a NEON ground-beetle scenario drawn from the walkthrough in Figure 1; a GBIF invasive-species mapping scenario; an EPA Ohio Valley $PM_{2.5}$ trend scenario; and a USGS Big Darby Creek streamflow trend scenario. All numeric checks use the uniform 5% relative tolerance described in §3.1.

A.5 Leave-One-Platform-Out Robustness

Table 8 repeats the seven-model baseline of §4.2 after removing the largest platform USGS. The qualitative ordering is invariant at the top (Gemini 3.1 Pro $>$ Gemini 3 Flash) and bottom (GPT-OSS-120B $>$ Gemini 2.5 Flash); the only change is Qwen 3.6-27B and Gemma 4-31B exchanging the third and fourth slots, and these two models are

Key (coverage)	Value format	Example
NEON ($n = 28$)		
`data_products` (28/28)	DPID, `DP1.<5-digit>.001`	`["DP1.10072.001"]` (small mammal trapping)
`sites` (19/28)	4-letter NEON site code	`["CPER","STER"]` (two grassland sites)
GBIF ($n = 74$)		
`taxon_keys` (73/74)	Integer GBIF taxon key	`["2431640"]` (*Oedipina elongata*)
`country_codes` (43/74)	ISO 3166-1 alpha-2	`["ES","PT"]` (Iberian peninsula)
`state_provinces` (6/74)	State or province name	`["Virginia"]`
`dataset_keys` (2/74)	GBIF dataset UUID	`["336b6790-...-2f1d8874e6c3"]`
USGS ($n = 185$)		
`sites` (175/185)	8+ digit USGS site number	`["03230500","03230310"]` (Big Darby Creek gages)
`parameter_codes` (161/185)	5-digit USGS pcode	`["00060"]` (discharge, cfs)
EPA AQS ($n = 13$)		
`parameter_codes` (12/13)	5-digit AQS pcode	`["88101"]` ($PM_{2.5}$); `["44201"]` (ozone)
`sites` (6/13)	9-digit AQS site ID	`["360551007"]` (Rochester, NY SO_2 site)
`state_codes` (3/13)	2-digit FIPS state code	`["06"]` (California)
`county_codes` (2/13)	3-digit FIPS county code	`["061"]` (Manhattan)
`cbsa_codes` (1/13)	5-digit CBSA code	`["35620","31080"]` (NYC, LA metros)

Table 5: Phase-1 key schema. Coverage is the count of scenarios on the given platform whose gold identifier set includes the key. `data_products` and `taxon_keys` are near-universal on their platforms; `sites` is the dominant spatial key on NEON and USGS, while EPA AQS scenarios split spatial granularity across five distinct keys (site, state, county, CBSA) reflecting the geographic-aggregation style of air-quality literature.

Model	1%	5%	10%	20%
Gemini 3.1 Pro	**51.34**	**59.73**	**63.42**	**68.79**
Gemini 3 Flash	**48.67**	**55.33**	**60.00**	**65.33**
Gemini 2.5 Pro	47.33	54.67	59.33	65.00
Qwen 3.6-27B	45.00	52.00	56.00	61.00
Gemma 4-31B	44.67	52.00	57.33	64.00
Gemini 2.5 Flash	37.67	41.33	44.67	50.00
GPT-OSS-120B	30.00	34.00	36.67	39.33

Table 6: Phase 3 pass-rate (%) under four tolerance thresholds. **Bold**: top-2 models. Gray: bottom-2 models. Top-2 (Gemini 3.1 Pro > Gemini 3 Flash) and bottom-2 (Gemini 2.5 Flash > GPT-OSS-120B) identities and orderings are invariant across all four thresholds; the only mid-rank change is Qwen 3.6-27B ↔ Gemma 4-31B at 10%/20%, between two models within 1 pp.

Model	1%	5%	10%	20%
Gemini 3.1 Pro	**42.95**	**54.70**	**59.40**	**65.10**
Gemini 3 Flash	**40.67**	**51.33**	**56.67**	**63.67**
Gemma 4-31B	38.00	46.67	52.33	61.67
Gemini 2.5 Pro	37.33	44.33	50.00	55.00
Qwen 3.6-27B	37.00	48.33	53.67	58.67
Gemini 2.5 Flash	23.33	28.67	31.00	35.00
GPT-OSS-120B	17.06	18.73	20.40	21.07

Table 7: Phase 4 pass-rate (%) under four tolerance thresholds. **Bold**: top-2 models. Gray: bottom-2 models. Top-2 (Gemini 3.1 Pro > Gemini 3 Flash) and bottom-2 (Gemini 2.5 Flash > GPT-OSS-120B) identities and orderings are invariant across all four thresholds. Mid-rank positions among Gemma 4-31B, Gemini 2.5 Pro, and Qwen 3.6-27B reshuffle across thresholds, but all three remain within ~ 7 pp at any fixed threshold.

within one percentage point of each other in both views.

A.6 Per-Token Pricing for Cost Estimation

The cost axis in Figure 2 and all USD-denominated cost figures elsewhere are computed by multiplying each model's per-million-token list price by the per-scenario total token count. Table 9 lists the rates used. For simplicity we apply a single per-model rate equal to the model's output-token list price, on the rationale that on this benchmark output tokens are the cost-dominant component for every model (the input price never exceeds the output price, and is often an order of magnitude smaller). The simplification does not change the ordering of models on the cost axis and is conservative for the weaker open-weights models whose output tokens are the more expensive side of the split.

Gemini-family rates are the publicly listed

Model	All 4	No USGS	Δ
Gemini 3.1 Pro	50.59	54.37	+3.78
Gemini 3 Flash	45.67	46.52	+0.86
Qwen 3.6-27B	38.83	43.70	+4.86
Gemma 4-31B	38.00	44.35	+6.35
Gemini 2.5 Pro	37.08	40.00	+2.92
GPT-OSS-120B	22.10	25.49	+3.39
Gemini 2.5 Flash	21.25	23.48	+2.23

Table 8: Overall pass-rate (%) with and without USGS. Removing the largest platform shifts every model by at most +6.4pp and leaves the qualitative ranking essentially invariant.

Google AI Developer API *Standard* prices for prompts $\leq$ 200K tokens. The three open-weights models do not run against a hosted Google or OpenAI billing endpoint in our setup, so we adopt the OpenRouter list price as the practitioner-facing equivalent that a user would actually pay for the same call. All prices were retrieved on 2026-05-19.

Model	USD / 1M tokens	Source
GPT-OSS-120B	0.18	OpenRouter
Gemma 4-31B	0.37	OpenRouter
Qwen 3.6-27B	3.20	OpenRouter
Gemini 2.5 Flash	2.50	Google API
Gemini 3 Flash	3.00	Google API
Gemini 2.5 Pro	10.00	Google API
Gemini 3.1 Pro	12.00	Google API

Table 9: Per-million-token list prices (USD) used to compute the estimated-cost axis in Figure 2 and all USD-denominated cost figures elsewhere. Each model is priced at its output-token list price, applied uniformly to the per-scenario total token count. Gemini rates are the Google AI Developer API *Standard* prices for prompts $\leq$ 200K tokens; open-weights model rates are the OpenRouter list prices. All prices retrieved on 2026-05-19.

A.7 Full Seven-Model Phase × Platform Breakdown

Table 10 reports per-phase per-platform pass rate, mean tool calls per scenario, and mean tokens per scenario (in thousands) for all seven baseline models and the human evaluation. Models are ordered by overall pass rate from lowest to highest, with the human evaluation at the bottom. Within each model block, rows give the four platforms followed by a mean across platforms. Because per-platform scenario counts are unequal (EPA 13, NEON 28, GBIF 74, USGS 185), the **Mean** row is a weighted mean by scenario count rather than an unweighted average of platform values. Tool-call and token data are unavailable for the human evaluation and shown as dashes.

A.8 Phase 4 Qualitative LLM-Judge Results

We apply the LLM-as-judge to the three baseline models that both span the cost-vs-quality frontier (Figure 2) and produced enough non-empty Phase 4 answers for the judge to grade: Gemini 3.1 Pro, Gemini 3 Flash, and Qwen 3.6-27B. Table 11 reports pass rates on the three Phase 4 grading regimes alongside each other. The *Numeric* column reproduces the Phase 4 numeric pass rate from Table 10, pooled across the 258 numeric-grounded scenarios. The *Qualitative* column is the LLM-judge pass rate over the 272 qualitative-component scenarios. The *Overall* column is the unified Phase 4 pass rate over all 300 scenarios, where each scenario must pass every ground-truth check it carries (numeric, qualitative, or both); this is the combined view that takes the qualitative component into account alongside the numeric main metric.

C1. Document type.
Reject: editorials; prefaces or special-issue introductions; perspectives, commentaries, or opinions without original analysis; review articles; protocol or methods-only papers; data papers describing only a dataset; book chapters, fact sheets, summary documents.
Accept: papers with original analysis containing a methodology section and quantitative results.

C2. Data accessibility via standard API.
Required: retrievable through the platform's standard API or package (`neonutilities`, `pyaqsapi`, `pygbif`, `dataretrieval`).
Reject: pre-processed derivative datasets that cannot be reconstructed from raw queries, including pre-built fusion or gridded surfaces (satellite-AQS fusion, MAIAC AOD blends); custom harmonized datasets; aggregated regional summaries (GAGES-II, NHDPlus attributes, StreamCat, WaterWatch); model outputs consuming platform data (CMAQ, exposure surfaces); non-API upstream access (iNaturalist direct rather than the GBIF mirror); sample types not exposed by the API (NEON Biorepository samples, AOP raw imagery).

C3. External-data dependency.
Allowed (reference data): geographic boundaries, coordinate systems and projections, administrative code lookups (FIPS, ISO), regulatory thresholds (NAAQS), naming taxonomy (GBIF backbone), and platform-internal metadata.
Reject (analytical data): measurements from other monitoring networks (AmeriFlux, IMPROVE); satellite-derived predictors (MODIS, Landsat, MAIAC, PRISM, Daymet); demographic or health records (Census ACS, BRFSS, Medicare); species-trait databases or climate covariates (WorldClim, CHELSA, Elton Traits); land-cover layers (NLCD, MRLC); reservoir or dam-operation records; emissions inventories.
Operational test: if the external dataset were removed, could the main analysis still be performed meaningfully? YES means reference (accept); NO means analytical (reject).

C4. Central analytical use.
Reject when platform data is not the analytical focus: when the main contribution is an external model (HEC-RAS, SWAT, MODFLOW, HSPF, VIC, CMAQ, LSTM) calibrated against platform data; when AQS is used only as an exposure source in a health-effects study; when platform data serves as a demographic overlay in an environmental-justice study; when platform data appears only as ground-truth reference for a remote-sensing product.
Indicator: methods section dominated by model setup (grid, parameterization, boundary conditions) rather than data processing (retrieval, filtering, aggregation).
Accept: analysis *of* the platform data is the main contribution (trend analysis, flow frequency, baseflow separation, water-quality trend, occurrence pattern, air-quality trend); hybrid theory plus empirical-validation papers whose empirical portion is reproducible from platform data alone.

C5. Methodological detail.
Reject if any of the following is unspecified: *what data* (NEON DP-code, GBIF taxa, USGS site numbers plus parameter codes, EPA pollutant plus AQS parameter code); *geographic scope* (sites, region, coordinate bounds); *temporal scope* (date range); *sampling characteristics* (data type, frequency, granularity); *processing steps* (cleaning, filtering, joining, aggregation). Vague descriptions such as "we used NEON data from several sites" are insufficient.

C6. Reproducible quantitative result.
Required: at least one specific numeric result usable as a Phase-3 or Phase-4 checkpoint.
Acceptable forms: statistical value (correlation, NSE, R^2, p-value, regression coefficient, slope, Kendall τ); count (species, individuals, monitors, records before or after cleaning); summary statistic (mean, median, total, range); percentage or ratio; fitted model parameter; domain-specific quantity (NAAQS exceedance days, baseflow index, return-period estimate). Qualitative findings or figures without accompanying numbers are insufficient.

C7. Appropriate scope.
Accept all complexity tiers: single site and single parameter (Tier 1); multi-site, multi-year, single domain (Tier 2); many sites over decades or multi-pollutant filtering (Tier 3). Large data volumes (millions of GBIF records, multi-decade USGS time series) are acceptable because asynchronous download and chunking are available.
Reject only when platform-data use is trivial (a single value cited as context); when execution is clearly impossible (custom hardware, non-standard pipelines); or when platform data plays a brief reference role rather than being the analytical focus. A research question that logically implies a specific taxon, site, or region (for example "mainstem gages on the Mississippi") is acceptable because the agent can recover identifiers through platform filters.

Figure 4: Full seven-point Stage-2 rejection rubric used by the scenario-generation agent. Each criterion is applied independently; a single failure causes rejection, and the agent is instructed to reject when in doubt. C3 (external-data dependency) is the Stage-2 instantiation of the API-only-reproducibility constraint that defines the benchmark scope.

NEON
Research question: Do *Discoderus* species exhibit large episodic fluctuations in annual abundance in Sonoran Desert habitats?

Phase 1: Knowledge discovery. Find the NEON data product for pitfall trap sampling of ground beetles, and the 4-letter site code for the Santa Rita Experimental Range.
Expected: `data_products = ["DP1.10022.001"],sites = ["SRER"]`

Phase 2: API-call construction. Download ground beetle pitfall trap data for SRER from 2016 to 2022.
Expected: `neonutilities.zips_by_product(dpid="DP1.10022.001", site="SRER", startdate="2016-01", enddate="2022-12")`

Phase 3: Preprocessing. Count the total number of individual Carabidae specimens collected at SRER across 2016–2022.
Expected: `9047` individuals

Phase 4: Research-question answering. Aggregate annual Carabidae and annual *Discoderus* abundance; excluding 2016 (a mass-emergence year), compute R^2 between the two series for 2017–2022. Produce a stacked bar plot of annual carabids per trap-day broken into *D. robustus*, *D. obsidianus*, other *Discoderus* spp., and other carabids.
Numeric check: expected $R^2 = 0.91$.
Qualitative check: reference conclusion: "Annual *Discoderus* abundance is a major driver of overall carabid abundance at SRER; 2016 saw a massive episodic emergence of *D. robustus* that dwarfed all other years." Reference figure: paper Figure 4.

Figure 5: NEON example scenario.

GBIF
Research question: What is the global distribution of the redclaw crayfish (*Cherax quadricarinatus*), and how many countries has it invaded outside its native range?

Phase 1: Knowledge discovery. Identify the correct GBIF taxonomic identifier for the redclaw crayfish.
Expected: `taxon_keys = ["4648609"]`

Phase 2: API-call construction. Retrieve all global occurrence records containing geographic coordinates.
Expected: `pygbif.occurrences.search(hasCoordinate=True, taxonKey=4648609)` with pagination

Phase 3: Preprocessing. Remove records with missing or invalid coordinates, out-of-bounds locations, exact-zero coordinates, and spatial duplicates. Count remaining unique valid occurrences.
Expected: `390` records

Phase 4: Research-question answering. Count unique countries of occurrence on the cleaned dataset. Produce a global distribution map distinguishing the native range (Australia and Papua New Guinea) from the invaded range.
Numeric check: expected `29` countries.
Qualitative check: reference conclusion: "*C. quadricarinatus* occurs in at least 29 countries; the native range is restricted to Australia and Papua New Guinea, with widespread invasive presence across the Americas, Europe, Asia, and Africa." Reference figure: paper Figure 1.

Figure 6: GBIF example scenario.

EPA AQS
Research question: How have $PM_{2.5}$ mass concentrations changed in the EPA Ohio Valley climate region between 2002 and 2017 across continuously operating monitors?

Phase 1: Knowledge discovery. Find the primary 5-digit AQS parameter code for standard $PM_{2.5}$ local conditions mass measurements.
Expected: `parameter_codes = ["88101"]`

Phase 2: API-call construction. Retrieve annual summary $PM_{2.5}$ data for all monitoring sites in the Ohio Valley region (IL, IN, KY, MO, OH, TN, WV) across 2002–2017.
Expected: seven calls to `pyaqsapi.bystate.dailysummary` with `parameter="88101"`, `bdate="2002-01-01"`, `edate="2017-12-31"`, one per state FIPS code (17, 18, 21, 29, 39, 47, 54).

Phase 3: Preprocessing. Exclude records flagged for exceptional events; identify "continuous" sites with a valid annual mean for every year from 2002 to 2017. Count continuous sites.
Expected: `66` sites

Phase 4: Research-question answering. On the continuous-site subset, compute the regional average $PM_{2.5}$ for 2002 and 2017. Produce a bar chart comparing the two regional averages.
Numeric check: expected 2002 average $= 13.6\ \mu g/m^3$.
Qualitative check: reference conclusion: "Average $PM_{2.5}$ across continuously operating Ohio Valley monitors decreased by approximately 44%, from $13.6\ \mu g/m^3$ in 2002 to $7.6\ \mu g/m^3$ in 2017." Reference figure: paper Figure 4 (Ohio Valley panel).

Figure 7: EPA AQS example scenario.

USGS
Research question: How have long-term streamflow conditions and hydrological metrics changed over the last century in the Big Darby Creek Basin (Ohio)?

Phase 1: Knowledge discovery. Identify the main USGS streamgages on the mainstems of Big Darby Creek, Little Darby Creek, and Hellbranch Run with long-term daily discharge records (near Darbyville, West Jefferson, and Harrisburg).
Expected: `sites = ["03230500","03230310","03230450"]`, `parameter_codes = ["00060"]`

Phase 2: API-call construction. Download daily mean discharge for the three streamgages through the end of water year 2021.
Expected: three calls to `dataretrieval.waterdata.get_daily` with `parameter_code="00060"`, `statistic_id="00003"`, `time="1880-01-01/2021-09-30"`, one per site.

Phase 3: Preprocessing. For site 03230500, compute the average annual runoff depth (inches) over WY1922–WY2020, using the site's drainage area from USGS metadata to convert from cfs to inches per year.
Expected: `12.6` inches

Phase 4: Research-question answering. Extract the annual median daily flow at site 03230500 for WY1922–WY2021. Run a Mann-Kendall trend test and compute Sen's slope as percent change per year (slope divided by data median, times 100). Produce a scatter plot of annual median daily flow versus water year with a LOWESS overlay.
Numeric check: expected Sen's slope $= 0.8\%$ per year.
Qualitative check: reference conclusion: "Streamflow conditions at Big Darby Creek generally increased between WY1922 and WY2021; annual median daily flows show a significant positive trend." Reference figure: paper Figure 22.

Figure 8: USGS example scenario.

		Phase 1			Phase 2			Phase 3			Phase 4			All-phase		
Model	**Plat.**	Pass	Tool	Tok	Pass	Tool	Tok	Pass	Tool	Tok	Pass	Tool	Tok	Pass	Tool	Tok
	EPA	30.8	6.1	42	53.8	3.1	30	38.5	3.9	44	18.2	2.9	39	35.3	4.0	39
	GBIF	10.8	0.5	4	8.1	1.5	9	47.3	3.1	16	15.7	3.3	22	20.5	2.1	13
Gemini 2.5 Flash	NEON	17.9	3.2	21	7.1	1.5	6	10.7	12.6	458	15.4	4.8	209	12.8	5.5	173
	USGS	4.9	2.1	35	2.2	1.7	15	43.8	6.2	74	26.5	5.1	72	19.3	3.8	49
	Mean	**8.7**	**2.0**	**26**	**6.3**	**1.7**	**14**	**41.3**	**5.9**	**94**	**22.9**	**4.6**	**75**	**19.7**	**3.5**	**51**
	EPA	38.5	15.8	81	30.8	9.0	31	38.5	5.5	48	9.1	5.8	58	29.2	9.0	55
	GBIF	43.2	5.7	28	6.8	3.1	13	39.2	5.5	31	13.7	5.5	37	25.7	5.0	27
GPT-OSS-120B	NEON	35.7	10.4	47	21.4	5.4	21	3.6	14.0	206	4.0	10.6	178	16.2	10.1	113
	USGS	20.5	15.6	94	3.8	7.2	30	36.2	7.0	72	18.2	6.7	73	19.7	9.1	67
	Mean	**28.3**	**12.7**	**73**	**7.3**	**6.1**	**25**	**34.0**	**7.2**	**73**	**15.6**	**6.8**	**75**	**21.5**	**8.3**	**61**
	EPA	46.2	9.7	63	15.4	1.5	13	61.5	3.5	37	18.2	4.1	66	35.3	4.7	45
	GBIF	67.6	4.0	24	10.8	0.5	5	54.0	4.3	27	25.5	5.0	39	39.5	3.4	24
Gemini 2.5 Pro	NEON	53.6	6.0	48	3.6	1.9	8	28.6	13.5	438	15.4	17.2	627	25.3	9.6	280
	USGS	35.7	7.0	73	0.0	0.7	6	58.4	6.7	94	42.9	7.5	130	34.3	5.5	76
	Mean	**45.7**	**6.3**	**58**	**3.7**	**0.8**	**6**	**54.7**	**6.6**	**107**	**35.7**	**7.9**	**160**	**34.9**	**5.3**	**80**
	EPA	46.2	7.9	45	53.8	4.2	22	53.8	5.1	48	18.2	4.6	50	43.0	5.4	41
	GBIF	78.4	3.9	16	29.7	3.0	14	51.3	3.5	17	23.5	2.7	15	45.7	3.3	16
Gemma 4-31B	NEON	14.3	6.1	52	28.6	2.0	6	28.6	12.9	241	23.1	10.1	248	23.6	7.8	137
	USGS	29.2	13.4	120	0.5	1.8	9	55.7	6.7	96	46.5	6.6	99	33.0	7.1	81
	Mean	**40.7**	**10.2**	**85**	**12.7**	**2.2**	**11**	**52.0**	**6.4**	**88**	**38.4**	**6.1**	**96**	**35.8**	**6.2**	**69**
	EPA	23.1	15.5	110	46.2	5.0	46	53.8	9.3	137	36.4	12.2	202	39.9	10.5	124
	GBIF	64.9	6.6	49	24.3	8.9	82	54.0	5.9	38	23.5	9.4	85	41.7	7.7	64
Qwen 3.6-27B	NEON	46.4	13.0	177	21.4	5.0	34	28.6	17.4	483	38.5	19.8	504	33.7	13.8	299
	USGS	34.6	17.4	224	3.8	6.3	51	54.6	7.5	110	48.2	10.3	175	35.3	10.4	140
	Mean	**42.7**	**14.3**	**172**	**12.3**	**6.8**	**57**	**52.0**	**8.1**	**128**	**41.9**	**11.2**	**191**	**37.0**	**10.1**	**135**
	EPA	23.1	15.8	121	23.1	12.5	154	53.8	8.5	109	18.2	7.5	101	29.5	11.1	121
	GBIF	68.9	7.2	58	47.3	5.2	65	51.3	8.0	49	25.5	6.5	41	48.3	6.7	53
Gemini 3 Flash	NEON	46.4	6.7	63	14.3	3.1	13	39.3	10.1	205	26.9	13.9	356	31.7	8.4	159
	USGS	62.7	11.9	121	1.6	6.5	58	59.5	7.0	93	53.5	8.3	125	44.3	8.4	99
	Mean	**61.0**	**10.4**	**100**	**15.0**	**6.1**	**60**	**55.3**	**7.6**	**93**	**43.8**	**8.5**	**131**	**43.8**	**8.1**	**95**
	EPA	46.2	9.0	54	23.1	10.1	130	61.5	16.9	266	27.3	10.2	167	39.5	11.5	154
	GBIF	90.5	4.0	17	47.3	2.9	28	53.4	8.1	82	28.0	7.6	141	54.8	5.7	67
Gemini 3.1 Pro	NEON	82.1	4.6	42	21.4	5.0	31	39.3	19.7	503	34.6	17.6	448	44.4	11.7	256
	USGS	65.9	11.6	117	1.1	4.3	30	65.2	9.9	169	59.2	10.9	231	47.9	9.2	137
	Mean	**72.7**	**9.0**	**82**	**15.3**	**4.3**	**34**	**59.7**	**10.7**	**183**	**49.2**	**10.9**	**233**	**49.2**	**8.6**	**129**
	EPA	100.0	–	–	76.9	–	–	84.6	–	–	45.5	–	–	76.7	–	–
	GBIF	94.6	–	–	100.0	–	–	76.7	–	–	66.0	–	–	84.3	–	–
Human†	NEON	96.4	–	–	92.9	–	–	78.6	–	–	57.7	–	–	81.4	–	–
	USGS	87.0	–	–	93.5	–	–	85.9	–	–	72.8	–	–	84.8	–	–
	Mean	***90.3***	–	–	***94.3***	–	–	***82.9***	–	–	***68.8***	–	–	***84.7***	–	–

Table 10: Per-phase per-platform breakdown of pass rate (%), mean tool calls per scenario, and mean tokens per scenario (thousands) for all seven baseline models and the human evaluation (marked with †). *Pass* is pass rate, *Tool* is mean tool invocations, *Tok* is mean input plus output tokens in thousands. Models are ordered by overall pass rate from lowest to highest, with the human evaluation at the bottom for visual emphasis as the calibrated upper anchor. Within each model block, the **Mean** row is a weighted mean across the four platforms by scenario count (EPA 13, NEON 28, GBIF 74, USGS 185); the All-phase **Mean** cell is the global weighted mean over all 16 (platform, phase) cells. Tool-call and token data are unavailable for the human evaluation.

	Phase 4 pass rate (%)		
Model	**Num.**	**Qual.**	**Overall**
Qwen 3.6-27B	41.9	48.2	32.0
Gemini 3 Flash	43.8	53.3	34.3
Gemini 3.1 Pro	49.2	57.4	39.9

Table 11: Phase 4 pass rate (%) under three grading regimes. *Num.* is the numeric pass rate over the 258 numeric-grounded scenarios (reproduced from Table 10). *Qual.* is the LLM-judge pass rate over the 272 qualitative-component scenarios. *Overall* is the unified Phase 4 pass rate over all 300 scenarios, requiring each scenario to pass every ground-truth check it carries. The qualitative view runs several percentage points above the numeric view, indicating that "approximately right" conclusions outpace exactly correct numbers; the overall pass rate, which holds models accountable to both signals jointly where both are available, sits below either single view.