Latent Mining for Verifiable Scientific Task Generation

Jarrod Barnes[1]

Abstract

Progress in scientific agents depends on converting ambiguous research workflows into decisions with inspectable evidence and checkable outcomes. We introduce Latent Mining, a data-generation and benchmark-construction method that turns public model disagreement into verifier-backed scientific-agent tasks. Latent Mining mines proxy-attractive decisions, admits tasks through a model-backed hidden verifier after leakage and shortcut audits, and preserves replayable trajectories for process diagnosis. Each generated task must name the workflow decision, public evidence, hidden verifier, failure label, and replayable trajectory. We instantiate the method in regulatory genomics, where terminal agents choose among candidate noncoding DNA edits for specified cell contexts and regulatory readouts using release-safe public evidence from genomic foundation models, motif evidence, chromatin context, and locus metadata. The released benchmark contains 165 tasks across 11 loci and 4 regulatory readout families. Across 6 complete frontier-agent evaluations, all model-task runs are public-valid and grading-valid, the aggregate strict solve rate is 11.3%, and no model solves more than 15% of tasks, showing that the generated tasks remain unsaturated under strict hidden verification. Trace diagnostics reveal a scientifically meaningful failure mode. Agents often inspect relevant evidence but over-trust attractive public signals, weakly compare plausible alternatives, or drift from the requested biological readout. Workflow scaffolding improves process more than hidden outcomes, separating ordinary planning failures from a persistent public-prior/verifier gap. Latent Mining provides replayable verifier-backed data for evaluating scientific agents and studying how they act under public-proxy uncertainty.

🤗 **Benchmark dataset**

1 Introduction

Scientific-agent benchmarks increasingly measure retrieval, tool use, pipeline execution, and locally checkable research tasks. Regulatory genomics exposes a different decision problem, where an agent must choose an intervention from incomplete public evidence and the outcome is resolved only by a later assay, simulator, expensive computation, or experiment.

Regulatory-intervention triage makes this concrete. A noncoding edit can look promising because it changes a genomic foundation-model score, disrupts a motif, shifts an embedding, sits in an enhancer, or overlaps a cell-type annotation. These signals matter, but the right intervention still depends on the requested molecular readout, the cell or tissue context, the locus, and whether public evidence is relevant to promoter activity (CAGE), chromatin accessibility (ATAC), protein-DNA binding (ChIP), or transcript output (RNA-seq). A generic sequence-model outlier is not automatically the best intervention.

We introduce Latent Mining, a benchmark-construction method for scientific decisions under public-proxy uncertainty (Figure 1). Latent Mining works backward from one design test. Can we name the lab workflow decision, public evidence, hidden verifier, failure label, and replayable trajectory? A task is admitted only when the public evidence surface gives a domain agent enough to reason and the hidden verifier separates the objective from plausible public-proxy alternatives. Public evidence becomes the decision surface while the hidden objective remains withheld.

The first release instantiates this method as a regulatory-genomics benchmark for terminal agents. Agents choose among candidate noncoding edits using release-safe evidence, including genomic-model priors, motif and position-weight-matrix changes, embedding and mutagenesis features, chromatin and locus metadata, and disagreement across public signals. A hidden AlphaGenome-backed regulatory objective [1] admits and grades the task. The claim is about hidden-verifier-backed regulatory triage, and selected edits still need experimental validation.

Strict hidden verification yields an aggregate solve rate of 11.3%, and no evaluated frontier agent solves more than 15% of the benchmark. The released biology matrix contains 165 tasks and 6 complete model evaluations. All model-task runs pass the public interface and grading interface, with no missing answers, runner exits, timeouts, or malformed submissions.

[1] Dynamical Systems. Correspondence to: Jarrod Barnes <jarrod@dynamicalsystems.ai>. Preprint.

Trace labels show where valid non-solves fail, with concentrations in public-proxy collapse, single-channel overweighting, assay-context mismatch, weak rejected-alternative contrast, and commitment-control failures. Agents often identify assay-relevant evidence, then lose it during the final contrastive choice. In the paired expert-method rescue diagnostic, a hidden-blind workflow checklist improves assay orientation, proxy-decoy control, evidence-channel inspection, contrast, and rejected-option count, while strict solves move only modestly. The split separates workflow-planning gaps from the harder public-prior/verifier gap.

We make four contributions. We define Latent Mining as a method for constructing verifier-backed scientific intervention tasks from incomplete public evidence, and we release a 165-task assay-aware regulatory-genomics benchmark for noncoding intervention triage. We report a 6-model frontier-agent calibration matrix with an 11.3% aggregate strict solve rate and no model above 15%, and we use trace diagnostics and a paired expert-method rescue intervention to localize failures that pass/fail scoring hides.

2 Related Work

2.1 Scientific-agent evaluation in biology

Biology-agent evaluation has moved from question answering toward work that resembles scientific practice. GeneBench evaluates multi-stage inference in genomics and quantitative biology [2], and scBench evaluates realistic single-cell RNA-seq workflows [3]. LAB-Bench, LABBench2, ScienceAgentBench, SpatialBench-Long, and PromptBio-Bench extend the same direction across laboratory reasoning, data-driven discovery, tool use, spatial reasoning, and end-to-end bioinformatics workflows [4–8]. The shared move is to evaluate chains of scientific judgment under visible evidence, where a model can retrieve the right concept, inspect the right file, or execute plausible code and still fail because it chooses the wrong assay context, quality-control decision, estimator, or final action.

Latent Mining fits this family at the intervention-triage boundary. Its task object is a choice among candidate scientific actions rather than a completed analysis artifact. In the biology release, the action is a noncoding edit selected for a promoter-activity, chromatin-accessibility, protein-DNA-binding, or transcript-output objective in a cell or tissue context. The benchmark asks whether an agent can identify the requested readout, compare imperfect evidence channels, reject attractive decoys, and commit to a choice that survives withheld verification. That places the evaluation at the point where public biological priors are useful enough to guide search and incomplete enough to mislead it.

The public evidence stack comes from regulatory genomics, where motif changes, position-weight-matrix shifts, chromatin annotations, sequence-model priors, embedding changes, and locus context all supply legitimate evidence but none of them is a general answer rule for any of those readouts. Evo 2 [9], Carbon [10], ENCODE-derived annotations [11, 12], GTEx-style context [13], and motif resources [14, 15] enter the benchmark as parts of this evidence stack. Public priors shape the agent-visible decision surface, while the AlphaGenome-backed objective supplies the hidden task-construction verifier. That separation is the benchmark object.

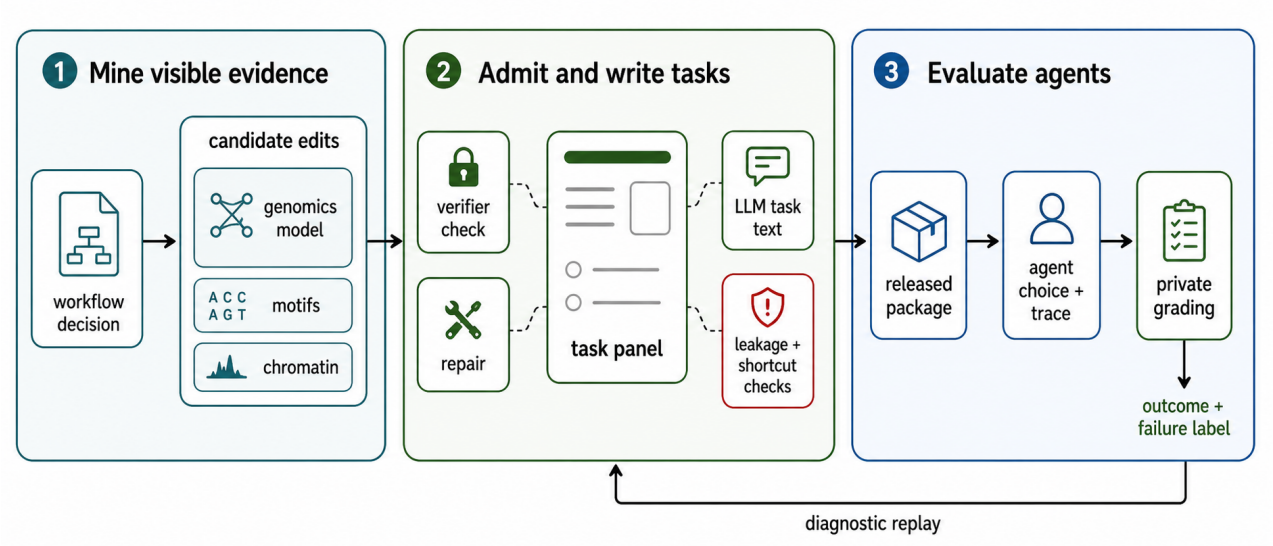

Figure 1: Latent Mining loop. The method mines visible evidence from a workflow decision, admits and writes tasks through hidden-verifier checks and public shortcut repair, then evaluates agent choices and traces under private grading. The withheld outcome and failure label support diagnostic replay without exposing hidden verifier values.

2.2 Benchmark design and trajectory evidence

Benchmark quality is an evidence-design problem. Agentic benchmark checklists and evidence-centered benchmark design [16, 17] ask researchers to specify intended use, capability construct, content source, adaptation policy, assembly procedure, and evidence that a score supports the intended claim. This standard is especially important for scientific-agent benchmarks, where the task must preserve a real judgment call without becoming ungradeable. A raw accuracy number can hide runner failures, invalid answers, leakage, shortcuts, or mismatch between the task surface and the claimed capability.

Latent Mining adopts that validity discipline as a construction constraint. Validators enforce public-interface and grading-interface validity, and task admission rejects panels solved by one public scalar, one leaked field, or an obvious metadata artifact. The reported score is one part of the evidence package. Validity gates make attempts comparable, while trace labels and rescue runs make low solve rates interpretable.

ACT*ONOMY and related trajectory analyses [18] supply the second half of this design. For agents, failure is often a path property because a model can look at a file without using it, name a decoy without controlling final commitment, or acknowledge assay context and still overweight a public scalar. Latent Mining turns those path properties into biology-specific process labels for assay orientation, evidence table construction, proxy-decoy control, contrastive triage, commitment control, and valid finalization. The labels do not replace the hidden outcome; they explain how a public-valid attempt reached it.

2.3 Expert-method diagnostics

Drug-discovery agent evaluation has used unaided performance, failure steps, exploration differences, shared wrong answers, and answer-blind expert-method reruns to distinguish workflow-planning failures from target-solving failures [19].

Latent Mining uses this form to ask a domain-specific question. If recovery is high, the benchmark is mainly exposing workflow-organization failures. If process improves but strict solves move little, the bottleneck is not only planning, and public priors remain incomplete or misleading relative to the hidden verifier. This makes rescue a diagnostic over failure source.

2.4 Terminal-agent evaluation infrastructure

Terminal-agent and coding-agent benchmarks provide the execution conventions that make this paper's evidence auditable. SWE-bench [20], Terminal-Bench [21], FeatureBench [22], SciCode [23], and related repository or shell-based benchmarks evaluate agents in environments where success depends on reading files, running commands, producing artifacts, and passing hidden or public validators. Fixed task bundles, executable runners, traces, hidden graders, and interface validity checks give an agent benchmark a reproducible surface.

Latent Mining reuses those conventions for scientific intervention decisions. The benchmark gives the agent a terminal environment with public files and answer schemas, then withholds the verifier-backed objective. This is closer to a scientist opening a dossier than to a short-answer biology exam. The terminal surface matters because regulatory-triage failures often occur after several valid actions. The agent can inspect a motif table, compare model priors, cite chromatin context, and still choose the wrong edit. Capturing the path makes that failure visible.

Together, these lines of work locate Latent Mining as a construction method for workflow-grounded intervention choices where public evidence creates plausible alternatives and a hidden verifier resolves the decision. The biology benchmark is the first completed instantiation. Its value comes from putting validity gates, regulatory evidence, hidden verification, trace labels, and expert-method rescue into one evaluation object.

3 Methods

3.1 Latent Mining design

Latent Mining constructs scientific-agent tasks by working backward from a replayable scientific decision. We first name the workflow decision, then assemble public evidence that could plausibly guide that decision, mine public model or assay-derived signals for candidate choices, and use a withheld verifier to admit only panels where the hidden objective is separated from attractive public alternatives. After leakage and shortcut audits, the accepted task is packaged as a terminal episode with public files, a constrained answer schema, and a trace that can be replayed for process analysis. Figure 1 shows this loop, with the agent-visible evidence surface separated from the withheld verifier.

Each task is defined by a fixed design tuple that names the lab workflow decision, the public evidence surface, the hidden verifier, the failure label space, and the replayable trajectory captured during agent execution. As the construction unit for the benchmark, the tuple prevents task generation from collapsing into generic question answering because every item must connect an agent-visible decision to a private objective and to interpretable failure evidence.

The benchmark is intended for a narrow capability claim. It evaluates whether terminal agents can make assay-aware, locus-grounded, contrastive regulatory-intervention choices under public-proxy uncertainty, not whether they possess broad biological knowledge. The task surface fixes the public files, candidate panels, answer schema, runner policy, and hidden-grader boundary. The evidence package combines strict hidden outcomes, public-validator status, trace-level process labels, and a paired expert-method rescue diagnostic.

3.2 Regulatory-genomics task construction

The biology release instantiates Latent Mining as regulatory-intervention triage. Each task asks an agent to choose one candidate noncoding edit for a specified assay and cell or tissue context. The benchmark covers 165 tasks across 11 loci, 9 chromosomes, 11 biology systems, and 4 assay families. Each locus maps to one biology system, so the locus and system axes are aligned rather than independent. An earlier APOE pilot locus is excluded from the released corpus. The four readout families are chromatin immunoprecipitation (ChIP) profiles for protein-DNA or regulatory-element binding, assay for transposase-accessible chromatin (ATAC) for chromatin accessibility, cap analysis of gene expression (CAGE) for promoter output, and RNA sequencing (RNA-seq) for transcript output. The task set includes 60 ChIP tasks, 45 RNA-seq tasks, 30 ATAC tasks, and 30 CAGE tasks.

Each task panel contains plausible candidate edits rather than random distractors, built to force comparison among public evidence channels. An admitted panel contains at least one candidate supported by salient public evidence, one or more public-scalar or motif-attractive decoys, and alternatives that differ in edit geometry, sequence context, local regulatory annotation, or assay relevance. This construction makes the final decision a regulatory-triage problem rather than a lookup from one feature column.

The benchmark uses real regulatory loci and assay contexts as workflow anchors. The loci span cholesterol metabolism, autoimmunity, hematopoiesis, hemoglobin switching, hemoglobinopathy, obesity and metabolism, cancer proliferation, cystic fibrosis and epithelial transport, cancer and aging, diabetes and endocrine regulation, and lactase regulation. These loci are used as benchmark contexts, not as claims about clinical actionability or experimental tractability.

3.3 Public evidence surface

The public task surface contains release-safe evidence that a domain scientist could inspect before selecting an intervention. Public features include genomic foundation-model priors from Evo 2 [9] and Carbon [10], embedding-distance features, motif and position-weight-matrix changes, edit geometry, locus metadata, assay family, chromatin context, mutagenesis-derived features, composition shifts, and disagreement across public model signals. The public evidence gives biological traction without exposing a one-feature answer rule. A task is rejected if a single public score, source model, motif count, or generic mechanism story solves it.

The assay readout controls how evidence should be interpreted. ATAC is a chromatin-accessibility readout [24], not an expression readout. ChIP asks about transcription-factor or chromatin regulatory enrichment [25], not direct gene function. CAGE measures promoter or transcription-start-site output [26, 27]. RNA-seq measures transcript output [28], which can depend on distal enhancer-promoter wiring [29] that local sequence and motif evidence may not expose. The task surface rewards agents that connect evidence to the requested readout rather than treating all regulatory objectives as one sequence-model ranking problem.

Public evidence is also the source of hard negatives. A candidate can look strong because it maximizes a Carbon perturbation, disrupts the largest number of motifs, sits near a promoter, or has a large embedding shift. Those signals are useful priors, but they can be wrong under the hidden assay-specific objective. The hidden verifier tests cases where public priors guide search but fail as answer rules.

3.4 Hidden verifier and admission criteria

Hidden verification uses an AlphaGenome-backed regulatory objective [1] for the requested assay and context. The verifier supports task construction and grading under a fixed private objective. It is not treated as wet-lab truth, a universal biological oracle, or evidence that any selected edit is experimentally validated. The paper's claim is about hidden-verifier-backed regulatory triage under one verifier regime.

Hidden verifier values, ranks, thresholds, cache identifiers, target identifiers, and known solutions are withheld from the public task surface. The public bundle states the verifier family and the public evidence available to the agent, but it does not expose the private score used to grade a final choice. The reported outcome tables expose class labels stripped of private values, while exact solve and near-miss labels require the private grader. Task admission requires private separation from plausible wrong answers and public shortcut resistance. A panel that can be solved by one public scalar, one action type, one leaked field, or one obvious metadata artifact is rejected or repaired.

The hidden grader assigns each completed attempt to one of three outcome classes. The solve threshold is fixed during task construction, before any agent evaluation. A solve clears the private objective threshold. A valid below-threshold near-miss is outcome-proximal under the hidden verifier but does not clear the solve threshold. A scientific-search failure is a valid submitted answer that does not reach the near-miss region. Near-miss is an outcome label, not partial credit and not a process-quality claim.

3.5 Terminal-agent environment and validation

Tasks are packaged as terminal-agent environments with public task files, candidate files, public evidence guides, answer schemas, and public validators. Agents must produce a valid answer file that selects one candidate and supplies a rationale. The public validator checks answer presence, schema validity, and public-interface constraints without revealing hidden verifier scores, ranks, thresholds, target IDs, cache IDs, known solutions, or option IDs.

The main biology calibration matrix contains six frontier models evaluated over 165 tasks. Every model-task run in

this matrix is public-interface valid and grading-interface valid. The ranked matrix has no timeouts, nonzero exits, duplicate task-agent pairs, or missing task-agent pairs, which rules out these failure modes, along with interface-invalid answers, as explanations for the low solve rates.

The evaluated models are Codex GPT-5.5, Kimi K2.6, GLM 5.1, Claude Opus 4.7, Claude Sonnet 4.6, and Gemini 3.1 Pro. The Kimi K2.6 row is an effective completed row containing 121 Moonshot API runs and 44 Hugging Face Router fallback fills. The paper reports provider-level outcomes and reads provider process contrasts as bounded comparisons, because runner affordances and trace depth differ across providers. Appendix G reports the per-provider execution protocol.

3.6 Process diagnostics

Outcome labels answer whether an agent selected the hidden-verified objective, while process diagnostics ask why non-solves occurred. The diagnostic layer labels near-misses, failures, exploration patterns, and final-answer rationales without exposing private verifier data. Labels are derived from public answer rationales, sanitized trace spans, deterministic trace features, and quote-backed manual or semi-automated review. Private scores, private ranks, thresholds, private numeric diagnostics, target identifiers, cache identifiers, known solutions, option identifiers, and evaluator internals are excluded from process artifacts.

The biology process taxonomy is adapted to regulatory triage. Assay orientation records whether the agent identifies the requested regulatory readout before ranking candidates, such as promoter output for CAGE, chromatin accessibility for ATAC, protein-DNA binding for ChIP, or transcript output for RNA-seq. Evidence table construction records whether the agent inspects and compares public channels. Proxy-decoy control records whether it names the strongest public scalar, motif, or source-model decoy. Contrastive triage records whether it compares plausible alternatives rather than only justifying the selected edit. Commitment control records whether the final choice follows the contrast or drifts toward a proxy. Finalization records whether the agent produces a valid public answer without hidden leakage.

Process labels are diagnostic evidence rather than hidden partial credit. They distinguish good biological near-misses from proxy-attracted near-misses, public-proxy overfit, assay-context failures, contrast failures, and commitment-control failures. This separation matters because outcome proximity and scientific process can diverge. A near-miss can be biologically thoughtful, proxy-driven, or lucky under the hidden verifier.

3.7 Expert-method rescue diagnostic

The expert-method rescue experiment is a paired diagnostic intervention, not a second benchmark setting, and it does not change the main leaderboard. The intervention asks whether failed tasks recover when the agent receives a general public workflow checklist, or whether agents still choose proxy-attractive candidates that fail hidden verification.

For each paired cell, the baseline replay arm uses the standard task surface, standard prompt, and public files. The expert-method rescue arm uses the same task and public files with a hidden-blind workflow checklist appended. The rescue protocol limits the checklist to general public workflow guidance describing how to orient to the assay, compare public evidence channels, identify decoys, contrast alternatives, and avoid scalar collapse. It does not include the hidden answer, hidden rank, hidden score, target identifier, hidden option identifier, candidate-specific hint, private cache identifier, or designer rationale. A leakage scan checks rescue prompts, raw outputs, answer files, sanitized traces, public-validator outputs, and summaries before public reporting.

After excluding invalid retry residue, the completed paired core requires both baseline replay and rescue runs to be present and public-valid for each model-task unit. The rescue analysis reports arm-specific outcome labels and process deltas, including assay orientation, proxy-decoy control, contrast, evidence-channel count, and rejected-option count.

Rescue outcomes are reported with distinct terminology. Baseline solves are solves in the baseline-replay arm, and method-conditioned solves are solves in the rescue arm. Rescued solves are the baseline non-solves that become solves under rescue, a subset of the method-conditioned solves. The rescue delta is the change in a measure between the two arms. A method-conditioned solve is not counted as an ordinary unaided benchmark solve. The main benchmark claim remains the unaided baseline calibration. Rescue tests whether a hidden-blind workflow checklist changes outcome labels and process labels relative to baseline replay.

4 Results

4.1 The released matrix is valid and difficult

The released regulatory-genomics matrix contains six frontier models evaluated over 165 tasks. Every model-task run passes the public interface and the grading interface. The matrix has no timeouts, no nonzero exits, no duplicate task-agent pairs, and no missing task-agent pairs. These checks do not make the hidden objective easy to interpret, but they rule out the most direct runner and packaging explanations for the low solve rate.

Strict hidden verification yields 112 solves, 252 valid below-threshold near-misses, and 626 scientific-search failures, for an aggregate solve rate of 11.3%. No evaluated frontier agent solves more than 15% of the benchmark (Figure 3). Kimi K2.6 has the highest strict solve rate at 14.5%, followed by Codex GPT-5.5 at 13.3% and Gemini 3.1 Pro at 11.5%. Because the Kimi row is an effective completed row with mixed Moonshot API and Hugging Face Router

provenance, provider-level outcome comparisons are descriptive, and the provider process contrasts reported later remain bounded by trace depth rather than fine-grained runner comparisons.

All assay families remain low-solve under strict hidden verification, with a visible gradient by regulatory readout. Chromatin-accessibility triage is the least difficult slice in this verifier regime, with 36 solves in 180 attempts, a 20.0% solve rate. Binding and regulatory-element triage reaches 45 solves in 360 attempts, a 12.5% solve rate, and transcript-output triage 21 solves in 270 attempts, a 7.8% solve rate. Promoter-output triage has the lowest strict solve rate, with 10 solves in 180 attempts and the largest near-miss concentration, 77 valid below-threshold near-misses.

These family-level differences are descriptive slices of the same intervention-triage mechanic rather than separate leaderboards, and they motivate the atlas view in Figure 2. The benchmark keeps task coverage narrow enough to audit, while still forcing agents to change how they read public evidence across promoter output, chromatin accessibility, binding or regulatory-element signals, and transcript output.

4.2 Process diagnostics identify proxy-driven non-solves

The behavior taxonomy turns valid non-solves into interpretable evidence without replacing hidden grading or treating near-misses as partial credit (Appendix D). Across 252 valid below-threshold near-misses, 153 are mixed proxy-attracted near-misses, 55 are public-proxy-overfit near-misses, and 32 are good scientific near-misses, with the remaining 12 uncertain or outcome-near process-poor cases. Public scalar collapse is the dominant near-miss proxy label, appearing in 189 of 252 near-miss rows, followed by single-channel overweighting in 38 rows and assay-context mismatch in 18 rows.

The same behavior appears at the non-solve failure-step level. Among 878 non-solves, proxy-collapse failure accounts for 640 rows, insufficient evidence for 140, commitment-control failure for 57, assay-context failure for 26, and contrast failure for 15. These labels come from public rationales, sanitized traces, deterministic trace features, and quote-backed review. They show that agents often find usable public evidence, but final choices still concentrate around attractive public proxies.

Exploration volume alone does not explain the benchmark. The exploration audit labels 866 runs as narrow but valid, 86 as broad and controlled, 30 as uncertain, and 8 as broad but uncontrolled. The typical failure is not a blank or malformed run; many traces inspect multiple channels, name assay-relevant evidence, and still fail at the step where public evidence must be turned into a contrastive commitment against plausible decoys. This is the regulatory-genomics version of a notice-act gap, with proxy control, assay context, and final commitment replacing generic workflow completion as the behavior under test.

4.3 How providers reason and act

The gap between stating the proxy-rejection principle and acting on it holds across every provider (Figure 4). Across the matrix, agents express proxy uncertainty and contrast rejected alternatives in most attempts, while every provider still collapses to a salient public signal at the commitment step. The providers separate in how they reach that collapse and in how often they avoid it.

Outcomes spread over the 165 tasks, with Kimi K2.6 and Codex GPT-5.5 leading at 14.5 and 13.3% strict solves, Gemini 3.1 Pro at 11.5%, and Claude Opus 4.7, GLM 5.1, and Claude Sonnet 4.6 trailing at 9.7, 9.7, and 9.1%. Near-miss volume does not track solve rate. Codex produces the most valid below-threshold near-misses at 55, far more than Gemini's 31, so Codex reaches the outcome-proximal region often and converts little of it into solves.

Gemini is the behavioral outlier. Its non-solves are the least proxy-driven of the six, with proxy-collapse at 42% against 87% for Opus and GLM. Instead, Gemini shows the most insufficient-evidence failures at 34% and the most assay-context failures at 15%, and its near-misses carry the most assay-context mismatch at 39%. Gemini more often stops short on evidence or drifts from the requested readout, while the others gather public evidence and then over-trust a salient scalar.

Near-miss quality also splits by provider. The Claude family carries the largest share of near-misses labeled good scientific, half of Sonnet's and a quarter of Opus's, while Codex, Kimi, and GLM near-misses are dominated by proxy attraction, Codex at 95% mixed proxy-attracted. We read this as a labeling contrast rather than a competence ranking. The Claude entries are compact adjudicated answer summaries while the tool-native entries are long heuristically coded traces, and trace depth ranges from about two turns for Claude to between 28 and 68 turns for the tool-native providers, so the deterministic process features saturate unevenly. We therefore restrict provider claims to outcome counts and adjudicated labels and avoid keyword-rate comparisons.

Exploration breadth varies in the same bounded way. Opus 4.7 has the highest rate of broad controlled exploration at 27%, while GLM 5.1 and Kimi K2.6 explore most narrowly, at 99 and 96% narrow-but-valid. Broad exploration does not buy solves on its own, since the two highest solve rates come from providers that explore narrowly, which matches the aggregate finding that exploration volume does not explain the benchmark.

The task-family gradient holds across the matrix. Models solve chromatin accessibility most and promoter output least, and the family they solve least is the one where the salient public scalar sits farthest from the readout (Table 1). Provider identity changes how often an agent reaches the commitment step with usable evidence, while task family

Table 1: Behavioral signatures by task family. Process labels decomposed across the four assay families under the fixed hidden verifier. *Solve* is the strict solve rate; *states rule* is the proxy-uncertainty-expressed rate, which stays near-constant while solves fall as the readout decouples from salient public signals. *Scalar*, *1-channel*, and *assay ctx.* are the public-scalar-collapse, single-channel-overweighting, and assay-context-mismatch shares of each family's near-misses. All entries are release-safe class labels, not hidden scores.

Family (assay)	Solve	States rule	Scalar	1-channel	Assay ctx.
Chromatin-accessibility (ATAC)	20.0%	83%	89%	8%	3%
Binding / regulatory-element (ChIP)	12.5%	80%	67%	23%	6%
Transcript-output (RNA-seq)	7.8%	81%	72%	12%	12%
Promoter-output (CAGE)	5.6%	83%	79%	12%	6%

changes how badly the public signal misleads it once there. These are diagnostic process labels bounded by the trace-depth differences above, not psychometric measurements or a capability ranking.

4.4 Expert-method rescue improves process more than hidden outcomes

The expert-method rescue experiment tests whether a hidden-blind workflow checklist changes the paired result. The 44 rescue tasks are a diagnostic subset, not a random sample. They are stratified across the four workflow families and all 11 loci, and they oversample near-miss-dense and all-model-fail tasks because those cases most sharply separate planning gaps from public-prior gaps. The baseline-replay solve rate over this subset is lower than the unaided leaderboard and is not comparable to it.

After excluding invalid retry residue, the completed core contains 132 paired model-task units across Codex GPT-5.5, Gemini 3.1 Pro, and GLM 5.1. Both arms are present and public-valid for every paired unit. Baseline replay produces 4 solves, 76 valid below-threshold near-misses, and 52 scientific-search failures. Expert-method rescue produces 6 method-conditioned solves, 66 valid below-threshold near-misses, and 60 scientific-search failures (Table 2).

The checklist changes observable workflow behavior much more than strict solves, which move only from 4 to 6 while every process variable rises sharply (Table 2). The behavioral lift matches the taxonomy's failure modes, because the checklist makes agents orient to the assay, build a wider evidence table, name public decoys, and contrast alternatives before committing.

The paired transitions make the interpretation asymmetric. Rescue converts 5 baseline non-solves into method-conditioned solves, with 2 scientific-search failures and 3 near-misses becoming solves. It also creates 3 checklist regressions, where baseline solves become rescue non-solves.

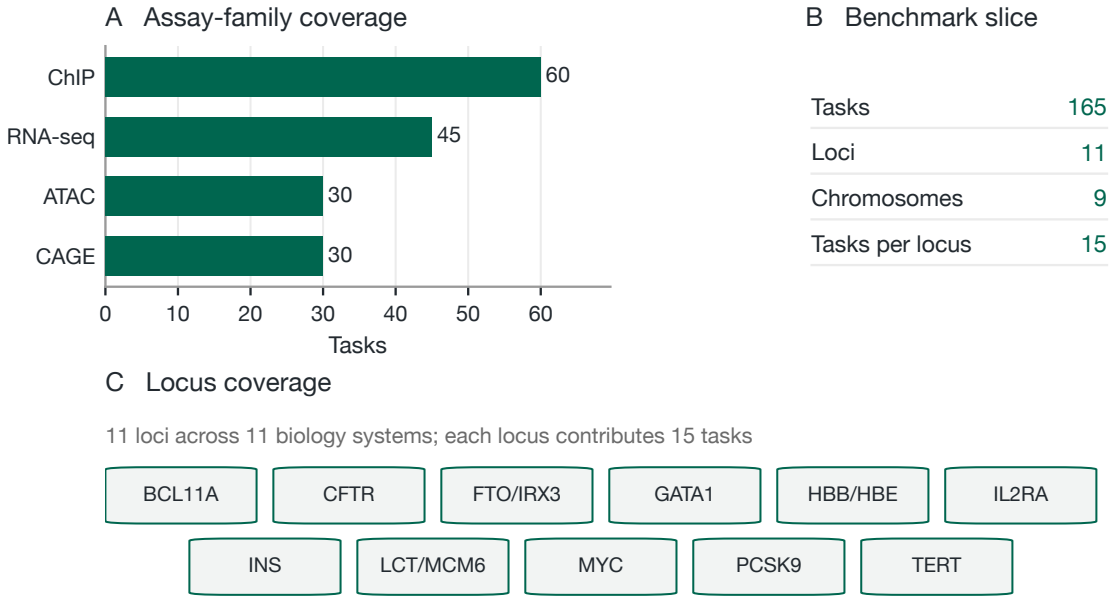

Figure 2: Regulatory-genomics task atlas. The release contains one terminal-agent mechanic, regulatory-intervention triage, stratified across 165 tasks, 11 loci, 11 biology systems, and 4 assay families. This figure describes benchmark coverage rather than performance or biological tractability.

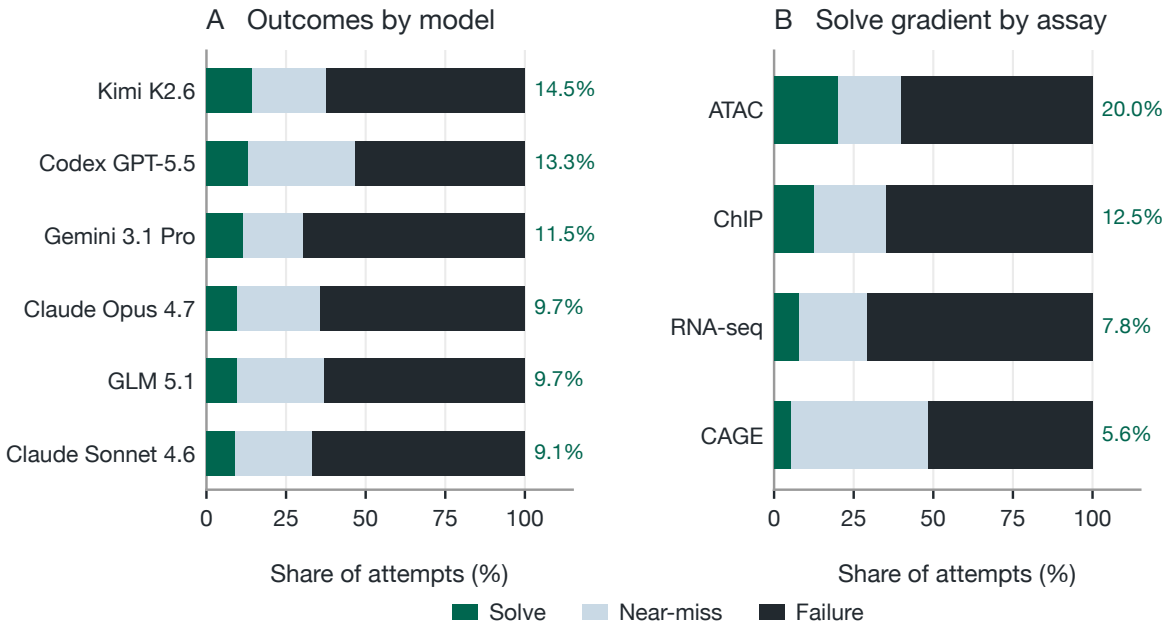

Figure 3: Outcome structure across models and assay families. Panel A reports solve, near-miss, and scientific-search failure shares by model. Panel B reports the same outcome classes by assay family. Near-misses are valid below-threshold outcomes, separate from solves, and assay-family differences are descriptive slices under the same hidden verifier rather than separate leaderboards.

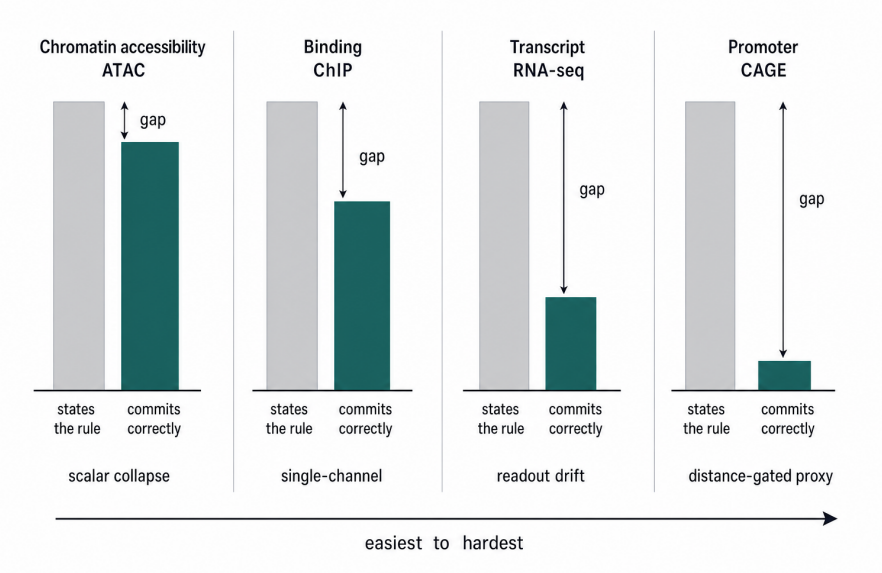

Figure 4: Stated rule versus committed answer. For each assay family, agents state the proxy-rejection principle, the grey bar, far more often than they commit to the hidden-verified choice, the green bar, and the gap widens as the requested readout decouples from the most salient public signals. The green bar marks a hidden-verified solve, not a wet-lab outcome. The pattern holds within every provider. The bars are schematic; verified per-family rates appear in Table 1.

Among the 44 rescue tasks, all 3 core providers choose the same wrong answer on 12 baseline-replay tasks but only 5 rescue tasks, while at least 2 providers still choose the same wrong answer on 35 baseline-replay tasks and 34 rescue tasks. The checklist reduces full three-model convergence on the same decoy and improves trace behavior, while public-prior attraction persists under hidden verification.

Table 2: Expert-method rescue paired core. Rescue is a paired diagnostic separate from the unaided leaderboard. Outcome rows show hidden-verifier classes, and process rows show public trace behavior over the same 132 paired model-task units. Method-conditioned solves are excluded from ordinary unaided benchmark solves.

Measure	Baseline replay	Expert-method rescue
Paired model-task units	132	132
Hidden outcome labels		
Solves	4	6
Valid below-threshold near-misses	76	66
Scientific-search failures	52	60
Public process behavior labels		
Assay orientation present	98/132	127/132
Proxy-decoy control present	94/132	132/132
Rejected-alternative contrast present	124/132	132/132
Mean evidence channels used	4.88	7.30
Mean rejected options	2.73	3.91
Broad and controlled exploration	54/132	122/132

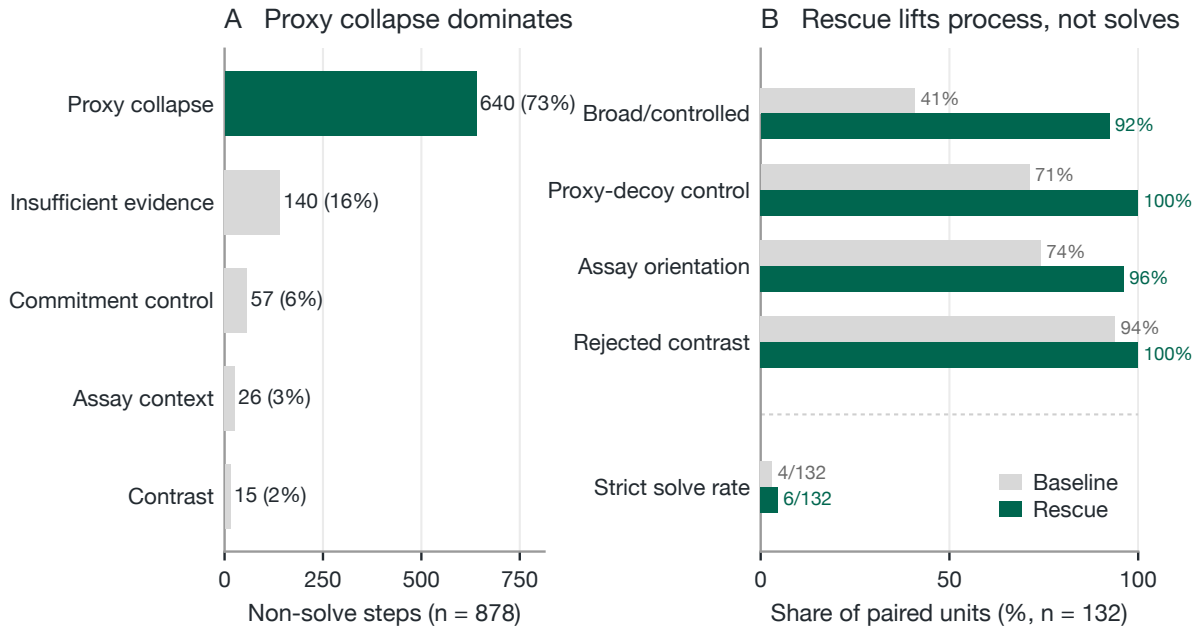

Figure 5: Process diagnostics and expert-method rescue. Panel A shows failure-step labels for 878 non-solves. Panel B separates process-behavior deltas from strict solves over the same 132 paired model-task units. The checklist improves observable workflow behavior more than hidden-verifier solves, so rescue is diagnostic rather than a replacement leaderboard.

5 Discussion

All 6 frontier agents use the released interface, produce valid submissions across the full matrix, and still solve few regulatory-intervention triage tasks under the hidden AlphaGenome-backed verifier. That combination rules out the easiest packaging explanation and ties the difficulty to the task and verifier regime rather than to runner failure.

The process diagnostics locate that difficulty. Agents often find public biological evidence, then convert it into a brittle final choice. The bottleneck appears after valid answer formation, when an attractive public proxy survives the agent's reasoning and controls the final intervention choice.

A regulatory edit can be plausible for the wrong reason. It can disrupt a strong motif, sit in a regulatory element, maximize a public sequence-model perturbation, or look

good under a generic locus story while failing the requested assay objective. Latent Mining makes that failure measurable by requiring the task to expose public evidence and withhold the verifier-backed outcome. The benchmark tests the step between evidence inspection and intervention commitment, which is where many biology and scientific-agent evaluations otherwise compress behavior into a single score.

The expert-method rescue experiment sharpens the interpretation. The hidden-blind checklist improves the process variables it was designed to affect, while strict solves move only from 4 baseline solves to 6 method-conditioned solves. That split separates two failure sources. Some failures are planning gaps, visible in the 5 baseline non-solves that become rescued solves, while the larger pattern is that a better public workflow still does not reliably recover the AlphaGenome-backed objective.

The checklist regressions bound this interpretation. Three baseline solves become rescue non-solves, and 2-provider wrong-answer convergence barely changes. Rescue reduces the strongest all-3-provider convergence on the same wrong answer, from 12 tasks to 5, but at least 2 providers still converge on the same wrong answer on 34 rescue tasks. The scaffold changes behavior without erasing public-proxy attraction, distinguishing underprompted workflow failures from cases where public evidence still points agents toward edits that fail the hidden AlphaGenome-backed objective.

Latent Mining is a benchmark-construction method, and the biology release is its first stress test. The release shows that task difficulty can remain interpretable when each score is tied to a replayable trajectory. Its contribution is a replayable evaluation object that holds hidden outcomes, public shortcut resistance, trace labels, and paired rescue evidence together without exposing the hidden answer.

6 Limitations

The hidden verifier is AlphaGenome-backed. It is not wet-lab truth, a universal biological oracle, or evidence that selected edits would validate experimentally. The narrower benchmark claim is that agents struggle to recover withheld AlphaGenome-backed regulatory targets from release-safe public evidence. Treating any selected edit as a biological intervention claim would require experimental validation through assays such as massively parallel reporter assays or CRISPR interference perturbations [29, 30]. Because the verifier is a model rather than an assay, some scientific-search failures may reflect disagreement between the agent and the AlphaGenome-backed objective rather than agent error, which bounds how strongly the failure labels can be read.

The biology release is also narrow by construction. It covers regulatory-intervention triage over 165 tasks, 11 loci, and 4 assay families. That scope is enough to test assay-aware public-proxy uncertainty, but it does not measure broad biological competence, clinical reasoning, wet-lab planning, or full-stack genomic discovery. The task panels also reflect the public evidence channels chosen for this release, fixing the public evidence surface to the models, annotations, and feature set used for this benchmark; future evidence stacks may shift which proxy gaps remain difficult.

The model matrix should be read with provider and runner boundaries intact. The Kimi K2.6 row is an effective completed row with mixed Moonshot API and Hugging Face Router provenance, and provider behavioral contrasts are bounded by the trace-depth differences detailed in Appendix G. The paired rescue core is complete for Codex GPT-5.5, Gemini 3.1 Pro, and GLM 5.1 over 132 paired model-task units, a diagnostic subset separate from the full unaided leaderboard.

The process labels are diagnostic annotations, not psychometric measurements. They combine deterministic trace features, public answer rationales, and quote-backed review over sanitized traces. They should be used to interpret failure modes, not to assign hidden partial credit. Near-misses have the same boundary. A near-miss is outcome-proximal under the hidden verifier but below the solve threshold, and it can be scientifically thoughtful, proxy-driven, or lucky. The paper keeps near-miss, process quality, and strict solve status separate because conflating them would make the benchmark easier to narrate and less useful.

The rescue intervention is an auxiliary diagnostic. It gives evidence about whether a hidden-blind workflow checklist changes behavior and outcomes under the completed paired core. It does not define a new benchmark setting, and method-conditioned solves are not ordinary unaided solves. The small solve delta should be interpreted with the process deltas and checklist regressions together. The result supports a mixed failure model rather than a single explanation for non-solves.

7 Conclusion

Latent Mining builds scientific-agent tasks from the gap between public evidence and hidden verification. In regulatory genomics, that gap produces a difficult but interpretable benchmark for assay-aware intervention triage. The aggregate strict solve rate is 11.3%, and no evaluated frontier agent solves more than 15% of tasks. Agents can inspect useful public evidence and still collapse to public proxies that fail the withheld AlphaGenome-backed regulatory objective.

The method contribution is this replayable evaluation object. The first biology release separates interface validity from scientific decision failure, and workflow-planning gaps from persistent public-prior/verifier gaps. Failures can be named, audited, and replayed without exposing the hidden answer.

References

[1] Žiga Avsec, Natasha Latysheva, Jun Cheng, Guido Novati, Kyle R. Taylor, Tom Ward, Clare Bycroft,

Lauren Nicolaisen, Eirini Arvaniti, Joshua Pan, Raina Thomas, Vincent Dutordoir, Matteo Perino, Soham De, Alexander Karollus, Adam Gayoso, Toby Sargeant, Anne Mottram, Lai Hong Wong, Pavol Drotár, Adam Kosiorek, Andrew Senior, Richard Tanburn, Taylor Applebaum, Souradeep Basu, Demis Hassabis, and Pushmeet Kohli. Advancing regulatory variant effect prediction with AlphaGenome. *Nature*, 649:1206–1218, 2026. doi: 10.1038/s41586-025-10014-0.

[2] Jeremy Li and Andrew Ho. GeneBench: Assessing AI agents for multi-stage inference problems in genomics and quantitative biology. *bioRxiv*, 2026. doi: 10.64898/2026.04.22.720113. URL https://www.biorxiv.org/content/10.64898/2026.04.22.720113v1.

[3] Kenny Workman, Zhen Yang, Harihara Muralidharan, Aidan Abdulali, and Hannah Le. scBench: Evaluating AI agents on single-cell RNA-seq analysis, 2026. URL https://arxiv.org/abs/2602.09063.

[4] Jon M. Laurent, Joseph D. Janizek, Michael Ruzo, Michaela M. Hinks, Michael J. Hammerling, Siddharth Narayanan, Manvitha Ponnapati, Andrew D. White, and Samuel G. Rodriques. LAB-Bench: Measuring capabilities of language models for biology research, 2024. URL https://arxiv.org/abs/2407.10362.

[5] Jon M. Laurent, Albert Bou, Michael Pieler, Conor Igoe, Alex Andonian, Siddharth Narayanan, James Braza, Alexandros Sanchez Vassopoulos, Jacob L. Steenwyk, Blake Lash, Andrew D. White, and Samuel G. Rodriques. LABBench2: An improved benchmark for AI systems performing biology research, 2026. URL https://arxiv.org/abs/2604.09554.

[6] Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://arxiv.org/abs/2410.05080.

[7] Ian Diks, Harihara Muralidharan, Tim Proctor, and Kenny Workman. Verifiable benchmarking of long-horizon spatial biology, 2026. URL https://arxiv.org/abs/2605.28065.

[8] Wenbin Guo, Minzhe Zhang, Bowei Han, Youjia Ma, Yang Leng, Shishir Hebbar, Xiaoyuan Zhou, Wenhao Gu, Xiao Yang, and Shashi Dhar. PromptBio-Bench: Benchmarking LLM-based bioinformatics agents for end-to-end data analysis. *bioRxiv*, 2026. doi: 10.64898/2026.05.05.723092. URL https://www.biorxiv.org/content/10.64898/2026.05.05.723092v2.

[9] Garyk Brixi, Matthew G. Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang, Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, Mohsen Naghipourfar, Eric Nguyen, Chiara Ricci-Tam, David W. Romero, Gwanggyu Sun, Ali Taghibakshi, Anton Vorontsov, Brandon Yang, Myra Deng, Liv Gorton, Nam Nguyen, Nicholas K. Wang, Etowah Adams, Stephen A. Baccus, Steven Dillmann, Stefano Ermon, Daniel Guo, Rajesh Ilango, Ken Janik, Amy X. Lu, Reshma Mehta, Mohammad R. K. Mofrad, Madelena Y. Ng, Jaspreet Pannu, Christopher Ré, Jonathan C. Schmok, John St. John, Jeremy Sullivan, Kevin Zhu, Greg Zynda, Hani Goodarzi, Patrick D. Hsu, and Brian L. Hie. Genome modelling and design across all domains of life with Evo 2. *Nature*, 652:1349–1361, 2026. doi: 10.1038/s41586-026-10176-5.

[10] Loubna Ben Allal, Qiuyi Li, Maurizio Fiusco, Lewis Tunstall, Kashif Rasul, Ed Beeching, Dana Aubakirova, Carlos Patiño, Thibaud Frere, Anton Lozhkov, Georgia Channing, Thomas Wolf, Diego di Bernardo, and Leandro von Werra. Carbon: Decoding the language of life. *bioRxiv*, 2026. doi: 10.64898/2026.05.22.727119. Preprint; model collection available at https://huggingface.co/collections/HuggingFaceBio/carbon.

[11] The ENCODE Project Consortium. An integrated encyclopedia of DNA elements in the human genome. *Nature*, 489(7414):57–74, 2012. doi: 10.1038/nature11247.

[12] Roadmap Epigenomics Consortium. Integrative analysis of 111 reference human epigenomes. *Nature*, 518(7539):317–330, 2015. doi: 10.1038/nature14248.

[13] The GTEx Consortium. The GTEx consortium atlas of genetic regulatory effects across human tissues. *Science*, 369(6509):1318–1330, 2020. doi: 10.1126/science.aaz1776.

[14] Jaime A. Castro-Mondragon, Rafael Riudavets-Puig, Ieva Rauluseviciute, Roza Berhanu Lemma, Laura Turchi, Romain Blanc-Mathieu, Jeremy Lucas, Paul Boddie, Aziz Khan, Nicolás Manosalva Pérez, Oriol Fornes, Tiffany Y. Leung, Alejandro Aguirre, Fayrouz Hammal, Daniel Schmelter, Damir Baranasic, Benoit Ballester, Albin Sandelin, Boris Lenhard, Klaas Vandepoele, Wyeth W. Wasserman, François Parcy, and Anthony Mathelier. JASPAR 2022: the 9th release of the open-access database of transcription factor binding

profiles. *Nucleic Acids Research*, 50(D1):D165–D173, 2022. doi: 10.1093/nar/gkab1113.

[15] Ivan V. Kulakovskiy, Ilya E. Vorontsov, Ivan S. Yevshin, Ruslan N. Sharipov, Alla D. Fedorova, Eugene I. Rumynskiy, Yulia A. Medvedeva, Arturo Magana-Mora, Vladimir B. Bajic, Dmitry A. Papatsenko, Fedor A. Kolpakov, and Vsevolod J. Makeev. HOCOMOCO: towards a complete collection of transcription factor binding models for human and mouse via large-scale ChIP-seq analysis. *Nucleic Acids Research*, 46 (D1):D252–D259, 2018. doi: 10.1093/nar/gkx1106.

[16] Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre, Kevin Meng, Rebecca Weiss, Fazl Barez, Rahul Gupta, Jwala Dhamala, Jacob Merizian, Mario Giulianelli, Harry Coppock, Cozmin Ududec, Jasjeet Sekhon, Jacob Steinhardt, Antony Kellermann, Sarah Schwettmann, Matei Zaharia, Ion Stoica, Percy Liang, and Daniel Kang. Establishing best practices for building rigorous agentic benchmarks, 2025. URL https://arxiv.org/abs/2507.02825.

[17] Yu Lu Liu, Su Lin Blodgett, Jackie Cheung, Q. Vera Liao, Alexandra Olteanu, and Ziang Xiao. ECBD: Evidence-centered benchmark design for NLP. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16349–16365, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.861. URL https://aclanthology.org/2024.acl-long.861/.

[18] Jie Gao, Kaiser Sun, Jen-tse Huang, Katherine Van Koevering, Sijie Ji, Heyuan Huang, Weiyan Shi, Zhuoran Lu, Ziang Xiao, Daniel Khashabi, and Mark Dredze. How to interpret agent behavior, 2026. URL https://arxiv.org/abs/2605.13625.

[19] Afra Feyza Akyürek, Xinming Tu, Sofia Monasdotter, Yuanhao Qu, Sergey Chekhov, and Sami Hassaan. Can coding agents tackle early-stage drug discovery? Scale Labs (in collaboration with Phylo), blog post, June 2026. URL https://labs.scale.com/blog/coding-agents-drug-discovery. Published 4 June 2026.

[20] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

[21] Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, Junhong Shen, Guanghao Ye, Haowei Lin, Jason Poulos, Maoyu Wang, Marianna Nezhurina, Jenia Jitsev, Di Lu, Orfeas Menis Mastromichalakis, Zhiwei Xu, Zizhao Chen, Yue Liu, Robert Zhang, Leon Liangyu Chen, Anurag Kashyap, Jan-Lucas Uslu, Jeffrey Li, Jianbo Wu, Minghao Yan, Song Bian, Vedang Sharma, Ke Sun, Steven Dillmann, Akshay Anand, Andrew Lanpouthakoun, Bardia Koopah, Changran Hu, Etash Guha, Gabriel H. S. Dreiman, Jiacheng Zhu, Karl Krauth, Li Zhong, Niklas Muennighoff, Robert Amanfu, Shangyin Tan, Shreyas Pimpalgaonkar, Tushar Aggarwal, Xiangning Lin, Xin Lan, Xuandong Zhao, Yiqing Liang, Yuanli Wang, Zilong Wang, Changzhi Zhou, David Heineman, Hange Liu, Harsh Trivedi, John Yang, Junhong Lin, Manish Shetty, Michael Yang, Nabil Omi, Negin Raoof, Shanda Li, Terry Yue Zhuo, Wuwei Lin, Yiwei Dai, Yuxin Wang, Wenhao Chai, Shang Zhou, Dariush Wahdany, Ziyu She, Jiaming Hu, Zhikang Dong, Yuxuan Zhu, Sasha Cui, Ahson Saiyed, Arinbjörn Kolbeinsson, Jesse Hu, Christopher Michael Rytting, Ryan Marten, Yixin Wang, Alex Dimakis, Andy Konwinski, and Ludwig Schmidt. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces, 2026. URL https://arxiv.org/abs/2601.11868.

[22] Qixing Zhou, Jiacheng Zhang, Haiyang Wang, Rui Hao, Jiahe Wang, Minghao Han, Yuxue Yang, Shuzhe Wu, Feiyang Pan, Lue Fan, Dandan Tu, and Zhaoxiang Zhang. FeatureBench: Benchmarking agentic coding for complex feature development. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2602.10975.

[23] Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. SciCode: A research coding benchmark curated by scientists, 2024. URL https://arxiv.org/abs/2407.13168. Project page: https://scicode-bench.github.io/.

[24] Jason D. Buenrostro, Paul G. Giresi, Lisa C. Zaba, Howard Y. Chang, and William J. Greenleaf. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. *Nature Methods*, 10(12):1213–1218, 2013. doi: 10.1038/nmeth.2688.

[25] David S. Johnson, Ali Mortazavi, Richard M. Myers, and Barbara Wold. Genome-wide mapping of in vivo protein-DNA interactions. *Science*, 316(5830):1497–1502, 2007. doi: 10.1126/science.1141319.

[26] Toshiyuki Shiraki, Shinji Kondo, Shintaro Katayama, Kazunori Waki, Takeya Kasukawa, Hideya Kawaji, Rimantas Kodzius, Akira Watahiki, Mari Nakamura, Takahiro Arakawa, Shiro Fukuda, Daisuke Sasaki, Anna Podhajska, Matthias Harbers, Jun Kawai, Piero Carninci, and Yoshihide Hayashizaki. Cap analysis gene expression for high-throughput analysis of transcriptional starting point and identification of promoter usage. *Proceedings of the National Academy of Sciences*, 100(26):15776–15781, 2003. doi: 10.1073/pnas.2136655100.

[27] The FANTOM Consortium and the RIKEN PMI and CLST (DGT). A promoter-level mammalian expression atlas. *Nature*, 507(7493):462–470, 2014. doi: 10.1038/nature13182.

[28] Zhong Wang, Mark Gerstein, and Michael Snyder. RNA-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009. doi: 10.1038/nrg2484.

[29] Charles P. Fulco, Joseph Nasser, Thouis R. Jones, Glen Munson, Drew T. Bergman, Vidya Subramanian, Sharon R. Grossman, Rockwell Anyoha, Benjamin R. Doughty, Tejal A. Patwardhan, Tung H. Nguyen, Michael Kane, Elizabeth M. Perez, Neva C. Durand, Caleb A. Lareau, Elena K. Stamenova, Erez Lieberman Aiden, Eric S. Lander, and Jesse M. Engreitz. Activity-by-contact model of enhancer–promoter regulation from thousands of CRISPR perturbations. *Nature Genetics*, 51(12):1664–1669, 2019. doi: 10.1038/s41588-019-0538-0.

[30] Alexandre Melnikov, Anand Murugan, Xiaolan Zhang, Tiberiu Tesileanu, Li Wang, Peter Rogov, Soheil Feizi, Andreas Gnirke, Curtis G. Callan Jr, Justin B. Kinney, Manolis Kellis, Eric S. Lander, and Tarjei S. Mikkelsen. Systematic dissection and optimization of inducible enhancers in human cells using a massively parallel reporter assay. *Nature Biotechnology*, 30(3):271–277, 2012. doi: 10.1038/nbt.2137.

A Benchmark construction constraints

Table A.1 lists the construction constraints a candidate task must satisfy before admission. Each constraint pairs a requirement with the failure mode that appears if the requirement is dropped. The constraints are enforced during construction rather than asserted after the fact, so a panel that fails any row is repaired or rejected before it enters the released matrix.

Table A.1: Latent Mining construction constraints. Each row states a requirement and the failure mode if it is violated. These constraints define task admission and separate task quality from packaging success.

Principle	Requirement	Failure mode if violated
Workflow-grounded decision	Each task begins from a domain workflow and a decision a scientist could plausibly make.	The benchmark becomes generic question answering.
Public-evidence traction	Public model, motif, annotation, and assay-context signals carry useful but incomplete evidence.	Agents have no scientific basis for search and low scores are uninformative.
Hidden verifier identifiability	The hidden objective defines a private answer without exposing scores, thresholds, ranks, cache or target identifiers, or known solutions.	The task is under-specified or leaks the answer.
Clear wrong-answer separation	Plausible public-proxy and shortcut answers fail by a meaningful margin under hidden verification.	Grading depends on tolerance tuning rather than scientific correctness.
Minimal viable prompt	The public prompt defines the decision, files, output schema, and constraints without hinting at the hidden path.	The task collapses into prompt-following.
Shortcut resistance	No single public scalar, action type, locus artifact, or metadata field predicts hidden winners.	The task measures shortcut discovery, not scientific reasoning.
Repair without leakage	Post-verifier repair improves a panel only if it does not leak the hidden winner or weaken thresholds.	Construction contaminates the public surface.
Release-safe packaging	Public bundles include only solver-facing files and release-safe manifests.	Hidden verifier state leaks into the release.
Calibration headroom	Frontier agents execute validly but do not saturate the task set.	The benchmark is invalid or too easy to measure progress.
Behavior interpretability	Near-misses and failures map to scientific-search behaviors rather than runner defects.	Low solve rates cannot be read as capability headroom.

B Public and private boundary

The benchmark separates an agent-visible public surface from a withheld private surface, as Table B.1 states. The public surface holds everything a domain scientist could inspect before committing to an intervention. The private surface holds the hidden verifier state used to admit and grade tasks. No private field appears in the released bundle, the public validator output, the trajectory artifacts, or this paper.

Table B.1: Public and private field boundary. The agent reasons over the public surface and selects one candidate. The private surface is withheld from the released bundle, the public validator, the trajectories, and this paper.

Public surface (agent sees)	Private surface (withheld)
Assay family and molecular readout	Hidden verifier values for each candidate
Cell or tissue context	Solve and near-miss thresholds
Locus metadata	Candidate ranks under the hidden objective
Candidate edits and their in-task option labels	Target identifier of the hidden winner
Sequence-model likelihood summaries	Cache identifiers
Embedding-distance features	Known solution
Motif and position-weight-matrix changes	Designer rationale
Edit geometry, chromatin context, mutagenesis features	
Composition shifts and public-signal disagreement	
Public validator status	

C Task atlas detail

The main-text atlas keeps the locus strip and assay counts in the foreground. Table C.1 gives the full per-locus outcome breakdown, and Table C.2 gives the assay-family, regulatory-element, and edit-type coverage. Each locus contributes 15

tasks and 90 ranked attempts across the six complete model rows. Outcome counts are release-safe class labels, not hidden scores.

Table C.1: Per-locus outcomes. Each locus contributes 15 tasks and 90 ranked attempts. Gene symbols follow standard nomenclature. Counts are solve, near-miss, and failure attempts under hidden verification.

Gene	Biology system	Solves	Near-misses	Failures
BCL11A	hemoglobin switching	7	25	58
CFTR	cystic fibrosis, epithelial transport	3	25	62
FTO/IRX3	obesity, metabolism	18	18	54
GATA1	hematopoiesis	7	27	56
HBB/HBE	hemoglobinopathy	19	20	51
IL2RA	autoimmunity	18	18	54
INS	diabetes, endocrine	6	17	67
LCT/MCM6	lactase regulation, digestion	12	16	62
MYC	cancer proliferation	12	9	69
PCSK9	cholesterol metabolism	4	26	60
TERT	cancer, aging	6	51	33
Total	11 loci	112	252	626

Table C.2: Atlas coverage axes. Regulatory-element types count tasks. Dominant edit type is the task-level action; option-level counts span all candidates across panels.

Axis	Category	Count
Regulatory element	promoter/enhancer	75
	distal enhancer	30
	distal enhancer / super-enhancer	15
	enhancer	15
	locus-control-region enhancer	15
	promoter/enhancer / TF-bound element	15
Dominant edit type (task)	deletion	128
	insertion	23
	single-nucleotide variant	14
Edit type (option pool)	deletion	849
	insertion	383
	single-nucleotide variant	418

D Agent-behavior taxonomy

The process layer labels every non-solve and every attempt without exposing private verifier data. Labels are derived from public answer rationales, sanitized trace spans, deterministic trace features, and quote-backed review. Private scores, ranks, numeric diagnostics, cache identifiers, known solutions, and evaluator internals are excluded from the process artifacts. The labels are diagnostic evidence, not hidden partial credit.

The phase taxonomy follows the six regulatory-triage steps used in the main analysis, namely assay orientation, evidence-table construction, proxy-decoy control, contrastive triage, commitment control, and valid finalization.

The near-miss quality labels separate principled near-misses from proxy-driven ones. A good scientific near-miss integrates public sequence-model evidence with motif and regulatory context, compares alternatives, and treats the hidden objective as uncertain rather than directly recoverable. A mixed proxy-attracted near-miss shows real biological search structure whose final decision is still pulled toward a public-proxy maximum or composite ranking, while a public-proxy overfit near-miss leans on public rankings or a single proxy family with weak biological grounding. The outcome-near process-poor and uncertain labels cover outcome-proximal runs with thin reasoning and runs whose public trace is too sparse to label confidently.

The proxy-failure label records the attractor that pulled a near-miss off the hidden-verified choice. Public scalar collapse marks a decision driven by the largest public scalar, such as the largest sequence-model perturbation, rather than assay-matched evidence; single-channel overweighting marks one channel dominating the others; assay-context mismatch

marks evidence that does not match the requested readout; and weak rejected-alternative contrast marks alternatives that were not separated sharply from plausible decoys. The failure-step labels apply the same vocabulary to every non-solve at the step where the run failed, from proxy-collapse failure and insufficient evidence through commitment-control, assay-context, and contrast failures. The exploration labels grade search breadth, separating a focused narrow but valid search from broad and controlled exploration that stays assay-aware and from broad but uncontrolled exploration that does not.

Every label is assigned from public answer rationales, sanitized trace spans, deterministic trace features, and quote-backed review, with a per-row confidence recorded. Release-provider near-miss adjudications are preserved where available, and the remaining rows carry lower-confidence quote-backed heuristics, which is why the paper reports these labels as diagnostic evidence rather than calibrated measurements and keeps provider comparisons bounded by trace depth.

Tables D.1 and D.2 give the label counts. Near-miss quality labels are assigned over the 252 valid below-threshold near-misses. Failure-step labels are assigned over all 878 non-solves, and exploration labels over all attempts.

Table D.1: Near-miss quality and proxy-failure labels. Counts over 252 valid below-threshold near-misses. Quality labels separate good scientific near-misses from proxy-attracted ones. Proxy-failure labels record the dominant attractor.

Near-miss quality	Rows	Proxy-failure label	Rows
Mixed proxy-attracted	153	Public scalar collapse	189
Public-proxy overfit	55	Single-channel overweighting	38
Good scientific near-miss	32	Assay-context mismatch	18
Uncertain	8	None	4
Outcome-near process-poor	4	Weak rejected-alternative contrast	3

Table D.2: Failure-step and exploration labels. Failure-step labels are assigned over 878 non-solves. Exploration labels are assigned over all attempts. The dominant non-solve label is proxy-collapse failure, not interface failure.

Failure step	Rows	Exploration quality	Rows
Proxy-collapse failure	640	Narrow but valid	866
Insufficient evidence	140	Broad and controlled	86
Commitment-control failure	57	Uncertain	30
Assay-context failure	26	Broad but uncontrolled	8
Contrast failure	15		

E Expert-method rescue checklist and leakage controls

The expert-method rescue arm appends a hidden-blind workflow checklist to the standard task surface. The checklist is general public workflow guidance that tells the agent to orient to the requested assay and molecular readout, build an evidence table across public channels, name the strongest public-scalar and motif decoys, contrast plausible alternatives under the assay objective, and avoid collapsing to a single public scalar. The checklist contains no candidate-specific content.

The rescue protocol forbids a fixed list of fields in any rescue prompt. The forbidden fields are the hidden answer, the hidden rank, the hidden score, the target identifier, any candidate-specific hint, the private cache identifier, and the designer rationale. A leakage scan distinguishes prompt and release-facing surfaces from internal audit artifacts. Candidate option identifiers are absent from rescue prompts, and hidden verifier values, ranks, thresholds, cache identifiers, known solutions, and designer rationales are absent from release-facing validator and paper-summary surfaces. Boundary-language hits in prompts are the prohibition statements quoted above, not leaked values. Internal manifests and quote-locator JSONL retain local paths and answer locators for audit reproducibility and are excluded from public release.

F Representative task anatomy

A task bundle is a directory of public files plus a hidden grader the agent never reads, holding about 15 public files per task. The public files are a task specification, a candidate file, a public evidence guide, an instruction file, a public validator, and per-candidate feature files. The 165 bundles contain 2,475 public files in total, all covered by the public bundle scan at the frozen release checkpoint. This layout is what makes a low solve rate interpretable, because the agent can open the dossier, read every public channel, write a schema-valid answer, and still fail the hidden objective, with the failure recorded as a replayable trajectory rather than a single score.

G Evaluation protocol

Every model received an identical task prompt and an equivalent shell-scoped, network-isolated workspace. The agent was told to inspect the local task files, use public tools if helpful, and write an answer file that selects one candidate, lists at least two public evidence channels and at least two rejected options, and gives a rationale of at least forty characters. A public validator checked interface validity only, while the hidden AlphaGenome grading ran separately and was never visible to the agent. Each task ran offline under a per-task wall-clock limit, with network commands blocked at the runner. The runners differ in one way that the paper treats as a confound, because the three coding-CLI models ran inside their native agent toolsets while the three API models were each given a single shell function, which is why trace verbosity differs by an order of magnitude across providers (Table G.1).

Table G.1: Per-provider execution protocol. The task prompt, answer schema, public validator, offline sandbox, and per-task wall-clock limit are shared across all six models. Runner and tool surface differ between the coding-CLI models and the API models, which is the trace-depth confound the paper bounds rather than compares. Mean turns are measured over the sanitized traces. Gemini 3.1 Pro and GLM 5.1 used temperature 0; the other rows left sampling at provider defaults, which were not recorded. The API models used a 131,072-token output cap and a 64 tool-call budget per task, while the coding-CLI rows were bounded by their harness budgets. Retries were manual re-runs of timed-out or interface-invalid attempts, with the last valid attempt kept per task. Runs span 31 May to 4 June 2026. †The Kimi K2.6 row mixes 121 native Moonshot API runs and 44 Hugging Face Router fallback fills.

Model	Runner	Model identifier	Tool surface	Mean turns
Codex GPT-5.5	Codex CLI	`gpt-5.5`	native toolset	28
Claude Opus 4.7	Claude Code CLI	`claude-opus-4-7`	native toolset	2
Claude Sonnet 4.6	Claude Code CLI	`claude-sonnet-4-6`	native toolset	2
Gemini 3.1 Pro	google-genai API	`gemini-3.1-pro-preview`	shell function	51
Kimi K2.6	Moonshot API†	`kimi-k2.6`	shell function	68
GLM 5.1	HF Router API	`zai-org/GLM-5.1`	shell function	42

Because the coding-CLI runners expose their full agent toolset while the API runners expose a single shell function, the provider process labels in the main-text results are bounded by trace depth rather than compared.