CREDITAGENTBENCH: MEASURING AGENTIC CREDIT-MODEL REPAIR UNDER PRODUCTION CONSTRAINTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Repairing an existing credit model differs from developing a new one. A model may rank borrowers well but miscalibrate default probabilities, or improve on a development sample while weakening under temporal shift. A repair is useful only if the other operational requirements remain intact. Credit and financial-risk data make this tension concrete because ranking, calibrated probabilities, prediction error, and temporal stability matter together. We introduce CREDITAGENTBENCH, an executable benchmark of *aspirational constrained model repair*. Each task fixes an incumbent, an improvement goal, and protected boundaries that must each be met; a participant explores on visible data and commits one final model. Evaluation reports progress toward the goal and the margin of the least-satisfied constraint. The task bank has 57 Natural and Controlled cells from six sources. In the matched 27-task Controlled comparison within the 1,026-assignment reported visible-development set, HPO attains higher average margins, while the dynamic agent crosses both boundaries slightly more often. That joint-validity advantage does not persist when the committed models are replayed on sealed data. Five further model–provider backends change both the agent–HPO comparison and the fraction of runs that yield a scoreable model. These results show why repair progress, joint validity, and final-model availability must be reported separately.

1 INTRODUCTION

A deployed credit-risk model can rank borrowers well yet systematically misestimate default probabilities. Improving discrimination on a development sample may also worsen calibration slope or temporal stability. A single predictive score therefore does not settle whether a repair is usable in production. MLAgentBench and MLE-bench, for example, evaluate research-code changes and competition modeling (Huang et al., 2024; Chan et al., 2025). Here the starting point is a functioning incumbent that fails a particular audit: can an autonomous system repair that defect without violating existing operational constraints?

Repair requires a sequence of decisions: what to test, how to use multi-metric feedback, and which candidate to commit when the budget is spent. In the Controlled track, holding the candidate space and modeling opportunities fixed lets us study how controllers allocate experiments and select a final model. Finding a good candidate is not enough if the participant commits a different one.

We formalize this challenge as *aspirational constrained model repair* (Figure 1). Each task pairs an incumbent with a frozen repair goal and production boundaries that must each be met. The goal is an aspiration, not a promise that the available model family can attain it. Under our *scoreability-first* protocol, a task enters the bank only if its data, required metrics, and artifact replay define a reproducible evaluation path. Its goal is not chosen from a candidate already known to win. A committed model receives numerical margins only when its required measurements are defined; execution and measurement failures remain explicit outcomes.

CREDITAGENTBENCH instantiates this framework in credit and financial-risk modeling, where discrimination, probability accuracy, calibration, and temporal robustness require different algorithmic choices. Spanning calibration-slope, discrimination, and probability-accuracy repairs, the benchmark pairs natural empirical defects with controlled defects across static, retrospective temporal, and chronological temporal regimes. Participants explore candidates via raw WORKING feedback but

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

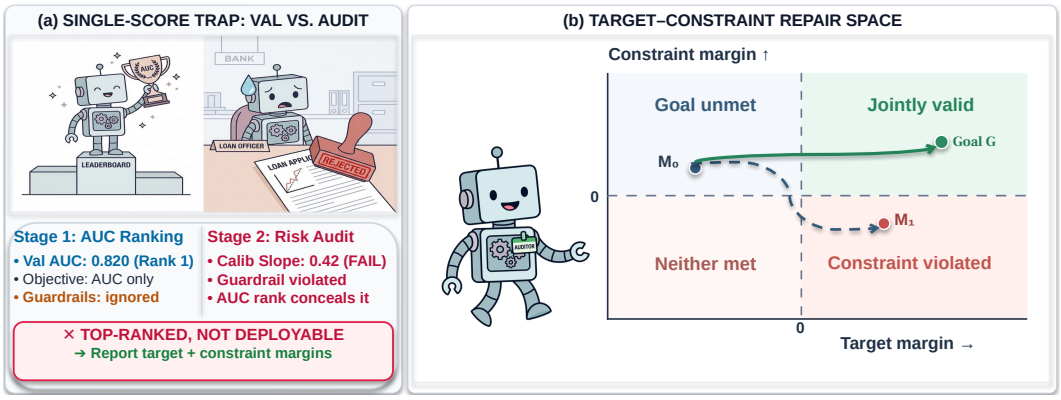

Figure 1: **Why single-score selection is insufficient.** (a) A high validation rank can hide a failed production audit (illustrative values). (b) CREDITAGENTBENCH evaluates a committed repair on separate target and protected-constraint margins. Schematic paths contrast the jointly valid goal (green) with a target-only repair that violates a constraint (dashed).

must commit a single artifact for a one-shot evaluator replay on HIDDEN data, evaluating autonomous model selection rather than test-set oracle cherry-picking.

Evaluation decouples performance into two primary coordinates: the signed target margin in goal units (m_{target}) and the minimum signed constraint margin in minimally consequential difference (MCID) units ($m_{\text{constraint}}$). A surplus on the target axis cannot compensate for a violated constraint, nor does a conservative model devoid of repair progress constitute success (Figure 1b). Within the Controlled track, a shared hyperparameter optimization (HPO) substrate lets us compare search-allocation and commitment policies. Native open systems provide a separate view of operational performance without requiring identical internal model spaces.

The paper contributes:

- An incumbent-centered repair protocol with a frozen aspiration and protected constraints. Only the participant's committed model is scored on sealed data; the trajectory remains available for diagnosis.
- A scoreability-first bank of 57 natural and controlled task cells across six sources, three repair geometries, and static and temporal regimes. Target and constraint margins remain separate, with diagnostics for final selection and runtime.
- Evidence from 1,026 reported assignments and five additional 279-assignment model–provider cohorts. On matched Controlled tasks, HPO has higher mean margins, while the agent has a small visible joint-validity advantage that does not persist on sealed replay. Backend changes also alter final-model scoreability.

2 RELATED WORK

Machine learning engineering and research agents. Executable benchmarks test agents on research code modification (Huang et al., 2024), machine-learning competitions (Chan et al., 2025), repository issues (Jimenez et al., 2024), interactive coding (Yang et al., 2023), and data-science workflows (Hong et al., 2025; Chen et al., 2025). CREDITAGENTBENCH asks a different question within an executable workflow: can an agent repair an existing model while preserving separately specified production metrics?

Production-oriented data science. GRACE-DS evaluates tabular agents with hidden checks for leakage, protocol adherence, and reproducibility (Tsymbalov et al., 2026); such checks matter because validation mistakes can invalidate apparently strong ML results (Kapoor & Narayanan, 2023). It selects a terminal candidate using validation evidence and replays that candidate once on Hidden data. CREDITAGENTBENCH instead starts from an audited incumbent and requires the participant to

choose and commit the model evaluated against its repair target and protected metrics. The distinction is who selects the final artifact, not whether either benchmark uses a sealed test.

Agents, hyperparameter optimization, and AutoML. AgentHPOBench tests sequential configuration choices in executable research repositories and scores the last intervention, while analyzing the trajectory separately (Huai et al., 2026). Other work uses language models to propose configurations or programs (Yang et al., 2024; Chen et al., 2022; Zhang et al., 2023; Romera-Paredes et al., 2024). Classical optimizers and AutoML systems remain important comparisons (Bergstra et al., 2011; Akiba et al., 2019; Deb et al., 2002; Erickson et al., 2020). Our Controlled track compares controllers with a shared candidate space, tools, feedback, and opportunity budget; the Natural track retains each method's native procedure. Appendix A discusses the broader HPO and AutoML literature.

Positioning. Table 1 compares the artifact used for each benchmark's reported outcome. CREDITAGENTBENCH evaluates the model committed by the participant against a repair target and separate production boundaries.

Table 1: **The evaluation object differs across adjacent executable benchmarks.** "Final" denotes the artifact on which the reported outcome is based.

Benchmark	Evaluation object	Final	Primary outcome	Mechanism isolation
MLAgentBench	Research code	Agent's workspace	Task metric improvement	Agent / model comparison
MLE-bench	Competition modeling	Submitted predictions	Leaderboard score / medals	Scaffold comparison
AgentHPOBench	Sequential HPO	Last intervention	Final-step metric vs. baseline / anchor	Feedback and HPO comparisons
GRACE-DS	Data-science workflow	Validation-selected candidate	Hidden quality / protocol validity	Workflow / state ablations
CREDITAGENTBENCH	Incumbent repair	Participant-committed model	Target and constraint margins	Shared-HPO controller ablations

3 ASPIRATIONAL CONSTRAINED MODEL REPAIR

We formalize the repair problem independently of credit risk specifics. A task is defined as

$$\mathcal{T} = (D_{\text{train}}, D_{\text{work}}, D_{\text{gate}}, D_{\text{hidden}}, M_0, \mathcal{G}, \mathcal{C}, \mathcal{R}, \mathcal{F}, \mathcal{B}), \tag{1}$$

where M_0 is the operational incumbent model, $\mathcal{G}$ represents the repair aspiration, $\mathcal{C}$ is a set of non-compensatory production constraints (Gardner et al., 2014; Gelbart et al., 2014; Hernández-Lobato et al., 2016), $\mathcal{R}$ specifies the legal repair grammar, $\mathcal{F}$ governs visible development feedback, and $\mathcal{B}$ bounds allowable modeling opportunities. The four dataset roles are mutually disjoint. D_{train} supports model fitting and diagnostics, while D_{work} supplies the participant-visible development audit used by the repair loop. D_{gate} is a private, preregistered audit of the committed model: participants cannot query it or use its feedback to choose a candidate. D_{hidden} is reserved for sealed post-development replay and is not materialized as a feature–label evaluation frame during construction. A participant trains candidates $M_1, \ldots, M_K$ on D_{train} and queries the permitted multi-metric evaluations during development. Ultimately, the participant autonomously executes $\text{commit}(M_k)$ for one generated artifact; the evaluator checks that artifact at the gate and later replays it on D_{hidden}.

The submitted artifact, rather than the best archived candidate, is the unit of evaluation. We retain the trajectory to examine candidate discovery and final commitment separately; neither the Visible nor the Hidden score substitutes an uncommitted model for the participant's choice.

3.1 TARGET PROGRESS

Each metric is mapped to an adequacy scale where higher is better: identity for higher-is-better metrics, negation for losses, and negative absolute distance from one for calibration slope. Let $a_g(M)$ be target adequacy and $\underline{\Delta}_g(M)$ denote a conservative estimate of the improvement $a_g(M) - a_g(M_0)$. The task freezes a required improvement $\delta_g > 0$. The signed target margin is

$$m_{\text{target}}(M) = \frac{\underline{\Delta}_g(M) - \delta_g}{\delta_g}, \qquad d_{\text{target}}(M) = \max(0, -m_{\text{target}}(M)). \tag{2}$$

The incumbent has margin -1 by construction; 0 reaches the aspiration, and positive values exceed it. This normalization standardizes progress across heterogeneous metrics into commensurable goal units.

3.2 PRODUCTION VALIDITY

Each production constraint $j \in \mathcal{C}$ freezes an adequacy boundary b_j and a minimally consequential difference (MCID) scale s_j. For conservative estimate $\underline{a}_j(M)$, the individual margin is $m_j(M) = (\underline{a}_j(M) - b_j)/s_j$. The task-level constraint outcome is

$$m_{\text{constraint}}(M) = \min_{j \in \mathcal{C}} m_j(M), \qquad v_{\text{constraint}}(M) = \max(0, -m_{\text{constraint}}(M)). \tag{3}$$

Protected predictive-metric boundaries are frozen from the incumbent's WORKING bootstrap distribution; per-run scores use one-sided 95% bootstrap bounds, while paired method contrasts use separate two-sided 95% intervals. Taking the minimum enforces non-compensatory validity: surplus on one constraint cannot offset violation of another. We treat $(m_{\text{target}}, m_{\text{constraint}})$ as a vector rather than collapsing them into a scalar reward, aligning with multiobjective optimization principles (Miettinen, 1998; Emmerich & Deutz, 2018). As shown in Figure 1b, the two margins place a candidate in one of four regions defined by the boundaries and show how far it lies from each boundary, even when neither is crossed. Binary threshold indicators ($\mathbb{1}[m_{\text{target}} \geq 0], \mathbb{1}[m_{\text{constraint}} \geq 0]$) and joint crossing serve as secondary diagnostic summaries.

3.3 SCOREABILITY-FIRST TASKS

A task is *scoreable* when its data regime, target, constraints, uncertainty estimators, missingness policies, and artifact replay define an executable route to Equations 2–3. A particular submission may still fail to execute or yield an undefined required measurement; these outcomes are recorded rather than assigned an invented numerical margin. Task admission requires a verified incumbent vulnerability and finite scales, but not a construction-time model that already reaches the goal.

The aspiration follows a frozen logical-headroom policy (Section 4.3) rather than the performance of the best construction-time candidate. The target is therefore not calibrated to a known winner; whether the available candidates can reach it remains an empirical question for the evaluated systems.

4 CREDITAGENTBENCH

Credit risk modeling has long relied on empirical classification benchmarks (Hand & Henley, 1997; Thomas et al., 2002; Baesens et al., 2003; Crook et al., 2007; Lessmann et al., 2015), but its operational metrics demand different actions (Figure 2). Discrimination evaluates borrower default ranking, calibration assesses probability consistency, and probability accuracy penalizes prediction error. A monotone transformation can recalibrate probabilities without changing ranking, whereas a more flexible classifier may improve ranking while worsening calibration. Historical cohorts also expose models to changes that a random held-out split cannot represent. These tensions make credit modeling a useful setting for testing whether an agent repairs the requested property while protecting the others.

4.1 AUDIT-TO-REPAIR CONSTRUCTION

Task construction begins from an incumbent with an audited vulnerability (Figure 2a). The authoring protocol freezes the target, protected constraints, data regime, visible WORKING feedback, and legal

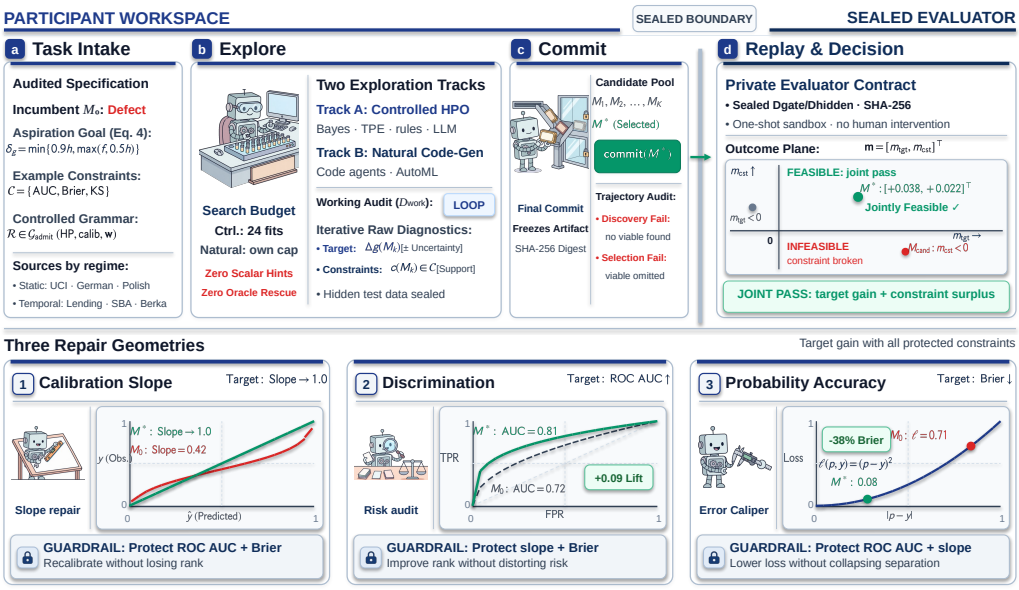

Figure 2: **CREDITAGENTBENCH evaluation workflow.** Task cards and sealed contracts define the repair problem. Participants explore candidates on WORKING feedback and commit one final artifact; the evaluator checks the gate and replays the artifact on HIDDEN data. The lower row shows the three repair geometries and their protected metrics.

repair grammar. It produces a participant-facing Task Card and a private contract binding data splits, labels, metric versions, uncertainty procedures, and artifact identities. Admission checks that the incumbent and evaluation path are executable; it does not require a reference candidate that already solves the task.

4.2 REPAIR GEOMETRIES

The task bank spans three repair geometries: (i) **calibration-slope repair** targets the discrepancy between calibration slope and unity (MCID 0.05), while ROC AUC and Brier score (Brier, 1950) remain protected against degradation; (ii) **discrimination repair** targets ROC AUC (MCID 0.01), while calibration slope and Brier score prevent ranking gains achieved via distorted probabilities; and (iii) **probability-accuracy repair** targets Brier score (MCID 0.005), preserving both AUC and calibration slope (Steyerberg et al., 2010; Van Calster et al., 2019; Guo et al., 2017). Auxiliary constraints monitor out-of-fold discrepancies, bound validation degradation, and restrict Kolmogorov–Smirnov drift under stressed sub-cohorts. These checks make specified side effects of a repair visible rather than assuming that target improvement is safe.

4.3 ASPIRATIONAL GOALS

For each task, let h be the incumbent's logical headroom to the metric ideal, and let f be three MCIDs in raw metric units. The frozen goal policy sets

$$\delta_g = \min\{0.9h,\ \max(f, 0.5h)\}. \tag{4}$$

This policy requires a material improvement when headroom permits, typically requesting half of theoretically available repair while never demanding the final 10% of the domain. Tasks with zero positive headroom are unscoreable rather than weakened. Equation 4 depends strictly on the incumbent and registered geometry without reference to construction-time winners.

4.4 TASK PORTFOLIO

Table 2 summarizes the six-source portfolio. UCI Default Credit Card (Yeh & Lien, 2009), South German Credit (Grömping, 2019), and Polish Companies Bankruptcy use static splits. Lending Club

(Ariza-Garzón et al., 2024) and Berka provide retrospective temporal tests; SBA 7(a) uses rolling origins. The ten dataset keys yield 57 admitted task cells across three repair geometries: 30 Natural and 27 Controlled. Berka contributes Natural tasks to the reported experiments; its prepared data are not part of the public release.

Table 2: **Task portfolio.** A key is a frozen static dataset or temporal origin. Each admitted cell has one repair geometry and either a Natural or Controlled defect.

Source	Regime	Keys	Natural	Controlled	Cells
UCI Default Credit Card	Static	1	3	3	6
South German Credit	Static	1	3	3	6
Polish Bankruptcy	Static	1	3	3	6
Lending Club	Retrospective temporal	3	9	9	18
SBA 7(a)	Rolling temporal	3	9	9	18
Berka	Retrospective temporal	1	3	0	3
Total		10	30	27	57

4.5 Natural and controlled task worlds

A *Natural* task targets an empirical weakness of the source-specific incumbent under production constraints, without an injected defect. A *Controlled* task applies a pre-registered, label-free inference-time perturbation to a fitted incumbent: logit temperature for calibration, intercept shift for probability accuracy, or feature-hashed logit noise for discrimination. Perturbation severity is drawn from a frozen global grid. These transforms leave labels, rows, and split membership unchanged. Their inverse operators are excluded from the participant's repair grammar, so the controlled task has a known defect but does not offer a direct undo operation.

4.6 Interaction and final submission

Participants receive the Task Card and training data, develop candidates, and query raw Working metrics with directional uncertainty and sample support (Figure 2b). The interface does not provide a composite reward or a recommended action. Controlled arms share a frozen HPO space and a 24-candidate budget: the optimizer selects configurations, while the agent or rule controller allocates searches and chooses when to commit. Natural/Open methods retain their native modeling procedures under the registered resource limits. Appendix F reports mean wall time for the matched Controlled arms separately from predictive outcomes.

Upon concluding exploration, the participant executes $\mathrm{commit}(M_{\mathrm{final}})$ for one previously generated artifact (Figure 2c). The evaluator verifies artifact integrity and performs a separate private Gate check. The reported Visible margins score the committed artifact on the Working audit; post-development evaluation replays that same artifact on sealed Hidden data (Figure 2d). The trajectory remains available for analysis, but its best uncommitted candidate does not replace the final model.

5 Experimental Design

We use three tracks to separate the effect of orchestration from the behavior of complete systems. The reported six-source set contains 324 Controlled-core, 162 mechanism, and 540 Natural/Open-System assignments (1,026 in total), with three seeds per assigned task–method pair. Methods are evaluated on their designated tracks rather than forced onto every task.

Controlled core. Four arms run on the same 27 Controlled tasks: C-Incumbent retains the starting model; C-HPO uses the shared search substrate; C-Rule-HPO routes interventions by a fixed rule over the audited metrics; and C-Agent-RL-Dyn lets an LLM choose interventions and commit the final candidate. The three search arms share the frozen candidate grammar, tools, initial design, visible evidence, seeds, and 24-candidate ceiling; the incumbent is their common starting point. The task–seed pair is the unit of comparison. In particular, C-Agent-RL-Dyn versus

C-HPO tests the value of the controller within a matched search setting; the rule arm probes whether metric-driven routing requires an LLM.

Mechanism interventions. On 18 designated tasks, C-AGENT-ANON anonymizes feedback labels, C-AGENT-PRECOMMITTED follows a fixed, non-LLM intervention schedule, and C-AGENT-RESTART removes cross-turn dialogue history. These interventions test semantics, online allocation, and conversational state without changing the repair objective.

Algorithmic references and open systems. The 30 Natural tasks include incumbent reference, Random Search, Optuna TPE, Optuna NSGA-II, AutoGluon, and an AgentHPO-style sequential optimizer (Akiba et al., 2019; Deb et al., 2002; Erickson et al., 2020; Huai et al., 2026). They face the same task objectives and final-submission audit but retain their native search and modeling choices. The historical campaign uses DeepSeek-V4-Flash-CPDEV for its LLM-driven methods. Five further model–provider backends each rerun C-AGENT-RL-DYN (81 assignments), C-AGENT-ANON and C-AGENT-RESTART (54 each), and AgentHPO-style sequential search (90): 279 reported assignments per backend. The 747 non-LLM assignments remain shared controls, not five additional trials. Tasks, seeds, method policies, and task-level budgets remain fixed; model access and call-level settings follow each backend's registered interface. These are backend substitutions, not an isolation of model weights alone.

Outcomes and hierarchical aggregation. We report target and worst-constraint margins, joint production validity, and the number of assignments with a score. Visible is the primary score of the committed artifact on the WORKING audit; the private Gate check is separate, and Hidden is sealed post-development replay. Score coverage uses all assigned runs as its denominator. Margins and joint validity are computed only for runs with defined required measurements; execution failures and undefined measurements receive no invented scores. Within scored runs, Visible and Hidden method summaries average seed-level margins or joint-validity indicators within tasks, then average tasks within sources and weight sources equally. Controlled contrasts pair the same task and seed, and are reported only for pairs scored on both sides. The model-backend comparison places these conditional contrasts beside score coverage. Hidden evaluation loads the previously committed final artifacts or frozen prediction packets; it neither searches for a better candidate nor retrains the model. Failed and unavailable submissions remain in its coverage denominator.

6 RESULTS

The six-source bank contains 57 executable task cells. Among the 1,026 assignments in the reported original DeepSeek-V4-Flash-CPDEV set, 868 yield a visible-development score; 154 end in execution failure, three have an unavailable measurement, and one has no final record. A scoreable task is therefore not a guarantee of a scoreable submission. Table 3 reports the 864 assignments in the two primary tracks; the remaining 162 mechanism assignments appear in Appendix F.

6.1 MATCHED SEARCH AND FINAL SELECTION

All 57 task cells compile to executable score contracts. The incumbent is at target margin $m_{\text{target}} = -1$ by construction, so a less negative margin represents progress toward the aspirational goal; the worst-constraint margin is measured in MCID units. Crossing both zero boundaries is the stricter production outcome.

On the matched Controlled tasks, C-HPO has higher average target and worst-constraint margins than C-AGENT-RL-DYN (-0.489 versus -0.556, and -3.752 versus -4.362). The Agent, however, reaches joint validity in 6.7% of scored runs under source-equal aggregation, against 5.2% for HPO. Pairing the same tasks and seeds gives Agent–HPO differences of -0.067 goal units and -0.610 MCIDs; the 95% intervals, $[-0.173, +0.039]$ and $[-1.384, +0.164]$, both include zero. The fixed rule controller matches HPO's joint rate and trails it on both mean margins.

Natural/Open-System methods have lower conditional target margins than the incumbent reference and no scored joint-valid final. Coverage ranges from 8/90 for AgentHPO-style search to 85/90 for Optuna NSGA-II. These source-equal summaries are not paired comparisons.

Table 3: **Primary-track methods in the original visible-development campaign.** Scored/assigned reports final-model availability. Margins and joint validity are equal-source means among scored runs; higher is better. The 162 mechanism assignments are reported in Appendix F.

Method	Scored/ assigned	Target (m_{target})	Constraint ($m_{\text{constraint}}$)	Joint (%)
Controlled core (27 tasks, matched search substrate)				
C-INCUMBENT	81/81	-1.000	-12.346	0.0
C-HPO	81/81	-0.489	-3.752	5.2
C-RULE-HPO	81/81	-0.607	-4.484	5.2
C-AGENT-RL-DYN	81/81	-0.556	-4.362	6.7
Natural/Open-System track (30 tasks, native methods)				
Incumbent reference	90/90	-1.000	-13.874	0.0
Random Search	81/90	-2.557	-7.813	0.0
Optuna TPE	81/90	-1.974	-7.751	0.0
Optuna NSGA-II	85/90	-1.861	-7.934	0.0
AutoGluon	39/90	-1.382	-11.643	0.0
AgentHPO-style sequential	8/90	-11.227	-22.547	0.0

On the same 81 Controlled task–seed blocks, the Agent's WORKING archive contains a jointly feasible candidate in 22 runs, while its committed final is jointly valid in nine; the corresponding HPO counts are 16 and seven. These are stage-specific counts, not interchangeable scores. Appendix F gives the commitment audit and the mechanism contrasts: masking diagnostic labels and restarting conversation history change target margin by $+0.011$ and -0.017 relative to the dynamic controller, with both intervals including zero. The fixed-schedule C-AGENT-PRECOMMITTED control loses 0.111 target-margin units (95% CI $[-0.192, -0.030]$). It changes both search routing and final selection, so this is a contrast between complete commitment policies.

6.2 SEALED REPLAY CHANGES THE HISTORICAL COMPARISON

The same final artifacts are replayed on HIDDEN data without another search. Of the 1,026 reported original assignments, 843 yield a Hidden score. In the matched Controlled core, HPO and the dynamic Agent each score 79/81; their Hidden target margins are -0.610 and -0.670, and their worst-constraint margins are -4.450 and -5.120, respectively. Joint validity is nearly identical at 4.8% for HPO and 4.7% for the Agent. The Agent's small visible crossing advantage therefore does not carry over to this sealed replay. The full method-level Hidden results and paired Visible-to-Hidden changes are in Appendix F.

6.3 THE MODEL–PROVIDER BACKEND CHANGES QUALITY AND COVERAGE

Five further backends rerun the same 279 reported LLM-dependent assignments each, while non-LLM controls are reused. Table 4 separates three quantities: coverage across all four rerun methods, an Agent–HPO comparison on matched scoreable Controlled assignments, and the dynamic Agent's Hidden joint-validity rate. None is a substitute for the other two.

The Controlled contrast changes with the backend: on 79 jointly scored GPT-5.6-Sol pairs, the Agent leads HPO by 0.092 goal units and 0.912 MCIDs; both differences are negative on 72 GPT-5.6-Luna pairs. Visible coverage ranges from 110 to 220 of 279 reported assignments. Shared controls score 673/747 on Visible and 655/747 on Hidden, yielding 1,565 and 1,497 scoreable runs in the corresponding 2,142-assignment matrices. The Sol Agent's highest Hidden joint rate is conditional on a scoreable final.

6.4 OPEN-SYSTEM SUBMISSION IS NOT THE SAME AS REPAIR

In the original campaign, AgentHPO-style search scores 8/90 Natural assignments. With GPT-5.6-Sol, its visible/Hidden coverage rises to 43/40, but source-equal visible joint validity among scored runs

Table 4: **Backend substitutions.** V/H scored counts cover four LLM-dependent methods, with 279 reported assignments per backend on each split. Paired differences are C-AGENT-RL-DYN minus the shared C-HPO on jointly scored *visible* task–seed blocks, aggregated by source. The final column concerns only the Agent's scoreable HIDDEN submissions, not all four methods.

Backend	Scored V/H	Pairs	Δ target	Δ constraint	Agent Hidden joint (%)
DeepSeek-V4-Pro	203/194	80	$+0.015$	$+0.155$	7.9
GLM-5.2	110/96	56	$+0.005$	$+0.200$	5.3
GPT-5.6-Sol	220/212	79	$+0.092$	$+0.912$	12.1
GPT-5.6-Luna	165/154	72	-0.080	-0.710	4.1
Gemini-3.7-Flash	194/186	77	$+0.046$	$+0.640$	9.8

is only 0.8%. An evaluable artifact is not necessarily a valid repair; Appendix F separates these outcomes.

7 ANALYSIS AND DISCUSSION

The Controlled results separate average improvement from clearing every production boundary. HPO leads on both mean margins, yet the dynamic Agent submits nine jointly valid finals versus seven for HPO on Visible; that small difference disappears on sealed replay. The candidate archives contain more jointly feasible Working candidates than either method commits as jointly valid finals. This makes final commitment a substantive part of the repair problem, not a bookkeeping choice.

Joint validity is difficult across the portfolio. The nine Agent and seven HPO jointly valid Controlled finals all come from temporal-source tasks; neither method crosses both boundaries on static sources, and the scored Natural methods do not cross them either. Continuous margins still show how methods differ when the binary joint-validity indicator is zero. In the mechanism subset, masking diagnostic labels or clearing dialogue history has little effect on target margin, whereas the fixed-schedule policy loses more. Because that policy changes both routing and commitment, the experiment identifies the effect of the policy as a whole.

Backend substitution changes the comparison and the chance of obtaining a scoreable final. Agent–HPO margin differences are positive with GPT-5.6-Sol and negative with GPT-5.6-Luna on jointly scored Visible assignments. AgentHPO-style search also produces more scoreable submissions with Sol, but few jointly valid repairs. Score coverage and conditional quality answer different questions; reporting either alone would obscure part of the outcome.

8 CONCLUSION

CREDITAGENTBENCH evaluates whether a participant can improve an incumbent, preserve protected metrics, and commit a scoreable final model. In matched Controlled search, HPO leads on average margins; the Agent's small Visible lead in joint validity disappears on sealed replay. Natural methods and model–provider backends differ sharply in score coverage. A best archived candidate or a target gain alone would miss these parts of the repair outcome.

AI USE STATEMENT

Generative AI assisted with experimental design, code implementation, result interpretation, literature discovery, figure design and editing, and manuscript language editing. The authors reviewed and tested AI-assisted code, checked reported results against retained experiment artifacts, verified references, and take responsibility for the final methods, analyses, and text.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2623–2631. ACM, 2019. doi: 10.1145/3292500.3330701.

Miller Janny Ariza-Garzón, Mario Sanz-Guerrero, Javier Arroyo Gallardo, and Lending Club. Lending club loan dataset for granting models, 2024.

B Baesens, T Van Gestel, S Viaene, M Stepanova, J Suykens, and J Vanthienen. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society*, 54(6):627–637, 2003. doi: 10.1057/palgrave.jors.2601545.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24, pp. 2546–2554, 2011.

Glenn W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950. doi: 10.1175/1520-0493(1950)078⟨0001:VOFEIT⟩2.0.CO;2.

Niklas Bussmann, Paolo Giudici, Dimitri Marinelli, and Jochen Papenbrock. Explainable machine learning in credit risk management. *Computational Economics*, 57(1):203–216, 2021. doi: 10.1007/s10614-020-10042-0.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *The Thirteenth International Conference on Learning Representations*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794. ACM, 2016. doi: 10.1145/2939672.2939785.

Yutian Chen, Xingyou Song, Chansoo Lee, Zi Wang, Richard Zhang, David Dohan, Kazuya Kawakami, Greg Kochanski, Arnaud Doucet, Marc'Aurelio Ranzato, Sagi Perel, and Nando de Freitas. Towards learning universal hyperparameter optimizers with transformers. In *Advances in Neural Information Processing Systems*, volume 35, pp. 10557–10570, 2022.

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 *The Thirteenth International Conference on Learning Representations*, 2025.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017. doi: 10.1089/big.2016.0047.

Jonathan N. Crook, David B. Edelman, and Lyn C. Thomas. Recent developments in consumer credit risk assessment. *European Journal of Operational Research*, 183(3):1447–1465, 2007. doi: 10.1016/j.ejor.2006.09.100.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.

Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery. *CoRR*, abs/2107.07002, 2021.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Katharina Eggensperger, Philipp Müller, Neeratyoy Mallik, Matthias Feurer, René Sass, Aaron Klein, Noor H. Awad, Marius Lindauer, and Frank Hutter. HPOBench: A collection of reproducible multi-fidelity benchmark problems for hpo. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Michael T. M. Emmerich and André H. Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural Computing*, 17(3):585–609, 2018. doi: 10.1007/s11047-018-9685-y.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander J. Smola. AutoGluon-Tabular: Robust and accurate AutoML for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1437–1446. PMLR, 2018.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, volume 28, pp. 2962–2970, 2015.

FICO. Explainable machine learning challenge. Fair Isaac Corporation, 2018.

Jacob R. Gardner, Matt J. Kusner, Zhixiang Eddie Xu, Kilian Q. Weinberger, and John P. Cunningham. Bayesian optimization with inequality constraints. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 937–945. PMLR, 2014.

Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 250–259, 2014.

Pieter Gijsbers, Marcos L. P. Bueno, Stefan Coors, Erin LeDell, Sébastien Poirier, Janek Thomas, Bernd Bischl, and Joaquin Vanschoren. AMLB: an AutoML benchmark. *Journal of Machine Learning Research*, 25(101):1–65, 2024.

Ulrike Grömping. South german credit. UCI Machine Learning Repository, 2019.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017.

D. J. Hand and W. E. Henley. Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 160(3):523–541, 1997. doi: 10.1111/j.1467-985X.1997.00078.x.

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, volume 29, pp. 3315–3323, 2016.

Xin He, Kaiyong Zhao, and Xiaowen Chu. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, 2021. doi: 10.1016/j.knosys.2020.106622.

José Miguel Hernández-Lobato, Michael A. Gelbart, Ryan P. Adams, Matthew W. Hoffman, and Zoubin Ghahramani. A general framework for constrained bayesian optimization using information-based search. *Journal of Machine Learning Research*, 17(160):1–53, 2016.

Sirui Hong, Yizhang Lin, Bang Liu, Bangbang Liu, Binhao Wu, Ceyao Zhang, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Li Zhang, Lingyao Zhang, Min Yang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Robert Tang, Xiangtao Lu, Xiawu Zheng, Xinbing Liang, Yaying Fei, Yuheng Cheng, Yongxin Ni, Zhibin Gou, Zongze Xu, Yuyu Luo, and Chenglin Wu. Data Interpreter: An LLM agent for data science. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19796–19821. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1016.

Tianyu Huai, Tingshuo Fan, Xinchi Chen, Yining Zheng, Yuxin Wang, Shuang Chen, Jie Zhou, and Xuanjing Huang. AgentHPOBench: A benchmark for evaluating LLM agents as sequential hyperparameter optimizers. *arXiv preprint arXiv:2607.29626*, 2026.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20271–20309. PMLR, 2024.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization (LION 5)*, pp. 507–523. Springer, 2011. doi: 10.1007/978-3-642-25566-3_40.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024.

Sayash Kapoor and Arvind Narayanan. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns*, 4(9):100804, 2023. doi: 10.1016/j.patter.2023.100804.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, pp. 3146–3154, 2017.

Nikita Kozodoi, Johannes Jacob, and Stefan Lessmann. Fairness in credit scoring: Assessment, implementation and profit implications. *European Journal of Operational Research*, 297(3): 1083–1094, 2022. doi: 10.1016/j.ejor.2021.06.023.

Stefan Lessmann, Bart Baesens, Hsin-Vonn Seow, and Lyn C. Thomas. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1):124–136, 2015. doi: 10.1016/j.ejor.2015.05.030.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Antonio Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

Kaisa Miettinen. *Nonlinear Multiobjective Optimization*, volume 12 of *International Series in Operations Research & Management Science*. Springer, 1998. doi: 10.1007/978-1-4615-5563-6.

Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014. doi: 10.1016/j.dss.2014.03.001.

Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and Jason H. Moore. Automating biomedical data science through tree-based pipeline optimization. In *European Conference on the Applications of Evolutionary Computation*, pp. 123–137. Springer, 2016. doi: 10.1007/978-3-319-31204-0_9.

Liudmila Ostroumova Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6639–6649, 2018.

Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence (eds.). *Dataset Shift in Machine Learning*. The MIT Press, 2008.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5389–5400. PMLR, 2019.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, volume 25, pp. 2960–2968, 2012.

Ewout W. Steyerberg, Andrew J. Vickers, Nancy R. Cook, Thomas Gerds, Mithat Gonen, Nancy Obuchowski, Michael J. Pencina, and Michael W. Kattan. Assessing the performance of prediction models: A framework for traditional and novel measures. *Epidemiology*, 21(1):128–138, 2010. doi: 10.1097/EDE.0b013e3181c30fb2.

Lyn C. Thomas, David B. Edelman, and Jonathan N. Crook. *Credit Scoring and Its Applications*. SIAM, 2002. doi: 10.1137/1.9780898718317.

Aleksandr Tsymbalov, Danis Zaripov, Artem Epifanov, and Anastasiya Palienko. GRACE-DS: A guarded reward-guided agent correction environment in data science. *arXiv preprint arXiv:2606.16000*, 2026.

Ben Van Calster, David J. McLernon, Maarten van Smeden, Laure Wynants, and Ewout W. Steyerberg. Calibration: the achilles heel of predictive analytics. *BMC Medicine*, 17(1):230, 2019. doi: 10.1186/s12916-019-1466-7.

Chi Wang, Qingyun Wu, Markus Weimer, and Erkang Zhu. FLAML: A fast and lightweight AutoML library. In *Proceedings of the 4th Conference on Machine Learning and Systems*, pp. 434–447, 2021.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2024.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. InterCode: Standardizing and benchmarking interactive coding with execution feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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

I-Cheng Yeh and Che-hui Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, 36(2): 2473–2480, 2009. doi: 10.1016/j.eswa.2007.12.020.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023.

Shujian Zhang, Chengyue Gong, Lemeng Wu, Xingchao Liu, and Mingyuan Zhou. AutoML-GPT: Automatic machine learning with GPT. *arXiv preprint arXiv:2305.02499*, 2023.

Ethics and Reproducibility Statements

Ethics statement

This work evaluates model-development systems and does not make lending decisions about individuals. The benchmark uses documented research datasets and records source provenance, licensing, preprocessing, and split construction. The public data contain historical decisions and may encode social and institutional bias. We do not infer protected attributes that are absent from a source, and we do not present aggregate benchmark performance as evidence of fairness. Credit models directly affect access to financial services; benchmark performance therefore must not be interpreted as evidence that a submitted model is suitable for deployment. A lending institution would still need to assess fairness (Hardt et al., 2016; Chouldechova, 2017; Kozodoi et al., 2022), explainability (Bussmann et al., 2021), data-use permissions, and post-deployment monitoring under its own operating conditions; none of these follows from the benchmark scores.

Reproducibility statement

The appendix records the task portfolio, evaluation protocol, controller interfaces, and result accounting. The codebase provides commands for task preparation, execution, and collection; frozen manifests and assignment identifiers connect reported scores to their runs. Visible-development and HIDDEN results are collected separately. Source provenance and access conditions are documented in Appendix H.

A Additional Related Work

A.1 Machine Learning Engineering and Autonomous Coding Agents

Code generation alone does not capture an agent's ability to run experiments and use their results. Work on code synthesis (Chen et al., 2021), structured reasoning (Wei et al., 2022; Wang et al., 2023), tool use (Schick et al., 2023), and action–observation loops (Yao et al., 2023; Shinn et al., 2023) provides different ways to support such work. STOP (Zelikman et al., 2023) studies recursive code improvement.

Executable benchmarks then test different outcomes: SWE-bench checks issue resolution in repositories (Jimenez et al., 2024); InterCode tests interactive coding (Yang et al., 2023); Data Interpreter and ScienceAgentBench cover data-science and scientific workflows (Hong et al., 2025; Chen et al., 2025). MLAgentBench evaluates changes to research code against an initial baseline (Huang et al., 2024), while MLE-bench scores competition submissions (Chan et al., 2025). These settings establish useful execution tests, but do not all ask whether a repaired incumbent satisfies a separate set of protected metrics.

GRACE-DS also tests an agent's development process: hidden validators check leakage, reproducibility, and protocol validity, and a validation-based rule selects a candidate for one Hidden replay (Tsymbalov et al., 2026). These checks address genuine methodological risks (Kapoor & Narayanan, 2023). CREDITAGENTBENCH studies a different decision within an executable workflow: given an incumbent with an audited defect, can the participant choose and commit a repair that improves the target without crossing protected metric boundaries?

A.2 Automated Machine Learning and Hyperparameter Optimization Foundations

The Controlled track gives agentic and algorithmic controllers the same candidate space and evaluation opportunities. The Natural track additionally tests methods that retain their own modeling procedures. Both comparisons require credible HPO and AutoML baselines.

Tree-boosting systems such as XGBoost, LightGBM, and CatBoost are widely used for tabular prediction (Chen & Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018). Their behavior depends on preprocessing, model configuration, and the available search budget; none is a universally best backbone.

Black-box search methods take different approaches to a limited evaluation budget. Bayesian optimization uses a surrogate model (Snoek et al., 2012); SMAC uses a random-forest surrogate for mixed and conditional spaces (Hutter et al., 2011); and TPE compares densities over better and worse trials (Bergstra et al., 2011), with implementations including Optuna (Akiba et al., 2019). Hyperband allocates resources by successive halving (Li et al., 2018), BOHB combines that allocation with model-based search (Falkner et al., 2018), and NSGA-II maintains a population of non-dominated candidates for multiple objectives (Deb et al., 2002). These methods offer different controls for search, not a single performance ordering across tasks.

AutoML frameworks expand the decision space beyond one model's hyperparameters (He et al., 2021): auto-sklearn uses meta-learning and ensembles (Feurer et al., 2015), TPOT searches pipelines (Olson et al., 2016), FLAML emphasizes resource-aware search (Wang et al., 2021), and AutoGluon uses model stacking (Erickson et al., 2020). Ray Tune supports distributed tuning (Liaw et al., 2018); HPOBench and AMLB provide test suites for optimizer and AutoML comparisons (Eggensperger et al., 2021; Gijsbers et al., 2024). Unlike the Controlled arms, these systems need not use the same candidate grammar as the agent.

A.3 LLMs as Optimizers and Metaprogramming

Language models can also propose search steps. OptFormer learns from HPO histories (Chen et al., 2022); OPRO uses natural-language optimization prompts (Yang et al., 2024); FunSearch combines a language model with evolutionary program search (Romera-Paredes et al., 2024); and AutoML-GPT applies language models to pipeline design (Zhang et al., 2023). AgentHPOBench compares agents and conventional HPO on sequential interventions in executable repositories. Its main score uses the last intervention, and its results vary by task and setting; some agents exceed conventional methods on aggregate measures (Huai et al., 2026). CreditAgentBench instead asks whether search leads to a committed repair that satisfies both a target and protected production metrics.

A.4 Constrained Bayesian Optimization and Multiobjective Optimization

Optimizing one metric need not produce a deployable model when other metrics have minimum acceptable values. This is the decision represented by the repair target and protected constraints in CreditAgentBench.

Constrained Bayesian optimization studies a black-box objective $f(x)$ subject to constraints such as $c_j(x) \geq 0$. Existing approaches model feasibility when choosing the next evaluation (Gardner et al., 2014), allow unknown or separately evaluated constraints (Gelbart et al., 2014), or use information-based acquisition (Hernández-Lobato et al., 2016). CreditAgentBench does not propose another acquisition function; it tests whether a controller's final model clears a specified target while satisfying separate production boundaries.

Multiobjective optimization studies trade-offs between criteria (Miettinen, 1998; Emmerich & Deutz, 2018). A weighted sum can rank a candidate highly even when it misses a required bound on one metric. We therefore report target progress and the weakest protected-metric margin separately, and count a final model as jointly valid only if it crosses both boundaries.

A.5 Credit Risk Benchmarking and Financial Machine Learning

Credit modeling has long considered discrimination, calibration, and the costs of lending decisions together (Hand & Henley, 1997; Thomas et al., 2002; Crook et al., 2007). It is therefore a useful setting for tasks in which one metric should improve without sacrificing another, though passing this benchmark alone does not establish deployment fitness.

Prior credit-scoring studies compare many classifiers across datasets, including both scorecards and ensembles (Baesens et al., 2003; Lessmann et al., 2015). Public datasets have supported related work on default, credit decisions, marketing, and explanations (Yeh & Lien, 2009; Grömping, 2019; Moro et al., 2014; FICO, 2018; Ariza-Garzón et al., 2024). Our dataset admission and task construction are specified separately in Appendix B.

A.6 Algorithmic Fairness, Ethical Standards, and Generalization Fragility

Credit decisions also raise fairness and explanation questions. Equality of opportunity compares true positive rates across groups (Hardt et al., 2016); predictive parity and error-rate equality can conflict when base rates differ (Chouldechova, 2017). Studies in lending examine fairness–accuracy trade-offs (Kozodoi et al., 2022) and model explanations (Bussmann et al., 2021). CreditAgentBench does not treat its protected predictive metrics as a substitute for demographic fairness or legal review.

Dataset shift can change out-of-sample behavior (Quiñonero-Candela et al., 2008); fresh test sets and benchmark-design choices can also change reported comparisons (Recht et al., 2019; Dehghani et al., 2021). Interpretability requires its own evaluation questions (Doshi-Velez & Kim, 2017). CreditAgentBench includes retrospective temporal splits and a sealed Hidden replay to test some aspects of generalization, but neither procedure removes the need for later deployment audits.

B Task Bank Details and Dataset Specifications

The task bank combines static credit and bankruptcy records with retrospective and chronological lending cohorts. Lending Club and SBA each contribute three temporal origins, so the six sources yield ten frozen source keys. Three repair geometries are defined for each available source and task setting, producing 57 Natural and Controlled task cells; Berka has Natural cells only. Table 7 gives the exact coverage, and source-specific thresholds and split manifests are frozen with each task artifact.

B.1 Dataset Provenance and Source Portfolios

The six source portfolios differ in population, scale, and time semantics. Table 5 reports the source sizes used by the released preparation views. The historical-source column refers to the selected published file or release, whereas the admitted column is the post-preparation task view used by the benchmark. Table 6 gives the four disjoint evaluation roles for each source key; temporal keys cover different windows and need not each use every admitted row.

1. **UCI Default of Credit Card Clients (Yeh & Lien, 2009):** A static credit-card default dataset with $30,000$ clients and repayment-history features.
2. **South German Credit (Grömping, 2019):** A smaller static loan dataset with $1,000$ records and 20 predictive variables.
3. **Lending Club (Ariza-Garzón et al., 2024):** Retrospective consumer-loan records curated into three 60-month-outcome origins. These test chronological transfer but do not reconstruct a prospective underwriting snapshot.
4. **Polish Companies Bankruptcy:** A static, imbalanced company-bankruptcy dataset; its three-year outcome and financial-ratio predictors extend the repair task beyond retail credit.
5. **U.S. SBA 7(a):** Small-business loan records with three chronological origins and a 36-month outcome horizon. This is the portfolio's chronological-source suite.
6. **Berka PKDD'99:** A relational financial dataset prepared as one retrospective loan-risk origin. Its derived cohort is used for internal development, not presented as a public benchmark release.

B.2 Temporal Cohorts and Outcome Maturity

An unresolved loan is not a negative example. The preparation materializes four mutually exclusive roles: training, Working, Gate, and Hidden. Working supplies participant-visible development feedback. Gate is a private, preregistered audit of the committed model; participants cannot query it or receive its results while choosing a candidate. Hidden is reserved for sealed post-development replay. The Lending Club preparation requires a resolved 60-month outcome and separates these windows at each of three origins. SBA 7(a) uses a 36-month outcome with three later chronological origins. Berka supplies one retrospective origin. Each temporal task freezes its own maturity boundary and split dates before model evaluation. Lending Club and Berka are retrospective tests of shift; only the SBA suite carries the chronological-source designation in the frozen task bank.

Table 5: **Source scale and admission.** Historical rows count the selected published file or release (Berka: loans), not the full platform. Admitted rows, task-visible features, and event rates follow the frozen preparation contract.

Source	Historical rows	Admitted rows	Features	Event rate (%)
UCI Default Credit Card	30,000	30,000	23	22.12
South German Credit	1,000	1,000	20	30.00
Polish Bankruptcy (3-year)	10,503	10,503	64	4.713
Lending Club	1,347,681	230,702	9	15.647
SBA 7(a), 3-year	1,573,117	1,378,559	43	4.123
Berka PKDD'99	682	219	25	13.699

Table 6: **Rows per frozen source key.** Train, Working, Gate, and Hidden are mutually disjoint within a key. Lending Club and SBA keys are distinct temporal origins; Berka's key covers 179 of its 219 admitted rows.

Source key	Train	Working	Gate	Hidden
UCI Default	15,000	6,000	4,501	4,499
South German	500	200	150	150
Polish Bankruptcy	5,252	2,101	1,575	1,575
Lending Club r1	8,273	12,537	21,721	53,367
Lending Club r2	20,810	21,721	53,367	134,804
Lending Club r3	61,054	34,844	53,374	81,430
SBA 7(a) o1	190,132	75,392	82,630	89,663
SBA 7(a) o2	338,634	157,368	39,600	43,910
SBA 7(a) o4	871,528	79,292	57,907	49,980
Berka	17	80	49	33

B.3 Incumbent Defect Injection and Controlled Perturbations

Each CreditAgentBench task pairs a dataset split with an incumbent model M_0 exhibiting an audited operational vulnerability.

Natural incumbents. On Natural tasks, the incumbent is selected from a frozen candidate pool because its measured Working behavior leaves a target repair opportunity. No synthetic perturbation is applied.

Controlled mechanisms. On Controlled tasks, a pre-registered, label-free perturbation is applied to an incumbent's inference logits. Training rows and outcome labels remain unchanged:

1. **Calibration-Slope Perturbation (Logit Temperature Scaling):**

$$p_{\text{pert}}(x) = \sigma\left(\frac{\text{logit}(p(x))}{T}\right), \qquad T \in \{0.67, 1.50, 0.50, 2.00\}, \tag{5}$$

   where $\sigma(z) = (1 + e^{-z})^{-1}$. This strictly monotone transform preserves ROC AUC while changing probability calibration.

2. **Probability-Accuracy Perturbation (Logit Intercept Shift):**

$$p_{\text{pert}}(x) = \sigma\left(\text{logit}(p(x)) + b\right), \qquad b \in \{-0.75, +0.75, -1.50, +1.50\}. \tag{6}$$

   The intercept shift alters probability accuracy while preserving ranking.

3. **Discrimination Perturbation (Deterministic Feature-Hashed Noise):**

$$p_{\text{pert}}(x) = \sigma\left(\text{logit}(p(x)) + s \cdot \epsilon(x)\right), \qquad s \in \{1, 2, 4, 8\}, \tag{7}$$

   where $\epsilon(x) \in [-1, +1]$ is deterministic feature-derived hash noise. Unlike the first two transforms, it can change rank ordering.

The controlled generator is evaluator-side; its inverse is not a participant action in the shared search grammar.

B.4 TASK-CELL COVERAGE

The frozen bank contains 57 task cells: three repair geometries for each available source-key–world pair. The geometry suffixes are `calibration`, `discrimination`, and `probability`; each row in Table 7 specifies whether the corresponding `natural` and `controlled` IDs exist. For example, the UCI calibration cell is `cab-v6-uci-calibration-natural`. Berka has no Controlled cells. Task-specific aspiration and protected boundaries are part of the frozen task contracts, rather than a common threshold imposed on all sources.

Table 7: **Frozen 57-cell task bank.** Each nonzero world count includes all three repair geometries. Source keys and task IDs are frozen by the six-source campaign configuration.

Source	Task ID stem	Regime	Role	Natural	Controlled
UCI Default	`cab-v6-uci`	Static	Public	3	3
South German	`cab-v6-south`	Static	Public	3	3
Lending Club 1	`cab-v6-lc-r1`	Retrospective	Public	3	3
Lending Club 2	`cab-v6-lc-r2`	Retrospective	Public	3	3
Lending Club 3	`cab-v6-lc-r3`	Retrospective	Public	3	3
Polish bankruptcy	`cab-v6-polish-3y`	Static	Public	3	3
SBA 7(a) 1	`cab-v6-sba3-o1`	Chronological	Public	3	3
SBA 7(a) 2	`cab-v6-sba3-o2`	Chronological	Public	3	3
SBA 7(a) 4	`cab-v6-sba3-o4`	Chronological	Public	3	3
Berka	`cab-v6-berka`	Retrospective	Internal	3	0
Total	10 source keys	6 portfolios		30	27

C Score Coordinates and Metric Definitions

C.1 Target and Constraint Margins

CreditAgentBench reports target improvement and protected-metric validity as two coordinates $(m_{\text{target}}, m_{\text{constraint}})$; a gain in discrimination, for example, does not erase a calibration violation.

1. Signed Target Margin (m_{target}). Let $a_g(M)$ denote the adequacy of model M on the registered target metric g (higher is better: AUC, negative Brier score, or $-|\beta - 1|$ for calibration slope). The task's frozen bootstrap procedure produces a conservative lower endpoint $\underline{\Delta}_g(M)$ for improvement over the incumbent, using $B = 999$ resamples in V6. Given the required improvement $\delta_g > 0$, the target margin is normalized into goal units:

$$m_{\text{target}}(M) = \frac{\underline{\Delta}_g(M) - \delta_g}{\delta_g}. \tag{8}$$

By construction, the unmodified incumbent has $m_{\text{target}}(M_0) = -1.0$. A value of 0.0 signifies exact attainment of the aspirational improvement, and positive values indicate that the target has been exceeded. The signed distance to goal is $d_{\text{target}}(M) = \max(0, -m_{\text{target}}(M))$.

2. Minimum Signed Constraint Margin ($m_{\text{constraint}}$). Each protected production constraint $j \in \mathcal{C}$ defines an operational validity boundary b_j and a minimally consequential difference scale s_j (MCID). The individual margin for constraint j is

$$m_j(M) = \frac{\underline{a}_j(M) - b_j}{s_j}, \tag{9}$$

where $\underline{a}_j(M)$ is the conservative adequacy bound under the registered metric procedure. The task-level constraint outcome is governed by the non-compensatory minimum:

$$m_{\text{constraint}}(M) = \min_{j \in \mathcal{C}} m_j(M). \tag{10}$$

A candidate satisfies all protected constraints if and only if $m_{\text{constraint}}(M) \geq 0$. The continuous violation is $v_{\text{constraint}}(M) = \max(0, -m_{\text{constraint}}(M))$. If a required metric is undefined, the score is recorded as unavailable rather than replaced with zero.

C.2 Metric Definitions

The task contracts combine discrimination, probability accuracy, calibration, and distributional stability:

1. Discrimination (ROC AUC). Discrimination evaluates the model's ability to rank defaulting borrowers higher than non-defaulting borrowers across all classification thresholds. The Area Under the Receiver Operating Characteristic Curve (ROC AUC) is equivalent to the Wilcoxon–Mann–Whitney statistic:

$$\mathrm{AUC} = \frac{1}{N_+ N_-} \sum_{i:y_i=1} \sum_{j:y_j=0} \left(\mathbb{1}[p(x_i) > p(x_j)] + \frac{1}{2}\mathbb{1}[p(x_i) = p(x_j)] \right), \tag{11}$$

where N_+ and N_- are the counts of positive (default) and negative (non-default) borrowers, and $p(x)$ denotes predicted default probability. ROC AUC depends on the ranking of predictions and is invariant under strictly increasing transformations of their scores.

2. Probability Accuracy and Proper Scoring (Brier Score). The Brier score (Brier, 1950) measures probability accuracy as mean squared error:

$$\mathrm{Brier} = \frac{1}{N} \sum_{i=1}^{N} (p(x_i) - y_i)^2. \tag{12}$$

3. Probability Consistency (Calibration Slope). A model is calibrated when predicted probabilities match observed event frequencies. We assess calibration slope through logistic calibration (Steyerberg et al., 2010; Van Calster et al., 2019):

$$\mathrm{logit}(P(Y = 1 \mid p(x))) = \alpha + \beta \cdot \mathrm{logit}(p(x)), \tag{13}$$

Here α is the intercept and β the slope; $\beta = 1$ is ideal. A slope below one indicates predictions that are too extreme, while a slope above one indicates insufficient spread. The target adequacy is $a_{\mathrm{slope}}(M) = -|\beta - 1|$.

4. Score Distribution Stability (Kolmogorov–Smirnov Statistic). To flag changes in the overall predicted-score distribution, CREDITAGENTBENCH monitors the two-sample Kolmogorov–Smirnov statistic between out-of-fold development scores and the operational candidate distribution:

$$\mathrm{KS} = \sup_{z \in [0,1]} |F_{\mathrm{oof}}(z) - F_{\mathrm{cand}}(z)|, \tag{14}$$

where $F(z) = P(p(X) \leq z)$ is the empirical cumulative distribution function. Additional stability checks, where specified, use the task's frozen stress slices.

C.3 CALIBRATION AND THE CONTROLLED TEMPERATURE PERTURBATION

The shared candidate grammar offers an optional sigmoid calibration step. The controlled calibration task instead perturbs the incumbent with a fixed logit temperature,

$$p_T(x) = \sigma(\mathrm{logit}(p(x))/T), \qquad T > 0. \tag{15}$$

The distinction matters: the perturbation is part of task construction, not a free temperature-setting action available to the Controlled agent. It preserves the ranking of unclipped scores and hence their ROC AUC. On the same evaluation sample and without clipping, replacing $\mathrm{logit}(p)$ by $\mathrm{logit}(p)/T$ changes the fitted calibration slope from β to $T\beta$, not β/T.

D PARTICIPANT LIFECYCLE AND SEARCH SPACE

D.1 DEVELOPMENT AND FINAL COMMITMENT

The participant receives a Task Card, permitted training data, a frozen incumbent, protected metric boundaries, and a resource budget. It fits candidates using the task's allowed training split and requests feedback on the WORKING split. Controlled methods use the same candidate generator and choose among registered search actions; Open-System methods use their own declared participant interface. Each run ends with one selected final artifact or an explicit failure to submit.

The score attaches to that final choice. A stronger candidate left in the development archive does not replace a weaker submitted one. The HIDDEN split is reserved for post-development final evaluation; it is not shown during search or final selection.

D.2 Shared Controlled Candidate Grammar

The Controlled comparison fixes the candidate grammar and HPO substrate across its Agent, rule, and non-Agent arms. A recipe specifies imputation (`median` or `most_frequent`), feature selection (a registered percentile or rank pair), a model family, and optional `sigmoid` calibration. The two model families are regularized logistic regression and histogram gradient boosting. Logistic recipes choose standard or robust scaling, regularization strength, and class weighting; histogram-boosting recipes choose learning rate, leaf count, iteration count, and regularization. The typed ranges and conditional parameters are frozen in the versioned `credit-pipeline-grammar/v1` implementation. Native Open-System baselines are reported separately because their modeling procedures are not restricted to this grammar.

D.3 Task Admission

A task enters the released bank only after its source, split, incumbent, target headroom, protected metrics, and closed grader pass the scoreability checks. This establishes a measurable task contract, not a guarantee that every participant submission will yield defined numerical metrics.

D.4 Execution Outcomes

Within each reported method, every assignment remains in its denominator. We distinguish a valid final with a score, an execution or submission failure, and a final artifact whose required measurement is unavailable; neither kind of missing score is imputed. Hidden evaluation replays only committed finals that pass intake.

E Controllers and Mechanism Comparisons

E.1 Dynamic Agent Router

C-Agent-RL-Dyn receives the public Task Card, the current candidate archive, and participant-visible multi-metric feedback. Its actions are restricted to three search lanes—`balanced`, `target`, and `bottleneck`—or `commit` for an eligible candidate. A search action spends a model evaluation; a commit action ends the episode. The LLM returns a structured `action_id` and rationale, not executable code or a newly invented model recipe. The optimizer and typed candidate grammar are the same as in the non-Agent Controlled arm. The default agent retains its previous action-and-rationale history in addition to the current observation; no Hidden feedback is included.

E.2 Deterministic Rule Router

C-Rule-HPO uses the same action catalog without an LLM. It begins with a short, fixed pass through the three lanes, then reads the normalized margins derived from visible metric estimates. If its best visible candidate satisfies the target and all protected margins, it commits; otherwise it directs the next search toward the weakest target or constraint margin. The comparison holds the HPO machinery fixed while contrasting LLM routing with this deterministic policy.

E.3 Mechanism Subset

The 18 designated Controlled task cells support three focused comparisons. C-Agent-Anon replaces semantic metric and task identifiers with stable aliases while preserving numerical evidence, directions, and boundaries. C-Agent-Restart clears the LLM's cross-turn conversation history, but still receives the current benchmark observation and candidate archive; it does not erase evidence produced by prior searches. C-Agent-Precommitted follows a benchmark-frozen, feedback-invariant lane schedule and commits its last evaluated candidate. It makes no LLM call. It is therefore a fixed-schedule control, not an agent asked to write a plan at turn zero.

E.4 Other Baselines

Random search, Optuna TPE, and Optuna NSGA-II use the registered candidate space under their assigned budgets. The Natural/Open track also includes a fixed incumbent reference, native AutoGluon, and an AgentHPO-style sequential configuration proposer. These native methods face the same task contract and final-submission rule, but their modeling procedures are not claimed to be identical to the Controlled grammar. The frozen registry also includes a general-purpose coding agent (90 assignments in the original campaign and 90 per backend), which we leave out of the comparative baseline set because its open-ended workflow is not a task-oriented search policy. It produced no scoreable final with the original backend (0/90), though other backends produced scoreable submissions; its records remain in the archive and are excluded from the reported denominators.

F Additional Results and Reliability

The reported original six-source set contains 1,026 method–task–seed assignments. Table 8 separates a missing score from a failed task: the four visible records in the last column comprise three undefined measurements and one unfinished record. The hidden evaluator reused the selected final submissions and preserved execution failures; it did not search or retrain. Berka's reported runs use prepared data that are not part of the public release.

Table 8: **Coverage of reported methods in the original campaign.** Each count is an assigned run, not a distinct task. Visible failures and other outcomes partition the visible assigned count; hidden scores are a separate replay of the same assignments.

Source	Assigned	Visible	Hidden	Vis. failed	Vis. other
UCI Default	117	108	106	9	0
South German	117	103	99	14	0
Lending Club	270	234	229	36	0
Polish bankruptcy	117	109	107	8	0
SBA 7(a)	351	274	263	76	1
Berka	54	40	39	11	3
Total	1,026	868	843	154	4

Table 9 gives the hidden method-level result behind the main-text comparison. Means and joint-validity rates condition on scored finals; the scored/assigned column does not. A method without a scoreable final has no margin, not a zero margin.

Table 9: **Original DeepSeek-V4-Flash-CPDEV hidden replay.** Margins and joint validity are source-equal means among scored final submissions; larger is better. Tracks use different assigned task sets.

Method	Scored/assigned	Target	Constraint	Joint (%)
Controlled core				
C-INCUMBENT	80/81	−1.000	−13.200	0.0
C-HPO	79/81	−0.610	−4.450	4.8
C-RULE-HPO	78/81	−0.730	−5.150	4.4
C-AGENT-RL-DYN	79/81	−0.670	−5.120	4.7
Natural/Open-System track				
Incumbent reference	88/90	−1.000	−14.500	0.0
Random Search	76/90	−2.820	−8.640	0.0
Optuna TPE	80/90	−2.180	−8.360	0.0
Optuna NSGA-II	82/90	−2.060	−8.610	0.0
AutoGluon	39/90	−1.540	−12.180	0.0
AgentHPO-style sequential	6/90	−11.900	−23.600	0.0
Mechanism subset				
C-AGENT-ANON	52/54	−0.690	−5.030	4.3
C-AGENT-PRECOMMITTED	53/54	−0.780	−5.600	4.3
C-AGENT-RESTART	51/54	−0.730	−5.540	3.8

F.1 MATCHED CONTROLLED COMPARISONS

The original Controlled methods also show a loss between visible evaluation and sealed replay on *the same* submitted models. Table 10 uses only runs scored on both splits, so its paired differences need not equal a subtraction of the source-macro method means in the visible and hidden tables.

Table 10: **Original Controlled finals, Hidden minus Visible.** Paired task–seed blocks only; negative margins mean the same committed model is farther from the corresponding boundary on sealed data. Joint differences are percentage points.

Method	Pairs	Δ target [95% CI]	Δ constraint [95% CI]	Δ joint
C-HPO	79	−0.120 [−0.228, −0.012]	−0.700 [−1.450, +0.050]	−0.4
C-RULE-HPO	78	−0.123 [−0.275, +0.029]	−0.668 [−1.540, +0.204]	−0.8
C-AGENT-RL-DYN	79	−0.108 [−0.233, +0.017]	−0.752 [−1.620, +0.116]	−2.0

The mechanism interventions share the same 54-assignment subset. The dynamic controller is recomputed on these cells in Table 11, rather than compared with its full 81-assignment average. Masking the diagnosis (ANON) leaves the mean target margin nearly unchanged. Precommitment has the largest target-margin loss. These are mechanism contrasts within the fixed candidate grammar, not a ranking of all methods.

Table 11: **Visible mechanism subset.** Joint percentages are source-equal among scored finals; counts are raw assignments.

Method	Scored	Target	Constraint	Joint (%; count)
C-AGENT-RL-DYN	54/54	−0.583	−4.788	5.6; 6
C-AGENT-ANON	53/54	−0.572	−4.297	5.6; 5
C-AGENT-PRECOMMITTED	54/54	−0.694	−4.924	5.6; 6
C-AGENT-RESTART	53/54	−0.600	−4.828	4.6; 5

The archive-versus-final distinction is visible even with a shared search substrate. Table 12 counts runs whose WORKING archive contained at least one jointly feasible candidate and runs whose committed final was jointly valid on visible evaluation. A difference between the columns is not, by itself, a causal estimate of a selector defect: the archive and final evidence are measured at different stages.

Table 12: **Candidate discovery and final commitment in the original campaign.** All counts are runs, not distinct candidates.

Controlled method	Assigned	Archive feasible	Visible final valid
C-HPO	81	16	7
C-RULE-HPO	81	14	7
C-AGENT-RL-DYN	81	22	9
C-AGENT-PRECOMMITTED	54	14	6

The stratum breakdown in Table 13 keeps the three repair geometries separate from the static-versus-temporal source regime; the five rows overlap and must not be summed. Agent–HPO mean-margin differences are negative in all three geometries. The nine Agent and seven HPO joint-valid finals in the Controlled core occur on temporal-source tasks; neither method crosses both boundaries on the static-source tasks.

Table 13: **Original visible results by geometry and source regime.** Coverage of reported methods is scored/assigned. Agent–HPO differences and joint-valid counts use matched Controlled task–seed blocks within each stratum; larger margins are better.

Stratum	All-method coverage	Pairs	Δ target	Δ constraint	Joint A/H
Calibration repair	288/342	27	-0.050	-0.300	2/1
Discrimination repair	289/342	27	-0.070	-0.520	4/3
Probability repair	291/342	27	-0.082	-1.010	3/3
Static sources	320/351	27	-0.053	-0.572	0/0
Temporal sources	548/675	54	-0.089	-0.667	9/7

Table 14 expands the Controlled comparison by source. These rows are the source-level quantities used in the source-equal aggregation; the source rows are not additional tasks.

Table 14: **Controlled results by source.** Rows aggregate the task–seed blocks for each source in the matched Controlled substrate. Values are scored/assigned, source-level means, and joint-valid percentages; larger margins are better. Berka is not in this substrate.

Source	Method	Scored/assigned	Target	Constraint	Joint (%)
Lending Club	C-HPO	27/27	-0.388	-1.468	3.7
Lending Club	C-RULE-HPO	27/27	-0.493	-1.938	7.4
Lending Club	C-AGENT-RL-DYN	27/27	-0.448	-2.657	11.1
Polish bankruptcy	C-HPO	9/9	-0.666	-6.630	0.0
Polish bankruptcy	C-RULE-HPO	9/9	-0.677	-6.649	0.0
Polish bankruptcy	C-AGENT-RL-DYN	9/9	-0.554	-6.606	0.0
South German	C-HPO	9/9	-0.680	-5.838	0.0
South German	C-RULE-HPO	9/9	-0.829	-3.982	0.0
South German	C-AGENT-RL-DYN	9/9	-0.827	-3.838	0.0
UCI Default	C-HPO	9/9	-0.388	-5.450	0.0
UCI Default	C-RULE-HPO	9/9	-0.532	-10.400	0.0
UCI Default	C-AGENT-RL-DYN	9/9	-0.511	-9.189	0.0
SBA 7(a)	C-HPO	27/27	-0.321	$+0.626$	22.2
SBA 7(a)	C-RULE-HPO	27/27	-0.504	$+0.549$	18.5
SBA 7(a)	C-AGENT-RL-DYN	27/27	-0.439	$+0.481$	22.2

The mechanism means in Table 11 are complemented by the paired contrasts below. The dynamic controller is the reference on the same 54 assignments; intervals are source-equal paired 95% intervals.

F.2 FIVE MODEL–PROVIDER BACKEND SUBSTITUTIONS

Each backend reruns four reported LLM-dependent methods on 279 fixed task–seed assignments. The 747 non-LLM assignments are shared, not repeated. Table 16 shows why a single scored total obscures method-level behavior: GPT-5.6-Sol yields more scoreable AgentHPO-style submissions, while GLM-5.2 yields none for that method. The five backend cohorts jointly provide 892 scoreable LLM-dependent finals on visible development and 842 on hidden replay; after adding shared non-LLM controls, the full matrices have 1,565/2,142 and 1,497/2,142 scoreable assignments, respectively.

Table 15: **Paired mechanism contrasts on the visible mechanism subset.** Differences are variant minus the dynamic controller; larger values are better.

Variant	Pairs	Δ target [95% CI]	Δ constraint [95% CI]
C-AGENT-ANON	53	$+0.011\,[-0.052, +0.074]$	$+0.491\,[-0.163, +1.145]$
C-AGENT-PRECOMMITTED	54	$-0.111\,[-0.192, -0.030]$	$-0.136\,[-0.818, +0.546]$
C-AGENT-RESTART	53	$-0.017\,[-0.091, +0.057]$	$-0.040\,[-0.702, +0.622]$

Table 16: **Method-level score availability by backend.** Each entry is visible/hidden scored count. Assigned denominators are 81 for the dynamic controller, 54 each for the anonymized and restart controllers, and 90 for AgentHPO-style search.

Backend	RL-Dyn	Anon	Restart	AgentHPO
DeepSeek-V4-Pro	80/78	53/51	52/50	18/15
GLM-5.2	56/54	38/33	16/9	0/0
GPT-5.6-Sol	79/77	51/50	47/45	43/40
GPT-5.6-Luna	72/69	44/41	32/29	17/15
Gemini-3.7-Flash	77/75	48/46	41/39	28/26

For the paired contrasts in Table 17, the same task–seed block contributes to both methods only when both finals are scoreable. The 95% intervals resample paired task–seed blocks within each source and weight sources equally; we do not adjust for multiple comparisons. The five comparisons share the same fixed HPO result.

Table 17: **Dynamic Agent minus shared HPO on visible development.** Margins are source-equal paired differences. The last column gives raw joint-valid counts within the paired set (Agent/HPO), not source-equal percentages.

Backend	Pairs	Δ target [95% CI]	Δ constraint [95% CI]	Joint A/H
DeepSeek-V4-Pro	80	$+0.015\,[-0.058, +0.088]$	$+0.155\,[-0.380, +0.690]$	12/7
GLM-5.2	56	$+0.005\,[-0.088, +0.098]$	$+0.200\,[-0.490, +0.890]$	6/5
GPT-5.6-Sol	79	$+0.092\,[+0.005, +0.179]$	$+0.912\,[+0.240, +1.584]$	17/7
GPT-5.6-Luna	72	$-0.080\,[-0.165, +0.005]$	$-0.710\,[-1.440, +0.020]$	5/6
Gemini-3.7-Flash	77	$+0.046\,[-0.039, +0.131]$	$+0.640\,[-0.020, +1.300]$	14/7

Among the 1,395 reported LLM-dependent assignments across five backends, 503 lack a scoreable visible final. In the original campaign, visible Natural/Open coverage is 39/90 for AutoGluon-native and 8/90 for AgentHPO-style sequential search. Missing scores are reported as missing, not imputed as zero performance.

On the original matched 81 Controlled task–seed blocks, collected mean wall times were 1,091 seconds for C-HPO, 502 seconds for C-AGENT-RL-DYN, and 249 seconds for C-RULE-HPO. These times describe the original implementations and assigned run sets, not a normalized measure of search efficiency across the native Open-System methods.

G ILLUSTRATIVE TASK–SEED OUTCOMES

The following comparisons use final visible-development scores, not reconstructed agent dialogue. The task ID, method, track, and seed identify each assignment in the collected `assignments.jsonl`; target margins are in goal units and worst-constraint margins in MCID units. In both examples, a positive target margin means that the repair goal was reached, whereas a negative constraint margin means that at least one protected boundary was crossed.

Repair without validity. On `cab-v6-uci-calibration-controlled` (controlled core, seed 2026081301), C-HPO reaches a target margin of $+0.630$ but leaves the worst constraint margin at -2.413. The incumbent has margins of -1.000 and -12.096, respectively. The search therefore makes substantial progress on both axes without producing a jointly valid final submission. A conventional target-only score would obscure the remaining constraint violation.

Validity despite a smaller gain. On `cab-v6-lc-r3-calibration-controlled` (controlled core, seed 2026081301), C-HPO has the higher target margin, $+0.328$, but its worst constraint margin is -1.362. The final submission of C-AGENT-RL-DYN has a target margin of $+0.255$ and a worst constraint margin of $+0.359$, crossing both boundaries; C-RULE-HPO reaches the same two margins on this task–seed cell. This contrast illustrates why the target and protected margins must be read together.

H RUN PROVENANCE AND DATA SCOPE

H.1 FROZEN TASKS AND COLLECTION ENTRY POINTS

The frozen task-bank and experiment manifests identify the task cells, dataset keys, assignments, and backend variants. Each result collection retains assignment identifiers, selected result paths, terminal statuses, and its manifest digest, so a score can be traced to a task and run. The three seeds are 2026081301, 2026081302, and 2026081303. Backend substitution fixes the task, split, seed, method policy, and budget. Its assignment-level run options record the model and transport configuration, including call-level token and timeout settings where applicable.

The commands below launch and collect the original visible campaign, collect GLM-5.2 results, and inspect final-holdout status. They are entry points, not a self-contained reproduction package: running them requires frozen data and task-bank artifacts, a compatible Python environment, and credentials for remote backends. Final-holdout evaluation separately replays frozen final artifacts or prediction packets; it does not call participants or retrain models.

```
export CAB_V6_PYTHON=/path/to/compatible/python
./scripts/task_bank_six_source_v6.sh status
./scripts/v6_six_source_experiment.sh prepare
./scripts/v6_six_source_experiment.sh preflight
./scripts/v6_six_source_experiment.sh resume --workers 4
./scripts/v6_six_source_experiment.sh collect
./scripts/v6_antchat_experiment.sh collect --run-id glm52-t900-v1
./scripts/v6_final_hidden.sh status --output-dir results/final-hidden-original-glm-v1
```

H.2 SOURCE PROVENANCE AND USE

The six source families are UCI Default Credit Card, South German Credit, Lending Club, Polish Companies Bankruptcy, U.S. SBA 7(a), and Berka/PKDD'99. Appendix B and the frozen dataset manifests document their preprocessing, feature policies, and split construction. The reported Berka tasks use internally prepared files that are not part of the public release.

H.3 INTERPRETATION

The target and protected metrics operationalize a bounded model-repair problem. A valid benchmark result is evidence about those specified metrics and splits, not a certification of fairness, legal compliance, or readiness for credit deployment. The benchmark does not replace institution-specific monitoring or human model-risk review.