INFORMATION-GUIDED ACTIVE SAMPLING FOR AUTONOMOUS SKILL EVOLUTION IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Model (LLM) agents fail on complex, domain-specific operations that require heuristics absent from pretraining data. Skill-file self-evolution addresses this by iteratively rewriting a skill file from failure analysis, but existing methods sample training instances uniformly at random, wasting budget on redundant or uninformative failures. We propose **Information-Guided Active Sampling (IGAS)**, which casts instance selection as active learning over a non-parametric (text) skill: a multiplicative acquisition utility combining a failure-signal factor (failure similarity shaped by an entropy peak) with task representativeness, and utility-weighted facility-location batch selection under a monotone submodular coverage objective with a $(1 - 1/e)$ greedy guarantee. Because a purely informative sampler drifts toward atypical instances whose improvements do not transfer, we introduce ε**-anchored mixing** and a re-verification quota. Our evaluation protocol uses a four-way split (Train / gating Val_A / final-selection Val_B / Test) with strict gated acceptance ($\Delta > 0$), a cross-round rejection memory, and a single post-hoc Test evaluation; it also exposes an **inflation-accumulation phenomenon** in single-validation-set selection, which our Best-of-N final selection counters through same-window dual evaluation and a paired-bootstrap discount. We evaluate on six datasets (LiveMathematician-Bench, SearchQA, DocVQA, SpreadsheetBench, ALFWorld, OfficeQA) under a **preregistered controlled two-arm design**—the original SkillOpt baseline versus our full method—with 10 seeds per dataset per arm (120 runs in total). **[PLACEHOLDER: headline results pending the 10-seed data: per-dataset two-arm Test scores (mean $\pm$ std), paired differences, bootstrap 95% CIs, and Wilcoxon p-values.]**

1 INTRODUCTION

Large Language Model (LLM) agents fail on complex, domain-specific tasks—spreadsheet manipulation, data pipeline orchestration, database administration—because the required heuristics (implicit rules about error handling, format conventions, and tool quirks) are absent from pretraining data. A common remedy is a **skill file**: a markdown document of domain instructions and error-recovery strategies, traditionally hand-authored at great cost. **Skill-file self-evolution** instead evolves the file autonomously: each round the agent attempts training tasks, analyzes failures, and merges improvements into the file.

A critical bottleneck is **instance selection**: which tasks to attempt each round? Existing methods sample uniformly at random—suboptimal for three reasons: *redundancy* (instances similar to solved ones carry no new signal), *difficulty mismatch* (trivial or far-too-hard instances yield little actionable diagnosis), and *no history awareness* (rounds sample independently, ignoring earlier failure patterns).

We propose **Information-Guided Active Sampling (IGAS)**, which casts instance selection as active learning over a non-parametric (text) skill: a failure-signal factor $\hat{u} \cdot h(\hat{u})$ (failure similarity shaped by a binary-entropy peak around $\hat{u} \approx 0.70$, where diagnosis is most informative) combined multiplicatively with log-scaled representativeness, $\mathrm{Utility}(i) = \hat{u} \cdot h(\hat{u}) \cdot (1 + \log_2(1 + \mathrm{repr}))$; batches are chosen by a utility-weighted facility-location greedy (Section 3.3) that credits a candidate only for the utility mass it *newly covers*, over both the tried set and the current batch.

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.

Our contributions:

1. We formalize instance selection in skill-file self-evolution as active learning over a non-parametric (text) skill, with the acquisition utility above built from two complementary signals—failure signal peaking at $\hat{u}_i \approx 0.70$ and log-scaled representativeness—plus an optional UCB bonus (a removed centroid-distance novelty multiplier: Appendix B.0.1).
2. We propose **utility-weighted facility-location batch selection** for the monotone submodular coverage objective (Section 3.3.1), carrying a $(1 - 1/e)$ greedy guarantee with no diversity hyperparameter and folding cross-round redundancy control into the same objective (Theorem 1).
3. We design a **four-way evaluation protocol** (Train / gating Val_A / final-selection Val_B / Test) with strict gated acceptance ($\Delta > 0$), cross-round rejection memory, and a single post-hoc Test evaluation; we identify an **inflation-accumulation phenomenon** in single-validation-set selection, countered by Best-of-N final selection (Section 3.4).
4. We evaluate on six datasets (Section 4.1.1) under a preregistered controlled two-arm design—the original SkillOpt baseline (uniform sampling, merged validation) versus the full method (IGAS + four-way split + strict gating + BoN)—with 10 seeds per dataset per arm (120 runs; Sections 4.1.4 and 4.1.2). **[PLACEHOLDER: headline numbers pending the fullbench 10-seed data—per-dataset two-arm Test scores (mean $\pm$ std), IGAS$-$SkillOpt paired differences, paired percentile-bootstrap 95% CIs, and Wilcoxon signed-rank p-values.]**

2 Related Work

LLM agent skill learning. In-context approaches store successful trajectories for retrieval (Shinn et al., 2023; Zhao et al., 2024), and fine-tuning approaches train on agent trajectories (Chen et al., 2023; Zeng et al., 2023) at significant compute cost. Closest to us, skill-file approaches maintain explicit skill documents updated from agent performance: Trace2Skill (Ni et al., 2026) consolidates parallel patches from trajectories, and SkillOpt (Yang et al., 2026) treats skill documents as trainable state under bounded edits and validation gates. These address *skill synthesis* (how to extract and update skills); we address *instance selection* (which tasks to attempt each round)—complementary, with IGAS a plug-and-play front-end for any synthesis method (Section 4.3.4). Unlike inference-time self-improvement such as Reflexion (Shinn et al., 2023), Voyager's skill library (Wang et al., 2023), and tree-search agents (Zhou et al., 2024), skill-file self-evolution produces persistent, human-readable, transferable skill files with no inference-time overhead.

Active learning and failure clustering. Active learning (Settles, 2009; Gal et al., 2017; Sener & Savarese, 2018) and Bayesian experimental design (Lindley, 1956; Chaloner & Verdinelli, 1995) maximize information gain but assume a parametric model trained by gradients; because our "model" is a text skill file, we use similarity-based estimates in an embedding space. For failure diagnosis we use agglomerative clustering with cosine distance on SBERT embeddings (hash TF-IDF fallback), deterministic and consistent with the IGAS kernel matrix, in the spirit of density- and exemplar-based clustering (Campello et al., 2013; Frey & Dueck, 2007).

3 Method

3.1 Overview

Our framework consists of an iterative evolution loop with six components (Figure 1):

1. **Active Sampling**: Select training instances maximizing acquisition utility (Section 3.3).
2. **Trace Generation**: The agent attempts selected instances using the current skill file.
3. **Failure Clustering**: Failed traces are embedded and clustered by failure pattern (Section 3.2).
4. **Cluster-Level Diagnosis**: An LLM diagnoses each cluster and proposes skill improvements.

Figure 1 — Six-component evolution loop with cross-round Failure Pattern Memory, followed by Best-of-N final selection over the candidate pool and a single test evaluation.

Figure 1: Overview of the cluster-driven skill-evolution framework with Information-Guided Active Sampling.

5. **Candidate Gating**: Candidates must strictly outperform the current skill on Val_A ($\Delta > 0$) or are rolled back into a cross-round rejection memory (Section 3.4.1).
6. **Skill Merging**: Approved proposals are merged into the skill file for the next round.

A **Failure Pattern Memory** (Section 3.3.7) persists across rounds, feeding diagnosis and re-verification. After all rounds, Best-of-N selection (Section 3.4.2) ranks the candidate pool by Val_B with same-window Val_A as an overfitting probe; the winner is evaluated exactly once on Test (Section 3.4).

3.2 Cluster-Driven Failure Analysis

Feature extraction and clustering. Each failed trace is embedded as a text summary (instruction, error messages, tool calls, output, failure reason) with SBERT (`all-MiniLM-L6-v2`, local; hash TF-IDF 256-dim fallback), then grouped by **agglomerative clustering with cosine distance** on L2-normalized embeddings: deterministic, zero LLM calls, consistent with the IGAS kernel matrix L; the distance threshold τ_{dist} (default 0.7) controls granularity. Singleton clusters are legitimate: isolated trajectories are retained as long-tail representatives, with the diagnosis budget controlled by top-$K_{\max}$ truncation (Section 3.3.7). Per cluster we compute the centroid, common errors and common tools (frequency >50% of members), and a human-readable label.

Cluster-level diagnosis and skill update. For each cluster, an LLM receives common error patterns, medoid trace summaries, and the current skill content, then generates targeted fixes as executable code templates, following Ni et al. (2026)—this mechanism is not a contribution. Two safeguards prevent degradation: a length constraint (80%–150% of the original), and rollback into a cross-round persistent rejection memory if strict gating $\Delta > 0$ on Val_A fails (Section 3.4.1). Our contribution is the instance selection strategy (Section 3.3), orthogonal to skill updates.

3.3 Information-Guided Active Sampling

3.3.1 Problem Formulation

Let $\mathcal{D}$ be the training instances, $\mathcal{F}_t$ the instructions failed in rounds $1, \ldots, t$, $\mathcal{C}_t$ the failure-pattern centroids through round t, and $\mathcal{T}_t$ the tried instances. **Problem (Skill-Evolution Instance Selection).** At round $t+1$, select $S_{t+1} \subset \mathcal{D}_{\mathrm{untried}}$ of size n maximizing batch diagnostic value: instances likely to fail in informative ways whose diagnosis generalizes to other untried tasks. The per-instance acquisition score is

$$\mathrm{Utility}(i \mid \mathcal{F}_t) = \hat{u}_i \cdot h(\hat{u}_i) \cdot \left(1 + \log_2\left(1 + \mathrm{repr}(i)\right)\right), \tag{1}$$

optionally augmented by the UCB exploration bonus (Section 3.3.5), where $\hat{u}_i \in [0,1]$ is the failure-similarity score (Section 3.3.2), $h(\cdot)$ binary entropy, and $\mathrm{repr}(i)$ the representativeness count (Section 3.3.4). Independent scoring cannot prevent within-batch duplicates; redundancy control is therefore folded into a utility-weighted facility-location objective,

$$f_{\theta_t}(S) = \sum_{j \in \mathcal{D}_{\mathrm{untried}}} U_{\theta_t}(j) \cdot \max_{i \in S \cup \mathcal{T}_t} \mathrm{sim}(i,j), \tag{2}$$

where $U_{\theta_t}(j)$ is the utility of Eq. (1) and $\mathrm{sim}(i,j)$ cosine similarity over L2-normalized embeddings: a batch is credited for the utility mass of untried instances it *covers*, with $\mathcal{T}_t$-initialization carrying cross-round redundancy control. Exhaustive search is infeasible; greedy is justified by the following guarantee.

Theorem 1 (Per-Round Approximation Guarantee)**.** *For any fixed skill file θ_t and non-negative utility weights $U_{\theta_t}(j) \geq 0$, the objective of Eq. (2) is monotone submodular in S; greedy selection*

of the highest marginal gain achieves a $(1 - 1/e)$ *approximation under a cardinality constraint* $|S| \leq n$ *(Nemhauser et al., 1978).*

Proof and scope discussion are deferred to Appendix B.0.1.

Design rationale. Diagnostic signal is only available when an instance fails, so the two axes form an AND condition that additive forms would violate; the entropy-shaped factor concentrates budget near the failure boundary—likely to fail, yet not in an already-diagnosed way. The utility is a **heuristic diagnostic value score**, not a Bayesian expected information gain; the multiplicative form is validated in Appendix B.7. Non-attributable failures (network timeouts, API errors, environment crashes) carry zero diagnostic value and are excluded from $\mathcal{F}_t$ in preprocessing.

3.3.2 Failure Similarity Scoring (Temperature-Calibrated)

We compute a **failure similarity score**—a heuristic uncertainty indicator, not a calibrated probability—by temperature-scaled kernel-weighted aggregation over similarities to past failures:

$$\hat{u}_i = \frac{\sum_j w_j \cdot K_\tau(\mathrm{emb}(i), \mathrm{emb}(f_j)) \cdot \cos(\mathrm{emb}(i), \mathrm{emb}(f_j))}{\sum_j w_j \cdot K_\tau(\mathrm{emb}(i), \mathrm{emb}(f_j))}, \qquad K_\tau(u, v) = \exp\left(\frac{\cos(u, v)}{\tau}\right), \tag{3}$$

with temperature $\tau = 0.1$, clipped to $[0, 1]$; absent past failures we default to $\hat{u}_i = 0.5$. The estimator follows Nadaraya-Watson kernel regression (Nadaraya, 1964; Watson, 1964), using cosine similarity as a continuous proxy for the failure indicator—avoiding binary labels, which depend on the current skill file and are non-stationary. The product $\hat{u}_i \cdot h(\hat{u}_i)$ (binary entropy $h(x) = -x \log_2 x - (1 - x) \log_2(1 - x)$) is a soft "failure boundary" detector.

Temporal decay. Otherwise historical failure embeddings form an "echo chamber" of stale, over-estimated uncertainty. Each failure is weighted by $w_j = \gamma^{\,t_{\text{current}} - t_j - 1}$ with $\gamma = 0.8$ (half-life $\sim$3 rounds); formal consistency analysis remains future work.

3.3.3 Cross-Round Redundancy Control: From Novelty to Coverage Handoff

Our first variant embedded cross-round redundancy control in the utility as a centroid-distance novelty factor, $1 - \max_{c \in \mathcal{C}_t} \cos(\mathrm{emb}(i), \hat{c})$; this static prior was dwarfed by the failure-similarity factor's dynamic range and boosted peripheral instances. The current design moves the control into the objective: coverage of every untried j is initialized from the tried set as $m_j = \max_{s \in \mathcal{T}_t} \mathrm{sim}(s, j)$ and enters the marginal gain (Section 3.3.5) as $[\,\mathrm{sim}(i, j) - m_j\,]_+$—utility mass already covered by a *tried* instance contributes zero marginal gain. This is the standard lazy-greedy treatment of facility location (Cornuéjols et al., 1977; Nemhauser et al., 1978); our contribution is using it to unify cross-round and within-round redundancy control. The centroid library $\mathcal{C}_t$ is still maintained for *diagnosis* and re-verification ranking (Sections 3.3.7 and 3.3.7) but no longer enters the sampling path.

3.3.4 Representativeness (Log-Scaled)

Representativeness counts the local density of similar untried instances, $\mathrm{repr}(i) = |\{j \in \mathcal{D}_{\text{untried}} \setminus \{i\} : \cos(\mathrm{emb}(i), \mathrm{emb}(j)) > \tau\}|$ with threshold $\tau = 0.7$. We apply log-scaling: $f(\mathrm{repr}) = 1 + \log_2(1 + \mathrm{repr})$.

Theoretical justification. The log transformation captures *diminishing returns*; the "1+" keeps unique instances ($\mathrm{repr} = 0$) at baseline; scaling is motivated by Hartley entropy.

Utility weights vs. coverage. repr and the coverage of Section 3.3.5 are not redundant: in facility location (Cornuéjols et al., 1977) the distinction between *node weights* w_j (value of serving j) and *coverage* (whom a facility serves) is standard. $U(j)$ of Eq. (1) is exactly the node weight—per-instance diagnostic value—while the coverage term $\max_{i \in S \cup \mathcal{T}_t} \mathrm{sim}(i, j)$ determines how much of it a batch captures: structurally distinct, no double counting. The repr threshold is hard, but the weight is computed once per round and enters as a non-negative node weight, so threshold noise can neither affect monotone submodularity nor turn near-duplicate rankings into near-duplicate selections.

3.3.5 UTILITY-WEIGHTED FACILITY-LOCATION BATCH SELECTION

Core algorithmic contribution. Rather than top-n by score, we maximize the *monotone submodular facility-location objective* of Eq. (2) over the untried pool, with $U_{\theta_t}(j)$ the utility of Section 3.3.1 (optionally with the exploration bonus below). Writing $m_j = \max_{s \in \mathcal{T}_t \cup S} \mathrm{sim}(s, j)$ for the current coverage of j, the marginal gain of adding i is

$$\Delta(i \mid S) = \sum_{j \in \mathcal{D}_{\mathrm{untried}}} U(j) \cdot [\,\mathrm{sim}(i, j) - m_j\,]_+ , \tag{4}$$

so only *residual* coverage counts—re-sampling near-duplicates of tried instances is zeroed by the $[\cdot]_+$ truncation. Greedy thus carries the $(1-1/e)$ guarantee of Theorem 1, maintaining the coverage vector m and computing all marginal gains as one matrix–vector product per step at $O(n \cdot N^2)$ total cost—under 5% of per-round runtime at our batch sizes ($n \leq 50$).

Exploration bonus. By default we add a UCB1-style bonus, $u(i) \leftarrow u(i) + \beta\sqrt{\ln(N_t + 1)/(\mathrm{repr}(i) + 1)}$ with $\beta = 0.3$ (N_t the untried-pool size), so instances with few near neighbors are occasionally selected despite weaker failure signals. The bonus is folded into $U(j)$ before selection, so Theorem 1 applies unchanged; it can be disabled ($\beta = 0$, ablation X3; validated in Section 4.3, hyperparameters in Appendix B.5).

3.3.6 ε-ANCHORED MIXING: A DISTRIBUTION GUARD FOR INFORMATIVENESS-DRIVEN SAMPLING

Pure informativeness-driven sampling over-samples *atypical* instances, yielding skills that fail to transfer to the typical instances dominating the evaluation distribution—observed in pre-anchoring runs. Each round's exploration quota n_{explore} (net of the re-verification share, Section 3.3.7) is therefore split into an anchored share $n_{\mathrm{anchor}} = \lfloor \varepsilon \cdot n_{\mathrm{explore}} \rceil$ (uniform random sampling) and an IGAS share $n_{\mathrm{igas}} = n_{\mathrm{explore}} - n_{\mathrm{anchor}}$ (facility-location selection), With $\varepsilon = 0.25$ a quarter of each round's exploration budget tracks the population distribution, and any skill improvement must generalize across both shares to survive gating. Anchoring is also conservative: the IGAS arm stays closer to the random baseline than pure-IGAS sampling would, so the measured two-arm gap (Section 4.2) is a lower bound. Anchored items still enter the same clustering and diagnosis pipeline.

3.3.7 RE-VERIFICATION QUOTA: CLOSING THE LOOP ON FAILED INSTANCES

Skill evolution has a closure problem: an update motivated by a round-t failure is never re-measured on that instance, conflating "cannot solve this" with "could not solve this *at diagnosis time*". We therefore reserve a fraction $\rho = 0.25$ of each round's quota for *re-verification*—re-attempting historically failed instances—gated by two hard conditions: acceptance ($\Delta > 0$) *after* the instance was last attempted, and at least 2 rounds since. Eligible instances are ranked by soft maximum similarity to *active* failure-mode centroids (Section 3.3.7), prioritizing still-live failure modes. Re-verified instances now solved provide the cleanest causal signal that a skill update fixed its target failure mode; persistent failures re-enter the pattern memory.

Failure pattern memory. We maintain a pattern memory storing, per cluster, a centroid, a human-readable label, first/last-seen rounds, trace count, and common-error/common-tool sets (schema in Appendix B.1). A new cluster merges into an existing pattern when cosine similarity exceeds $\tau_{\mathrm{merge}} = 0.8$, otherwise it spawns a new entry. **Temporal decay:** a pattern unobserved for more than 2 rounds carries weight $w(c) = 0.9^{\,r_{\mathrm{now}} - r_{\mathrm{last}} - 2}$ (1 otherwise) for re-verification ranking, dropping from the active set below 0.01—mirroring Section 3.3.2's instruction-level decay at pattern granularity.

The memory serves three purposes: (i) cluster-level diagnosis (Section 3.2, subject to pruning below); (ii) re-verification ranking via decay-weighted active centroids (Section 3.3.7); and (iii) convergence tracking. The uncertainty estimator $\hat{u}$ (Section 3.3.2) consumes the same failure history at *instruction* granularity ($\mathcal{F}_t$), not the memory's centroids.

Top-K cluster pruning. We diagnose only the top $K_{\max} = 5$ clusters by size, halving per-round LLM diagnosis calls while preserving coverage of dominant failure modes (Section 4.3). The $(1 - 1/e)$ guarantee applies only to *sampling*: *all* failure traces, including pruned clusters', enter $\mathcal{F}_t$ as kernel-regression support points for $\hat{u}$.

Table 1: Four-way data splits for the main (fullbench) experiment. Both arms share the same physical split; the rightmost column shows the SkillOpt baseline's merged validation set ($\text{Val}_A \cup \text{Val}_B$).

Dataset	Train	Val_A (gating)	Val_B (final sel.)	Test	SkillOpt merged Val
LiveMathematicianBench	426	8	49	125	57
SearchQA	400	100	100	1400	200
DocVQA	107	25	28	374	53
SpreadsheetBench	80	20	20	280	40
ALFWorld	39	9	9	134	18
OfficeQA	50	12	12	172	24

3.4 Data Splitting, Gating, and Final-Stage Selection

Skill self-evolution involves two decision classes—*in-round* (whether to accept a candidate skill) and *final-stage* (which candidate to deploy). If both share one validation signal, selection pressure repeatedly mines its monotone noise, producing inflation accumulation (Section 3.4.2). We therefore split each fullbench dataset into four segments, pre-materialized as fixed split directories with frozen `env.split_seed=42` and **shared physically by both arms** (Table 1): the IGAS arm uses the four-way split, while the baseline retains SkillOpt's original three-way protocol, merging Val_A and Val_B into one validation set serving both gating and final selection—exposing its final-selection signal to in-round selection pressure, precisely the contrast the two-arm design measures. Segment responsibilities are strictly separated: **Train** is the sole sampling source (Section 3.3); **Val**$_A$ the sole in-round gating signal (Section 3.4.1), doubling as the overfitting probe A′; **Val**$_B$ drives Best-of-N selection (Section 3.4.2), never consulted during evolution; **Test** is evaluated exactly once, on the final-selection winner—keeping Test genuinely blind.

Group-level leakage prevention. Random instance-level splits carry leakage risk for task families that share context. **[PLACEHOLDER: The group-level isolation facts of each fullbench dataset's materialized split are pending verification—for each dataset, confirm whether document-level / table-level / scene-level grouping isolation is imposed (e.g., multiple questions on the same OfficeQA document, near-duplicate question pairs sharing the same retrieval target in SearchQA), and provide the isolation granularity plus zero-overlap evidence.]**

3.4.1 Candidate Gating: The Strict Criterion

Each round's diagnosis produces a candidate θ'; the gate evaluates it and the current skill on Val_A, $\Delta = \text{score}(\theta', \text{Val}_A) - \text{score}(\theta_t, \text{Val}_A)$. The **acceptance criterion** is strict: accept only when $\Delta > 0$ (bootstrap round: $\Delta \geq 0$); the candidate enters the pool and becomes the next round's baseline. On rejection the system rolls back to θ_t and writes the proposal into a **rejection memory**—a cross-round blacklist steering subsequent diagnoses away from falsified directions. We retired the Metropolis-style annealed acceptance of an earlier version: replay analysis showed it admitted mostly deteriorating candidates (4 of 6 acceptances in an 8-round run; one survivor scored only 0.510 on test, the worst in the field), and even a deteriorated candidate crossing the gate is eliminated by final-stage Val_B ranking. Both noise-induced error directions on finite Val_A are likewise harmless: an acceptance is backed up by the unbiased final-stage B ranking; a rejection merely keeps the previous skill and feeds the rejection memory—small-Val_A quantization noise is absorbed by the "multi-round runs + final-stage selection" architecture, so Val_A can be as small as 100–200. For cost control, a cross-round urgency value (incremented by $1 - \text{sim}(\theta_{\text{old}}, \theta_{\text{new}})$ per rewrite, 0.5 residual after rejections) lets trivial amendments skip verification, and the current skill's Val_A score is cached—each round's incremental gating cost is one evaluation of the new candidate.

3.4.2 Final-Stage Best-of-N Selection: The Combined Criterion

The strict gate plus a single validation set carries a structural risk: accepting on $\Delta > 0$ each round repeatedly mines the positive sampling noise of Val_A, so the in-round Val_A score of the i-th accepted candidate carries a cumulative selection offset growing with chain position r that does not materialize on test—**inflation accumulation** (evidence in Section 4.4.1). After all rounds, the K pooled candidates are evaluated back-to-back on $B(\text{Val}_B)$ and $A'(\text{Val}_A)$ *within the same time window*: server-side model drift (observed range 0.022) becomes a common offset that cancels in cross-candidate

rankings. For candidate i define the inflation gap $\mathrm{gap}_i = A'_i - B_i$ and its chain-head-relativized form $\mathrm{rgap}_i = \mathrm{gap}_i - \mathrm{gap}_{\mathrm{first}}$; the final score is

$$\mathrm{score}_i = B_i - \max\left(0,\ \mathrm{rgap}_i - z \cdot \hat{\sigma}(\mathrm{rgap}_i)\right), \qquad z = 2.0. \tag{5}$$

The final selection is $\arg\max_i \mathrm{score}_i$: B is the primary signal—never consulted during any round, hence free of selection contamination; A′ the overfitting probe; chain-head relativization cancels the A/B difficulty-baseline difference; the soft threshold discounts inflation only past the one-sided 97.5% line, with data-driven $\hat{\sigma}$ removing manual per-dataset calibration. $\hat{\sigma}(\mathrm{rgap}_i)$ comes from a paired bootstrap ($n_{\mathrm{boot}} = 2000$; the *same* resample index sets applied to all K candidates, so pool difficulty cancels—with $\mathrm{Val}_B = 500$, paired $\hat{\sigma}_{\mathrm{diff}} \approx 0.011$ versus ≈0.031 under independence). The criterion is adaptive by construction: on shallow chains ($K \approx 3$) genuine inflation (≈0.02) stays below $z \cdot \hat{\sigma}$ and selection degenerates to pure B argmax; on deep chains ($K \geq 5$) inflation of 0.04–0.06 crosses the threshold and the penalty switches on. Ties on B go to the higher A′. The pool also salvages gating-rejected *full rewrites* (Section 4.1.4) at zero extra risk, an inflation-free reference for rgap calibration; when data is too small for a Val_A, the criterion degenerates to pure B argmax while retaining the "test evaluated exactly once" safeguard.

4 Experiments

4.1 Setup

4.1.1 Datasets

We evaluate on six diverse datasets: **LiveMathematicianBench** (Exact Match on the MCQ label; SkillOpt's home terrain, where it reports +29.3 MCQ—we contest it where it is strongest), **SearchQA** (Exact Match, with F1 and substring EM additionally recorded), **DocVQA** (ANLS), **SpreadsheetBench** (Ma et al., 2024) (Cell Accuracy after LibreOffice recalculation), **ALFWorld** (environment-determined success rate), and **OfficeQA** (Exact Match). Split sizes and merged validation sets are in Table 1; task descriptions and protocols in Appendix A.

4.1.2 Models

The main experiment uses a dual-model setup through the Volcengine `openai_compatible` gateway: **glm-5.3** as the optimizer (diagnoses failures and rewrites the skill; `reasoning_effort=high`, 65,536-token output budget, timeout 1,800 s), and **qwen3.8-flash** as the executor (attempts tasks; thinking disabled, 16,384-token output budget). Roles and parameters are identical across arms and enter the run fingerprint, so the model is not a confounder; cross-model transfer is future work (Section 5.1). The 16,384-token executor budget is itself protocol: earlier per-dataset budgets (1,500/3,000/4,000 tokens by tier) silently truncated long trajectories—misread as task failures—and under truncation pressure the gating layer must refuse to force-accept a slow-updating candidate (Sections 3.4.1 and 4.2).

4.1.3 Baselines

Main comparison (fullbench, preregistered). A **two-arm design × 10 random seeds** in which every component is held fixed except the sampling and selection protocols:

1. **Baseline arm (original SkillOpt protocol)**: uniform random instance selection; three-way split (Train / merged Val / Test), merged Val serving in-round gating and final-stage selection; final selection deploys the chain-tail skill (no BoN), default gating.
2. **IGAS full-method arm (ours)**: IGAS (Section 3.3: ε-anchored mixing, $\varepsilon = 0.25$, Section 3.3.6; re-verification quota $\rho = 0.25$, Section 3.3.7); four-way split (Train / Val_A / Val_B / Test); strict gating (Section 3.4.1); Best-of-N final selection (Section 3.4.2).

Both arms share the bounded-editing skill synthesis, the same optimizer/executor models and invocation parameters, the same physical data split (frozen `env.split_seed=42`), the same round-and-budget protocol (4 epochs × a proportionally scaled per-dataset batch), and the same 10-seed set and seed-injection policy (Table 2). Because ε-anchoring reverts a quarter of the IGAS arm's

exploration quota to uniform sampling, the design is biased *against* finding an IGAS effect: any measured gap is a lower bound on pure informativeness-driven selection.

Historical single-seed arms (protocol-sensitivity references, not the main comparison). The earlier ablation matrix (Windows, single seed 42, 8 rounds $\times$ 20 instances, single-model qwen3.8-flash) covered: E0 (no-skill, 0 rounds), E1/E3 (random/IGAS $\times$ cluster-level diagnosis), E1_be/E3_be (random/IGAS $\times$ bounded editing), E9_be (uncertainty top-n sampling), E8_be (core-set sampling), E3_be_nouncert (uncertainty-term ablation), and E_upper (staged full-coverage reference); retained as protocol-sensitivity references, not directly comparable to the fullbench two-arm results (dataset, budget, seed, and model protocols all differ).

4.1.4 HYPERPARAMETERS

The complete hyperparameter configuration—round-and-budget protocol, budgets, sampling quotas, gating and BoN settings, seed policy, module thresholds—is given in Table 2 (Appendix B.4).

4.1.5 EVALUATION METRICS

Primary quality metrics are defined per dataset in Section 4.1.1. The framework additionally records convergence speed (rounds to a target percentage of the primary metric), the generalization gap (Train $-$ Test, lower is better), cost metrics (total tokens, API calls, Δprimary per 1K tokens), and diagnostic counters (turns to complete, failure-pattern count).

Primary reported quantities (aggregation protocol, Section 4.5). For each dataset and arm we report the selected-skill Test score (`selected_skill_test_hard`) and the same-run no-skill baseline (`baseline_test_hard`; S_0) as mean $\pm$ std over the 10 seeds, together with the four classes of paired effects (Section 4.5).

4.2 MAIN RESULTS

Figure 2—Convergence curves for the two arms (rounds 1–4): per-round training accuracy and the final-selection signal trajectory, 10-seed mean $\pm$ error bands, faceted by dataset.

Budget-dimension sample-efficiency ablations (B1/B2 total-budget, B3/B4 budget-matched allocation; B-series, SpreadsheetBench only, historical single-seed protocol) are reported in Appendix B.5 (Table 3).

Table 1—Main-results matrix. Rows (one group per dataset): S_0 no-skill baseline, SkillOpt selected skill, IGAS selected skill; columns: primary metric (mean $\pm$ std, $N = 10$ seeds). Additional blocks per dataset: IGAS$-$SkillOpt paired selected difference (mean $\pm$ std), 95% percentile-bootstrap CI (10,000 resamples, rng seed 0, two-sided), two-sided Wilcoxon signed-rank p (normal approximation, tie correction); within-arm gains (IGAS sel$-$IGAS S_0, SkillOpt sel$-$SkillOpt S_0) and the cross-arm gain difference. Data from `scripts/aggregate_full_bench.py` after the fullbench 120 jobs complete.

Findings (pending data). [PLACEHOLDER: Main-results findings bullets, one per dataset + a cross-dataset summary (counts of positively-gained datasets and CIs excluding zero). Expectations: the IGAS$-$SkillOpt gain is larger on datasets with concentrated failure modes and information-rich pools; it narrows on small train pools (ALFWorld 39 / OfficeQA 50) or near the capability ceiling. IGAS-slot degeneration on the two batch=2 datasets (Section 4.1.4) makes their IGAS effect near pure exploration sampling—interpret as a lower bound. No directional statement is verified until the data are in.]

Per-round training accuracy (Table 1a), rounds-to-threshold (Table 2), and generalization gap (Table 3) are specified in Appendix C.

4.3 ABLATION STUDIES

Component-replacement (Table 4), batch-selection (Table 5), embedding-comparison (Table 6), and pluggability-factorial (Table 9) ablations are specified in Appendix C; the subsections below report the ablations that settle design choices.

4.3.1 EMBEDDING METHOD ABLATION

We compare two embedding backends—SBERT (`all-MiniLM-L6-v2`, 384-dim semantic embeddings, local, default) and Hash TF-IDF (256-dim feature hashing, a lightweight fallback)—to assess whether semantic embeddings improve failure-pattern discrimination. Hash TF-IDF may conflate semantically distinct errors with similar surface forms (e.g., "SQL JOIN error" vs. "SQL WHERE error"), while SBERT captures semantics, potentially improving cluster quality and the representativeness neighborhood counts. **Metrics**: (1) Silhouette clustering quality; (2) final performance (CA/EX/EM); (3) failure-pattern discrimination; (4) computational overhead.

Embedding robustness. We expect Hash TF-IDF to retain $\sim$95% of SBERT's final success rate on purely textual tasks (e.g., SearchQA / OfficeQA): the utility provides two complementary signals and the coverage selector relies only on *relative* similarities, so even a coarse backend identifies high-utility instances—IGAS is thus robust to embedding quality, with Hash TF-IDF a viable fallback for constrained deployments.

4.3.2 CLUSTERING DISTANCE THRESHOLD ABLATION

The distance threshold τ_{dist} (cosine distance, default 0.7) is the single switch for cluster granularity. Calibration is harder than in conventional clustering: problem statements and erroneous solutions occupy 60–70% of failure-trajectory text (median pairwise similarity of MATH failure texts ≈ 0.25 under SBERT). An 8-round sweep on old MATH (Hendrycks et al., 2021b) (historical single-seed protocol) shows $\tau_{\text{dist}} = 0.7$ (sim > 0.3, within the 0.3–0.5 range of same-mode failures) aligns cluster count with the Top-K diagnostic budget while preserving large-cluster support; smaller values shred large clusters, and 0.8 merges heterogeneous failure modes. The full sweep table is in Appendix B.5.1; deterministic agglomerative clustering ensures reproducibility.

4.3.3 DIAGNOSTIC BUDGET (TOP-K) ABLATION

The Top-K diagnostic budget (default $K_{\max} = 5$) truncates clusters for diagnosis in descending size order. The cumulative-coverage-versus-K curve has an elbow at $K = 5$: $\sim$90% coverage of dominant failure modes at $\sim$50% cost reduction versus $K = \infty$ (K=3 $\rightarrow$ 5 yields $\sim$15% coverage gain; K=5 $\rightarrow$ 10 only $\sim$5%)—top-5 is the Pareto-optimal operating point. **Failure Recovery Rate (FRR)**—the success rate, under the updated skill θ_{R+1}, of re-running instances from clusters diagnosed in round R, off the training sampling path—measures how effectively diagnosed failure modes are fixed (definition and the expected $\sim$60–80% vs. $\sim$30–50% Top-5-vs-small-cluster contrast: Appendix B.5.2).

4.3.4 PLUGGABILITY: IGAS SAMPLING $\times$ SKILL SYNTHESIS METHODS

Motivation and configurations. IGAS is an instance-selection strategy orthogonal to skill synthesis. We verify this with a **2$\times$2 factorial design** (sampling strategy $\times$ diagnosis granularity) fixing the SkillOpt bounded-editing protocol (Yang et al., 2026): (i) `skillopt`—minibatch per-trajectory diagnosis with bounded editing (top-4 proposals per round, rejection memory, full rewrite every 3 rounds); (ii) `skillopt_clusterdiag`—the same protocol with cluster-level diagnosis. ClusterIGAS, Trace2Skill-style (Ni et al., 2026), and OPRO (Yang et al., 2024) are one-shot manual reference runs outside the automated matrix.

Experiment design. 2 sampling strategies (`random`, `info_gain`) $\times$ 2 optimizer configurations (`skillopt`, `skillopt_clusterdiag`) = 4 configurations (P1–P4), on SpreadsheetBench $\times$ 3 seeds (12 runs), sharing trace generation, evaluation, validation pool, and embedder; the only variables are sampling strategy and diagnosis granularity. **Factorial separation**: sampling gain = (`info_gain` + `skillopt`) $-$ (`random` + `skillopt`); diagnosis gain = (`random` + `skillopt_clusterdiag`) $-$ (`random` + `skillopt`).

Why does IGAS improve any skill synthesis method? IGAS does not modify synthesis internals; it improves the **input quality** of failure trajectories fed to synthesis: (1) the coverage selector feeds diverse, non-redundant high-utility failures; (2) the failure-signal factor $\hat{u} \cdot h(\hat{u})$ makes failures diagnosable (likely to fail, but not in already-diagnosed ways); (3) representativeness ties patterns to broader instance populations; (4) pattern memory with temporal decay keeps new failure types arriving across rounds.

Adapter Note: We do not use the original Trace2Skill/SkillOpt codebases but implement adapters faithfully reproducing their core design within our framework (same model, harness, and evaluation), ensuring fair comparison.

4.4 Analysis

Qualitative analyses—failure-pattern growth curves (Figure 3), selected-instance IG-score distributions (Figure 4), skill-text embedding trajectories (Figure 5), and per-dataset failure-mode inventories (Tables 10–11)—are specified in Appendix C.

4.4.1 Inflation-Accumulation Diagnostics

Two diagnostics close the mechanism-to-cost loop (Appendix C): **Evidence 1** calibrates every BoN-pool candidate's validation score against Test (Figure 6), expecting the A′ group to lie above the diagonal with an offset growing in chain position; **Evidence 2** tracks rgap versus chain position (Figure 7), expecting monotone rise—directly measurable evidence of inflation accumulation in single-validation-set architectures.

4.4.2 Round Budget: 8 vs. 12 vs. 16 Rounds (Old MATH)

The old MATH protocol used 8 evolution rounds; both arms were additionally run at 12 and 16 rounds (historical single-seed protocol; evidence of round-budget saturation). The IGAS arm is flat across budgets (86.6/86.6/86.8, all pairwise McNemar $p = 1.0$): final-stage selection converges early, no candidate accepted after round 8 exceeds the round-6 validation peak, and the skill text stops growing—the 8-round protocol already captured the attainable gain. This motivated spending the full-protocol budget on seeds and per-round sample size rather than round count. The full saturation table is in Appendix B.5.3.

The final-selection strategy ablation (BoN vs. last round vs. pure B $\arg\max$ vs. in-round chained A $\arg\max$, replayed on the same logs) quantifies the Test cost of inflation accumulation (Appendix C, Table 12).

4.5 Statistical Significance

The fullbench statistical protocol was preregistered and is fixed in `scripts/aggregate_full_bench.py`. The paired unit is the **seed**: under the same (dataset, seed) the arms share the physical split, and every run co-evaluates the zero-skill baseline S_0, making all four effect classes pairwise computable per seed: (1/2) within-arm gains, selected-skill Test $-$ S_0 per arm; (3) cross-arm selected difference, IGAS $-$ SkillOpt selected-skill Test; (4) cross-arm gain difference, (IGAS sel $-$ IGAS S_0) $-$ (SkillOpt sel $-$ SkillOpt S_0).

Intervals, tests, and aggregation rules. Each effect is reported over the 10 seeds as mean $\pm$ std, a **paired percentile bootstrap 95% CI** (10,000 resamples, rng seed 0, two-sided), and a **two-sided Wilcoxon signed-rank test** (normal approximation with tie correction, zero-difference pruning; at $n = 10$ the minimum achievable p is $\approx$0.002). Cross-dataset summaries are descriptive only (counts of positive-direction datasets and of CIs excluding zero); no pooled cross-dataset p-value is computed (metrics differ in scale and difficulty). Before aggregation, every run is validated (completion status, seed consistency with the directory name, split-hash, task-ID digest, test-segment size); datasets with incomplete pairing move to exploratory results (`NOT_FOR_PAPER`), excluded from the formal counts. Per-dataset effect tables are specified in Appendix C.

Repeated-evaluation noise floor. Decoding temperature is 0, but serving is not strictly deterministic: historical audits observed 0–7 pp score shifts re-evaluating the same skill on the same instances (old LiveMath). The 10-seed protocol folds this noise floor into the variance estimates: between-seed standard deviation and paired CI jointly cover sampling variance and evaluation nondeterminism. **[PLACEHOLDER: The measured between-seed standard deviations of each fullbench dataset, to be compared against the historical noise band once the data are in.]**

5 DISCUSSION

The multiplicative utility $\text{Utility}(i) = \hat{u} \times h(\hat{u}) \times (1 + \log_2(1 + \text{repr}))$ encodes two necessary conditions: **failure signal**—similar enough to past failures to be challenging, but not so similar that its failure mode is already diagnosed (the entropy weight h suppresses $\hat{u}$ near 1; the composite peaks at $\hat{u} \approx 0.70$)—and **generalization** ($\text{repr} \geq 0$: the instance represents a category of similar tasks). The **multiplicative** combination enforces the **AND-necessity property** (Section 3.3.1): an additive form would let one dimension compensate for a zero in the other, whereas the multiplicative form is positive only when both axes are non-zero; it is a heuristic design choice, not a probabilistic derivation.

A third condition—*non-redundancy* with what is already diagnosed—is enforced by the batch selector (Section 3.3.5): the facility-location objective collapses the marginal gain of candidates from already-covered failure modes. Unlike Bayesian Experimental Design, our formulation is **parameter-free**, operating directly on the embedding geometry—practical when the "parameter" (the skill text) is non-parametric and admits no gradient-based posterior inference.

5.1 LIMITATIONS AND FUTURE WORK

1. **Embedding granularity**: SBERT (`all-MiniLM-L6-v2`, 384-dim) may conflate overlapping surface forms with different intents; the Hash TF-IDF fallback (256-dim) captures only lexical overlap (Section 4.3.1).
2. **Uncertainty prior and cold start**: the round-0 default $\hat{u}_i = 0.5$ gives no discrimination, so round 1 prioritizes representative instances (Section 4.2); thereafter $\hat{u}_i$ is estimated from positive evidence with temporal decay ($\gamma = 0.8$, Section 3.3.2; round-0 weights drop to $0.8^{16} \approx 0.028$ by round 16, Section 4.4.2), leaving a self-correcting conservative bias in early rounds.
3. **Surrogate objective gap**: the $(1 - 1/e)$ guarantee (Theorem 1) holds for any non-negative utility weights—the contribution is the heuristic utility design, validated empirically (Appendix B.7); residual gaps lie in the utility decomposition and in $f_{\theta_t}(S)$ not modeling the interventional effect of skill updates.
4. **Soft center sampling bias**: $\text{repr}(i)$ rewards dense regions while coverage discounts already-covered neighbors, so the algorithm prefers one high-representativeness "center" per cluster (implicit soft k-center behavior)—a design property: fixing a centroid implicitly covers its periphery.
5. **Limited task types**: beyond the six evaluated task types (Appendix A), broader coverage (web navigation, long-horizon software engineering) remains future work.

Three of these limitations point directly to future work:

1. **Domain-specific embeddings** fine-tuned on spreadsheet/SQL language (e.g., VLOOKUP vs. INDEX-MATCH).
2. **Richer theory**: tighter-than-$(1 - 1/e)$ bounds (continuous greedy), matroid constraints (per-cluster quotas), and convergence bounds for IGAS.
3. **Scaling out**: multi-skill evolution, cross-domain skill reuse, and cross-model transfer of skill files.

6 CONCLUSION

We presented **Information-Guided Active Sampling (IGAS)**, which targets the instances a skill is most likely to learn from: failure similarity and representativeness, combined multiplicatively and diversified by a utility-weighted facility-location batch objective with submodular guarantees. In a preregistered two-arm design against the original SkillOpt baseline—six datasets $\times$ 10 seeds $\times$ 2 arms = 120 runs, identical in models, splits, seeds, and proportionally scaled budgets, differing only in the sampling and selection protocol—ε-anchored mixing and re-verification quotas further bias the design *against* finding an IGAS effect.

[PLACEHOLDER: Headline results (pending fullbench data): the IGAS−SkillOpt selected-skill difference per dataset (mean ± std, 95% CI, Wilcoxon p), the count of positively-gained datasets, and the net cross-arm gain difference; adverse directions reported honestly and discussed against Section 5.1 (boundary conditions: training-pool size, failure-mode concentration, capability-ceiling proximity).]

Beyond the headline numbers, we regard the honest-baseline discipline as a contribution in itself: Best-of-N final selection never ranks candidates on their training distribution, salvage of rejected rewrites gives an inflation-free reference point, and re-verification quotas yield direct causal evidence of failure-mode fixes. A 2×2 pluggability factorial across skill-synthesis paradigms is designed but not yet run at scale; we make no demonstrated-orthogonality claim. Practically, IGAS uses local SBERT embeddings (no API dependency) and cross-round failure-pattern memory, deployable in restricted environments.

AI USE STATEMENT

In this work, we used generative AI tools for assistance with code development (experiment harness and analysis scripts, verified and tested by the authors) and for language polishing of the manuscript. We have not used generative AI tools for producing the experimental results, the theoretical analysis, or the claims presented; all numbers reported are produced by the preregistered pipeline of Sections 3.4 and 4.5. LLM-generated code was verified against the protocol specification, and AI-polished text was checked by the authors for accuracy. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI. **[PLACEHOLDER: Authors: adjust the disclosure items to match your actual AI usage before submission.]**

ETHICS STATEMENT

This work evaluates sampling strategies for LLM-agent skill evolution on six public benchmarks. No human subjects are involved. All datasets are used under their respective licenses for non-commercial research; we release no new personal data. The compute cost of the 120-run main experiment and the budget-discipline mechanisms used to contain it are documented in Section 4.1.4. We see no foreseeable negative societal impact beyond the general considerations applicable to LLM-agent research.

REPRODUCIBILITY STATEMENT

The full evaluation protocol is specified in the main text: the four-way data split and seed policy (Section 3.4), model and budget configuration (Section 4.1.2), all hyperparameters (Section 4.1.4, Table 2 in Appendix B.4), the preregistered statistical protocol (Section 4.5), and the historical-protocol caveats (Sections 4.4.2 and 4.3.2). Implementation details (embedding backend, failure pattern memory, budget-ablation table, utility-design ablations, skill-text quality correlation, uncertainty calibration, and computational cost) are provided in Appendices B–B.10; deferred result tables and figures are specified in Appendix C. Source code and run logs will be released for review **[PLACEHOLDER: anonymous repository link to be added]**.

REFERENCES

Ricardo J. G. B. Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2013.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. FireAct: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*, 2023.

Gérard Cornuéjols, Marshall L. Fisher, and George L. Nemhauser. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science*, 23 (8):789–810, 1977.

Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *International Conference on Machine Learning*, pp. 1183–1192, 2017.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, and Jacob Steinhardt. Measuring coding challenge competence with APPS. In *NeurIPS Datasets and Benchmarks Track*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks Track*, 2021b.

Dennis V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. SpreadsheetBench: Towards challenging real world spreadsheet manipulation. In *NeurIPS Datasets and Benchmarks Track*, 2024.

Elizbar A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294, 1978.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2Skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Burr Settles. Active learning literature survey. Technical Report 1648, University of Wisconsin-Madison, 2009.

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*, 2023.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Geoffrey S. Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26(4):359–372, 1964.

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

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive strategy for self-evolving agent skills. *arXiv preprint arXiv:2605.23904*, 2026.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. *arXiv preprint arXiv:2310.12823*, 2023.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. In *International Conference on Machine Learning*, pp. 62138–62160, 2024.

A Dataset Details

The six fullbench datasets are pre-materialized as fixed split directories by per-dataset scripts (`scripts/materialize_*.py`, frozen `env.split_seed=42`) and shared physically by both arms; the four-way split sizes are reported in Table 1 in the main text. Per-dataset task descriptions and evaluation protocols:

- **LiveMathematicianBench** (Train/Val$_A$/Val$_B$/Test = 426/8/49/125; SkillOpt merged Val = 57, comprising 17 frozen and 40 stratified instances). Advanced-mathematics theorem comprehension: each instance presents a graduate-level theorem statement with multiple-choice candidate interpretations, and the executor must select the correct label. Primary metric: Exact Match on the multiple-choice label. This is the home terrain of SkillOpt-style bounded editing (the SkillOpt paper reports +29.3 MCQ), so we contest bounded editing where it is strongest.
- **SearchQA** (400/100/100/1400; SkillOpt merged Val = 200). Retrieval question answering: the executor generates a short answer to a question over retrieved evidence. Primary metric: Exact Match over generated short answers, with F1 and substring EM recorded as auxiliary readings.
- **DocVQA** (107/25/28/374; SkillOpt merged Val = 53). Document visual question answering: questions about the textual and layout content of document images. Metric: ANLS (average normalized Levenshtein similarity).
- **SpreadsheetBench** (Ma et al., 2024) (80/20/20/280; SkillOpt merged Val = 40). Spreadsheet manipulation: given a spreadsheet and a natural-language operation request, the executor produces the edit; the resulting file is recalculated headlessly in LibreOffice and scored by Cell Accuracy against the ground truth.
- **ALFWorld** (39/9/9/134; SkillOpt merged Val = 18). Embodied instruction following in text-based household environments (ALFRED-style tasks rehosted in ALFWorld): the executor issues a sequence of text actions to complete a household goal, and success is determined by the environment's own success checker.
- **OfficeQA** (50/12/12/172; SkillOpt merged Val = 24). Office-document question answering: questions about the content of office documents (slides, sheets, rich-text documents). Metric: Exact Match.

B Implementation Details

B.0.1 Design Remarks and Removed Components

Scope of the greedy guarantee. Theorem 1 guarantees near-optimality of *immediate coverage*—the utility-weighted mass newly covered by the selected batch under the current skill file—not end-to-end skill-evolution performance: it does not bound how much the induced skill update improves Test scores, which depends on the diagnosis and synthesis stages downstream. The guarantee is fully general with respect to the utility weights: it holds for any non-negative $U_{\theta_t}(j)$, so nothing in the approximation depends on the specific failure-signal/representativeness semantics.

No diversity hyperparameter. Because redundancy control is folded into the coverage objective itself (the marginal gain of a candidate already covered by $\mathcal{T}_t$ or by an earlier batch member collapses

to near zero), the method needs no separate diversity weight or minimum-distance hyperparameter—one fewer knob to tune, and one fewer way to misallocate budget.

Smoothness of the utility. $\text{Utility}(i) = \hat{u}_i \cdot h(\hat{u}_i) \cdot (1 + \log_2(1 + \text{repr}_i))$ is continuous in its inputs; small perturbations of $\hat{u}$ or of the neighborhood count shift the score smoothly, so selection is stable under embedding noise (in contrast to hard distance thresholds inside the scoring path).

One objective, two redundancy scales. The same facility-location marginal serves both cross-round redundancy (through the tried-set initialization of m_j) and within-round redundancy (through greedy incremental selection): a candidate adds value only for the untried mass it *newly* covers relative to everything already selected or tried, so no separate mechanism is needed for either scale.

Why exploitation-dominated. The design deliberately favors exploitation over exploration: the failure-signal factor concentrates budget on instances near the current failure boundary, and the only exploration mechanisms are the ε-anchored mixing quota (Section 3.3.6) and the optional UCB bonus ($\beta = 0.3$). In skill evolution, unlike classic bandit settings, a purely exploratory sample yields no diagnostic signal (an instance far from all known failures that succeeds teaches nothing actionable), so the exploration budget is bounded and structured rather than open-ended.

Proof of Theorem 1. For each j, the set function $g_j(S) = \max_{i \in S \cup \mathcal{T}_t} \text{sim}(i, j)$ is a coverage function of S: monotone, and submodular because for $S \subseteq T$ the increment $g_j(T \cup \{x\}) - g_j(T)$ never exceeds $g_j(S \cup \{x\}) - g_j(S)$. A non-negative linear combination of monotone submodular functions is monotone submodular, so the greedy $(1 - 1/e)$ guarantee applies (Nemhauser et al., 1978; Cornuéjols et al., 1977).

Removed: centroid-distance novelty multiplier. An earlier design multiplied the utility by a novelty term $1/(1 + \text{dist}(i, \text{centroid}))$ (distance from the embedding centroid of tried instances). It was a static prior dwarfed by the failure-similarity factor's dynamic range and, more damaging, it demonstrably misallocated sampling budget: it favored outliers—atypical instances whose failures were rarely representative of any broader pattern and are hard to act on—starving the diagnosable boundary region; it was one identified cause of an earlier controlled run (historical ablation matrix, old WTQ dataset) in which the informative arm lost to random sampling by 7.3 pp. It was removed in favor of the coverage structure, whose tried-set initialization zeroes the marginal contribution of already-covered instances rather than distorting the utility estimate.

B.1 Failure Pattern Memory Schema

```
FailurePattern:
    centroid: np.ndarray (384-dim for SBERT / 256-dim for hash)
    label: str (human-readable)
    first_seen_round: int
    last_seen_round: int
    trace_count: int
    common_errors: list
    common_tools: list
```

On each merge, the centroid is re-normalized to unit length after the trace-count-weighted running-average update, and the common-error/common-tool sets are unioned; a cluster below the merge threshold $\tau_{\text{merge}} = 0.8$ spawns a new entry (Section 3.3.7).

B.2 SBERT Embedding (all-MiniLM-L6-v2)

```
from sentence_transformers import SentenceTransformer

# Lazy-load model (384-dim, local, no API)
_sbert_model = None
def _get_model():
    global _sbert_model
    if _sbert_model is None:
        _sbert_model = SentenceTransformer('all-MiniLM-L6-v2')
    return _sbert_model
```

```
def embed_instruction(instruction: str) -> np.ndarray:
    """Embed instruction using SBERT (384-dim, semantic)."""
    model = _get_model()
    embedding = model.encode(instruction, convert_to_numpy=True)
    return embedding / np.linalg.norm(embedding)  # L2 normalize
```

B.3 Utility-Weighted Facility-Location Batch Selection

```
def info_gain_sample(instances, n, past_failed_instructions,
                     tried_instructions, beta=0.3):
    # Step 1: Compute the two-factor utility for each untried instance (Sec 3.3.1)
    untried = [inst.instruction for inst in instances]
    N = len(untried)
    utilities = zeros(N)
    for i, inst in enumerate(instances):
        u_hat = estimate_failure_similarity_temporal(       # Sec 3.3.2 (gamma = 0.8)
            inst.instruction, past_failed_instructions)
        p_fail = clip(u_hat, 1e-6, 1 - 1e-6)
        fail_signal = p_fail * binary_entropy(p_fail)        # h(u_hat) * u_hat
        repr_count = estimate_representativeness(           # neighborhood count
            inst.instruction, untried)                      # (cos > 0.7)
        u = fail_signal * (1.0 + log2(1.0 + repr_count))    # two-factor utility (v4
        if beta > 0:                                        # UCB1 exploration bonus
            u += beta * sqrt(log(N + 1) / (repr_count + 1))
        utilities[i] = max(u, 0.0)                          # non-negative node weigh

    # Step 2: Similarity matrix (coverage oracle; NOT a log-det volume)
    # L = XX^T + eps*I - Gram matrix of L2-normalized embeddings
    L = pairwise_similarity_matrix(untried) + 1e-6 * eye(N)

    # Step 3: Utility-weighted facility-location greedy (Sec 3.3.5, Theorem 1)
    # f(S) = sum_j U(j) * max_{i in S union Tried} sim(i, j)   (monotone submodular)
    # Marginal: Delta(i|S) = sum_j U(j) * [sim(i,j) - m_j]_+
    # m initialized from the tried set: cross-round redundancy control (Sec 3.3.3)
    m = max_tried_similarity(tried_instructions, untried)   # m_j = max_{s in T} sim

    selected = []
    for _ in range(n):
        marginals = maximum(L - m[newaxis, :], 0.0) @ utilities   # O(N^2) per step
        marginals[selected] = -inf                               # mask already-selec
        best = int(argmax(marginals))
        selected.append(best)
        maximum(m, L[best], out=m)   # m_j <- max(m_j, sim(best, j)): within-round co

    # Greedy achieves the (1-1/e) approximation guarantee (Theorem 1,
    # Nemhauser et al., 1978); no matrix inversion or factorization is needed.
    return [instances[i] for i in selected]
```

B.4 Full Hyperparameter Configuration

Note. This table describes the fullbench main experiment. The earlier ablation matrix used a historical single-seed protocol (Windows, seed 42, 8 rounds $\times$ 20 instances fixed budget, single-model qwen3.8-flash); its arm definitions are summarized in Section 4.1.3 and its budget ablations in Table 3 below.

Table 2: Hyperparameter configuration of the main (fullbench) experiment. Both arms share every entry except the seed injection and the selection protocol (SkillOpt: merged validation, chain-tail deployment; IGAS: four-way split with strict gating and BoN).

Group	Parameter	Value
Round protocol	Epochs × steps	4×1
	Batch size	$\mathrm{round}(\lvert\mathrm{Train}\rvert \times 20/426)$: LiveMath 20, SearchQA 19, DocVQA 5, SpreadsheetBench 4, ALFWorld 2, OfficeQA 2
Budgets	Executor output budget	16,384 tokens per call (qwen3.8-flash, thinking disabled)
	Optimizer output budget	65,536 tokens per call (glm-5.3, `reasoning_effort=high`)
	Optimizer timeout	1,800 s
	Trace budget	12,000 tokens
	Max turns per instance	10
Sampling	ε-anchoring	$\varepsilon = 0.25$
	Re-verification quota	$\rho = 0.25$
	Re-verification gap	`min_reverify_gap` $= 2$
Gating	Acceptance criterion	Strict: $\Delta > 0$ (bootstrap round $\Delta \geq 0$)
	Slow updates	force-accept
	Addition avoidance	urgency-based skip (0.5 residual retained after rejections)
BoN	Significance level	$z = 2.0$
	Paired bootstrap	$n_{\mathrm{boot}} = 2000$ resamples
	Tie-break	on A′ when B difference $< z \cdot \hat{\sigma}_{\mathrm{diff}}$
	Salvage	gating-rejected full rewrites enter the candidate pool
Seeds	10-seed set	{42, 123, 456, 789, 1010, 1111, 1212, 1313, 1414, 1515}
	Seed injection	SkillOpt: `train.seed`; IGAS: `train.seed`, `sampling.seed`, `protocol.bon_seed`
	Split seed	`env.split_seed` $= 42$ (frozen, shared by both arms)
Modules	Embedding	SBERT `all-MiniLM-L6-v2` (384-dim, local)
	Clustering	agglomerative, $\tau_{\mathrm{dist}} = 0.7$
	Temporal decay	$\gamma = 0.8$
	UCB exploration bonus	$\beta = 0.3$
	Representativeness threshold	$\cos > 0.7$
	Pattern merge threshold	0.8

B.5 Ablation Extensions

The main IGAS pipeline (Section 3.3.5) uses the utility-weighted facility-location greedy with temporal-decay uncertainty estimation ($\gamma = 0.8$, Section 3.3.2), with an explicit UCB exploration bonus enabled by default. We document the ablation parameters below. The UCB exploration bonus (β) is **enabled by default** ($\beta = 0.3$) and can be disabled via API parameters for ablation analysis.

Extension 1: Temporal Decay (γ). The base uncertainty estimator uses a Nadaraya–Watson kernel regression with temporal decay weights. Each historical failure embedding is weighted by $w_j = \gamma^{t_{\mathrm{current}} - t_j - 1}$ where $\gamma = 0.8$. This lets the estimator automatically "forget" older failures, adapting to the non-stationary distribution caused by skill updates. Default $\gamma = 0.8$ (half-life $\sim$3 rounds); ablation uses $\gamma \in \{0.6, 0.9\}$.

Extension 2: UCB1-Style Exploration Bonus (β). The base acquisition utility is exploitation-dominated—it favors instances similar to known failures. The exploration bonus adds a UCB1-style term instantiated on representativeness counts:

$$\mathrm{Utility}(i) = \mathrm{Utility}_{\mathrm{core}}(i) + \beta \sqrt{\frac{\ln(N_t + 1)}{\mathrm{repr}(i) + 1}} \tag{6}$$

where N_t is the size of the current untried pool and $\mathrm{repr}(i)$ is the neighborhood count of Section 3.3.1. Instances with few near neighbors (poorly covered by the current pool) receive a larger bonus, so they are occasionally selected despite a weaker failure signal. Because the bonus is a static per-instance adjustment folded into the node weight $U(j)$ before selection, Theorem 1 applies unchanged. Default $\beta = 0.3$ (enabled); ablation uses $\beta = 0$ (disabled, X3).

Ablation results.

Table A1—Ablation of extensions on SpreadsheetBench (X-series single-variable, historical single-seed protocol—8 rounds × 20 instances, single model, old three-/four-way split; protocol differs from the full-bench main experiment, retained only as a hyperparameter-sensitivity reference). Configurations: Base (γ=0.8, β=0.3, full main pipeline); X1: No temporal decay (γ=1.0, all failures weighted equally); X2: Strong decay (γ=0.6, faster forgetting); X3: No exploration (β=0, UCB-style exploration disabled); X4: Weak decay (γ=0.9, between base and no decay). Column: final SR ($\mu \pm \sigma$). Expected: base pipeline is already strong; temporal decay adapts to non-stationarity; disabling UCB exploration may reduce boundary coverage in later rounds.

Budget-Allocation Ablation (B-series). The B-series is a **SpreadsheetBench-specific baseline ablation** (historical single-seed protocol: baseline configuration of 8 rounds × 20 samples per round, total budget 160), independent of the main experiment's 4-epoch × proportionally scaled batch protocol (Section 4.1.4); the mainline comparison along the round-budget dimension is covered by the 8/12/16-round sweep of Section 4.4.2, while the B-series focuses on the budget-allocation structure itself. We ablate two budget dimensions with the E3 single arm (SpreadsheetBench, multi-seed): (1) the **total-budget axis**—fixing the allocation (8 rounds) while sweeping the per-round sample size (B1: 8×10=80, B2: 8×40=320); and (2) the **budget-matched allocation axis**—fixing the total budget at 160 while varying the iteration granularity (B3: 4×40, B4: 2×80). The number of rounds itself is not ablated separately within the B-series: comparisons at fixed total budget (E3 vs. B3 vs. B4) are the informative round-related ablation. Note that the per-round sample size jointly affects both sampling and the input scale of cluster-level diagnosis (smaller batches yield fewer per-round failures and sparser clusters), so the B-series measures their combined effect.

Table 3: Budget-allocation ablation on SpreadsheetBench (B-series, historical single-seed protocol). Values to be filled once the runs complete.

Configuration	Rounds × samples/round	Total budget	Final test SR ($\mu \pm \sigma$)
B1 (budget lower)	8 × 10	80	**[PLACEHOLDER: ...]**
E3 (baseline ref)	8 × 20	160	**[PLACEHOLDER: ...]**
B2 (budget upper)	8 × 40	320	**[PLACEHOLDER: ...]**
B3 (coarse)	4 × 40	160	**[PLACEHOLDER: ...]**
B4 (coarsest)	2 × 80	160	**[PLACEHOLDER: ...]**

Expected: SR increases monotonically with total budget with diminishing marginal gains (B1 $<$ E3 $<$ B2, with B2$-$E3 $<$ E3$-$B1); within the budget-matched group, fine-grained iteration dominates (E3 $\geq$ B3 $\geq$ B4, as more rounds feed incremental failure evidence to cluster-level diagnosis); B1 probes the robustness of cluster-level diagnosis under sparse per-round input.

B.5.1 Clustering Distance Threshold Sweep

Table 4 reports the full 8-round empirical sweep of the distance threshold τ_{dist} on the old MATH dataset (Hendrycks et al., 2021b) (historical single-seed observation, protocol differing from the fullbench main experiment; retained as the calibration basis for the default value).

B.5.2 Diagnostic Budget (Top-K) and Failure Recovery Rate

Table 8—Impact of Top-K cluster pruning on diagnosis quality and LLM cost: $K_{\max} = \infty$ (all clusters): full coverage, high LLM cost; $K_{\max} = 10$: ∼95% coverage, moderate cost; $K_{\max} = 5$ (default): ∼90% coverage of dominant failure modes, ∼50% cost reduction; $K_{\max} = 3$: ∼75% coverage, aggressive cost reduction. Expected: $K_{\max} = 5$ captures the majority of failure traces (Pareto-optimal); smaller clusters contribute diminishing returns to skill improvement while consuming disproportionate LLM budget.

Table 4: Impact of distance threshold τ_{dist} on clustering granularity and diagnostic quality (old MATH dataset, 8-round empirical sweep, historical single-seed observation).

τ_{dist}	Clustering outcome	Diagnostic usability
0.25 / 0.5	All singletons (requires sim $>$ 0.75 / 0.5 for co-membership; only near-duplicate texts cluster)	No cluster-level patterns; degrades to per-trajectory diagnosis
0.6–0.65	K = 8–12, fragmented	Cluster count exceeds the Top-K diagnostic budget (K=5); large clusters are shredded
0.7 (default)	**K = 4–8 with genuine large clusters (e.g., 6/2/2/1/1)**	**Cluster count aligned with the Top-K budget; large-cluster support signal preserved**
0.8	K = 2–5, over-coarse	Heterogeneous failure modes are merged; insufficient diagnostic resolution

Failure Recovery Rate (FRR). To measure the effectiveness of skill updates in fixing diagnosed failure modes, we allocate a **fixed diagnostic evaluation budget** at the start of each round: we re-run **all instances from clusters diagnosed in the previous round** using the updated skill file θ_{t+1}. This re-evaluation is performed **off the training sampling path** (these instances are not added to $\mathcal{D}_{\text{tried}}$ and do not affect the active sampling for round $t{+}1$). The success rate on this re-run set defines FRR. This budget is fixed and excluded from the training sample count, ensuring the main evaluation protocol remains uncontaminated. For cluster C_k diagnosed in round R, FRR is:

$$\text{FRR}(C_k, R) = \frac{\left|\left\{ i \in C_k^{(R)} : i \text{ succeeds when re-run with } \theta_{R+1} \right\}\right|}{\left|C_k^{(R)}\right|} \tag{7}$$

We expect Top-5 clusters to have significantly higher FRR ($\sim$60–80%) than smaller clusters ($\sim$30–50%), confirming that the dominant failure modes are not only the most frequent but also the most fixable. FRR should increase across rounds (early rounds $\sim$40% $\rightarrow$ final stage $\sim$80%) as the skill file matures, and IGAS should achieve higher FRR than Random sampling (targeted diagnosis yields more effective skill updates).

B.5.3 Round-Budget Saturation: Full Analysis (Old MATH Dataset)

This subsection expands the round-budget analysis summarized in Section 4.4.2. To test whether the round budget binds, both sampling arms of the old MATH protocol were additionally run at 12 and at 16 rounds (same seed, same per-round budget, post-fix code), giving three independent round budgets per arm (**historical single-seed protocol—this dataset is no longer in the fullbench main matrix; the fullbench main experiment adopts the 4-epoch $\times$ proportionally scaled batch budget design, whose saturation is to be discussed separately once the main data are in**). All entries are graded by the official-aligned scorer (Section 3.4, final Test score, %); Table 5 reports the full saturation matrix.

Table 5: Round-budget saturation on the old MATH dataset (historical single-seed observation; final Test score, %).

Arm	8 rounds	12 rounds	16 rounds
E3_be (IGAS)	86.6	86.6	86.8
E1_be (Random)	83.2	79.6	83.2

Three observations explain the IGAS arm's flatness. First, final-stage selection converges early: in the 16-round runs the selected candidate is the round-6 candidate in both arms (12-round runs: round 9 for IGAS, round 7 for Random), so BoN final selection is not starved by a shorter chain. Second, no candidate accepted after round 8 ever exceeds the round-6 validation peak (Val$_B$ 0.910 for IGAS, 0.878 for Random), so the deeper candidate pool never contributes a better winner. Third, the skill text itself stops growing: in the 16-round chains the bounded-editing skill is frozen at 838 characters (IGAS) for the final 12 rounds and at 863 characters (Random) for the final 8 rounds;

the 12-round IGAS chain likewise freezes from round 3 onward. The Random arm's 12-round dip (79.6) is a single-run fluctuation (8r vs. 12r $p = 0.09$; 8r vs. 16r $p = 1.0$), consistent with the 0–7 pp repeated-evaluation noise band of Section 4.5. Extending rounds costs ≈110 additional minutes per arm for zero expected benefit.

B.6 Reproducibility

All code is available at **[PLACEHOLDER: GitHub repository URL]**. The main experiment (fullbench) is reproduced end-to-end on a Linux server with a single command: `bash scripts/run_full_bench.sh` orchestrates 6 datasets × {SkillOpt, IGAS} × 10 seeds = 120 jobs (bounded concurrency, resumable, per-job ledger; external environments such as the SSB LibreOffice pool and the ALFWorld data are pre-checked by the script, with launch instructions emitted on failure), and `python scripts/aggregate_full_bench.py --out-root outputs/fullbench` aggregates the main results tables under the pre-registered statistical protocol (Section 4.5). The full manual—environment configuration (`.env` model gateway, per-dataset materialization commands, the LibreOffice/UNO service, `ALFWORLD_DATA`), config paths and seed-injection rules, and resume semantics—is provided in `docs/guide/fullbench-commands.md`. Single-arm ablation experiments are still driven by `run_evolution_cluster.py` (the semantics of `--sampling-strategy`, `--skill-method`, `--dataset`, `--seed`, `--max-rounds`, `--sample-size`, `--reverify-ratio`, `--salvage-rejected-rewrites`, `--embedding-method` and other arguments are documented in that script's help), and the 2×2 factorial pluggability experiment is driven by `run_pluggability_experiment.py`.

B.7 Utility Function Form Ablation

To validate our choice of multiplicative utility form, we compare three acquisition function designs, writing $F(i) = \hat{u}_i\, h(\hat{u}_i)$ for the composite failure-signal factor and $R(i) = 1 + \log_2(1 + \text{repr})$ for the representativeness factor:

1. **Additive**: $\text{Utility}(i) = F(i) + R(i)$;
2. **Multiplicative** (ours): $\text{Utility}(i) = F(i) \cdot R(i) = \hat{u}_i \cdot h(\hat{u}_i) \cdot (1 + \log_2(1 + \text{repr}))$;
3. **Harmonic Mean**: $\text{Utility}(i) = 2/(1/F(i) + 1/R(i))$.

The multiplicative and harmonic mean forms satisfy the AND-gating property (utility approaches zero if any factor approaches zero), while the additive form does **not**—it allows a high score in one dimension to compensate for a zero in another. In implementation, we add a small epsilon $\epsilon = 10^{-8}$ to each denominator in the harmonic mean to avoid division by zero; the gating property is preserved in the limit.

Table A3—Utility function form ablation across 3 datasets, 10 seeds. Columns: Strategy, Final Test SR (mean ± std), Rounds to 80% SR. Expected: Multiplicative achieves fastest convergence and highest final SR, with additive underperforming due to compensation between dimensions and harmonic mean performing between additive and multiplicative.

The multiplicative form is expected to outperform additive (which allows high scores in one dimension to compensate for zeros in another, violating the AND-necessity principle) and to match or exceed harmonic mean (which is more sensitive to the minimum factor but introduces additional parameters). This ablation empirically validates our design choice.

B.8 Skill Text Quality Correlation Analysis

To investigate whether sampling quality propagates through to skill file updates, we analyze the correlation between skill file changes and performance improvements. For each round, we compute:

1. **Skill change magnitude**: ROUGE-L similarity between θ_t and θ_{t+1} (lower similarity = larger change);
2. **Performance gain**: $\Delta\text{SR} = \text{SR}_{t+1} - \text{SR}_t$ on the test set.

Figure A6—Scatter plot of skill change ($1-$ ROUGE-L) vs. performance gain (ΔSR) across all rounds and datasets. Each point represents one round's skill update. Expected: A positive monotonic relationship for moderate-to-large changes, indicating that sampling quality propagates through to skill updates. A weak correlation would suggest the bottleneck lies in LLM generation quality rather than sampling strategy, which we would acknowledge as a limitation.

This diagnostic analysis helps identify whether the performance improvements stem from better sampling (our contribution) or are limited by LLM skill generation quality. A strong correlation would validate that IGAS's targeted sampling leads to more effective skill updates; a weak correlation would suggest future work should focus on improving the LLM's skill revision mechanism.

B.9 Uncertainty Score Calibration (Reliability Diagram)

To assess the quality of the uncertainty score $\hat{u}_i$, we provide a **Reliability Diagram** showing the relationship between predicted uncertainty and empirical failure rates. For each bin $b \in [0, 0.1, \ldots, 0.9, 1.0]$, we compute the mean predicted $\hat{u}_i$ and the empirical failure rate among instances falling in that bin. This analysis helps verify whether instances with high uncertainty scores actually fail more often.

Figure A1—Reliability Diagram. X-axis: mean predicted $\hat{u}_i$ per bin; Y-axis: empirical failure rate per bin. Computed across 10 seeds $\times$ 1 round on SpreadsheetBench. Expected: monotonic relationship confirming that higher uncertainty scores correlate with higher failure rates. Note: $\hat{u}_i$ is a heuristic uncertainty indicator, not a calibrated probability, so we do not expect perfect calibration along the diagonal.

This analysis complements the ablation in Appendix B.5 (which shows performance robustness to γ) by verifying the *relative* quality of the uncertainty estimates, not their absolute calibration as probabilities.

B.10 Computational Cost

Table A4—Computational cost breakdown. Rows: Embedding computation **[PLACEHOLDER: X]** ms per instance; IG score computation **[PLACEHOLDER: X]** ms per instance (for 600 instances); Clustering **[PLACEHOLDER: X]** s per round; LLM diagnosis **[PLACEHOLDER: X]** s per cluster; Total overhead **[PLACEHOLDER: X]**% of total experiment time. Expected: IGAS overhead is negligible ($<5\%$) compared to trace generation time.

C Additional Result Tables and Figures

This appendix collects the full specifications of result tables and figures deferred from the main text for space; each will be instantiated with the corresponding artifact once the fullbench jobs complete.

Table 1a—Per-round training accuracy of the two arms (per dataset, 10-seed mean $\pm$ std, 4 rounds); columns: dataset $\times$ arm $\times$ rounds 1–4, plus cumulative failure-pattern counts as a trajectory. Expected: SkillOpt (uniform) is noisier, with faster, more scattered failure-pattern growth (never-repeating failures resist skill consolidation); IGAS's re-verification quota may depress late-round training accuracy, as budget is spent re-verifying historical failures and informative-but-hard instances. Test never enters these in-round curves (Section 3.4); Test scores are annotated as single points at the final-stage winner (BoN winner for IGAS, chain-tail skill for SkillOpt).

Table 2—Rounds required to reach 80% success rate. Expectation: IGAS reaches 80% in fewer rounds than Random. If neither reaches 80%, report the rounds needed to attain the maximum success rate achieved by Random.

Table 3—Final-round training vs. test success rate, showing the generalization gap. Expectation: IGAS achieves a smaller generalization gap thanks to representative sampling.

Table 4—Component ablation removing each utility factor: Full Utility $\hat{u} \cdot h(\hat{u}) \times (1 + \log_2(1 + \text{repr})) + \beta \cdot \sqrt{\ln(N+1)/(\text{repr}+1)}$ (ours); No peak-shaping $\hat{u} \times (1 + \log_2(1 + \text{repr}))$ (raw failure similarity, no boundary focus); No representativeness $\hat{u} \cdot h(\hat{u})$ (no generalization scaling); No failure signal $1 + \log_2(1 + \text{repr})$ (typicality only; equals E3_be_nouncert of Table 5); Random baseline. Expected: Full Utility performs best, and removing the failure signal causes the most degradation (the selector then spends budget on typical-but-already-solved instances).

Table 5—Batch selection ablation (in-matrix single-variable comparison): E3: IGAS (Utility + facility-location coverage, $\beta = 0.3$, ours)—full method; E9_be: uncertainty sampling ($h(\hat{u})$ top-n, point-wise greedy, no coverage-based batch selection)—representative of greedy point-wise selection (no batch redundancy reduction); E8_be: core-set sampling (farthest-point, coverage diversity only, no Utility)—representative of diversity-dominated selection; E3_be_nouncert: IGAS with the failure-signal factor removed (Utility $= 1+\log_2(1+\text{repr})$)—Utility component decomposition. Expected: E3 $>$ E9_be (utility-weighted coverage outperforms greedy point-wise selection, which picks redundant instances and wastes batch budget), E3 $>$ E8_be (Utility ranking is necessary—pure coverage selects diverse but low-information instances); together they show that the **combination** of Utility weights and coverage structure outperforms either alone.

Table 6—Embedding method comparison after experiments.

Table 9—Pluggability factorial experiment results (final test CA %, 3 seeds). Columns: Optimizer Configuration / Random Sampling ($\mu \pm \sigma$) / IGAS Sampling ($\mu \pm \sigma$) / Δ (IGAS−Random) / Relative. Rows: SkillOpt-style (`skillopt`) **[PLACEHOLDER: ...]**; SkillOpt + Cluster-Level Diagnosis (`skillopt_clusterdiag`) **[PLACEHOLDER: ...]**. Expected: IGAS outperforms Random on both configurations (sampling gain $\Delta > 0$), demonstrating the universality of the sampling strategy; cluster-level diagnosis brings an additional diagnosis gain (row-wise difference), demonstrating that the diagnosis granularity contribution is independent of the sampling strategy.

Figure 3—Number of distinct failure patterns over rounds. Expected: pattern count increases initially (discovering new failures) then plateaus (skill matures, fewer new patterns). IGAS should plateau faster.

Table 10—Failure pattern categories discovered, with first-seen round and trace count. Categories: `syntax_error`, `path_escaping`, `file_lock`, `logic_error`, `timeout`, `tool_misuse`, `environment`.

Figure 4—Distribution of IG scores for selected vs. unselected instances. Expected: selected instances have significantly higher IG scores.

Figure 5—Skill evolution trajectory in SBERT embedding space (PCA 2D projection). Each curve traces `SKILL.md`'s semantic position across the rounds of a run (R0→R4 in the main 4-epoch protocol). Random (SkillOpt) vs. IGAS (ours) trajectories are overlaid per dataset. Expected: IGAS trajectories show larger, more consistent directional movement (targeted skill improvement), while Random trajectories exhibit smaller, less consistent shifts. IGAS should also show faster convergence (most movement in early rounds, stabilizing in later rounds).

Table 11—Comparison of dominant failure-mode categories across the six datasets (LiveMathematician-Bench theorem misapplication / SearchQA retrieval-evidence grounding / DocVQA document visual grounding / SpreadsheetBench formula and cross-sheet reference / ALFWorld scene-action mapping / OfficeQA cross-format content grounding). Expected: the Top-3 failure-mode proportions and fix rates (FRR) of each dataset exhibit domain specificity, validating the domain-agnostic nature of cluster-level diagnosis.

Figure 6—Calibration scatter of validation versus Test scores for all BoN-pool candidates (main experiments, 10 seeds; A′ and B groups, annotated by chain position, $y = x$ diagonal). Expected: the B group lies close to the diagonal (well calibrated, an unbiased selection signal); the A′ group lies systematically above it, and **the offset grows monotonically with chain position**—the attribution key, since a pool-difficulty offset would be constant; only a growing offset is attributable to selection-pressure-driven A-specific overfitting. Annotate the Test scores of the $\arg\max$-A′, $\arg\max$-B, and combined-criterion candidates, quantifying the Test cost of single-validation-set $\arg\max$ selection. Methodological statement: this Test evaluation is a **post-hoc diagnostic**, performed after all decisions are complete; Test scores never enter any decision path, and the main results remain the protocol-internal single Test evaluation (Section 3.4).

Figure 7—$\mathrm{rgap}_i = \mathrm{gap}_i - \mathrm{gap}_{\mathrm{first}}$ ($\mathrm{gap} = A' - B$, Section 3.4.2) versus chain position for the same pool (10 seeds overlaid), with the $z \cdot \hat{\sigma}(\mathrm{rgap}_i)$ threshold band: shallow-chain candidates fall below (penalty off; criterion degenerates to pure B $\arg\max$), deep-chain candidates cross it (penalty on). Expected: **rgap rises monotonically with chain position**—directly measurable evidence of inflation accumulation in single-validation-set architectures. Supplementary: final-selection win frequency by chain position (shallow-to-mid positions win at least as often as deep ones, whose apparent advantage is offset by the inflation discount). Evidence 2 traces the inflation trajectory through the internal differential signal, Evidence 1 anchors its magnitude and cost against Test; together with the final-selection ablation (Table 12) they close the mechanism-to-cost loop.

Table 12—Final-selection strategy comparison (replayed on the same logs of the main experiments; the Test segment is evaluated once for the winner only). Strategies compared: (i) **BoN combined criterion** (ours: B as primary signal + A′ probe + paired-bootstrap soft threshold, Section 3.4.2); (ii) **last-round candidate** (no final selection: directly deploy the last accepted skill); (iii) **pure B** $\arg\max$ (no A′ probe: rank by Val_B alone); (iv) **in-round chained A** $\arg\max$ (old-protocol control: select the best candidate directly by in-round Val_A scores; expected to be most contaminated by inflation accumulation, Section 3.4.2). Metric: single Test evaluation score (10 seeds). Expected: BoN $\geq$ pure B $\arg\max \geq$ last round; in-round A $\arg\max$ is significantly worst, and its gap to BoN directly quantifies the cost of inflation accumulation.

Tables of mean $\pm$ std, bootstrap 95% CI, and Wilcoxon p-value for the four effect classes per dataset, to be produced by the aggregation script after the fullbench 120 jobs complete. Intended reporting framing: the cross-arm selected difference (effect 3) as the headline, the cross-arm gain difference (effect 4) as the net protocol effect after controlling for baseline drift; datasets with CIs excluding zero and $p < 0.05$ are counted as significant.