# Not All Correct Responses Are Equal: Behavior Matters in Synthetic Data Generation for Post-Training

Anonymous ACL submission

## Abstract

Post-training pipelines for language models increasingly rely on synthetic data generated through rejection sampling, best-of-$N$ selection, and preference-based filtering. These approaches typically assume that responses with similar quality scores are interchangeable as training supervision. We revisit this assumption by observing that correctness does not imply a single path to a task: under the same requirements, a response may directly answer, decompose a problem, compare alternatives, or ask for clarification. We formalize this distinction through a quality floor ($F$), which specifies what a response must satisfy, and a behavior plan ($B$), which specifies how the task is carried out. We construct synthetic training data using behavior-conditioned responses and compare it with standard best-of-$N$ selection under matched quality floors. Across two Qwen3 backbones under top-1 selection, behavior-conditioned supervision yields modest positive changes in average downstream score relative to matched plain best-of-10 supervision (from $0.491$ to $0.506$ on Qwen3-8B and from $0.553$ to $0.570$ on Qwen3-14B across five benchmarks). The gains are backbone-dependent and not uniformly large, suggesting that scalar quality scores can miss supervision-relevant differences among equally acceptable responses.

## 1 Introduction

Modern language-model post-training increasingly relies on synthetic data, where multiple candidate responses are generated and selected through methods such as rejection sampling, best-of-$N$, preference optimization, and judge-based filtering (Stiennon et al., 2020; Ouyang et al., 2022; Rafailov et al., 2023; Hong et al., 2024; Meng et al., 2024; Wang et al., 2023; Xu et al., 2024a). Despite their differences, these approaches share a common assumption: responses with similar quality scores are largely interchangeable as training supervision. Implicitly, they assume that satisfying task requirements also determines supervision value. However, if multiple valid execution behaviors exist for the same task, scalar quality scores alone may miss differences that matter for learning.

This assumption is natural but potentially incomplete. Correctness does not imply a single way to solve a task. Under the same requirements, a response may directly answer, decompose a problem into steps, compare alternatives, ask clarifying questions, or explicitly state uncertainty and boundaries. Multiple responses may therefore satisfy the same task requirements while differing substantially in how they execute the task. While these responses may appear equivalent under scalar evaluation scores, they may encode different information for learning. This raises a fundamental question for post-training using synthetic data: **when two responses satisfy the same task requirements, are they equally useful as supervision?**

We study this question through a two-level view of responses. A *quality floor* $F$ specifies the minimum requirements (e.g., those operationalized via evaluation rubrics) a response must satisfy to be acceptable. A *behavior plan* $B$ specifies how the task is carried out. The quality floor captures *what* must be achieved, while the behavior plan captures *how* the task is executed.

To test this hypothesis, we intervene on synthetic-data construction. For each task, a teacher model generates multiple candidate responses conditioned on diverse hidden behavior plans while holding fixed the quality floor, candidate budget, evaluation procedure, selection rule, and student training recipe. We refer to this setup as a *matched protocol*. As a baseline, we compare against standard best-of-$N$, which generates candidates without behavior conditioning but uses the same evaluation and selection process. After selection, both pipelines produce ordinary supervision

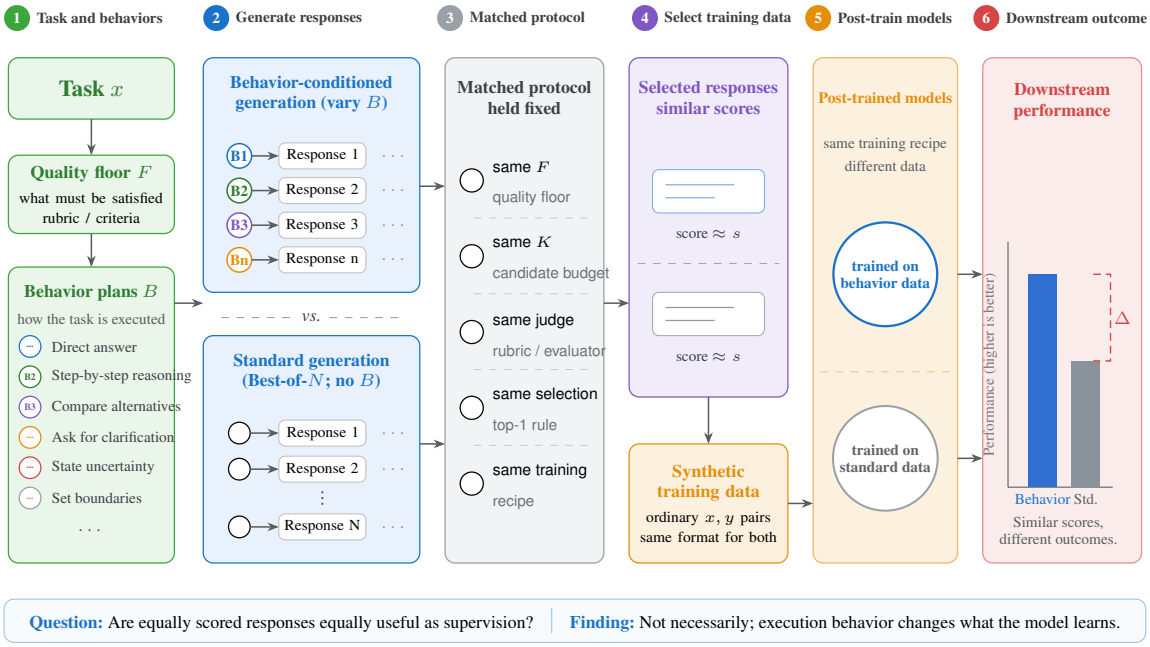


Figure 1: Overview of our framework. A fixed quality floor $F$ specifies *what* a response must satisfy, while multiple behavior plans $B$ specify *how* the task is carried out, so the same task admits several acceptable responses. We construct synthetic supervision by generating responses under different behavior plans and compare this with standard best-of-$N$ generation under a *matched protocol* that holds the candidate budget, evaluator, selection rule, and student training recipe fixed; after selection, both pipelines yield ordinary $(x, y)$ pairs in the same format. Because both pipelines use the same evaluator and selection rule, any downstream difference after post-training exposes a *learnability gap*: equally acceptable responses need not be equally useful supervision.

and use identical student training procedures. The only difference lies in how the synthetic responses are generated before selection.

Figure 1 illustrates this process. A fixed quality floor admits multiple possible execution behaviors. Candidate responses generated under different behaviors are filtered through an identical selection pipeline and transformed into standard supervision for post-training. If responses satisfying similar requirements produce different downstream models, then supervision quality depends not only on correctness but also on how correctness is achieved.

We evaluate this setting on 2,000 medical tasks paired with item-level scoring rubrics. In our main experiment, both lanes use the same response teacher, candidate budget, decoding settings, judge, selection rule, student training recipe, and top-1 supervision size; only behavior conditioning during candidate generation differs. Behavior-conditioned supervision produces modest positive average changes on the two evaluated backbones, from 0.491 to 0.506 on Qwen3-8B and from 0.553 to 0.570 on Qwen3-14B. We therefore read the result as a scoped, backbone-sensitive signal rather than as a uniformly large improvement recipe. Candidate-set diagnostics further show that behavior-conditioned generation produces broader execution diversity before selection.

**Contributions.** Our contributions are threefold:

- We distinguish **task requirements** from **execution behaviors** through a two-level framework including a quality floor $F$ and a behavior plan $B$, making explicit an assumption underlying selection-based post-training pipelines.
- We introduce **behavior-conditioned synthetic data construction** as a matched test for studying the role of execution behaviors in post-training while keeping evaluation and training procedures fixed.
- We identify a **learnability gap**: under a matched selection and training protocol, changing the execution paths available before selection can change downstream performance after training. This suggests that scalar quality scores can miss supervision-relevant differences among equally acceptable responses.

## 2 Related Work

**Selection, verification, and best-of-$N$ supervision.** Many post-training pipelines use scalar quality or preference signals to select, rank, or optimize model outputs, including rejection-sampling fine-tuning (Ouyang et al., 2022; Touvron et al., 2023), AI feedback and constitutional approaches (Bai et al., 2022), instruction-tuning and synthetic-data generation (Taori et al., 2023; Wang et al., 2023; Xu et al., 2024b,a), preference-based optimization (Rafailov et al., 2023; Hong et al., 2024; Meng et al., 2024; Ziegler et al., 2019; Stiennon et al., 2020), verifier or reward-model reranking (Cobbe et al., 2021; Lightman et al., 2023; Lambert et al., 2024; Malik et al., 2026), and analyses of best-of-$N$ alignment policies (Beirami et al., 2024; Sessa et al., 2025). Process-supervision work further evaluates whether a reasoning trajectory is valid step by step (Zheng et al., 2025; Song et al., 2025). These lines primarily improve how candidates are scored, verified, reranked, optimized, or distilled, rather than testing the path-equivalence assumption: whether candidates that clear the same quality floor are interchangeable as supervision. We target the construction step before selection by holding the quality floor fixed and using the same teacher, candidate budget, judge-side scoring, selection rule, student backbone, and training recipe while varying only the teacher's generation-time conditioning.

**Behavior control and prompting.** Controllable generation and prompting research shows that model responses can be steered through attributes, styles, personas, roles, system messages, prefixes, or task instructions (Keskar et al., 2019; Dathathri et al., 2020; Liu et al., 2021; Li and Liang, 2021; Zhang et al., 2018; Wei et al., 2022; Sanh et al., 2022). These methods establish that generation behavior can be steered, but they typically study user- or operator-visible control of model outputs. Our behavior plans serve a different role: they are hidden construction variables used to shape the candidate pool before judge-side selection, not an interface exposed to the student or deployment user. This distinction lets us study whether execution-path variation under a fixed quality floor changes the supervision signal selected for student training.

## 3 Problem Formulation

### 3.1 Quality Versus Behavior

Selection-based post-training evaluates each candidate response $y$ to a task $x_i$ using a scalar score $J(y; r_i) \in [0, 1]$ under rubric $r_i$. While effective for deciding admissibility (clearing a quality floor), this score is incomplete for supervision because selected responses also act as training demonstrations. Crucially, scalar scores conflate *quality* with *behavior* (the execution path taken). Two responses may be equally admissible via different paths (Section 1), yet treating them as interchangeable teaches different response contracts. We term this the *path-equivalence assumption*: assuming candidates clearing the same floor provide equivalent supervision. This assumption becomes risky when ranking relies on a single source rubric; maximizing $J(\cdot; r_i)$ may reinforce a specific contract that harms downstream tasks requiring different ones. We call this failure mode *source-contract over-specialization*.

### 3.2 Quality Floor and Behavior Plan

We formalize this distinction via two components. The evaluator-side *quality floor* $F_i$ defines the minimum acceptable requirements for task $x_i$, operationalized via judge-side rubric $r_i$ (e.g., factuality, safety) rather than generation-time instructions. Conversely, a hidden generation-time *behavior plan* $B$ dictates *how* a candidate achieves this floor (e.g., via step-by-step reasoning, direct answers, or requesting clarification). Figure 2 illustrates this on a medical query: candidates meet the same floor via different behavior plans, highlighting the floor–behavior distinction rather than clinical superiority. Table 1 demonstrates this at scale: even when different models correctly answer the same prompt, their realized behaviors (derived from post-hoc judge descriptors) diverge significantly, further motivating the path-equivalence problem.

This distinction yields a falsifiable prediction: if floor scores sufficiently capture supervision value, training on equally-scored candidates generated via different paths should yield identical downstream student performance. If holding the learner fixed still produces downstream differences, these must stem from the candidates' execution paths. We test for this divergence—which we term the *learnability gap* when it appears under the matched protocol—in Section 6.

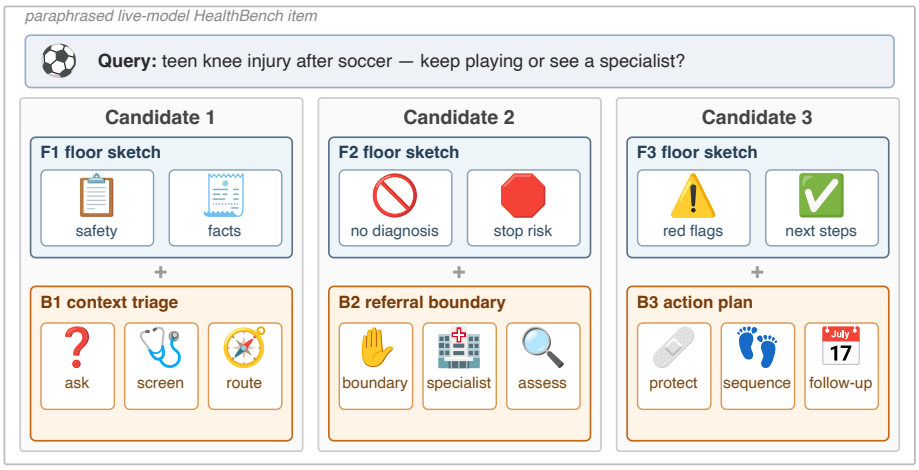


Figure 2: Qualitative illustration of paired floor-oriented guidance and behavior plans on a live-model HealthBench item, paraphrased for readability. The figure is illustrative and is not medical advice.

Table 1: Live-model diagnostic of realized behavior paths, used as motivation for the path-equivalence problem. The annotations are post-hoc, judge-derived behavior descriptors over three models and five benchmarks; floor/fact labels are excluded. The same-correctness block restricts to same-item triads where all three models are correct. This diagnostic is not evidence about the internals of the behavior-conditioned generator.

*Source of behavior-descriptor variation (%)*

| Descriptor | Benchmark | Model | Inter. | Descriptor | Benchmark | Model | Inter. |
|---|---|---|---|---|---|---|---|
| All behavior descriptors | 67.0 | 16.0 | 17.0 | Directness | 29 | 37 | 34 |
| Brevity | 50 | 26 | 24 | Decomp. depth | 54 | 19 | 27 |
| Initiative | 66 | 18 | 16 | Uncertainty | 97 | 1 | 2 |
| Boundary | 99 | 0 | 1 | Warmth | 95 | 3 | 2 |

*Same-correctness triads still differ*

| Benchmark | All-correct $n$ | Dist. $\geq .20$ (%) | Any diff. (%) | Active-count $\Delta \geq 2$ (%) |
|---|---|---|---|---|
| Math500 | 434 | 86.9 | 98.8 | 100.0 |
| MedQA | 1,072 | 71.2 | 98.8 | 99.9 |
| GPQA | 92 | 58.7 | 97.8 | 72.8 |
| BBH | 624 | 22.6 | 75.0 | 58.8 |

Same-correctness triads are reported only for benchmarks with item-level correctness labels in this diagnostic; active count denotes the number of active behavior descriptors.

## 4 Method

Given a task $x_i$ with rubric $r_i$, behavior-conditioned synthetic data generation produces a set of $K$ candidate responses under the same task requirements while differing in how the task is executed. We treat this as a controlled intervention on the candidate-set paths available under a fixed evaluator quality floor: we vary the teacher's generation-time guidance under the matched protocol. This isolates candidate construction, rather than student learning, as the manipulated factor.

**Quality floor and behavior plans.** We instantiate the two-level contract from Section 3 at the candidate level: each behavior-conditioned candidate receives a rubric-blind floor sketch $F_{i,k}$ and a sampled behavior plan $b_{i,k}$, merged into hidden guidance $g_{i,k}$. The fixed evaluator-side floor $F_i$ and its rubric $r_i$ are unchanged and remain judge-side information; the floor sketch $F_{i,k}$ is a generation-time object, shown as $F_k$ in Figure 2 and distinct from $F_i$ and $r_i$.

Each behavior plan draws from a compact vocabulary of 25 behavior atoms: seven interaction atoms, nine situation-state atoms, and nine ordinal intensity atoms (Appendix Table 8). These atoms are engineering labels for a generator-side representation, not claims about psychological constructs. They are designed to induce meaningfully different execution strategies rather than superficial paraphrases. The rubric-blind floor sketches and behavior plans together form the hidden generation

guidance; these sketches do not expose the original rubric or change the fixed evaluator-side floor.

The field routing is a construction rule rather than a behavioral ontology. The quality floor records requirements that must hold regardless of execution; the interaction mode records the turn-level contract; and the intensity fields record default tendencies on individual dimensions. For example, clarification can appear as a quality-floor requirement when missing information cannot be assumed, as an interaction mode when the candidate should elicit information first, or as a high clarification-threshold tendency. This routing keeps externally checked requirements separate from hidden execution choices; Appendix B states the convention in full.

**Behavior-conditioned candidate generation.** For each task, a hidden planner samples $K$ guidance packages $g_{i,1}, \ldots, g_{i,K}$, each combining the rubric-blind floor-oriented fields inferred from the visible task (a generic response contract, generic floor labels, coverage targets, and specificity slots) with a behavior plan $b_{i,k}$. The sampled packages span contrasting but plausible paths to the same task, and the planner is explicitly instructed not to copy or invent scoring criteria.

We generate two independent candidate pools per task. The behavior-conditioned lane merges the floor sketch and behavior plan into the hidden guidance package and conditions the teacher on it; the plain best-of-$N$ lane conditions the teacher on the task input alone:

$$\begin{aligned} g_{i,k} &= \mathrm{merge}(F_{i,k}, b_{i,k}), \\ y^{B}_{i,k} &\sim p_{\text{teacher}}(y \mid x_i, g_{i,k}), \\ y^{P}_{i,k} &\sim p_{\text{teacher}}(y \mid x_i). \end{aligned} \tag{1}$$

Here, the lane index $a$ takes values $B$ (behavior-conditioned) and $P$ (plain). The index $i \in [n]$ ranges over the $n$ training tasks, and $k \in [K]$ over the $K$ candidates per task. The two lanes differ only in the teacher's conditioning: the behavior-conditioned lane ($a{=}B$) samples under $g_{i,k}$, while the plain best-of-$N$ lane ($a{=}P$) samples under the task input alone. The shared index $k$ does not pair a behavior-conditioned candidate with a plain one; the pools are sampled independently and matched only through $K$, the judge, and the selection rule. The guidance package is used only during candidate generation and is not observed by the student.

**Judge scoring and candidate selection.** A separate LLM judge scores every candidate in both pools, and the top-scoring index in each lane is retained:

$$\begin{aligned} s^{a}_{i,k} &= J(y^{a}_{i,k}; r_i) \in [0,1], \\ I^{a}_{i} &= \mathrm{Top}_1\big(\{s^{a}_{i,k}\}_{k\in[K]}\big). \end{aligned} \tag{2}$$

In Eq. (2), $J$ checks each rubric item and aggregates the per-item verdicts into the normalized score $s^{a}_{i,k} \in [0,1]$, and $\mathrm{Top}_1$ returns the index of the highest-scoring candidate. Selection is rubric-primary: candidates are ranked by this score, with additional quality signals used only for tie-breaking. The rubric items $r_i$ never enter either teacher's conditioning in Eq. (1) and are used only by $J$. The behavior-conditioned and plain lanes use exactly the same judge and selection procedure, so downstream differences cannot arise from changes in scoring or filtering.

**Student training.** The retained indices define the per-lane supervised dataset $\mathcal{D}^a = \{(x_i, y^{a}_{i,k}) : i \in [n],\ k \in I^{a}_{i}\}$, used for standard supervised fine-tuning of the student parameters $\theta$ to produce $\theta^a$. The student receives only plain input–output pairs, never behavior plans, rubrics, teacher scores, or judge rationales.

**Matched plain best-of-$N$ baseline.** The plain best-of-$N$ lane ($a{=}P$) is the matched baseline that isolates generation-time conditioning. The prompt visibility invariant is simple: the plain teacher sees only $x_i$; the behavior-conditioned teacher sees $x_i$ and $g_{i,k}$; only the judge sees the rubric items $r_i$; and the student sees only the selected $(x, y)$ pairs. Every other component is the matched protocol introduced in Section 1: quality floor, candidate budget $K$, teacher, judge-side scoring, selection procedure, student backbone, and training recipe are all shared, and the planner is likewise rubric-blind. Appendix Table 7 gives the field-level audit.

## 5 Experimental Setup

**Training corpus.** Our experiments use the medical subset (2,000 tasks) of RubricHub (Li et al., 2026); each task is paired with multiple weighted rubric items defining evaluation requirements. We use one source domain to test whether supervision-quality differences arise under a fixed rubric system and transfer across benchmarks with different domains and answer contracts.

Table 2: Top-1 downstream student performance under matched best-of-$N$ candidate construction on the 2,000-task medical corpus. Rubric score is the mean selected-candidate rubric score used during data construction and is not applicable to the base rows. Within each backbone, the plain and behavior-conditioned rows share the response teacher, candidate budget, decoding settings, judge, selection rule, and SFT recipe; only behavior conditioning during candidate generation differs. Avg. averages the five downstream benchmark columns. Single-run estimates; no SFT seed variance reported.

| Backbone | Supervision | Rubric score | Math500 | MedQA | GPQA | BBH | Health Bench | Avg. |
|---|---|---|---|---|---|---|---|---|
| Qwen3-8B | Base | — | 0.519 | 0.606 | 0.338 | 0.306 | 0.270 | 0.408 |
| Qwen3-8B | Plain best-of-10 | 0.437 | **0.651** | 0.609 | **0.389** | 0.504 | 0.304 | 0.491 |
| Qwen3-8B | Behavior-conditioned | **0.661** | 0.626 | **0.671** | 0.379 | **0.522** | **0.333** | **0.506** |
| Qwen3-14B | Base | — | 0.662 | 0.679 | 0.424 | 0.567 | 0.288 | 0.524 |
| Qwen3-14B | Plain best-of-10 | 0.437 | 0.660 | **0.723** | 0.475 | 0.578 | 0.330 | 0.553 |
| Qwen3-14B | Behavior-conditioned | **0.661** | **0.703** | 0.705 | **0.495** | **0.593** | **0.351** | **0.570** |

**Candidate construction settings.** Both lanes generate $K=10$ candidates per task under identical decoding settings. The behavior-conditioned lane additionally samples planner guidance packages, but the response teacher, candidate budget, judge, and selection rule are shared. A single-pass judge evaluates every candidate against the task rubric, using the same rubric-primary top-1 selection rule in both lanes.

**Models and training.** Teacher responses are generated by `MiniMax-M2_5`, served through an OpenAI-compatible endpoint. Rubric scoring is performed by a separate `gpt-oss-120b` judge; rubric information is exposed only to this judge and never to the teacher. We train Qwen3-8B and Qwen3-14B base students (Qwen Team, 2025) using standard supervised fine-tuning (SFT) (Ouyang et al., 2022) on the `conversations` format. Students receive only input-output pairs and do not observe behavior plans, rubrics, scores, or judge rationales. Within each backbone, the plain and behavior-conditioned arms use the same optimization setup, tokenization, and training recipe; decoding settings, judge protocol, and SFT hyperparameters are listed in Appendix A.

**Evaluation protocol.** We evaluate on five downstream benchmarks: Math500 (Lightman et al., 2023), MedQA (Jin et al., 2021), GPQA (Rein et al., 2023), BBH (Suzgun et al., 2022), and HealthBench (HealthBench Team, 2025). Two of these (MedQA, HealthBench) share the medical domain of the training corpus, while three (Math500, GPQA, BBH) lie outside it; this split lets us separate domain transfer from answer-contract transfer, since the benchmarks also differ in answer form (open-ended generation for Math500 and HealthBench versus multiple-choice or short-answer for MedQA, GPQA, and BBH). The primary results compare plain best-of-$N$ top-1 and behavior-conditioned top-1 supervision on two Qwen3 backbones. Across all compared arms, only the teacher's generation-time conditioning differs within a matched backbone (Appendix Table 7).

## 6 Main Results: Matched Top-1 Comparison

We begin by asking whether candidates selected under the same quality floor are equivalent as supervision. Table 2 reports the primary top-1 setting across two Qwen3 backbones. The comparison is within each backbone: both rows use the same source corpus, response teacher, candidate budget, decoding settings, judge, selection rule, and SFT recipe. The only manipulated factor is whether candidate generation receives hidden behavior plans before the shared selector ranks the pool.

Behavior-conditioned supervision yields positive average changes on both reported backbones, but the effects are small and backbone-dependent rather than uniformly large. On Qwen3-8B, the average changes from $0.491$ under plain supervision to $0.506$ under behavior-conditioned supervision, with gains on MedQA, BBH, and HealthBench but decreases on Math500 and GPQA. On Qwen3-14B, the average changes from $0.553$ to $0.570$, with gains on four of five benchmarks but a decrease on MedQA. This supports the main claim in a scoped form: changing the execution paths available to the selector can change downstream learnability even when the student-side training interface remains ordinary SFT on $(x, y)$ pairs, but the observed advantage should be read as model-sensitive rather than as a guaranteed improvement.

Two qualifications keep the claim scoped. First, the gains are backbone-dependent and the per-benchmark pattern is mixed, so we do not treat either the direction on every benchmark or the magnitude of the average change as invariant. Second, the selected-candidate rubric score is a construction-time diagnostic, not the downstream metric itself. We therefore read Table 2 as evidence that changing the execution paths available to the selector can change the supervision distribution that the student learns from, rather than as a claim that behavior conditioning uniformly improves every benchmark cell.

## 7 Diagnosing Candidate-Set Structure

The previous section reported backbone-sensitive downstream differences under the matched protocol. We now examine whether the two candidate-generation strategies produce candidate pools with different structural properties. Our analysis is descriptive rather than causal: we characterize the candidate sets presented to the selection mechanism rather than explaining why any particular candidate is selected. Concretely, we ask whether behavior conditioning widens the space of admissible execution paths that the floor-based selector ranks over.

**Candidate diversity.** Table 3 contrasts the candidate pools produced by the two strategies under the same tasks, teacher, and candidate budget. Behavior conditioning widens the pool along two distinct axes. The rubric-score spread and the top-minus-mean score are both larger, so the pool spans a wider range of candidate quality and offers a larger selectable upper tail to the floor-based selector. Pairwise token-set (Jaccard) and token-count (cosine) distances are also larger, so the candidates differ in surface form, not only in score. Behavior conditioning therefore creates a broader candidate distribution before selection rather than merely more paraphrases of a single response. Within the behavior-conditioned pool, each task covers on average $5.06$ distinct interaction atoms; we report this as a within-pool plan-metadata diagnostic, since plain best-of-$N$ candidates carry no behavior-plan metadata and the count has no cross-method counterpart.

**Rubric coverage within the behavior-conditioned pool.** As a within-method selection diagnostic, top-ranked candidates inside the behavior-conditioned pool cover more rubric content than bottom-ranked candidates from the same pool, with average gains of $124.4$ rubric points and $17.2$ rubric items per task. The added content concentrates on task-relevant rubric requirements rather than incidental detail. This indicates that, inside the behavior-conditioned pool, the floor-based selector favors candidates that more fully instantiate the task's requirements rather than arbitrary behavioral variation; the pattern is consistent with execution paths mattering for selection, but we do not extend it to a cross-method coverage claim.

Table 3: Candidate-pool structure under matched generation over the $2{,}000$ medical training tasks (same tasks, teacher, candidate budget, judge-side scoring, and selection rule; only behavior conditioning differs). Rubric-score spread and top-minus-mean are computed on normalized rubric scores within each task's candidate pool; Jaccard and cosine are mean pairwise distances between candidates. Higher values indicate a wider pool. These describe the candidate sets, not downstream SFT results.

| Pool metric | Behavior | Plain |
|---|---|---|
| Rubric-score spread | **0.567** | 0.392 |
| Top-minus-mean score | **0.287** | 0.212 |
| Pairwise Jaccard dist. | **0.717** | 0.585 |
| Pairwise cosine dist. | **0.370** | 0.234 |

**Interaction-atom signature among selected candidates.** Figure 3 shows the most interpretable categorical slice among the 25 atoms (the full 25-atom top-1 audit is in Appendix Table 9): the interaction atom assigned to top-1 selected candidates and bottom-1 candidates within the behavior-conditioned pool. This is a sharp selection-boundary diagnostic: rubric scores average $0.661$ for top-1 and $0.094$ for bottom-1 candidates. Selected candidates concentrate in explanation-oriented execution (49.7%). Bottom-ranked candidates are dominated by clarification-oriented execution (63.3%). This pattern is floor-relative: it says that, under this medical quality floor and rubric-based selector, clarification-first candidates are less often selected; it does not imply that clarification is intrinsically worse on tasks that truly require eliciting missing information.

**Task-plan matching ablation.** The diversity diagnostics above show that behavior-conditioned generation widens the candidate pool, but they do not by themselves show that the behavior plans must be matched to the task. We therefore run a

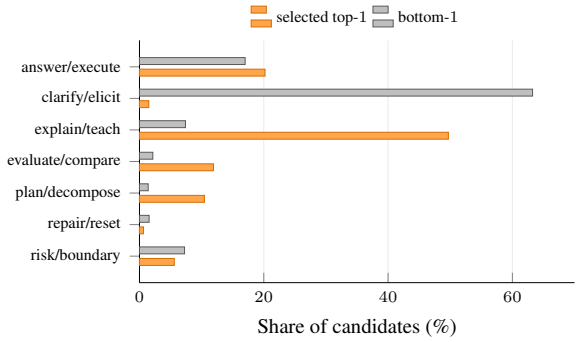


Figure 3: Interaction-atom slice of the unified 25-atom behavior plan $B$, measured within the behavior-conditioned pool.

Table 4: HealthBench plan-matching ablation. All rows share tasks, teacher, judge, budget, and selection rule; only behavior-plan conditioning changes. Values are selected-candidate rubric scores and threshold pass rates, not downstream SFT results.

| Condition | top-1 | $\geq .7$ | $\geq .8$ |
|---|---|---|---|
| Behavior-conditioned | **0.780** | **69.8** | **55.4** |
| Plain best-of-10 | 0.719 | 58.6 | 44.4 |
| Task-shuffled plans | 0.599 | 40.2 | 28.0 |
| Atom-shuffled plans | 0.617 | 41.0 | 27.8 |

HealthBench construction-level control that keeps the same 500 tasks, candidate budget, response teacher, rubric judge, and selection rule, but breaks the plan–task link. In the task-shuffled condition, each task receives complete behavior plans sampled for other tasks. In the atom-shuffled condition, behavior-plan atoms are locally recombined from non-target tasks.

Table 4 shows that task-matched behavior-conditioned generation produces stronger selectable candidates than either shuffled control. The top-1 score drops from 0.780 to 0.599 under task shuffling and to 0.617 under atom shuffling, while the $\geq .8$ pass rate falls by roughly half. The useful variation is therefore task-conditioned rather than arbitrary expansion of the sampling space.

**Other construction checks.** Two further HealthBench checks vary the construction setting without training a separate student (Table 5). Behavior-conditioned generation remains higher under an alternate rubric schema (0.751 vs. 0.724), but adding an explicit rule layer reverses the sign (0.713 vs. 0.759). We read both as construction-level evidence: the effect is neither tied only to the source rubric nor explained by adding more behavioral constraints.

Table 5: Additional HealthBench construction checks under an alternate rubric schema. Rows report selected-candidate rubric score; these are construction-level checks, not downstream SFT results.

| Construction variant | Behavior | Plain |
|---|---|---|
| Alternate rubric schema | **0.751** | 0.724 |
| + rule-based constraints | 0.713 | **0.759** |

## 8 Conclusion

We asked whether responses that clear the same quality floor are equivalent as supervision. Under a matched protocol, behavior-conditioned construction yields small, backbone-dependent positive changes in top-1 downstream averages while also changing the candidate pool available to selection. The HealthBench construction checks further show that useful behavior variation must be task-matched and is not explained by a particular rubric design or by simply adding more constraints. A quality floor decides whether a response is admissible, but not which admissible execution path becomes learnable supervision. Not every task needs the same behavior.

## Limitations

This study focuses on matched best-of-$N$ candidate construction under the matched protocol (Section 4); we do not compare against reinforcement learning or online policy-learning approaches, nor vary training recipes or candidate budgets. The reported results are single-run estimates from a controlled setting with a 2,000-task medical corpus, two Qwen3 student backbones, item-level scoring rubrics, and five benchmarks, so we treat magnitudes as setup-specific and do not claim that behavior-conditioned construction reliably dominates plain best-of-$N$ in every setting. The behavior-plan schema is compact and manually specified; we use it to expose execution-path variation under a fixed quality floor, not as a complete taxonomy of model behavior or a comprehensive behavioral benchmark.

Finally, the candidate-pool and HealthBench analyses are diagnostic rather than a complete causal mechanism for downstream learning. They show that task-matched behavior plans change the selectable candidates presented to a fixed rubric-based selector, but they do not identify which properties of the selected responses each student internalizes during SFT.

## References


Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. (2022). Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*.

Beirami, A., Agarwal, A., Berant, J., D'Amour, A., Eisenstein, J., Nagpal, C., and Suresh, A. T. (2024). Theoretical guarantees on the best-of-n alignment policy. *arXiv preprint arXiv:2401.01879*.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., and Liu, R. (2020). Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

HealthBench Team (OpenAI). (2025). HealthBench: Evaluating large language models towards improved human health. *OpenAI Technical Report*.

Ho, N., Schmid, L., and Yun, S.-Y. (2023). Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.

Hong, J., Lee, N., and Thorne, J. (2024). ORPO: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Jin, D., Pan, E., Oufattole, N., Weng, W.-H., Fang, H., and Szolovits, P. (2021). What disease does this patient have? A large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Keskar, N. S., McCann, B., Varshney, L. R., Xiong, C., and Socher, R. (2019). CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Kim, S., Suk, J., Longpre, S., Lin, B. Y., Shin, J., Welleck, S., Neubig, G., Lee, M., Lee, K., and Seo, M. (2024). Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

Lambert, N., Pyatkin, V., Morrison, J., Miranda, L. J., Lin, B. Y., Chandu, K., Dziri, N., Kumar, S., Zick, T., Choi, Y., Smith, N. A., and Hajishirzi, H. (2024). RewardBench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

Li, S., Zhao, J., Wei, M., Ren, H., Zhou, Y., Yang, J., Liu, S., Zhang, K., and Chen, W. (2026). RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. *arXiv preprint arXiv:2601.08430*.

Li, X. L. and Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. (2023). Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Liu, A., Sap, M., Lu, X., Swayamdipta, S., Bhagavatula, C., Smith, N. A., and Choi, Y. (2021). DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Malik, S., Pyatkin, V., Land, S., Morrison, J., Smith, N. A., Hajishirzi, H., et al. (2026). RewardBench 2: Advancing reward model evaluation. In *International Conference on Learning Representations*.

Meng, Y., Xia, M., and Chen, D. (2024). SimPO: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems*.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. (2021). WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Qwen Team. (2025). Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. (2023). Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. (2023). GPQA: A graduate-level Google-proof Q&A benchmark. *arXiv preprint arXiv:2311.12022*.

Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Le Scao, T., Raja, A., et al. (2022). Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Sessa, P. G., Dadashi, R., Hussenot, L., Ferret, J., Vieillard, N., Rame, A., Shahriari, B., Perrin, S., Friesen, A., Cideron, G., et al. (2025). BOND: Aligning LLMs with best-of-$N$ distillation. In *International Conference on Learning Representations*.

Song, M., Su, Z., Qu, X., Zhou, J., and Cheng, Y. (2025). PRMBench: A fine-grained and challenging benchmark for process-level reward models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. (2020). Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*.

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q. V., Chi, E. H., Zhou, D., et al. (2022). Challenging BIG-Bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. (2023). Stanford Alpaca: An instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. (2023). Self-Instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. (2022). Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Xu, Z., Jiang, F., Niu, L., Deng, Y., Poovendran, R., Choi, Y., and Lin, B. Y. (2024a). Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. *arXiv preprint arXiv:2406.08464*.

Xu, C., Sun, Q., Zheng, K., Geng, X., Zhao, P., Feng, J., Tao, C., and Jiang, D. (2024b). WizardLM: Empowering large pre-trained language models to follow complex instructions. In *International Conference on Learning Representations*.

Yuan, Z., Yuan, H., Li, C., Dong, G., Lu, K., Tan, C., Zhou, C., and Zhou, J. (2023). Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., and Weston, J. (2018). Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*.

Zheng, C., Zhang, Z., Zhang, B., Lin, R., Lu, K., Yu, B., Liu, D., Zhou, J., and Lin, J. (2025). ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*.

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., Ma, X., Efrat, A., Yu, P., Yu, L., Zhang, S., Ghosh, G., Lewis, M., Zettlemoyer, L., and Levy, O. (2023). LIMA: Less is more for alignment. In *Advances in Neural Information Processing Systems*.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. (2019). Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

# A Reproducibility and Prompt Visibility

This appendix collects the audit material needed to reproduce the matched comparison and verify the visibility boundary. Table 6 summarizes the candidate-construction and student-training settings used in the main 2,000-task SFT experiments. Both the behavior-conditioned and the plain best-of-$N$ pipelines share the same response teacher, the same rubric judge, the same candidate budget, and the same rubric-primary selection rule; they differ only in what enters the teacher prompt (Table 7). The HealthBench construction checks use the same response teacher and judge but do not train additional students. The training stack is LLaMA-Factory full-parameter SFT on top of DeepSpeed ZeRO-3; remaining optimizer, scheduler, and warmup details follow the LLaMA-Factory full-SFT default recipe pinned to the training YAML and will be released alongside the training scripts in the camera-ready version.

Table 6: Implementation details for the main candidate construction and student SFT experiments. Settings are shared across the behavior-conditioned and plain best-of-$N$ pipelines unless noted otherwise.

| | |
|---|---|
| *Candidate construction* | |
| Candidate budget $K$ | 10 per task |
| Planner temperature | 0.2 (behavior-conditioned lane) |
| Response teacher | `MiniMax-M2_5` |
| Teacher response temperature | 0.2 (both lanes) |
| Rubric judge | `gpt-oss-120b` |
| Judge temperature | 0.2 for main 2,000-task results and HealthBench construction checks (Sec. 7) |
| Judge protocol | single pass per candidate; per-rubric PRESENT/NOT_PRESENT verdicts aggregated to a normalized $[0, 1]$ score |
| Selection rule | rubric-primary top-1; ties broken by judge-side overall quality |
| *Student SFT* | |
| Backbones | Qwen3-8B base; Qwen3-14B base (Qwen Team, 2025) |
| Framework | LLaMA-Factory + DeepSpeed ZeRO-3 |
| Finetuning type | full-parameter (no LoRA) |
| Hardware | 4× H200 GPUs |
| Per-device batch size | matched within backbone |
| Gradient accumulation | matched within backbone |
| Max sequence length | 8,192 |
| Gradient checkpointing | enabled |
| Epochs | 3 |
| Optimization steps | backbone dependent |
| Precision | bfloat16 |

Table 7: Prompt visibility audit for the matched generation lanes. Both teachers receive the task input and neither teacher receives rubric items; only the judge sees the rubric. The behavior-conditioned lane additionally receives the planner-derived guidance package $g_{i,k}$.

| Prompt field | Plain | Behavior-conditioned |
|---|---|---|
| task input $x_i$ | ✓ | ✓ |
| behavior plan $b_{i,k}$ | | ✓ |
| answer contract | | ✓ |
| generic floor labels | | ✓ |
| coverage targets | | ✓ |
| specificity slots | | ✓ |
| rubric items $r_i$ | | |

# B Behavior-Plan Schema and Routing

This appendix states the hidden planner schema, the field-routing convention used to avoid overlapping labels, and two example plans. The visibility boundary is defined in Section 4 and audited at the prompt-field level in Table 7.

Table 8: Complete behavior-plan atom inventory used by the hidden planner. These are generator-side engineering labels, not psychological constructs. Behavior plans are visible only to the teacher during candidate generation; selection remains rubric-based.

| Atom set | Values and role |
|---|---|
| Interaction | answer/execute; clarify/elicit; evaluate/compare; explain/teach; plan/decompose; repair/reset; risk/boundary. These specify the turn-level execution contract. |
| Situation state | clean; underspecified; ambiguous intent; common-ground gap; conflicting constraints; high-stakes/risk; prior misalignment; repair needed; abrupt frame shift. These describe task-local context that changes which execution path is appropriate. |
| Ordinal intensity | directness; brevity; initiative; uncertainty calibration; assumption management; clarification threshold; decomposition depth; warmth; boundary firmness, each on a 0/1/2 scale. These specify default tendencies on individual execution dimensions. |

**Field routing rule.** A common concern with multi-field plans is that the same surface phenomenon can plausibly be assigned to multiple fields. The planner fixes the assignment by anchoring each field to a distinct question: *What is the agent's default tendency on this dimension?* (discrete policy intensities); *What contract is this turn enacting?* (`interaction_mode`); *What must hold regardless of the executed behavior?* (quality floor). For example, clarification touches all three: a high `clarification_threshold` is a tendency to ask; `interaction_mode=clarify` is the contract for the current turn; and a `clarification_trigger` entry in the quality floor is the constraint that

missing-but-needed information cannot be silently assumed. The routing is therefore not a list of overlapping labels but a construction convention for keeping externally checked requirements separate from hidden execution choices.

**Example plan: direct execution of a well-specified task.** A short, well-specified task with no missing information and no obvious risk. The intended candidate is direct, brief, and rubric-aligned, with no clarification.

```
Quality floor:
  - factual_correctness
  - required_coverage
  - instruction_following
Behavior plan:
  interaction_mode: answer
  directness: 2
  brevity: 1
  initiative: 1
  uncertainty_calibration: 1
  clarification_threshold: 0
  assumption_management: 1
  decomposition_depth: 1
  warmth: 1
  boundary_firmness: 1
```

**Example plan: clarification under risk and underspecification.** A task in which the user's request is missing details that materially affect the answer, and the domain has non-trivial downside risk. The intended candidate elicits the missing information first, surfaces key assumptions explicitly, and only then proceeds.

```
Quality floor:
  - factual_correctness
  - required_coverage
  - uncertainty_calibration
  - clarification_trigger
Behavior plan:
  interaction_mode: clarify
  directness: 1
  brevity: 0
  initiative: 2
  uncertainty_calibration: 2
  clarification_threshold: 2
  assumption_management: 2
  decomposition_depth: 2
  warmth: 1
  boundary_firmness: 1
```

The two plans make the contrast concrete: the same teacher, given the same task requirements on the same input, produces qualitatively different candidates depending on which plan it is conditioned on. The judge then decides which of these candidates best satisfies the quality floor on the actual task.

**Design calibration.** Pilot iterations of the planner prompt with more restrictive behavior constraints (broader forbidden-behavior layers, denser boundary requirements) produced lower top-1 rubric scores on a held-out probe than the version used in the main experiments. The rule-layer ablation in Section 7 is consistent with this pilot observation: behavior constraints must be task-grounded; additional constraints are not automatically beneficial.

## C Full Behavior-Atom Diagnostics

Table 9 reports the full 25-atom selected-minus-rejected diagnostic behind Figure 3. It is a within-pool mechanism diagnostic: the table describes which hidden behavior-plan atoms are enriched among candidates selected by the rubric-based selector, not a separate downstream SFT result.

Table 9: Full selected-minus-rejected enrichment audit over the 25 behavior atoms in $B$. Rows are sorted by absolute top-1 contrast rather than by any legacy grouping. For categorical atoms, the unit is percentage-point share; for ordinal intensity atoms, mean differences are normalized by the 0–2 scale and reported as percentage-of-range points. Positive values indicate enrichment among selected candidates; negative values indicate enrichment among rejected candidates.

| Behavior atom | Unit | top-1 | Behavior atom | Unit | top-1 |
|---|---|---|---|---|---|
| clarify/elicit | share | −61.8 | evaluate/compare | share | +9.8 |
| clean | share | +47.5 | plan/decompose | share | +9.1 |
| decomposition depth | range | +45.5 | boundary firmness | range | +8.4 |
| initiative | range | +43.1 | common-ground gap | share | −5.6 |
| clarification threshold | range | −42.7 | uncertainty calibration | range | −5.1 |
| explain/teach | share | +42.3 | answer/execute | share | +3.2 |
| underspecified | share | −34.1 | high-stakes/risk | share | +3.0 |
| directness | range | +31.0 | conflicting constraints | share | +2.0 |
| brevity | range | −24.2 | risk/boundary | share | −1.7 |
| assumption management | range | −18.4 | warmth | range | −1.1 |
| ambiguous intent | share | −12.2 | repair/reset | share | −0.9 |
| | | | repair needed | share | −0.4 |
| | | | prior misalignment | share | −0.3 |
| | | | abrupt frame shift | share | +0.2 |