WHEN AGENTS REFUSE TO ASK FOR HELP: PRIVILEGED SELF-DISTILLATION FOR HELP-SEEKING SOFTWARE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language-model agents increasingly execute long-horizon software-engineering tasks, yet some tasks require information present neither in the issue text nor anywhere in the repository. Exposing an `ask_human` tool does not by itself induce reliable help-seeking. Using HiL-Bench, we first localize the failure by inserting two diagnostic setups between `AskHuman` and `FullInfo` that withhold resolutions and supply only a cue about what is missing. `AskHuman-Description` recovers 91.9% and 91.2% of the `AskHuman` to `FullInfo` Pass@3 gap for GLM-5.2 and Qwen3.8-27B, while `AskHuman-Hint` recovers 66.1% and 81.3%. Identifying which information is missing, rather than acting on it once supplied, is therefore the dominant measured bottleneck. We then ask whether cue-elicited behaviour transfers to a model that receives no cue at inference time. We construct SWE-REBENCH-V2-BLOCKED, a corpus of blocker-augmented tasks built from SWE-ReBench V2 by rewriting only the observable task specification while leaving the repository, revision, test suite and target patch unchanged. We introduce Privileged Self-Distillation (PSD), in which a teacher rolls out with privileged blocker cues, the cues are removed, and an unprivileged student initialized from the same weights is trained on the resulting trajectories. We study a 2×2 matrix of privileged cue source (`AskHuman-Hint` versus `AskHuman-Description`) and supervision policy (`Ask-All` versus `Ask-Clean`, which masks the training loss on judge-rejected asking turns while leaving those turns in context). All four recipes fine-tune Qwen3.8-27B with reasoning enabled. `Description-Clean` attains the highest task success (Pass@1 31.73, Pass@3 49.64) and the highest blocker recall (68.13), whereas `Description-All` attains the highest question precision (38.87) and Ask-F1 (48.78). Cleaning raises task success and recall under both cue sources, and lowers precision and Ask-F1 under both.

1 INTRODUCTION

Software-engineering agents increasingly solve repository-level tasks by inspecting code, editing files, and executing tests (Pan et al., 2025; Golubev et al., 2025). Some task-critical information appears neither in the specification nor anywhere in the repository, such as an undocumented threshold or an unspecified default. A reliable agent recognizes the dependency, asks for the missing item, and uses the answer.

Exposing an interaction tool does not produce that behaviour. HiL-Bench adds three to five missing-information blockers to otherwise solvable tasks (Trinh et al., 2026) and gives the agent a blocked task alone (`Baseline`), an `ask_human` tool (`AskHuman`), or every blocker description with its resolution (`FullInfo`). The `AskHuman` to `FullInfo` difference is the *judgment gap*, the cost of noticing what is missing and recovering from it. Tool access leaves most of that gap open.

Diagnosis. We separate *blocker identification* from *post-identification recovery*, the work of formulating a question and using the answer. Two inserted setups withhold resolutions and supply only a cue, `AskHuman-Hint` a short abstract one and `AskHuman-Description` the full blocker description. Across the 100 public HiL-Bench tasks, `AskHuman-Description` recovers 91.9%

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.

and 91.2% of the `AskHuman` to `FullInfo` Pass@3 gap for GLM-5.2 and Qwen3.8-27B, and `AskHuman-Hint` recovers 66.1% and 81.3%. Autonomous blocker identification is the dominant measured bottleneck. The evidence is behavioural. We do not claim the capability is latent and merely unexpressed.

That asymmetry suggests a training signal. We elicit successful behaviour from the model itself with privileged information, then train an unprivileged copy of the same weights to reproduce it without the cue. Teacher and student differ only in what they observe. We call this *Privileged Self-Distillation* (PSD).

The 100 public tasks are too few to train on, so we build SWE-REBENCH-V2-BLOCKED from SWE-ReBench V2 (Badertdinov et al., 2026). The pipeline generates blockers and resolutions, rewrites the observable specification so the required information is unavailable, and emits an `ask_human` registry. Repository, revision, test suite and target patch are untouched, so executable tests remain the final correctness oracle. Our filter establishes empirical *set-level* necessity only (Section 4), and we do not call the constructed tasks fully verified.

What we find. We cross cue source (`AskHuman-Hint`, `AskHuman-Description`) with supervision policy. `Ask-All` supervises every retained teacher turn. `Ask-Clean` applies target-level loss masking to judge-rejected asking turns and leaves those turns in the context. All four recipes fine-tune Qwen3.8-27B with reasoning enabled on approximately 4,100 blocker-augmented tasks shared across the matrix. `Description-Clean` attains the highest task success (Pass@1 31.73, Pass@3 49.64) and recall (68.13). `Description-All` attains the highest precision (38.87) and Ask-F1 (48.78). Cleaning raises Pass@1, Pass@3 and recall under both cue sources. It lowers precision and Ask-F1 under both. Description-derived supervision is numerically higher on task success, precision and Ask-F1 under both policies. We name no best configuration. Cleaning removes 45.4% of Hint-corpus asks against 28.8% of Description-corpus asks, so the arms are not dose matched, and we report every difference descriptively, without significance claims. We compute no comparison between these arms and the untrained `AskHuman` baseline or the `FullInfo` ceiling, since protocol equivalence is not established.

We contribute the decomposition of help-seeking into identification and recovery, the SWE-REBENCH-V2-BLOCKED pipeline with its executable checks and set-level necessity filter, PSD, a 2×2 study reporting all four recipes, and the failure modes that weaken our own claims.

2 HELP-SEEKING UNDER MISSING INFORMATION

2.1 BLOCKER-AUGMENTED SOFTWARE-ENGINEERING TASKS

A repository-level software-engineering task is a tuple $x = (R, s, \mathcal{T})$ of repository R, observable specification s, and correctness oracle $\mathcal{T}$. A blocker-augmented task $\widetilde{x} = (R, \widetilde{s}, \mathcal{T}, B)$ removes task-critical information from $\widetilde{s}$ and changes nothing else. Each blocker in the registry $B = \{b_1, \ldots, b_m\}$ carries an information-gap description d_i and an exact resolution r_i unrecoverable by the agent.

`ask_human` returns r_i for a question targeting blocker b_i and `irrelevant question` for anything else. The hidden registry signals neither that a blocker exists nor what to ask.

2.2 TWO-STAGE HELP-SEEKING

Blocker identification is recognizing that the observations are insufficient and deciding to ask. *Post-identification recovery* is formulating a targeted question, integrating the answer, and completing the task, treated jointly because success requires all three.

`AskHuman` trajectories alone do not identify the failed stage. Missed diagnosis, poor answer use, and ordinary coding failure all end in the same observed outcome. Controlled interventions supply blocker information without the resolution. Table 1 lists the five setups, with h_i a short abstract hint derived from d_i.

Setup	Supplied information	Required behavior
`Baseline`	None, tool disabled	Solve despite unresolved blockers
`AskHuman`	None	Identify blockers, ask, integrate answers, solve
`AskHuman-Hint`	Hints $\{h_i\}$	Refine each cue into a targeted question, integrate, solve
`AskHuman-Description`	Descriptions $\{d_i\}$	Ask for each missing resolution, integrate, solve
`FullInfo`	Descriptions $\{d_i\}$ and resolutions $\{r_i\}$, tool disabled	Integrate supplied information and solve

Table 1: Information and interaction available under the five setups.

The same distinction shapes training, where a privileged teacher observes an additional cue $z_B \in \{\{h_i\}, \{d_i\}\}$ and the student it supervises sees the blocked task alone (Section 5).

2.3 Evaluation measures

Let $\mathcal{Q}$ be a trajectory's questions and $B_{\text{addr}} \subseteq B$ the distinct blockers addressed by a relevant question. Precision, recall, and their harmonic mean are

$$\text{Prec} = \frac{|B_{\text{addr}}|}{|\mathcal{Q}|}, \qquad \text{Rec} = \frac{|B_{\text{addr}}|}{|B|}, \qquad \text{Ask-F1} = \frac{2\,\text{Prec}\,\text{Rec}}{\text{Prec} + \text{Rec}}. \tag{1}$$

Both numerators use the deduplicated count, as in the reference implementation we follow, so repeated asks about one blocker are penalized. Precision is 0 when $\mathcal{Q} = \emptyset$, so an episode that correctly asked nothing scores like one that addressed nothing. Counts are pooled across an evaluation. We do not average per episode. Appendix A.3 gives both consequences in full. Recall can rise through indiscriminate asking, and we do not establish the mechanism behind the precision differences we report.

Task performance uses the standard Pass@k estimator over n trajectories for each of N tasks, with c_t successful for task t.

$$\text{Pass@}\,k = \frac{1}{N}\sum_{t=1}^{N}\left(1 - \frac{\binom{n-c_t}{k}}{\binom{n}{k}}\right). \tag{2}$$

3 Diagnosing the Help-Seeking Gap on HiL-Bench

3.1 Diagnostic design and evaluation protocol

HiL-Bench's 150 software-engineering tasks each carry three to five blockers unresolvable from the task statement or repository, and we evaluate the 100 public tasks with 50 held out (Trinh et al., 2026). `AskHuman-Description` gives the descriptions $\{d_i\}$ without resolutions or example questions. `AskHuman-Hint` gives a category cue h_i naming the missing information.

Llama-3.3-70B-Instruct-AWQ, which also backs the judge, wrote one fixed hint for each of the 353 blockers from d_i alone, screened against r_i for leakage. Hints average 12.0 words and 79 characters. Descriptions average 322 characters.

GLM-5.2 and Qwen3.8-27B run in the SWE-Agent harness (Yang et al., 2024) with matched prompts, sampling, and limits, so only tool access and prompt information vary. We sample five trajectories per task and setup, including infrastructure failures. Every metric comes from the same judge as mean $\pm$ standard deviation over five seeds. Per-seed values are omitted by choice. Paired values are GLM-5.2/Qwen3.8-27B.

Model	Baseline	AskHuman	Hint	Description	FullInfo
GLM-5.2	1.4±0.6	26.4±2.0	62.1±1.9	76.0±1.2	80.4±3.6
Qwen3.8-27B	4.5±2.1	23.0±1.9	74.9±1.8	81.2±1.2	86.8±1.1

Model	Setup	Precision	Recall	Ask-F1
GLM-5.2	`AskHuman`	28.8±0.9	49.0±2.3	36.2±1.0
	`AskHuman-Hint`	42.3±1.4	87.4±0.6	57.0±1.4
	`AskHuman-Description`	54.9±1.5	92.4±0.5	68.9±1.1
Qwen3.8-27B	`AskHuman`	42.8±9.2	43.0±1.6	42.3±4.8
	`AskHuman-Hint`	57.2±13.1	89.3±1.9	69.1±11.3
	`AskHuman-Description`	68.4±3.3	93.9±1.0	79.2±2.6

Table 2: Diagnostic results on the 100 public HiL-Bench tasks. Upper panel, Pass@3. Lower panel, help-seeking metrics. All values are percentages, mean ± standard deviation over five seeds. Hint and Description denote `AskHuman-Hint` and `AskHuman-Description`.

3.2 Tool access leaves a large judgment gap

Adding `ask_human` raises Pass@3 from 1.4%/4.5% in `Baseline` to 26.4%/23.0% in `AskHuman`. `FullInfo` reaches 80.4%/86.8%, leaving judgment gaps of 54.0/63.8 points (Table 2).

The models address 49.0%/43.0% of registered blockers at 28.8%/42.8% precision, so coverage and targeting both fail. These aggregates do not locate the failed stage.

3.3 Short hints recover most of the gap

`AskHuman-Hint` raises Pass@3 to 62.1%/74.9%, gains of 35.7/51.9 points that recover 66.1%/81.3% of the respective judgment gaps by Equation 8. Recall rises 38.4/46.3 points to 87.4%/89.3% and precision 13.5/14.4 points, with greater variation across Qwen3.8-27B seeds.

3.4 Complete descriptions isolate the identification bottleneck

`AskHuman-Description` raises Pass@3 to 76.0%/81.2%, recovering 91.9%/91.2% of the judgment gap and finishing 4.4/5.6 points below `FullInfo`, so autonomous identification accounts for most of it. Over `AskHuman-Hint` it adds 13.9/6.3 Pass@3 points, with recall up 5.0/4.6 and precision up 12.6/11.2. Precision stays below 70%, so question formulation, answer integration, and execution still fail. The precision mechanism is not established.

3.5 Implication for bootstrapping help-seeking

A salient blocker lets the same base model ask a relevant question and solve the task, so the deficit depends on the agent's information state. The cues alter context, so we do not establish that the unprivileged policy is already latent. The trained arms are not run under this protocol, so we draw no comparison with the numbers here. We next test whether these privileged demonstrations transfer to an agent given neither cue.

4 Constructing SWE-ReBench-V2-Blocked

The 100 public HiL-Bench tasks support diagnosis. Training needs far more, so we build SWE-ReBench-V2-Blocked, a corpus of *blocker-augmented tasks* derived from SWE-ReBench V2 (Badertdinov et al., 2026).

Exactly one thing changes. Repository, base revision, test suite and target patch carry over unchanged. The observable specification is rewritten to withhold information needed to derive the target behaviour. A blocked task is graded by the executable criterion that graded its source.

Pipeline. We select 17,548 SWE-ReBench V2 tasks across four languages, dropping 232 poison-pill identifiers and 187 container commands over the 131,072-byte argument limit. Survivors are verified by applying the reference and test patches and running the task's own install and test commands. A task verifies when FAIL_TO_PASS $\subseteq$ PASSED and PASS_TO_PASS $\subseteq$ PASSED under strict patch application. 14,116 verify. At 90 concurrent workers verification returned roughly 33% *false* failures from peer-to-peer contention, so 1,655 verdicts were purged and re-measured and verification is capped at 16 workers.

A generator then rewrites the observable specification with read, list and search tools only, in a container where the fix is absent from the git object store. *Missing parameters* removes a statement, *ambiguous requirements* leaves one that reads two ways, and *contradictory requirements* keeps a tension the source already contains. We then run a solver on the blocked statement alone, with no `ask_human` and no reachable resolution, for $k = 2$ seeds at temperature 1.0, keeping the group when no seed passes.

The registry holds 5,600 blocker-augmented tasks carrying 19,164 blockers, distributed 1:8, 2:19, 3:3,506, 4:1,735, 5:332 per task. By type, 8,970 are missing parameters, 8,723 ambiguous requirements and 1,457 contradictory requirements, with 12 across 8 tasks mistyped and 2 unlabelled. Training and evaluation split at the *repository* level, and the corpus is decontaminated against HiL-Bench.

4.1 Construction Failure Modes

Five properties bound what this corpus supports. The prompt enforces the blocker floor. No filter checks it, and the floor is soft, with 27 tasks below it. Necessity is set-level, so no individual blocker is shown to be load-bearing. A keep verdict covers two outcomes the filter cannot separate, failure blocked with a pass when answered, and failure blocked *and* when answered (over-hidden and unusable). The filter is two-sample and we ran no corpus-wide sufficiency check, so a high keep rate is equally consistent with well-formed blockers and impossible tasks. The frozen interface section, restored verbatim after generation, can in principle restore a blocker's answer, and nothing records whether the generator respected the prohibition.

We also did not verify that a blocker is recoverable by a plausible question, ran no leakage or uniqueness checks, and ran no human audit of whether blocked specifications read naturally. We therefore do **not** call these tasks fully verified. The reference patch passes and the blocked specification defeats a two-seed solver. Appendix B gives the full pipeline, per-stage counts, and a scaffold defect that silently cost 21.5% of generator sessions.

5 Privileged Self-Distillation

5.1 Formulation

A blocker cue elicits help-seeking the same model does not produce unaided (Section 3), and PSD turns that asymmetry into supervision. Let H_t be the agent's observable history at step t under unprivileged `AskHuman` and z_B a privileged cue from the blocker registry B. Teacher and student are

$$\pi_{\text{teacher}}(a_t \mid H_t, z_B), \qquad \pi_{\text{student}}(a_t \mid H_t), \tag{3}$$

where z_B is either short blocker hints (`AskHuman-Hint`) or complete blocker descriptions (`AskHuman-Description`). Neither cue reveals the resolution, so the teacher must still ask through `ask_human` and integrate the reply. Both start from Qwen3.8-27B, so the teacher's advantage is informational alone and the ceiling is what the cues elicit from this model.

Training minimizes token-level cross-entropy on teacher trajectories after cue removal,

$$\mathcal{L}(\theta) \;=\; -\sum_{(H_t,\, a_t)\,\in\,\mathcal{D}} \;\sum_{j} m_{t,j}\, \log \pi_\theta(a_{t,j} \mid H_t,\, a_{t,<j})\,, \tag{4}$$

where $m_{t,j} \in \{0, 1\}$ masks the tokens of turn a_t. The two policies below differ only in how m is set on asking turns.

5.2 Collection, Deprivileging, and Supervision Format

The teacher rolls out with z_B in context on the roughly $4{,}100$ blocker-augmented tasks of Section 4, shared across the full matrix, under the evaluation harness and capped at 20 `ask_human` calls per episode. Before training z_B leaves the prompt. Resolutions survive only where they entered as the reply to an accepted `ask_human` call, in their original position. Reasoning traces stay in the supervised target, since the student trains and evaluates with reasoning enabled.

Episodes are long, so we tokenize each one once as a *trunk* and supervise every turn as a *branch*, a trunk prefix plus the turn, with loss on the turn's own tokens only. Every later turn reads the trunk. Appendix C gives the layout.

5.3 Ask-All and Ask-Clean

On the Hint corpus the teacher issued $36{,}567$ questions, of which the expert answered $16{,}614$ (45.4%) with `irrelevant question`, and 46.5% of episodes contain at least one rejected ask. Imitating that wholesale teaches the student to ask badly as well as to ask. Cleaning removes 45.4% of Hint asks and 28.8% of Description asks, so the four arms are not dose matched.

`Ask-All` supervises every retained teacher turn, including rejected asks. `Ask-Clean` is target-level loss masking with the rejected ask left in context. It drops the branch whose target is a rejected ask, and that ask and the expert's reply stay verbatim in the trunk every later branch reads. A separate implementation in our codebase excises rejected asks from the message list and is defective, since later turns generated after the rejection survive that excision. It produced no evaluated checkpoint and none of our results come from it.

5.4 Supervision Failure Modes

Six issues bear on how these results should be read, and Appendix C quantifies each. Over-asking is already in the teacher, and `Ask-Clean` leaves those episodes in training. We dropped no trajectory for over-asking, since the filter we evaluated made the two arms byte-identical at a zero-rejection threshold. Our literal leak checks cover the description field only, so cue paraphrases are less well checked on the Hint side. The student learns from states a privileged policy generated. Supervision keys on the judge's verdict of each ask, and task resolution plays no part. Only 80.3% of tasks fit the training cap, and the excluded fifth are the longest and hardest, so every result is conditional on an easier population. We found and fixed two preprocessing defects and rebuilt the corpora before the reported runs, a dropped provenance field that mis-assembled 21.6% of Hint-corpus episodes, one-sided on the cue axis, and a mask helper that would have overwritten the trunk-and-branch loss mask, which we verified on hardware.

6 Experiments and Results

6.1 Experimental Protocol

Each recipe full fine-tunes Qwen3.8-27B with reasoning enabled, so reasoning traces are supervised targets. Training runs one epoch of AdamW ($\epsilon = 10^{-8}$, weight decay 0.01) at learning rate 1×10^{-6}, held flat after a 5-iteration warmup, with gradient-norm clipping 1.0, seed 42, global batch 40 sequences, micro-batch 1, packing disabled, tensor-parallel 8, pipeline-parallel 4, and sequence length $131{,}072$ tokens, with the corpus capped at $126{,}976$. Appendix D lists the rest.

We report each arm's *final* checkpoint. Selection uses no evaluation metric and never touches the evaluation set. An arm that peaked earlier is still reported at its endpoint.

We evaluate all four recipes on the 100 public HiL-Bench software-engineering tasks containing 353 blockers, under the unprivileged `AskHuman` setup with no hint and no blocker description at inference. The agent runs in SWE-Agent at temperature 1.0, capped at 200 turns. The HiL-Bench judge, Llama-3.3-70B-Instruct-AWQ, scores question relevance unchanged, and executable repository tests decide task success.

Each cell is the mean ± standard deviation over five seeds. We deliberately do not tabulate per-seed values, and we run no pairwise statistical test. Every difference between arms is numerical, and we claim no significance.

Section 3 reports untrained models under `Baseline`, `AskHuman`, `AskHuman-Hint`, `AskHuman-Description` and `FullInfo`. Those runs are not established as protocol equivalent to these, so we make *no* numerical comparison with the PSD arms and place no untrained `AskHuman` baseline and no `FullInfo` ceiling beside the PSD table.

6.2 PSD Training Variants

The matrix crosses the privileged cue the teacher observed with the supervision policy applied to its trajectories. All four cells are completed and reported.

Table 3: The 2×2 matrix of PSD training recipes.

Privileged cue	`Ask-All`	`Ask-Clean`
`AskHuman-Hint` (short hint)	`Hint-All`	`Hint-Clean`
`AskHuman-Description` (full description)	`Description-All`	`Description-Clean`

All four arms are built from approximately $4{,}100$ blocker-augmented tasks shared across the full matrix, with repository-level train and evaluation separation and decontamination against HiL-Bench (Section 4). They are still not dose matched, by design. `Ask-Clean` masks the training loss at the target level on judge-rejected asking turns. Those turns stay in context. The judge rejects 45.4% of Hint-corpus asks and 28.8% of Description-corpus asks. The arms also differ in optimizer steps and supervised turns. We therefore call the four cells four controlled training recipes. They are not a dose-matched causal factorial, and we isolate no causal effect at fixed training dose.

6.3 Main Task-Success Results

Table 4 reports all four PSD recipes on the 100 public HiL-Bench tasks under unprivileged `AskHuman` inference.

Table 4: The complete 2×2 PSD matrix. Qwen3.8-27B, reasoning enabled, evaluated without privileged cues. Mean ± standard deviation over five seeds. All differences are numerical and we report no significance tests. Bold marks the best value in each column.

Cue	**Supervision**	**Pass@1**	**Pass@3**	**Precision**	**Recall**	**Ask-F1**
Description	Ask-All	25.93±1.50	43.23±5.18	**38.87**±5.12	65.46±0.50	**48.78**±4.04
	Ask-Clean	**31.73**±1.51	**49.64**±5.44	22.30±4.85	**68.13**±0.36	33.37±5.69
Hint	Ask-All	22.59±2.72	39.87±5.24	9.06±2.63	64.58±4.28	15.83±4.18
	Ask-Clean	24.55±2.94	42.97±5.57	8.48±2.94	66.37±4.26	15.04±4.62

`Description-Clean` attains the highest task success (Pass@1 31.73, Pass@3 49.64) and the highest blocker recall (68.13). Its precision and Ask-F1 fall well below `Description-All`'s, so we do not call it best overall. We tabulate aggregates only and omit per-seed values by choice. We quote no untrained `AskHuman` baseline and no `FullInfo` ceiling beside these arms, since protocol equivalence does not hold.

6.4 Cue Source, Hint versus Description

Under both supervision policies, Description-derived training is numerically stronger than Hint-derived on task success, precision, and Ask-F1. The precision gap is the largest effect in the table, 38.87 against 9.06 under `Ask-All` and 22.30 against 8.48 under `Ask-Clean`. Recall spans only 64.58 to 68.13 across all four recipes, a range of 3.55 points, against 30.39 points for precision and 9.14 for Pass@1.

Blocker descriptions come from our pipeline's Qwen3.8-27B, the family of both teacher and student. Llama-3.3-70B-Instruct-AWQ authored the short hints offline, so the two cue conditions also differ in provenance. PSD is self-distillation only in that the trajectory-generating policy is the student's own initialization given privileged context, and the artifacts come from two models.

6.5 Supervision Policy, Ask-All versus Ask-Clean

`Ask-Clean` applies target-level loss masking and leaves the rejected ask in context. Under both cues, cleaning moves each metric in the same direction. Pass@1, Pass@3 and recall rise. Precision and Ask-F1 fall. The precision cost is 16.57 points under Description and 0.58 under Hint, so the trade-off sits in Description-derived training. We do not establish the mechanism behind that drop. No over-asking claim in this paper rests on precision alone.

The four arms are built from approximately $4{,}100$ blocker-augmented tasks shared across the full matrix. They are still not dose matched. The judge rejects 45.4% of Hint-corpus asks against 28.8% of Description-corpus asks. We present the matrix as four recipes and claim no factorial design.

6.6 Help-Seeking Precision, Recall, and Ask-F1

`Description-All` attains the highest precision and Ask-F1, 16.57 points above `Description-Clean` on precision and 15.41 above on Ask-F1. Optimizing task success and blocker coverage therefore does not optimize question quality. The choice depends on the cost of a wasted question to the human in the loop, which this benchmark does not model, since `ask_human` answers are free and instantaneous.

The standard deviation on precision (2.63 to 5.12) and on Ask-F1 (4.04 to 5.69) is large relative to several differences here. The Description-versus-Hint precision gap of 29.81 points under `Ask-All` is many times that spread. The 0.79-point Ask-F1 difference between `Hint-All` and `Hint-Clean` is smaller than either arm's spread and should not be read as an effect. Every Pass@3 standard deviation exceeds its Pass@1 counterpart, so Pass@3 differences are the least resolved values in the table.

7 Related Work

Clarification and help-seeking. QuestBench formalizes asking the right question in reasoning tasks (Li et al., 2025), When2Call studies when *not* to invoke a tool (Ross et al., 2025), and structured-uncertainty methods trigger clarification from a model's own uncertainty (Suri et al., 2026). In code, ClarEval scores clarification under ambiguity (Li et al., 2026a) and DiscoBench asks when search agents should query (Tao et al., 2026b). Gulati et al. (2026) isolate timing in long-horizon tasks, and Pu et al. (2026) introduce controllable underspecification. Our contribution to this line is a diagnosis of where clarification fails and training data that changes it.

HiL-Bench supplies the benchmark, setups, blocker registries, and judge we inherit (Trinh et al., 2026), and establishes that exposing `ask_human` leaves a large gap to an oracle supplying the missing information. We localize that gap and narrow it without oracle access at inference time. Li et al. (2026b) quantify what distinct oracle signals contribute to SWE agents. Agent evaluation has broadened to multi-turn boards such as AgentBoard (Ma et al., 2024).

Privileged information and self-distillation. PSD inherits a structure long studied in learning with privileged information and in asymmetric actor-critic imitation, a teacher observing more than the deployed student, with the asymmetry confined to the *input*. Here teacher and student are the *same weights*, the student's own initialization reading a richer context, which removes the capacity gap and makes the transferred quantity purely behavioural. Self-distillation on a model's own outputs is established practice, including born-again training against a same-architecture teacher. The advantage here is a privileged observation that is then deliberately removed, so the student must learn to *infer* a context. Ensembling, temperature, correctness filtering, and distribution sharpening play no part in it.

Trajectory-level training of coding agents is a standard recipe (Pan et al., 2025; Golubev et al., 2025), spanning multi-task and RL variants (He et al., 2025; Luo et al., 2025; Cao et al., 2025),

data scaling laws (Zeng et al., 2025), the limits of SFT for issue resolution (Tao et al., 2026a; Song et al., 2026), and agentless or skill-prior-first policies (Xia et al., 2025; Yang et al., 2025b; Xie et al., 2025). We adopt this machinery and change what the trajectories contain, since the imitated behaviour was elicited by information the student will not have, and we ablate which asking turns carry gradient. SWE-bench established the issue-resolution format (Jimenez et al., 2024), and SWE-rebench with its V2 successor supplies our decontaminated source pool (Badertdinov et al., 2025; 2026). Others synthesize bugs or environments (Pham et al., 2025; Yang et al., 2025a; Sonwane et al., 2025; Jain et al., 2025), scale their construction (Chen et al., 2026; Fu et al., 2026; Sun et al., 2026), or build training-data factories (Guo et al., 2026). These manufacture *solvable* tasks. Our tasks also start solvable as specified, and we remove information until they cannot be solved without asking, holding the repository, tests, and target patch fixed. The construction is adversarial to the agent's information state and leaves its engineering skill untouched.

8 LIMITATIONS

The simulated human is deterministic and free. `ask_human` returns a fixed resolution when the judge rules that a question targets a blocker, and `irrelevant question` otherwise. A real collaborator answers variably and costs something to interrupt, so the benchmark under-penalizes over-asking.

Precision, recall, and Ask-F1 all rest on that judge, used unchanged with *no* manual audit, so its bias reaches every help-seeking number. The judge also wrote the short hints, so the Hint condition and the scorer share a model. `AskHuman-Hint` and `AskHuman-Description` disclose how many cues exist, so the gap-recovery figures are optimistic.

9 CONCLUSION

An `ask_human` tool does not suffice to make an agent ask for help. In two coding models, cues naming what is missing, without resolving it, recover 91.9% and 91.2% of the `AskHuman`-to-`FullInfo` Pass@3 gap under blocker descriptions and 66.1% and 81.3% under short hints. Deciding *which* information is absent is the dominant measured bottleneck. The claim is behavioural. We do not claim the full unprivileged capability is latent.

SWE-REBENCH-V2-BLOCKED builds $5{,}600$ tasks carrying $19{,}164$ blockers from SWE-ReBench V2 environments by rewriting only the observable specification, so tests remain the oracle. Reference patches pass and blocked specifications defeat a two-seed solver. Other properties remain unverified.

Privileged Self-Distillation trains a student on its own cue-stripped rollouts and evaluates it cue-free. Description-derived supervision beats Hint-derived numerically under both policies. `Ask-Clean` is target-level loss masking that leaves judge-rejected asks in context, and it raises Pass@1, Pass@3, and recall and lowers precision and Ask-F1. `Description-Clean` leads Pass@1 (31.73), Pass@3 (49.64), and recall (68.13). `Description-All` leads precision (38.87) and Ask-F1 (48.78). Differences are numerical and we ran no significance tests. Cleaning's precision cost has no established mechanism.

Results are specific to Qwen3.8-27B. The four recipes share approximately $4{,}100$ blocker-augmented tasks and are not dose matched, since cleaning removes 45.4% of Hint asks against 28.8% of Description asks. We establish no protocol equivalence with an untrained `AskHuman` baseline or a `FullInfo` ceiling, and compare no trained arm with the diagnostic results.

REFERENCES

Ibragim Badertdinov, Alexander Golubev, Maksim Nekrashevich, Anton Shevtsov, Simon Karasik, Andrei Andriushchenko, Maria Trofimova, Daria Litvintseva, and Boris Yangel. SWE-rebench: An automated pipeline for task collection and decontaminated evaluation of software engineering agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL `https://openreview.net/forum?id=nMpJoVmRy1`.

Ibragim Badertdinov, Maksim Nekrashevich, Anton Shevtsov, and Alexander Golubev. Swe-rebench v2: Language-agnostic swe task collection at scale, 2026. URL https://arxiv.org/abs/2602.23866.

Shiyi Cao, Dacheng Li, Fangzhou Zhao, Shuo Yuan, Sumanth R. Hegde, Connor Chen, Charlie Ruan, Tyler Griggs, Shu Liu, Eric Tang, Richard Liaw, Philipp Moritz, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Skyrl-agent: Efficient rl training for multi-turn llm agent, 2025. URL https://arxiv.org/abs/2511.16108.

Mouxiang Chen, Lei Zhang, Yunlong Feng, Xuwu Wang, Wenting Zhao, Ruisheng Cao, Jiaxi Yang, Jiawei Chen, Mingze Li, Zeyao Ma, Hao Ge, Zongmeng Zhang, Zeyu Cui, Dayiheng Liu, Jingren Zhou, Jianling Sun, Junyang Lin, and Binyuan Hui. Swe-universe: Scale real-world verifiable environments to millions, 2026. URL https://arxiv.org/abs/2602.02361.

Dayuan Fu, Shenyu Wu, Yunze Wu, Zerui Peng, Yaxing Huang, Jie Sun, Ji Zeng, Mohan Jiang, Lin Zhang, Yukun Li, Jiarui Hu, Liming Liu, Jinlong Hou, and Pengfei Liu. davinci-env: Open swe environment synthesis at scale, 2026. URL https://arxiv.org/abs/2603.13023.

Alexander Golubev, Maria Trofimova, Sergei Polezhaev, Ibragim Badertdinov, Maksim Nekrashevich, Anton Shevtsov, Simon Karasik, Sergey Abramov, Andrei Andriushchenko, Filipp Fisin, Sergei Skvortsov, and Boris Yangel. Training long-context, multi-turn software engineering agents with reinforcement learning, 2025. URL https://arxiv.org/abs/2508.03501.

Anmol Gulati, Hariom Gupta, Elias Lumer, Sahil Sen, and Vamse Kumar Subbiah. Ask early, ask late, ask right: When does clarification timing matter for long-horizon agents?, 2026. URL https://arxiv.org/abs/2605.07937.

Lianghong Guo, Yanlin Wang, Caihua Li, Wei Tao, Pengyu Yang, Jiachi Chen, Haoyu Song, Duyu Tang, and Zibin Zheng. Swe-factory: Your automated factory for issue resolution training data and evaluation benchmarks, 2026. URL https://arxiv.org/abs/2506.10954.

Zhenyu He, Qingping Yang, Wei Sheng, Xiaojian Zhong, Kechi Zhang, Chenxin An, Wenlei Shi, Tianle Cai, Di He, Jiaze Chen, and Jingjing Xu. SWE-Swiss: A multi-task fine-tuning and RL recipe for high-performance issue resolution. Notion Blog / GitHub Repository, 2025. URL https://github.com/zhenyuhe00/SWE-Swiss. Accessed: 2026-03-17.

Naman Jain, Jaskirat Singh, Manish Shetty, Tianjun Zhang, Liang Zheng, Koushik Sen, and Ion Stoica. R2e-gym: Procedural environment generation and hybrid verifiers for scaling open-weights SWE agents. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=7evvwwdo3z.

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

Belinda Z. Li, Been Kim, and Zi Wang. Questbench: Can llms ask the right question to acquire information in reasoning tasks?, 2025. URL https://arxiv.org/abs/2503.22674.

Jialin Li, Yuan Wu, and Yi Chang. Clareval: A benchmark for evaluating clarification skills of code agents under ambiguous instructions, 2026a. URL https://arxiv.org/abs/2603.00187.

Kenan Li, Qirui Jin, Liao Zhu, Xiaosong Huang, Yijia Wu, Yikai Zhang, Xin Zhang, Zijian Jin, Yufan Huang, Elsie Nallipogu, Chaoyun Zhang, Yu Kang, Saravan Rajmohan, Qingwei Lin, Wenke Lee, and Dongmei Zhang. Oracle-swe: Quantifying the contribution of oracle information signals on swe agents, 2026b. URL https://arxiv.org/abs/2604.07789.

M. Luo, N. Jain, J. Singh, S. Tan, A. Patel, Q. Wu, A. Ariyak, C. Cai, T. Venkat, S. Zhu, B. Athiwaratkun, M. Roongta, C. Zhang, L. E. Li, R. A. Popa, K. Sen, and I. Stoica. DeepSWE: Training a fully open-sourced, state-of-the-art coding agent by scaling RL. https://www.together.ai/blog/deepswe, Jul. 2025. Together AI Blog post. Accessed: 2025-12-22.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn llm agents, 2024. URL https://arxiv.org/abs/2401.13178.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-gym. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 47717–47737. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/pan25g.html.

Minh V. T. Pham, Huy N. Phan, Hoang N. Phan, Cuong Le Chi, Tien N. Nguyen, and Nghi D. Q. Bui. Swe-synth: Synthesizing verifiable bug-fix data to enable large language models in resolving real-world bugs, 2025. URL https://arxiv.org/abs/2504.14757.

George Pu, Michael S. Lee, Udari Madhushani Sehwag, David J. Lee, Bryan Zhu, Yash Maurya, Mohit Raghavendra, Yuan Xue, and Samuel Marc Denton. Lhaw: Controllable underspecification for long-horizon tasks, 2026. URL https://arxiv.org/abs/2602.10525.

Hayley Ross, Ameya Sunil Mahabaleshwarkar, and Yoshi Suhara. When2call: When (not) to call tools, 2025. URL https://arxiv.org/abs/2504.18851.

Huatong Song, Lisheng Huang, Shuang Sun, Jinhao Jiang, Ran Le, Daixuan Cheng, Guoxin Chen, Yiwen Hu, Zongchao Chen, Yiming Jia, Wayne Xin Zhao, Yang Song, Tao Zhang, and Ji-Rong Wen. Swe-master: Unleashing the potential of software engineering agents via post-training, 2026. URL https://arxiv.org/abs/2602.03411.

Atharv Sonwane, Isadora White, Hyunji Lee, Matheus Pereira, Lucas Caccia, Minseon Kim, Zhengyan Shi, Chinmay Singh, Alessandro Sordoni, Marc-Alexandre Côté, and Xingdi Yuan. Bugpilot: Complex bug generation for efficient learning of swe skills, 2025. URL https://arxiv.org/abs/2510.19898.

Shuang Sun, Huatong Song, Lisheng Huang, Jinhao Jiang, Ran Le, Zhihao Lv, Zongchao Chen, Yiwen Hu, Wenyang Luo, Wayne Xin Zhao, Yang Song, Hongteng Xu, Tao Zhang, and Ji-Rong Wen. Swe-world: Building software engineering agents in docker-free environments, 2026. URL https://arxiv.org/abs/2602.03419.

Manan Suri, Puneet Mathur, Nedim Lipka, Franck Dernoncourt, Ryan A. Rossi, and Dinesh Manocha. Structured uncertainty guided clarification for LLM agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 40811–40838, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.2028. URL https://aclanthology.org/2026.findings-acl.2028/.

Chaofan Tao, Jierun Chen, Yuxin Jiang, Kaiqi Kou, Shaowei Wang, Ruoyu Wang, Xiaohui Li, Sidi Yang, Yiming Du, Jianbo Dai, Zhiming Mao, Xinyu Wang, Lifeng Shang, and Haoli Bai. Swe-lego: Pushing the limits of supervised fine-tuning for software issue resolving, 2026a. URL https://arxiv.org/abs/2601.01426.

Yiling Tao, Shihan Deng, Meiling Tao, Pengzhi Wei, Zhichao Hu, and Zhihao Zhu. When search agents should ask: Discobench for clarification-aware deep search, 2026b. URL https://arxiv.org/abs/2606.27669.

Tu Trinh, Mohamed Elfeki, Guangze Luo, Kelvin Luu, Nathan Hunt, Ernesto Hernandez, Nandan Marwaha, Yannis Yiming He, Charles Wang, Fernando Carabedo, et al. Hil-bench (human-in-loop benchmark): Do agents know when to ask for help? *arXiv preprint arXiv:2604.09408*, 2026.

Chunqiu Steven Xia, Yinlin Deng, Soren Dunn, and Lingming Zhang. Demystifying llm-based software engineering agents. *Proc. ACM Softw. Eng.*, 2(FSE), June 2025. doi: 10.1145/3715754. URL https://doi.org/10.1145/3715754.

Chengxing Xie, Bowen Li, Chang Gao, He Du, Wai Lam, Difan Zou, and Kai Chen. SWE-fixer: Training open-source LLMs for effective and efficient GitHub issue resolution. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 1123–1139, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.62. URL https://aclanthology.org/2025.findings-acl.62/.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 50528–50652. Curran Associates, Inc., 2024. doi: 10.52202/079017-1601. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/5a7c947568c1b1328ccc5230172e1e7c-Paper-Conference.pdf.

John Yang, Kilian Lieret, Carlos E Jimenez, Alexander Wettig, Kabir Khandpur, Yanzhe Zhang, Binyuan Hui, Ofir Press, Ludwig Schmidt, and Diyi Yang. SWE-smith: Scaling data for software engineering agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025a. URL https://openreview.net/forum?id=63iVrXc8cC.

Zonghan Yang, Shengjie Wang, Kelin Fu, Wenyang He, Weimin Xiong, Yibo Liu, Yibo Miao, Bofei Gao, Yejie Wang, Yingwei Ma, Yanhao Li, Yue Liu, Zhenxing Hu, Kaitai Zhang, Shuyi Wang, Huarong Chen, Flood Sung, Yang Liu, Yang Gao, Zhilin Yang, and Tianyu Liu. Kimi-dev: Agentless training as skill prior for swe-agents, 2025b. URL https://arxiv.org/abs/2509.23045.

Liang Zeng, Yongcong Li, Yuzhen Xiao, Changshi Li, Chris Yuhao Liu, Rui Yan, Tianwen Wei, Jujie He, Xuchen Song, Yang Liu, and Yahui Zhou. Skywork-swe: Unveiling data scaling laws for software engineering in llms, 2025. URL https://arxiv.org/abs/2506.19290.

A Definitions and Metrics

A.1 Task and Registry

An ordinary software-engineering task is a tuple $x = (R, s, \mathcal{T})$ of repository R at a fixed revision, observable specification s, and executable correctness oracle $\mathcal{T}$. A blocker-augmented task $\widetilde{x} = (R, \widetilde{s}, \mathcal{T}, B)$ withholds task-critical information from $\widetilde{s}$ and leaves R, the revision, $\mathcal{T}$, and the target patch unchanged.

Each task carries a blocker registry $B = \{b_1, \ldots, b_m\}$. A blocker b_i contains a unique identifier, a type, an information-gap description d_i, a resolution r_i, and example questions. A hint h_i is a short abstract paraphrase of d_i naming the gap without supplying r_i. Example questions are never shown to the agent. Descriptions d_i are shown in `AskHuman-Description` and `FullInfo`, and resolutions r_i only in `FullInfo`. Table 1 lists what each setup supplies.

A.2 Interaction Semantics

The agent submits a natural-language question q through `ask_human`, and

$$\texttt{ask_human}(q) = \begin{cases} r_i & \text{if } q \text{ targets blocker } b_i, \\ \texttt{irrelevant question} & \text{otherwise.} \end{cases} \tag{5}$$

A question returning a resolution is *relevant*, and that blocker is *addressed*. The simulated human is deterministic, free, and instantaneous.

A.3 METRICS

Let Q be the questions an episode asked, D the distinct blockers addressed, and $|B|$ the registry size.

$$\text{Precision} = |D|/|Q|, \qquad \text{Recall} = |D|/|B|, \qquad \text{Ask-F1} = 2\,\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{6}$$

The numerator of both precision and recall is the deduplicated count $|D|$, so asking one blocker twice raises the precision denominator alone. Precision is 0 when $|Q| = 0$. Counts are pooled across an evaluation and then divided.

Pass@k uses the unbiased estimator over n samples per task with c resolved,

$$\text{Pass@}k = \mathbb{E}\left[1 - \binom{n-c}{k} \Big/ \binom{n}{k}\right], \tag{7}$$

and task success is decided by the source task's test suite. For a setup S between `AskHuman` and `FullInfo`,

$$\text{GapRecovery}(S) = \frac{\text{Pass@3}(S) - \text{Pass@3}(\texttt{AskHuman})}{\text{Pass@3}(\texttt{FullInfo}) - \text{Pass@3}(\texttt{AskHuman})}, \tag{8}$$

defined on Pass@3 throughout. Question relevance is scored by the released HiL-Bench judge, Llama-3.3-70B-Instruct-AWQ, used unchanged.

B CORPUS CONSTRUCTION

Table 5: Construction funnel, and the blocker distribution of the shipped registry.

Stage	Tasks	Blockers/task	Tasks	Type	Blockers
Selected from SWE-ReBench V2	17,548	1	8	Missing parameters	8,970
Identifier blacklist	−232	2	19	Ambiguous req.	8,723
Oversized container command	−187	3	3,506	Contradictory req.	1,457
Verified executable	14,116	4	1,735	Non-canonical label	12
Exported registry	5,600	5	332	Type absent	2
Blockers in registry	19,164				

A task verifies when FAIL_TO_PASS $\subseteq$ PASSED and PASS_TO_PASS $\subseteq$ PASSED, under strict patch application requiring a clean apply. Grading a solver's patch later uses best-effort application.

The generator receives the specification, FAIL_TO_PASS test names, test patch, reference patch, changed-file list, and PR description in banner-delimited sections, and only the observable specification is rewritten. It has read, list, and search tools only. Its container checks out the base commit, deletes every other head, tag, and remote, expires the reflog, and garbage-collects, so the fix is absent from the git object store.

The three blocker types are distinct editing operations. *Missing parameters* removes a statement. *Ambiguous requirements* leaves a statement that reads two ways and deletes nothing. *Contradictory requirements* preserves a tension the source already contains and keeps both sides.

The necessity keep rule is

$$\text{KEPT} \;=\; \text{COMPLETE} \;\wedge\; \big(\text{PASSES} \le \text{MAX_PASS}\big), \tag{9}$$

with $k = 2$ seeds and MAX_PASS $= 0$. The solver sees the blocked statement alone, with no `ask_human` tool and no reachable resolution, sampling at temperature 1.0 and top-p 0.95. Per-seed outcomes are exported because the predicate is monotone in k, so raising k later is a wave over survivors and needs no regeneration. Training and evaluation material are separated at the repository level, and the corpus is decontaminated against HiL-Bench.

C SUPERVISION FORMAT AND POLICIES

The teacher is Qwen3.8-27B, the student's own initialization, rolled out on the blocker-augmented tasks with the privileged cue z_B in context, using the same agent framework and tool set as evaluation. Asking is capped at 20 `ask_human` calls per episode. Before training the cue is removed from the prompt. Blocker resolutions are retained only where they entered as the reply to an `ask_human` call the judge accepted, and they stay in their original dialogue position. Reasoning traces remain in the supervised target.

On the Hint corpus the teacher issued $36{,}567$ questions, of which the expert answered $16{,}614$ (45.4%) with `irrelevant question`, and 46.5% of Hint episodes contain at least one rejected ask. On the Description corpus the rejection rate is 28.8%.

The full episode is tokenized once as a *trunk*, and each supervised turn is a *branch* whose input is

$$\texttt{trunk_ids}[:\texttt{share}_k] \;+\; \texttt{patch_ids}_k \;+\; \texttt{target_ids}_k, \tag{10}$$

where only $\texttt{target_ids}_k$ carries loss. The per-branch patch exists because chat templates are not guaranteed prefix-stable, so each branch is reconciled against the shared trunk and any divergence is materialized explicitly.

`Ask-All` supervises every retained teacher turn, including asking turns the judge rejected. `Ask-Clean` omits the branch whose target is a judge-rejected ask, so no gradient is taken on that turn. The rejected ask and the expert's `irrelevant question` reply remain verbatim in the trunk, which is the context every later branch reads. The mechanism is target-level loss masking. The cleaning flag is the only configuration key that differs between an `Ask-All` arm and its `Ask-Clean` sibling, and both arms carry identical action-turn targets.

D TRAINING AND EVALUATION CONFIGURATION

All four recipes share the configuration in Table 6 and differ in their training corpus, Hint-derived or Description-derived, and in the `Ask-Clean` flag. The minimum learning rate equals the initial learning rate, so the schedule is constant after warmup. We report each arm's final checkpoint, selected against no evaluation metric. We report aggregate mean and standard deviation over five seeds and do not tabulate individual per-seed values. Every comparison in Section 6 is descriptive and we report no significance tests.

Table 6: Training configuration, identical across the four PSD recipes, and the evaluation configuration used for all of them.

Training	**Value**	**Evaluation**	**Value**
Base model	Qwen3.8-27B	Task set	100 public HiL-Bench SWE
Reasoning	Enabled	Blockers	353
Adaptation	Full fine-tune	Setup	`AskHuman`, no cue
Optimizer	AdamW, $\epsilon = 10^{-8}$	Harness	SWE-Agent
Weight decay	0.01	Temperature	1.0
Learning rate	1×10^{-6}	Maximum turns	200
Minimum learning rate	1×10^{-6}	Judge	Llama-3.3-70B-AWQ
Warmup	5 iterations	Correctness	Executable tests
Gradient clipping	1.0	Seeds	5
Epochs	1	Uncertainty	Std. dev. over seeds
Global batch size	40 sequences		
Micro-batch size	1		
Sequence packing	Disabled		
Max sequence length	131,072 tokens		
Corpus length cap	126,976 tokens		
Tensor parallelism	8		
Pipeline parallelism	4		
Random seed	42		
Training tasks	$\approx 4{,}100$		

AI ASSISTANCE

Large language models (LLMs) were used during the preparation of this work for writing assistance and language polishing, retrieval and discovery of related work, and research ideation and execution. LLMs were also used as part of the experimental methodology to generate synthetic data, as described in the paper. AI tools assisted with drafting and revising portions of the manuscript. All AI-generated content, experimental procedures, analyses, and citations were reviewed and verified by the authors, who take full responsibility for the content of the paper.