ASKING FOR WHAT WAS NEVER REQUESTED: HORIZONTAL AND VERTICAL PROACTIVITY IN AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An agent that uses tools typically responds to what the user explicitly asks, yet completing the task may require information the user never requested. Work on proactive agents mainly studies whether and when an agent should act on its own, not what information it should pursue. We study a distinct axis of proactivity: its content. Horizontal proactivity pursues unstated information that the current context already identifies, and vertical proactivity pursues needs that only earlier evidence reveals. A need graph, recovered from a benchmark's own decomposition, records which needs depend on which, so both forms, and whether the agent stops at the right time, can be scored from a transcript without a model judge. To learn this behavior, we propose Q&D (questioner and drafter), which trains a questioner to prefer the question whose continuation retrieves more of the required evidence, with no reward model or judge. On held-out splits of three multi-hop question-answering benchmarks, at equal retrieval spend, the trained questioner improves both forms of proactivity over the same model, prompted, and outperforms a prompted model $15\times$ larger in the same role on two of the three, and the gain persists after controlling for question volume and length. Without further training, we place the questioner in an interactive customer-service agent with a simulated customer, where it completes more tasks while asking fewer questions, and in retail it outperforms the $15\times$ larger model with fewer follow-up turns from the customer. These results show that proactivity depends not only on whether an agent acts without being asked, but also on what it chooses to pursue and when it stops.

1 INTRODUCTION

A tool-using agent usually does what the user asks, yet a task often needs information the user never mentions (Lu et al., 2025). A customer who asks a store's agent to change the boots in a pending order to size 8, same material, gives a name and ZIP code but not the order or an email they no longer remember (Figure 1a). An agent that waits for the customer to supply these asks for the email and order number sixteen times, and the boots are never changed. A proactive agent instead looks up the account from the name and ZIP code, then the order holding the boots, then size-8 boots in their material, and completes the task. Pursuing a need the current state already names, like the account, is horizontal proactivity, and pursuing one that only newly found evidence names, like the order and then the size-8 boots, is vertical proactivity. Without either, an agent stalls or hands the work back to the user as follow-up questions.

Work on proactive agents asks whether and when an agent should act on its own (Horvitz, 1999; Lu et al., 2025; Tang et al., 2026b), and rarely what information it should seek unasked. Agentic search methods decide what to retrieve next and when to stop, but they either follow the chain the question itself spells out (Press et al., 2023; Trivedi et al., 2023) or are trained only on whether the final answer is correct (Jin et al., 2025; Chen et al., 2025b), so what each question pursues is neither measured nor rewarded. Evaluations can also mistake asking more for asking better, since many imprecise questions may find as much as a few precise ones (Kapoor et al., 2025; Erol et al., 2026).

We make the content of proactivity measurable with what we call a need graph: the evidence a task requires, with an edge wherever one need can be named only after another is found. Multi-hop question-answering benchmarks supply these graphs naturally through their own decompositions.

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

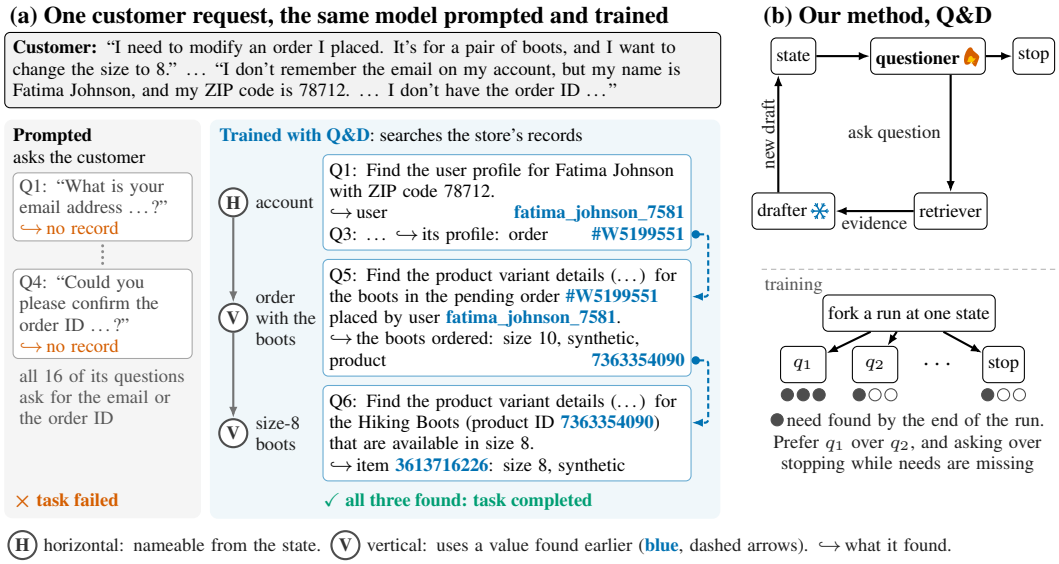


Figure 1: Trained with Q&D (questioner and drafter), our algorithm, an agent finds for itself what a prompted agent asks the customer to supply. (a) A retail customer-service task. A dedicated prompted agent asks for the email and order number, which the customer cannot give, and the boots are never changed. The same model, trained, looks up the account from the name and ZIP code (horizontal), then the order and the size-8 boots from values it found (vertical), and completes the task. (b) Q&D: the trained questioner (flame) asks one question or stops, and a frozen drafter (snowflake) rewrites the draft. Training forks a run at one state and prefers the candidate whose continuation finds more of the task's needs (dots).

A run is scored from its transcript against the graph, with no model judge, and agents are compared after the same number of questions, so asking more cannot pass for asking better.

We propose Q&D (questioner and drafter), an algorithm that embeds proactive information-seeking in an agent. It gives the agent a questioner, which asks one question at a time or stops, and a frozen drafter, which folds evidence into a draft. Because the drafter is frozen, every change in what the agent holds is caused by a question, so each question can be credited with what followed it. To train the questioner, we fork a run at one step (Figure 1b), continue it after several candidates, and prefer the question whose continuation finds more required evidence, so a question that reaches an unstated need early wins. While evidence is still missing, asking is preferred over stopping. Every label comes from what followed a question, so Q&D needs no reward model or judge. The questioner never sees a graph, so it can run where none exists.

Training on consequences makes the agent proactive in both directions. On held-out tasks from three multi-hop question-answering benchmarks (Trivedi et al., 2022; Geva et al., 2021; Ho et al., 2020), where a question is a search query over the task's evidence, the trained 8B questioner recovers more of the unstated evidence than the same model, prompted, both what the current state already names and what only newly found evidence names. On the benchmark whose questions need the most lookups in sequence, it finds 90% of the required evidence against 78%. The gain comes from what it asks, not from asking more or longer questions, and the 8B questioner also leads a prompted model $15\times$ larger in the same role on two of three benchmarks. Two problems remain: training teaches what to ask more readily than when to stop, and the extra evidence does not yet reach answers.

Q&D also generalizes beyond question answering. Without further training, in a customer-service agent serving a simulated customer (Barres et al., 2026), it more than doubles retail task success, and, acting proactively, it completes more retail tasks than the $15\times$ larger model with fewer follow-up turns from the customer, doing work the customer would otherwise supply.

We make two contributions.

- A measurable content axis of proactivity. We define horizontal and vertical proactivity on a need graph and propose evaluation metrics for both and for stopping.

Table 1: What is read from a run against its need graph. V is the set of required needs, S_t the evidence held after t questions (S at the end), d_v a need's depth, and C_d the coverage at depth d. Appendix A gives each estimator and its edge cases.

quantity	definition
required-evidence coverage	share of the required needs recovered, $\lvert V \cap S\rvert/\lvert V\rvert$
breadth	independent lines in which the run resolves a need at depth ≥ 1
depth-weighted recall	$\sum_d w_d C_d / \sum_d w_d$ with $w_d = 0.5, 1, 1.5, 2$ for $d = 0, 1, 2$ and 3 to 5
deepest need resolved	$\max\{d_v : v \in V \cap S\}$, the level the run reached
out-of-order rate	share of resolved needs taken before a prerequisite (lower is better)
stop-when-done	$\Pr[\text{stop} \mid V \subseteq S_t]$ over the decision points the policy reaches
ask-when-not-done	$\Pr[\text{ask} \mid V \not\subseteq S_t]$ over the same decision points

- Q&D, an algorithm that embeds both forms of proactivity in an agent by training its questioner on what each of its questions goes on to retrieve, with no reward model or judge.

2 A NEED GRAPH MAKES THE CONTENT OF PROACTIVITY MEASURABLE

We grade the content of proactivity by where each question goes next. At any point of a run, the frontier holds the needs the agent can already name but has not resolved. Vertical proactivity goes deeper: it pursues a need that the need just resolved made nameable, as the account's record names the order holding the boots. Horizontal proactivity goes wider: it pursues any other need on the frontier, one the request implied, like the account itself at the start, or one that earlier evidence opened, like another order in the same account. The two are separable: a policy can resolve every surface need and follow no chain, or follow one chain and miss the rest.

A need graph makes this measurable. For each task it lists the needs, the units of evidence the task requires, with a prerequisite edge wherever one need can be named only after another is resolved. A need can open several others, so the graph branches wherever a need does, as an account opens each of its orders. A need's depth is the length of the longest prerequisite path above it, so needs at depth zero can be named from the request and deeper ones only after the evidence above them. A need is resolved when the retriever returns its gold paragraph, matched by identifier. Needs joined by prerequisite edges form one independent line of inquiry. We recover the graphs mechanically from the decompositions the benchmarks ship (Section 4).

We propose the evaluation metrics in Table 1, each read from a finished run against its graph. Breadth, the number of independent lines a run advances past their first need, reads horizontal proactivity, and depth-weighted recall and the deepest need resolved read vertical proactivity. Required-evidence coverage, the recall counterpart of the supporting-paragraph score of Trivedi et al. (2022), reads both, including branches inside a line. Two rates read stopping over the states each policy reaches: whether it stops once everything is found, and whether it keeps asking while it is not. Where a user is in the loop, proactivity should also spare them: we count the user's follow-up turns, read together with task success, since giving up also spares the user. The agent never sees a graph: graphs only score runs and label training data (Appendix N).

3 Q&D TRAINS A QUESTIONER ON WHAT ITS QUESTIONS RETRIEVE

Q&D splits the agent into a questioner, which decides what to ask and when to stop, and a drafter, which keeps the answer (Figure 1b). At each step the questioner sees the state: the task, the evidence so far, the current draft and its past questions. It then asks one question, which a retriever answers from the task's evidence pool, or stops. A frozen drafter rewrites the draft from the evidence. At the end a frozen answerer, the same in every arm and held to a fixed word cap, writes the final answer from the task, the evidence and the draft, so only the questioner differs between policies, and answer length favors none of them. The draft exposes the frontier: once a need's prerequisite is in the draft, the need can be named, so at every step the questioner chooses between going deeper along the chain the last evidence opened and opening a need already nameable beside it.

Because the drafter is a fixed function of the evidence, every change in the state is caused by a question, so Q&D labels each decision by its consequences, not its wording. A recorded run is forked at a step, eight alternative questions are sampled there, and each is continued to the end by the policy that produced the run (Figure 1b, bottom). The candidates share the task, the evidence and the history, so they differ only in the question asked. Pairs are ordered by consequence alone: first by whether the run answered the task, which decides 6% of the pairs we train on, then by which reached the complete evidence sooner, then by how much evidence each turn added. The preferred question is thus usually the more proactive one, which reaches the needs the request left unstated sooner, whether by going deeper or wider. At a state the need graph marks unfinished, asking is ranked above stopping. Training needs tasks whose required evidence and prerequisites are known, as multi-hop benchmarks provide, but the trained questioner reads no graph, which is why it can serve a customer-service agent that has none.

The questioner is trained in three stages. It first imitates good decisions: where the required evidence was already in hand the target is to stop, and where it was not, the best sampled question if its consequence score clears a fixed floor (Appendix D). It is then trained by direct preference optimization (Rafailov et al., 2023) on question pairs, and last on question pairs and stop contrasts together. Stop contrasts rank asking above stopping at unfinished states, the lesson imitation cannot give, because it writes no target where no sampled question clears the floor (Section 5).

4 Experimental setup

Training data are mined from MuSiQue (Trivedi et al., 2022), StrategyQA (Geva et al., 2021) and 2WikiMultiHopQA (Ho et al., 2020), whose need graphs cover 800, 2,290 and 12,576 tasks. Results are read on held-out test splits of 200 tasks per benchmark, which we call suites, at two rollout seeds each, and the few read on development tasks say so. MuSiQue carries the largest share of needs at depth two or more, StrategyQA the decompositions its questions leave implicit, and 2WikiMultiHopQA the only natural branching. Checkpoints were selected on 132 MuSiQue and 333 StrategyQA development tasks (Appendix D). FRAMES, which ships no need graph, is scored on whether the gold answer or an alias appears in the answer (Krishna et al., 2025), and transfer is read on the retail and airline domains of τ^2-bench (Barres et al., 2026), neither used for training.

Need graphs are recovered mechanically from structure each benchmark ships, so no person and no model wrote a need or an edge. MuSiQue writes a later step as, for example, "What is the birthplace of #1?", its authors' statement that the step waits on step one. StrategyQA carries the same references on 2,999 of its 6,720 steps, and 2WikiMultiHopQA links a need to the one whose subject is its object (Appendix A). Depth here is therefore depth in the benchmark's decomposition.

The questioner is Qwen3-8B with a low-rank adapter (Yang et al., 2025; Hu et al., 2022), and we report two training seeds of its final stage. The main comparator is the same model, prompted with the same template. Beside it we run GPT-OSS-120B, a model $15\times$ larger, prompted (Agarwal et al., 2025), Claude Opus 5 prompted plainly, a baseline that never asks, and the structured retrieval algorithm PAR^2-RAG (Li et al., 2026), reimplemented with our retriever and drafter and run on Qwen3-8B, GPT-OSS-120B and Claude Opus 5. The drafter and the answerer are GPT-OSS-120B in every arm except the single-model control of Section 5. Retrieval is BM25 over each task's released pool of 20 paragraphs (10 on 2WikiMultiHopQA), returning the top 5, 2 and 3 on MuSiQue, StrategyQA and 2WikiMultiHopQA deterministically, and cost is counted in retrieval calls, one per question.

Two readings are kept apart. Equal spend reads both policies, on each task, at the lower of their two question counts: since the questioner is never told a budget, its first k questions are the ones a budget of k would have produced, so the reading is one of evidence per question. Own stop, the second reading, lets each policy decide for itself when to stop, with at most eight calls, and reads it where it stopped. Each contrast is paired at the task, whose value averages its two rollout seeds and, for the trained questioner, its two training seeds, and intervals are bias-corrected and accelerated bootstraps over tasks (Efron, 1987), printed at 10,000 resamples. We call a cell decided when its intervals from three independent 50,000-resample bootstrap runs all exclude zero, a guard against resampling error rather than a multiplicity correction, and no question-answering contrast was registered in advance as a confirmatory test. Each comparator is read at its own lower count, so contrasts do not subtract.

Table 2: At equal spend, the trained questioner recovers more of the required evidence than the same model, prompted, on every suite, and more than GPT-OSS-120B on two of three. Each column pair gives a baseline's level and Q&D's over 200 held-out tasks per suite, the baseline being the same model, prompted, except on rows that name GPT-OSS-120B. At equal spend a pair is read at its lower question count, so Q&D's level varies by comparator. Bold marks the better of a pair where the difference is decided (Section 4), and ↓ a measure where lower is better. Differences in the text are computed before rounding (Table 10).

	MuSiQue		StrategyQA		2WikiMultiHopQA	
	Baseline	Q&D	Baseline	Q&D	Baseline	Q&D
Equal spend						
Required-evidence coverage (%)	78.3	**89.5**	78.4	**85.5**	88.2	**93.1**
Depth-weighted recall (%)	77.9	**90.5**	51.2	**56.6**	84.8	**90.3**
Deepest need resolved (depth)	1.53	**1.75**	0.54	**0.64**	0.57	0.63
Breadth (lines)	0.87	**0.99**	0.47	**0.52**	0.77	**0.84**
Out-of-order rate (%) ↓	**17.1**	22.8	11.5	12.0	4.4	3.2
Coverage vs GPT-OSS-120B (%)	77.8	**84.8**	80.7	**85.0**	**92.4**	88.9
Own stop (at most eight calls)						
Required-evidence coverage (%)	81.6	**91.2**	87.6	86.4	94.6	96.1
Answer token F1 (%)	41.1	43.6	43.5	43.1	61.4	64.5
Questions per task	5.9	4.2	6.0	3.0	3.0	2.3
Coverage vs GPT-OSS-120B (%)	81.1	**91.2**	**90.0**	86.4	96.8	96.1
Questions vs GPT-OSS-120B	5.3	4.2	6.1	3.0	3.1	2.3

5 RESULTS

Q&D recovers more of the required evidence, and more of it deep. Q&D makes the agent more proactive on every held-out suite. After the same number of questions, the trained questioner recovers more of each task's required evidence than the same model, prompted, by 11.2 percentage points on MuSiQue, 7.0 on StrategyQA and 5.0 on 2WikiMultiHopQA (Table 2). The gain is stable: each training seed alone is decided on all three suites, and all four training runs agree within a standard deviation of 0.8 points (Appendix L). The first training stage alone (imitation, Section 3) also raises coverage at every base size from 1.7B to 8B, on development tasks (Appendix J).

Much of the extra evidence is deep, which is vertical proactivity: depth-weighted recall rises by 12.5, 5.4 and 5.5 points, most on MuSiQue, whose chains run deepest. The deepest need a run resolves also lies deeper, by 0.22 levels on MuSiQue and 0.10 on StrategyQA (Table 2). Breadth rises as well, so the questioner goes deeper without narrowing its search. We attribute the depth gain to the labels, which prefer the question whose continuation reaches all the required evidence sooner and so favor starting a chain early enough for later questions to follow it.

Where a task needs both directions at once, the questioner is more proactive in both. On synthetic tasks built from two to four independent lines of inquiry, chains of needs independent of one another, each several steps deep, it advances more of the lines than the same model, prompted, when a task has two or three lines, ties with four, and recovers more of the deep evidence in every case (Appendix A). On real data, when two held-out MuSiQue questions are joined into one task whose answer needs both chains, it advances more of the two chains past their first step and recovers 15.2 points more of the required evidence, as its gains on each question alone predict (Appendix A).

One side effect appears on MuSiQue: the trained questioner more often finds a deep piece of evidence before the one it depends on, by 5.7 points (the out-of-order rate in Table 2). It comes from paragraphs retrieved by chance, not from what the questions ask for: the questions themselves skip ahead no more often than the prompted model's, by 1.2 points spanning zero (Appendix E).

Better questions also make the evidence cheaper. For each required need it finds, the trained questioner spends 40% fewer tokens on MuSiQue, 11,610 against 19,354, and fewer on the other two suites as well (Appendix C). This ratio needs no matching of spend, so the gain does not come from how spend is matched. We attribute the saving to proactive questioning: each question the trained questioner asks recovers more of the required evidence.

The gain comes from what Q&D asks. What the questions say matters, not their number. Replacing the trained questioner's questions with as many drawn at random from other tasks of the same benchmark loses 34.7 points of coverage on MuSiQue and 63.3 on StrategyQA (Appendix F).

The trained questioner's later questions build on what earlier ones found, which is vertical proactivity at work: if its questions ignored what it read, hiding the evidence would cost nothing. On MuSiQue, removing the evidence from its state costs 8.4 and 8.7 points of coverage, and letting it plan its questions from the request alone costs 4.3 and 6.4 (per training seed). On StrategyQA neither makes a detectable difference, because there the ablated questioners ask little more than one question.

The gain comes from training the questioner, not from splitting the agent in two. When the trained questioner also writes the draft itself, it stays within a 5-point equivalence margin: equivalent at its own stop, and 2.2 points behind the split at equal spend (Appendix F).

Longer questions do not explain the gain either. The trained questioner's questions are longer, 17.5 words against 13.1 on MuSiQue, but when the prompted model may spend as many question tokens, it still recovers 14.9 and 4.7 points less on MuSiQue and StrategyQA development tasks (Appendix C). The advantage lies in what the trained questioner asks, not in how much it writes.

People recognize these proactive questions but differ on which is best. Two human raters agree on 88% of candidate questions on whether they reach a need the request never states (Cohen's $\kappa = 0.76$), but on only 68% of the pairs both judged about which question is the better move ($\kappa = 0.24$), and each agrees with our rule on 65% and 68% of pairs, against 50% by chance (Appendix I). Here, a better question is one that recovers more of the required evidence, not one people prefer.

Q&D's 8B questioner leads one $15\times$ larger on two of three suites. Training a small questioner can beat scale (Figure 2). At equal spend, the trained 8B questioner recovers more of the required evidence than GPT-OSS-120B, prompted, a model $15\times$ larger in the same role, on MuSiQue and StrategyQA, by 7.0 and 4.3 points, and trails it on 2WikiMultiHopQA, by 3.5 points (Table 2). The lead follows depth: it is largest on MuSiQue, whose chains run deepest, and reversed on 2WikiMultiHopQA, whose needs lie at most one step deep, which suggests that learned proactivity pays most where evidence must be followed step by step.

The trained questioner's advantage is efficiency, which call budgets make visible (Figure 2a). On MuSiQue it leads both prompted models at every budget from 4 to 24 calls, and allowed only 4 calls it recovers 4.1 points more of the required evidence than GPT-OSS-120B allowed 24, using about a quarter of the calls (Appendix G). Elsewhere the prompted models catch up at larger budgets: on StrategyQA the trained questioner ties the same model with fewer questions and trails GPT-OSS-120B once it spends more, and on FRAMES it trails GPT-OSS-120B at budgets of 16 and 24.

A hand-designed alternative does not close the gap either. PAR^2-RAG (Li et al., 2026), a structured retrieval algorithm that plans sub-questions and checks whether the evidence suffices, never beats plain prompting of its own model at equal spend on any of three models (Figure 2b, Appendix C), and the trained questioner leads it on every model on MuSiQue. The one comparator ahead of the trained questioner at equal spend is Claude Opus 5, prompted plainly, on MuSiQue, by 2.9 points. On FRAMES, when each policy stops on its own, the one decided difference is that PAR^2-RAG run with Claude Opus 5 answers better, by 6.1 points, while asking more questions, 4.5 against 3.6.

Stopping on its own, Q&D asks fewer questions and still leads on the deepest chains. When each agent decides for itself when to stop, with at most eight calls (own stop), the trained questioner asks fewer questions than either prompted model on every suite and keeps its lead on MuSiQue, by 9.6 points over the same model, prompted, and 10.1 over GPT-OSS-120B (Table 2). On the other two suites it does not differ detectably from the same model, and on StrategyQA it trails GPT-OSS-120B by 3.6 points, while that model asks twice as many questions.

The trained questioner stops early by choice, not because it runs out of calls. It uses its whole budget in only 34% of MuSiQue runs when allowed 4 calls and in 8% when allowed 24, against 13% to 69% for the prompted questioners across both suites and budgets (Appendix G).

What each stage of Q&D's training teaches, and why stopping lags. Each stage of Q&D's training teaches something different. Imitation, the first stage, copies good decisions, but at an unfinished state where no sampled question clears the floor it has nothing to copy, so 24,538 such

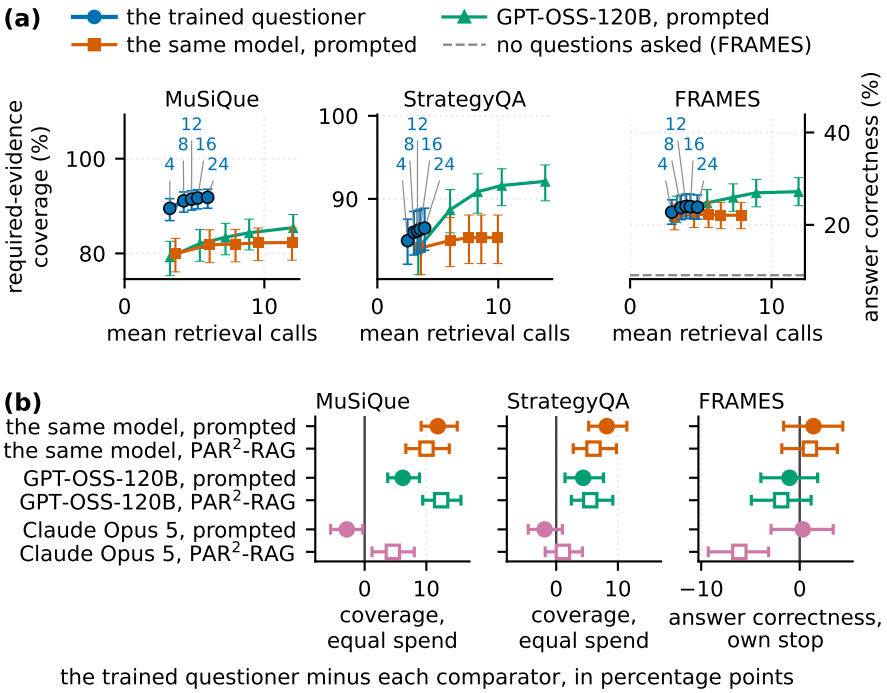


Figure 2: On MuSiQue the trained questioner gets more evidence from fewer calls, and a hand-designed retrieval algorithm does not close the gap. (a) Coverage on MuSiQue and StrategyQA and answer correctness on FRAMES against the calls used on average, when each policy stops on its own under a budget of 4, 8, 12, 16 or 24 calls (numbered on the trained questioner's points). (b) The trained questioner minus three models, prompted plainly (filled) or running the structured retrieval algorithm PAR^2-RAG (Li et al., 2026) (hollow), in percentage points: coverage at equal spend and FRAMES answers at each policy's own stop. Bars are bootstrap intervals over tasks.

Table 3: What each training signal teaches, on development tasks (132 MuSiQue and 333 StrategyQA), each row training the same model on one signal (Section 3), the final stage on both. Gain is coverage over the same model, prompted, in points at a matched spend (Appendix C). Stop | done and ask | not done are the percent of decision points at which a policy stops once the required evidence is in and asks while it is not, and asks counts questions per task. Bold marks each column's largest value, not a tested difference, and no row leads on all three (intervals in Appendix G).

	MuSiQue				StrategyQA			
supervision	gain	stop | done	ask | not done	asks	gain	stop | done	ask | not done	asks
same model, prompted	—	15.3	96.4	6.29	—	14.6	96.3	5.59
imitation (first stage)	+4.9	95.1	86.8	3.05	+8.0	90.7	83.0	1.34
question pairs	**+16.7**	**98.7**	86.0	2.47	**+8.4**	**98.0**	83.4	1.28
stop contrasts alone	−0.5	10.1	**99.6**	6.95	−1.7	9.7	**98.1**	5.75
final stage, seed 1	+15.7	42.7	94.2	4.23	+8.2	54.6	91.2	2.13
final stage, seed 2	+15.4	42.7	94.5	4.24	+7.8	54.5	90.6	2.14

states drop out of its training data (Appendix D), and with them the lesson to keep asking. The second stage learns from question pairs, which prefer the better of two questions, and the final stage adds stop contrasts, pairs that prefer asking over stopping at unfinished states. In Table 3 the same model is trained on each signal alone: question pairs raise coverage but stop too early, stop contrasts alone almost never stop, and the final stage, trained on both, raises coverage about as much while asking when work remains, though it stops half as often as imitation once the work is done.

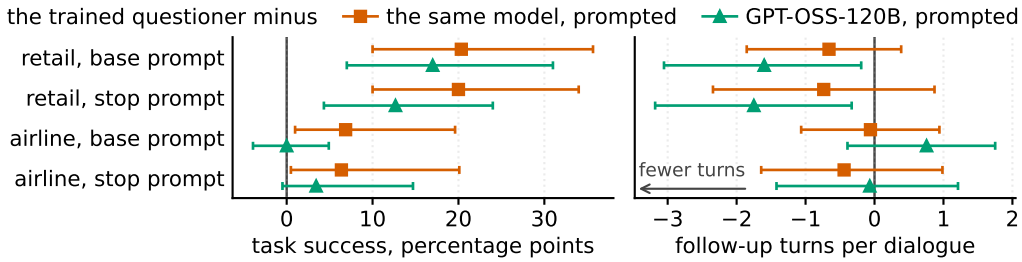


Figure 3: In retail, Q&D's trained questioner completes more τ^2-bench tasks than either prompted baseline, and needs fewer follow-up turns from the customer than GPT-OSS-120B. Each point is the trained questioner minus a baseline over 25 retail or 17 airline tasks under one of two prompt variants (Appendix H): task success in percentage points (left) and the customer's follow-up turns per dialogue (right). Bars are bootstrap intervals over tasks, each averaging both training seeds.

On held-out tasks the trained questioner stops more often once the work is done, by 29 to 33 points over the same model, prompted, on every suite and seed, and by 25, 43 and 12 points over GPT-OSS-120B (Appendix G). Where evidence is still missing, it keeps asking as often as the prompted model on MuSiQue and 2WikiMultiHopQA, but 7 points less often on StrategyQA.

The extra evidence does not yet show in the answers, for two reasons. On MuSiQue the trained questioner finds the evidence the answer rests on 9.7 points more often, which at the prompted model's answer rates implies an answer gain of about 4.3 points, below the 4.7 our sample can detect and consistent with the 2.4 observed (Appendix G.1). On StrategyQA it stops before finding that evidence, by choice in 92% and 86% (per seed) of the runs that end without it, against 39% for the prompted model, and supervising that stop directly does not help (Appendix G).

Q&D's proactivity generalizes to a customer-service agent. Proactivity learned with Q&D is a general behavior, not a skill tied to question answering. Placed with no further training in a customer-service agent on τ^2-bench, where a simulated customer brings a task and each dialogue allows sixteen questions, the trained questioner completes more tasks than the same model, prompted, in both domains (Figure 3). In retail, success rises from 13% and 12% to 34% and 32% under two prompt variants (Appendix H), better on ten of twenty-five tasks and worse on none in each, and in airline it rises by 6.9 and 6.4 points. It asks 1.3 and 1.4 fewer questions per retail dialogue, and more of them reach the records the task needs: it asks less and finds more.

Against a model $15\times$ larger, the agent completes more tasks while the customer supplies fewer turns, the payoff proactivity promises. GPT-OSS-120B, prompted in the same role and declared as a baseline before its dialogues ran, completes 17% and 19% of retail tasks. The trained questioner beats it in retail by 17.0 and 12.7 points of success, with 1.6 and 1.8 fewer follow-up turns from the customer, both decided. Against the same model, prompted, follow-up turns do not differ detectably, and in airline no difference from GPT-OSS-120B is decided (Appendix H). We attribute the fewer turns to proactive questioning: the agent finds more of what the task needs in the store's records.

6 Related Work

Learning what to ask. The nearest ancestors supervise a question by what it causes, by simulating the turns that follow it (Zhang et al., 2025), contrasting clarifying with answering (Chen et al., 2025a) or pairing terminate-against-continue trajectories (Liu et al., 2025). Others learn from counterfactual information gain (Kong et al., 2026), causal preferences (Zhang et al., 2026a), task success (Yao et al., 2026), a judge's verdict or the user effort spared (Acikgoz et al., 2026; Sun et al., 2026b; Zhao et al., 2026; Qian et al., 2025b), or self-play with a simulated user (Andukuri et al., 2024). Expected value of information (Rao & Daumé III, 2018; Aliannejadi et al., 2019) needs a model of what an answer will do, where ours needs only the answer, and ranking sampled continuations by outcome also underlies step-level preference training (Wang et al., 2024a; Lai et al., 2024).

Proactivity. Accounts of initiative ask whether and how far an agent acts on its own (Horvitz, 1999; Händler, 2023; Sapkota et al., 2026) and what triggers it (Bui & Evangelopoulos, 2026). The nearest

content-side work grades implied needs under a model judge, without asking whether a need could have been named from the task (Harfi et al., 2026). Benchmarks that isolate a missing variable, an ambiguous query or a need not yet stated score whether a model recognizes the gap (Li et al., 2025a; Deng et al., 2026; Tao et al., 2026; Wu et al., 2026a), and simulated and recorded users score the timing and form of an intervention (Tang et al., 2026a; Nathani et al., 2026; Tang et al., 2026b; Qian et al., 2025a). These measure the decision or its timing. We measure what was pursued.

Agentic search and retrieval. Interleaving reasoning with actions (Yao et al., 2023) and decomposing a question into retrieval steps (Press et al., 2023; Trivedi et al., 2023) follow prerequisite chains by design. Decomposition formalisms write a question as a graph of dependent steps (Wolfson et al., 2020) that prompting can solve in order (Zhou et al., 2023). Our need graphs have that shape, but their nodes are the evidence a task requires, and they only score runs and are never shown to the policy. Policies trained to interleave search with reasoning use outcome rewards (Jin et al., 2025; Chen et al., 2025b), and adaptive retrieval learns when to retrieve (Asai et al., 2024; Jiang et al., 2023; Jeong et al., 2024). Structured retrieval instead changes the pipeline, planning and checking sufficiency (Li et al., 2026), synthesizing a program of retrieval calls (Sun et al., 2026a), or curating the pool or a corpus graph (Nahid & Rafiei, 2025; Wang et al., 2024b; Cahoon et al., 2025; Zhu et al., 2026; Liu et al., 2026). Holding the retriever, pool and drafter fixed, we compare against PAR^2-RAG, whose mechanism is a sequence of ask-or-stop decisions.

Length and model judges. Preference training tracks response length (Singhal et al., 2024; Liu et al., 2024; Dubois et al., 2024) and model judges carry position and self-preference bias (Zheng et al., 2023; Wataoka et al., 2024; Norman et al., 2026), so the content claim rests on a length control, no quantity is model-judged, and agreement is chance-corrected (Cohen, 1960; Krippendorff, 2019).

7 Limitations

The two reported seeds share one supervised checkpoint and one pair export with two further runs, so intervals are over tasks alone and do not estimate variation from that checkpoint and export. No checkpoint met the pre-committed selection rule as written, and the reported configuration was kept on a second ground fixed in advance (Appendix D). Training labels and the primary quantity share the need graphs, answers decide only 6% of training pairs, and answers do not yet move detectably. The released retrieval pools are small enough that most prerequisite edges do not gate retrieval, so the vertical result concerns depth in a decomposition (Appendix I), and on MuSiQue and StrategyQA, with one line of inquiry per task, breadth records only whether a run got past its first need. Candidate questions and rollouts come from language models, two human raters checked the label rule on one suite, and every user-facing number comes from a simulated customer. Q&D is one round of off-policy training, not compared with on-policy reinforcement learning or iterated on its own rollouts, and the length control was not run held-out. Prompt search and scale are in Appendices K and J.

8 Conclusion

We introduced the content of proactivity, what an agent pursues unasked, as an axis beside whether and when it acts, and made its horizontal and vertical forms measurable with a need graph. We proposed Q&D, which trains an agent to question proactively, learning from the consequences of its questions, with no reward model or judge. At equal spend, the trained 8B questioner recovers more of the required evidence than the same model, prompted, reaches deeper into each task's dependencies, and leads a prompted model $15\times$ larger in the same role on two of three benchmarks. The gain lies in what it asks, not how much, and is largest where evidence must be followed step by step. It generalizes to a customer-service agent, completing more tasks with fewer questions, and in retail it outperforms the larger model with fewer follow-up turns from the customer: a proactive agent does work the user would otherwise supply. Training teaches what to ask more readily than when to stop, so next are stop supervision that carries the evidence gain into answers and retrieval pools where a prerequisite must be found before its dependents.

Reproducibility statement

Reproduction here means cache replay rather than bitwise determinism, because two identical requests to a batched decoder can differ. The released artifact is therefore the content-addressed request cache: given it, the numbers here regenerate with no network access and no keys, and the appendix records where a replay and the recorded run disagree. The pipeline hands off from rollout to compaction to scoring, and re-scoring never re-rolls: a scoring identity over the metric definitions, the need graphs and the identifier matcher, which uses no model judge for any quantity reported here, writes new rows and leaves old ones untouched. The released records carry, for every number in the main text, its run identifiers, that scoring identity and the version of the need graphs. Appendix N gives the prompts and the arrangement that keeps the need graphs unreachable from any policy. Rendered tables are locked to the hash of the records they were built from. Code, the data shards and the trained adapters are released as supplementary material, anonymized. A third run of the final stage, set aside before the other two were read because a trainer defect kept about a tenth of its training, is reported in Appendix L.

Ethics statement

A corpus of real sessions between a person and an agent was collected for an annotation study this paper does not report, and no row of it reaches the questioner reported here. Those sessions belong to one contributor, who consented in writing to their being ingested, scrubbed, labeled and used as data, and the build checks that consent record. Third parties appear in any working session and cannot consent through its owner, so neither the raw transcripts nor the corpus is redistributed. The sessions carried live credentials, among them 129 GitHub tokens, and the scrubber replaces each secret with a stable opaque handle, re-scans its own output, and aborts the build on any residue. Every user-facing number comes from a simulated user, every rater of the training corpus is a language model, one of which was replaced after it was measured near chance, and the only human rating is two people on one suite, reported in Appendix I.

AI use statement

Language models wrote the candidate questions and the dialogue the training corpus is built from. Language models served as the raters and proposers that labeled that corpus and as the instruments that validate those labels, a limitation this paper states. Code was written with AI-assisted tooling and verified by the test suite, by contracts that keep the need graphs out of the policy path, and by the authors. The manuscript was edited with AI assistance, and every claim and number was checked by the authors against the provenance records released with the code. References whose venue could not be confirmed are cited as preprints. We did not use generative AI to produce, adjust or select any experimental measurement, to choose which results are reported, or to originate the research question or the study design. We have reviewed all AI-assisted work and take full responsibility for this paper's content, including text, claims and artifacts produced with generative AI.

References

Emre Can Acikgoz, Jinoh Oh, Jie Hao, Joo Hyuk Jeon, Heng Ji, Dilek Hakkani-Tur, Gokhan Tur, Xiang Li, Chengyuan Ma, and Xing Fan. Speakrl: Synergizing reasoning, speaking, and acting in language models with reinforcement learning. In *Proceedings of the 16th International Workshop on Spoken Dialogue System Technology*, pp. 312–325, 2026.

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*, pp. 475–484, 2019.

Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah Goodman. STar-GATE: Teaching language models to ask clarifying questions. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=CrzAj0kZjR.

Ron Artstein and Massimo Poesio. Survey article: Inter-coder agreement for computational linguistics. *Computational linguistics*, 34(4):555–596, 2008.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International conference on artificial intelligence and statistics*, pp. 4447–4455. PMLR, 2024.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik R Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=OC2z7iSQKa.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

Nghi DQ Bui and Georgios Evangelopoulos. Agentic coding needs proactivity, not just autonomy. *arXiv preprint arXiv:2605.06717*, 2026.

Joyce Cahoon, Prerna Singh, Nick Litombe, Jonathan Larson, Ha Trinh, Yiwen Zhu, Andreas Mueller, Fotis Psallidas, and Carlo Curino. Optimizing open-domain question answering with graph-based retrieval augmented generation. In *Proceedings of the 1st workshop connecting academia and industry on Modern Integrated Database and AI Systems*, pp. 1–11, 2025.

Maximillian Chen, Ruoxi Sun, Tomas Pfister, and Sercan Arik. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *International Conference on Learning Representations*, volume 2025, pp. 32244–32279, 2025a.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, ZhouYijie, Chenzheng Zhu, Haofen Wang, Jeff Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and weipeng chen. Research: Learning to reason with search for llms via reinforcement learning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 85287–85307. Curran Associates, Inc., 2025b. doi: 10.52202/085713-2858. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/7b3a0d3a0864b66f229c0a9c84f9e950-Paper-Conference.pdf.

Yuxin Chen, Yi Zhang, Zhengzhou Cai, Yaorui Shi, Zhiyuan Yao, Chenhang Cui, Jingnan Zheng, Yaqi Huo, Xi Su, Qi Gu, Xunliang Cai, Xiang Wang, An Zhang, and Tat-Seng Chua. Vitabench 2.0: Evaluating personalized and proactive agents in long-term user interactions. *arXiv preprint arXiv:2605.27141*, 2026.

Yizhou Chi, Jessy Lin, Kevin Lin, and Dan Klein. Clarinet: Augmenting language models to ask clarification questions for retrieval. *arXiv preprint arXiv:2405.15784*, 2024.

Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 659–666, 2008.

João Coelho, Jingjie Ning, Jingyuan He, Kangrui Mao, Abhijay Sai Paladugu, Pranav Setlur, Jiahe Jin, Jamie Callan, João Magalhães, Bruno Martins, and Chenyan Xiong. Deepresearchgym: A free, transparent, and reproducible sandbox for deep research. In Alistair Moffat, Falk Scholer, Keping Bi, and Charles L. A. Clarke (eds.), *Proceedings of the 2026 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR), Melbourne, VIC, Australia, 25 July 2026*, pp. 34–43. ACM, 2026. doi: 10.1145/3805713.3820409. URL https://doi.org/10.1145/3805713.3820409.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Alejandro Cuadron, Pengfei Yu, Yang Liu, and Arpit Gupta. SABER: Small actions, big errors — safeguarding mutating steps in LLM agents. In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems*, 2026. URL https://openreview.net/forum?id=En2z9dckgP.

Mingyi Deng, Lijun Huang, Yani Fan, Fanqi Kong, Jiayi Zhang, Fashen Ren, Jinyi Bai, Fuzhen Yang, Dayi Miao, Zhaoyang Yu, Yifan Wu, Yanfei Zhang, Fengwei Teng, Yingjia Wan, Song Hu, Yude Li, Xin Jin, Conghao Hu, Haoyu Li, Qirui Fu, Tai Zhong, Xinyu Wang, Xiangru Tang, Nan Tang, Chenglin Wu, and Yuyu Luo. Interactcomp: Evaluating search agents with ambiguous queries. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=AiawINF2T0.

Wenzhang Du. When+ 1% is not enough: A paired bootstrap protocol for evaluating small improvements. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 519–531. Springer, 2026.

Yann Dubois, Percy Liang, and Tatsunori Hashimoto. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=CybBmzWBX0.

Bradley Efron. Better bootstrap confidence intervals. *Journal of the American statistical Association*, 82(397):171–185, 1987.

Leon Engländer, Sophia Althammer, Ahmet Üstün, Matthias Gallé, and Tom Sherborne. Agents explore but agents ignore: Llms lack environmental curiosity. *arXiv preprint arXiv:2604.17609*, 2026.

Mehmet Hamza Erol, Batu El, Mirac Suzgun, Mert Yuksekgonul, and James Y Zou. Cost-of-pass: An economic framework for evaluating language models. In *International Conference on Learning Representations*, volume 2026, pp. 34103–34127, 2026.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Model alignment as prospect theoretic optimization. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=iUwHnoENnl.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

Thorsten Händler. Balancing autonomy and alignment: A multi-dimensional taxonomy for autonomous llm-powered multi-agent architectures. *arXiv preprint arXiv:2310.03659*, 2023.

Sepehr Harfi, Ahmad Salimi, Dongming Shen, and Alex Smola. Proactbench: Beyond what the user asked for. *arXiv preprint arXiv:2605.09228*, 2026.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 159–166, 1999.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Tenghao Huang, Sihao Chen, Muhao Chen, Jonathan May, Longqi Yang, Mengting Wan, and Pei Zhou. Teaching language models to gather information proactively. In *EMNLP (Findings)*, pp. 15588–15599, 2025.

Xiaohongshu Inc. Vibesearchbench: Benchmarking long-horizon proactive search in the wild. *arXiv preprint arXiv:2605.27882*, 2026.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 conference of the north american chapter of the association for computational linguistics: Human language technologies (volume 1: Long papers)*, pp. 7036–7050, 2024.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 7969–7992, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Rwhi91ideu.

Sayash Kapoor, Benedikt Stroebl, Zachary S Siegel, Nitya Nadgir, and Arvind Narayanan. AI agents that matter. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=Zy4uFzMviZ.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. Abstentionbench: Reasoning llms fail on unanswerable questions. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-5729. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/fb122bfc3f0127a94ded048b5b03496f-Paper-Datasets_and_Benchmarks_Track.pdf.

Fanqi Kong, Jiayi Zhang, Mingyi Deng, Chenglin Wu, Yuyu Luo, and Bang Liu. InfoPO: Information-driven policy optimization for user-centric agents. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=O7UHBoLrx4.

Klaus Krippendorff. *Content Analysis: An Introduction to Its Methodology*. SAGE Publications, 4 edition, 2019. doi: 10.4135/9781071878781.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4745–4759, 2025.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

LangChain. Introducing ambient agents. https://www.langchain.com/blog/introducing-ambient-agents, 2025.

Belinda Li, Been Kim, and Zi Wang. Questbench: Can llms ask the right question to acquire information in reasoning tasks? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025a. doi: 10.52202/085713-4503. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/c42c8d51556fabb4b57fc86d3d3d0d09-Paper-Datasets_and_Benchmarks_Track.pdf.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 110–119, 2016.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. In *Advances in Neural Information Processing Systems 38*, 2025b. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/ae03bdef276132fae089692445725635-Abstract-Conference.html.

Xingyu Li, Rongguang Wang, Yuying Wang, Mengqing Guo, Chenyang Li, Tao Sheng, Sujith Ravi, and Dan Roth. PAR^2-RAG: Planned active retrieval and reasoning for multi-hop question answering. In Yunyao Li, Georg Rehm, and Mei Tu (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 1727–1737, San Diego, California, USA, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-394-4. doi: 10.18653/v1/2026.acl-industry.118. URL https://aclanthology.org/2026.acl-industry.118/.

Grace Liu, Yuxiao Qu, Jeff Schneider, Aarti Singh, and Aviral Kumar. CaRT: Teaching LLM agents to know when they know enough. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025. URL https://openreview.net/forum?id=kkKsmPeasY.

Jiate Liu, Zebin Chen, Shaobo Qiao, Mingchen Ju, Danting Zhang, Bocheng Han, Shuyue Yu, Xin Shu, Jinglin Wu, Dong Wen, Xin Cao, Guanfeng Liu, and Zhengyi Yang. A2rag: Adaptive agentic graph retrieval for cost-aware and reliable reasoning. In *2026 IEEE 42nd International Conference on Data Engineering Workshops (ICDEW)*, pp. 187–196. IEEE, 2026.

Wei Liu, Yang Bai, Chengcheng Han, Rongxiang Weng, Jun Xu, Xuezhi Cao, Jingang Wang, and Xunliang Cai. Length desensitization in direct preference optimization. *arXiv preprint arXiv:2409.06411*, 2024.

Hongliang Lu, Yuhang Wen, Pengyu Cheng, Ruijin Ding, Jiaqi Guo, Haotian Xu, Chutian Wang, Haonan Chen, xiaoxi jiang, and guanjunjiang. Search self-play: Pushing the frontier of agent capability without supervision. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZmGirmNJqE.

Yaxi Lu, Shenzhi Yang, Cheng Qian, Guirong Chen, Qinyu Luo, Yesai Wu, Huadong Wang, Xin Cong, Zhong Zhang, Yankai Lin, Weiwen Liu, Yasheng Wang, Zhiyuan Liu, Fangming Liu, and Maosong Sun. Proactive agent: Shifting llm agents from reactive responses to active assistance. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 47431–47457, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/75c37811e830bf029584b1c6fac17726-Paper-Conference.pdf.

Davide Mazzaccara, Alberto Testoni, and Raffaella Bernardi. Learning to ask informative questions: Enhancing llms with preference optimization and expected information gain. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 5064–5074, 2024.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Evan Miller. Adding error bars to evals: A statistical approach to language model evaluations. *arXiv preprint arXiv:2411.00640*, 2024.

Md Mahadi Hasan Nahid and Davood Rafiei. Prism: Agentic retrieval with llms for multi-hop question answering. *arXiv preprint arXiv:2510.14278*, 2025.

Deepak Nathani, Cheng Zhang, Chang Huan, Jiaming Shan, Yinfei Yang, Alkesh Patel, Zhe Gan, William Yang Wang, Michael Saxon, and Xin Eric Wang. Proactive agent research environment: Simulating active users to evaluate proactive assistants. *arXiv preprint arXiv:2604.00842*, 2026.

Justin D Norman, Michael U Rivera, and D Alex Hughes. Reliability without validity: A systematic, large-scale evaluation of llm-as-a-judge models across agreement, consistency, and bias. *arXiv preprint arXiv:2606.19544*, 2026.

Ellie Pavlick and Tom Kwiatkowski. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694, 2019.

Barbara Plank. The problem of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 10671–10682, 2022.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Zhiwei Liu, Jianguo Zhang, Haolin Chen, Heng Ji, Weiran Yao, Shelby Heinecke, Silvio Savarese, Caiming Xiong, and Huan Wang. Userbench: An interactive gym environment for user-centric agents. *arXiv preprint arXiv:2507.22034*, 2025a.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Jielin Qiu, Zhiwei Liu, Haolin Chen, Shirley Kokane, Heng Ji, Weiran Yao, Shelby Heinecke, Silvio Savarese, Caiming Xiong, and Huan Wang. Userrl: Training interactive user-centric agent via reinforcement learning. *arXiv preprint arXiv:2509.19736*, 2025b.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2737–2746, 2018.

Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. Provably robust DPO: Aligning language models with noisy feedback. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42258–42274. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ray-chowdhury24a.html.

Ranjan Sapkota, Konstantinos I. Roumeliotis, and Manoj Karkee. Ai agents vs. agentic ai: A conceptual taxonomy, applications and challenges. *Information Fusion*, 126:103599, 2026. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2025.103599. URL https://www.sciencedirect.com/science/article/pii/S1566253525006712.

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. A long way to go: Investigating length correlations in RLHF. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=G8LaO1P0xv.

Jiashuo Sun, Jimeng Shi, Yixuan Xie, Saizhuo Wang, Jash Rajesh Parekh, Pengcheng Jiang, Zhiyi Shi, Jiajun Fan, Qinglong Zheng, Peiran Li, Shaowen Wang, Ge Liu, and Jiawei Han. Retrieval is cheap, show me the code: Executable multi-hop reasoning for retrieval-augmented generation, 2026a. URL https://arxiv.org/abs/2605.12975.

Weiwei Sun, Xuhui Zhou, Weihua Du, Xingyao Wang, Sean Welleck, Graham Neubig, Maarten Sap, and Yiming Yang. Training proactive and personalized LLM agents. In *Third Conference on Language Modeling*, 2026b. URL https://openreview.net/forum?id=jzp29vOA6c.

Yan Tang, Tingyu Cao, Yuanbo Tang, Huaze Tang, and Keer Hu. Proactive service agents: A unified decision framework, methods, and evaluation. *arXiv preprint arXiv:2609.03727*, 2026a.

Yuanbo Tang, Huaze Tang, Tingyu Cao, Lam Nguyen, Anping Zhang, Xinwen Cao, Chunkang Liu, Wenbo Ding, and Yang Li. Proagentbench: Evaluating llm agents for proactive assistance with real-world data. *arXiv preprint arXiv:2602.04482*, 2026b.

Yiling Tao, Shihan Deng, Meiling Tao, Pengzhi Wei, Zhichao Hu, and Zhihao Zhu. When search agents should ask: Discobench for clarification-aware deep search. *arXiv preprint arXiv:2606.27669*, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475. URL https://aclanthology.org/2022.tacl-1.31/.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037, 2023.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024a.

Yuhao Wang, Ruiyang Ren, Junyi Li, Wayne Xin Zhao, Jing Liu, and Ji-Rong Wen. Rear: A relevance-aware retrieval-augmented framework for open-domain question answering. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 5613–5626, 2024b.

Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. Self-preference bias in llm-as-a-judge. *arXiv preprint arXiv:2410.21819*, 2024.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198, 2020.

Bin Wu, Guanyun Zou, Bingbing Wang, Huan Zhao, and Chuan Shi. Ask now, use later: Benchmarking the proactivity gap in long-lived llm agents. *arXiv preprint arXiv:2605.28108*, 2026a.

Mingqi Wu, Zhihao Zhang, Qiaole Dong, Zhiheng Xi, Jun Zhao, Senjie Jin, Xiaoran Fan, Yuhao Zhou, Huijie Lv, Ming Zhang, Yanwei Fu, Qin Liu, Songyang Zhang, and Qi Zhang. Reasoning or memorization? unreliable results of reinforcement learning due to data contamination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33944–33952, 2026b. doi: 10.1609/aaai.v40i40.40687.

Yifan Wu, Zhuokai Zhao, Songlin Li, Ho Hin Lee, Jiacheng Zhu, Shirley Wu, Tianhe Yu, Serena Li, Lizhu Zhang, Xiangjun Fan, and Shengzhi Li. Swe-together: Evaluating coding agents in interactive user sessions. *arXiv preprint arXiv:2606.29957*, 2026c.

Abel Yagubyan. The coin flip judge? reliability and bias in LLM-as-a-judge evaluation. In *Grounding Language Models: Learning Faithfully and Efficiently @ EMNLP 2026*, 2026. URL https://openreview.net/forum?id=bAABRmjFLs.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Xuan Yang, Hao Xu, Tingfeng Hui, Hongsheng Xin, Kaike Zhang, Chunxiao Liu, and Ning Miao. Beyond ideal instruction: A comprehensive framework for evaluating llms in realistic interactions. *arXiv preprint arXiv:2606.03318*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. {τ}-bench: A benchmark for \underline{T}ool-\underline{A}gent-\underline{U}ser interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=roNSXZpUDN.

Yihang Yao, Zhepeng Cen, Haohong Lin, Shiqi Liu, Zuxin Liu, Jiacheng Zhu, Zhang-Wei Hong, Laixi Shi, and Ding Zhao. Pushing forward pareto frontiers of proactive agents with behavioral agentic optimization. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=pckR7Y6V1j.

Peijie Yu, Wei Liu, Yifan Yang, Jinjian Li, Zelong Zhang, Xiao Feng, and Feng Zhang. Benchmarking LLM tool-use in the wild. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4fb6b4482e4e24a33a3de5b01a011eb0-Abstract-Conference.html.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 57654–57689. Curran Associates, Inc., 2025. doi: 10.52202/085713-1933. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/537d5aa768c2d534016a4d06f87bc8fb-Paper-Conference.pdf.

Haotian Zhai, Jingcheng Liang, and Dongyeop Kang. Abstain-r1: Calibrated abstention and post-refusal clarification via verifiable rl. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 19674–19695, 2026.

Michael Zhang, W Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach llms to ask clarifying questions. In *International Conference on Learning Representations*, volume 2025, pp. 60722–60742, 2025.

Wenlin Zhang, Kuicai Dong, Junyi Li, Yingyi Zhang, Xiaopeng Li, Pengyue Jia, Yi Wen, Derong Xu, Maolin Wang, Yichao Wang, Yong Liu, and Xiangyu Zhao. To search or not to search: Aligning the decision boundary of deep search agents via causal intervention. In *Proceedings of the ACM Web Conference 2026*, pp. 2049–2059. Association for Computing Machinery, 2026a. doi: 10.1145/3774904.3792235.

Yaocheng Zhang, Yuanheng Zhu, Wenyue Chong, Songjun Tu, Qichao Zhang, Jiajun Chai, Xiaohan Wang, Wei Lin, Guojun Yin, and Dongbin Zhao. π-Play: Multi-agent self-play via privileged self-distillation without external data. *arXiv preprint arXiv:2604.14054*, 2026b.

Yuxuan Zhang, Yubo Wang, Yipeng Zhu, Penghui Du, Junwen Miao, Xuan Lu, Zhuofeng Li, Xingwei Qu, Zhengkang Guo, Yuanzhe Shen, Dingjie Song, Han Zhou, Tuney Zheng, Xian Wu, Hao Yu, Songcheng Cai, Yi Lu, Yunzhuo Hao, Minyi Lei, Liang Chen, Kai Zou, Huifeng Yin, Wendong Xu, Dongfu Jiang, Ping Nie, Jiaheng Liu, Wenhu Chen, and Kelsey R. Allen. Clawbench: Can ai agents complete everyday online tasks?, 2026c. URL https://arxiv.org/abs/2604.08523.

Jiale Zhao, Ke Fang, and Lu Cheng. When and what to ask: Askbench and rubric-guided rlvr for llm clarification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17120–17140, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623. Curran Associates, Inc., 2023.

URL https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WZH7099tgfM.

Yikai Zhu, Kunfeng Chen, Qihuang Zhong, Juhua Liu, and Bo Du. Conrag: Consensus-driven multi-view retrieval for multi-hop question answering. *arXiv preprint arXiv:2605.28093*, 2026.

Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre, Kevin Meng, Rebecca Weiss, Fazl Barez, Rahul Gupta, Jwala Dhamala, Jacob Merizian, Mario Giulianelli, Harry Coppock, Cozmin Ududec, Antony Kellermann, Jasjeet Sekhon, Jacob Steinhardt, Sarah Schwettmann, Arvind Narayanan, Matei A Zaharia, Ion Stoica, Percy Liang, and Daniel Kang. Establishing best practices in building rigorous agentic benchmarks. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-5547. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/f316275b44ee2de533102913828a8107-Paper-Datasets_and_Benchmarks_Track.pdf.

Deyu Zou, Yongqiang Chen, Fan Feng, Mufei Li, Pan Li, Yu Gong, and James Cheng. On information self-locking in reinforcement learning for active reasoning of LLM agents. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=oPqm8k1CMQ.

A METRIC DEFINITIONS AND ESTIMATOR

This section defines the need graph and every quantity of Section 2, reports the two-axis readings behind Table 2, and states the populations, eligibility rule and estimator of Section 4.

The need graph. A task's need graph $N(x) = (V, E)$ holds the units of evidence the task requires, its needs V, with a prerequisite edge $(u, v) \in E$ wherever v becomes nameable only once u is resolved. The prerequisites of v are $\mathrm{pa}(v)$, its depth d_v is the longest prerequisite path above it, and the task's facets $\mathcal{F}$ are the weakly connected groups of prerequisite edges left once the surface needs are removed, a facet being one independent line of inquiry, the term Table 1 uses.

How each suite's graph is recovered. No person and no model wrote any node or edge (Table 4). One pattern recovers MuSiQue's edges from its authors' references to an earlier step, such as #1: every one of its 1,464 needs below the surface carries such a reference and none of the 1,196 on the surface does. StrategyQA carries the same references on 2,999 of 6,720 steps and leaves the rest unconnected rather than joining them by position, which would manufacture the structure being measured, so its depth is a lower bound. 2WikiMultiHopQA links a need to another whose subject is the first's object. In a random-deletion test on stored runs (Appendix I), a random edge error could hide the depth result but not manufacture it, and a systematic error is not covered.

Table 4: The three need-graph sets, every depth recomputed from the stored edges.

suite	graphs	nodes	edges	depth histogram	behind an edge
MuSiQue	800	2,660	1,860	0:1,196 1:800 2:530 3:134	55.0%
StrategyQA	2,290	6,720	4,483	0:3,721 1:2,320 2:615 3:60 4:4	44.6%
2WikiMultiHopQA	12,576	31,120	11,956	0:19,164 1:11,956	38.4%

The quantities and the taxonomy. A run holds evidence S_t after its tth question and S at the end. It resolves v at $t_v = \min\{t : v \in S_t\}$, and v becomes reachable at $r_v = \min\{t : \mathrm{pa}(v) \subseteq S_t\}$, both ∞ if never. Table 5 defines what is read off this and Table 6 arranges the content axis as a ladder.

Table 5: Everything this paper measures, read from a finished run against its need graph. Breadth is a count, since tasks differ in how many lines they have, and a line's facet holds only its needs below the surface, so breadth counts the lines in which a run resolves a need beyond the line's first. Depth-weighted recall averages coverage at each depth the task has, C_d, with weights w_d of 0.5, 1.0 and 1.5 at depths 0, 1 and 2 and 2.0 at depths 3 to 5, set in the scorer. A depth beyond five carries no weight, which matters only on the two deeper constructed corpora, whose chains run to six and eight. The stopping pair is read at every decision point, excluding states the budget forced. τ is the tokens spent.

quantity	definition	reads
required-evidence coverage	$\lvert V \cap S\rvert \,/\, \lvert V\rvert$	evidence recovered
breadth	$\lvert\{f \in \mathcal{F} : f \cap S \neq \emptyset\}\rvert$	independent lines touched
depth-weighted recall	$\sum_d w_d\, C_d \,/\, \sum_d w_d$, over depths present	coverage per level, weighted deep
coverage at depth k	$\lvert\{v : d_v = k,\ v \in S\}\rvert \,/\, \lvert\{v : d_v = k\}\rvert$	one level on its own
deepest need resolved	$\max\{d_v : v \in V \cap S\}$	the level reached at all
out-of-order rate	$\lvert\{v \in S : \exists u \in \mathrm{pa}(v),\ t_u > t_v\}\rvert \,/\, \lvert V \cap S\rvert$	taken out of order
promptness	$\lvert\{v \in S : t_v = r_v\}\rvert \,/\, \lvert\{v \in S : r_v < \infty\}\rvert$	taken when first reachable
stop-when-done	$\Pr[\textsc{stop} \mid \lvert V \cap S_t\rvert = \lvert V\rvert]$	halts once the evidence is in
ask-when-not-done	$\Pr[\textsc{ask} \mid \lvert V \cap S_t\rvert < \lvert V\rvert]$	continues while evidence is missing
cost per need found	$\tau \,/\, \lvert V \cap S\rvert$	the price of what it found

Table 6: The content axis of proactivity as a ladder, each level defined on the need graph. Order and stopping complete the axis: a policy that reaches deep needs by luck takes them out of prerequisite order, and knowing that the required evidence is in hand is the content-axis counterpart of staying silent. Levels 1 to 3 sit one above the depth of the needs they pursue, since level 0 is the request.

level	name	the needs it pursues	read by
0	stated	named in the request	coverage of stated needs
1	horizontal	at depth zero, unstated	breadth, not separable
2	vertical, one step	after one prerequisite	depth-weighted recall
3	vertical, deep	behind two or more prerequisites	deepest need resolved
4	open	in no annotated graph	not scored by a graph
	order	taken before its prerequisite	out-of-order rate (lower better)
	stopping	none, the evidence is in hand	stop-when-done, ask-when-not-done

Coverage at depth two or more weights each depth by the needs it holds, so a shallow frontier of two needs cannot outvote a deep one of forty. Absent is not zero: a task with nothing at depth two is left out of the depth-two criterion, and a task with no graph or a suite with no user simulator is omitted rather than filled in, because a fabricated zero cannot be told from a measurement.

Order. Lower is better on the out-of-order rate: where a step gates an action rather than a citation, acting out of order is acting on an unconfirmed premise, although no downstream failure is measured here. Neither it nor promptness separates real vertical proactivity from luck on the evidence available. Promptness is defined only where each arm's own trajectory makes it so, which selects its paired population on both arms' behavior, and the out-of-order rate points the wrong way: on MuSiQue it excludes zero in the adverse direction on the same runs whose depth-weighted recall rises, a separate measurement and not an offset (Table 2, and Appendix E for where the excess comes from).

Why no benchmark suite separates the horizontal axis. MuSiQue gives every one of its 800 tasks exactly one facet and StrategyQA none of its 2,290 tasks two or more, so on the two suites this work trains and selects on, breadth and every variant of it have one bit of range or none. Only 2WikiMultiHopQA branches, on 21.2% of its 12,576 tasks, and there no annotated need sits at the second level. Nor could a separating suite be assembled from these annotations: no prerequisite edge crosses a facet boundary in 19,532 edges, and among the 2,735 tasks with two or more facets none has a facet made entirely of dispensable needs. So on these suites breadth is a measurement and never an independent axis, and the vertical claim has two controls that could falsify it (Appendix F) where breadth has none.

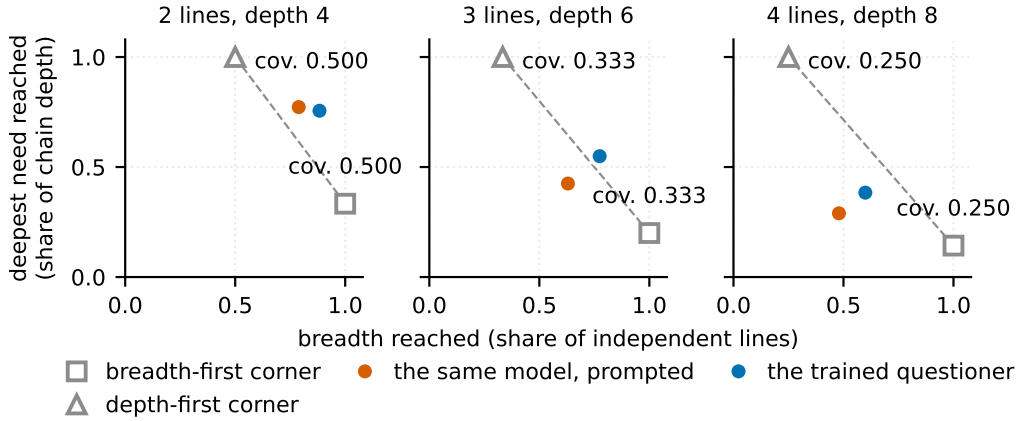


Figure 4: The two axes are separable, and the trained questioner reaches further along both than the same model, prompted, on the two longer chains. Each panel is one constructed corpus, both axes normalized to what it makes available. A breadth-first corner (square) and a depth-first corner (triangle) reach identical coverage, labeled at each. Positions are per-arm levels over 400 runs each and are not matched on spend: the corners ask exactly 4, 6 and 8 questions, the prompted arm 9.6, 8.8 and 8.5 and the trained questioner 5.3, 7.4 and 8.4, so a position relative to the line between the corners reflects spend as well as policy. The trained questioner asks more than the corners at every shape, yet it sits above the line at the two shorter chains and below it at the longest, on 8.4 questions against 8, so the line is not an equal-spend frontier and neither side of it ranks an arm against the corners.

Constructed corpora, where both axes are live. Facets exclude the surface, so breadth already requires depth and the two could have been nested rather than orthogonal. In each of three constructed shapes the chains run twice as deep as the number of lines, so that two hand-written corner policies pay equal cost: a breadth-first one takes one hop into every line and a depth-first one follows one chain. They reach identical coverage within each shape and opposite extremes of breadth and depth (Figure 4), so coverage cannot separate the axes and breadth and depth can. Being hand-written, they say nothing about whether a trained policy trades one axis for the other.

The trained questioner sits in the interior of both axes at every shape, each shape its own corpus and never pooled. At equal spend against the same model, prompted, it gains breadth at the two shorter chains, by $+0.215$ $[+0.138, +0.296]$ and $+0.151$ $[+0.046, +0.264]$, ties at the longest, gains depth-weighted recall on all three, and resolves deeper needs at the middle shape by $+0.354$ $[+0.222, +0.500]$. Each training seed alone decides the same cells. The out-of-order rate takes one value on all four arms, both corners included, so these corpora cannot express it.

On natural questions composed in pairs, both axes are live and the gain carries over. Two held-out MuSiQue questions that share no template, title or answer, neither in the training set and at least one with a chain of depth two or more, are joined into one task that states both, their evidence pools and need graphs unioned with no edge between them, so each task has exactly two independent lines of inquiry. Both orders of each pair are separate tasks, giving 48 pairs and 96 tasks, with a ceiling of 16 questions. The metrics, the prediction and the rule for reporting a negative result were declared before the tasks were built. At equal realized spend against the same model, prompted, with the bootstrap clustered on the pair, the trained questioner advances more of the two lines, by $+0.276$ $[+0.172, +0.396]$, covers more of the required evidence, by $+0.152$ $[+0.112, +0.197]$, and weights more of it toward deep needs, by $+0.159$ $[+0.110, +0.217]$, all three decided. The prediction pools each pair's two questions' own held-out readings, weighted by their need counts, and no departure from it is detected: the composed gain minus the predicted one is $+0.019$ $[-0.025, +0.064]$ for coverage and $+0.018$ $[-0.037, +0.077]$ for depth-weighted recall, over the 43 pairs whose two questions both have held-out runs. At each policy's own stop the composed gain exceeds the prediction, by $+0.111$ $[+0.059, +0.166]$ in coverage, a reading about stopping rather than equal spend. The pairing is ours, so these are composed natural tasks and not a natural benchmark, and answer correctness is not read, because the composed answer has no aliases. A comparison against GPT-

OSS-120B, prompted, on the same tasks was declared after the result above had been read and before its own first unit, so it is exploratory. At equal spend the trained questioner covers more of the required evidence than GPT-OSS-120B, by $+0.083$ $[+0.044, +0.124]$, and weights more of it toward deep needs, by $+0.082$ $[+0.028, +0.137]$. Both are decided. Its lead in breadth, $+0.073$ $[-0.034, +0.182]$, is not.

Stopping, read per decision point. An episode of n questions offers $n+1$ decisions, and a run-level summary keeps only the last, so a run that asked six questions after it was done and then stopped would score a perfect one. A point is done when the coverage the policy held then was complete, and one whose coverage cannot be read is skipped. Every point before the last is an ask, and the last is a stop only when the policy chose to stop. A state the harness halted at its ceiling or turn limit was never offered to the policy, so it carries no decision and is counted apart as a forced stop. Each rate is reported with its denominator, a property of the policy, beside the questions bought at points already done.

Quantities that are structurally determined, and quantities deleted. A task's total facet count, fixed by its graph, and stop undershoot, zero by algebra, cannot take another value and so are never cited as nulls. Five candidate metrics were deleted: anticipation depth (non-monotone), proactive precision as first stated (circular), the probability of asking at least one user-directed question (a property of the ordering), stop regret in quality units (it charges a perfect stopper) and future conversation coverage (it scores a miss on exactly the success case).

Suites, splits and eligibility. FRAMES, 824 tasks, ships no need graph, so coverage, depth and breadth are undefined there rather than zero. The tool-use domains are never mined, and runs forked from a dialogue prefix another system generated are flagged exploratory (Appendix H). The partition is a property of the task identifier: a bucket hashed from the suite and the template identifier where one exists, else the task identifier, with buckets 0 to 59 training, 60 to 74 development and 75 to 99 held out, so two instantiations of one template never straddle the wall. A row reaches a question-answering table only if its split is held out, it is not flagged exploratory, and the run reconciles on tokens and on documents. Each suite carries 200 distinct eligible held-out tasks, out of held-out corpora of 214, 581 and 3,080. MuSiQue's held-out partition counts 212 tasks when its adapter is constructed directly and 214 over run directories. All 200 reported MuSiQue tasks are held out under the split function itself, so the difference cannot move a task into the reported population.

Arms and the estimator. Every arm shares a ceiling of eight questions and a limit of sixteen turns, and the number of passages and a hash of the pool are inside a run's identity, so a different index is a different run rather than a confound. Every difference is paired on suite, task and seed, and seeds and any other replicate of a task, such as a prompt variant, are averaged into the task before resampling, so a count of pairs is never an n. Intervals are task-clustered bias-corrected and accelerated bootstraps printed at ten thousand resamples, and a cell is decided only when its intervals at fifty thousand resamples under three bootstrap seeds all exclude zero (Section 4). No endpoint of the preregistered stage names a trained arm, so every trained-arm contrast on the question-answering suites is exploratory and is read from its estimate and interval. The prompt search's probability values are the measurement rather than the evidence, since a searched prompt's apparent gain on the slice that selected it did not hold on fresh tasks (Appendix K). Every number belongs to one scorer version, which composes the metric register, graph set, matcher and judge pins, and comparisons stay within one.

What is new, and what is borrowed. The estimator families are ordinary: coverage is set recall, depth-weighted recall a discounted-gain recall (Järvelin & Kekäläinen, 2002) weighted by depth instead of rank, breadth an aspect count (Clarke et al., 2008) read from the graph, question diversity distinct-n (Li et al., 2016), cost per need a cost-normalized effectiveness (Erol et al., 2026), and the stopping pair's nearest relative abstention (Kirichenko et al., 2025; Zhai et al., 2026). What we claim is the substrate, a required-evidence graph whose edges are prerequisites, which makes d_v and r_v definable, and three readings on it: the out-of-order rate and promptness, which we have not found elsewhere as transcript-level readings, and the stopping pair read at every decision point, a change of unit rather than of formula. The paper as a whole combines this content axis with candidates sampled at one state and ordered by consequence, comparison at equal retrieval spend (Appendix C), and zero-shot transfer to a tool-use benchmark with a simulated customer (Yao et al., 2025; Barres et al., 2026). Two of these parts are stated more specifically: one model running the protocol is equivalent to the two-model split at the shared cap and separated from it at equal

spend (Appendix F), and on the tool-use benchmark the trained questioner raises task success, with follow-up turns reported in Appendix H.

Evaluation principles. A move is credited by its consequence against the graph, never by how it reads, the role grounding plays on the initiative axis (Bui & Evangelopoulos, 2026), and the two axes are never pooled. When only a person can resolve a need, the joint claim is two numbers, user turns and success, never one ratio (Yao et al., 2026). Interruption cost and adaptation after feedback stay on the initiative axis (Bui & Evangelopoulos, 2026), since this questioner interrupts no one.

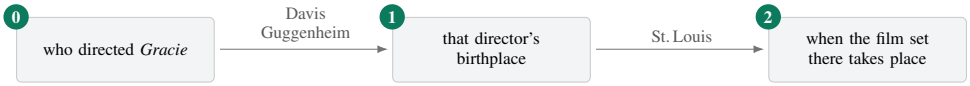
B ONE TASK, THREE POLICIES

This section supports Section 2, and it illustrates the axis rather than measuring it: the aggregate reading is Table 2.

Figure 5 shows one held-out task in full, with the questions each policy asked in the order it asked them, and the required need each question resolved. The task's third need is the clearest case of the vertical axis this paper can show: the film the question asks about is named in it only by the start of its title, and where it is set cannot be known until the director and then that director's birthplace are. The no-question baseline reports that the evidence does not settle the question, which is true of the evidence it gathered. The same model, prompted, resolves the first two needs, then asks for the birthplace it has already been given four more times, and stops one need short after six questions. Both training seeds of the trained questioner reach the third need first, on their third question, by searching for a work whose title begins with the words the task supplies, before either prerequisite is in hand, one of them while naming the wrong director. The second seed then resolves both prerequisites and stops after five questions with every required need resolved, and the first resolves one of them and is stopped by the ceiling of eight. That is out of order, but of the kind a question aimed at the need itself produces. Most of the excess out-of-order rate Table 2 reports on this suite is the other kind, evidence a question aimed elsewhere brought back (Appendix E), so the example is not its typical source. It is an evidence outcome, not an answer outcome: the strict answer match is zero for all four runs, so the figure reports resolved needs and not a correct answer.

(a) What the task requires

"When does Meet Me in the birthplace of Gracie's director take place?" The film is named only by the start of its title, and where it is set cannot be known until the first two needs are resolved.



badge: the depth of the need

(b) What each policy asked, and what it answered

No questions asked — 0 of 3 needs — needs unresolved

− *asks nothing*

Answer. The evidence does not specify when "Meet Me in the birthplace of Gracie's director" takes place. The date is unsupported by the provided evidence.

The same model, prompted — 2 of 3 needs — needs unresolved

0 "Who is Gracie's director?"

1 "Where was Davis Guggenheim's birthplace located?"

− "What is the birthplace of Davis Guggenheim?" (asked 4 times)

Answer. The time cannot be determined from the provided evidence; the evidence lacks information about when "Meet Me" takes place.

The trained questioner, training seed 1 — 2 of 3 needs — needs unresolved

− "Who is the director of the film or work titled 'Gracie' and what is the director's birthplace?"

− "What is the birthplace (city or location) of the director of the work titled 'Gracie'?"

2 "What is the setting (city, town, or location) where the film or work titled 'Meet Me in' is set, and in what time period or year does it take place?"

− "Which paragraph mentions the director of the work titled 'Gracie' and provides the director's birthplace (city or location)?"

1 "Which paragraph mentions the director of the work titled 'Gracie' and also states the city or location where that director was born?"

− *three further rewordings of the question above, none resolving a need*

stopped by the budget of eight

Answer. Summer 1903 through spring 1904.

The trained questioner, training seed 2 — 3 of 3 needs — every need resolved

− "Who is the director of the film or work titled 'Gracie' and what is the birthplace of that director?"

− "What is the birthplace (city and country) of film director Nelson Cragg, the director of photography for the series Homeland?"

2 "What is the birthplace (city and country) of film director Nelson Cragg, the director of photography for the series Homeland, and does any paragraph describe a work titled 'Meet Me in' that takes place in that city?"

0 "Which paragraph describes a film, book, or other work titled 'Gracie' and names its director?"

1 "What is the birthplace (city and country) of film director Davis Guggenheim, the director of the film Gracie?"

Answer. Summer 1903 – spring 1904.

Figure 5: One held-out MuSiQue task, and what each policy asked. A filled badge on a question gives the depth of the required need that question resolved, and a hollow badge marks a question that resolved none. **This is an illustration and not evidence.** The trained questioner is shown at both of its training seeds, and the task was chosen by a rule fixed before any of its transcripts was read: among the 175 held-out tasks of this suite where every policy has a completed run at the same rollout seed, take those where each training seed resolves a need at depth two or more that the same model, prompted, does not, and select the first by task identifier. Sixteen tasks satisfy that rule and three satisfy it with the two policies exchanged, so the direction shown here is the more common one and not the only one. The chain's labels are glossed from the benchmark's own decomposition forms, which the released record carries verbatim.

C COMPARISON AT EQUAL RETRIEVAL SPEND

This appendix supports the equal-spend rule of Table 2, the efficiency reading of Figure 6, the claim of Section 5 that the gain is not bought with longer questions, and the structured-retrieval comparison of Figure 2b.

A prefix of a long run is a short run. The policy that chooses questions is a function of the dialogue state alone, which holds no ledger of spend, no remaining allowance and no cap: the usage stamped on past turns says what has been spent, not where the ceiling is, and truncation is applied from outside and never announced. The question it asks fifth is therefore the one it would have asked had its cap been five, so coverage after k questions of a long run is the terminal coverage of a true k-question run. The equal-spend rule rests on this: the prefix read from the arm that asked more is a run it would in fact have produced. A budget-aware policy would lose the property twice over, spending its last question differently for knowing it was the last and stopping as an announced cap approached, so the stopping measurement would report the experiment's own parameter back to it.

The assumption, tested. To build the policy the argument rules out, GPT-OSS-120B, prompted, was given one added prompt block stating in words how many questions it may ask, and run against its unmodified self at a retrieval cap of four on all three suites, 200 tasks at two seeds each. Its required-evidence coverage rises on none of them, with paired differences of -0.0244 $[-0.0496, +0.0004]$ on MuSiQue, $+0.0033$ $[-0.0162, +0.0231]$ on StrategyQA and -0.0056 $[-0.0200, +0.0075]$ on 2WikiMultiHopQA, and the informed comparator asks slightly fewer questions rather than better early ones, 3.07 against 3.24, 3.28 against 3.41 and 2.38 against 2.42. This exploratory control, not a sealed endpoint, tests whether a comparator told its allowance asks better early questions, not whether a truncated trajectory equals a true short run, since both arms here were given a true cap.

What is read at the shared cap alone. A shared cap charges the arm that stops earlier for the calls it declined and credits nothing for the calls it saved, so a reading at the shared cap answers a different question, which policy does better when each decides when to stop (Table 2), and is not a second reading of evidence per question. Stop overshoot has no equal-spend form, because the rule sets the comparator's stopping index to the trained arm's, the quantity the metric measures. Nor does answer quality, since a prefix-length answer would need a drafting call that was not taken. The selection gate records coverage at depth two or more at the shared cap only, since rebuilding it at a prefix needs the gold map from evidence to depth, which the gate may not read.

The currency is retrieval calls, and one question is one call. On these suites question count and retrieval calls are the same quantity on every run, so equal questions is equal retrieval spend by identity and not by approximation. The identity does not travel: in the tool-use benchmark of Appendix H a question costs a variable number of tool calls, so there questions and tool calls are two currencies, each enforced separately. A third currency, unique documents reaching the drafter, is neither matched nor reported, and its unmatched level points the other way: on held-out MuSiQue the trained questioner asks 1.41 times fewer questions than the same model, prompted, but sees 2.6% more documents, since the comparator's extra questions retrieve heavily overlapping documents.

A reading that needs no matching rule. Tokens spent per required need the run actually found put the cost in the denominator, so no prefix is read, no pairing convention enters, and the currency means the same for a locally served policy and a hosted one. On paired held-out tasks the trained questioner's reductions are $-7{,}744$ $[-9{,}786, -6{,}222]$, $-6{,}046$ $[-6{,}992, -5{,}135]$ and $-1{,}663$ $[-2{,}386, -1{,}065]$ on MuSiQue, StrategyQA and 2WikiMultiHopQA, from $19{,}354$ tokens per need found to $11{,}610$ on MuSiQue (Figure 6).

Question length, and the budget that controls for it. A longer question costs the same single retrieval call, so no call-denominated comparison can test whether the gain was bought with words rather than with better questions. On development (132 MuSiQue and 333 StrategyQA tasks) the trained questioner's questions average 17.5 and 22.6 words against 13.1 and 13.0 for the same model, prompted, and 21.7 and 29.2 question tokens against 16.0 and 15.6. It fails the one-sided not-longer rule, whose declared margin is 0.1, on both suites and in both units, with pooled relative lower bounds of $+0.304$ and $+0.320$ on MuSiQue and $+0.705$ and $+0.829$ on StrategyQA, in words and in tokens, and each seed fails alone.

The control therefore charges the comparator the trained questioner's own question budget in question tokens, the text the drafter receives as the model counts it, with question words as the check.

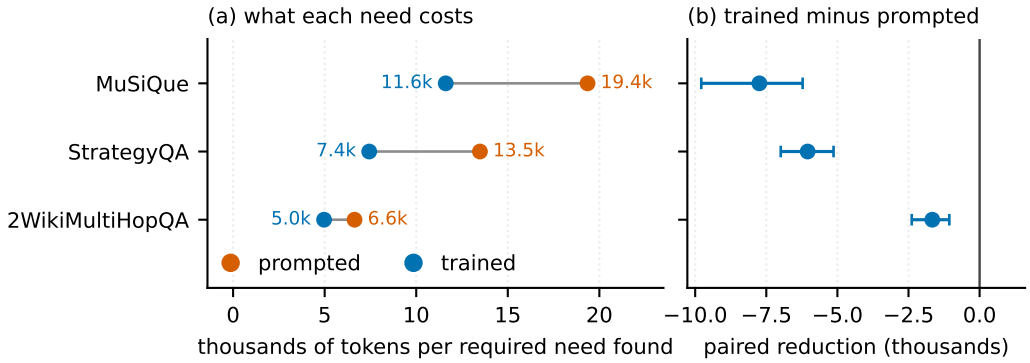


Figure 6: Efficiency with no matching rule. (a) Tokens spent per required need found, lower being better, as paired means over the tasks both arms completed. (b) The paired reductions, every interval excluding zero. The trained questioner's two training seeds are averaged within each task, against the same model, prompted.

The comparator is granted complete questions in the order it asked them until the next would overrun the allowance, so its realized spend sits below the allowance. The equal-call reading it is compared with is the selection gate's form of the rule: the trained questioner at its natural stop, the comparator at its own prefix of the same length or its own end, whichever comes first. That matches spend in one direction only: where the comparator stops first, the trained questioner keeps its extra calls. This appendix calls it gate-matched spend and keeps equal spend for both arms read at the lower count.

Table 7: Coverage when the comparator is charged the questioner's own question budget, on development. The trained questioner averages seeds 1 and 2 within each task. Intervals are the task-clustered bootstrap at 1,000 resamples.

suite	unit	allowance granted	comparator spend	comparator questions	Δ coverage under budget	95% interval
MuSiQue	question tokens	91.8	65.5	4.261	+0.1487	[+0.1130, +0.1949]
MuSiQue	question words	74.3	53.3	4.250	+0.1534	[+0.1174, +0.2019]
StrategyQA	question tokens	62.2	42.0	2.974	+0.0466	[+0.0255, +0.0692]
StrategyQA	question words	48.2	33.2	2.881	+0.0515	[+0.0295, +0.0766]

Charged that budget, the comparator does not catch up, and all four intervals exclude zero. At equal retrieval calls it was charged 3.981 and 2.080 questions and trailed by +0.1556 and +0.0799 on MuSiQue and StrategyQA. On StrategyQA the budget buys it 43% more asking and the difference falls to +0.0466, and on MuSiQue it moves to +0.1487. The allowance exceeds everything the comparator asks within eight calls on 41% to 45% of MuSiQue tasks, so the comparator was free to spend the verbal budget on more questions and could not convert it into evidence: the gain is in what was asked, not in how much was said. The instrument is not generous to every arm. The arm trained on stop contrasts alone, which asks the most questions of any trained arm, fails both tests on the same tasks: its differences at matched retrieval calls, −0.0051 and −0.0169, and all four under its own question budget have intervals containing zero.

A structured retrieval method on three models (Figure 2b). PAR^2-RAG (Li et al., 2026), reimplemented without training in the same harness (Section 6), never beats plain prompting of its own model at equal retrieval spend, on Qwen3-8B, GPT-OSS-120B or Claude Opus 5. On the larger two it is lower, by −0.037 [−0.070, −0.005] on MuSiQue for GPT-OSS-120B and by −0.074 and −0.017 on MuSiQue and StrategyQA for Claude Opus 5. The trained questioner leads the reimplementation by +0.100 and +0.060 on Qwen3-8B, its own base weights, by +0.124 and +0.055 on GPT-OSS-120B, and on Claude Opus 5 by +0.046 on MuSiQue. The one comparator ahead of it is Claude Opus 5 prompted plainly, on MuSiQue, where the trained questioner minus it is −0.029 [−0.055, −0.003], and on MuSiQue Claude Opus 5 asks fewer questions, 3.5 against the trained questioner's 4.2.

On FRAMES. FRAMES has no need graph, so the third panel reads answer correctness at each policy's own stop under a ceiling of eight calls, over 824 tasks, and is not a comparison at equal spend. No difference between the trained questioner and a comparator prompted plainly is decided, from -0.010 $[-0.039, +0.018]$ against GPT-OSS-120B to $+0.014$ $[-0.016, +0.044]$ against the same model, and none against the reimplementation on Qwen3-8B or GPT-OSS-120B. Claude Opus 5 running PAR^2-RAG is ahead: the trained questioner minus it is -0.061 $[-0.093, -0.032]$, decided, with 0.237 of tasks answered correctly against 0.299, and on FRAMES the trained questioner uses 3.6 retrieval calls against its 4.5, so Claude Opus 5 running PAR^2-RAG spends more to lead.

D CORPUS, EXPORT RULES AND TRAINING RECIPES

This section gives the corpus, export rules and recipes behind the trained questioner of Section 3 and the supervision comparison of Table 3, and reads every arm on the held-out split.

What the corpus holds. A decision is a state, rendered exactly as the policy sees it at inference, an action taken there, and a label derived from what that action caused. At a chosen state of a recorded run we sample eight alternative questions at temperature one, with no nucleus or top-k truncation, and roll each to completion. The eight share the state byte for byte, so whatever value it carries cancels within a pair and the preference needs neither a baseline nor a critic. Each is labeled with whether its episode answered the task, how many turns it still needed to reach complete evidence, and whether it resolved a need as soon as that need became askable. The widest export, with no filter on which arm produced a run, holds 44,624 imitation rows and 50,730 preference pairs over 32,958 states and 1,918 tasks. Of its pairs, 16,089 set one question against another, 9,668 of them at a state where evidence has already arrived. The rest set a question against stopping, 1,958 where a candidate actually stopped and 32,683 derived from the need graph at complete states where no candidate stopped, kept separately selectable because a gold-derived stop is not the same evidence as one a policy took.

The export rules. The export reads the need graphs to value every decision point, so it runs on the gold side and the trainer never does. It re-renders the prompt the policy saw from the recorded artifacts and refuses a run whose evidence hash does not re-derive or whose draft is not recoverable. Every prompt that shows evidence, in every arm and in the training data alike, cuts each retrieved paragraph at its first 1,200 characters, which 3.5%, 6.1% and 5.2% of the paragraphs in the three corpora exceed. The acceptance floor is specified as a percentile of pilot values plus a term in the judge's deviation, but the export ran at its command-line defaults, a floor of 0.05 and a judge deviation of zero, so the rule that ran was a flat threshold at 0.05 on the consequence score.

The imitation signal, and the refusal. The imitation target is a stop where the required evidence was already in hand before the decision, and otherwise the best sampled question if it clears the floor. Where nothing clears the floor, no row is emitted. That refusal discarded 5,030 unfinished states from the fixed-cohort export the imitation reference trained on and 24,538 from the pooled export, which keeps only runs of the same model, prompted, and applies no cohort filter. The pooled export is not the widest export above, which admits every arm.

The preference signal. The pairs the trained questioner's preference stage trains on are of three kinds. A question pair sets two sampled questions against each other, and a recorded stop contrast sets a question against a stop the policy actually took. A synthesized stop contrast, emitted at a state the graph marks unfinished where some candidate asked and none stopped, chooses the best available question over stopping, which restores as contrasts the states the imitation refusal drops. It is never emitted at a state of unknown completion, and a state where a candidate actually stopped keeps its recorded pair instead. Sampled questions are ordered by outcome, first by whether the episode answered the task (Section 3). A key ranking anticipation first is only a flag, because under it the finding that anticipation emerges from ranking on outcome would hold by construction.

How the arms were compared on development. Each arm of Table 8 is read as a paired required-evidence coverage difference against the same model, prompted, at gate-matched spend, at its own stopping point against the comparator's prefix of that length (Appendix C). Intervals are task-clustered bias-corrected bootstraps at 1,000 resamples over 132 MuSiQue and 333 StrategyQA development tasks, so this is a selection record rather than a confirmatory result. Of the arms whose development cells are printed (Tables 3 and 16), every one but the arm trained on stop contrasts

Table 8: The arms. The reference is an imitation adapter fitted to the 14,302 recorded decisions of the same model, prompted, in the fixed-cohort export, two epochs at learning rate 10^{-4}, every other arm varies one thing against it, and the trained questioner is the last row. Its combined set holds 11,305 question pairs, 1,249 recorded stop contrasts and 18,919 synthesized ones, 31,473 of the 36,349 pairs in its preference file, the 4,876 decided by a finished state being left out of training. By source they are 19,448 MuSiQue, 9,786 StrategyQA and 2,232 2WikiMultiHopQA pairs and 7 synthesized stop contrasts from a small synthetic fixture. The stop-contrast-only set is those 18,919 synthesized contrasts alone, and the question-pair arm's 17,430-pair export is the fixed-cohort preference export. Every adapter uses rank 32, scaling 64, dropout 0.05 and no bias term, over the same seven projection modules. The preference arms train for one epoch at learning rate 5×10^{-6} with a divergence weight of 0.1.

arm	fitted from	training records	what it varies
imitation reference	base model	14,302 decisions	nothing (the comparison point)
stop-weighted imitation	base model	14,302 decisions	stop rows down-weighted to 0.1
question pairs	imitation reference	question pairs of a 17,430-pair export	preference on question pairs
stop contrasts only	imitation reference	18,919 pairs, one kind	preference on stop contrasts only
both kinds, from reference	imitation reference	31,473 pairs, three kinds	preference on both kinds at once
both kinds, from pairs	question pairs	31,473 pairs, three kinds	the same set, from a trained start

alone gains coverage on both suites with an interval excluding zero. What each signal moves, in coverage, persistence and stopping when done, is read in Appendix G.

Ordering rule and selection. Three arms that vary only the rule ordering the sampled questions all gain coverage, and against one another only the consequence ordering separates from an ordering by whether a question reaches an unstated need, on StrategyQA, at $+0.0139$ $[+0.0025, +0.0293]$, a separation that does not survive a correction for all six contrasts, so which ordering to prefer is exploratory. Of the twenty-three development checkpoints, seventeen gain coverage at gate-matched spend on both suites with an interval excluding zero, and none passes the pre-committed selection rule as written. The checkpoint carried forward was chosen on another pre-committed ground, that an arm keeps both stopping cells and gains coverage, read on the first registered training run of the recipe (Appendix L). The trained questioner's two training seeds gain coverage at gate-matched spend on both suites, each interval excluding zero, but do not keep the stopping cell (Appendix G).

Question length. The trained questioner fails the gate's length test on both suites (Table 15), and Appendix C reads coverage with the comparator charged the trained questioner's own question budget (Section 5).

Table 9: The same arms on the held-out split, each against the same model, prompted, at gate-matched spend (Appendix C), 200 tasks per suite under one contrast function. The last row, both pair kinds fitted to the question-pair arm, has the largest point estimate on every suite, about a hundredth above the question-pair arm it starts from, a point ordering and not a tested difference. The two question-pair rows start from different imitation checkpoints, the first from the reference row above and the second from the one fitted to the pooled export, the question-pair row of Table 3. Intervals are bias-corrected bootstraps at 10,000 resamples, with any bound within 0.01 of zero re-read at 50,000 across three reseeds.

supervision	MuSiQue	StrategyQA	2WikiMultiHopQA
imitation reference filtered demonstrations	$+0.0858$ $[+0.0492, +0.1231]$	$+0.0271$ $[-0.0070, +0.0631]$	$+0.0587$ $[+0.0299, +0.0938]$
stop-weighted imitation the same, stop rows down-weighted	$+0.0746$ $[+0.0410, +0.1100]$	$+0.0260$ $[-0.0106, +0.0638]$	$+0.0594$ $[+0.0325, +0.0925]$
question pairs question pairs, from the reference	$+0.1177$ $[+0.0848, +0.1531]$	$+0.0693$ $[+0.0337, +0.1067]$	$+0.0719$ $[+0.0456, +0.1025]$
question pairs, pooled question pairs, from the pooled imitation checkpoint	$+0.1110$ $[+0.0797, +0.1448]$	$+0.0657$ $[+0.0319, +0.1008]$	$+0.0413$ $[+0.0125, +0.0725]$
stop contrasts only‡ stop contrasts, from the reference	$+0.0177$ $[-0.0142, +0.0496]$	-0.0623 $[-0.0961, -0.0304]$	$+0.0175$ $[-0.0075, +0.0469]$
both kinds, from reference both pair kinds, from the reference	$+0.0800$ $[+0.0446, +0.1165]$	$+0.0127$ $[-0.0237, +0.0503]$	$+0.0625$ $[+0.0344, +0.0975]$
both kinds, from pairs† both pair kinds, from the question-pair arm	$\mathbf{+0.1284}$ $[+0.1021, +0.1593]$	$\mathbf{+0.0799}$ $[+0.0511, +0.1116]$	$\mathbf{+0.0791}$ $[+0.0519, +0.1113]$

† the trained questioner, its two training seeds averaged within each task. ‡ this arm asks more questions than its comparator, so the rule every other row uses is not a matched reading for it, and its cells are the symmetric reading with both sides held to the lower question count. **Bold** marks the largest value in a column.

E HELD-OUT RESULTS IN FULL

This section carries the held-out cells behind the coverage, depth and ordering claims of Table 2. No declared endpoint of the preregistered stage names the trained questioner as a treatment, so as contrasts these readings are exploratory as a class, reported as point estimates and intervals with no probability value.

The differences behind Table 2. Table 2 prints levels, and Table 10 prints each of its pairs as the trained questioner's difference from the comparator, computed before rounding, with its interval.

Each seed alone. Table 11 prints every Table 2 cell for each of the two training seeds (Section 4) and for the two averaged within the task, against one set of comparator runs under one rule. Each seed alone decides every coverage cell, and the two agree in sign on every cell but the out-of-order rate on StrategyQA, where both span zero.

Development against held out, and the shared cap. In Table 12 the trained questioner is above the same model, prompted, at gate-matched spend on all three suites and at the shared cap of eight on MuSiQue, within two points of it on the other two. On held-out runs its mean question count is 4.20, 3.05 and 2.32, the comparator is charged 3.93, 2.87 and 2.11 against its own 5.90, 5.96 and 3.04, and the comparator's episode was shorter than the charged prefix on 108, 69 and 100 of 800 run pairs, where it is charged its own length.

A diversity gate fails under pooling and passes per seed. The declared diversity statistic, gated at 0.65, pools every rollout seed of a task before counting, so a questioner that repeats its own questions across rollout seeds reads as collapsed, at 0.458 to 0.600 held out against 0.827 to 0.888 on development, which scores one rollout seed. Grouped by task and rollout seed, as Table 13 prints it, the same runs read 0.805 to 0.849: pooling counts repetition across rollout seeds, not a loss of variety within one.

Table 10: Table 2 as differences: trained minus the named comparator over 200 held-out tasks per suite (178, 83 and 113 on the out-of-order row, where it is defined), with bias-corrected intervals over tasks at 10,000 resamples. ° marks a cell not decided (Section 4) and † a row where lower is better. Differences are in percentage points, except breadth (in lines) and the deepest need resolved (in levels).

	MuSiQue	StrategyQA	2WikiMultiHopQA
Equal spend: both arms at the lower of their two question counts			
required-evidence coverage	$+11.2$ $[+8.7, +14.3]$	$+7.0$ $[+4.0, +10.3]$	$+5.0$ $[+1.7, +8.4]$
breadth	$+0.120$ $[+0.084, +0.165]$	$+0.050$ $[+0.008, +0.098]$	$+0.070$ $[+0.015, +0.128]$
depth-weighted recall	$+12.5$ $[+9.5, +16.1]$	$+5.4$ $[+2.6, +8.4]$	$+5.5$ $[+2.0, +9.3]$
deepest need resolved	$+0.218$ $[+0.151, +0.306]$	$+0.098$ $[+0.035, +0.176]$	$+0.056$ $[+0.001, +0.118]^\circ$
out-of-order rate†	$+5.7$ $[+0.7, +10.6]$	$+0.5$ $[-4.9, +5.3]^\circ$	-1.2 $[-5.3, +1.9]^\circ$
coverage vs GPT-OSS-120B	$+7.0$ $[+4.4, +9.6]$	$+4.3$ $[+1.5, +7.4]$	-3.5 $[-7.1, -0.3]$
Own stop: each arm stops itself, with at most eight calls			
required-evidence coverage	$+9.6$ $[+7.1, +12.5]$	-1.2 $[-3.7, +1.4]^\circ$	$+1.5$ $[-0.6, +4.0]^\circ$
coverage vs GPT-OSS-120B	$+10.1$ $[+7.5, +13.0]$	-3.6 $[-5.8, -1.5]$	-0.7 $[-2.6, +0.8]^\circ$
answer token F1	$+2.4$ $[-0.6, +5.8]^\circ$	-0.3 $[-4.3, +3.7]^\circ$	$+3.1$ $[-0.5, +6.9]^\circ$

Table 11: Table 2's cells per training seed, trained minus the same model, prompted, at equal spend, $n = 200$ tasks per suite on every row but the out-of-order one. The last column averages the two seeds within each task, and its intervals and verdicts are in Table 2. The out-of-order rate is read only on the pairs where both arms define it, so each seed's cell is over its own tasks and the averaged cell over every task where either seed defines it, on MuSiQue 175 and 174 tasks against 178. The averaged cell is therefore not the midpoint of the two seeds' cells. † Lower is better.

	suite	seed 1	seed 2	seeds 1+2
required-evidence coverage	MuSiQue	$+0.108$	$+0.116$	$+0.112$
	StrategyQA	$+0.064$	$+0.076$	$+0.070$
	2WikiMultiHopQA	$+0.048$	$+0.051$	$+0.050$
breadth	MuSiQue	$+0.120$	$+0.120$	$+0.120$
	StrategyQA	$+0.043$	$+0.058$	$+0.050$
	2WikiMultiHopQA	$+0.065$	$+0.075$	$+0.070$
depth-weighted recall	MuSiQue	$+0.122$	$+0.129$	$+0.125$
	StrategyQA	$+0.048$	$+0.059$	$+0.054$
	2WikiMultiHopQA	$+0.052$	$+0.058$	$+0.055$
deepest need resolved	MuSiQue	$+0.218$	$+0.218$	$+0.218$
	StrategyQA	$+0.090$	$+0.105$	$+0.098$
	2WikiMultiHopQA	$+0.055$	$+0.058$	$+0.056$
out-of-order rate†	MuSiQue	$+0.051$	$+0.058$	$+0.057$
	StrategyQA	-0.011	$+0.027$	$+0.005$
	2WikiMultiHopQA	-0.013	-0.011	-0.012

Where the out-of-order excess comes from. The out-of-order rate counts an edge as violated when the child's evidence arrives before the parent's, whatever brought it, whether the questioner asked for the child, skipping the parent, or a question aimed elsewhere, often at the parent, retrieved the child's paragraph as well. Each violation event on MuSiQue is classified by whether the question at the turn that surfaced the child's evidence names the child under the matcher's own ask test. Against the same model, prompted, the trained questioner's asked-for part is -0.012 $[-0.042, +0.014]$ and its incidental part $+0.069$ $[+0.023, +0.115]$. Counting the child as named when all of its own terms appear gives the same split, $+0.068$ for the incidental part. A looser test, at least half of the child's terms, moves events from incidental to asked-for and leaves the incidental part at $+0.040$ $[+0.001, +0.079]$, which holds at fifty thousand resamples under three seeds but crosses zero at one thousand. The asked-for part spans zero under all three tests. In incidental events the trained questioner's question named the parent 29 times, the comparator's 2. The other two suites show no excess to split.

What the disorder costs is not detectable here. Taking needs out of order should show up as answers built on unconfirmed premises. Answers are read where each run stops, so this reading

Table 12: Required-evidence coverage under the selection gate's rule, the only rule with a development reading: the trained questioner (two training seeds averaged within the task) at its own stopping point, the same model, prompted, at its own prefix of that length. Table 2 reads the same held-out runs at equal spend. Both suites with a development reading agree with it in sign and in excluding zero, and the MuSiQue gain shrinks from development to held out, as selection on the development split predicts. 2WikiMultiHopQA has no development reading, so its cell is a first reading rather than a replication. The splits were scored under different scoring hashes, so across splits this is an agreement in sign and interval, never a difference of differences. Development intervals are at 1,000 resamples, held-out ones at 10,000. The last column is the difference at each policy's own stop under the ceiling of eight.

suite	split	Δ	95% interval	n tasks	own-stop Δ
MuSiQue	development	$+0.156$	$[+0.121, +0.204]$	132	$+0.103$
MuSiQue	held out	$+0.128$	$[+0.102, +0.159]$	200	$+0.096$
StrategyQA	development	$+0.080$	$[+0.059, +0.103]$	333	-0.009
StrategyQA	held out	$+0.080$	$[+0.051, +0.112]$	200	-0.012
2WikiMultiHopQA	held out	$+0.079$	$[+0.052, +0.111]$	200	$+0.015$

Table 13: Every gated criterion for the trained questioner's two training seeds, development against held out, under one instrument and estimator. Question diversity is printed per training seed, since pooling two policies' questions measures something else, and read per task and rollout seed. Question length fails on every suite and split and makes every overall verdict a fail. Ungated rows are read at the shared cap of eight, not at equal spend (Appendix C). Coverage at depth two or more is absent, not zero, on 2WikiMultiHopQA, which has no required need below depth one. Promptness on MuSiQue falls from $+0.142$ on development, whose interval at 1,000 resamples excludes zero, to $+0.015$ held out, whose interval spans zero, so held-out promptness there is not decided and the drop between the splits is not a tested difference.

criterion	gated	MuSiQue		StrategyQA		2WikiMultiHopQA
		dev	held out	dev	held out	held out
required-evidence coverage	yes	$+0.156$	$+0.128$	$+0.080$	$+0.080$	$+0.079$
question diversity, seed 1	yes	0.827	0.826	0.888	0.849	0.805
question diversity, seed 2	yes	0.841	0.833	0.884	0.830	0.808
question length	yes	17.53 fail	17.76 fail	22.57 fail	22.67 fail	12.36 fail
malformed output	yes	0.000	0.001	0.003	0.001	0.005
coverage at depth ≥ 2	no	$+0.097$	$+0.076$	-0.070	-0.005	absent
breadth	no	$+0.045$	$+0.084$	-0.021	-0.005	$+0.021$
promptness	no	$+0.142$	$+0.015$	$+0.031$	$+0.006$	$+0.086$
overall verdict		fail	fail	fail	fail	fail

takes complete runs, over the 184 MuSiQue tasks on which both arms' complete runs define the out-of-order rate, not the population of the out-of-order row of Table 11, which holds both arms to the lower question count. Split by whether the trained questioner resolved more out of order than the comparator on the same task, and paired within task so that difficulty is held fixed, its answer advantage is larger where it resolved more out of order, not smaller: the strata differ by $+0.081$ $[+0.014, +0.150]$ in answer token overlap, holding at fifty thousand resamples under three seeds, and by $+0.087$ $[-0.004, +0.181]$ on exact answers, and the share of tasks with every required need resolved moves the same way, by $+0.140$ $[+0.027, +0.255]$. That is not evidence that disorder helps, since the strata are chosen by the trained questioner's own behavior and may simply be the tasks where it reaches more. What it rules out, at this resolution and on the trained questioner, is the harm the ordering result invites. Each training seed alone leans the same way on all four measures. The other two suites have 7 and 6 more-disordered tasks and are not read.

Independent rollouts, and the population. The structured-baseline comparison's own rollouts, drawn independently, agree with Table 2 in sign and in excluding zero: against the same model, prompted, the trained questioner gains $+0.119$ and $+0.082$ of coverage at equal retrieval spend on MuSiQue and StrategyQA, where the table prints $+0.112$ and $+0.070$. Every cell reads the 200 held-

out tasks per suite that the eligibility filter admits (Appendix A). The train-split contamination guard returned no violations over all $2{,}400$ of the trained questioner's held-out runs, and 40 violations over 40 development-split runs of the trained arm, so the zero is evidence rather than a guard that cannot fire.

F THE EIGHT CONTROLS

This appendix gives the readings behind the controls of Section 5: what each part of the protocol is worth to the trained questioner, and which controls can decide anything at all.

What a control is here. Seven comparator arms were registered in advance in code, each with a written consequence for the case where it matches the questioner it was declared against, the prompted one, and this appendix reads them against the trained questioner. They are answer length, compute budget, random questions, depth one, evidence blind, the single model and a determinism check. An eighth, a state-blind coverage template, was read without a rule, so it can make an unmatched table read wrong but cannot fire one. Figure 7 draws random questions, depth one, evidence blind and the single model, and Table 14 adds answer length, compute budget and the template. Each declared equivalence margin is 0.05 of required-evidence coverage except the answer-length control's 0.03, tighter because its consequence is an unconditional stop. The declared verdict reads a control as separated when its interval lies wholly above zero, which is checked first, as equivalent when the interval lies inside the margin without excluding zero, which fires the rule, and otherwise as inconclusive. No declared primary endpoint names the trained questioner, so these contrasts are exploratory as a class and carry intervals and no probability value. No control arm was run on 2WikiMultiHopQA or on the tool-use domains, so nothing here extends beyond the two suites read.

Per suite, the trained questioner leads random questions at equal spend by $+0.3474$ $[+0.3033, +0.3910]$ on MuSiQue, over the 176 tasks the control keeps, and by $+0.6325$ $[+0.5923, +0.6757]$ on StrategyQA, over 200.

With both arms at the lower question count, on MuSiQue the trained questioner leads its depth-one ablation by $+0.043$ and $+0.064$ for its two seeds and its evidence-blind ablation by $+0.084$ and $+0.087$, every interval excluding zero, and on StrategyQA neither separates.

The single model: equivalent at the shared cap, separated at equal spend. The single-model variant, one model playing questioner and drafter, reads -0.0028 $[-0.0129, +0.0075]$ against the trained questioner at the shared cap of eight, inside the margin of 0.05 without excluding zero, so the declared verdict there is equivalent and its consequence applies: the contribution is the Q&D algorithm rather than the two-model split. At equal spend it trails the split by $+0.0220$ $[+0.0114, +0.0335]$, and per suite by $+0.0257$ $[+0.0145, +0.0398]$ on MuSiQue and $+0.0186$ $[+0.0029, +0.0380]$ on StrategyQA. Each of those intervals lies inside the margin but excludes zero, which the declared verdict, checking separation first, reads as separated: at equal spend the split buys a small, detectable amount of coverage.

Three controls that cannot decide. The answer-length and compute-budget arms ask nothing, so their coverage is zero on every task and their deltas of $+0.885$ and $+0.884$ are true and are not evidence, which leaves those two explanations unfalsified rather than ruled out, and the determinism check reissues a recorded run's own questions, so its delta against that run is zero by construction and it has no row here. The question-length control of Section 5 is a separate instrument.

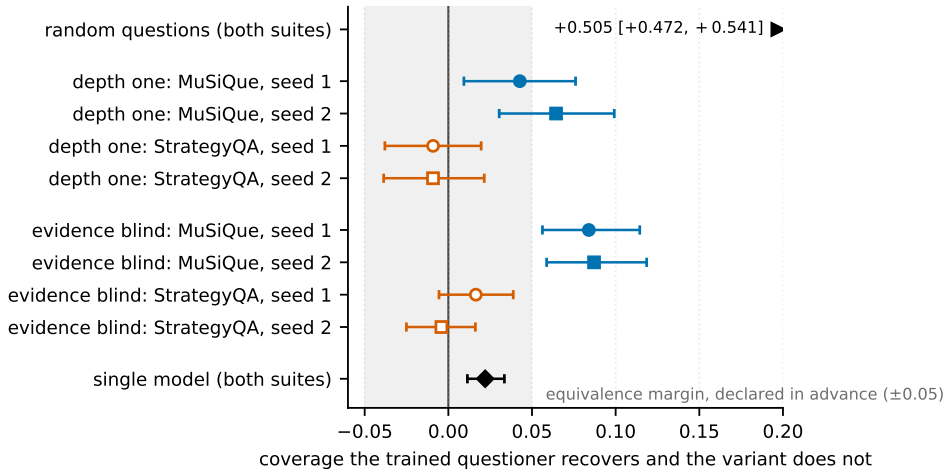


Figure 7: What each part of the protocol is worth to the trained questioner, each control read against the trained questioner's own weights and at the rule its spend requires. Random questions and the single model ask more than the trained questioner and are read with the control truncated to the trained questioner's spend, pooled over both suites and both seeds. The depth-one and evidence-blind ablations ask fewer and are read with both arms at the lower question count, per seed and per suite. A filled point excludes zero, decided at 50,000 resamples under three seeds for the ablations, and an open point spans zero. Removing what the questions say costs an order of magnitude more than anything else, and that reading lies beyond the axis, printed at its edge. Sequencing and the evidence each cost MuSiQue on both seeds and separate nothing on StrategyQA, where the shared budget is one or two questions. The single model's interval sits inside the grey margin declared in advance in code but excludes zero, which the declared rule reads as separated. At the shared cap it is equivalent (Table 14).

Table 14: The controls against the trained questioner, its two training seeds averaged within each task, pooled over the held-out splits of MuSiQue and StrategyQA, as required-evidence coverage of the trained questioner minus the control. The first column charges each control at most the trained questioner's spend, and the second reads both arms at the shared cap of eight retrieval calls. The three ablations of the questioner's own weights are read for each seed against its own ablation. Each control takes its own overlap with the trained questioner as its population, which is why the counts differ. †The control is read at the longest prefix of its own run that stays within the trained questioner's retrieval calls on the same task and rollout seed, and the trained questioner is read whole. On a pair where the control asks at least as many questions this is equal spend, both arms at the lower count. On a pair where the control asks fewer, the trained questioner keeps its extra questions, the gate-matched spend of Appendix C. *The depth-one and evidence-blind ablations ask fewer questions than the trained questioner, so their first column is not an equal-spend reading, and Figure 7 reads them with both arms at the lower question count.

control	at most the trained questioner's spend†	shared cap of eight	n
answer length	+0.8848 [+0.8660, +0.9020]	+0.8848 [+0.8660, +0.9020]	375
compute budget	+0.8841 [+0.8654, +0.9013]	+0.8841 [+0.8654, +0.9013]	375
random questions	+0.5049 [+0.4723, +0.5408]	+0.3648 [+0.3333, +0.4021]	376
state-blind template	+0.2439 [+0.2177, +0.2718]	+0.1892 [+0.1614, +0.2171]	400
depth one*	+0.1448 [+0.1242, +0.1668]	+0.1380 [+0.1176, +0.1596]	400
evidence blind*	+0.0701 [+0.0529, +0.0861]	+0.0369 [+0.0200, +0.0525]	400
single model	+0.0220 [+0.0114, +0.0335]	−0.0028 [−0.0129, +0.0075]	400

G STOPPING IN FULL

This section supports the stopping result of Section 5 (Table 3) and the call-budget sweep of Figure 2a: why filtered imitation teaches no persistence where it is needed, how the preference stage

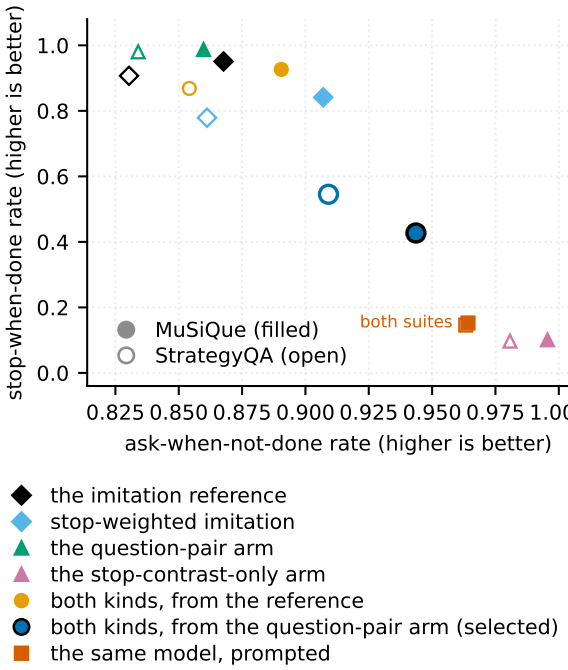


Figure 8: The trade every training arm makes between persistence and stopping: each marker is one trained arm at its rate of asking when the task is not done and of stopping when it is, with the same model, prompted, for reference. Arms trained on one kind of contrast sit at opposite corners, and the trained questioner (larger marker, two training seeds pooled) sits between them, asking at an unfinished state more often and stopping at a finished state less often than every other trained arm but the one trained on stop contrasts alone, with a coverage gain within 0.012 of the largest on MuSiQue and within 0.005 on StrategyQA. Shape marks the kind of training and color the arm. A filled marker is MuSiQue and an open one StrategyQA, and the same model's two suites nearly coincide. Development split, rates over the decision points each policy reaches.

Table 15: The gated one-sided length non-inferiority test on StrategyQA, with the MuSiQue verdict beside it: mean question words against the same model, prompted, where the upper bound of the relative interval may not exceed a margin of 0.10. All four trained arms fail on StrategyQA and the one from the question-pair arm fails MuSiQue as well. The reference passes by 0.007 of margin, so the criterion decides more than its precision can bear.

arm	words	comparator	relative 95% CI	StrategyQA	MuSiQue
imitation reference	13.80	12.99	$[+0.031, +0.093]$	pass	pass
stop contrasts alone	16.59	12.99	$[+0.256, +0.297]$	fail	pass
both kinds	14.31	12.99	$[+0.072, +0.129]$	fail	pass
both kinds, from the question-pair arm	22.57	12.99	$[+0.705, +0.768]$	fail	fail
stop rows down-weighted to 0.1	14.36	12.99	$[+0.079, +0.134]$	fail	pass

restores it and at what cost, the answer-node arm, and where the evidence gain is lost before the answer.

Two criteria fixed in advance that the trained questioner does not clear. The selection rule asked for a rise of at least 0.03 in the rate of asking at an unfinished state, a rate of stopping at a finished state of at least 0.85, and a non-inferiority test on question length. The trained questioner clears persistence on both suites, its rate of asking at an unfinished state up by $+0.076$ and $+0.079$ against the reference, and fails the other two on both: it stops at a finished state at 0.427 and 0.545, and its questions run four and a half and nine and a half words longer than the comparator's, each relative interval wholly above the margin (Table 15).

Table 16: Training composition against behavior on the development split. Persistence is the rate of asking when the task is not done and stop the rate of stopping when it is, per decision point the policy itself reaches. Coverage is against the same model, prompted, at gate-matched spend, with a task-clustered interval. Diversity is the within-task distinct-trigram share, against a gate floor of 0.65 and untrained values of 0.380 and 0.458. Asks is questions per episode. Every arm but the last of each block is one training run, and the last averages two, scored in a separate pass against the same comparator runs.

composition	pairs	persist.	stop	Δ cov.	95% CI	div.	asks
MuSiQue, 132 tasks							
imitation reference	none	0.868	0.951	$+0.049$	$[+0.002, +0.097]$	0.830	3.05
stop contrasts alone	18,919	0.996	0.101	-0.005	$[-0.056, +0.046]$	0.447	6.95
both kinds	31,473	0.890	0.926	$+0.068$	$[+0.021, +0.117]$	0.801	3.36
both kinds, from the question-pair arm	31,473	0.944	0.427	$+0.156$	$[+0.121, +0.204]$	0.834	4.24
StrategyQA, 333 tasks							
imitation reference	none	0.830	0.907	$+0.080$	$[+0.052, +0.111]$	0.845	1.34
stop contrasts alone	18,919	0.981	0.097	-0.017	$[-0.039, +0.004]$	0.445	5.75
both kinds	31,473	0.854	0.869	$+0.084$	$[+0.056, +0.111]$	0.845	1.47
both kinds, from the question-pair arm	31,473	0.909	0.545	$+0.080$	$[+0.059, +0.103]$	0.886	2.13

Imitation leaves the hard states without a lesson, and a contrast supplies one. The imitation exporter writes a stop where the required evidence was in hand, the best sampled question where evidence was missing and that question clears the acceptance floor, and nothing where none clears it. The last branch suits an imitation target, since a state every sample missed says more about the sampler than the state, but it drops precisely the states whose lesson is that stopping is wrong, so the policy trained on the rest has no feature that separates a hard state from a finished one. It refuses $5{,}030$ states beside $14{,}302$ exported rows on the export the reference trained on, and $24{,}538$ beside $43{,}837$ on the pooled export, which keeps only runs of the same model, prompted, with no cohort filter, and is not the widest export of Appendix D, which admits every arm. A preference need not name a good question, only that asking beat stopping, which gold coverage settles on its own, so at every unfinished state, the refused ones included, the export places the best available question against stopping, ordered without consulting the question's text: $18{,}919$ of the file's $36{,}349$ pairs. Three compositions were gated against one comparator (Figure 8): those contrasts alone, and both kinds together, $31{,}473$ pairs, fitted to the imitation reference or to the policy already trained on question pairs.

What the composition decides. The stop contrasts move stopping only when they dominate the objective or meet a policy that already asks well (Table 16). Alone they raise persistence and buy nothing else: the coverage point estimate is negative on both suites with neither interval excluding zero, within-task diversity is the lowest of any trained arm measured here, and more than twice the reference's questions recover no more required evidence. Fitted to the imitation reference beside question pairs, where they are still the majority, they leave stopping when done intact and return almost none of the persistence gain. Fitted to the policy already trained on question pairs from the reference, the same file moves both cells and gains more than twice as much MuSiQue coverage as when fitted to imitation, so the final stage moves the policy toward persistence. That starting policy has no development reading here. Held out, the final stage and the policy it starts from gain $+0.1284$ and $+0.1177$ on MuSiQue (Table 9), a point ordering and not a tested difference. The question-pair row of Table 3 is a different checkpoint, fitted from the pooled imitation export: it gains $+0.167$ $[+0.127, +0.215]$ and $+0.084$ $[+0.057, +0.113]$ and leaves both stopping rates about where imitation left them. Down-weighting the stop rows in the imitation loss cannot put back the refused states and is not free: with the stop rows down-weighted to a tenth, stopping at a finished state falls to 0.7792 on StrategyQA, below the selection floor.

Against GPT-OSS-120B, the trained questioner keeps the stopping half of the calibration and not the adverse half. On the same instrument and held-out split, it stops at 0.64 to 0.92 of its own finished states against that model's 0.23 to 0.81, higher by $+0.251$, $+0.430$ and $+0.116$, every interval excluding zero. It asks when evidence is still missing more often than that model on MuSiQue, by $+0.035$ $[+0.017, +0.059]$, less often on StrategyQA, by -0.067 $[-0.088, -0.050]$, and indistinguishably on 2WikiMultiHopQA, at -0.007 $[-0.022, +0.005]$, so it is lower only on the suite where it is also lower than the same model, prompted. Each rate is read only where a policy reached

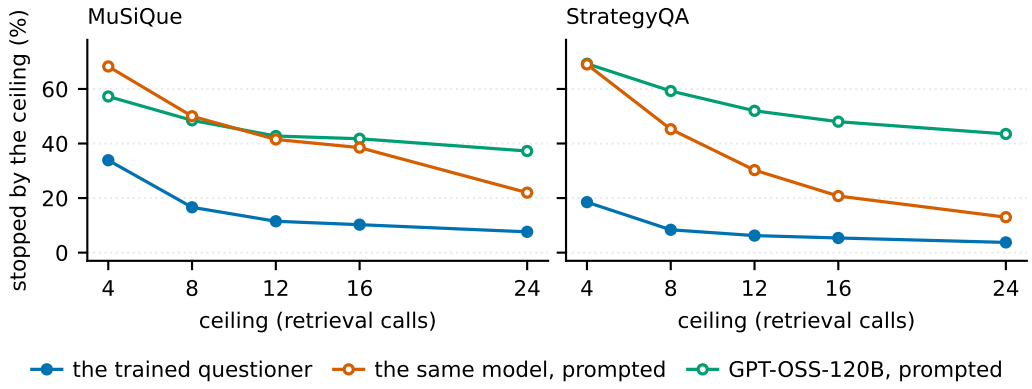


Figure 9: Who decides when a run ends: the share of units (one task at one rollout seed) that the retrieval ceiling rather than the policy stopped, at five ceilings across a sixfold budget range. From the smallest ceiling to the largest the trained questioner is stopped on 34% to 8% of MuSiQue units and 19% to 4% of StrategyQA units, below the same model, prompted, and GPT-OSS-120B, prompted, at every ceiling on both suites, which are stopped on 13% to 69%. The same policy also stops on unfinished work (Section 5).

a decision point of the matching kind, a population selected on both policies' trajectories, and rates rather than counts are compared because the two take very different numbers of turns.

The call-budget sweep of Figure 2a. On MuSiQue the trained questioner leads GPT-OSS-120B by $+0.102$ $[+0.074, +0.134]$ at the cheapest ceiling, where both spend about 3.2 calls, and by $+0.064$ $[+0.037, +0.095]$ at the most generous, spending 5.9 calls against 12.1. Its coverage at a ceiling of four is above that model's at twenty-four, by $+0.041$ $[+0.013, +0.071]$. It leads the same model, prompted, by $+0.093$ to $+0.096$ at every ceiling, and each training seed alone is ahead of both comparators at every ceiling.

On StrategyQA no difference from the same model, prompted, is decided at any ceiling, $+0.008$ to $+0.012$ with every interval spanning zero, on 2.5 to 3.9 calls against 3.6 to 10.0. No difference from GPT-OSS-120B is decided at the cheapest ceiling, $+0.009$ $[-0.016, +0.037]$, and the trained questioner trails that model from the second ceiling upward, by -0.027 $[-0.050, -0.002]$ at a ceiling of eight and -0.056 $[-0.079, -0.035]$ at twenty-four, where that model spends two to three and a half times as many calls. From the smallest ceiling to the largest the trained questioner gains $+0.023$ on MuSiQue and $+0.015$ on StrategyQA, GPT-OSS-120B $+0.062$ and $+0.080$. On FRAMES, which has no need graph and is read on answers, no difference from the same model, prompted, is decided at any ceiling, the point estimates running $+0.010$ to $+0.019$, and none from GPT-OSS-120B up to a ceiling of twelve. The trained questioner trails GPT-OSS-120B at sixteen and twenty-four, by -0.030 and -0.033, where that model spends two to two and a half times as many calls. Figure 9 shows how often the cap rather than the policy ends a run.

The answer-node arm against its own control. The repair that follows from a policy that stops with the answer-bearing need uncovered, as the trained questioner still does on StrategyQA, is to supervise stopping on that node itself. An imitation arm trained that way covers that node less often than its control, refreshed by the same imitation recipe without the conditioning, by -0.095 $[-0.148, -0.048]$, -0.075 $[-0.125, -0.033]$ and -0.293 $[-0.358, -0.233]$ on the three suites (200 paired tasks each), every interval excluding zero and every sign stable across resample counts and seeds. Against the same model, prompted, the same endpoint reads -0.128, -0.175 and -0.370.

The node's depth tests the alternative reading. That reading is that the deficit measures needs not yet nameable rather than stopping that was wrong. On 2WikiMultiHopQA it inverts: the deficit is larger where the answer node was nameable from the task, -0.394 $[-0.500, -0.303]$, than behind a prerequisite, -0.203 $[-0.293, -0.132]$. On StrategyQA the nameable stratum is too small to decide, at -0.043 $[-0.114, +0.014]$ over 70 tasks, and MuSiQue has none by construction, no graph in it having a required terminal need at depth zero. So the stopping defect is not a mislabeled target:

handed the answer node directly, an imitation policy does worse on it, which locates the problem in what it generalizes at inference rather than in what it was taught to predict.

G.1 Where the evidence gain is lost before the answer

At each arm's own stop under the ceiling of eight, with the trained questioner's two training seeds averaged within the task, only MuSiQue has extra answer-node coverage to convert (Table 17), and the gain it implies, from the comparator's own rate of answering well with and without that node, is below the detectable size. The declared test of a stronger reader, Claude Opus 5 given the retrieved paragraphs uncut and told to answer only from them, is decided in none of twelve cells, at each arm's own stop and at equal spend alike. On StrategyQA that reader answers that the evidence does not settle the question on about 45% of states, becoming a strict entailment reader rather than a measure of headroom. The drafter's cut at $1{,}200$ characters hides no answer on MuSiQue (0 of 212 answer-bearing paragraphs) and 6 of 174 on 2WikiMultiHopQA, and is not read on StrategyQA, whose answers are yes or no. The frozen answerer is not deterministic either: redrawing it on byte-identical requests turns the MuSiQue answer-overlap difference into $+0.036$ $[+0.002, +0.069]$.

Table 17: The answer gain the trained questioner's evidence would imply, the gain observed, and a stronger reader. The node column is the difference in covering the answer-bearing need. Answer token overlap, the trained questioner minus the same model, prompted, 200 held-out tasks per suite, $10{,}000$-resample intervals. The minimum detectable effect is at 80% power and two-sided 0.05 from the observed paired spread. The last column is the declared primary, the difference between that difference read under a Claude Opus 5 evidence-only reader and read under the frozen answerer, at each arm's own stop. Intervals here cluster on the MuSiQue question template, and the point moves with them: the estimate is the mean over clusters, so here each template weighs the same whatever its number of tasks, where Table 2, clustered on the task, weighs each task the same. That is why the observed MuSiQue cell reads $+0.026$ here and $+0.024$ there for the same runs.

suite	node	implied	observed	detectable	stronger reader
MuSiQue	$+0.097$	$+0.043$	$+0.026$ $[-0.006, +0.060]$	0.047	$+0.026$ $[-0.015, +0.067]$
StrategyQA	-0.005	-0.001	-0.003	0.056	-0.001 $[-0.053, +0.049]$
2WikiMultiHopQA	$+0.001$	$+0.001$	$+0.031$	0.053	$+0.007$ $[-0.018, +0.035]$

H Transfer in full

This appendix supports the transfer result of Section 5 (Figure 3): the trained questioner's gain in task success on a tool-use benchmark, the follow-up turns that were its declared primary endpoint, and its comparison with GPT-OSS-120B. It also reports two campaigns of the questioner-drafter protocol, the questioner-drafter split of Q&D, run with prompted models and no training: with Claude Sonnet 5 in every role it needs fewer follow-up turns on retail, a difference not decided on airline, and with GPT-OSS-120B in every role it decides nothing on banking. Neither campaign has a trained arm, so neither bears on the trained questioner's declared primary endpoint.

The trained questioner, at equal and enforced budgets. We compared the trained questioner from two training seeds with the same model, prompted, on a tool-use benchmark with a simulated customer (Barres et al., 2026), on its retail and airline domains. A fork point is a prefix of a recorded dialogue from which a run continues, and each domain has thirty-four, drawn from 25 retail and 17 airline tasks. Each fork point was run by three policies, the same model, prompted, and the two training seeds, at three rollout seeds and under two prompt variants, $1{,}224$ units over the two domains, every one completed. Both prompt variants add to the questioner's question-answering prompt one paragraph saying that a question reaches the store's records and never the customer. The base prompt keeps the question-answering stopping rule, to stop once every required need is resolved, and the stop prompt replaces it with one written for tool use, to stop once the evidence names the records the task's next action needs, since the drafter acts only after the questioner stops. The two are read separately and never pooled. Every difference is read over tasks, with the pairs inside a task averaged first. In both arms the harness enforced sixteen questions and sixteen tool calls separately during the dialogue, the questioner ran with its reasoning mode off, and the drafter

saw each evidence record whole. The analysis rules were fixed in writing before the data each rule governs.

Retail success rose in every seed and prompt variant (Table 18). Airline success rose in each pooled cell and in both variants for the second seed, and the first seed's airline cells are undecided, with two or three of seventeen tasks better per seed and at most one worse. Follow-up turns, the declared primary endpoint, fell in every pooled estimate, and no pooled interval excludes zero, so the endpoint is not established.

On the two confirmatory question measures, the gold-object hit rate, the share of questions whose retrieved records include one the task's gold actions touch, rose by 0.185 to 0.265 (in retail on twenty-one of twenty-five tasks, with none against), and gold-object recall rose by 0.324 to 0.512, while the number of questions fell slightly.

Two further measures, declared after one interim look and so exploratory, point the same way in every domain and prompt (Table 19): more questions retrieve at least one record, and fewer exactly repeat an earlier question. A looser repeat measure falls in retail. In airline no difference in it is decided, and about half of the trained questioner's questions nearly repeat an earlier one, 0.497 against 0.421 under the base prompt.

In a worked example chosen by a rule fixed before any transcript was opened (retail task 15), no question of the same model, prompted, retrieved any of the five records the task needs, and it failed. Under the base prompt ten questions of the first trained seed each retrieved at least one of those records, and that seed succeeded under both prompt variants.

Table 18: The trained questioner against the same model, prompted, on the tool-use benchmark, at equal, separately enforced budgets. Differences are trained minus prompted, over tasks, with bias-corrected intervals at $10{,}000$ resamples.

		Task success			Follow-up turns	Hit rate	Recall
domain	prompt	Seed 1	Seed 2	Pooled	Pooled	Pooled	Pooled
retail	base	$+0.213$	$+0.193$	$+0.203$ $[+0.100, +0.357]$	-0.660 $[-1.853, +0.387]$	$+0.254$	$+0.512$
retail	stop	$+0.233$	$+0.167$	$+0.200$ $[+0.100, +0.340]$	-0.737 $[-2.343, +0.870]$	$+0.265$	$+0.504$
airline	base	$+0.059$	$+0.078$	$+0.069$ $[+0.010, +0.196]$	-0.059 $[-1.064, +0.941]$	$+0.185$	$+0.368$
airline	stop	$+0.029$	$+0.098$	$+0.064$ $[+0.005, +0.201]$	-0.441 $[-1.643, +0.985]$	$+0.224$	$+0.324$

Table 19: Exploratory question measures on the tool-use benchmark, declared after one interim look and so exploratory whatever they show. Differences are trained minus prompted, over tasks (25 retail, 17 airline), with the two training seeds pooled within each task and bias-corrected intervals at $10{,}000$ resamples, not read at $50{,}000$. A near-exact repeat shares at least four content words with an earlier question in the same dialogue.

domain	prompt	retrieved a record	exact repeat	near-exact repeat
retail	base	$+0.582$ $[+0.447, +0.693]$	-0.613 $[-0.675, -0.527]$	-0.205 $[-0.288, -0.126]$
retail	stop	$+0.587$ $[+0.450, +0.701]$	-0.609 $[-0.671, -0.522]$	-0.237 $[-0.313, -0.163]$
airline	base	$+0.382$ $[+0.258, +0.512]$	-0.516 $[-0.605, -0.427]$	$+0.076$ $[-0.065, +0.249]$
airline	stop	$+0.394$ $[+0.256, +0.538]$	-0.487 $[-0.576, -0.393]$	$+0.027$ $[-0.100, +0.202]$

Against GPT-OSS-120B, prompted. We ran the same comparison with GPT-OSS-120B, prompted, as the comparator's questioner, under the same harness, budgets and roles, declared before any of its units ran. As questioner, GPT-OSS-120B came within a tenth of its token limit on 0.6% of calls. In retail the trained questioner cut follow-up turns and raised success in both prompt

Table 20: The trained questioner against GPT-OSS-120B, prompted, on the tool-use benchmark at equal, separately enforced budgets. Differences are trained minus prompted, over tasks, with the two training seeds pooled within each task, and 95% intervals. A negative difference in follow-up turns means fewer turns from the simulated user. Bold: decided, every interval from 50,000 resamples at three bootstrap seeds excludes zero. Hit rate and questions are secondary point differences.

domain	prompt	Follow-up turns	Task success	Hit rate	Questions
retail	base	**−1.60** [−3.05, −0.19]	**+0.170** [+0.070, +0.310]	+0.099	−0.87
retail	stop	**−1.75** [−3.18, −0.33]	**+0.127** [+0.043, +0.240]	+0.130	+0.19
airline	base	+0.75 [−0.39, +1.75]	0.000 [−0.039, +0.049]	+0.052	+1.09
airline	stop	−0.07 [−1.42, +1.21]	+0.034 [−0.005, +0.147]	+0.067	+1.40

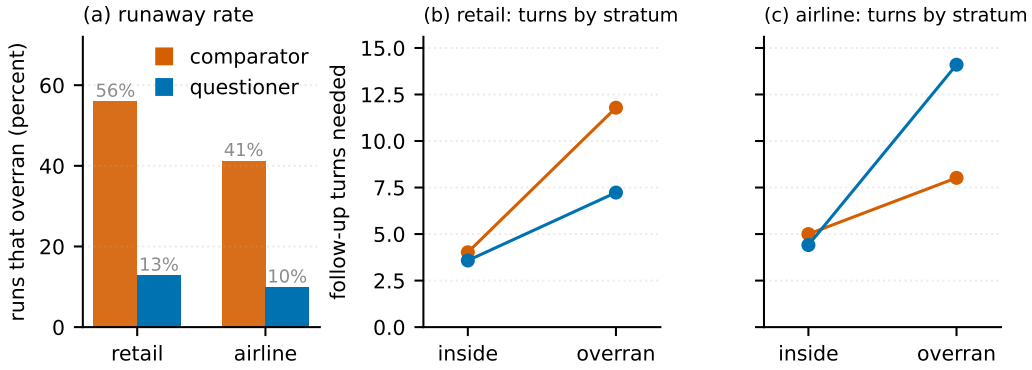


Figure 10: What transfers on a user-facing benchmark, and what does not. (a) The share of runs that overran the recorded budget: the questioner-drafter protocol's dialogues run away roughly four times less often, on both domains. (b) On retail the protocol needs fewer follow-up turns in both populations, almost all of the effect in the overrun one: the inside-budget levels sit 0.44 turns apart. (c) On airline the rate still favors the protocol, 10 runs against 42, but its few runaways are longer, so the mean cannot separate and only retail's interval excludes zero. The arms are the questioner-drafter protocol and a self-ask decomposition comparator, both run with Claude Sonnet 5 in every role, on an exploratory-flagged population that nothing here was trained on. The inside-budget levels describe two populations and are not a second estimate of the effect.

variants (Table 20). The reduction in follow-up turns holds for the pooled seeds and for the first training seed, and the second seed points the same way without excluding zero. Success was higher on ten of twenty-five tasks and lower on one, in each variant. Per seed, the first seed's base-prompt follow-up reduction passes a Holm-corrected sign test at 0.05 and no success cell does, while the intervals decide several. The trained questioner's questions retrieved a record the gold actions touch more often, by 0.099 and 0.130 per question. In an exploratory analysis, a larger share of its questions retrieved at least one record than GPT-OSS-120B's in all four settings, 0.595 against 0.323 and 0.602 against 0.314 in retail and 0.555 against 0.433 and 0.548 against 0.409 in airline, while in airline under the stop prompt it also repeated earlier questions more often. In airline neither follow-up turns nor success showed a decided difference, and success differed on three of seventeen tasks, one higher and two lower under the base prompt and the reverse under the stop prompt. There the trained questioner asked between one and one and a half more questions per continuation than GPT-OSS-120B. Two units in which GPT-OSS-120B asked nothing were read as zero spend under a rule fixed before any comparison was read.

The questioner-drafter protocol with Claude Sonnet 5. This is a separate fork campaign with no trained arm, so its result concerns the protocol and not the trained questioner's declared endpoint above. With Claude Sonnet 5 in every role on both sides, the questioner-drafter protocol was compared with a self-ask decomposition comparator on forked dialogue prefixes, 34 fork points per domain at three seeds, 204 runs per domain, paired at every fork point, with the follow-up turns the simulated customer supplies as the endpoint. The runs are flagged exploratory, so the contrast is read as an interval with no probability value. Both arms saw only the first 1,200 characters of each tool result, so the comparison is symmetric but read on truncated records. It rests on the recorded

runs of that fork campaign, which drew five evidence units per search. Every spend total for this environment is a lower bound. On retail the protocol needs 4.47 fewer follow-up turns $[-6.28, -3.04]$, clustered on the 32 recorded dialogues, and every other clustering unit also excludes zero. The difference is in how often a dialogue runs away rather than in the typical dialogue (Figure 10): the comparator overran a retrieval cap of 16, charged only after the dialogue, on 57 of its 102 retail runs against 13 of the protocol's. In both arms every retail question addresses the retrieval index and none the customer.

On airline the interval contains zero at every clustering unit (Figure 10c). In airline held-out runs the only arm permitted to address the customer, a prompted questioner, almost never does so, an exploratory reading: 8 of its $1{,}243$ questions go to the customer. Banking, a third domain, is the banking customer-service domain with a document knowledge base that the benchmark's public code repository distributes. It was run as a separate live campaign of three arms with GPT-OSS-120B, prompted, in every role and no trained arm: the questioner-drafter protocol, its single-model control and a no-question baseline. All $1{,}472$ recorded ask turns address the retrieval index and none the customer. Over the 83 of 97 tasks that survive in all three arms, the questioner-drafter protocol's difference from the no-question baseline is -1.29 $[-3.51, +1.13]$ follow-up turns over tasks and -0.78 $[-2.99, +2.04]$ over its 18 templates, the benchmark's declared unit. Every interval on every contrast covers zero, and every detectable floor exceeds the observed difference by between 2.03 and 13.78 times, so this null is a statement about the population's size and not a demonstration that the protocol does nothing.

What a simulated customer leaves untested. Users who are heterogeneous, impatient or working from shifting instructions (Yu et al., 2026; Yang et al., 2026), long-running personalized interaction (Chen et al., 2026) and interactive coding sessions (Wu et al., 2026c) have their own benchmarks, and whether a need graph survives a user who does not behave is open.

I LABEL VALIDITY

This appendix supports the check of the pair-ordering rule against a model panel and two people (Section 5) and the statement that prerequisite edges rarely gate retrieval (Section 7).

I.1 THE MODEL PANEL, AND ONE REPLACEMENT

Pairs are ordered by a rule, and the rule is checked against readers. Three model families rated blinded pairs ordered by the rule, with planted items whose answer is fixed in advance. One member, the model that generated the corpus it rated, agreed with the rule barely more often than chance and was replaced by a model from a fourth family presented in mirrored option order. The other two members' votes were not re-collected. The replacement reached the pairwise orderings only: $4{,}476$ of the $4{,}610$ verdicts of the instrument that ranks every candidate at a state carry the replaced member.

Table 21: chance-corrected agreement on the axis that decides a pair, slot coding, sided votes

comparison	raw	Cohen's κ [95% clustered]	n / clusters
the two retained members, both panels	77.8%	0.555 [0.481, 0.624]	2,508 / 201
first retained with replaced	65.2%	0.304 [0.244, 0.364]	3,156 / 211
second retained with replaced	58.2%	0.163 [0.053, 0.259]	2,648 / 202
first retained with replacement	79.0%	0.581 [0.522, 0.638]	2,596 / 202
second retained with replacement	84.1%	0.682 [0.629, 0.734]	2,223 / 197
rule with first retained	61.5%	0.230 [0.167, 0.294]	3,480 / 215
rule with second retained	67.5%	0.350 [0.265, 0.437]	2,931 / 207
rule with replaced	52.2%	0.044 [−0.027, 0.110]	4,312 / 216
rule with replacement	65.5%	0.310 [0.236, 0.382]	3,250 / 208
rule with panel majority, original	64.0%	0.279 [0.217, 0.341]	3,218 / 213
rule with panel majority, revised	68.1%	0.361 [0.289, 0.430]	2,986 / 207

Agreement with the rule, corrected for chance and clustered at the question. Every coefficient (Cohen, 1960; Krippendorff, 2019) is computed on the presentation slot, a per-item coin flip every

party is blind to, since coding a pair by which side the rule chose makes the coefficient identically zero. Intervals are clustered over 219 questions and run 2.2 to 2.7 times wider than row-level ones. Against the rule (Table 21), the replaced member's interval contains zero and the retained members sit at 0.230 and 0.350, so the panel is two fair-to-moderate raters and one rater not distinguishable from chance. The rule's agreement with the panel majority, 64.0% on the 3,218 pairs the original panel ordered and 68.1% on the 2,986 the revised panel ordered, is not over a fixed population. The pooled 0.279 averages a stratum at 0.386 with one at or below chance, the pairs where the chosen side resolved no required need, at -0.095 $[-0.241, 0.044]$ and -0.028 $[-0.205, 0.142]$ under the two panels.

I.2 Two human raters

Both people passed every hidden check, and they sit where the panel sits. A blinded bundle of 113 items, a hundred real pairs from MuSiQue with ten foils and three attention checks, followed the key design the model raters saw. Each of the two raters answered every item. Both raters scored 13 of 13 on the planted items and chose the first slot on 52.4% [40.3, 64.2] and 50.0% [38.4, 61.6] of the items they decided, so neither answered from position, which matters because the slot order is shared across raters. They agree with the rule on 65.1% [52.8, 75.7] of 63 decided items and 67.6% [55.8, 77.6] of 68, against the panel majority's 68.1% on the 72 of these items it decided, both intervals containing the panel's value, so the rule's agreement with its raters is not a consequence of the raters being models.

Table 22: Agreement between the two people on the human anchor's items, per question asked of them, with chance-corrected agreement.

question asked of the rater	items	raters agree	chance-corrected
does candidate A reach an unstated need	98	87.8%	+0.756
does candidate B reach an unstated need	92	88.0%	+0.761
which candidate is the better move	56	67.9%	+0.241

The two labels split sharply. The reaching judgment, whether a candidate reaches a need the task never stated, has a human anchor at 0.76, and the preference ordering does not. The two retained model raters show the same split on 280 matched pairs in one forced-pair format, 0.878 [0.798, 0.939] on which candidate reaches further past what the user stated against 0.556 [0.373, 0.709] on which is the better next question.

Scope of the two-rater reading. Two raters cannot separate agreement between people in general from agreement between these two, no three-way figure exists, and the pairs come from one suite. The two are not interchangeable: the second tracks the model panel more closely than the first, 86.7% against 68.5% on the 51 items both people and the panel decided, 13 against 4 on the items where the two differ. Where the panel was undecided the people split too. Neither person judged the prerequisite edges.

I.3 What readers make of the need graphs

Two models read the nodes, on one suite. On 420 nodes of 2WikiMultiHopQA shown blind, with twenty planted foils all caught, two raters of different families agreed on 96.9%, a chance-corrected 0.776 over three categories, and endorsed 265 of the 267 nodes the graph calls required. A graph calls a need optional when its builder links it to no evidence paragraph, which happens only on StrategyQA and 2WikiMultiHopQA, and coverage is a ratio over the required needs alone, so an optional need never enters it. On 2WikiMultiHopQA a need is optional when the builder's paragraph locator finds no paragraph for it, and of the nodes the graph calls optional there, 96.2% [91.5, 98.4] are really required, so **coverage is overstated, in one direction**, on the 3.3% of 2WikiMultiHopQA's nodes that are optional. On StrategyQA the 18.6% of nodes that are optional are the steps no annotator linked to a paragraph, operations over earlier answers, so that the same overstatement holds there is an assumption, not a reading. The reading covers 1.3% of one suite, and the other two suites have no node-level reading.

A rater cannot read an edge, so the edges were corrupted on purpose. As the benchmark writes it, the dependent need contains the prerequisite as a literal substring on 200 of 200 items, so every

edge reads as necessary. With the prerequisite resolved, a rater rejected six of six true prerequisites. That instrument returns no value rather than a perfect one. A recomputation of depth over the stored runs of the run set aside (Appendix L) dropped a fifth and then two fifths of the prerequisite edges at random, five draws each. Depth-weighted recall moves by at most 0.006 on any suite and never changes sign, and the deepest need resolved moves by up to 0.044, toward zero on all three suites, so a random edge error can hide a depth result and cannot manufacture one. The test covers random deletion only, not a systematic error or edges that ought to exist and do not, which matter most on StrategyQA, and it is read at the shared cap.

I.4 Do the edges gate retrieval? An automated probe

The criterion was committed before the probe produced any output. For each prerequisite edge in the held-out tasks, Claude Opus 5, seeing the task and the answers of the parent's ancestors, never the parent's answer or the child, writes five search queries under each of two prompts, one asking what to look up next and one asking to answer the task directly. A query hits when the task's own retriever returns all of the child's evidence in its top five from the released pool. The queries are written again with the parent's answer added, and with a random same-depth need's answer from another task instead. An edge is *validated* when the bare question misses the child, at most one query of five hits without the parent's answer, adding it raises the rate by at least 0.4 over the better of the two other contexts, and a model answering the parent from memory fails. It is *refuted* when three or more queries hit without the parent's answer, or the parent is answerable from memory, and *untestable* otherwise, including every child with no evidence of its own or evidence shared with its parent (Table 23).

Table 23: Whether a prerequisite edge gates retrieval: per-query rates at which a query retrieves the child's evidence in the top five, over testable edges, the better of two prompts. StrategyQA parents carry no answer, so their evidence title stands in for it.

	MuSiQue	StrategyQA	2WikiMultiHopQA
prerequisite edges	466	383	191
bare task question	0.35	0.72	0.48
another task's question (chance)	0.28	0.24	0.60
queries without the parent's answer	0.70	0.72	0.71
queries with the parent's answer	0.83	0.72	0.86
queries with a random entity instead	0.70	0.74	0.69
validated / refuted / untestable	12 / 391 / 63	0 / 119 / 264	8 / 144 / 39

The pools are small, twenty paragraphs a task on two suites and ten on the third, and the retriever returns five, so a child's paragraph is often in reach of any sensible query. The parent's answer helps on MuSiQue and 2WikiMultiHopQA, where a random entity adds nothing, so the edges carry some of the dependence they encode. On StrategyQA the parent's evidence title is often already in the question, and no edge can pass. In these pools an edge records the benchmark's decomposition far more than a barrier to retrieval, and only 20 of the $1{,}040$ edges pass the declared test, so the depth measures credit reaching the evidence a decomposition places deeper, not evidence that could not have been retrieved before its prerequisite.

With the refuted edges removed and depth recomputed, the trained questioner's depth-weighted recall gain is $+0.109$, $+0.052$ and $+0.053$ and its deepest-need gain $+0.054$, $+0.043$ and $+0.040$, every one decided. Graphs that keep only validated edges are nearly flat and cannot separate a depth gain from a flat one. Eleven of the twelve cells of the two prunings sit inside the band from 200 random removals of as many edges per suite and depth, and the twelfth, the StrategyQA deepest need under the first pruning, falls below its band and sits inside it under the other reading of which class an edge takes when it both fails the evidence condition and is refuted. On the children themselves, the trained questioner resolves those of validated edges more often than the same model, prompted, by $+0.205$ on MuSiQue and $+0.344$ on 2WikiMultiHopQA, over 12 and 8 children, and those of refuted edges by $+0.141$ and $+0.059$, the last with an interval that spans zero, so its gain is not confined to the edges that fail the test.

I.5 Gold defects, exclusions, and what remains unmeasured

Gold defects and excluded populations. Twenty tool-use episodes carry a reference reward that disagrees with the task's own recorded outcome, 4 airline and 5 retail on the held-out split and 5 and 6 on the training split, all produced by the benchmark's own reference agent (Barres et al., 2026), so no policy's success on those tasks is interpretable and every success figure excludes them. No corrected release addresses them, and the one for its predecessor (Yao et al., 2025; Cuadron et al., 2026) states that the two-agent copies of the same domains had not been revised. Nineteen episodes minted a second time are excluded by identifier. The 8,694 episodes produced from an uncommitted working tree, 227 of them stamped held-out, are admissible as training data only, because their code cannot be named, which leaves 62,185 scored episodes. The reporting code's eligibility predicate also requires the held-out split.

What remains unmeasured. Whether a policy can be proactive without harassing its user needs a precision over questions put to the simulated user, which no scorer computes, so that claim is unmeasurable with the present code although the episodes exist.

J Scale and family

This section gives the scale reading Section 7 points to: parameter count does not order the trained arms, preference training at 32 billion parameters shows no larger ranking gain than at 8 billion, and a second model family was trained by imitation only. Table 24 holds the imitation recipe of record fixed across three base sizes under one scoring version, since comparing each size's strongest checkpoint would attribute a selection effect to scale.

Table 24: The matched-recipe ladder: each cell a paired coverage difference against the same model, prompted, at gate-matched spend, on the development tasks. Three-seed rows give the mean across training seeds with the observed range in parentheses, and the largest size is one training run.

base	training seeds	MuSiQue ($n = 132$)	StrategyQA ($n = 333$)
1.7B	3	$+0.1098$ $(+0.1016, +0.1187)$	$+0.0309$ $(+0.0273, +0.0349)$
4B	3	$+0.0434$ $(+0.0215, +0.0549)$	$+0.0640$ $(+0.0611, +0.0691)$
8B	1	$+0.0492$	$+0.0804$

Parameter count does not order these arms. Every middle-size seed departs from the straight line between the smallest and largest size, below it on MuSiQue and above it on StrategyQA, each by more than the seed half-range at that size. The three middle-size seeds are not separable, the comparison's minimum detectable difference at conventional power being 0.054, so the replicates settle the direction of the departure, not its size. A genuine effect of scale does not reverse sign between two suites on the same checkpoints, so we report no ordering by size, and the ladder is an exploratory development reading.

The prompted comparators stop differently by size. The smallest model, prompted, never stops at a covered state, against 0.0066 and 0.0081 for the middle size and 0.1525 and 0.1460 for the largest, so a contrast at gate-matched spend pays a checkpoint for stopping well against a comparator that cannot. Read at each arm's own stop under the shared cap of eight, eleven of the fourteen checkpoint-by-suite cells behind the ladder go negative: the ladder supports efficiency per question at gate-matched spend, not more evidence in the end.

Where size does show. On FRAMES, where the trained questioner trails GPT-OSS-120B, prompted, at the two largest ceilings (Appendix G), the same model, prompted, trails GPT-OSS-120B further at every ceiling, with an interval excluding zero from a ceiling of twelve upward, so that shortfall is size and not the absence of training.

Preference training at 32 billion parameters. The preference stage that fits both pair kinds to the imitation reference was also run from a 32-billion-parameter imitation checkpoint and read offline on the development question pairs, as the gain in ranking accuracy over that starting checkpoint, a development ranking measure and not held-out coverage. Imitation loss rose at every saved checkpoint after the first, so under a rule fixed before any preference value was read the final checkpoint is reported first and the lowest-loss checkpoint second (Table 25). The final checkpoint ranks pairs

better than its start by $+0.031$ $[+0.001, +0.063]$, better on 24 of 58 tasks and worse on 12, and the lowest-loss checkpoint by $+0.016$ $[0.000, +0.034]$. Among pairs whose two questions are within ten tokens of each other in length the final checkpoint's gain is $+0.009$ $[-0.029, +0.044]$, while the 8B arm trained the same way gains $+0.059$ $[+0.017, +0.096]$. At the larger scale the gain does not survive matching question length and the 8B gain does. The two scales are not tested against each other.

Table 25: Preference training at two scales, read offline on the development question pairs as the gain in ranking accuracy over each run's own starting checkpoint, with 95% intervals over tasks. The 32-billion-parameter final checkpoint is reported first by a rule fixed before the reading, because imitation loss rose at every checkpoint after the first. Length matched keeps the pairs whose two questions are within ten tokens of each other in length.

run	all pairs	length matched
32B, final checkpoint	$+0.031$ $[+0.001, +0.063]$	$+0.009$ $[-0.029, +0.044]$
32B, lowest imitation loss	$+0.016$ $[0.000, +0.034]$	$+0.003$ $[-0.023, +0.029]$
8B, same recipe	$+0.043$ $[+0.012, +0.074]$	$+0.059$ $[+0.017, +0.096]$

A second family, imitation only. Three Granite 3.3 8B seeds trained with the identical imitation recipe gain on the held-out split at equal spend against the same model, prompted, by $+0.038$ to $+0.049$ on MuSiQue and $+0.053$ to $+0.061$ on 2WikiMultiHopQA, every interval excluding zero. On StrategyQA one seed's interval spans zero and the other two exclude it narrowly, none turning negative. At each arm's own stop under the shared cap of eight, all three Granite seeds read below the same model, prompted, on all three suites. Granite has no preference-trained checkpoint, so this is a first-rung reading, not a family contrast, and a mixture-of-experts base cannot hold the recipe, since its fused experts leave the adapter four of the seven modules. With every preference pair from one prompted questioner, a family ladder could show that the recipe carries over, not that it carries over with family-native preference data, which has not been run.

K Prompt search over the questioner

This appendix reports the prompt search over the questioner to which Section 7 points.

Two searches found nothing better than the shipped template. Reflective prompt evolution (Agrawal et al., 2026) was run over the questioner's template before anything was trained, scoring every candidate by rolling out the full episode on multi-hop training tasks. The first search reported an apparent gain of 87% on its tie-break slice, but that slice also selected the winner, and on 24 fresh tasks the gain fell to $+0.027$ at $p = 0.49$. What it had optimized was length. Across all 25 scored candidates the template ran from 425 to 683 tokens, and only the shipped template fit the budget within which arms are compared. A length control that keeps the shipped content and pads a reserved block with inert text to the winner's length reproduced the winner's entire advantage, the winner minus that control reading $+0.0032$ at $p = 0.96$. The second search used disjoint search, selection and verdict slices and an adoption rule committed before its winner existed. On the verdict slice its winner reads -0.0613 $[-0.176, +0.054]$ against the shipped template and -0.1394 $[-0.224, -0.055]$ against the length control, winning 3 of 36, in a design whose minimum detectable difference was 0.1585. The best prompt here is the one the work started with, and the headroom is in what the questioner learns to ask rather than in how it is asked to ask. Two lessons carry beyond this work: a slice that also selects the winner is not held out, and a search over an arm that will later be compared to ablations must carry the length budget as a constraint rather than as a check afterward.

L The run set aside, and four complete runs

The recipe's final preference stage was trained three times. One run, the first registered, resumed that stage from a checkpoint, and a trainer defect reloaded the stage's starting weights while keeping its step count, so it trained for about the last tenth of the stage. In the space the adapter adds to the model its weights sit 0.007 from the stage's starting point, against 0.080 for its own last checkpoint before the resume and for the other two runs. It was set aside on that measurement before any outcome of the other two was read, and the paper reports the other two. It is also the run the configuration was

selected on: on development it kept stopping when done above the selection floor, which the two reported runs miss (Appendix D).

It behaves like the question-pair arm it started from, and the difference is what the final stage buys. It asks 2.6, 1.6 and 2.0 questions on the three held-out suites where the trained questioner asks 4.2, 3.0 and 2.3, finishes in under three questions on 57%, 92% and 80% of runs, and asks at an unfinished state less often than the same model, prompted, by 0.087, 0.196 and 0.032, where the trained questioner is within 0.07 of it. Its coverage gain at equal spend is nearly the trained questioner's, +0.108, +0.060 and +0.049, and at the shared cap of eight it loses on StrategyQA, by −0.099 [−0.132, −0.070], where the trained questioner does not. On the constructed corpora it lost breadth at the widest shape, by −0.213, which the trained questioner does not.

Four complete runs. The final stage was trained twice more from the start after this run was set aside, once with the same seed and once with seed 3, from the same starting adapter, preference file and configuration, and checks declared before either was read held for both: neither resumed, both ran all 1,968 steps, and their weights sit from the stage's starting point inside the band of 0.07 to 0.09 declared around the two reported runs. Read under the rule and comparator of Table 2, every run's coverage interval lies above zero on every suite (Table 26). The runs differ by less than half of each run's own sampling error, no two of them separate, and pooled over the four the gain is +0.113, +0.070 and +0.053, each interval excluding zero at 50,000 resamples under three seeds. On the other metrics of Table 2 every run keeps the sign of each cell that table decides, and the out-of-order rate stays undecided for every run on the two suites where it is undecided there. The reading is exploratory, declared before either new run finished, and the reported questioner remains the average of seeds 1 and 2.

Table 26: Coverage at equal spend for the four complete runs of the final stage, each against the same model, prompted, with both arms at the lower of their two question counts, 200 held-out tasks per suite. Seeds 1 and 2 are the runs Table 2 averages, and seed 3 and the retrained seed 0 were trained from the start afterwards. Beneath each point is its 95% interval at 10,000 resamples. The pooled column averages the four runs within each task. The runs differ by a between-run standard deviation of 0.005, 0.007 and 0.004 on the three suites, against a mean within-run bootstrap standard error of 0.015, 0.017 and 0.018.

	seed 1	seed 2	seed 3	seed 0, retrained	four runs pooled
MuSiQue	+0.108	+0.116	+0.118	+0.109	+0.113
	[+0.081, +0.141]	[+0.090, +0.148]	[+0.092, +0.148]	[+0.082, +0.139]	[+0.088, +0.142]
StrategyQA	+0.064	+0.076	+0.077	+0.064	+0.070
	[+0.032, +0.099]	[+0.045, +0.110]	[+0.046, +0.111]	[+0.031, +0.099]	[+0.041, +0.102]
2WikiMultiHopQA	+0.048	+0.051	+0.057	+0.055	+0.053
	[+0.015, +0.084]	[+0.019, +0.085]	[+0.026, +0.091]	[+0.021, +0.093]	[+0.021, +0.087]

M FURTHER RELATED WORK

This section extends Section 6 with work that bears on this paper's design but is not needed to follow the main text.

Asking for missing information. Preference optimization toward questions of high expected information gain teaches models to ask informative questions (Mazzaccara et al., 2024), language models can be taught to gather information proactively (Huang et al., 2025), and they can be augmented to ask clarification questions for retrieval (Chi et al., 2024). Reinforcement learning for active reasoning can suffer from what Zou et al. (2026) call information self-locking, and agents that explore can still ignore what they find (Engländer et al., 2026). Long-horizon proactive search has its own benchmark (Inc, 2026), and ambient agents act on event streams without waiting for a prompt (LangChain, 2025). Q&D differs in its label: each question is credited with the evidence its continuation actually retrieved.

Deep-research and search agents. Reasoning models have been extended into deep-research agents that explore the web (Li et al., 2025b), agents have been trained by self-play without supervision or external data (Lu et al., 2026; Zhang et al., 2026b), and deep research and everyday

online tasks have open evaluation sandboxes and benchmarks (Coelho et al., 2026; Zhang et al., 2026c). Reinforcement learning with verifiable rewards may sharpen rather than extend what the base model can reach (Yue et al., 2025), and reported gains can reflect contamination (Wu et al., 2026b), one reason the held-out runs here are checked against the training split (Appendix E).

Preference optimization with noisy labels. Labels read from sampled continuations are noisy. Alternatives to direct preference optimization change its objective (Azar et al., 2024; Ethayarajh et al., 2024; Meng et al., 2024), and a provably robust variant targets noisy preferences (Ray Chowdhury et al., 2024). We kept standard direct preference optimization and did not compare these variants.

Agreement and evaluation statistics. Disagreement among annotators can be inherent rather than noise (Pavlick & Kwiatkowski, 2019; Plank, 2022), which is how we read the raters' low agreement on the better proactive move (Appendix I), and chance-corrected agreement for such data is surveyed by Artstein & Poesio (2008). Language-model judges have reliability and bias problems of their own (Yagubyan, 2026). Evaluations of language models need error bars and paired comparisons (Miller, 2024), small improvements call for a paired bootstrap (Du, 2026), and our decided rule guards against resampling error rather than controlling a false discovery rate (Benjamini & Hochberg, 1995). Agentic benchmarks need explicit checks that a task measures what it claims and that its outcome is scored correctly (Zhu et al., 2025).

N PROMPTS AND THE GOLD FIREWALL

This appendix supports the rule of Section 2 that nothing on the rollout path may read a need graph, and reproduces the template that is the policy of Section 3. Four layers enforce the rule, each catching a different class of leak.

An import contract. A forbidden-import contract, checked on every continuous integration run, prevents any package on the rollout or training path from importing the gold-reading package. It catches a reference to gold written directly into code that runs during a rollout.

A type wall. The task view shares no field name and no base class with a gold node, so a leak by assignment does not typecheck, and its one constructor raises on an unknown field name and on a known field whose value is not a string, so gold cannot reach a prompt inside a permitted field by stringification. It catches code that never imports gold but is handed it.

A process split. A rollout worker runs without the environment variable that names the gold root, so the accessor raises when anything calls it, and that raise is the layer working. It catches a read attempted at run time by code that got past the first two layers.

A marker in one gold string. The first three layers govern code, and a gold string already copied into a prompt is not code. So a marker token is planted in one gold string, and every serialized request is scanned for it inside the client immediately before dispatch, where the request still exists and has not been sent, since a cached record keeps only a digest of it. A hit voids the run, and a worker refuses to begin when gold is built but the marker registry cannot be found, because an unarmed scanner reports nothing, exactly as a clean one does.

Coverage of the fourth layer is per suite by design. The marker rides in the gold answer, and every question-answering suite carries it, while the five tool-using and transfer suites have no gold answer and their need text legitimately reaches judges and two ceiling arms, so on those suites a leak through a string is caught by no layer. What can be bounded there is verbatim echo of secret need text in recorded replies: on banking and one other of those five suites a scan of $27{,}465{,}239$ characters over $2{,}053$ runs found no match, with the same scanner shown to fire once the secrecy exclusion is removed, while on airline and retail every need string already occurs in the domain text the agent is handed, so no content bound is available there, and nothing bounds paraphrase, or gold entering a prompt, on any suite.

The questioner runs one template, rendered fresh at every turn. There is no separate system message: the policy renders the whole template from the current state and dispatches it as a single user turn. The template takes no budget and no cap, and the renderer refuses any placeholder that would name one, which keeps the policy budget-blind and a prefix of a long rollout exchangeable with a true short run. The reserved block is the one the length control in Appendix K extends.

```
You are the Inquirer. A Drafter is preparing an answer to the task below and you interrogate
it, one internal question at a time, until the answer would not change if you asked again.
You are an internal component. You never address the end user and you never write the answer
yourself. Your only output is the next question, or a decision to stop.

TASK
{{question}}

SUITE INSTRUCTIONS
{{instructions}}

EVIDENCE RETRIEVED SO FAR
{{evidence}}

CURRENT DRAFT
{{draft}}

QUESTIONS ALREADY ASKED, AND WHAT CAME BACK
{{history}}

HOW TO CHOOSE
Ask about a need the task requires and the evidence above does not yet resolve. A need is
latent when the task does not name it: it becomes visible only once earlier evidence has
been read. Prefer such a need over one already named in the task statement, and prefer
either over a rephrasing of a question already asked. Name the entity you are asking about
explicitly, because the retriever matches text and cannot resolve a pronoun. Cite in
parent_uids the units whose content made you aware of this need, or leave it empty if the
need is stated in the task itself.

WHEN TO STOP
Stop when every need the task requires is resolved by the evidence above, or when the next
question you can think of would return something the evidence already contains. Stopping is
a decision on the same footing as asking, not a failure.

{{user_channel}}

RESERVED
This block is reserved. It appears in every condition so that the amount of text surrounding the
task is the same in each. It states no requirement and expresses no preference. It carries no
fact about the task, about the evidence or about the answer. It is not a hint, and nothing in it
should be treated as one. It describes neither what to ask nor when to stop. It names no entity
and refers to no document.

OUTPUT
Reply with exactly one JSON object and no other text, no markdown fence, no commentary:
{"action": "ask", "question": "<a single question>", "rationale": "<one clause>",
 "parent_uids": ["<uid>"]}
or
{"action": "stop", "rationale": "<one clause>"}
```

The template opens with “You are the Inquirer” (Inquirer is the template’s name for the questioner, and Drafter its name for the drafter). The output contract is one line in the file, wrapped here at the comma before `parent_uids` to fit the column, and nothing else in the block is altered. The user-channel hole takes a placebo fragment in every arm but one, so opening the channel changes what the prompt says and not how long it is.

```
A NOTE ON THIS SECTION
This paragraph is reserved. It appears in every one of the conditions so that the surrounding
text has the same size in each, and it carries no instruction, no preference and no
information about the task, the evidence or the answer.

THE USER CHANNEL IS OPEN
You may address one question to the user instead of the knowledge base by setting
"target": "user". Use it only for a need no document can resolve, such as a preference or a
private constraint held by the user alone.
```