

Don't Guess, Ask Wisely: Benchmarking and Training Proactive Elicitation in Open-Domain LLM Conversations

Rongyu Zhang^{1*}, Mutian Bao^{1*}, Jinjian Zhang^{2*}, Yuetian Du¹, He Xu¹, Wei Zhou²,
Xiaojun Zhu¹, Ming Kong¹, Jie Liu^{3†}, Linjian Mo^{2†}, Qiang Zhu^{1†}

¹Zhejiang University ²Ant Group ³City University of Hong Kong
{rongyu.zhang, zhuq}@zju.edu.cn

Abstract

When user queries are incomplete, large language models should ask targeted questions before committing to an answer, a capability we call proactive elicitation, yet evaluation on real user-assistant queries across seven LLMs (8B to frontier) shows that fewer than 11% of underspecified queries trigger a follow-up question. To benchmark and train this capability, we introduce VagueBench, where each seed query is annotated with structured intent and detail units then degraded into intent-vague and fully-vague variants via controlled removal of linguistic cues, yielding known hidden information sets for evaluating both when and what to ask. We propose DialogGRPO, an offline reinforcement learning method that organizes multi-turn elicitation as a dialogue-state directed acyclic graph, where candidate questions at each node receive dense turn-level rewards combining immediate information gain with lookahead value through group-relative comparison, enabling fine-grained credit assignment without runtime simulation. DialogGRPO elicits missing information on over 88% of vague queries with substantial gains in both recovery efficiency and answer coverage, without over-asking on complete queries, outperforms prompting and test-time-scaling baselines, and generalizes zero-shot to MultiWOZ 2.4 and AmbigQA.

1 Introduction

Today's large language models (LLMs) default to *answering*, not *asking*. When user queries are incomplete or underspecified, models silently infer unstated intent and constraints instead of asking targeted questions to resolve the gap (Figure 1). Yet real-world user queries are rarely self-contained:

*Equal contribution.

†Corresponding authors.

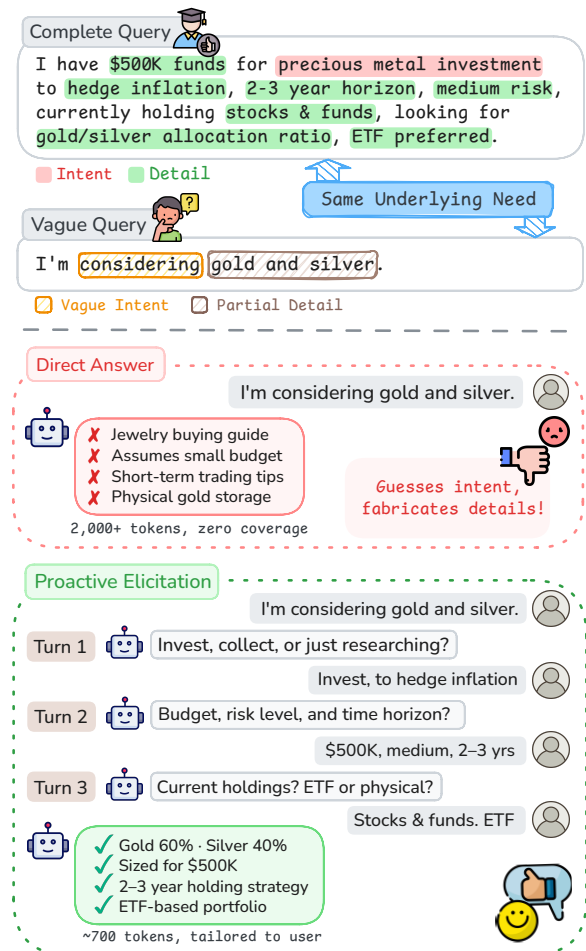


Figure 1: **Top:** The same underlying need expressed as a complete query and a fully-vague query. **Bottom:** A direct-answer LLM fabricates all missing information, while proactive elicitation recovers it in 3 targeted turns.

information gaps are pervasive in natural human communication (Taylor, 1968), and users routinely issue underspecified queries, overestimating the model's ability to infer unstated needs: task accuracy drops by 39% when instructions are fragmented (Laban et al., 2025), and medical diagnostic accuracy plummets from 94.9% to 34.5% when

symptoms are underspecified (Bean et al., 2026).

Our analysis of 5,000 real-world queries from a commercial AI assistant further quantifies this gap: *over half of user queries lack critical information, yet models from 8B to over 120B ask clarification in fewer than 11% of such cases*, with no clear improvement from scaling. These findings motivate *proactive elicitation*: identifying missing information and asking targeted questions *before* committing to an answer.

This capability is challenging because the information gap arises from both sides of the conversation. On the **user side**, users often leave critical constraints implicit, assuming the model shares context it does not have, a cognitive bias known as the *curse of knowledge* (Camerer et al., 1989). On the **model side**, post-training pipelines dominated by single-turn supervision and rewards for direct answers implicitly penalize pausing to ask. We formalize this gap at two levels: (1) **Intent ambiguity**, where the user’s overarching goal is unclear—e.g., whether “help me work on this text” means translation or summarization; and (2) **Detail absence**, where the goal is clear but critical parameters are missing—e.g., budget or use case in “recommend a laptop”. These two levels require different elicitation strategies and evaluation criteria.

Recent work has explored LLM clarification behavior (Shaikh et al., 2025; Zhang et al., 2025; Wu et al., 2025; Andukuri et al., 2024), but remains limited along three axes. **Scope**: prior work mostly targets *disambiguation*, resolving queries with multiple valid interpretations (Min et al., 2020; Zhang et al., 2024), rather than the broader case where essential information was *never expressed* at all. **Data**: benchmarks are confined to narrow domains (Gan et al., 2024) or synthetic ambiguity (Luo et al., 2025), with no training-ready resource grounded in real conversations. **Training**: methods rely on online user simulators or trajectory-level feedback (Wu et al., 2025; Huang et al., 2025; Andukuri et al., 2024; Zhang et al., 2025), limiting fine-grained, per-step credit assignment.

To address these gaps, we propose a unified framework for evaluating and training proactive elicitation, grounded in real user–model conversations. On the evaluation side, we introduce **VagueBench**, a benchmark built from real open-domain queries through *controlled degradation*: intent-bearing linguistic cues are systematically weakened and task details removed, yielding vague variants whose hidden information set is precisely

known. A simulated user holds the complete query and reveals a hidden information unit only when the model asks a semantically matching question, providing a controlled, reproducible testbed for multi-turn elicitation. We measure three complementary aspects: *decision appropriateness*, whether the model asks rather than prematurely answers; *elicitation efficiency*, the number of hidden information units recovered per turn; and *answer coverage*, the F1 between the final response and the gold information set. On the training side, we propose **DialogGRPO**, an offline reinforcement learning framework that models multi-turn elicitation as a dialogue-state directed acyclic graph (DAG). Each node groups alternative elicitation actions for local GRPO (Shao et al., 2024) comparison, and each edge receives a dense reward combining immediate information gain with long-term lookahead value, enabling the model to learn *when* and *what* to ask without the prohibitive cost of runtime user simulation.

Our contributions are threefold:¹

- **Benchmark.** We introduce VagueBench, a benchmark built from real open-domain queries with controlled vagueness and known hidden information for evaluating proactive elicitation capabilities of LLMs.
- **Method.** We propose DialogGRPO, which models multi-turn elicitation as a dialogue-state DAG and applies group-relative advantage estimation at each state for dense offline credit assignment.
- **Findings.** Evaluation on VagueBench across open-source and closed-source LLMs (8B to frontier) reveals systematic elicitation suppression at all scales. DialogGRPO outperforms prompt-engineering and test-time-scaling baselines on all metrics, and zero-shot transfers to MultiWOZ 2.4 and AmbigQA.

2 Related Work

2.1 Proactive Elicitation

Clarification and mixed-initiative interaction are well studied in conversational search and task-oriented dialogue (Budzianowski et al., 2018; Ye et al., 2022; Gan et al., 2024), where systems ask follow-up questions to resolve facets, slots, or preferences (Aliannejadi et al., 2019, 2020). For LLM assistants, the challenge is broader: open-ended requests often omit intent or constraints, yet

¹Code and data are available at <https://github.com/zhangrongyu0101/VagueBench>.

instruction-tuned models tend to answer directly and fill gaps by assumption (Yang et al., 2025; Laban et al., 2025). Recent LLM work explores clarification via prompting, grounding analysis, and collaborative agents (Shaikh et al., 2025; Wu et al., 2025), but mainly targets multiple valid interpretations. We instead study open-domain elicitation of unspoken intent and task details.

2.2 Vagueness Benchmarks

Existing benchmarks capture different forms of underspecification. Qulac and ClariQ evaluate clarification in conversational search (Aliannejadi et al., 2019, 2020); AmbigQA and related datasets target questions with multiple valid interpretations (Min et al., 2020; Zhang et al., 2024); recent LLM benchmarks further test clarification in task-oriented dialogue, tool use, or synthetic ambiguity settings (Zhang et al., 2024; Gan et al., 2024; Luo et al., 2025). However, they assume a predefined domain, a fixed slot schema, or a small set of alternative interpretations, providing limited coverage of real open-domain vagueness where intent and task details may have never been stated. VagueBench fills this gap with real assistant queries, unrestricted intents, and a multi-turn evaluation protocol.

2.3 Training Elicitation Policies

Prior methods improve clarification behavior through prompting, supervised fine-tuning, preference learning, simulated users, or downstream task rewards (Chen et al., 2025; Andukuri et al., 2024; Zhang et al., 2025; Huang et al., 2025). Table 1 compares these methods by whether they target open-domain interactions, use real user queries, avoid online user simulation, and provide turn-level rewards (Wu et al., 2025; Shani et al., 2024). While effective, existing approaches often depend on synthetic interactions, domain-specific settings, or coarse dialogue-level feedback. DialogGRPO instead precomputes a dialogue-state DAG and rewards each asking turn using both acquired information and downstream answer value.

3 VagueBench

A user submits a query q_0 that may express only part of a complete need q^* . We decompose q^* into an *intent set* $\mathcal{T} = \{I_j\}_{j=1}^m$ of task-level goals and a *detail set* $\mathcal{D} = \{d_i\}_{i=1}^n$ of key-value constraints for executing those goals; the gap between q_0 and q^* may involve *intent ambiguity*, *detail absence*, or both, as defined in §1. Each benchmark instance

Method	No Runtime Sim.	Open Domain	Turn Reward	Real Queries
ClarifyGPT	✓	✗	✗	✗
ACT	✗	✗	✗	✗
STaR-GATE	✓	✓	✗	✗
CollabLLM	✗	✓	✓	✗
Ours	✓	✓	✓	✓

Table 1: Comparison of proactive elicitation methods. No Runtime Sim.: the parameter-update phase does not invoke an LLM-based user simulator at runtime. Turn Reward: per-turn rather than trajectory-level credit assignment.

begins with an information-complete seed whose $(\mathcal{T}, \mathcal{D})$ is fully annotated, from which we derive an *intent-vague* and a *fully-vague* variant by progressively removing information units, so every gap is precisely quantifiable (Figure 2).

3.1 Data Collection

We source over 100,000 open-domain queries from a commercial AI assistant.² A rule-based pass enforces length bounds and removes unsafe content; two LLM passes then discard greetings, test strings, context-dependent utterances, and queries too detail-sparse for meaningful degradation. The surviving 4,930 seeds ($\approx 5\%$) span five task families: *Information Query*, *Recommendation & Decision*, *Task Execution*, *Creation & Generation*, and *Consultation & Analysis*, with the distribution shown in Figure 6. Each seed carries on average $\bar{m} = 1.36$ intents and $\bar{n} = 4.63$ details. Filtering rules and full statistics are in Appendix A.

3.2 Data Construction

The annotation scheme and variant construction share a single backbone: *clarity anchors*, recurring linguistic cues that make a query’s information explicit.

Clarity anchors. Drawing on argument-structure theory (Levin and Hovav, 2005) and speech-act theory (Searle, 1969), we identify four anchor types in two layers. Consider the query “Help me summarize this article into five bullet points focusing on the economic implications” as a running example, where solid and dashed underlines mark intent-layer and detail-layer anchors, respectively. The **intent layer** signals what the user wants done: *predicate anchors* are action verbs that pin

²From users who opted into data collection policy, with PII removed. See the Ethics Considerations.

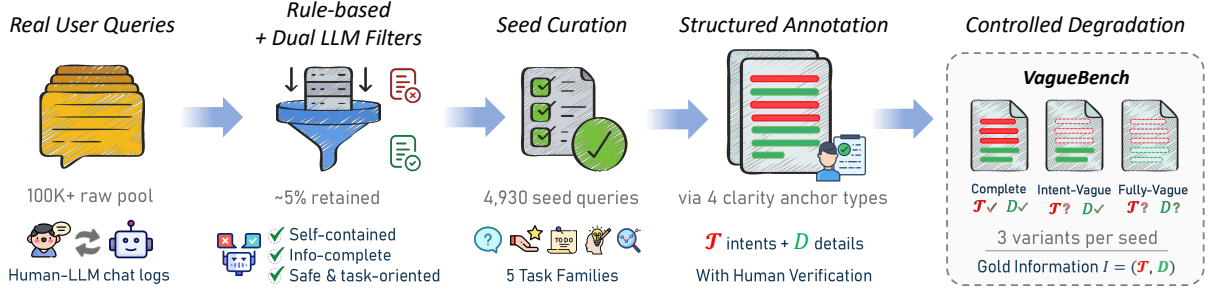


Figure 2: VagueBench construction pipeline. Real user queries are filtered for self-containedness and information completeness, annotated with intent and detail units via four clarity anchor types, and then systematically degraded into intent-vague and fully-vague variants with known gold information for evaluation.

the task type (*summarize*); *speech-act anchors* are directive frames (*Help me...*). The **detail layer** specifies how the task should be executed: *output-type anchors* mark the expected answer form (*five bullet points*); *scope anchors* bound response dimensions (*focusing on economic implications*). This partition determines both what to annotate and what to remove.

Annotation. The anchor taxonomy makes annotation straightforward: intent-layer anchors are grouped into task-level goals $\mathcal{T}$, while detail-layer anchors and remaining surface constraints become key-value pairs in $\mathcal{D}$, each linked to the intent it serves. In the running example, the predicate anchor *summarize* and the speech-act anchor *Help me...* together identify a single summarization intent. On the detail side, the output-type anchor *five bullet points* yields a key-value pair $\langle \text{format}, \text{five bullet points} \rangle$, and the scope anchor *focusing on the economic implications* yields $\langle \text{focus}, \text{economic implications} \rangle$; both are linked to that summarization intent. Two frontier LLMs (GLM-5) annotate each seed independently; disagreements are resolved by a third LLM (GPT-5.4) acting as adjudicator, and a human validation study (Appendix B.2) confirms strong agreement between LLM and human annotations ($\kappa \geq 0.81$).

Controlled degradation. Because anchors are partitioned into two layers, variant construction is a principled removal procedure. An *intent-vague* variant weakens or removes only intent-layer anchors while preserving every detail unit in $\mathcal{D}$. A *fully-vague* variant further strips detail-layer anchors and a subset of detail units, retaining a core-topic phrase of typically 5–10 tokens. For example, the complete query “*Help me summarize this article into five bullet points focusing on the economic implications*” reduces to

“*Work on this article, five bullet points, focusing on the economic implications*” at the intent-vague level, and to “*Help me work on this article*” at the fully-vague level. Both variants undergo an LLM fluency rewrite; hidden-unit labels are verified by set difference against the gold annotation, and non-conforming cases are regenerated (Appendix B.3).

3.3 Evaluation

Interaction. A conversation begins with query q_0 . At each turn the model either asks a question or commits to a final answer; the dialogue ends when the model answers, all hidden units are recovered, or a budget of $T_{\max}$ turns is reached. A simulated user $\mathcal{S}$, holding q^* and the hidden units, reveals a unit only when the model’s question semantically matches it, releases no more than the matched units per turn, and responds with a non-informative deflection to irrelevant or repeated questions. Details of simulated user is provided in Appendix C.

Metrics. *Decision appropriateness* (Ask%): the first LLM output is classified as *ask* or *answer*; the gold label is *ask* for vague variants and *answer* for complete queries.

Elicitation efficiency (IAR): the information acquisition rate measures hidden units recovered per elicitation turn, computed only when the model chose to ask:

$$\text{IAR} = \frac{|\mathcal{T}_{\text{acq}}| + |\mathcal{D}_{\text{acq}}|}{\max(T, 1)}. \quad (1)$$

Answer coverage (Cov. F1): the final answer is compared against the gold information set via F1 score. It is formulated as:

$$\text{Cov. F1} = \frac{2P_{\text{cov}}R_{\text{cov}}}{P_{\text{cov}} + R_{\text{cov}}} \quad (2)$$

where precision P_{cov} penalizes unsupported assumptions, and recall R_{cov} penalizes omissions.

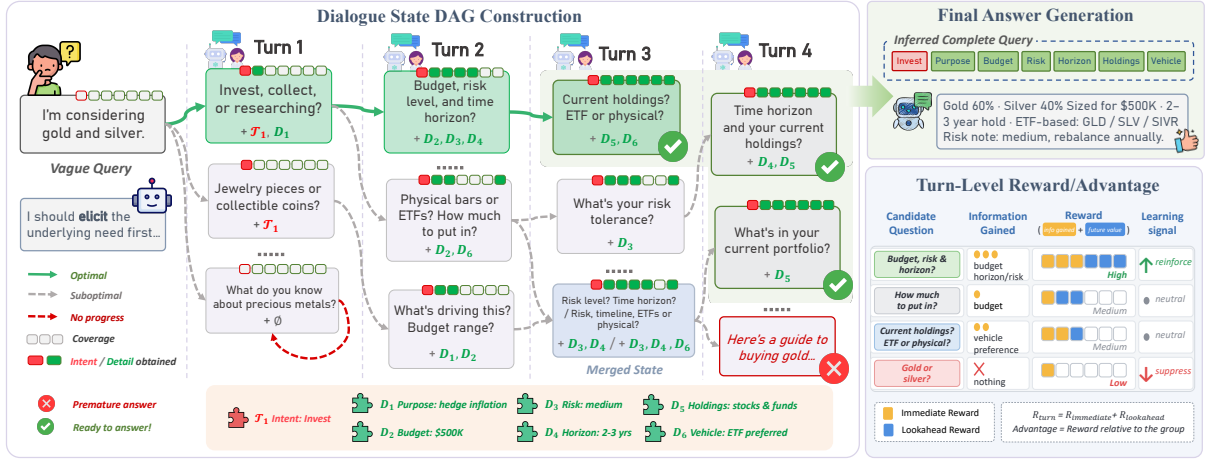


Figure 3: DialogGRPO pipeline. **Left:** A dialogue-state DAG is built by expanding candidate questions at each turn and merging nodes with identical acquired sets. **Right:** Recovered units ground the final answer (top); turn-level rewards and group-relative advantages guide training (bottom).

4 DialogGRPO

Training proactive elicitation faces two coupled challenges. First, *sparse credit assignment*: multi-turn rewards arrive only at the trajectory’s end, leaving each intermediate question without direct feedback on whether it was worth asking. Second, *simulation cost*: on-policy rollouts require invoking an LLM-based user simulator at every gradient step, which is prohibitively expensive when each dialogue spans multiple turns.

DialogGRPO addresses both through a unified offline pipeline organized in three stages (Figure 3). We first construct a *dialogue state DAG* (§4.1) that maps out candidate elicitation questions and their consequences using an external LLM, so that the outgoing edges of any node form a natural GRPO comparison group. We then assign each edge a *turn-level reward* (§4.2) that combines immediate information gain with lookahead value estimated from the pre-built graph, converting the single end-of-trajectory signal into dense per-action feedback. Finally, we train the policy via *advantage-weighted learning* (§4.3) over (prompt, candidate, advantage) tuples extracted from the DAG, requiring no runtime simulation.

4.1 Dialogue State DAG

Nodes and termination. For a training query with information units $U^* = \mathcal{T} \cup \mathcal{D}$, each node v represents the acquired set $K_v \subseteq U^*$, initialized at the root with units already present in q_0 . A node is terminal when all units are acquired, an LLM answerability classifier deems K_v sufficient for a satisfactory response, or the depth reaches $D_{\max}$.

The classifier is a prompted LLM conditioned on q_0 , K_v , and the dialogue history (Appendix J).

Oracle-guided expansion. At each non-terminal node v , an external strong LLM (GLM-5.1), distinct from the policy model π_θ , proposes N candidate questions conditioned on q_0 , the dialogue history reaching v , and the remaining gold units $U^* \setminus K_v$. The policy model never observes these gold units; the external LLM serves solely as an *exploration engine* that populates diverse candidates. The policy learns to distinguish high-value questions through the group-relative advantage signal (§4.3), rather than through behavioral cloning or distillation of the external model.

For each candidate question c , the simulated user $\mathcal{S}$ produces a reply u_c , and an extraction function $\text{Ext}(\cdot)$ maps the reply to a set of information units. The *information gain* of edge $e = (v, c)$ is:

$$\Delta_e = (\text{Ext}(u_c) \cap U^*) \setminus K_v. \quad (3)$$

If $\Delta_e \neq \emptyset$, the edge leads to a child node v' with $K_{v'} = K_v \cup \Delta_e$, and expansion continues from v' . Candidates that acquire no new unit are recorded as *null-progress* actions: they enter the comparison group at v as negative examples but are not expanded further.

State merging. Different elicitation orders can reach the same acquired set. We merge nodes sharing identical K_v , converting the expansion tree into a compact DAG. Merging avoids redundant expansion of duplicate states and allows reward computations to be shared across paths that converge on the same node. When multiple dialogue

histories converge at a merged node, we retain the shortest-path history as the *canonical prompt* for that state; §4.3 describes how this interacts with training-tuple extraction.

4.2 Turn-Level Reward

A root-to-terminal path $\tau = (e_1, \dots, e_L)$ has a natural trajectory-level reward:

$$R_{\text{traj}}(\tau) = F_1(v_L) - \lambda L, \quad (4)$$

where $F_1(v_L)$ is the answer-coverage F1 at the terminal node and λ penalizes interaction length. However, this signal is available only at trajectory endpoints and cannot distinguish a good intermediate question from a bad one. We therefore design two complementary per-edge signals.

Immediate information gain. Each edge e receives:

$$R_{\text{imm}}(e) = \alpha \min(|\Delta_e|, \delta_{\max}), \quad (5)$$

rewarding questions that elicit more units per turn. The cap $\delta_{\max}$ prevents a single high-gain question from dominating the advantage distribution, keeping the reward discriminative across candidates of varying quality. Null-progress edges receive $R_{\text{imm}} = 0$.

Lookahead value. The immediate reward is a *single-step* signal that ignores downstream consequences. We complement it with a lookahead value estimated via Monte Carlo rollouts in the pre-built DAG: for each edge e , we sample K paths from its child state v_e and average their trajectory returns:

$$Q_{\text{la}}(e) = \frac{1}{K} \sum_{k=1}^K R_{\text{traj}}(\tau_k \mid v_e), \quad (6)$$

where v_e is the child state reached by e . For terminal v_e we set $Q_{\text{la}}(e) = F_1(v_e)$ directly.

Combined reward. The turn-level reward combines both signals:

$$R_{\text{turn}}(e) = Q_{\text{la}}(e) + R_{\text{imm}}(e). \quad (7)$$

For null-progress edges, we set $R_{\text{turn}} = \bar{Q}_{\text{la}}(v) - \lambda$, where $\bar{Q}_{\text{la}}(v)$ is the mean lookahead at the parent node, reflecting that a null-progress question wastes one turn without narrowing the information gap between user and assistant.

4.3 Advantage-Weighted Training

Comparison groups and advantage estimation.

For each non-terminal node v , the candidate set $\mathcal{G}_v = \{e : \text{src}(e) = v\}$ forms a GRPO comparison group. A group contains different productive questions. Following the group-relative principle of GRPO (Shao et al., 2024), we compute a normalized advantage:

$$\hat{A}_e = \frac{R_{\text{turn}}(e) - \mu_v}{\sigma_v + \epsilon}, \quad (8)$$

where μ_v and σ_v are the within-group mean and standard deviation of R_{turn} .

Training objective. Each training instance is a tuple $(p_v, y_e, \hat{A}_e)$, where p_v is the canonical prompt at node v (§4.1) and y_e is the candidate response associated with edge e . We optimize:

$$\mathcal{J}(\theta) = \mathbb{E}_v \left[\frac{1}{|\mathcal{G}_v|} \sum_{e \in \mathcal{G}_v} \exp\left(\frac{\hat{A}_e}{\beta}\right) \log \pi_\theta(y_e \mid p_v) \right] \quad (9)$$

where β controls the strength of the KL penalty toward the reference policy π_{ref} . Because all candidates are pre-generated and fixed in our offline setting, there are no importance-sampling ratios or clipping (Schulman et al., 2017), and the objective reduces to *advantage-weighted regression* (Peng et al., 2019). A full derivation from the KL-regularized policy optimization objective to Eq. 9 is provided in Appendix H.

5 Experiments

5.1 Setup

The VagueBench test set comprises 250 complete queries and their *intent-vague* and *fully-vague* variants (750 instances); the remaining 4,680 seeds and their DAG expansions form the training set. For cross-dataset generalization we report on MultiWOZ 2.4 (Ye et al., 2022) and AmbigQA (Min et al., 2020), neither seen during training. We evaluate seven LLMs spanning open-source (Qwen3-8B, Llama-3.1-8B-Instruct, Qwen3.5-27B, GPT-oss-120B) and closed-source (GPT-5.4, Claude Sonnet 4.6, Gemini 3.1 Pro) families. Using Qwen3.5-27B as the backbone, we compare four methods: **Prompt**, an elicitation-encouraging system prompt; **UoT** (Hu et al., 2024), a test-time-scaling approach; **SFT**, trained on highest-reward DAG trajectories; and our **DialogGRPO**. All dialogues are capped at $T_{\text{max}}=10$ turns. Implementation details are in Appendix D.

Model	Intent-Vague			Fully-Vague		
	Ask% ($\uparrow$)	Cov. F1 ($\uparrow$)	IAR ($\uparrow$)	Ask% ($\uparrow$)	Cov. F1 ($\uparrow$)	IAR ($\uparrow$)
<i>Closed-source Models</i>						
GPT-5.4	1.6	0.58	0.06	10.8	0.53	0.65
Claude Sonnet 4.6	8.8	<u>0.60</u>	0.08	8.4	0.52	0.56
Gemini 3.1 Pro	2.4	0.57	0.00	3.6	0.54	<u>0.80</u>
<i>Open-source Models</i>						
Qwen3-8B	1.2	0.47	0.00	1.6	0.36	0.21
Llama-3.1-8B-Instruct	0.8	0.50	0.00	1.2	0.35	0.23
Qwen3.5-27B	1.6	0.49	0.08	2.0	0.38	0.37
GPT-oss-120B	2.0	0.46	0.02	3.6	0.36	0.35
<i>Methods (built on Qwen3.5-27B)</i>						
+ Prompt	38.4	0.52	0.22	47.2	0.50	0.38
+ UoT	58.8	0.50	0.45	62.4	0.49	0.41
+ SFT (Ours)	<u>82.0</u>	<u>0.60</u>	<u>0.64</u>	<u>85.6</u>	<u>0.63</u>	<u>0.80</u>
+ DialogGRPO (Ours)	88.4	0.64	0.76	91.2	0.67	0.87

Table 2: Main results on VagueBench. *Ask%*: proportion of samples on which the model chose to ask. *Cov. F1*: coverage F1 of the final answer against the gold information set. *IAR*: information acquisition rate (Eq. 1), computed only on samples where the model asked. Best in **bold**; second best underlined.

5.2 Main Results

Elicitation suppression persists across scales. As shown in Table 2, across all evaluated models, *Ask%* on *intent-vague* queries stays at or below 9% and on *fully-vague* queries never exceeds 11%. This suppression holds regardless of model family or size: the four open-source models achieve *Ask%* of only 0.8–2.0 on *intent-vague*, while the three closed-source frontier models range from 1.6 to 8.8, with larger models showing no consistent advantage. The shortfall suggests a post-training tendency to answer directly, rather than a capability limit that further scaling would fix.

Prompting reveals latent elicitation ability but quickly plateaus. A single elicitation-encouraging prompt lifts Qwen3.5-27B *Ask%* from 1.6% to 38.4% on *intent-vague* and from 2.0% to 47.2% on *fully-vague*, confirming that the ability resides in the pretrained weights, suppressed rather than absent. However, the prompted model asks more often without asking more efficiently: *IAR* stays at 0.22/0.38, far below the levels that training-based methods reach. Prompting can shift the ask-versus-answer decision boundary but provides no signal for improving per-turn elicitation efficiency, motivating an approach that optimizes both *when* and *what* to ask.

DialogGRPO asks better, not just more often. Compared with SFT trained on the highest-reward trajectories, DialogGRPO further improves *Ask%*

Model	Ask% ($\downarrow$)	Cov. F1 ($\uparrow$)
Qwen3.5-27B	1.6	0.58
+ Prompt	42.8	0.59
+ UoT	56.4	0.48
+ SFT (Ours)	6.4	<u>0.71</u>
+ DialogGRPO (Ours)	<u>5.2</u>	0.74

Table 3: Behavior on information-complete seed queries. *Ask%* is lower-the-better. Best in **bold**; second best underlined.

by up to 6.4 points and *IAR* by 0.07–0.12, confirming that group-relative advantage estimation provides a finer learning signal than trajectory-level imitation. The Cov. F1 improvement over Qwen3.5-27B (0.64/0.67 vs. 0.49/0.38) directly reflects the value of elicitation: the model answers on *recovered* information rather than *assumed* defaults. UoT presents a different bottleneck: per-turn recovery is reasonable when it does ask (*IAR* 0.45/0.41), but its uncertainty heuristic triggers elicitation on only $\sim 60\%$ of vague queries, leaving substantial information unrecovered.

High elicitation on vague queries does not cause over-asking on complete ones. On information-complete queries (Table 3), DialogGRPO asks on only 5.2% of samples, close to Qwen3.5-27B’s 1.6%, while lifting Cov. F1 from 0.58 to 0.74. Prompt and UoT, by contrast, push *Ask%* to 42.8% and 56.4%, unnecessarily interrupting roughly half

Variant	Intent-Vague		Fully-Vague	
	Cov. F1	IAR	Cov. F1	IAR
Full (Ours)	0.64	0.76	0.67	0.87
w/o lookahead reward	0.58	<u>0.74</u>	0.60	<u>0.84</u>
w/o immediate reward	<u>0.60</u>	0.68	<u>0.66</u>	0.85

Table 4: Reward ablation on VagueBench. The lookahead term primarily affects Cov. F1; the immediate term primarily affects IAR. Best in **bold**; second best underlined.

of already well-specified queries. This calibration arises naturally from the group-relative advantage: when all candidate questions at a state yield uniformly low information gain, the advantage signal *suppresses* asking without requiring an explicit threshold.

General capability is preserved. Beyond elicitation metrics, we verify that DialogGRPO introduces no catastrophic forgetting relative to the untrained Qwen3.5-27B: on six benchmarks spanning coding, math, commonsense reasoning, and general knowledge, it matches or slightly exceeds the original model (Appendix E), indicating that offline advantage-weighted RL stays close enough to the reference policy to avoid capability regression.

5.3 Ablation Study

Table 4 ablates the two reward signals defined in §4.2. Without the lookahead term, Cov. F1 drops by 0.06–0.07 while IAR remains nearly intact, suggesting that the policy becomes myopic and favors locally greedy questions over those that lead to higher final answer coverage. Conversely, removing the immediate term leaves Cov. F1 largely unchanged but reduces IAR by up to 0.08, as the model loses the per-turn incentive to pack multiple information units into a single question. Neither ablated variant matches the full method, confirming that the two reward signals are complementary.

5.4 Generalization

Across datasets. To test whether the learned elicitation ability overfits to VagueBench, we evaluate zero-shot on two out-of-distribution benchmarks: MultiWOZ 2.4, a task-oriented slot-filling dialogue dataset, and AmbigQA, an ambiguous question-answering dataset. We train DialogGRPO on Qwen3.5-27B and Gemma3-27B and compare their performance against two frontier models (GPT-5.4 and GLM-5). For MultiWOZ, we fol-

Model	MultiWOZ 2.4	AmbigQA	
	Success%	Ask%	Ask F1%
GPT-5.4	39.9	5.1	9.6
GLM-5	23.8	0.9	1.9
Qwen3.5-27B	11.4	1.1	2.1
+ DialogGRPO (Ours)	<u>24.7</u>	<u>16.3</u>	<u>26.3</u>
Gemma3-27B	15.1	1.1	2.2
+ DialogGRPO (Ours)	17.8	34.4	44.7

Table 5: Zero-shot transfer. *Success%*: MultiWOZ task-completion rate. *Ask%*: fraction of queries on which the model asks in the AmbigQA *multipleQAs* subset. *Ask F1%*: ask-versus-answer F1 on the full AmbigQA development set. Best in **bold**; second best underlined.

low the zero-shot end-to-end simulated interaction protocol of Acikgoz et al. (2026). For AmbigQA, *Ask%* is the fraction of queries on which the model asks in the ambiguous *multipleQAs* subset. *Ask F1%* is computed on the full development set with *Ask* as the positive class, assigning gold *Ask* to ambiguous queries and gold *Answer* to unambiguous queries.

On AmbigQA, the benchmark most aligned with the elicitation capability we train, both models exhibit substantial gains (Table 5). Notably, Gemma3-27B is an instruction-tuned model while Qwen3.5-27B is a reasoning model, the elicitation behavior therefore transfers consistently across two distinct model types. On MultiWOZ, *Success%* is a task-completion metric bounded by the base model’s downstream execution rather than elicitation alone, yet both models still achieve clear improvements. The consistent gains across slot-filling dialogue and ambiguous QA suggest that proactive elicitation is a transferable capability: one that is suppressed by default but recoverable through targeted training with dense credit assignment.

Across base models. To verify that the gains stem from the training recipe rather than from a particular base model, we re-ran the full DialogGRPO pipeline on Gemma3-27B with identical hyperparameters (Appendix F). Before training, both Qwen and Gemma 27B models exhibit the same suppression pattern (*Ask%* below 3% on both splits). After training, both converge to comparable *Ask%*, Cov. F1, and IAR, with similarly sized lifts on every metric (Figure 4), confirming that DialogGRPO generalizes across model families.

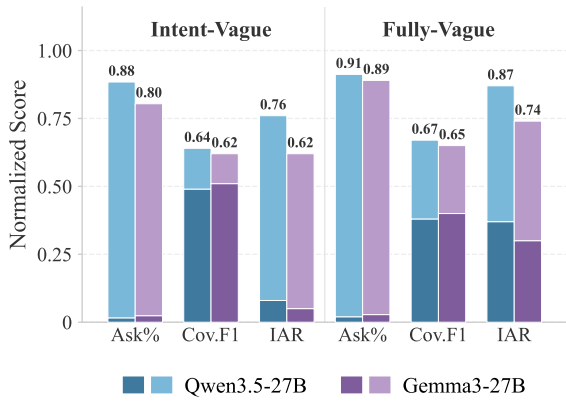


Figure 4: Generalization across base models on VagueBench. Darker/lighter segments denote base model scores and DialogGRPO lifts, respectively. Both 27B models show comparable gains, confirming transferability across model families.

6 Conclusion

We introduce VagueBench, a benchmark exposing that elicitation suppression in LLMs persists across model families and scales, and DialogGRPO, an offline RL method that recovers this latent ability via dialogue-state DAG construction and dense turn-level credit assignment. DialogGRPO teaches models to ask better questions, not merely more, reaching 88–91% Ask% on vague queries while preserving calibration on complete ones, and generalizes zero-shot to MultiWOZ 2.4 and AmbigQA.

Limitations

VagueBench is built from a single commercial assistant’s traffic, which limits its coverage of conversational scenarios, languages, and modalities. Our evaluation relies on a simulated user that cooperatively releases information on matching questions, abstracting away real human behaviors such as partial answers, sudden topic switches, noise, or refusal to elaborate, so the reported gains likely represent an upper bound on what can be achieved with cooperative interlocutors. Open domain dialogue spaces are unbounded, so the DAG covers only a sparse set of trajectories. Local contrastive coverage and state merging reduce this gap, but interactions outside the training distribution remain a limitation. Large preference spaces also make exhaustive questioning impractical. In shopping, a few targeted questions can narrow the options, but preferences may emerge or change during the interaction. VagueBench fixes the hidden preference set for reproducibility and does not capture

this dynamic. Asking frequency is controlled by reward hyperparameters and should be calibrated to application needs.

Ethical Considerations

We have conducted this work with a strong commitment to ethical and responsible research practices. The raw queries used to construct VagueBench originate from a commercial AI assistant that offers tiered data-sharing preferences, only interactions from users who explicitly opted into research use were included. All queries were stripped of personally identifiable information prior to our access, and a dedicated unsafe-content filter removed harmful or sensitive material during the rule-based filtering stage (§3.1). The released benchmark contains only post-filtering, annotation-enriched seeds with no metadata linking back to individual sessions. For the human validation study, two graduate student annotators were recruited and compensated at \$7.50 USD per hour; given the flexibility of the remote task and the economic context of the participants’ residential region, this rate exceeds the local average hourly wage, ensuring fair remuneration. Participation was voluntary, and annotators provided informed consent for the use of their labels. Proactive elicitation is inherently dual-use: it reduces unfounded assumptions and prevents costly errors in high-stakes domains, but could also be steered into over-soliciting personal information unnecessary for the task. Our reward formulation mitigates this by crediting only information units causally required for the final answer; nevertheless, we recommend that production deployments constrain what the model may solicit, audit questions touching sensitive attributes, and give users a clear option to decline. We also note that VagueBench covers English-only queries from a single platform, limiting its linguistic and cultural generalizability. All data released are for research purposes and recommend that any production use be paired with appropriate guardrails.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 42394060 and 42394064, and the Ant Group Research Fund.

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. 2026. Speakrl: Synnergizing reasoning, speaking, and acting in language models with reinforcement learning. In *Proceedings of the 16th International Workshop on Spoken Dialogue System Technology*, pages 312–325.
- Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2020. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq). *arXiv preprint arXiv:2009.11352*.
- Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. 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*, pages 475–484.
- Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367.
- Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah D Goodman. 2024. Star-gate: Teaching language models to ask clarifying questions. *arXiv preprint arXiv:2403.19154*.
- Andrew M Bean, Rebecca Elizabeth Payne, Guy Parsons, Hannah Rose Kirk, Juan Ciro, Rafael Mosquera-Gómez, Sara Hincapié M, Aruna S Ekanayaka, Lionel Tarassenko, Luc Rocher, and 1 others. 2026. Reliability of llms as medical assistants for the general public: a randomized preregistered study. *Nature Medicine*, pages 1–7.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 5016–5026.
- Colin Camerer, George Loewenstein, and Martin Weber. 1989. The curse of knowledge in economic settings: An experimental analysis. *Journal of political Economy*, 97(5):1232–1254.
- Maximillian Chen, Ruoxi Sun, Tomas Pfister, and Serkan Arik. 2025. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *International Conference on Learning Representations*, volume 2025, pages 32244–32279.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.
- Felix Faltings, Michel Galley, Kianté Brantley, Baolin Peng, Weixin Cai, Yizhe Zhang, Jianfeng Gao, and William B Dolan. 2023. Interactive text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4450–4468.
- Yujian Gan, Changling Li, Jinxia Xie, Luou Wen, Matthew Purver, and Massimo Poesio. 2024. *Clarq-llm: A benchmark for models clarifying and requesting information in task-oriented dialog*. *Preprint*, arXiv:2409.06097.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Joey Hong, Sergey Levine, and Anca Dragan. 2023. Zero-shot goal-directed dialogue via rl on imagined conversations. *arXiv preprint arXiv:2311.05584*.
- Zhiyuan Hu, Yue Feng, Anh Tuan Luu, Bryan Hooi, and Aldo Lipani. 2023. Unlocking the potential of user feedback: Leveraging large language model as user simulators to enhance dialogue system. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3953–3957.
- Zhiyuan Hu, Chumin Liu, Xidong Feng, Yilun Zhao, See-Kiong Ng, Anh T Luu, Junxian He, Pang W Koh, and Bryan Hooi. 2024. Uncertainty of thoughts: Uncertainty-aware planning enhances information seeking in llms. *Advances in Neural Information Processing Systems*, 37:24181–24215.
- Tenghao Huang, Sihao Chen, Muhao Chen, Jonathan May, Longqi Yang, Mengting Wan, and Pei Zhou. 2025. Teaching language models to gather information proactively. *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15588–15599.
- Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pages 58791–58831.
- Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. 2025. Llms get lost in multi-turn conversation. *arXiv preprint arXiv:2505.06120*.
- Beth Levin and Malka Rappaport Hovav. 2005. *Argument realization*, volume 10. Cambridge university press Cambridge.

- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let’s verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.
- Sichun Luo, Yi Huang, Mukai Li, Shichang Meng, Fengyuan Liu, Zefa Hu, Junlan Feng, and Qi Liu. 2025. Clarifymt-bench: Benchmarking and improving multi-turn clarification for conversational large language models. *arXiv preprint arXiv:2512.21120*.
- Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 5783–5797.
- Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. 2019. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*.
- Jan Peters and Stefan Schaal. 2007. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pages 745–750.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- John R Searle. 1969. Speech acts: An essay in the philosophy of language. *Cambridge University*.
- Omar Shaikh, Hussein Mozannar, Gagan Bansal, Adam Fourney, and Eric Horvitz. 2025. Navigating rifts in human-llm grounding: Study and benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20832–20847.
- Lior Shani, Aviv Rosenberg, Asaf Cassel, Oran Lang, Daniele Calandriello, Avital Zipori, Hila Noga, Orgad Keller, Bilal Piot, Idan Szpektor, and 1 others. 2024. Multi-turn reinforcement learning with preference human feedback. *Advances in Neural Information Processing Systems*, 37:118953–118993.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Weiyan Shi, Kun Qian, Xuewei Wang, and Zhou Yu. 2019. How to build user simulators to train rl-based dialog systems. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 1990–2000.
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158.
- Robert S Taylor. 1968. Question-negotiation and information seeking in libraries.
- Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. 2025. Collablmm: From passive responders to active collaborators. *arXiv preprint arXiv:2502.00640*.
- Chenyang Yang, Yike Shi, Qianou Ma, Michael Xieyang Liu, Christian Kästner, and Tongshuang Wu. 2025. What prompts don’t say: Understanding and managing underspecification in llm prompts. *arXiv preprint arXiv:2505.13360*.
- Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 351–360.
- Michael Zhang, W Bradley Knox, and Eunsol Choi. 2025. Modeling future conversation turns to teach llms to ask clarifying questions. In *International Conference on Learning Representations*, volume 2025, pages 60722–60742.
- Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. 2024. Clamber: A benchmark of identifying and clarifying ambiguous information needs in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10746–10766.

A Data Construction and Statistics

This appendix expands on §3.2 with the filtering pipeline that produces the 4,930-seed pool and four statistical views of that pool: its task-family composition, the per-seed counts of annotated intents and details, and the initial information coverage of the two vague variants. Together they characterise the bench used throughout the main text.

Filtering pipeline. Figure 5 traces the flow of queries through the four-stage filtering pipeline described in §3.1. *Rule-based Filtering* on length and unsafe content removes 48.7% of raw queries. Two subsequent stages of *LLM-based Filtering* strip greetings, test strings, and context-dependent utterances (39.3%), then queries that are too detail-sparse to degrade meaningfully (6.1%). A final *Intent & Detail Annotation* stage rejects another 0.9% for annotation failure, leaving 5.0% of the original ~100K queries as the information-complete seed pool used throughout the paper.

Task-family distribution. Figure 6 reports the composition of the 4,930-seed pool by task family. *Consultation & Analysis* and *Information Query* dominate at 49.1% and 31.3% respectively, with *Recommendation & Decision* queries forming a further 13.6%. *Creation & Generation* and *Task Execution* form the long tail at 4.4% and 1.5%. The distribution inherits the empirical mix observed in the source assistant traffic.

Information density of seeds. Figure 7 shows the per-seed distribution of annotated intents and details. VagueBench is anchored in single-intent queries, with 69.0% of seeds carrying exactly one intent and a mean of 1.36 intents per seed. Detail counts form a roughly bell-shaped distribution peaking at four details per seed (34.9%) with a mean of 4.63. This information density is what makes the bench non-trivial: every seed carries multiple removable units, so the degraded variants expose real hidden information rather than vacuous gaps.

Initial information coverage of the variants. Figure 8 measures the initial information coverage of each query variant, that is, the fraction of gold information units already expressed in the initial query. Intent-Vague preserves 77.0% of units on average by construction, since it weakens only intent anchors; Fully-Vague drops coverage to 18.7%, removing intent anchors together with a sample of

detail anchors. The two distributions barely overlap, confirming that the controlled degradation produces two cleanly separated evaluation regimes that correspond exactly to the two splits used in §5.

B Human Study

To validate three components of our pipeline, we conducted a small human study with two annotators who were graduate students familiar with dialogue research and blind to each other’s labels. The three experiments verify, in order, whether the vague queries in VagueBench genuinely warrant clarification, whether the synthesized intent and detail annotations align with human judgement, and whether the LLM extractor used in our evaluation protocol recovers the same information units that a human reader would. All agreement scores are reported as Cohen’s κ at the unit level.

B.1 Experiment 1: Do Vague Queries Genuinely Warrant Clarification?

Setup. We sampled 50 queries uniformly at random from the Intent-Vague split and 50 from the Fully-Vague split, for a total of 100 queries. To verify if the omitted details were genuinely necessary, the two annotators were shown the vague query alongside the complete gold query q^* . Each annotator independently judged whether the information gap objectively warrants at least one clarifying question before attempting an answer, recording a binary Ask or Answer decision.

Results. Table 6 reports the two annotators’ Ask rates on each split. On Intent-Vague queries both annotators chose to ask on at least 96% of samples, and on Fully-Vague queries both chose to ask on 100%. The results indicate that the controlled degradation in VagueBench produces vague queries that human readers consistently recognise as information-deficient, and that the gap between this human Ask rate and the 1.6% and 2.0% Ask rate of un-prompted Qwen3.5-27B in Table 2 reflects a behavioural suppression in the model rather than ambiguity in the data.

B.2 Experiment 2: Are the LLM-Annotated Intent and Detail Units Accurate?

Setup. We sampled 50 complete queries q^* from the test pool. For each query, the two human annotators and the LLM annotator (GLM-5) each independently extracted the intent and detail units $\mathcal{TUD}$ directly from the query, with no access to one

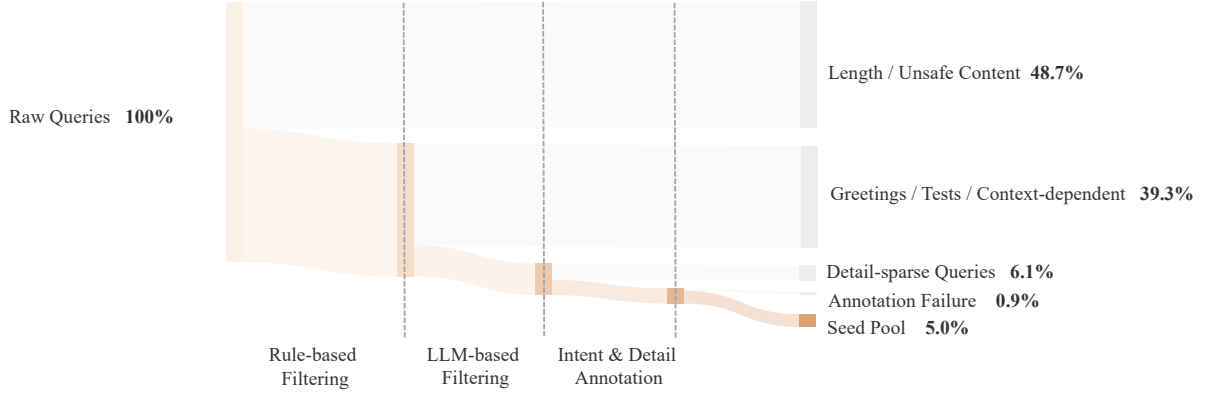


Figure 5: Flow of queries through the four-stage VagueBench filtering pipeline. *Rule-based Filtering* removes 48.7% of the $\sim 100K$ raw queries on length or unsafe-content grounds. *LLM-based Filtering* then discards greetings, test strings, and context-dependent utterances (39.3%), followed by detail-sparse queries that lack the structure required for degradation (6.1%). A final *Intent & Detail Annotation* stage yields 4,930 information-complete seeds, 5.0% of the input. The lighter peach tones on the left mark earlier stages and the deepest tone on the right marks the final seed pool; grey side-flows are the discarded branches at each stage.

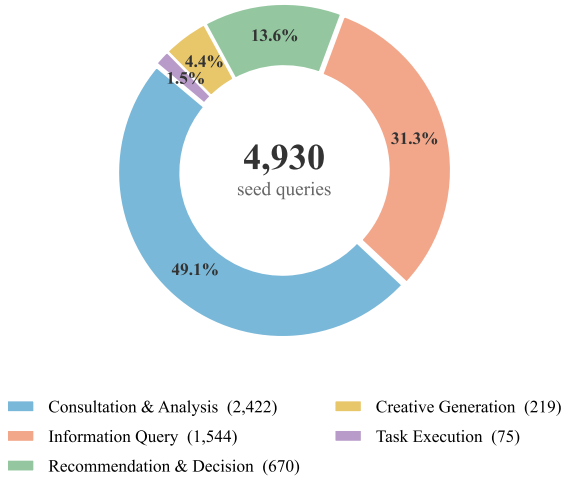


Figure 6: Task-family composition of the 4,930 VagueBench seed queries. *Consultation & Analysis* and *Information Query* together account for over 80% of the pool, while *Task Execution* forms the rarest 1.5%. The bench inherits a long-tailed mix that mirrors real-world assistant traffic.

another’s outputs. After this independent extraction, the human judges manually aligned the three annotation sets while viewing the complete query. They treated two units as identical when they denoted the same intent or the same detail key–value pair, regardless of surface wording. The union of all aligned units forms the per-query decision pool, on which each unit receives a binary label from every annotator indicating whether that annotator extracted it. Cohen’s κ is computed pairwise at the unit level, pooled over all 50 queries. Because the

Split	Human 1 Ask%	Human 2 Ask%
Intent-Vague	98.0	96.0
Fully-Vague	100.0	100.0

Table 6: Human Ask rates on vague queries. Both annotators were given the vague query q together with its complete gold query q^* and judged whether the information gap warranted a clarifying question before answering. High Ask rates confirm that the vague queries genuinely warrant clarification.

pool is the union of all three sources rather than a pre-given candidate list, this protocol reflects both spurious units and missing units, so the agreement scores capture annotation completeness as well as precision.

Results. Table 7 reports the three pairwise Cohen’s κ values between the two annotators and the LLM annotation. Human 1 and Human 2 reached $\kappa = 0.87$, indicating that the unit-level Present or Absent criterion is well-defined. The agreement between each annotator and the LLM annotation was $\kappa = 0.83$ and $\kappa = 0.81$, about 0.05 below the inter-annotator ceiling, supporting the use of LLM-extracted annotations as gold labels in VagueBench.

B.3 Experiment 3: Does the LLM Extractor Recover the Same Information Units as Humans?

Setup. The LLM extractor in our evaluation pipeline maps a simulated-user reply to the set of information units the reply discloses. To check

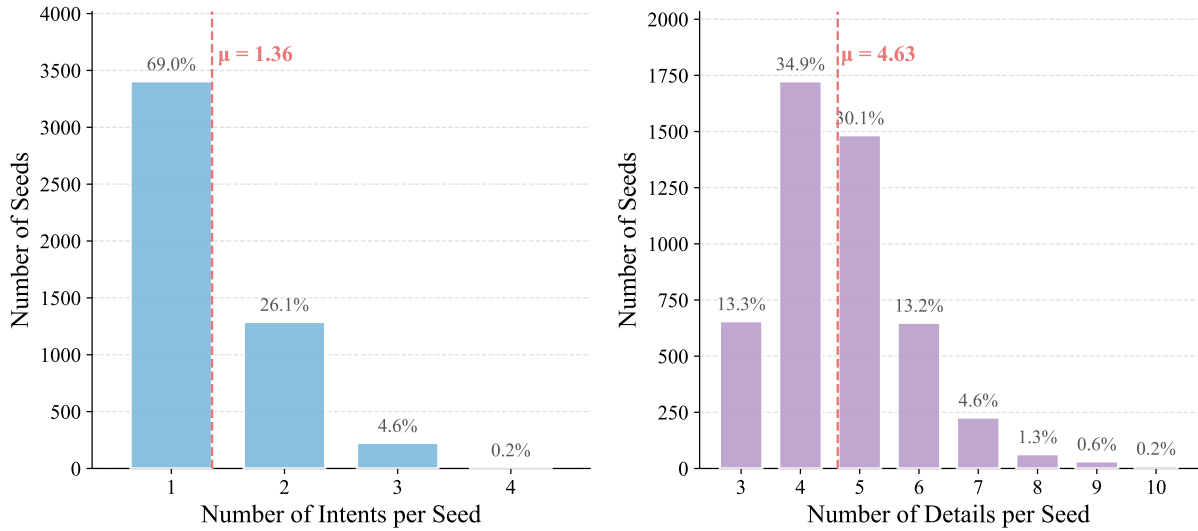


Figure 7: Per-seed counts of annotated intents (left) and details (right) in the 4,930-seed pool. Seeds carry on average 1.36 intents and 4.63 details; 69.0% have exactly one intent, and detail counts cluster in the 3 to 6 range.

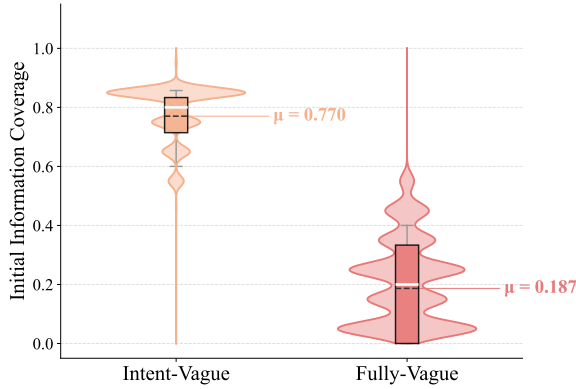


Figure 8: Initial information coverage of each query variant. Coverage is the fraction of gold information units, intents and details combined, that are already expressed in the initial query. Intent-Vague (peach) preserves 77.0% of units on average since only intent anchors are weakened, while Fully-Vague (coral) drops to 18.7% as both intent and a sample of detail anchors are removed. KDE smoothed; box-and-whisker overlays show median, interquartile range, and 5/95 percentiles.

that this extractor agrees with human judgement, we sampled 50 dialogue turns from completed VagueBench evaluations, each consisting of the assistant’s clarifying question, the simulated user’s reply, and the set of candidate units that could plausibly be disclosed at that turn. For every candidate unit, each annotator independently judged Disclosed or Not Disclosed by the reply, and the LLM extractor contributed its own Disclosed or Not Disclosed label on the same candidate set. Agreement is again computed at the unit level, pooled over all

Pair	Cohen’s κ
Human 1 vs Human 2	0.87
Human 1 vs LLM	0.83
Human 2 vs LLM	0.81

Table 7: Unit-level agreement on whether each candidate intent or detail unit is present in the query. Computed on a candidate-set pool drawn from 50 queries.

50 turns.

Results. Table 8 reports the three pairwise Cohen’s κ values. Inter-annotator κ was 0.84, comparable to Experiment 2. The LLM extractor agreed with Human 1 at $\kappa = 0.80$ and with Human 2 at $\kappa = 0.79$. The closeness of these three numbers indicates that the extractor is interchangeable with a human annotator for the purpose of unit-level recovery, and therefore that Cov. F1 and IAR as measured against the extractor’s output are reliable proxies for what a human would have scored.

Pair	Cohen’s κ
Human 1 vs Human 2	0.84
Human 1 vs LLM	0.80
Human 2 vs LLM	0.79

Table 8: Unit-level agreement on whether each candidate unit was disclosed by a simulated-user reply. Computed on a candidate-set pool drawn from 50 dialogue turns.

C Simulated User

This appendix specifies the simulated user $\mathcal{S}$ (Shi et al., 2019; Hong et al., 2023; Faltings et al., 2023; Hu et al., 2023) introduced in §3.3. $\mathcal{S}$ is the core component of the VagueBench evaluation protocol: it plays the role of a real user who holds the hidden information and decides, on each turn, which units to release in response to the model’s question. Two properties anchor its design: *consistency*—identical questions elicit responses of the same kind—and *minimal disclosure*—only precisely-matched units are released, with no volunteered extras.

C.1 Behaviour Rules

The behaviour of $\mathcal{S}$ is governed by five rules over the hidden intent set $\mathcal{T}_{\text{hidden}}$ and detail set $\mathcal{D}_{\text{hidden}}$.

R1. Information release. When the model’s question semantically matches the trigger keywords of some hidden intent $I_j \in \mathcal{T}_{\text{hidden}}$ or detail $d_i \in \mathcal{D}_{\text{hidden}}$, $\mathcal{S}$ releases that unit in short, natural language. For example, the question “Which area would you like to dine in?” matches the trigger of $\langle \text{area, near University Avenue} \rangle$ and $\mathcal{S}$ replies “near University Avenue in downtown Palo Alto”; an intent-side match is symmetric, e.g. “Are you looking for restaurant recommendations or something else?” matches the `recommend_restaurant` intent and $\mathcal{S}$ replies “yes, recommend a few restaurants”. Matching is judged at the semantic level by a strong LLM rather than by exact keyword overlap, so phrasings such as “roughly how much do you want to spend?” still match a budget trigger listing `budget / price / spending`.

R2. Deflection. If the question matches no hidden unit, $\mathcal{S}$ returns a deflective reply drawn at random from a predefined pool: “I haven’t decided yet”, “I’m not sure about that”, “you decide for me”, “no particular preference”, and similar. Deflections are short, natural, and carry no cues that the model could use to infer the hidden set.

R3. No proactive disclosure. $\mathcal{S}$ never volunteers a unit that was not asked about. Questions that approach several units but precisely match only one—for example “any requirements for your meal?” matching only `scenario`—release only the matched unit, regardless of how close the others appear.

R4. Release once. Once released, a unit is removed from $\mathcal{T}_{\text{hidden}}$ or $\mathcal{D}_{\text{hidden}}$. Subsequent questions that re-hit the same unit are answered with “I mentioned that earlier” or by repeating the previous value; no new credit is granted at evaluation.

R5. Multi-unit matches. When a single question matches multiple hidden units—for example “which city and what occasion?” hitting both `city` and `scenario`— $\mathcal{S}$ releases all matched units in one reply. This rule ensures that elicitation efficiency is not artificially deflated by simulator-side fragmentation.

C.2 Direct-Answer Behaviour

If the model commits to a final answer at the initial turn without asking, the dialogue terminates immediately and the evaluation proceeds without further simulator input. On information-complete queries this is the expected behaviour and requires no simulator participation. On Intent-Vague and Fully-Vague queries, the absence of a question means the model answered under missing information, and the resulting answer is naturally penalised by Cov. F1, which compares the response against the gold information set. Whether a model turn counts as *ask* or *answer* is judged by the LLM-based behaviour judge (Prompt 3 in Appendix J).

C.3 System-Prompt Template

Each Intent-Vague and Fully-Vague sample carries a system prompt that instantiates $\mathcal{S}$ for that sample. The template has four components:

- **Role description.** The complete underlying need together with the meta query actually shown to the model.
- **Expressed information.** The content already present in the meta query, which $\mathcal{S}$ does not need to release.
- **Hidden-information table.** A structured listing of every hidden intent and detail, each annotated with its trigger keywords and the natural-language reply to use on release.
- **Deflection pool.** The set of non-informative replies drawn upon when no unit is matched.

A representative Fully-Vague system prompt is shown in Figure 9.

C.4 Implementation

$\mathcal{S}$ is realised with GLM-5.1, which receives the per-sample system prompt and produces a reply to each

Role. You are a user asking an AI assistant for help. Your actual need is: a recommendation for a Japanese restaurant near University Avenue in downtown Palo Alto suitable for business dining, around \$120 per person, ideally with a private room. You only said: “Palo Alto Japanese food”.

Hidden intents.

- `recommend_restaurant` — if the AI asks what you want / what kind of help / your goal → reply “*recommend a few restaurants*”.

Hidden details.

- `area` = near University Avenue — if the AI asks about location / district / where → reply “*near University Avenue in downtown Palo Alto*”.
- `scenario` = business dining — if the AI asks about occasion / company / purpose → reply “*business dining*”.
- `budget` = around \$120 per person — if the AI asks about budget / price / spending → reply “*around \$120 per person*”.
- `private_room` = required — if the AI asks about ambience / seating / private rooms → reply “*private room preferred*”.

Deflection. For unmatched questions, randomly reply with one of “*I don’t have particular preferences*”, “*you decide for me*”, “*I haven’t thought about that*”.

Notes. Reply briefly and naturally, like a real user. Do not volunteer information that was not asked about. Answer only what was asked.

Figure 9: **System prompt example** for a Fully-Vague sample. The hidden-information table lists the four detail units and one intent unit that $\mathcal{S}$ may release on a matching question.

model question. Semantic matching between the question and the hidden-unit triggers is performed inside the same inference call: the LLM judges, on the basis of the trigger keywords and the hidden-information table in the system prompt, whether the model’s question hits any unit, and emits a release reply or a deflection accordingly. To keep simulator behaviour consistent and auditable, we set the sampling temperature to 0.1 to suppress drift and log every simulator turn for post-hoc inspection and spot-checking.

D Implementation Details

This appendix lists the models evaluated in the experiments of §5, the hyperparameters used to train DialogGRPO, and the hardware on which training and inference were carried out.

Evaluated models. Table 9 enumerates the closed and open-source models compared on VagueBench and the two transfer benchmarks.

Provider	Model	Model Link
OpenAI	GPT-5.4 GPT-oss-120B	OpenAI Docs Hugging Face
Anthropic	Claude Sonnet 4.6	Anthropic
Google	Gemini 3.1 Pro Gemma3-27B	Google DeepMind Hugging Face
Meta	Llama-3.1-8B-Instruct	Hugging Face
Qwen	Qwen3-8B Qwen3.5-27B	Hugging Face Hugging Face
Zhipu	GLM-5 GLM-5.1	Hugging Face Hugging Face

Table 9: Models evaluated in this work. The same Qwen3.5-27B checkpoint serves as a base-model baseline (Table 2); the cross-base experiment in §5.4 additionally trains Gemma3-27B with identical hyperparameters. GLM-5.1 plays the simulated user $\mathcal{S}$ (Appendix C).

Hyperparameter	Value
Learning rate	10^{-4}
Warm-up ratio	0.1
LR scheduler	cosine
Batch size	48
Epochs	3
LoRA rank	16
LoRA alpha	32

Table 10: Training hyperparameters for DialogGRPO. The same configuration is used in the cross-base-model experiment of §5.4.

Closed-source models are accessed via their official APIs; open-source models are loaded from the Hugging Face checkpoints linked in the table.

Training hyperparameters. DialogGRPO is trained with LoRA adapters on top of the Qwen3.5-27B base; the Gemma3-27B run in §5.4 uses the same configuration with only the base weights swapped. Table 10 reports the full set of training hyperparameters.

Hardware. All training and inference runs are carried out on a single node equipped with eight NVIDIA H200 GPUs. DialogGRPO training of Qwen3.5-27B and the cross-base Gemma3-27B run in §5.4 both use the same eight-GPU node; open-source baseline inference and the simulated-user rollouts used for evaluation are also run on this hardware.

E General Capability Preservation

A natural concern about any offline reinforcement learning recipe is that the trained model may have drifted away from the base distribution and lost general task ability. To address this, we evaluated our DialogGRPO model on six general-capability benchmarks alongside the un-modified Qwen3.5-27B base, covering coding, mathematics, commonsense, and broad knowledge: LiveCodeBench (Jain et al., 2025), MathQA (Amini et al., 2019), CommonsenseQA (Talmor et al., 2019), MATH-500 (Lightman et al., 2024), MMLU (Hendrycks et al., 2020), and GSM8K (Cobbe et al., 2021). No clarification-related training data overlap with any of these benchmarks.

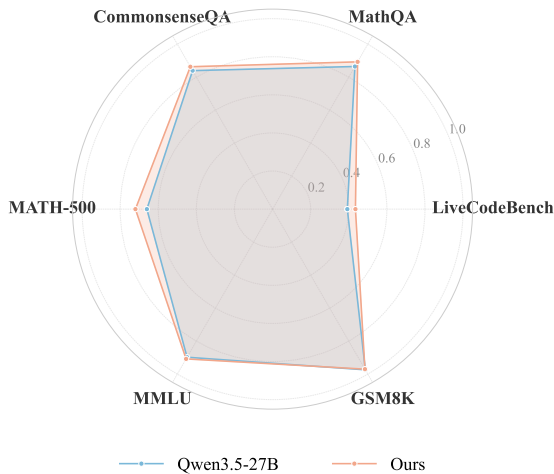


Figure 10: General-capability check on six external benchmarks: LiveCodeBench, MathQA, CommonsenseQA, MATH-500, MMLU, and GSM8K. The two envelopes trace nearly identical shapes, with the trained DialogGRPO model staying at or above the Qwen3.5-27B base on every benchmark and slightly improving on LiveCodeBench and MATH-500. The result indicates that the proactive elicitation behaviour reported in the main text is acquired without sacrificing general task ability.

Figure 10 shows the comparison as a radar chart. The two envelopes are nearly identical, with our model matching the base within measurement noise on four of the six benchmarks and slightly exceeding it on LiveCodeBench and MATH-500. The result mitigates a potential reviewer concern: the proactive elicitation behaviour reported in the main text is acquired without sacrificing general task accuracy. We attribute this to the nature of what DialogGRPO trains. The objective optimises *when* and *what* to ask, a decision-making behaviour over the dialogue state, rather than injecting domain

knowledge into the model; it reshapes how the model acts on missing information without overwriting the factual, mathematical, or coding competence already present in the base weights. This behaviour-level target is reinforced by the diversity of our training data, which spans five task families and a broad range of open-domain topics rather than any single narrow domain. Because no benchmark-specific knowledge is added and no narrow distribution is over-fit, the elicitation policy is learned on top of the base model’s general ability rather than at its expense.

F Generalization to a Different Base Model

This appendix expands on the cross-base-model generalization experiment summarised in §5.4 and Figure 4.

Setup. To check that our method generalises beyond a single base model, we re-ran the full training pipeline on Gemma3-27B, keeping every hyperparameter identical to the Qwen3.5-27B run reported in the main text. The DAG construction, reward design were unchanged; only the base-model weights differed.

The gains carry over. Gemma3-27B shows the same suppression pattern as Qwen3.5-27B before training, with the base Ask% below 3% on both splits, confirming that clarification suppression is shared across base-model families rather than being a property of one model. After training the Gemma version reaches Ask%, Cov. F1, and IAR levels that closely track those observed on Qwen3.5-27B, and the lift bars in Figure 4 are of similar height on every metric. The consistency across two unrelated base models indicates that the training recipe transfers, and that the improvements in the main paper are attributable to the DialogGRPO design rather than to a peculiarity of any single base model.

G Additional Experiments

We additionally compare DialogGRPO with ColabLLM (Wu et al., 2025) and test transfer to a sub-10B backbone. Because the released ColabLLM checkpoint uses Llama-3.1-8B-Instruct, we re-ran the full DialogGRPO pipeline on the same backbone for a controlled same-backbone comparison.

ColabLLM produces only modest gains over the base model on VagueBench, increasing Ask% to 3.2/4.4. This is consistent with its focus on vertical

Method (Llama-3.1-8B-Instruct)	Intent-Vague			Fully-Vague		
	Ask%	Cov. F1	IAR	Ask%	Cov. F1	IAR
Base	0.8	0.50	0.00	1.2	0.35	0.23
CollabLLM	3.2	0.59	0.00	4.4	0.37	0.27
DialogGRPO	76.6	0.61	0.37	82.0	0.49	0.43

Table 11: Same-backbone comparison with CollabLLM and transfer of DialogGRPO to a sub-10B model on VagueBench.

tasks such as document editing, code, and mathematics, using task-specific rewards and online user simulation rather than open-domain recovery of never-stated information. On the same Llama backbone, DialogGRPO reaches 76.6/82.0 Ask% while improving both Cov. F1 and IAR. The suppression-then-recovery pattern therefore also holds at the 8B scale, providing evidence that the training recipe transfers beyond the two 27B backbones reported in §5.4.

H Derivation of the Advantage-Weighted Objective

This appendix derives the training objective in Eq. 9 from a KL-regularized policy optimization problem. The derivation follows the classical reward-weighted regression line (Peters and Schaal, 2007; Peng et al., 2019); the same closed-form step underlies the optimal policy in DPO (Rafailov et al., 2023). We make explicit how the KL constraint is absorbed into an exponential advantage weight, which is why Eq. 9 carries no separate KL term.

KL-regularized objective. For a state s (a canonical prompt p_v) with candidate actions a (responses y_e), we seek a policy π that maximizes the advantage $A(s, a) = \hat{A}_e$ while staying close to the reference policy π_{ref} :

$$\max_{\pi} \mathbb{E}_{s \sim \mathcal{D}} \left[\sum_a \pi(a | s) A(s, a) - \beta \text{KL}(\pi(\cdot | s) \| \pi_{\text{ref}}(\cdot | s)) \right], \quad (10)$$

subject to the normalization constraint $\sum_a \pi(a | s) = 1$ for every s .

Closed-form optimum. Introducing a Lagrange multiplier $\lambda(s)$ for the normalization constraint at each state and setting the derivative of the Lagrangian with respect to $\pi(a | s)$ to zero gives

$$A(s, a) - \beta \log \frac{\pi(a | s)}{\pi_{\text{ref}}(a | s)} - \beta - \frac{\lambda(s)}{P(s)} = 0. \quad (11)$$

Solving for π and folding all a -independent terms into a per-state normalizer $Z(s)$ yields the optimal policy

$$\pi^*(a | s) = \frac{1}{Z(s)} \pi_{\text{ref}}(a | s) \exp\left(\frac{1}{\beta} A(s, a)\right), \quad (12)$$

where $Z(s) = \sum_a \pi_{\text{ref}}(a | s) \exp(A(s, a)/\beta)$ ensures $\sum_a \pi^*(a | s) = 1$. The KL constraint in Eq. 10 is thus not discarded: it is absorbed analytically into the exponential factor $\exp(A/\beta)$ and the reference term π_{ref} .

Projection onto a parametric policy. $Z(s)$ requires summing over all actions and is intractable, so π^* cannot be used directly. We instead fit a parametric policy π_{θ} to π^* by minimizing the forward KL divergence $\text{KL}(\pi^* \| \pi_{\theta})$, which, after dropping θ -independent terms, is equivalent to

$$\max_{\theta} \mathbb{E}_s \left[\sum_a \pi^*(a | s) \log \pi_{\theta}(a | s) \right]. \quad (13)$$

Substituting Eq. 12 gives

$$\max_{\theta} \mathbb{E}_s \left[\sum_a \frac{1}{Z(s)} \pi_{\text{ref}}(a | s) \cdot \exp\left(\frac{A(s, a)}{\beta}\right) \log \pi_{\theta}(a | s) \right]. \quad (14)$$

Empirical objective. In our offline setting the candidate set $\mathcal{G}_v$ at each node is pre-generated by the external strong LLM, so it does not coincide with samples from π_{ref} . We therefore treat the candidates as an empirical sampling distribution and approximate the expectation over $\pi_{\text{ref}}(a | s)$ by an average over $\mathcal{G}_v$; this is a stated approximation rather than an exact identity, and is made practical by the advantage weighting, which down-weights candidates far from high-advantage regions. Since $Z(s)$ is independent of θ and acts only as a per-state scaling, it can be dropped during optimization. This yields the per-node objective in Eq. 9. Unlike

trajectory-level reward-weighted regression (Peters and Schaal, 2007), the advantage $\hat{A}_e$ here is computed per edge within a dialogue-state DAG (§4.2), giving dense turn-level credit assignment rather than a single trajectory return.

I Case Studies

This appendix shows three trajectories produced by DialogGRPO on representative test queries, rendered as chat transcripts. Each turn shows a clarifying question asked by the model on the left and the simulated user’s reply on the right; the green panel at the bottom summarises the final answer with its key points highlighted.

J Prompt Templates

This section collects the four prompt templates that drive VagueBench’s construction and evaluation pipelines. Variables enclosed in $\{ \}$ are filled with run-time values. The four prompts map to the paper as follows: Prompt 1 is the seed filter referenced in §3.1; Prompt 2 is the variant-synthesis prompt described in §3.2; Prompt 3 is the behavior judge that produces the Ask% metric in §3.3; Prompt 4 is the information extractor that produces Cov. F1 and supports IAR in the same section.

For the model evaluation, we additionally use two system prompts. In the naive setting, the system prompt is “You are a helpful assistant.” In the question-guided setting, we use “You are a helpful assistant. When you think it is necessary, you may ask the user a question. Ask at most one question at a time.”

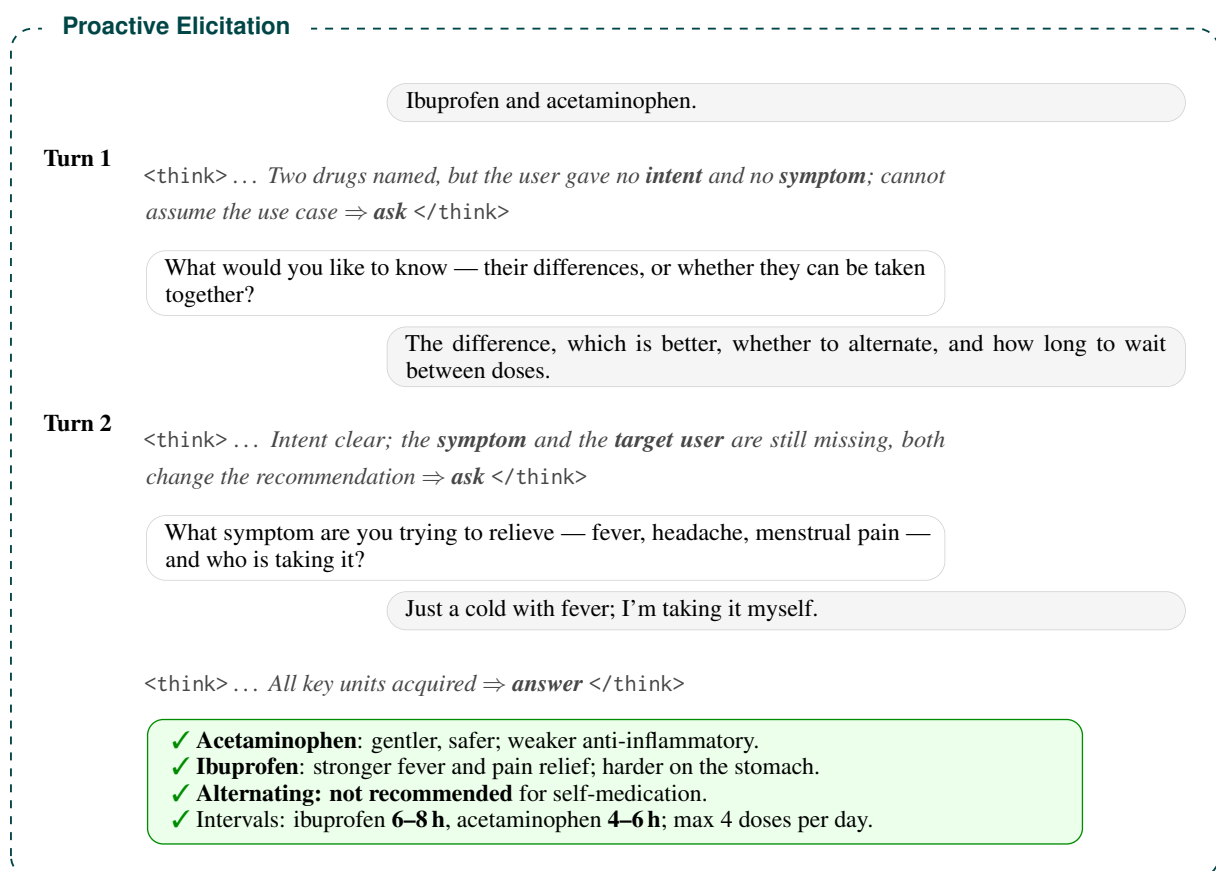


Figure 11: **Case 1.** Drug comparison — the model elicits the user's intent, then the symptom and the target user, before delivering a focused recommendation.

Proactive Elicitation

Short-term stocks for next Monday.

Turn 1

<think> ... Time, strategy, and asset type are stated; the *specific intent*, recommendation versus analysis, is missing ⇒ *ask* </think>

Are you asking for short-term recommendations for next Monday, or technical analysis of a particular sector or stock?

Short-term recommendations — hot sectors or direct tickers.

Turn 2

<think> ... Hot short-term plays swing widely; the user's *risk preference* is not stated ⇒ *ask* </think>

Hot short-term plays swing widely. Aggressive moves such as chasing limit-up stocks, or something more conservative?

Aggressive is fine; short-term is about chasing returns. Higher risk is OK.

Turn 3

<think> ... Risk profile clear; the *investor identity*, retail vs. institutional, still affects liquidity matching ⇒ *ask* </think>

Are you an individual retail investor? Liquidity needs differ between retail and institutional accounts.

Yes, retail.

<think> ... All key units acquired ⇒ *answer* </think>

- ✓ Focus on **leading names in currently hot sectors**, e.g. technology or new energy.
- ✓ Track **capital flow at Monday's open**; enter the strongest names on confirmation.
- ✓ Always preset a **stop-loss** given the volatility.
- ✓ Compliance limits prevent listing tickers; pair sector analysis with a licensed advisor.

Figure 12: **Case 2.** Short-term stock query — the model successively elicits intent, risk preference, and investor identity before delivering a strategy sketch.

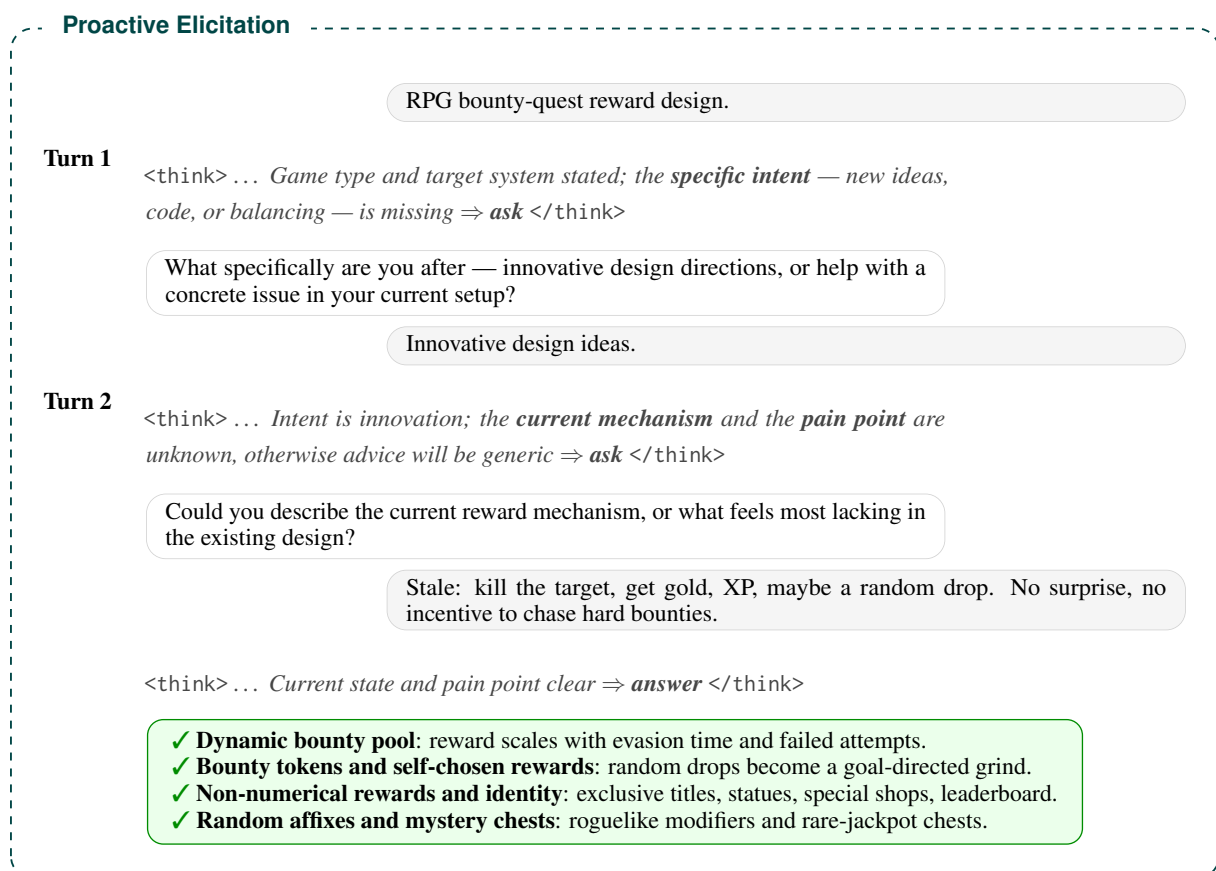


Figure 13: **Case 3.** RPG reward design — the model elicits intent, then the current mechanism and pain point, before delivering targeted design ideas.

[System Prompt]

Determine whether the given query qualifies as an information-complete *seed query* for VagueBench.

A seed query must simultaneously satisfy all of the following conditions:

1. **Clear intent.** The query contains explicit intent anchors: precise action verbs, explicit speech-act framing, or clear output expectations, so that the user’s task can be identified directly.
2. **Sufficient details.** The query carries the key parameters and constraints needed to execute that task, so that a competent assistant could produce a high-quality answer without asking any clarifying question.
3. **Independent and self-contained.** The query does not depend on images, attached files, or prior dialogue turns; it can be understood out of context.
4. **Compliant content.** The query is not pure chit-chat, a test string such as “hello” or “test”, policy-violating content, or a code-debugging snippet pasted with no accompanying question.

Reject the query if any of the following holds:

- *Vague intent.* Only entities or keywords without an action intent, for example “Tesla” or “Shanghai restaurants”.
- *Missing details.* The intent is clear but key parameters are absent, for example “Help me write an email” without recipient, subject, or content.
- *Context-dependent.* The query uses pronouns or references such as “what I just said”, “the one above”, or “help me modify”.
- *Privacy.* The query exposes the user’s personal private information.

Output JSON: {"is_seed": true|false, "reason": "One-sentence reason"}

Figure 14: **Prompt 1: Seed Query Filter.**

[System Prompt]

You are a dialogue-data annotation expert. Given an information-complete seed query, annotate it into the information set $\mathcal{I} = (\mathcal{T}, \mathcal{D})$ and synthesise the **Intent-Vague** and **Fully-Vague** variants following the controlled-degradation procedure.

Step 1: Query-type classification.

Choose exactly one type from: Information Query / Recommendation & Decision / Task Execution / Creation & Generation / Consultation & Analysis.

Step 2: Annotate $\mathcal{I} = (\mathcal{T}, \mathcal{D})$.

Intent set $\mathcal{T}$. All task-level goals the user expects to complete, each represented as $\{id, label\}$. Examples of intent labels: “recommend restaurant”, “translate text”, “compare plans”. A seed may have a single intent or multiple parallel intents.

Detail set $\mathcal{D}$. Specific parameters and constraints required to execute the intents, each represented as $\{id, key, value, intent_ids\}$. Examples of detail keys: city, budget, scenario, cuisine. Each detail records the list of intents it serves.

Step 3: Synthesise the Intent-Vague variant.

Goal. Weaken or remove *only* the intent-layer anchors while preserving every detail-layer anchor and every detail unit in $\mathcal{D}$. After rewriting it should be impossible to identify the user’s intended task, yet every parameter, constraint, output-type cue, and scope cue must remain.

Operations.

- *Predicate anchors* (precise action verbs) $\rightarrow$ *Generalise*. For example: “summarize” $\rightarrow$ “work on”, “translate” $\rightarrow$ “process”, or drop the verb entirely.
- *Speech-act anchors* (sentence frames such as “Help me X”, “I need X”) $\rightarrow$ *Weaken or remove*. For example: “Help me X” $\rightarrow$ “About X”; “I need X” $\rightarrow$ “How is X”.
- *Output-type anchors* and *scope anchors* (detail layer) $\rightarrow$ *Keep verbatim*.

Rewriting rules.

- Every detail-layer expression and every detail unit in $\mathcal{D}$ must be retained.
- The rewritten text must be natural and fluent, resembling a real underspecified user message.
- `iv_blurred_intent_ids`: list the intent IDs whose anchors were weakened or removed (typically all of $\mathcal{T}$).

Step 4: Synthesise the Fully-Vague variant.

Goal. Hide all intent-layer anchors, remove the detail-layer anchors, and additionally strip a subset of detail units, retaining only a core-topic phrase of typically 5 to 10 tokens.

Operations.

- *Predicate anchors* $\rightarrow$ *Generalise or delete*.
- *Speech-act anchors* $\rightarrow$ *Weaken or remove*.
- *Output-type anchors* $\rightarrow$ *Delete*. For example: “five bullet points” $\rightarrow \emptyset$; “give me a report” $\rightarrow$ drop “report”.
- *Scope anchors* $\rightarrow$ *Hedge or delete*. For example: “analyse from a nutritional perspective” $\rightarrow$ “about nutrition”.

Rewriting rules.

- Identify the most core entity or keyword of the seed and express it in a minimalist way, such as “[Core Entity]” or “[Core Entity] + short vague modifier”.
- Retaining one to two core detail words is allowed, but the intent must remain hidden.
- The result should read like a casual, partially-typed user input.
- `fv_retained_detail_ids`: list the detail IDs that remain visible in the Fully-Vague variant; the list may be empty.

Output constraints.

- All id fields, including `intent_ids`, `iv_blurred_intent_ids`, and `fv_retained_detail_ids`, must reference only the ids defined in this same output.
- Do not describe any set operations in prose; only emit the structured fields.
- Do not reason about what was hidden; only record what was retained.

Output format (strict JSON).

```
{
  "query_type": "Type",
  "intents": [
    {
      "id": "I1",
      "label": "Description of task intent",
      "triggers": ["trigger phrase 1", "trigger phrase 2"],
      "response": "Colloquial short answer"
    }
  ],
  "details": [
    {
      "id": "d1",
      "key": "Field name",
      "value": "Field value",
      "intent_ids": ["I1"],
      "triggers": ["trigger word 1", "trigger word 2"],
      "response": "Colloquial short answer"
    }
  ],
  "iv_query": "Intent-Vague variant (intent blurred, all details kept)",
  "iv_blurred_intent_ids": ["I1"],
  "fv_query": "Fully-Vague variant (5-10 token core-topic phrase)",
  "fv_retained_detail_ids": []
}
```

[User Prompt]

The seed query is: {seed_query}

Figure 15: Prompt 2: Variant Synthesis.

[System Prompt]

Classify the assistant's response to a query as either **ask** (a clarifying question) or **answer** (a direct attempt at the task). This classification supplies the *decision-appropriateness* signal that produces the Ask% column of Table 2.

- **ask**: The core purpose of the response is to elicit additional information from the user. It contains interrogative sentences or explicit information requests directed at the user and does not simultaneously provide a complete answer to the task.
- **answer**: The response provides a substantive answer to the task. Even if an optional follow-up question is appended at the end, such as "these are the recommendations, do you have any other preferences?", the response is still considered a direct answer.

Output JSON: {"behavior": "ask" | "answer"}

[User Prompt]

The user's query is: {query}

The assistant's response is: {response}

Figure 16: **Prompt 3: Behavior Judge.**

[System Prompt]

You are an information-extraction expert. You are given a piece of *text*, the user's *original query*, and a structured *ground-truth information set* containing an intent list and a detail list. Your task is to decide, for every intent and every detail in the ground truth, whether the text covers it. Also list any intent or detail that the text mentions but the ground truth does not contain.

Rules

- Mark an intent or detail as covered only when the text gives explicit textual evidence. Do not over-infer.
- If the text touches on a topic generally without specifying parameters, it still counts as covering the corresponding intent.
- If the text references content related to a detail with different surface wording, the detail still counts as covered.
- Any intent or detail that appears in the text but not in the ground-truth list must be reported under `extra_intents` or `extra_details`.
- Return only the JSON object shown below. Do not output any explanation or commentary outside the JSON.

Output format (strict JSON).

```
{
  "covered_intent_ids": ["I1"],
  "covered_detail_ids": ["d1", "d3"],
  "extra_intents": [
    "Description of intents in the response that are not in the ground truth"
  ],
  "extra_details": [
    "Description of details in the response that are not in the ground truth"
  ]
}
```

[User Prompt]

The user's original query is: {query}

The text to evaluate is: {response}

Ground-truth intent list: {intents_json}

Ground-truth detail list: {details_json}

Figure 17: **Prompt 4: Information Extractor.**