CLEAT: LEARNING VALUE-GUIDED CLARIFICATION FOR INTERACTIVE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM agents can execute increasingly complex tasks, but effective assistance also requires aligning their actions with goals and constraints that users may leave unstated. We study this problem as *interactive intent alignment* through clarification. Yet clarification is useful only when the agent can turn the answer into better decisions. As training changes how the agent handles incomplete information and uses additional answers, the same clarification can become more or less useful. We introduce **CLEAT** (**CL**arification through valu**E**-guided **A**gent co-**T**raining), a framework that co-trains a clarification-value controller and the task policy from shared interaction outcomes. We formulate *residual query value* to measure the task utility gained by asking rather than proceeding under the current policy, and estimate it through paired rollouts from the same interaction history. These paired returns train the controller to predict clarification value and the task policy to select useful questions and act on the answers. The controller guides subsequent rollout collection, while outcomes from the updated agent provide new value labels that allow clarification guidance to adapt as the agent learns. With Qwen3-4B, CLEAT achieves the highest mean score in all ten evaluation settings across four benchmarks. Gains over the strongest baseline reach 13.2 and 6.5 percentage points on UserGym Travel and IN3 underspecification judgment accuracy, respectively. Ablations show that co-training improves the task policy even when the controller is removed at inference, while adapting clarification guidance to the evolving policy yields further gains. Code is available here.

1 INTRODUCTION

Large language model (LLM) agents have made rapid progress in planning and tool use, enabling them to carry out increasingly complex tasks (Zhou et al., 2024; Yang et al., 2024). However, stronger execution alone does not ensure that an agent's actions reflect what the user actually wants. In practice, users may leave important constraints unstated, rely on mistaken assumptions, or struggle to articulate their preferences (Qian et al., 2024; Vijayvargiya et al., 2026), creating a gap between their explicit requests and intended goals. Effective assistance therefore requires agents to help close this gap through interaction, rather than expect users to specify everything upfront. We study this problem as *interactive intent alignment*, where agents align their task decisions with the user's intended goals and constraints through interaction.

Clarification can support this alignment by resolving uncertainty about the user's goals and constraints.[1] Yet clarification can reveal more about the user without improving the agent's decisions, while still requiring the user's time and attention. For example, suppose an agent must choose between two flights operated by the same airline but arriving at different times. Asking about airline preference may reveal more about the user, but it does not help distinguish between the two options. By contrast, asking about an unstated arrival deadline can determine which flight meets the user's needs (Figure 1). The challenge is therefore to recognize when clarification can improve the task outcome and what information the agent needs to obtain.

Prior work guides clarification through information-based signals (Deng et al., 2026; Kong et al., 2026), value-of-information estimates (Rao & Daumé III, 2018; Dong et al., 2026), and rewards

[1] We use *clarification* broadly to refer to questions an agent asks before acting, whether to fill in missing information, resolve ambiguity, or challenge a questionable premise.

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.

from subsequent interaction outcomes (Zhang et al., 2025; Wu et al., 2025). Despite this progress, identifying which clarifications help the agent remains challenging, particularly while its policy is being trained. **First, a clarification can appear useful without actually improving the task outcome.** A question may reveal additional information or be followed by a successful outcome, yet the same agent might have achieved an equally good result by proceeding on its own. **Second, clarification value can change even at a fixed interaction history.** Training can improve how the agent uses an answer and how it proceeds on its own by different amounts. The incremental value of the same clarification can therefore change as the policy learns, requiring renewed value supervision.

To address these challenges, we introduce CLEAT, which couples clarification-value learning with task-policy learning through paired interaction outcomes. For each evaluated clarification target, paired rollouts compare asking with native continuation under the current policy from the same interaction history. Native continuation retains the agent's own ability to ask, so the utility difference measures the added benefit of the targeted clarification. We call this difference the *residual query value* (RQV). The same outcomes provide controller value targets and policy advantages; for each target, the mean query advantage equals its RQV label. During training, controller predictions prioritize which comparisons to collect, while sampling-corrected policy updates use observed task returns. Fresh outcomes renew value supervision as the agent learns. At inference, the trained agent acts on clarification answers, while the controller guides when and what to ask using predicted RQV.

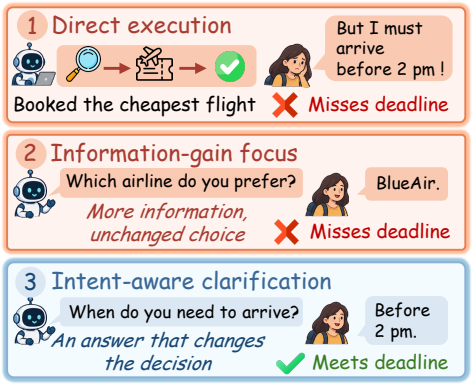

Figure 1: **Interactive intent alignment.** CLEAT targets questions that improve task outcomes, not merely reduce uncertainty.

We evaluate CLEAT on four benchmarks spanning personalized decision making, user-in-the-loop tool use, and requests with missing information or false premises. With Qwen3-4B as the backbone, CLEAT achieves the highest mean score in all ten evaluation settings. Gains over the strongest baseline reach 13.2 and 6.5 percentage points on UserGym Travel and IN3 underspecification judgment accuracy, respectively. Ablations show that the trained policy improves over the original backbone even without controller-based action selection at inference. Updating the controller during policy training also yields higher mean scores than freezing it in all six ablation settings.

Our contributions are summarized as follows.

- We construct **paired policy-relative supervision** from matched ask-versus-proceed outcomes, providing controller value targets and task-policy advantages relative to the agent's native continuation.
- We develop **value-guided co-training**, in which predicted values guide rollout collection, observed returns drive sampling-corrected policy updates, and fresh outcomes renew value supervision as the agent learns.
- We demonstrate **gains beyond inference-time guidance**, with ablations showing that co-training improves the task policy itself and that updating clarification values provides benefits beyond fixed guidance.

2 Preliminaries

2.1 Interactive Intent Alignment

We consider interactive tasks in which the user's initial request may not fully or accurately express the goals and constraints that determine success. For example, a travel assistant may need to establish whether the user prioritizes price or ticket flexibility, while a service agent may need to identify which reservation the user wants to change. Clarification can help resolve these gaps, but its purpose is to inform task decisions rather than serve as an objective in itself.

Formally, let G denote the user's latent goal and h the interaction history, which provides only partial information about G. The task policy π_θ selects actions based on h and may ask for clarification

when permitted by the environment, with replies added to h. We evaluate the resulting episode τ by its task utility $U(\tau, G)$, including any benchmark-specified interaction costs. To compare asking with proceeding, we consider their *continuations* from h, each comprising the remainder of an episode generated under π_θ. During training, we assume access to an environment that can sample multiple continuations from the same h and G, with the policy held fixed within each comparison.

2.2 Group-Relative Policy Optimization

To train the task policy from these sampled outcomes, we build on Group Relative Policy Optimization (GRPO) (Shao et al., 2024). GRPO estimates advantages by normalizing returns within a group, without a learned critic. In our setting, consider n continuations sampled under the matched conditions above, yielding completed episodes $\tau_1, \ldots, \tau_n$ with utilities $U_i = U(\tau_i, G)$. Standard GRPO constructs the group-relative advantages as

$$A_i = \frac{U_i - \bar{U}}{s_U + \delta}, \qquad \bar{U} = \frac{1}{n}\sum_{j=1}^{n} U_j, \tag{1}$$

where s_U is the standard deviation of the group utilities and $\delta > 0$ ensures numerical stability. These advantages are used in a clipped policy-ratio objective with KL regularization toward a reference policy. Our method retains policy-ratio clipping and reference-policy regularization while modifying how rollouts are collected and how their utilities are used to construct advantages. Implementation details are provided in Appendix B.

3 CLEAT

CLEAT co-trains a task policy and a clarification-value controller from paired rollouts. At each training history, we compare asking about selected targets with letting the current policy continue on its own. The observed returns provide value labels for the controller and advantages for updating the task policy. The updated controller guides target selection in the next round, and new rollouts under the updated policy provide fresh value labels (Figure 2).

3.1 Problem Setup and Residual Query Value

At an interaction history h, we compare letting the task policy proceed (PASS) with first asking for additional information (QUERY(z)). Here z identifies what information to request, such as a travel date or budget. Choosing PASS delegates the remaining interaction to the task policy, which retains its own ability to ask questions.

A fixed target constructor supplies the finite candidate set $\mathcal{Z}(h)$. A history is *clarification-eligible* when another user exchange is permitted and this set is nonempty. For QUERY(z), a fixed question realizer turns the target into a question, and the question–answer exchange is appended to h before task execution resumes. This separates the decision of what to ask from question wording.

For a fixed task policy π, let $V^\pi(h)$ denote the expected utility of choosing PASS and $Q^\pi(h, z)$ that of choosing QUERY(z). Both continuations use the same policy π after the initial action. Their difference defines the *residual query value* (RQV),

$$g_\pi(h, z) = Q^\pi(h, z) - V^\pi(h). \tag{2}$$

The expectations average over the latent goal $G \mid h$ and rollout randomness. RQV measures what a targeted clarification adds beyond native continuation, including any questions the agent would ask on its own. Benchmark-specified interaction costs are included in utility.

Information gain, $\mathrm{IG}(h, z) = I(G; Y_z \mid h)$, measures what an answer Y_z reveals about the user's goal. An informative answer need not improve the agent's decisions, so IG and RQV can rank clarification targets differently. This reflects the classical distinction between information gain and decision value (Howard, 1966).

RQV also depends on how the policy uses the answer. Ignoring it may leave utility unchanged, while misusing it can hurt even without interaction costs. Classical value of information evaluates an optimal decision maker; RQV evaluates the policy being trained. Training can improve answer

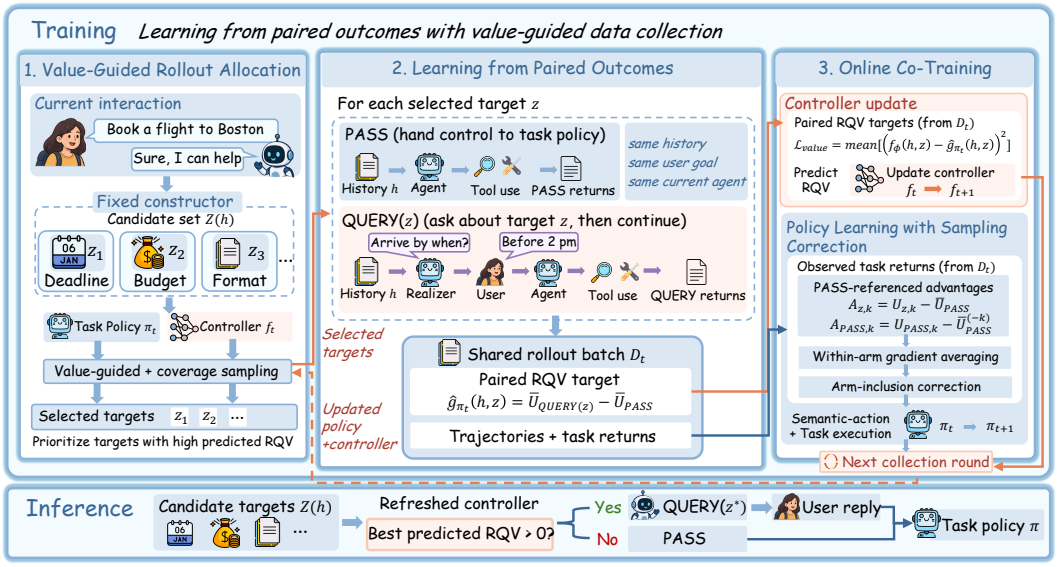

Figure 2: **Overview of CLEAT.** Value-guided paired rollouts train the controller and task policy from shared outcomes. At inference, the controller gates clarification by predicted incremental utility.

use and native continuation by different amounts, changing RQV even for a fixed history and target. We therefore renew value supervision as the policy learns (Appendix A.1).

3.2 Learning from Paired Outcomes

We estimate RQV by comparing $\textsc{query}(z)$ and $\textsc{pass}$ continuations from the same history h and latent user goal G. We hold the task policy π fixed throughout the comparison. After the initial action, both branches follow π without further controller intervention. Writing $\bar{U}_a$ for the mean task utility of continuations collected for action a, the paired estimate is

$$\widehat{g}_\pi(h, z) = \bar{U}_{\textsc{query}(z)} - \bar{U}_{\textsc{pass}}. \tag{3}$$

Averaging these labels over rollout randomness and $G \mid h$ recovers $g_\pi(h, z)$ (Appendix A.2). These measured gains provide the training labels for a lightweight controller $f_\phi(h, z)$. The controller predicts RQV from the history and candidate target, without observing the latent goal. At training step t, we fit it to labels collected under π_t by minimizing

$$\mathcal{L}_{\text{value}}(\phi; \mathcal{D}_t) = \frac{1}{|\mathcal{D}_t|} \sum_{(h,z)\in\mathcal{D}_t} \left(f_\phi(h, z) - \widehat{g}_{\pi_t}(h, z)\right)^2, \tag{4}$$

where $\mathcal{D}_t$ contains the current batch of history–target pairs and their labels. Once fitted, the controller scores candidates without running paired continuations for each one. We use these predictions to allocate subsequent rollouts.

3.3 Value-Guided Rollout Allocation

Under a fixed rollout budget, sampling only from the task policy may overlook useful low-probability targets. We therefore compare a value-guided target z_g and, when available, a distinct coverage target z_c against a shared $\textsc{pass}$ reference. The guided draw prioritizes predicted RQV, while the coverage draw gathers evidence about candidates the controller may undervalue.

The task policy provides the starting distribution for both draws. It assigns probabilities $\pi_\theta^A(a \mid h)$ over $\textsc{pass}$ and the available $\textsc{query}(z)$ actions by scoring fixed text representations of these actions and normalizing across them, without an additional action head. Conditioning on asking gives the target distribution $p_\pi(z \mid h)$. We mix it with a uniform component to form a reference that keeps every candidate reachable,

$$\bar{p}_\pi(z \mid h) = (1 - \epsilon)p_\pi(z \mid h) + \frac{\epsilon}{|\mathcal{Z}(h)|}, \qquad 0 < \epsilon < 1.$$

For the guided target z_g, we choose a sampling distribution μ that balances predicted RQV against deviation from the reference $\bar{p}_\pi$,

$$\max_{\mu \in \Delta(\mathcal{Z}(h))} \mathbb{E}_{z\sim\mu}[f_\phi(h,z)] - \beta^{-1} D_{\mathrm{KL}}(\mu \| \bar{p}_\pi).$$

The resulting proposal is

$$\mu^*(z \mid h) \propto \bar{p}_\pi(z \mid h) \exp\{\beta f_\phi(h,z)\}, \tag{5}$$

where $\beta > 0$ controls the strength of value guidance. We sample $z_g \sim \mu^*(\cdot \mid h)$.

For the coverage target z_c, we sample from $\bar{p}_\pi(\cdot \mid h)$ renormalized over the remaining candidates $\mathcal{Z}(h) \setminus \{z_g\}$. This draw reserves rollout budget for another candidate without value guidance. At each history h, we collect $K_p \geq 2$ PASS continuations, $K_g \geq 1$ continuations for QUERY(z_g), and $K_c \geq 1$ continuations for QUERY(z_c). Each distinct semantic action is an arm. With only one candidate, its query arm receives $K_g + K_c$ continuations. The arms and replica counts are chosen before observing continuation outcomes (Appendix A.3).

3.4 Policy Learning with Sampling Correction

Controller predictions guide collection; observed returns provide policy advantages. We correct for nonuniform arm inclusion to preserve the collecting policy's action weighting at a fixed history.

Task-return advantages. Replacing the group-mean baseline in Equation 1 with PASS references gives the query and PASS advantages,

$$A_{z,k} = U_{z,k} - \bar{U}_{\text{PASS}}, \qquad A_{\text{PASS},k} = U_{\text{PASS},k} - \bar{U}^{(-k)}_{\text{PASS}},$$

where $U_{z,k}$ and $U_{\text{PASS},k}$ are the observed utilities of the kth continuation in the corresponding arms. The baseline $\bar{U}^{(-k)}_{\text{PASS}}$ is the mean return of the other PASS continuations. This leave-one-out reference prevents a PASS trajectory's own return from entering its baseline. We treat advantages as fixed during optimization and do not apply group standard-deviation normalization.

For each query arm, averaging these advantages gives exactly the RQV label in Equation 3. Thus, the controller learns the mean gain from asking about a target, while the policy learns from the return of each individual continuation. Both use observed outcomes; the controller's predictions serve only to guide collection during training.

Correcting arm selection. Guided collection changes both how likely an arm is to appear and how many continuations it receives. We first average gradient contributions within each arm, then weight each arm mean to recover the collecting policy's action probabilities. At a fixed history h, let $\mathcal{I}_h$ be the set of included arms and $\rho_h(a)$ the probability that arm a is included. A query arm is included when selected by either the guided or the coverage draw, while $\rho_h(\text{PASS}) = 1$. With collecting policy π_θ, the Horvitz–Thompson correction is

$$\widehat{H}(h) = \sum_{a \in \mathcal{I}_h} \frac{\pi^A_\theta(a \mid h)}{\rho_h(a)} \bar{F}_a, \tag{6}$$

where $\bar{F}_a$ is the mean policy-gradient contribution for arm a. Each continuation contributes its advantage multiplied by the gradient of the task policy's log-probability. This log-probability covers the initial semantic action and subsequent agent-generated tokens. We hold the correction weights fixed when forming the gradient estimate.

Proposition 1 (Policy Gradient Preservation under Guided Sampling). Fix a clarification-eligible history h and the collecting policy π_θ. Let $m_a(h)$ denote the expected gradient contribution of arm a under on-policy continuation sampling. Suppose every arm with $\pi^A_\theta(a \mid h) > 0$ has positive inclusion probability and

$$\mathbb{E}[\bar{F}_a \mid h,\ a \in \mathcal{I}_h] = m_a(h).$$

Then the corrected estimator satisfies

$$\mathbb{E}[\widehat{H}(h) \mid h] = \sum_a \pi_\theta^A(a \mid h)\, m_a(h),$$

recovering the expected gradient contribution under on-policy arm sampling at h.

Uniform mixing and the always-included PASS arm ensure positive inclusion probabilities. Conditional on h, G, and the selected arms, continuations are sampled independently under the collecting policy. Under this conditioning, each reference is independent of the continuation it evaluates, and the shared and leave-one-out PASS references have the same expectation across arms. The baseline terms therefore cancel in expectation under on-policy arm weighting. At a fixed history and collecting policy, the correction preserves the expected unclipped task-return gradient. Guidance can still affect finite-budget learning by changing estimator variance and the controller's training distribution. This guarantee excludes policy-ratio clipping and reference-policy regularization (Appendix A.4).

3.5 Online Co-Training and Inference

Online co-training. We repeat collection and update both components to track how clarification value changes as the task policy learns. First, we bootstrap f_0 on paired rollouts under π_0, drawing targets from $\bar{p}_{\pi_0}$ without value guidance. At step t, we hold both components fixed during collection. f_t guides target selection, and π_t generates the continuations. We then use the same batch to fit f_{t+1} to the paired value labels and update π_t to π_{t+1} using the sampling-corrected policy update.

Sharing the batch requires only one rollout phase for the two updates. It also introduces a one-update lag because f_{t+1} learns from outcomes under π_t and then guides π_{t+1}. After the final policy update, we hold π_T fixed and collect fresh paired rollouts to refresh the controller with labels generated under the final task policy. Algorithm 1 summarizes the procedure, and Appendix A.5 discusses the supervision lag.

Inference. At inference, we retain the target constructor and question realizer. While active, the refreshed controller asks about the highest-scoring target at each clarification-eligible history h only if its predicted RQV is positive,

$$z^\star = \arg\max_{z \in \mathcal{Z}(h)} f_\phi(h, z), \qquad a = \begin{cases} \text{QUERY}(z^\star), & f_\phi(h, z^\star) > 0, \\ \text{PASS}, & \text{otherwise.} \end{cases} \tag{7}$$

After a query, the controller repeats this rule at the next eligible history. Choosing PASS ends controller intervention for the rest of the interaction; the task policy continues and may still ask questions on its own. The rule uses predicted gains relative to continuation under π_T, with no paired rollouts or sampling correction at inference. We fix the threshold at zero as implied by the RQV definition, without validation tuning. To separate the policy's learned behavior from controller guidance, we also evaluate the trained policy with semantic actions selected by π_T^A instead of controller scores.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate on four benchmarks spanning ten settings. **UserGym** (Qian et al., 2025) comprises Travel, Turtle, Intention, and Telepathy, with Intention and Telepathy held out from training. **τ^2-Bench** (Barres et al., 2026) evaluates user–agent coordination and tool use in the Airline, Retail, and Telecom domains. **IN3** (Qian et al., 2024) evaluates whether an agent recognizes underspecified instructions and recovers missing task requirements. **AskBench** (Zhao et al., 2026) distinguishes requests with missing information in AskMind from requests with false premises in AskOverconfidence.

Baselines. We compare four groups of baselines. **Prompting-based methods** use fixed model weights and guide behavior through prompting or agent scaffolds, including Vanilla (Yang et al., 2025), ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023), Proactive CoT (Deng et al., 2023), and PRIME (Wang & Jiang, 2026). **Threshold-based methods** include CLAM (Kuhn et al., 2023),

Table 1: Task performance with Qwen3-4B is reported as mean±std over 3 seeds (percentages except cumulative Intention credits), **bold** and underlined values indicate the best and second-best means, respectively.

Method	UserGym				τ^2-Bench			IN3	AskBench	
	Travel	Turtle	*Inten.*	*Tele.*	Air.	Ret.	Telec.	Judg.Acc	AskMind	AskOv.
Prompting-based										
Vanilla	26.74±1.49	17.97±1.73	185.50±2.00	49.12±1.87	23.78±2.89	20.89±2.61	20.00±1.25	65.74±1.54	38.70±1.27	57.24±1.18
ReAct	28.66±1.87	19.93±1.63	192.17±3.81	53.66±2.44	26.67±2.89	24.54±1.29	20.83±2.77	70.99±1.42	44.34±2.33	56.55±1.18
Reflexion	29.34±1.76	21.70±3.96	195.67±3.95	52.03±4.28	26.67±1.55	24.26±1.89	21.33±2.20	71.61±4.18	45.01±1.59	58.75±1.52
Proactive CoT	27.01±0.32	23.96±4.06	176.17±3.41	52.97±2.84	25.44±3.03	27.50±2.50	20.83±1.44	69.14±2.33	42.58±0.57	57.83±1.23
PRIME	28.20±0.59	19.83±2.58	173.17±2.74	53.22±4.45	25.00±0.00	25.83±1.44	18.33±1.44	68.83±1.42	41.24±0.87	58.25±1.00
Threshold-based										
CLAM	46.23±2.88	10.56±2.65	200.00±3.95	20.33±1.41	28.53±2.89	26.33±2.20	21.67±2.20	71.67±1.07	45.01±1.10	59.50±1.25
INTENT-SIM	27.59±1.54	18.78±1.26	198.42±2.32	52.03±1.41	33.33±2.89	27.17±2.46	18.33±2.89	73.06±2.95	42.07±2.06	59.00±1.30
VoI-Gate	49.19±1.67	21.15±2.45	198.08±3.68	38.96±2.63	31.67±2.77	27.50±3.01	21.17±1.82	71.91±1.42	44.64±2.03	57.17±1.88
Information-based training										
IG-Reward	43.75±1.20	28.58±2.99	194.68±2.43	54.22±3.45	25.00±2.50	17.99±3.82	22.06±2.89	66.67±4.24	42.87±1.48	58.92±1.76
InfoPO	35.41±1.13	25.52±2.38	198.75±3.88	53.66±3.14	25.00±0.00	30.02±1.84	22.50±1.25	68.21±5.35	46.32±2.61	60.15±1.77
IGPO	31.62±0.13	18.75±0.63	194.25±3.53	52.03±3.84	26.67±3.64	26.67±2.89	16.67±2.89	66.67±2.83	45.31±4.10	58.58±0.29
Supervised and outcome-based training										
SFT	50.48±1.76	27.92±3.11	198.25±4.72	38.96±3.63	25.77±1.82	26.80±3.12	21.83±3.20	77.47±5.26	43.92±1.19	58.50±1.30
GRPO	42.26±0.94	28.96±2.29	196.42±1.51	54.85±3.08	24.33±3.23	27.50±4.33	24.17±2.89	66.05±3.25	39.65±0.35	58.67±0.63
CLEAT (ours)	**63.72**±0.48	**29.79**±1.15	**209.17**±3.52	**57.67**±2.45	**35.00**±2.00	**32.50**±1.29	**24.94**±1.77	**83.95**±1.42	**48.37**±2.66	**61.27**±2.19

INTENT-SIM (Zhang & Choi, 2025), and VoI-Gate, which score ambiguity, intent uncertainty, and expected clarification utility, respectively. We tune their decision thresholds on validation data. VoI-Gate is our threshold-based adaptation of value-of-information reasoning (Dong et al., 2026). **Information-based training methods** include IG-Reward (Deng et al., 2026), InfoPO (Kong et al., 2026), and IGPO (Wang et al., 2026), which use information-derived signals either alone or alongside task-return supervision. **Supervised and outcome-based training methods** include SFT, which learns from demonstrations, and GRPO (Shao et al., 2024), which constructs group-relative advantages from task returns.

Evaluation and metrics. All main comparisons use Qwen3-4B as the backbone. We use the official evaluation harnesses with our configured turn budgets and report main results as mean and standard deviation over 3 seeds. Table 1 reports UserGym Pass-1 scores; domain-level task pass rates on τ^2-Bench; underspecification judgment accuracy (Judg.Acc) on IN3; and answer accuracy (Acc) on AskBench. All columns are percentages except UserGym Intention, whose per-detail credits accumulate within an episode. Benchmark-specific settings, baseline implementations, and metric definitions are detailed in Appendix C.

4.2 RQ1: Overall Performance

CLEAT achieves the highest mean score in all ten settings in Table 1, including gains of 13.24 points on Travel and 6.48 points in IN3 judgment accuracy over the strongest baselines. Baseline strengths vary across settings. SFT leads the baselines on Travel and IN3, while InfoPO leads on AskBench and GRPO on Turtle and Telepathy. CLEAT improves across these different interaction demands using the same paired-outcome learning procedure. The gains on Intention and Telepathy further support transfer to held-out environments without additional training of either the policy or controller.

On AskBench and IN3, CLEAT also achieves higher mean scores than ACT (Chen et al., 2025) and CollabLLM (Wu et al., 2025), which learn from interaction outcomes, across all six reported metrics (Table 8). We next examine how renewing value supervision and guiding collection contribute to these gains, and how much the trained policy retains without controller guidance at inference.

4.3 RQ2: Where the Gain Comes From

RQV compares targeted clarification with continuation under the current policy. Better answer use can increase the benefit of asking, while stronger native continuation can reduce it. This motivates renewing value supervision as the policy learns. We test its practical benefit by freezing the controller after bootstrap while continuing task-policy training. Freezing lowers mean scores by 1.27–6.70 points across all six settings (Figure 3, bottom). Both variants retain controller-guided collection, so this comparison assesses the benefit of updating the controller within the training loop. The results support renewing supervision over reusing guidance fitted to the initial backbone. All variants retain policy learning from observed task returns, with no additional reward from the controller's predictions.

Removing value guidance during collection reduces mean scores by 1.97–8.33 points (Figure 3, bottom). This variant retains paired comparisons, task-return advantages, sampling correction, and controller updates, with five continuations per eligible history. These gains support value-guided allocation for joint policy and controller learning under a fixed rollout budget.

To test the guidance signal, we replace predicted RQV with an IG proxy in the same proposal rule. RQV guidance achieves higher mean scores than both unguided collection and IG guidance in all six settings (Figure 3, top). Relative to unguided collection, IG guidance improves four settings but reduces Airline and Telecom by 3.34 and 2.00 points. With the proposal rule and task-return objective retained, this comparison tests which training outcomes the guidance signal favors. Under a shared turn budget, informative questions may leave fewer turns for execution; this is one possible explanation for the degradation on these two settings. RQV instead prioritizes the estimated benefit relative to native continuation while keeping policy updates grounded in observed returns.

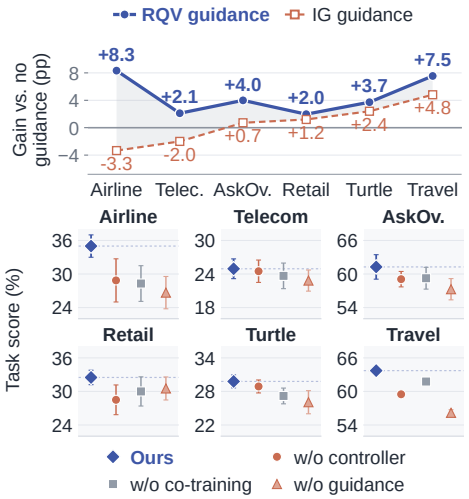

Figure 3: Effects of rollout guidance (top, gains over unguided collection in percentage points) and component ablations (bottom, mean±std over 3 seeds) on task performance.

The resulting gains also persist in the trained policy. Replacing the controller with policy-based semantic action selection at inference reduces mean scores by 0.44–6.14 points, but the policy still exceeds the strongest baseline on Travel (59.49% vs. 50.48%). In five of six settings, the mean drop from removing collection guidance exceeds that from removing controller-based selection at inference. This pattern highlights the controller's role in shaping training experience alongside its deployment role. Comparisons using the same constructor and realizer also support gains beyond the menu interface (Appendix D.1).

4.4 RQ3: What the Agent Resolves

Task success alone does not reveal how completely an agent resolves the user's request. Figure 4 examines this distinction. On IN3, *instruction recovery* (Recover) measures the importance-weighted fraction of missing details resolved. On AskBench, *clarification coverage* (Cov) measures the fraction of tasks with *every* required point resolved, including missing information in AskMind and false premises in AskOverconfidence. CLEAT achieves the highest mean on all three metrics, with recovery of 67.58% and coverage of 34.87% and 51.49%, respectively. Because coverage requires resolving all required points, these gains reflect more complete clarification, including correction of false premises rather than only elicitation of missing details.

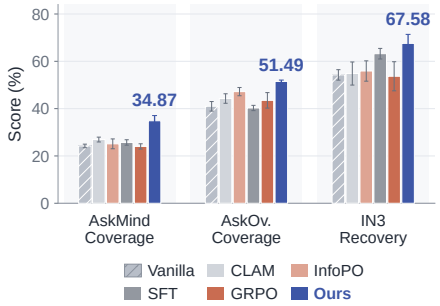

Figure 4: Instruction recovery on IN3 and complete clarification coverage on AskBench measure how fully agents resolve user requests (mean±std over 3 seeds).

Yet coverage remains below answer accuracy on both AskBench settings. An agent can therefore score well on final answers while leaving parts of a request unresolved. IN3 separates another pair of capabilities. Judgment accuracy evaluates whether the agent asks on underspecified instructions and proceeds on fully specified ones, while instruction recovery measures how much of the missing information the exchange resolves. Improvements in both support gains in recognizing clarification needs as well as recovering task-relevant details. Identifying an underspecified request alone does not ensure that subsequent questions obtain the

information needed to complete it. These metrics capture distinct aspects of interactive intent alignment, from recognizing a need to clarify to resolving requirements and producing a correct answer.

4.5 RQ4: Performance across Model Scales

At each model size (Qwen3-1.7B, 4B, and 8B), we compare CLEAT with the backbone and GRPO on AskBench, matching continuation budgets per clarification-eligible history for the two training methods. CLEAT achieves higher mean accuracy and coverage than both baselines at all three sizes (Figure 5). At 8B, accuracy exceeds GRPO by 2.08 points on AskMind and 2.50 points on AskOverconfidence. Coverage improves by 4.74 and 5.66 points, respectively. The benefit of clarification learning thus persists as the underlying policy becomes stronger within the tested range.

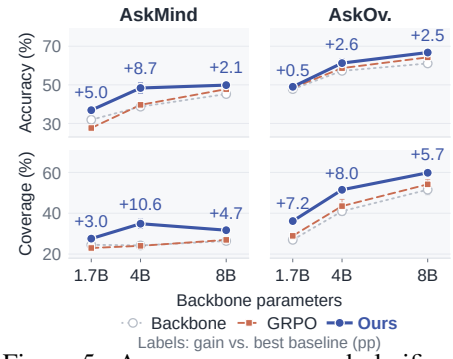

Figure 5: Answer accuracy and clarification coverage on AskBench, with gains over the stronger baseline in points.

Model scale does not improve both metrics in lockstep. For CLEAT on AskMind, mean answer accuracy increases with size, while mean coverage peaks at 4B rather than 8B. These mean trends suggest that stronger answering ability alone does not ensure more complete clarification.

5 Related Work

Clarification and intent alignment. Prior work studies clarification in ambiguous question answering (Kuhn et al., 2023; Zhang & Choi, 2025), code generation with ClarifyGPT (Mu et al., 2024), and planning for underspecified requests with Ask-before-Plan (Zhang et al., 2024). Ambiguity-based methods guide whether to ask, while value-of-information methods weigh expected benefits against interaction costs (Howard, 1966; Rao & Daumé III, 2018; Dong et al., 2026).

Learning from interaction outcomes. Zhang et al. (2025) derive preference labels from simulated future turns, while CollabLLM (Wu et al., 2025) trains with multiturn-aware rewards. STaR-GATE (Andukuri et al., 2024) improves question generation by selecting questions that increase the likelihood of high-quality personalized responses. ACT (Chen et al., 2025) constructs contrasting dialogue-action pairs and updates preference data using on-policy conversation simulations.

Information-based rewards and guided collection. IGPO rewards increases in answer likelihood (Wang et al., 2026), while clarification-specific information rewards measure changes in beliefs about the user's goal (Deng et al., 2026). InfoPO (Kong et al., 2026) measures changes in next-action likelihood under observed versus masked feedback and combines this signal with outcome-based advantages. In CLEAT, predicted RQV guides rollout collection, while observed returns drive policy updates corrected for nonuniform arm inclusion. This connects the collection strategy to experience prioritization (Schaul et al., 2016).

6 Conclusion, Limitations and Future Work

We presented CLEAT for interactive intent alignment through value-guided clarification. Paired interaction outcomes supervise both a clarification-value controller and the task policy, while renewed supervision tracks the policy's changing capabilities. Across four benchmarks, CLEAT improves task performance and clarification quality with Qwen3-4B.

The fixed target constructor and question realizer restrict the controller's available clarification choices. Training also relies on repeated continuations from the same history and user goal, an assumption that is difficult to satisfy in live interactions. Our evaluation uses simulated users, so effectiveness with real users and evolving goals remains to be established. Future work could jointly learn target construction and question realization, and develop value supervision that requires fewer paired rollouts.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

We used large language models to polish the writing and check grammar. The authors are responsible for the final manuscript.

ETHICS STATEMENT

Our experiments use existing benchmarks and simulated users. Real-world use should respect user privacy and avoid unnecessary clarification.

REPRODUCIBILITY STATEMENT

We provide implementation details, hyperparameters, and evaluation protocols in the appendix. Code is available at the anonymized repository linked in the abstract.

REFERENCES

Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah D. Goodman. Star-gate: Teaching language models to ask clarifying questions. In *COLM*, 2024.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. In *ICML*, 2026.

Maximillian Chen, Ruoxi Sun, Tomas Pfister, and Sercan Ö. Arık. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *ICLR*, 2025.

Mengyi Deng, Zhiwei Li, Xin Li, Tingyu Zhu, Ying Zhao, Zhijiang Guo, and Wei Wang. Uncertainty-aware clarification in llm agents with information gain. In *ICML*, 2026.

Yang Deng, Lizi Liao, Liang Chen, Hongru Wang, Wenqiang Lei, and Tat-Seng Chua. Prompting and evaluating large language models for proactive dialogues: Clarification, target-guided, and non-collaboration. In *EMNLP*, 2023.

Yijiang River Dong, Tiancheng Hu, Zheng Hui, Caiqi Zhang, Ivan Vulić, Andreea Bobu, and Nigel Collier. Value of information: A framework for human-agent communication. In *ACL*, 2026.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 1966.

Fanqi Kong, Jiayi Zhang, Mingyi Deng, Chenglin Wu, Yuyu Luo, and Bang Liu. Infopo: Information-driven policy optimization for user-centric agents. *arXiv preprint arXiv:2603.00656*, 2026.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Clam: Selective clarification for ambiguous questions with generative language models. *arXiv preprint arXiv:2212.07769*, 2023.

Fangwen Mu, Lin Shi, Song Wang, Zhuohao Yu, Binquan Zhang, Chenxue Wang, Shichao Liu, and Qing Wang. Clarifygpt: Empowering llm-based code generation with intention clarification. *PACMSE*, 2024.

Cheng Qian, Bingxiang He, Zhong Zhuang, Jia Deng, Yujia Qin, Xin Cong, Zhong Zhang, Jie Zhou, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Tell me more! towards implicit user intention understanding of language model driven agents. In *ACL*, 2024.

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*, 2025.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *ACL*, 2018.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2016.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*, 2023.

Sanidhya Vijayvargiya, Xuhui Zhou, Akhila Yerukola, Maarten Sap, and Graham Neubig. Ambigswe: Interactive agents to overcome underspecificity in software engineering. In *ICLR*, 2026.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *ICLR*, 2026.

Prince Zizhuang Wang and Shuli Jiang. Prime: Training free proactive reasoning via iterative memory evolution for user-centric agent. *arXiv preprint arXiv:2604.07645*, 2026.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. Collabllm: From passive responders to active collaborators. In *ICML*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. In *NeurIPS*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

Michael J. Q. Zhang and Eunsol Choi. Clarify when necessary: Resolving ambiguity through interaction with lms. In *ACL*, 2025.

Michael J. Q. Zhang, W. Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach llms to ask clarifying questions. In *ICLR*, 2025.

Xuan Zhang, Yang Deng, Zifeng Ren, See-Kiong Ng, and Tat-Seng Chua. Ask-before-plan: Proactive language agents for real-world planning. In *EMNLP*, 2024.

Jiale Zhao, Ke Fang, and Lu Cheng. When and what to ask: Askbench and rubric-guided rlvr for llm clarification. In *ACL*, 2026.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *ICLR*, 2024.

APPENDIX CONTENTS

A Additional Theory and Method Details **13**

A.1 Defining Policy-Relative Clarification Value 13

A.2 Estimating RQV from Paired Continuations 13

A.3 Value-Guided Collection under a Fixed Budget 15

A.4 Policy Learning with Sampling Correction 16

A.5 Online Co-Training and Inference 19

B Implementation Details **20**

C Experimental Details **22**

C.1 Benchmarks and Evaluation Metrics 22

C.2 Baselines 23

D Additional Experimental Results **24**

D.1 Component Ablations 24

D.2 Comparison with Larger Backbones 25

D.3 CLEAT across Backbone Sizes 25

D.4 Clarification-Specific Metrics 26

D.5 Comparison with ACT and CollabLLM 26

D.6 Asking on Underspecified Instructions 26

D.7 Co-Training Dynamics 27

D.8 Qualitative Examples of Policy Behavior 28

D.9 Cross-Method Clarification Cases 28

E Prompts **29**

A Additional Theory and Method Details

CLEAT learns clarification value relative to the task policy and uses that value to guide the experiences from which the policy learns. This appendix develops the reasoning behind this design. We first explain what the value comparison should measure and how paired continuations provide its supervision. We then derive the rollout allocation rule and the sampling correction needed to learn from these selectively collected outcomes. Finally, we connect the two learning processes through co-training as the task policy evolves.

A.1 Defining Policy-Relative Clarification Value

To learn useful clarification, we need to distinguish the utility achieved after an exchange from the utility added by that exchange. A successful outcome alone cannot make this distinction because the agent may already have been able to succeed by proceeding on its own. We therefore compare a targeted clarification with native continuation from the same interaction history under the same task policy.

We identify a clarification by its information target z, such as a travel date or a budget, rather than by its surface wording. At history h, a fixed constructor supplies candidate targets, and a fixed realizer turns a selected target into a question. The comparison is between $\textsc{query}(z)$, which obtains the corresponding user response before task execution resumes, and PASS, which delegates the continuation to the task policy, including its native ability to ask. Let $Q^\pi(h, z)$ and $V^\pi(h)$ denote the expected task utilities of these two continuations under policy π. Their difference defines residual query value,

$$g_\pi(h, z) = Q^\pi(h, z) - V^\pi(h).$$

This reference makes clarification value specific to both the information requested and the policy that will act on the answer. Both branches use the same task utility, including any benchmark-specified interaction costs.

Information and decision value can rank questions differently. To see why this comparison differs from information gain, consider a zero-cost, single-decision task with latent goal $G = (X, W)$, where $X \sim \mathrm{Bernoulli}(1/2)$ and $W \sim \mathrm{Unif}\{1, \ldots, K\}$ are independent, with $K > 2$. The agent receives reward one for predicting X correctly and zero otherwise. Revealing W provides information gain $\log K$ but leaves the best achievable expected reward at $1/2$, whereas revealing X provides only $\log 2$ information gain but raises that reward to one. The more informative question therefore need not provide greater task value, even when both answers are reliable and free to obtain.

Even when the answer is task-relevant, however, its benefit still depends on the policy receiving it. Keeping the same task and question, consider two policies that both predict zero without clarification. After observing X, one follows the answer and achieves reward one, whereas the other ignores it and retains expected reward $1/2$. Their residual query values are therefore $1/2$ and zero, respectively, despite receiving exactly the same information. This difference motivates evaluating clarification through the outcomes of the current task policy rather than through the information revealed alone.

Relation to classical value of information. Unlike classical value of information, which evaluates the benefit to an optimal decision maker, RQV measures the benefit under the policy that actually continues the task. Because training can improve the QUERY and PASS continuations by different amounts, the value of the same clarification can change even when the information it reveals remains fixed. We therefore estimate this policy-relative value from paired outcomes generated by the policy being trained.

A.2 Estimating RQV from Paired Continuations

To turn this policy-relative comparison into a learning signal, we collect paired continuations that share the same history, latent user goal, and task policy. Holding these quantities fixed isolates the initial clarification decision from differences in the task or the agent. For each selected target, the QUERY branch obtains a user response before execution resumes, while the PASS branch delegates

directly to the task policy. Both then continue without further controller intervention, including any questions the policy asks on its own.

Specifically, at a history h, the constructor supplies a finite candidate set $\mathcal{Z}(h)$. When another user exchange is permitted and $\mathcal{Z}(h)$ is nonempty, the semantic action set is

$$\mathcal{A}(h) = \{\text{PASS}\} \cup \{\text{QUERY}(z) : z \in \mathcal{Z}(h)\}.$$

Otherwise, it reduces to $\{\text{PASS}\}$.

Target construction and realization. The candidate constructor is fixed across training and inference. It uses predefined information fields from the task interface where available; otherwise, a frozen model constructs a small target set from the current history. Target construction does not use hidden annotations, rollout outcomes, task rewards, or RQV labels. We hold the realized candidate set fixed within each comparison, so every branch starts from the same decision menu.

Let $G \sim p(G \mid h)$ denote the latent user goal, which is held by the environment rather than supplied to the policy or controller. Executing $\text{QUERY}(z)$ invokes a fixed realizer that produces a benchmark-native question, followed by a user response $y \sim p_U(\cdot \mid G, h, z)$. The exchange is appended to the history before task execution resumes. Its tokens are environment-side and masked from the task-policy loss, so the learnable decisions are which target to select and how to act on the answer. Keeping the constructor and realizer fixed separates these two sources of improvement from changes in candidate generation or question wording.

Utility and interaction cost. To compare the two branches on the same scale, we use the task utility

$$U(\tau, G) = R(\tau, G) - \sum_{z \in \mathcal{Q}(\tau)} c(z), \tag{8}$$

where $R(\tau, G)$ is the benchmark return, $\mathcal{Q}(\tau)$ collects clarification events with multiplicity, and $c(z) \geq 0$ is the interaction cost of an event. We set $c(z) = 0$ when no such cost is specified. Comparing QUERY and PASS under the same utility measures whether the information obtained is worth the interaction required to obtain it.

From branch utilities to a paired estimate. With this common utility, we can estimate the comparison by sampling both continuations under a fixed collecting policy π. Let τ_a denote a continuation whose initial semantic action is fixed to a and whose subsequent behavior follows π. Conditional on the same history and user goal, the two branch values are

$$\begin{aligned} Q^\pi(h, z; G) &= \mathbb{E}[U(\tau_{\text{QUERY}(z)}, G) \mid h, G], \\ V^\pi(h; G) &= \mathbb{E}[U(\tau_{\text{PASS}}, G) \mid h, G]. \end{aligned} \tag{9}$$

These expectations include the user response and all subsequent rollout randomness. Each group holds h and G fixed, and its arms and replica counts are chosen before observing continuation outcomes. We then sample continuations independently conditional on these quantities. If a selected target receives K_z query continuations with utilities $U_{z,k}$ and the shared reference contains K_p continuations with utilities $U_{\text{PASS},j}$, Equation 3 becomes

$$\widehat{g}_\pi(h, z) = \bar{U}_{\text{QUERY}(z)} - \bar{U}_{\text{PASS}} = \frac{1}{K_z} \sum_{k=1}^{K_z} U_{z,k} - \frac{1}{K_p} \sum_{j=1}^{K_p} U_{\text{PASS},j}.$$

Conditional on the allocated arms and their replica counts, each sample mean has expectation equal to its branch value. Averaging over the allocation therefore preserves these expectations, and linearity gives

$$\begin{aligned} \mathbb{E}[\widehat{g}_\pi(h, z) \mid h, G,\ z \text{ included}] &= \mathbb{E}[\bar{U}_{\text{QUERY}(z)} \mid h, G,\ z \text{ included}] \\ &\quad - \mathbb{E}[\bar{U}_{\text{PASS}} \mid h, G,\ z \text{ included}] \\ &= Q^\pi(h, z; G) - V^\pi(h; G). \end{aligned}$$

Thus matching the user goal makes the paired difference estimate the effect of the initial clarification, rather than a difference in what the two users wanted. To recover the history-conditioned RQV in Section 3.1, we next average over the latent goal. In CLEAT, target construction and selection use

the visible history, not the hidden goal or future outcomes, so including z does not change $p(G \mid h)$. The law of total expectation then yields

$$\begin{aligned}\mathbb{E}[\widehat{g}_\pi(h,z) \mid h,\ z \text{ included}] &= \mathbb{E}_{G|h}\big[Q^\pi(h,z;G) - V^\pi(h;G)\big] \\ &= g_\pi(h,z).\end{aligned} \tag{10}$$

Learning from the paired labels. Accordingly, we fit $f_\phi(h,z)$ to these paired differences using the squared-error objective in Equation 4. Squared-error regression targets the conditional mean of the label, which the derivation above identifies as RQV under the collecting policy. The controller thereby learns to predict the utility added by clarification, rather than the utility attained after asking alone, and scores new state–target pairs without observing the latent goal or collecting paired continuations at inference.

A.3 Value-Guided Collection under a Fixed Budget

Paired evaluation identifies the value of a selected target, but evaluating every candidate at every history would spread the rollout budget thinly. We instead combine value-guided target selection with a separate coverage draw and repeated continuations. These serve complementary purposes. Guidance prioritizes promising questions, coverage collects evidence beyond that prioritization, and replication improves the precision of the resulting comparisons.

Semantic-action probabilities. Before applying value guidance, we construct a reference distribution that represents the task policy's own preferences. The task policy defines a categorical policy over the menu $\mathcal{A}(h)$ without an additional action head. Each action has a nonempty canonical serialization $s(a)$, scored by its mean token log-probability conditioned on the history and available actions,

$$L_\theta(a \mid h) = \frac{1}{|s(a)|} \log \pi_\theta\big(s(a) \mid h,\, \mathcal{A}(h)\big),$$

where $|s(a)|$ is its token count. Normalizing these length-adjusted scores gives

$$\pi_\theta^A(a \mid h) = \frac{\exp L_\theta(a \mid h)}{\sum_{a' \in \mathcal{A}(h)} \exp L_\theta(a' \mid h)}. \tag{11}$$

This distribution provides both the policy reference for collection and the on-policy arm weights used in Proposition 1.

A policy-based reference with full support. Conditioned on asking, the task policy induces

$$p_\pi(z \mid h) = \frac{\pi_\theta^A(\text{QUERY}(z) \mid h)}{\sum_{z'} \pi_\theta^A(\text{QUERY}(z') \mid h)}.$$

We use this distribution as the starting point for exploration, retaining the policy's preferences among targets. To keep low-probability candidates reachable, we mix it with a uniform distribution,

$$\bar{p}_\pi(z \mid h) = (1-\epsilon)p_\pi(z \mid h) + \frac{\epsilon}{|\mathcal{Z}(h)|}, \qquad 0 < \epsilon < 1.$$

This gives every candidate positive reference probability before applying value guidance.

Guided proposal. With this reference in place, we use the controller to concentrate part of the rollout budget on clarifications with higher predicted value. Rather than replace the policy's preferences with a greedy choice, we penalize departure from the reference. At fixed h and fixed collecting policy and controller, write $p(z) = \bar{p}_\pi(z \mid h) > 0$ and assume finite predicted values. For $\beta > 0$, the resulting tradeoff is

$$\max_{\mu \in \Delta(\mathcal{Z}(h))} \mathbb{E}_{z\sim\mu}[f_\phi(h,z)] - \beta^{-1} D_{\mathrm{KL}}(\mu \| p).$$

To derive the proposal, introduce a multiplier λ for the normalization constraint $\sum_z \mu(z) = 1$. Differentiating with respect to each positive $\mu(z)$ gives the stationarity condition and its solution,

$$\begin{aligned}0 &= f_\phi(h,z) - \frac{1}{\beta}\left(\log \frac{\mu(z)}{p(z)} + 1\right) + \lambda, \\ \mu(z) &= p(z)\exp\{\beta f_\phi(h,z) + \beta\lambda - 1\}.\end{aligned}$$

The multiplier contributes the same factor to every target. Enforcing normalization therefore yields

$$\mu^*(z \mid h) = \frac{p(z)\exp\{\beta f_\phi(h,z)\}}{\sum_{z'} p(z')\exp\{\beta f_\phi(h,z')\}},$$

which recovers Equation 5. Positive reference probabilities and finite predictions make this solution interior, and strict concavity of the objective makes it the unique maximizer. As $\beta \to 0$, the proposal approaches p; increasing β gives predicted value more influence relative to the reference. The tilt prioritizes predicted RQV, while its effect on actual task utility depends on how these predictions align with current-policy outcomes.

Coverage complements guidance. To obtain evidence beyond this prioritization, we reserve a separate draw for reference-based sampling. When at least two candidates are available, we first draw $z_g \sim \mu^*(\cdot \mid h)$, then draw a distinct target from the reference distribution over the remaining candidates,

$$\Pr(z_c = z \mid h, z_g) = \frac{\bar{p}_\pi(z \mid h)}{1 - \bar{p}_\pi(z_g \mid h)}, \qquad z \neq z_g.$$

Support smoothing and this second draw address different needs. Smoothing keeps every target reachable, whereas the coverage draw reserves rollout budget for sampling without the value tilt among the remaining targets. Excluding z_g ensures that this budget evaluates an additional candidate rather than replicating the guided choice. Guidance uses the controller's current estimates; coverage supplies comparisons that can revise them.

A shared reference for paired comparisons. Having chosen which targets to evaluate, we allocate continuations to the query arms and a shared PASS reference. Reusing the same PASS sample mean supports both comparisons without collecting a separate reference for each target. With at least two candidates, the group contains $K_p \geq 2$ PASS continuations, $K_g \geq 1$ continuations for the guided target, and $K_c \geq 1$ for the coverage target,

$$S_h = \{\text{PASS}^{\times K_p},\ \text{QUERY}(z_g)^{\times K_g},\ \text{QUERY}(z_c)^{\times K_c}\}.$$

Using at least two PASS replicas also allows each to use the others as its policy-learning baseline, as developed in the next subsection. When $|\mathcal{Z}(h)| = 1$, the single query target receives $K_g + K_c$ continuations. The GRPO comparison uses the same total continuation budget per clarification-eligible history. Experimental allocations are given in Appendix B.

Precision under a fixed budget. To quantify the effect of replication, fix the collecting policy and condition on $\mathcal{C}$, comprising the shared history h, latent goal G, selected arms, and replica counts. Let σ_z^2 and σ_P^2 denote the finite per-continuation return variances under QUERY(z) and PASS, respectively. Since continuations are sampled independently within and across arms, the variance of their sample-mean difference is

$$\begin{aligned}\text{Var}\big(\widehat{g}_\pi(h,z) \mid \mathcal{C}\big) &= \text{Var}\big(\bar{U}_{\text{QUERY}(z)} \mid \mathcal{C}\big) + \text{Var}\big(\bar{U}_{\text{PASS}} \mid \mathcal{C}\big) \\ &= \frac{\sigma_z^2}{K_z} + \frac{\sigma_P^2}{K_p},\end{aligned} \tag{12}$$

where K_z is the replica count of the selected query arm.

The two terms explain how replication improves the comparison. Increasing K_p reduces reference noise in both labels, whereas increasing K_z reduces the query-specific noise for target z. The two labels share reference noise, which cancels when comparing their empirical values with each other but still affects either comparison against proceeding. Under the fixed total budget, we assign more query replicas to the guided target, sharpening the prioritized comparison while retaining a coverage arm to evaluate another candidate.

A.4 Policy Learning with Sampling Correction

Guided collection changes which arms are observed and how many continuations each receives, but not the task utility that the policy is trained to improve. We therefore start from the task-return objective under the policy's own action distribution, then derive how the shared PASS reference and the sampling correction recover its gradient from the structured rollout groups.

The task-return objective. Fix the history h, candidate set, and conditional goal distribution $p(G \mid h)$. Let $P_\theta^a(\tau \mid h, G)$ denote the law of a continuation whose initial action is fixed to a and whose subsequent behavior follows the task policy. Conditional on G, the expected utility under the policy's own arm selection is

$$J_h(\theta; G) = \sum_{a \in \mathcal{A}(h)} \pi_\theta^A(a \mid h) \mathbb{E}_{\tau \sim P_\theta^a}[U(\tau, G)].$$

Averaging over the latent goal gives $J_h(\theta) = \mathbb{E}_{G|h}[J_h(\theta; G)]$. We first derive the update conditional on G and take this final average below. All gradients are evaluated at the collecting policy. We assume differentiable continuation laws with locally parameter-independent support and that differentiation can be interchanged with expectation; utility and environment kernels have no direct dependence on θ.

Both arm selection and subsequent execution depend on the task-policy parameters. Their joint score is

$$S_\theta(a, \tau \mid h, G) = \nabla_\theta \log \pi_\theta^A(a \mid h) + \nabla_\theta \log P_\theta^a(\tau \mid h, G). \tag{13}$$

The second term sums token scores over the agent's native continuation after the initial semantic action. Fixed question realization, user responses, and other environment transitions contribute no score, and G indexes the environment law rather than becoming an input to the policy. Applying the product rule and score identity to J_h therefore gives the gradient we aim to recover,

$$\nabla_\theta J_h(\theta; G) = \sum_a \pi_\theta^A(a \mid h) \mathbb{E}_{\tau \sim P_\theta^a}[U(\tau, G) S_\theta(a, \tau \mid h, G)].$$

PASS-referenced advantages. The paired group already supplies a common task-return reference. Reusing it for policy learning makes each query advantage measure utility relative to proceeding, consistently with the controller's supervision. For a query continuation, we use

$$A_{z,k} = U_{z,k} - \bar{U}_{\text{PASS}}.$$

For a PASS continuation, however, including its own return in the reference would make the baseline depend on the execution it evaluates. We therefore leave that continuation out,

$$A_{\text{PASS},k} = U_{\text{PASS},k} - \frac{1}{K_p - 1} \sum_{j \neq k} U_{\text{PASS},j}.$$

The mean query advantage is exactly the empirical RQV label in Equation 3, so the policy and controller learn from the same outcome comparison at trajectory and arm level, respectively. We treat advantages as fixed coefficients in the policy update and do not divide them by a group-dependent standard deviation.

Why the reference preserves the objective. Write $B_{a,k}$ for the reference subtracted from $U_{a,k}$. Conditional on h, G and the selected arms, $B_{a,k}$ is independent of its own continuation and has expectation $V^\pi(h; G)$ for every arm. This follows from independent continuation sampling, the fixed number of PASS replicas, and leave-one-out construction for the PASS arm itself. Because each continuation law is normalized, its expected score is zero,

$$\mathbb{E}_{\tau \sim P_\theta^a}[\nabla_\theta \log P_\theta^a(\tau \mid h, G)] = 0.$$

The independent baseline therefore contributes only the arm-selection score in expectation,

$$\mathbb{E}[B_{a,k} S_\theta(a, \tau_{a,k} \mid h, G) \mid h, G, \ a \text{ included}] = V^\pi(h; G) \nabla_\theta \log \pi_\theta^A(a \mid h).$$

Crucially, the expected reference is the same for all arms, so weighting by the policy's arm probabilities makes the remaining term cancel,

$$V^\pi(h; G) \sum_a \pi_\theta^A(a \mid h) \nabla_\theta \log \pi_\theta^A(a \mid h) = V^\pi(h; G) \nabla_\theta \sum_a \pi_\theta^A(a \mid h) = 0.$$

Thus replacing returns with PASS-referenced advantages preserves the gradient of $J_h(\theta; G)$ for both arm selection and execution. Although a PASS continuation has zero expected advantage, its advantage can still covary with the continuation score, so its execution gradient need not vanish. Native task behavior can therefore improve on PASS trajectories as well as on query trajectories.

Averaging within each arm. We now account for the two ways guided collection changes the data. First, an arm can receive more replicas without being intended to carry more weight in the task objective. We therefore average its contributions,

$$\bar{F}_a = \frac{1}{K_a} \sum_{k=1}^{K_a} A_{a,k} S_\theta(a, \tau_{a,k} \mid h, G).$$

Let $m_a(h, G)$ denote the expected contribution of one on-policy continuation of arm a with the same reference construction. Conditional on inclusion, the continuation law and expected reference are unchanged, whether the target was selected for guidance or coverage. Even when these roles receive different replica counts, we therefore have

$$\mathbb{E}[\bar{F}_a \mid h, G,\ a \text{ included}] = m_a(h, G).$$

The baseline calculation above establishes $\sum_a \pi_\theta^A(a \mid h) m_a(h, G) = \nabla_\theta J_h(\theta; G)$.

Correcting arm inclusion. Second, guided collection makes some arms more likely to appear in a group. Let $\mathcal{I}_h$ be the set of included arms and $\rho_h(a) = \Pr(a \in \mathcal{I}_h \mid h)$ their inclusion probabilities. For $|\mathcal{Z}(h)| \geq 2$, write $\mu(z) = \mu^*(z \mid h)$ and $p(z) = \bar{p}_\pi(z \mid h)$. A query arm enters either as the guided target or as the coverage target after a different guided draw. Summing these disjoint events gives

$$\rho_h(\text{QUERY}(z)) = \mu(z) + \sum_{z' \neq z} \mu(z') \frac{p(z)}{1 - p(z')}. \tag{14}$$

The PASS arm is always included, so $\rho_h(\text{PASS}) = 1$. When there is only one candidate, its query arm also has inclusion probability one.

Having removed replica-count weighting, we correct this second effect by weighting each included arm mean by $\pi_\theta^A(a \mid h)/\rho_h(a)$, as in Equation 6. Write $I_a = \mathbf{1}[a \in \mathcal{I}_h]$, with zero contribution for excluded arms. Target selection is independent of G conditional on h, so $\mathbb{E}[I_a \mid h, G] = \rho_h(a)$. Taking expectations makes the correction explicit,

$$\begin{aligned} \mathbb{E}[\widehat{H}(h) \mid h, G] &= \sum_a \frac{\pi_\theta^A(a \mid h)}{\rho_h(a)} \mathbb{E}[I_a \bar{F}_a \mid h, G] \\ &= \sum_a \frac{\pi_\theta^A(a \mid h)}{\rho_h(a)} \rho_h(a) m_a(h, G) \\ &= \sum_a \pi_\theta^A(a \mid h) m_a(h, G) = \nabla_\theta J_h(\theta; G). \end{aligned} \tag{15}$$

Finally, averaging over the fixed distribution $G \mid h$ gives

$$\mathbb{E}[\widehat{H}(h) \mid h] = \sum_a \pi_\theta^A(a \mid h) m_a(h) = \nabla_\theta J_h(\theta),$$

where $m_a(h) = \mathbb{E}_{G|h}[m_a(h, G)]$ and the objective $J_h(\theta)$ is the one defined at the start of this subsection. This proves Proposition 1 for the fixed-history task-return contribution of both action selection and execution.

All inclusion weights are treated as fixed when forming this estimator. The controller can therefore change which arms receive data without giving those arms extra weight in the expected task-return gradient at h. Reference-policy regularization and the subsequent optimization step are specified separately in Appendix B.

Coverage bounds the correction weights. The coverage draw also controls the size of the correction, independently of how strongly the controller concentrates the guided proposal.

Proposition 2 (Coverage-Induced Bounds on Correction Weights). Fix a clarification-eligible history h and $0 < \epsilon < 1$. Use the smoothed reference $p = \bar{p}_\pi$ and the guided-plus-coverage selection rule in Equation 14, including the single-candidate convention. For any guided distribution μ and every candidate z,

$$\frac{\pi_\theta^A(\text{QUERY}(z) \mid h)}{\rho_h(\text{QUERY}(z))} \le \frac{1}{1-\epsilon}, \qquad \frac{\pi_\theta^A(\text{PASS} \mid h)}{\rho_h(\text{PASS})} = \pi_\theta^A(\text{PASS} \mid h) \le 1. \tag{16}$$

Proof. For at least two candidates, $0 < p(z) < 1$, and Equation 14 implies

$$\rho_h(\text{QUERY}(z)) \ge \mu(z) + p(z) \sum_{z' \neq z} \mu(z') = p(z) + (1 - p(z))\mu(z) \ge p(z).$$

Since $p_\pi(z \mid h)$ conditions on asking, $p_\pi(z \mid h) \ge \pi_\theta^A(\text{QUERY}(z) \mid h)$. Support smoothing then gives

$$\rho_h(\text{QUERY}(z)) \ge p(z) \ge (1-\epsilon)p_\pi(z \mid h) \ge (1-\epsilon)\pi_\theta^A(\text{QUERY}(z) \mid h).$$

Dividing by the positive inclusion probability gives the query bound. The single-candidate case satisfies the same bound because its inclusion probability is one. The PASS bound follows from $\rho_h(\text{PASS}) = 1$. □

The bound depends neither on the guidance strength nor on the range of controller predictions. Thus, coverage serves both to broaden supervision and to keep guided collection compatible with bounded, untruncated correction weights. With our $\epsilon = 0.2$, every inclusion weight is at most 1.25.

Relation to information-based training signals. Information-based objectives provide intermediate supervision by measuring how an observation changes a reference quantity, such as the likelihood of a gold answer, the inferred user goal, or subsequent agent behavior (Wang et al., 2026; Deng et al., 2026; Kong et al., 2026). These signals can also be combined with task-return supervision. In CLEAT, predicted RQV instead guides which experiences are collected, while policy advantages use observed task utilities and the correction above preserves their intended arm weighting. The controller's prediction thus allocates the learning opportunity rather than becoming an additional reward, allowing the controller and task policy to learn different quantities from the same outcomes.

A.5 Online Co-Training and Inference

Co-training follows from the policy-relative definition of RQV. As the task policy learns, it can improve both its use of an answer and its native continuation, changing the comparison that the controller should predict. For a fixed history h and target z, subtracting Equation 2 across consecutive policy versions gives

$$g_{\pi_{t+1}}(h,z) - g_{\pi_t}(h,z) = \left[Q^{\pi_{t+1}}(h,z) - Q^{\pi_t}(h,z)\right] - \left[V^{\pi_{t+1}}(h) - V^{\pi_t}(h)\right].$$

Improved use of an answer can increase its residual value, whereas improved native execution can make the same exchange redundant. What matters is the relative change in the two branches, which motivates renewing the controller's supervision as the policy evolves rather than treating the values learned at bootstrap as fixed.

One collection phase, two updates. We implement this feedback loop through a shared rollout batch. First, we bootstrap f_0 on paired outcomes under π_0, drawing targets from the policy-based reference without value guidance. At step t, we hold π_t and f_t fixed during collection, so the controller guides target selection while all continuations follow the same task policy. The batch then provides paired RQV labels for fitting f_{t+1} and sampling-corrected task-return gradients for updating π_t to π_{t+1}. The controller learns the value of intervention, while the policy learns both target selection and subsequent task execution from observed returns. Controller fitting uses only

the current batch, keeping its labels tied to one collecting policy. The next round uses both updated components, so the controller shapes exploration and the policy's resulting behavior supplies fresh value supervision.

One-update lag and final refresh. Because both updates use outcomes generated before the policy step, f_{t+1} is fitted to g_{π_t} and then guides collection under π_{t+1}. This introduces a one-update lag between value supervision and the policy being guided. The next batch renews supervision under π_{t+1}, tying the targets to successive policy versions while retaining only one collection phase per training step.

After the final policy update, we hold π_T fixed and collect fresh paired outcomes solely to refresh the controller, without another policy update. Its regression targets are therefore generated by the same policy used at inference. Algorithm 1 summarizes bootstrap, joint updates, and final refresh, using the group allocation in Appendix A.3.

Algorithm 1 Online co-training in CLEAT

Require: task policy π_0, updates T, budgets (K_p, K_g, K_c)

Bootstrap f_0 on paired rollouts collected with $\mu = \bar{p}_{\pi_0}$
for $t = 0, 1, \ldots, T-1$ **do**
  Sample a batch $\mathcal{B}_t$ of clarification-eligible states (h, G)
  for each $(h, G) \in \mathcal{B}_t$ **do**
    $\mu_t(z \mid h) \propto \bar{p}_{\pi_t}(z \mid h) \exp\{\beta f_t(h, z)\}$
    Draw guided target $z_g \sim \mu_t(\cdot \mid h)$
    Draw a distinct coverage target from renormalized $\bar{p}_{\pi_t}$ when available
    Collect paired continuations S_h under π_t at the same (h, G)
    Compute $\widehat{g}_t(h, z) = \bar{U}_{\text{QUERY}(z)} - \bar{U}_{\text{PASS}}$ for each selected target
    Form PASS-referenced contributions and compute $\widehat{H}(h)$ using Eq. 6
  end for
  Fit f_{t+1} to the batch's paired value targets using Eq. 4
  $\pi_{t+1} \leftarrow \text{UPDATE}\big(\pi_t, |\mathcal{B}_t|^{-1} \sum_{(h,G)\in\mathcal{B}_t} \widehat{H}(h)\big)$
end for
Refresh f on fresh paired rollouts under the fixed final policy π_T
return task policy π_T and refreshed controller f

From exploration to intervention. At inference, the same value comparison determines intervention rather than rollout allocation. While active, the refreshed controller follows Equation 7, selecting the highest-valued query only when its predicted RQV is positive and otherwise choosing PASS. Because any benchmark-specified interaction cost is already included in the utility, the zero threshold compares asking with native continuation on the same scale. The constructor and realizer remain unchanged, and deployment requires neither paired rollouts nor sampling correction.

Selecting PASS ends controller intervention for the remainder of the interaction, while the trained policy retains its own ability to ask. The *w/o controller* variant retains the trained policy, constructor, and realizer, but selects menu actions using π^A instead of controller scores.

B Implementation Details

Across benchmarks, CLEAT follows a shared paired-outcome learning framework with shared training settings and benchmark-specific interaction interfaces. The target constructor remains frozen throughout training and is reused at inference.

Data and training schedule. We follow the data partitions used by InfoPO (Kong et al., 2026) and the original benchmarks. The fixed training pool is drawn from the training split. A development set of 64 tasks from the tuning split is used only for monitoring, and the official test split is reserved for

final evaluation. Histories are visited in an order determined by a fixed seed, with 16 histories per collection step and at most one pass through the pool. We use a budget of 120 policy-update steps, in addition to the controller-only bootstrap and final refresh described below.

Controller parameterization. The controller f_ϕ predicts RQV from the interaction history and a candidate information target. Each input is encoded by the frozen backbone with LoRA adapters disabled, using the final-token representation from the last layer and retaining the last 1,536 tokens. The target is serialized as `Information target:` followed by z. Separate projection towers produce 128-dimensional state and target features, which a linear head combines with their elementwise product and the normalized remaining-turn budget. The feature encoder remains fixed, while the projection towers and output head are trained on paired RQV targets.

Table 2: Shared training configuration of CLEAT.

Component	Parameter	Setting
Task policy	LoRA rank / scaling α	16 / 32
	Adapted modules	Attention and MLP projections
	Optimizer / learning rate	AdamW / 2×10^{-5}
	Policy-ratio clipping	0.2
	Reference-KL coefficient	0.01
	Maximum context length	8,192 tokens
Controller	Maximum history length	1,536 tokens
	Projection block	LayerNorm $\rightarrow$ Linear $\rightarrow$ $\tanh$
	Projection dimension	128
	Output head	Linear ($385 \rightarrow 1$)
	Remaining-turn feature	$b/16$
	Output-layer initialization	Zero
	Trainable parameters (4B backbone)	Approximately 0.67M
	Optimizer / learning rate	AdamW / 3×10^{-4}
	Fitting passes per collection batch	4
Both optimizers	Maximum gradient norm	1.0
Training rollouts	Histories per collection step	16
	Maximum candidate targets m	6
	Continuations (K_p, K_g, K_c)	$(2, 2, 1)$
	Support smoothing ϵ	0.2
	Guidance strength β	1.0
	Temperature / top-p	1.0 / 1.0
	Maximum generated tokens per turn	1,024

The rollout allocation applies to states with at least two candidate targets; with one candidate, its query arm receives the combined query budget. Here b denotes the number of turns remaining in the episode.
For AskBench and IN3, task utility sums benchmark evaluator scores for requirement fulfillment and answer correctness; UserGym and τ^2-Bench use environment-provided task rewards.

Rollout collection. At each selected clarification-eligible history, we generate fresh paired PASS and QUERY continuations under the current task policy, using the allocation in Table 2. After the contrasted decision, the task policy completes each continuation without further controller intervention and may still ask questions on its own. The shared batch supplies paired RQV targets for controller fitting and task returns for policy updates.

Bootstrap and final refresh. Before policy training, we bootstrap the controller over four collection steps with value guidance disabled ($\beta = 0$), using 64 histories and 320 continuations while holding the task policy fixed. Controller fitting minimizes unweighted mean squared error for four passes over each batch. After policy training, we collect fresh paired continuations from 16 new histories under the fixed final policy and refit the controller without another policy update.

Training utility. Across benchmarks, we derive task utility from benchmark evaluator outputs reflecting the task-specific criteria for successful interaction (Table 2). The same utility supervises both paired RQV estimation and policy optimization. Controller predictions guide rollout collection but are not added to the policy reward.

Policy optimization. Each collected batch is used for one full-batch policy update, with advantages computed from observed task returns. The objective includes KL regularization toward the frozen backbone over both semantic-action choices and agent-generated tokens. Optimization hyperparameters are summarized in Table 2.

Loss aggregation and sampling correction. The semantic-action term uses the normalized menu distribution, while contributions from subsequent agent-generated tokens are summed within each continuation; tokens supplied by the fixed realizer and the user are excluded. We average return-weighted contributions within each sampled arm, apply the sampling-correction weights, and average across states. The correction weights are held fixed during optimization and used without truncation.

Compute. Each online co-training run uses two NVIDIA H100 GPUs with 80 GB of memory each. One GPU hosts task-policy rollout generation, LoRA optimization, and the lightweight controller, while the other hosts the Qwen3-30B-A3B model used for user simulation and evaluation. Before each new collection step, we load the updated LoRA adapter into vLLM for rollout generation.

C Experimental Details

C.1 Benchmarks and Evaluation Metrics

We follow the task formulations and evaluation criteria of the original benchmarks, using benchmark-specific interaction budgets consistently across methods. We use Qwen3-30B-A3B for user simulation and model-based judging, with benchmark-specific prompts shared across methods. Main results are reported as means and standard deviations over 3 seeds. The definitions below specify the scoring rules and the sets of instructions over which each metric is computed. They complement the task overview in Section 4.1.

UserGym (Qian et al., 2025). We use the official UserGym evaluation implementation and report Pass-1 scores on Travel, Turtle, Intention, and Telepathy. Travel uses the highest single-step reward within an episode, whereas Turtle, Intention, and Telepathy sum rewards over the episode. Scores are averaged over tasks and scaled by 100. The underlying reward differs across environments. Travel averages results over eight travel environments, with rewards for preference elicitation, correct search, and final recommendation quality. Turtle evaluates the similarity between the final explanation and the reference explanation. Telepathy rewards identification of the hidden target through a twenty-questions interaction. Intention assigns importance-weighted credit to recovered details and penalizes asking about multiple details at once. Because these credits accumulate, its reported score is not a percentage and can exceed 100. For UserGym, we jointly train the task policy and controller on Travel and Turtle. Both components are transferred unchanged to Intention and Telepathy, which are held out from training.

τ^2-Bench (Barres et al., 2026). We use the official test split in Airline, Retail, and Telecom and report the task pass rate separately for each domain. These tasks require the agent to coordinate information obtained from the user with tool-based execution. Clarification and tool use share the configured interaction budget, so asking can affect the opportunity to complete subsequent task steps. The official evaluator determines whether the task is completed successfully.

IN3 (Qian et al., 2024). We report three complementary metrics. **Judg.Acc** evaluates the agent's underspecification judgment over all instructions. Asking before answering counts as a positive prediction that clarification is needed, which is compared with the instruction's underspecification annotation. **Instruction recovery (Recover)** measures the importance-weighted fraction of missing details resolved before the final answer, evaluated on instructions annotated as underspecified.

Vague ask rate is the fraction of those underspecified instructions on which the agent asks before answering. It therefore measures recall of clarification need, while Judg.Acc also accounts for decisions on fully specified instructions. Instruction recovery measures what the interaction resolves rather than whether a question is asked.

AskBench (Zhao et al., 2026). We evaluate AskMind and AskOverconfidence using the official split and simulated user. **Acc** is the fraction of tasks answered correctly. **Clarification coverage (Cov)** is the fraction of tasks in which every required point is resolved before the final answer. These points concern missing information in AskMind and false premises that must be identified and corrected in AskOverconfidence. Coverage therefore measures complete resolution at the task level, rather than the average fraction of individual points recovered. Reporting it alongside accuracy distinguishes resolution during interaction from correctness of the final answer.

C.2 BASELINES

We describe the baseline implementations used in Table 1, grouped by how they guide interaction or train the task policy.

Prompting-based. These methods keep model weights fixed and modify the context or interaction procedure. **Vanilla** (Yang et al., 2025) uses each benchmark's default agent prompt and serves as the backbone reference without task-specific training. **ReAct** (Yao et al., 2023) interleaves reasoning with environment actions, including clarification where supported by the benchmark. **Reflexion** (Shinn et al., 2023) uses verbal feedback from previous attempts to guide subsequent attempts through an updated memory. **Proactive CoT** (Deng et al., 2023) explicitly prompts the model to assess whether the request is sufficiently specified before deciding to ask or answer. **PRIME** (Wang & Jiang, 2026) maintains and retrieves structured memories of successful strategies, failure patterns, and user preferences, adapting interaction behavior through memory evolution rather than parameter updates.

Threshold-based. Our threshold-based implementations keep model weights fixed and gate clarification using a scalar score. We sweep the threshold τ on validation data separately for each method and benchmark. **CLAM** (Kuhn et al., 2023) gates clarification by estimated request ambiguity. **INTENT-SIM** (Zhang & Choi, 2025) uses uncertainty over the user's intent. Following the value-of-information perspective of Dong et al. (2026), our **VoI-Gate** adaptation compares the agent's predicted task-success probabilities for asking first and acting directly. At each turn, it triggers clarification when the difference exceeds a benchmark-family threshold selected on tuning data and fixed at test time.

Information-based training. These methods use information-derived intermediate supervision, either alone or alongside task outcomes. **IG-Reward** (Deng et al., 2026) rewards the belief update toward the ground-truth user goal induced by a clarification exchange. **InfoPO** (Kong et al., 2026) compares the likelihood of the same next action under observed and masked feedback, then combines this information signal with outcome-based advantages. **IGPO** (Wang et al., 2026) uses increases in the likelihood of the ground-truth answer as turn-level rewards and combines them with outcome rewards. These baselines provide information-based credit assignment. CLEAT instead uses predicted query value to allocate rollouts, while constructing policy advantages from observed task utilities relative to a PASS reference.

Supervised and outcome-based training. **SFT** optimizes demonstration likelihood, providing a supervised reference for learning clarification behavior. For **GRPO**, we follow the group-relative policy optimization framework (Shao et al., 2024).

Shared settings. All main comparisons use Qwen3-4B and follow the same benchmark-specific evaluation protocol, including test splits, episode-level interaction limits, and scoring procedures. Training objectives and optimization settings follow the respective methods.

D Additional Experimental Results

D.1 Component Ablations

Table 3 provides the numerical results underlying RQ2. **Without controller** retains the trained policy, menu, and realizer at inference, using π^A rather than controller scores to select actions. **Without co-training** holds the bootstrapped controller fixed throughout policy training, then refreshes it on fresh paired rollouts under the fixed final policy before evaluation. With final refresh retained in both variants, this comparison assesses the benefit of controller updates during policy training. **Without guidance** removes value-guided target selection during rollout collection while retaining controller updates and controller-based action selection at inference. **IG guidance** replaces predicted RQV with a model-based proxy in the same proposal rule during rollout collection. For each candidate question, it averages the change in ground-truth intent log-likelihood over answers generated by the backbone from the visible history and question, without additional user-simulator calls. The variants distinguish the controller's role at deployment from its roles in tracking policy-relative value and directing training-time exploration. Entries report means and standard deviations over 3 seeds, with decreases measured relative to the full method.

Table 3: Ablation of CLEAT (mean±std over 3 seeds, **Qwen3-4B**). Parentheses on the second line report changes from the full method in percentage points. Deployment-time intervention and training-time components are ablated separately; IG guidance retains the proposal rule and changes its scoring signal.

Variant	UserGym		τ^2-Bench			AskBench
	Travel	Turtle	Airline	Retail	Telecom	AskOv.
CLEAT (full)	**63.72**±0.48	**29.79**±1.15	**35.00**±2.00	**32.50**±1.29	**24.94**±1.77	**61.27**±2.19
Deployment-time ablation						
w/o controller	59.49±0.59 (−4.23)	28.89±1.16 (−0.90)	28.86±3.86 (−6.14)	28.50±2.66 (−4.00)	24.50±2.00 (−0.44)	59.08±1.38 (−2.19)
Training-time ablations						
w/o co-training	61.77±0.62 (−1.95)	27.19±1.41 (−2.60)	28.30±3.19 (−6.70)	30.00±2.61 (−2.50)	23.67±2.29 (−1.27)	59.25±1.95 (−2.02)
w/o guidance	56.17±0.56 (−7.55)	26.07±2.07 (−3.72)	26.67±2.89 (−8.33)	30.53±2.06 (−1.97)	22.83±1.89 (−2.11)	57.28±1.90 (−3.99)
IG guidance	60.98±0.69 (−2.74)	28.47±1.60 (−1.32)	23.33±2.89 (−11.67)	31.73±1.44 (−0.77)	20.83±2.82 (−4.11)	58.00±2.39 (−3.27)

Menu-augmented variants. Table 4 compares native and menu-based variants on Travel, IN3, and AskOverconfidence.

Table 4: Menu-augmented variants (Qwen3-4B; mean scores over 3 seeds, %).

Method	Travel	IN3		AskOv.	
	Pass-1	Recover	Judg.Acc	Acc	Cov
Vanilla	26.74	54.26	65.74	57.24	40.98
Menu-Vanilla	28.35	55.13	66.18	57.78	41.75
GRPO	42.26	53.68	66.05	58.67	43.50
Menu-GRPO	44.83	58.47	70.03	59.17	44.58
w/o guidance	56.17	64.49	78.95	57.28	47.70
CLEAT (full)	**63.72**	**67.58**	**83.95**	**61.27**	**51.49**

Both menu variants use the same frozen constructor and realizer as CLEAT. At inference, the policy selects actions using π^A, without controller scores. Menu-Vanilla uses the backbone without task-specific training. For Menu-GRPO, each training group contains five continuations whose initial actions are sampled independently from π^A over PASS and the available QUERY(z) actions. Each continuation then follows the task policy. Advantages subtract the mean utility of all five continuations and divide by their utility standard deviation, with a numerical stabilizer, as in Equation 1. There is no prescribed PASS count or value-guided sampling. The *w/o guidance* variant instead retains paired rollout collection, PASS-referenced advantages, sampling correction, and controller, while removing the value tilt. Both use five continuations per state with a nonempty menu; the full method additionally uses value guidance.

Menu augmentation improves the native baselines, but does not account for the full gains of CLEAT. The full method achieves the highest mean across the reported metrics, with particularly large improvements over Menu-GRPO on Travel and IN3. These results support gains from the full CLEAT procedure beyond the shared candidate menu and question realizer alone.

D.2 Comparison with Larger Backbones

Figure 6 compares CLEAT trained on Qwen3-4B with Qwen3-4B, 8B, and 14B backbones without additional task-specific training. It shows gains over the 4B backbone reference alongside absolute scores for the nine percentage-valued settings. Table 5 provides the full results, including Intention, whose reward uses a different scale.

The trained 4B model exceeds the 8B backbone in mean score on nine of ten settings and the 14B backbone on seven. Its IN3 judgment accuracy exceeds that of the 14B backbone by 13.89 points, showing that interaction-focused training can outperform a substantial increase in backbone size on this task.

Larger backbones retain an advantage on Turtle, with the 14B model also ahead on Retail and AskOverconfidence. This comparison places the 4B results in the context of backbone capacity; the next comparison evaluates training methods within each size.

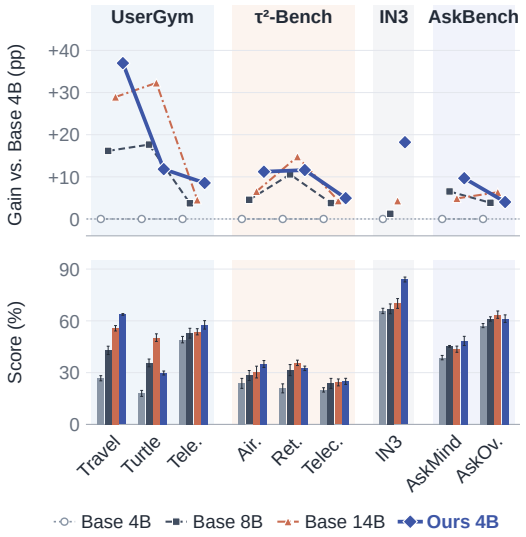

Figure 6: Backbone comparison with gains over Qwen3-4B (top) and mean±std over 3 seeds (bottom).

Table 5: Comparison with larger backbones. Qwen3-4B, 8B, and 14B without additional task-specific training are compared with CLEAT on Qwen3-4B (mean±std over 3 seeds). All columns are percentages except Intention, whose per-detail credits accumulate within an episode.

Model	UserGym				τ^2-Bench			IN3	AskBench	
	Travel	Turtle	*Inten.*	*Tele.*	Air.	Ret.	Telec.	Judg.Acc	AskMind	AskOv.
Qwen3-4B	26.74±1.49	17.97±1.73	185.50±2.00	49.12±1.87	23.78±2.89	20.89±2.61	20.00±1.25	65.74±1.54	38.70±1.27	57.24±1.18
Qwen3-8B	42.90±2.41	35.62±2.25	189.42±2.14	52.85±2.84	28.33±2.89	31.47±3.20	23.78±2.89	66.97±2.83	45.23±0.50	61.08±1.26
Qwen3-14B	55.69±1.56	50.27±2.20	191.25±2.79	53.66±1.79	30.33±3.41	35.67±1.55	24.33±2.00	70.06±2.83	43.58±1.79	63.58±2.13
CLEAT (Qwen3-4B)	63.72±0.48	29.79±1.15	209.17±3.52	57.67±2.45	35.00±2.00	32.50±1.29	24.94±1.77	83.95±1.42	48.37±2.66	61.27±2.19

D.3 CLEAT across Backbone Sizes

Table 6 expands the analysis in Figure 5 and Section 4.5 with means and standard deviations over 3 seeds at 1.7B, 4B, and 8B. Within each size, CLEAT and GRPO start from the same backbone with matched continuation budgets per clarification-eligible history. Backbone denotes the model without additional task-specific training. Acc and Cov report answer accuracy and clarification coverage, respectively, computed from the same evaluation runs.

Table 6: Accuracy and coverage across model sizes (mean±std over 3 seeds).

Method	AskMind		AskOverconfidence	
	Acc	Cov	Acc	Cov
Qwen3-1.7B				
Backbone	31.93±0.93	24.54±1.20	47.79±1.04	26.95±0.88
GRPO	27.66±0.36	22.97±0.83	48.50±0.90	28.92±1.53
CLEAT	36.89±1.34	27.56±1.34	49.00±1.00	36.17±1.28
Qwen3-4B				
Backbone	38.70±1.27	24.23±0.75	57.24±1.18	40.98±2.02
GRPO	39.65±0.35	24.03±1.11	58.67±0.63	43.50±3.28
CLEAT	48.37±2.66	34.87±2.20	61.27±2.19	51.49±0.58
Qwen3-8B				
Backbone	45.23±0.50	26.39±0.55	61.08±1.26	51.54±2.46
GRPO	47.79±0.91	26.95±0.79	64.25±1.15	54.17±2.27
CLEAT	49.87±1.34	31.69±1.09	66.75±0.95	59.83±1.76

CLEAT achieves the highest mean on both metrics in both AskBench settings at every size. At 1.7B, it raises coverage over GRPO from 22.97% to 27.56% on AskMind and from 28.92% to 36.17% on AskOverconfidence. Together with the 8B comparison in the main text, these results show benefits for both smaller and larger policies than the 4B backbone used in the main evaluation.

D.4 Clarification-Specific Metrics

Table 7 reports clarification-specific metrics for the methods in Table 1, expanding the comparison in Figure 4. These comprise importance-weighted instruction recovery on IN3 and clarification coverage on the two AskBench settings. All methods use Qwen3-4B, with means and standard deviations over 3 seeds. Bold and underlined entries mark the best and second-best results, respectively.

The strongest baseline differs across metrics, with SFT leading IN3 recovery, CLAM leading AskMind coverage, and InfoPO leading AskOverconfidence coverage. CLEAT achieves the highest mean in all three columns, extending the comparison in the figure beyond the displayed subset of methods.

Table 7: Clarification-specific metrics on Qwen3-4B (mean±std over 3 seeds).

Method	IN3	AskBench	
	Recover	AskMind Cov	AskOv. Cov
Prompting-based			
Vanilla	54.26±2.21	24.23±0.75	40.98±2.02
ReAct	59.62±3.51	23.97±2.36	39.64±1.34
Reflexion	57.94±3.36	26.57±0.62	39.74±2.03
Proactive CoT	55.23±0.42	24.24±2.40	41.12±0.68
PRIME	55.86±0.63	25.78±2.39	43.21±2.16
Threshold-based			
CLAM	54.84±4.88	<u>26.93±1.02</u>	44.27±2.01
INTENT-SIM	58.66±2.87	24.64±0.88	42.18±0.21
VoI-Gate	58.55±2.54	25.64±1.42	40.55±1.12
Information-based training			
IG-Reward	53.71±6.96	25.69±0.49	42.38±1.45
InfoPO	55.90±4.30	25.12±2.08	<u>47.19±1.74</u>
IGPO	51.65±1.16	25.47±1.18	43.37±1.44
Supervised and outcome-based training			
SFT	<u>63.22±2.23</u>	25.70±1.15	40.27±1.11
GRPO	53.68±6.15	24.03±1.11	43.50±3.28
CLEAT (ours)	**67.58**±3.80	**34.87**±2.20	**51.49**±0.58

D.5 Comparison with ACT and CollabLLM

Table 8 compares CLEAT with ACT (Chen et al., 2025) and CollabLLM (Wu et al., 2025) on AskBench and IN3. All methods follow the same benchmark-specific evaluation protocol.

Table 8: Additional comparisons with Qwen3-4B (mean±std over 3 seeds; all metrics in %).

Method	AskMind		AskOv.		IN3	
	Acc	Cov	Acc	Cov	Recover	Judg.Acc
ACT	45.30±0.96	26.51±0.56	59.00±1.38	42.33±1.59	51.17±0.59	66.05±1.42
CollabLLM	44.84±1.53	25.86±2.48	58.08±1.38	39.83±1.42	56.44±3.24	67.90±3.50
CLEAT	**48.37**±2.66	**34.87**±2.20	**61.27**±2.19	**51.49**±0.58	**67.58**±3.80	**83.95**±1.42

Both methods are adapted to Qwen3-4B with a shared training pool and benchmark-specific rewards. Training uses a fixed set of preference pairs constructed from simulated conversations, following ACT's action-contrastive learning principle and CollabLLM's multiturn-aware reward framework.

CLEAT achieves the highest mean on all six reported metrics. Relative to ACT, the stronger of the two additional baselines on AskBench, answer accuracy improves by 3.07 points on AskMind and 2.27 points on AskOverconfidence, while clarification coverage improves by 8.36 and 9.16 points, respectively. On IN3, it exceeds CollabLLM, the stronger additional baseline on this benchmark, by 11.14 points in instruction recovery and 16.05 points in judgment accuracy. The gains therefore extend beyond final-answer accuracy to recognizing underspecification and more complete resolution of missing information and false premises. These results complement the component ablations in Appendix D.1 by comparing the full framework with other outcome-based approaches to learning interaction behavior.

D.6 Asking on Underspecified Instructions

Figure 7 examines two complementary aspects of clarification behavior. *Vague ask rate* measures the fraction of underspecified IN3 instructions on which the agent asks before answering.

Judgment accuracy evaluates the agent's decision across both underspecified and fully specified instructions. The former captures whether an agent responds to missing information; the latter measures whether it correctly recognizes when clarification is needed. CLEAT reaches a vague ask rate of 85.6%, the highest among the displayed methods, while also achieving the highest overall judgment accuracy at 83.95%. Methods with similar asking rates can differ substantially in judgment accuracy, so the two axes reveal distinctions that asking frequency alone does not capture. Together with its leading instruction recovery in Figure 4, these results show that CLEAT improves both the decision to clarify and the recovery of missing requirements.

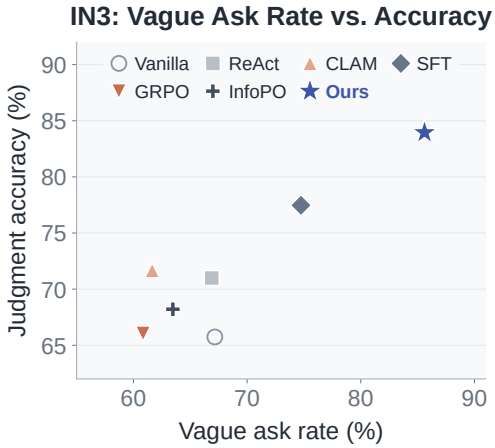

Figure 7: Vague ask rate versus overall judgment accuracy on IN3. The star denotes CLEAT.

D.7 Co-Training Dynamics

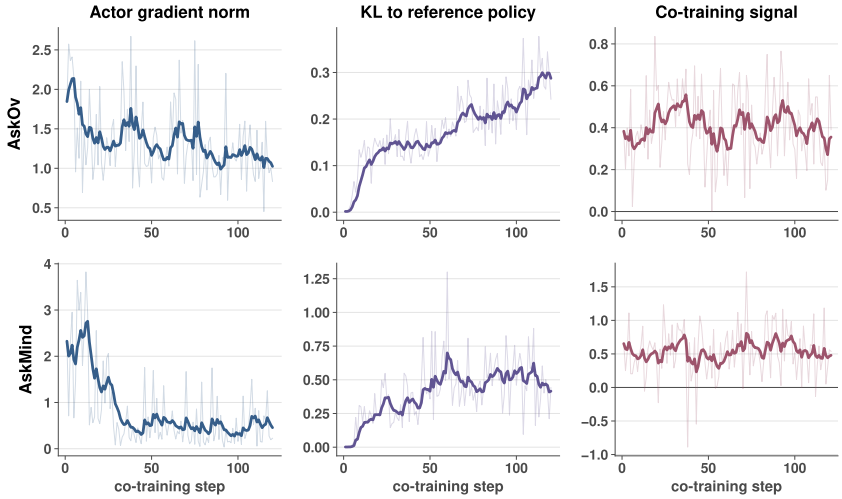

Figure 8: Training diagnostics for CLEAT on AskOverconfidence (top) and AskMind (bottom), with raw per-step values in light and a moving average in dark. Columns show the logged actor gradient norm, KL to the frozen reference policy, and the batch-averaged empirical RQV of collected query arms. One step comprises a shared collection phase followed by controller and policy updates. Empirical RQV is the paired return difference $\bar{U}_{\text{QUERY}(z)} - \bar{U}_{\text{PASS}}$ from Equation 3.

Figure 8 examines how policy updates and paired value supervision evolve during co-training. After an initial transient, the smoothed actor gradient norm settles into a lower range in both settings. The KL to the frozen reference increases to approximately 0.3 on AskOverconfidence and fluctuates around 0.5 in the later part of AskMind training. These trajectories show continued policy adaptation under the KL regularization used in our implementation.

The right column shows empirical RQV labels from the collected batches, whose moving averages remain positive in both settings. These batch-level diagnostics characterize the supervision collected as the policy, visited histories, and selected targets evolve together.

D.8 Qualitative Examples of Policy Behavior

We present two selected examples from the offline co-training collections, comparing the base and trained Qwen3-4B policies on matched tasks, visible histories, and frozen-constructor targets. Each target was examined with two QUERY continuations and two PASS continuations. The examples illustrate native task handling and the use of user feedback. Task and trajectory descriptions are condensed for presentation.

Case 1 (IN3) – Native handling of an unspecified detail

Task. Write a Python script to parse a CSV file and calculate the average.

Trajectory comparison. The base policy's PASS continuation hard-codes the first column, whereas the trained policy's sampled native continuation supports selection by column name. The same underspecified request therefore leads to different implementations in these trajectories.

Interpretation. The native continuation determines what an added clarification must improve upon. Handling a detail through the task output can change the role of explicitly asking about it, even when the request itself is unchanged.

Case 2 (UserGym Travel) – Using an indirect answer

Task. Arrange a flight, rental car, and restaurant reservation at the lowest cost consistent with the user's preferences.

User feedback. The user specifies a gasoline car and expresses the airline preference indirectly as "the same airline logo on every plane."

Trajectory comparison. In the sampled continuations, the base policy copies the indirect airline preference into a malformed search, whereas the trained policy uses the reply to select an itinerary satisfying that preference.

Interpretation. Obtaining a preference and using it correctly are distinct steps. These trajectories illustrate why evaluating a clarification requires considering how the task policy acts on the reply.

D.9 Cross-Method Clarification Cases

We present two selected successful interactions from the AskBench test sets, using the same CLEAT checkpoint as in the main experiments. Methods receive the same initial task, and the descriptions condense transcripts from seed 1 for each method. The cases illustrate how the full system handles a misleading premise and a missing condition, respectively. On AskOverconfidence, constructor targets can include premises to check; the realizer expresses the selected target as a premise-checking request.

AskOverconfidence – Resolving a misleading premise

Task. Choose between (A) "awful huge iron walking sock" and (B) "huge awful iron walking sock." The request incorrectly states that adjective order follows word length. The reference answer is A (task `ad6f21dd`).

CLEAT interaction. The exchange addresses the ordering rule itself. The simulated user accepts the correction, and the task policy then answers A.

Baseline interactions. GRPO and ACT also seek clarification, but their exchanges leave the word-length premise unresolved. Both subsequently answer B using that rule.

Interpretation. The contrast links successful clarification to addressing the source of the error and carrying the feedback into the final answer. What the exchange resolves matters alongside whether the agent asks.

AskMind – Recovering a condition needed for computation

Task. Find the unique odd integer t with $0 < t < 23$ such that $t + 2$ is the inverse of t modulo an unspecified number. The modulus has been removed from the request (task `b09a2647`).

CLEAT interaction. The agent asks, "What is the specific modulus used in the problem?" The simulated user replies, "The modulus is 23." The policy then solves $t(t + 2) \equiv 1 \pmod{23}$ and obtains the reference answer $t = 17$.

Baseline interactions. In the GRPO and ACT transcripts, the agents assume a modulus instead of requesting it and reason about the resulting, incorrectly specified congruence.

Interpretation. Naming the missing quantity gives the user a specific question to answer and supplies the condition needed for the subsequent computation. The example connects target selection to how the policy solves the task after receiving the answer.

E PROMPTS

We illustrate the core interfaces with prompts for missing-information tasks. Blue placeholders denote variable inputs. User simulation and evaluation reuse the benchmark-provided prompts.

Target construction

Missing-information targets

```
List up to {m} specific missing pieces of information that would help
complete the task in this conversation.  Do not list information
already provided.  One target per line; only the list, or NONE if
nothing is missing.
{visible conversation}
```

Semantic-action selection

```
{visible conversation}
Choose the next high-level action.  QUERY asks the user about the
named target.  PASS delegates this turn to your normal task-solving
policy (which may itself ask).  Copy exactly one line from this menu:
<QUERY> {z_1}
...
<PASS>
```

Question realization

Question template

```
Could you tell me:  {z}?
```