ClarifySAE: Steering Large Language Models Toward Clarification through Sparse Autoencoders

Anonymous ACL submission

Abstract

Instruction-tuned LLMs often resolve ambiguous instructions by guessing missing details rather than asking clarifying questions, which can cause task failure or safety risks for embodied agents. We propose ClarifySAE, an inference-time method that steers clarification-seeking by intervening on Sparse Autoencoder (SAE) features. ClarifySAE ranks features with ClarifyScore, a clarification-specific adaptation of ReasonScore, and filters them with OutputScore to retain features that affect the output distribution. We evaluate on AmbiK and ClarQ-LLM datasets using Gemma-2B, Gemma-9B, Llama-1B, and Llama-8B. On AmbiK, for Gemma-2-9B-IT clarification rate improves from 0.61 to 0.95 and task success from 0.06 to 0.21. On ClarQ-LLM, the best single-feature configurations improve success from 0.50 to 0.80 for Gemma-9B and from 0.40 to 0.90 for Llama-1B, with similar gains in step recall. Overall, ClarifySAE provides a lightweight, training-free mechanism for modulating clarification behavior through SAE features. The source code will be made publicly available.

1 Introduction

Large Language Models (LLMs) are increasingly used as high-level planners in interactive robotic systems, where natural-language instructions must be translated into concrete actions (Driess et al., 2023; Brohan et al., 2024). In human–robot interaction (HRI) users often rely on shared context rather than explicit specification (Ivanova et al., 2025; Park et al., 2023; Liang et al., 2024; Ren et al., 2023). As a result, an instruction such as "serve mashed potato with a side of chips" can be ambiguous when the environment contains multiple plausible "chips" options (e.g., jalapeño, potato, or rice chips), and the user does not specify which one they mean. If an embodied agent acts on the wrong interpretation, the result may range from task failure to unsafe or irreversible physical actions (Zhang et al., 2024a; Hundt et al., 2025; Ahn et al., 2022). Yet instruction-tuned LLMs often produce fluent, confident responses under ambiguity, implicitly guessing missing details rather than asking clarifying questions (Lin et al., 2022; Ji et al., 2023); in embodied deployments, this can limit the reliability and trustworthiness of LLM-driven agents (Amodei et al., 2016).

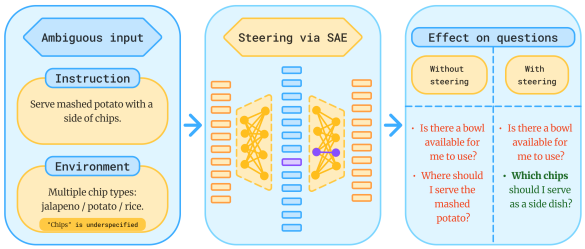

Figure 1: Overview of ClarifySAE. The method consists of constructing a clarification signal, identifying clarification-associated SAE features, and applying feature-level steering during inference.

Despite their strong instruction-following performance (Ouyang et al., 2022; Wei et al., 2021), current LLMs rarely ask clarifying questions when ambiguity is present. While prompting techniques can encourage clarification in isolated cases (Wei et al., 2022; Aliannejadi et al., 2019), their effects are brittle and inconsistent across tasks and domains. As a result, clarification-seeking behavior remains difficult to elicit reliably, especially in settings where the agent must decide when and how to ask a follow-up question.

Existing approaches to inducing clarification behavior largely operate at the level of inputs or outputs (Kuhn et al., 2022; Zhang et al., 2024b). Prior work has explored prompting strategies, supervised fine-tuning, and reinforcement learning to encourage question asking (Wei et al., 2021; Ouyang et al., 2022). However, these methods offer limited control over when clarification is triggered and provide little insight into the internal mechanisms responsi-

ble for the behavior. Consistent with this limitation, recent benchmarks such as AmbiK (Ivanova et al., 2025) and ClarQ-LLM (Gan et al., 2024) demonstrate that even strong LLMs struggle to resolve ambiguity by asking appropriate questions.

In this work, we take a mechanistic approach to clarification seeking by intervening directly on internal model representations. We introduce **ClarifySAE** (Figure 1), a method that steers LLMs toward clarification behavior by selectively activating features in a Sparse Autoencoder (SAE) (Bricken et al., 2023; Cunningham et al., 2023) trained on hidden states of an instruction-following model (Bricken et al., 2023). Rather than modifying prompts or retraining the LLM, ClarifySAE operates at inference time by identifying and amplifying internal features associated with clarification-relevant contexts. ClarifySAE *does not aim to solve ambiguity detection*; we evaluate it as an inference-time steering intervention in settings where ambiguity is present.

To identify clarification-associated features, we construct a clarification vocabulary from AmbiK and ClarQ-LLM datasets. We score all SAE features using ClarifyScore, an instantiation of ReasonScore (Galichin et al., 2025) adapted to clarification-seeking by replacing its original target signal with a clarification-specific vocabulary. This highlights that ReasonScore is a general, vocabulary-driven metric that can be repurposed for behaviors beyond its original application. We then filter features using OutputScore, which quantifies their causal influence on the model's output distribution (Arad et al., 2025). The resulting features are activated during inference on tasks where clarification is expected.

We evaluate ClarifySAE on AmbiK (Ivanova et al., 2025) and ClarQ-LLM (Gan et al., 2024) using four instruction-tuned models: Gemma-2B, Gemma-9B, Llama-1B, and Llama-8B. On AmbiK, for Gemma-2-9B-IT clarification rate increases from 0.61 to 0.95 and task success from 0.06 to 0.21. On ClarQ-LLM, the best single-feature configurations improve success from 0.50 to 0.80 for Gemma-9B and from 0.40 to 0.90 for Llama-1B, with corresponding step-recall gains from 0.747 to 0.930 and 0.450 to 0.900. Our key contributions are:

- We introduce **ClarifySAE**, an inference-time method for inducing clarification-seeking behavior via feature-level steering with SAEs.
- We evaluate ClarifySAE on AmbiK and ClarQ-LLM across four instruction-tuned models from two model families, observing increased clarification behavior relative to the unsteered baselines.
- We adapt ReasonScore to clarification-seeking by instantiating it with a task-specific vocabulary, demonstrating that the metric can be repurposed to identify steerable SAE features for new behaviors.

2 Related Work

2.1 SAEs for Interpretability and Steering

SAEs have recently been used to uncover interpretable internal features in LLMs and to control their behavior (Shu et al., 2025; McGrath et al., 2024). Closely related, (He et al., 2025a) propose SAE-SSV, which performs supervised steering directly in sparse representation spaces, offering an alternative methodology for selecting and applying steering directions (He et al., 2025b). (Arad et al., 2025) introduce OutputScore, a metric that quantifies how much intervening on an individual SAE feature affects the model's output. They show that selecting features with higher OutputScore yields more reliable and semantically meaningful steering. Relatedly, (Wu et al., 2025a) introduce AutoSteer, an inference-time intervention framework for safe multimodal LLMs that automates the choice of steering signals. SAEs have also been applied to steer Large Vision Language Models (LVLMs) for hallucination mitigation (Hua et al., 2025).

Recent SAE-steering work differs in how steering features or directions are selected. (Bayat et al., 2025) propose sparse activation steering, which uses contrastive prompt pairs to isolate behavior-specific sparse features. (Cho et al., 2025) introduce CorrSteer, which selects SAE features by correlating feature activations with downstream correctness and safety outcomes. (Wu et al., 2025b) propose mutual-information-based SAE explanations to reduce frequency-biased feature interpretations and use these explanations for runtime steering. SCAR instead trains sparse conditioned autoencoders for concept detection and bidirectional steering (Härle et al., 2024). ClarifySAE is distinct to these approaches: we use a clarification-specific lexical signal to retrieve candidate features and then use OutputScore to retain features with measurable causal influence on generation.

Complementary to this line of work, (Galichin et al., 2025) propose ReasonScore as a method for linking SAE features to a target lexical signal. While they apply it to "reasoning moments," the metric is vocabulary-driven: it measures how strongly feature activations are associated with occurrences of a user-specified token (or n-gram) set. By changing this vocabulary, ReasonScore can be used to study other behaviors.

We adopt the same ReasonScore metric but change both its target signal and its role in the pipeline: rather than analyzing reasoning moments, we score features against a clarification vocabulary and use the result to retrieve candidate features for steering. We then rank and filter these candidates using OutputScore, selecting features that reliably affect generation. To avoid confusion with the original "reasoning moments" setting, we refer to the clarification vocabulary-driven instantiation of ReasonScore as ClarifyScore throughout the paper.

A common alternative to steering internal representations is to induce question asking at the level of inputs or outputs, e.g., by prompting the model to reflect or reason step-by-step, or by post-training on clarification-oriented data. Techniques such as chain-of-thought prompting (Wei et al., 2022) can improve reasoning performance, but they do not consistently induce clarification questions under underspecification. Moreover, AR-Bench (Zhou et al., 2025) shows that advanced prompting and post-training methods still show limited gains on interactive information-seeking tasks.

In contrast, we explore a mechanistic route: we intervene directly on internal model activations to encourage clarification-seeking when ambiguity is present or externally suspected. This perspective is complementary to prior datasets and prompting methods, and tests whether clarification behavior can be shaped through interpretable internal features identified by SAEs (Zhang et al., 2025).

2.2 Clarification Under Ambiguity

Clarification-seeking (Ren et al., 2023; Liang et al., 2024) has long been studied in dialogue and instruction-following settings, where systems must decide when to ask a question rather than commit to an uncertain interpretation. Recent benchmarks study this behavior explicitly in LLM settings. AmbiK (Ivanova et al., 2025) targets ambiguous instructions in an embodied-assistant setting, while ClarQ-LLM (Gan et al., 2024) evaluates models' ability to ask useful follow-up questions in conversational instruction following.

Beyond dialogue-style datasets, several benchmarks formalize information gathering as part of solving underspecified tasks. QuestBench (Li et al., 2025) frames question asking as selecting minimal information to make a constraint satisfaction problem solvable, and reports that performance remains low even for models that perform well when tasks are fully specified. Similarly, AR-Bench (Zhou et al., 2025) evaluates "active reasoning" in multi-round interactive settings and finds that current models struggle to reliably acquire and exploit the missing information needed for correct solutions.

Together, these results suggest that strong passive reasoning does not automatically translate into effective clarification behavior, motivating methods that provide more targeted and controllable mechanisms for eliciting clarification.

3 Background

Let M denote the base instruction-tuned language model. We use SAEs to obtain a linear decomposition of hidden states into interpretable features. Given $h \in \mathbb{R}^d$, the encoder yields non-negative sparse activations $z = \mathrm{ReLU}(W_{\mathrm{enc}}h + b_{\mathrm{enc}}) \in \mathbb{R}^m_{\geq 0}$ and the decoder reconstructs $\hat{h} = W_{\mathrm{dec}}z + b_{\mathrm{dec}} \in \mathbb{R}^d$. SAEs are trained with a reconstruction loss plus an ℓ_1 sparsity penalty, $\mathcal{L}_{\mathrm{SAE}} = \mathbb{E}_{h\sim\mathcal{H}}\left[\|h - \hat{h}\|_2^2 + \lambda\|z\|_1\right]$, where $\mathcal{H}$ is the distribution of hidden states extracted from M and λ controls sparsity. Prior work finds that individual SAE features often align with meaningful directions and can be intervened on to steer behavior (Bricken et al., 2023; Huben et al., 2023).

We steer the model by applying additive interventions to selected coordinates of the SAE latent activations z (Turner et al., 2023; Arad et al., 2025). For a set of features $S \subseteq \{1, \ldots, m\}$ and strength α, we define z' coordinate-wise by $z'_j = z_j + \alpha$ if $j \in S$, and $z'_j = z_j$ otherwise, and reconstruct the modified hidden state as $\hat{h}' = W_{\mathrm{dec}}z' + b_{\mathrm{dec}}$. The model then continues generation using $\hat{h}'$ in place of h at the chosen layer. We treat α as a hyperparameter; larger values can increase steering but also degrade generation quality, so we sweep α in experiments. To select S, we use two established metrics. Let V denote a target vocabulary. ClarifyScore measures how strongly a feature's activation is associated with occurrences of tokens from V. OutputScore (Arad et al., 2025) measures how much intervening on a feature changes the model's next-

token distribution. We use ClarifyScore to retrieve candidate clarification features and OutputScore to prioritize those with a measurable causal effect on generation.

Following AmbiK (Ivanova et al., 2025), we call an instruction ambiguous when, given the environment, a valid plan contains a decision point with multiple plausible choices whose wrong resolution may cause an undesirable outcome. The model must then choose whether to answer directly or ask a clarification question.

4 Method

Given a user instruction, we expect the model to ask for missing information when the task is underspecified rather than guess. ClarifySAE encourages this behavior by steering decoding in an SAE feature space: at each step t, the hidden state h_t is mapped to sparse activations z_t, selected clarification-associated and output-influential features are shifted by fixed steering offsets, and the modified activations are decoded back to the model's hidden space. This biases generation toward clarification questions on underspecified inputs. The procedure is illustrated in Figure 2 and specified in Algorithm 1.

4.1 Clarification Vocabulary

We construct a compact lexical clarification signal to retrieve SAE features associated with question-asking under underspecification. Following vocabulary-driven feature discovery (Galichin et al., 2025), we build two corpora from AmbiK and ClarQ-LLM. The *clarifying* corpus contains AmbiK annotated clarifying questions and ClarQ-LLM seeker turns with simple interrogative cues, such as "?" or wh-/modal openers. The *non-clarifying* corpus contains unambiguous instructions, final answers, and provider turns. We then perform two-corpus keyword extraction over contiguous 3–5-grams using a log-likelihood statistic (Rayson and Garside, 2000) and remove low-frequency, stopword-dominated, or dataset-specific phrases, consistent with the observation that clarification requests exhibit recurring lexical forms (Purver et al., 2003). After collapsing overlapping candidates, we select 20 broadly applicable phrases as the target vocabulary for ClarifyScore; the final lists are provided in Appendix B.

4.2 Identifying SAE Features

To steer clarification behavior, we select SAE features using ClarifyScore and OutputScore (formal definitions in Appendix F). Unlike outcome-correlation approaches such as CorrSteer (Cho et al., 2025) ClarifyScore uses a lightweight task-specific vocabulary as the retrieval signal. It ranks features by how selectively they activate on positions matched by a clarification vocabulary in a clarifying corpus relative to a baseline corpus (optionally with an entropy-based concentration term). Because this is correlational, OutputScore filters candidates with an intervention test: we increase feature j during decoding and keep features for which the probability of a representative token increases (e.g., via a logit-lens projection), i.e., $\mathrm{OutputScore} > 0$. We then take the top-K ClarifyScore features that pass OutputScore and a lightweight coherence check on top decoded tokens, forming the final set F.

4.3 Single- and Multi-Feature Steering

We apply inference-time steering at a fixed residual-stream layer using SAE decoder directions, which is equivalent to adding offsets to selected SAE latent coordinates and decoding back to the residual stream. In the single-feature setting, adding strength α to latent feature j changes the hidden state by αf_j, where f_j is the corresponding decoder direction. In the multi-feature setting, we jointly steer a feature set S by combining decoder directions into a single steering vector, $v_S = \sum_{j \in S} w_j m_j f_j$, where m_j denotes a feature-specific scale estimated from activations and w_j an optional weight. The model hidden state is then updated according to $h'_t = h_t + \alpha v_S$, where α controls the overall steering strength. To construct multi-feature sets, we cluster clarification-associated candidate SAE features by similarity of their decoder directions and steer one cluster at a time. This extends single-feature steering to interacting feature subsets while preserving interpretability.

5 Experimental Setup

In our experiments we aim to answer the following research questions (RQ):

- **RQ1: Clarification Behavior.** Does feature steering improve ambiguity resolution and task success?
- **RQ2: Vocabulary Choice.** How sensitive is offline feature discovery to the vocabulary

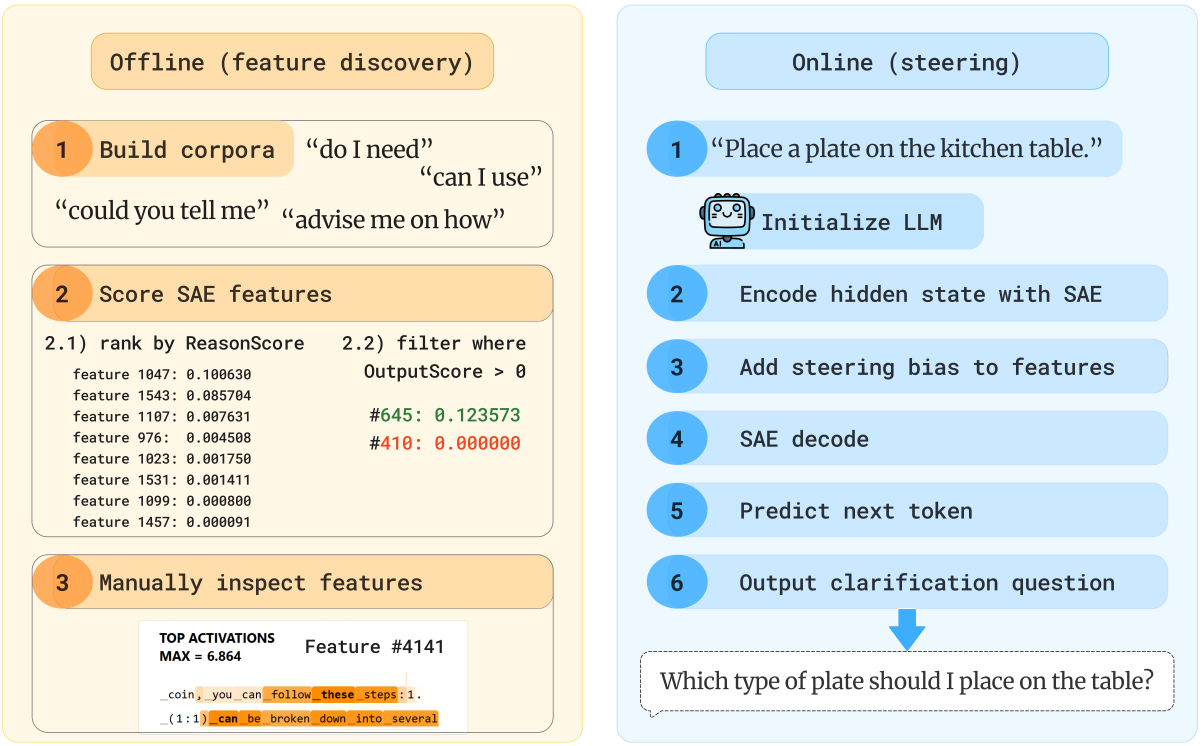

Figure 2: **ClarifySAE method overview.** Offline, we build a lexical clarification signal, score SAE features with ClarifyScore, and retain features with positive OutputScore. Online, we increase the selected SAE feature activations during decoding to steer the model toward a clarifying question.

signal—i.e., using question tokens (Q) versus clarification-context tokens (C)?

- **RQ3: Multi-feature steering.** Can jointly steering clustered clarification-associated SAE features outperform the best individual feature?
- **RQ4: Strength sensitivity.** How does steering strength α affect model behavior and performance?
- **RQ5: Random Features.** Do clarification-associated features outperform random SAE features?

5.1 Benchmarks

To cover both immediate and interactive clarification, we use **AmbiK** (Ivanova et al., 2025) for single-turn ambiguity and **ClarQ-LLM** (Gan et al., 2024) for multi-turn clarification dialogues. AmbiK is a fully textual kitchen-robot dataset with 1000 paired ambiguous/unambiguous tasks spanning three ambiguity types: preferences, commonsense knowledge, and safety. Each example includes an environment description, an ambiguous instruction with unambiguous rewrites, a reference clarifying question and answer, and plans marking where ambiguity begins. ClarQ-LLM is a bilingual (English–Chinese) benchmark for clarification in task-oriented dialogue. It comprises 31 task types with 10 task instantiations each, organized into 26 test types and 5 development types. Each task includes a background and seeker–provider interaction designed to contain multiple underspecified details; models act as the seeker in an interactive dialogue with a provider agent that replicates the original provider behavior.

5.2 Feature Discovery Conditions

To identify candidate SAE features offline, we score features on subsets of contexts defined by lightweight lexical signals. We evaluate two vocabularies: (i) a dataset-derived `Clarification` vocabulary ($|C| = 20$) and (ii) a minimal `Question-words` vocabulary ($|Q| = 11$). As a control (RQ5), we evaluate randomly sampled SAE features from the same layer. Appendix B lists the vocabularies.

5.3 Steering and Evaluation Protocol

We evaluate four instruction-tuned models from two families: Gemma-2B, Gemma-9B, Llama-1B, and Llama-8B.[1] For each model, we use a compatible public residual-stream SAE checkpoint and keep the SAE layer and checkpoint fixed across experiments. SAE checkpoint identifiers and im-

plementation details are provided in Appendix A.

We steer a single SAE feature by adding the corresponding SAE decoder direction to the residual stream with strength α (Algorithm 1), analogous to vector-based activation steering methods that add directions in hidden-state space (Konen et al., 2024). We sweep dataset-specific steering strengths (AmbiK: $\alpha \in \{1, 2, 3, 4, 5, 10, 15, 30\}$; ClarQ-LLM: $\alpha \in \{-10, -5, 0, 3, 5, 10\}$) and report best-observed configurations by AmbiK proxy task success and ClarQ-LLM success rate, with step recall as a secondary metric. Appendices E and D provide prompts and metric definitions.

6 Results and Discussion

We now present the main findings on two datasets: AmbiK (single-turn) and ClarQ-LLM (multi-turn).

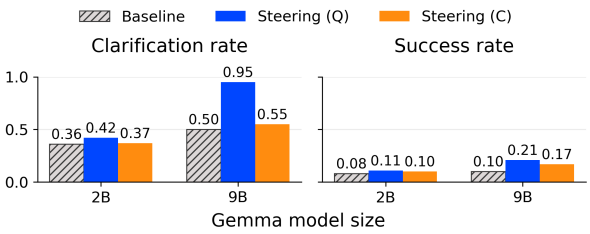

Figure 3: Evaluation on AmbiK (100 examples) for Gemma models.

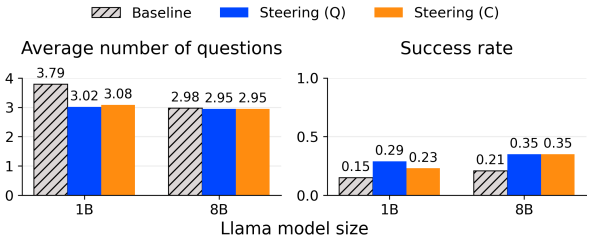

Figure 4: Evaluation on AmbiK (100 examples) for Llama models.

6.1 RQ1: Clarification Behavior

On AmbiK, ClarifySAE affects clarification behavior and task success differently across model families. For Gemma models, the clearest improvement appears for Gemma-9B with Q-vocabulary derived features: clarification rate increases substantially, and task success also improves. Gemma-2B shows smaller gains, suggesting weaker controllability under the tested feature and strength settings.

The Llama results show a different pattern. For Llama-1B, steering improves task success, with the strongest reported condition increasing success from 0.15 to 0.29 while reducing the average number of generated questions from 3.79 to 3.08. This suggests that the improvement is not simply caused by asking more questions. For Llama-8B, however, steering does not improve over the baseline: Q-derived steering reduces success from 0.35 to 0.23, while C-derived steering leaves success unchanged at 0.35. Therefore, feature steering can improve ambiguity-resolution metrics in some configurations, but the effect is not uniform across model families or model scales.

On ClarQ-LLM, selected configurations also improve performance, with gains varying across models and features. The best observed single-feature configurations improve both success rate and step recall over the comparable unsteered baseline for all four evaluated models (Table 1). For example, Gemma-9B with a C-derived feature improves from 0.50 to 0.80 success rate and from 0.747 to 0.930 step recall, while Llama-1B with a C-derived feature improves from 0.40 to 0.90 success rate and from 0.450 to 0.900 step recall. To check whether metric improvements correspond to better clarification behavior, we inspect baseline and steered generations for the same ClarQ-LLM task instances. The qualitative examples in Appendix I, Table 10, show a mixed pattern. In some cases, steering replaces generic questions with more task-specific clarification requests. However, in other cases, higher success rate or step recall coexists with repeated questions, unnatural wording, or role-play artifacts. Thus, step recall measures coverage of the reference missing information, not the fluency or usefulness of the questions.

To further separate question frequency from question quality, Appendix D reports slot-alignment rate (SAR) for single-feature AmbiK steering. SAR measures whether a generated clarification question targets the gold ambiguity slot, rather than merely being interrogative. The results show that ClarifySAE can improve the targeting of clarification questions in some configurations, but increased clarification frequency does not by itself guarantee better slot alignment.

6.2 RQ2: Vocabulary Choice

We compare two lexical signals for offline feature discovery: a dataset-derived clarification vocabulary (C) and a minimal list of question words (Q). The two vocabularies lead to substantially different selected feature sets in the ClarQ-LLM

[1]Model checkpoints: `google/gemma-2b-it`, `google/gemma-2-9b-it`, `unsloth/Llama-3.2-1B-Instruct`, `unsloth/Llama-3.1-8B-Instruct`.

Table 1: ClarQ-LLM evaluation: baseline and best observed single-feature steering configurations.

Model	Vocab	Feature	α	Baseline success rate	Steered success rate	Baseline step recall	Steered step recall
Gemma-2B	Q	538	-5.0	0.200	**0.400**	0.225	**0.582**
Gemma-9B	C	344	3.0	0.500	**0.800**	0.747	**0.930**
Llama-1B	C	6230	-5.0	0.400	**0.900**	0.450	**0.900**
Llama-8B	Q	62124	10.0	0.600	**0.900**	0.892	**0.975**

Table 2: Overlap between Q- and C-derived ClarQ-LLM feature sets (F_Q and F_C) used in the single-feature sweeps. Low overlap indicates that offline feature discovery is sensitive to the vocabulary signal.

Model	$\lvert F_Q \rvert$	$\lvert F_C \rvert$	$\lvert F_Q \cap F_C \rvert$	Jaccard
Gemma-2B	7	4	2	0.222
Gemma-9B	6	6	0	0.000
Llama-1B	13	6	3	0.188
Llama-8B	16	11	4	0.174

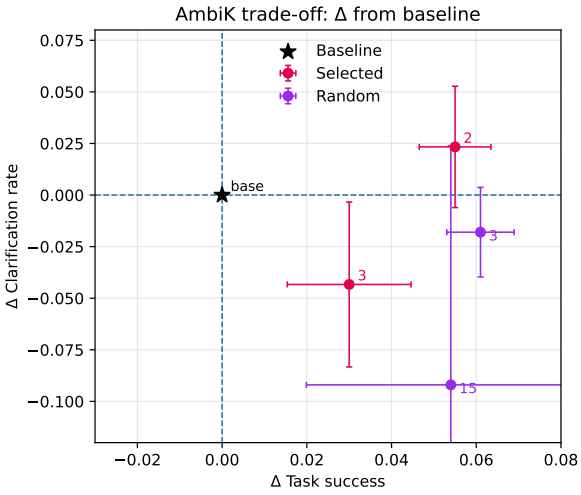

Figure 5: AmbiK trade-off between clarification rate and task success for selected vs. random Gemma-9B (C) features, shown as changes from the unsteered baseline across α.

Table 3: Best observed single- and multi-feature steering results. AmbiK (A) reports task success; ClarQ-LLM (C) reports success / step recall, with multi-feature settings selected by success. We use Gemma-2B and Gemma-9B models (M) in these experiments.

Data	M	Baseline	Best single	Best multi
A	2B	0.080	**0.110**	0.100
A	9B	0.060	0.160	**0.210**
C	2B	0.200 / 0.225	**0.4 / 0.582**	0.3 / 0.452
C	9B	0.500 / 0.747	**0.8 / 0.930**	0.6 / 0.838

sweeps (Table 2). The Jaccard overlap is low for all four models, which suggests that feature discovery is sensitive to the lexical signal used to identify clarification-associated SAE features.

However, lower feature-set overlap does not translate into a consistent advantage for either vocabulary. On AmbiK, Q-derived steering gives the strongest clarification-rate improvement for Gemma-9B, while C-derived steering gives the strongest task-success improvement for Llama-1B (Figures 3 and 4). On ClarQ-LLM, the best observed configurations are split across vocabularies: Q gives the best observed configuration for Gemma-2B and Llama-8B, whereas C gives the best observed configuration for Gemma-9B and Llama-1B (Table 1).

Therefore, results show that vocabulary choice affects both the selected SAE features and downstream steering behavior, but the current results do not support treating either Q or C as uniformly superior. Instead, vocabulary choice should be treated as a feature-selection hyperparameter whose effect depends on the model, dataset, and selected feature.

6.3 RQ3: Multi-feature steering

Table 3 compares the best observed single-feature and multi-feature steering configurations. On AmbiK, multi-feature steering is beneficial for Gemma-9B: it improves task success from 0.060 to 0.210, outperforming the best single-feature configuration at 0.160. For Gemma-2B, however, cluster steering does not improve over the best single feature. On ClarQ-LLM, multi-feature steering improves over the unsteered baseline for both Gemma models, but does not outperform the strongest single-feature configurations.

These results suggest that clarification behavior can involve interacting feature groups, but joint steering is not consistently better than single-feature steering. Multi-feature interventions are especially strength-sensitive: low strengths are often beneficial, while larger strengths can sharply degrade success and step recall. This supports

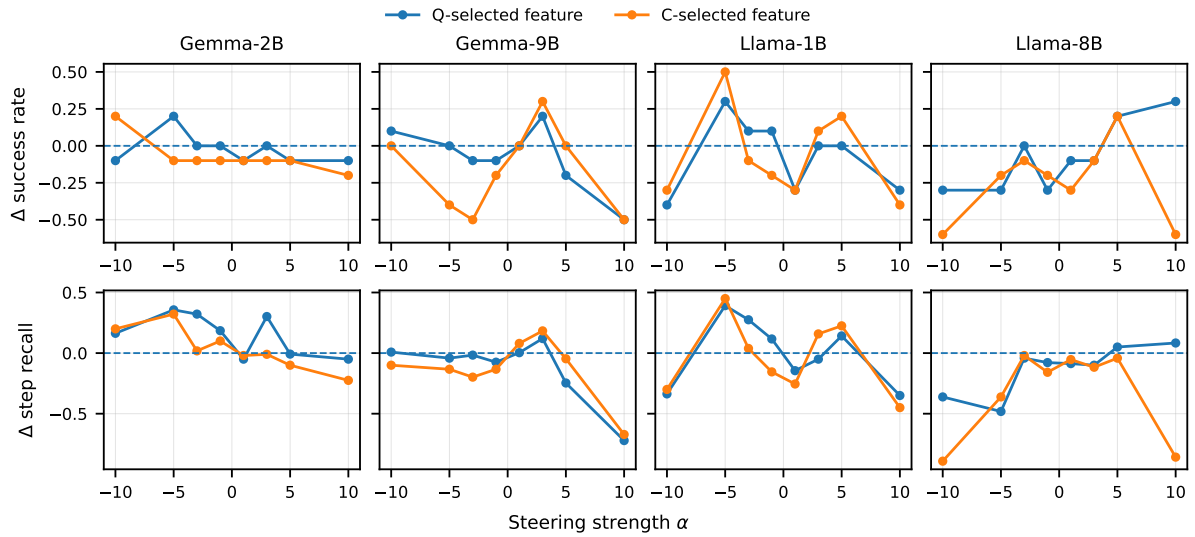

Figure 6: Strength sensitivity on ClarQ-LLM for selected single-feature interventions. Values show changes from the unsteered baseline; performance often peaks at moderate strengths and degrades at larger magnitudes.

treating the feature set and steering strength as configuration-sensitive choices rather than applying uniformly strong cluster-level interventions.

6.4 RQ4: Strength Sensitivity

Steering strength affects performance on both AmbiK and ClarQ-LLM, but its effect is not monotonic. On AmbiK, moderate values of α can improve task success and/or clarification rate, whereas larger strengths can saturate or degrade performance. This effect is especially visible for C-derived features in some Gemma-9B settings, while for Gemma-2B both vocabularies become less reliable as α increases. These results suggest that AmbiK steering is most effective in a limited strength range rather than under arbitrarily strong intervention.

On ClarQ-LLM, we observe a similar non-monotonic pattern. As Figure 6 shows, moderate strengths can improve success rate or step recall, but larger magnitudes often reduce one or both metrics. For example, the selected Gemma-9B C-derived feature performs best around $\alpha = 3$, whereas performance decreases at larger positive strengths. A similar pattern appears for Llama-1B, where selected features perform best at negative strengths but degrade substantially at $\alpha = 10$.

Overall, increasing α does not reliably increase clarification performance: depending on the dataset, model, vocabulary, and individual feature, stronger steering can improve coverage, saturate, or degrade success rate and step recall. This suggests that effective use of ClarifySAE requires careful strength selection rather than uniformly large interventions.

6.5 RQ5: Random Features

We compare vocabulary-selected features with 10 uniformly sampled Gemma-2-9B-IT SAE features from layer 20. Figure 5 shows that random-feature steering produces larger degradations at comparable shifts, while selected features achieve a better clarification–success trade-off. This suggests that the gains are not solely due to arbitrary activation perturbations, although broader matched controls across layers and models are needed to establish specificity.

7 Conclusion

In this work, we introduce **ClarifySAE**, an inference-time method for encouraging clarification-seeking in instruction-following LLMs by amplifying SAE features selected via ClarifyScore and OutputScore. On AmbiK, ClarifySAE can substantially increase clarification question-asking and improve task success under proxy resolution metrics, for example, on Gemma-2-9B-IT with Q-discovered features, clarification rate rises from **0.61** to **0.95** while task success improves from **0.06** to **0.21**. These results indicate that feature-level SAE interventions can modulate clarification behavior without additional training, complementing prompting and fine-tuning.

Ethical Considerations

In this paper we study clarification-seeking in a text-only setting, motivated by embodied assistants that receive an instruction together with a scene description. We use public benchmarks (AmbiK and ClarQ-LLM) and pretrained models/SAEs, and we do not collect new personal data or deploy the system in a physical robot.

Potential risks of LLM-based text generation include toxic or abusive content, social biases, and hallucinations. In our setting, these risks are mitigated by the task domain (robot instructions grounded in kitchen-like scenes), which is typically non-sensitive, and by the fact that the datasets are curated and human-validated.

A separate concern is misuse of behavior steering: feature-level interventions can be used to shape model outputs in ways that are not transparent to end users. In our work, the intervention is narrowly targeted to increase clarification questions under underspecification, and we report both benefits and trade-offs (e.g., over-asking) using established evaluation protocols.

Limitations

Model coverage. We evaluate ClarifySAE on four instruction-tuned models: Gemma-2B, Gemma-9B, Llama-1B, and Llama-8B. This broadens the evaluation beyond a single model family, but it is still limited to models for which compatible public SAE checkpoints are available. The SAE checkpoints also differ across model families in layer, training setup, and feature basis, so differences between Gemma and Llama results should not be interpreted as purely architectural effects.

Dataset coverage and evaluation scale. AmbiK and ClarQ-LLM capture ambiguity in specific formats (kitchen-robot instructions and task-oriented dialogue) and may underrepresent other underspecification (e.g., missing constraints, preferences, or conflicting goals). In addition, because feature steering requires sweeping features, vocabularies, and intervention strengths, our main steering experiments use fixed benchmark subsets rather than exhaustive full-dataset sweeps. We therefore interpret the results as evidence that SAE features can modulate clarification behavior under controlled benchmark settings, while larger-scale evaluation across broader domains and full test sets remains future work.

Interaction length. Our analysis primarily targets the decision to ask clarification questions rather than longer-horizon clarification strategies (e.g., iterative follow-ups, adapting questions based on user replies, or optimizing dialogue-level outcomes over multiple turns).

Feature interactions. Most analyses focus on single-feature interventions; our multi-feature experiments are exploratory. For simplicity we analyze single-feature interventions; clarification behavior may instead be encoded in interacting or distributed feature patterns (Ma et al., 2026). Future work can jointly steer feature subsets (e.g., sparse sets or learned linear combinations) to study interactions and reduce the required steering strength.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, and 1 others. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

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

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*.

Dana Arad, Aaron Mueller, and Yonatan Belinkov. 2025. Saes are good for steering–if you select the right features. *arXiv preprint arXiv:2505.20063*.

Reza Bayat, Ali Rahimi-Kalahroudi, Mohammad Pezeshki, Sarath Chandar, and Pascal Vincent. 2025. Steering large language model activations in sparse spaces, 2025. *URL https://arxiv.org/abs/2503.00177*.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. 2023. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, and 1 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, and 1 others. 2024. Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023. *URL https://arxiv. org/abs/2307.15818*, 1:2.

Seonglae Cho, Zekun Wu, and Adriano Koshiyama. 2025. Corrsteer: Generation-time llm steering via correlated sparse autoencoder features. *arXiv preprint arXiv:2508.12535*.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, and 1 others. 2023. Palm-e: An embodied multimodal language model.

Andrey Galichin, Alexey Dontsov, Polina Druzhinina, Anton Razzhigaev, Oleg Y Rogov, Elena Tutubalina, and Ivan Oseledets. 2025. I have covered all the bases here: Interpreting reasoning features in large language models via sparse autoencoders. *arXiv preprint arXiv:2503.18878*.

Yujian Gan, Changling Li, Jinxia Xie, Luou Wen, Matthew Purver, and Massimo Poesio. 2024. Clarq-llm: A benchmark for models clarifying and requesting information in task-oriented dialog. *arXiv preprint arXiv:2409.06097*.

Ruben Härle, Felix Friedrich, Manuel Brack, Björn Deiseroth, Patrick Schramowski, and Kristian Kersting. 2024. Scar: Sparse conditioned autoencoders for concept detection and steering in llms. *arXiv preprint arXiv:2411.07122*.

Zirui He, Mingyu Jin, Bo Shen, Ali Payani, Yongfeng Zhang, and Mengnan Du. 2025a. Sae-ssv: Supervised steering in sparse representation spaces for reliable control of language models. *arXiv preprint arXiv:2505.16188*.

Zirui He, Haiyan Zhao, Yiran Qiao, Fan Yang, Ali Payani, Jing Ma, and Mengnan Du. 2025b. Saif: A sparse autoencoder framework for interpreting and steering instruction following of language models. *arXiv preprint arXiv:2502.11356*.

Zhenglin Hua, Jinghan He, Zijun Yao, Tianxu Han, Haiyun Guo, Yuheng Jia, and Junfeng Fang. 2025. Steering lvlms via sparse autoencoder for hallucination mitigation. *arXiv preprint arXiv:2505.16146*.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Andrew Hundt, Rumaisa Azeem, Masoumeh Mansouri, and Martim Brandão. 2025. Llm-driven robots risk enacting discrimination, violence, and unlawful actions. *International Journal of Social Robotics*, pages 1–49.

Anastasia Ivanova, Bakaeva Eva, Zoya Volovikova, Alexey Kovalev, and Aleksandr Panov. 2025. Ambik: Dataset of ambiguous tasks in kitchen environment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33216–33241.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.

Kai Konen, Sophie Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik Opitz, and Tobias Hecking. 2024. Style vectors for steering generative large language model. *arXiv preprint arXiv:2402.01618*.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2022. Clam: Selective clarification for ambiguous questions with generative language models. *arXiv preprint arXiv:2212.07769*.

Belinda Z Li, Been Kim, and Zi Wang. 2025. Questbench: Can llms ask the right question to acquire information in reasoning tasks? *arXiv preprint arXiv:2503.22674*.

Kaiqu Liang, Zixu Zhang, and Jaime F Fisac. 2024. Introspective planning: Aligning robots' uncertainty with inherent task ambiguity. *Advances in Neural Information Processing Systems*, 37:71998–72031.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 3214–3252.

George Ma, Zhongyuan Liang, Irene Y Chen, and Somayeh Sojoudi. 2026. Do sparse autoencoders identify reasoning features in language models? *arXiv preprint arXiv:2601.05679*.

Thomas McGrath, Daniel Balsam, Myra Deng, and Eric Ho. 2024. Understanding and steering llama 3 with sparse autoencoders. *Goodfire. ai Research Blog*, 12.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1

others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Jeongeun Park, Seungwon Lim, Joonhyung Lee, Sangbeom Park, Minsuk Chang, Youngjae Yu, and Sungjoon Choi. 2023. Clara: classifying and disambiguating user commands for reliable interactive robotic agents. *IEEE Robotics and Automation Letters*, 9(2):1059–1066.

Matthew Purver, Patrick Healey, James King, Jonathan Ginzburg, and Greg J Mills. 2003. Answering clarification questions. In *Proceedings of the Fourth SIGdial Workshop of Discourse and Dialogue*, pages 23–33.

Paul Rayson and Roger Garside. 2000. Comparing corpora using frequency profiling. In *The workshop on comparing corpora*, pages 1–6.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Allen Z Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, and 1 others. 2023. Robots that ask for help: Uncertainty alignment for large language model planners. *arXiv preprint arXiv:2307.01928*.

Dong Shu, Xuansheng Wu, Haiyan Zhao, Daking Rai, Ziyu Yao, Ninghao Liu, and Mengnan Du. 2025. A survey on sparse autoencoders: Interpreting the internal mechanisms of large language models. *arXiv preprint arXiv:2503.05613*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Lyucheng Wu, Mengru Wang, Ziwen Xu, Tri Cao, Nay Oo, Bryan Hooi, and Shumin Deng. 2025a. Automating steering for safe multimodal large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 792–814.

Xuansheng Wu, Jiayi Yuan, Wenlin Yao, Xiaoming Zhai, and Ninghao Liu. 2025b. Interpreting and steering llms with mutual information-based explanations on sparse autoencoders. *arXiv preprint arXiv:2502.15576*.

Hangtao Zhang, Chenyu Zhu, Xianlong Wang, Ziqi Zhou, Changgan Yin, Minghui Li, Lulu Xue, Yichen Wang, Shengshan Hu, Aishan Liu, and 1 others. 2024a. Badrobot: Jailbreaking embodied llms in the physical world. *arXiv preprint arXiv:2407.20242*.

Michael JQ Zhang, W Bradley Knox, and Eunsol Choi. 2024b. Modeling future conversation turns to teach llms to ask clarifying questions. *arXiv preprint arXiv:2410.13788*.

Zhuoxuan Zhang, Jinhao Duan, Edward Kim, and Kaidi Xu. 2025. Sparse neurons carry strong signals of question ambiguity in llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 16092–16110.

Zhanke Zhou, Xiao Feng, Zhaocheng Zhu, Jiangchao Yao, Sanmi Koyejo, and Bo Han. 2025. From passive to active reasoning: Can large language models ask the right questions under incomplete information? *arXiv preprint arXiv:2506.08295*.

A Appendix – Reproducibility and Code

We conduct experiments using four instruction-tuned language models of various sizes and families: `gemma-2b-it`, `gemma-2-9b-it`, `llama-3.2-1b-instruct` and `llama-3.1-8b-instruct`. We use publicly released sparse autoencoder (SAE) checkpoints trained on residual-stream activations (Bricken et al., 2023; Cunningham et al., 2023). For Gemma, we use community-released SAEs for `gemma-2b-it` and Gemma Scope SAEs (Lieberum et al., 2024) for `gemma-2-9b-it`. For Llama, we use model-specific public SAE releases: BatchTopK residual-stream SAEs for `Llama-3.1-8B-Instruct` from `andyrdt/saes-llama-3.1-8b-instruct`, and residual-output SAEs for `Llama-3.2-1B-Instruct` from `apart/llama3.2_1b_instruct_saes_vader`. For each backbone model, we keep the SAE layer and checkpoint fixed across experiments, providing a reproducible feature basis within each model.

To support reproducibility, we include the implementation of the proposed ClarifySAE method and the code used to reproduce the experiments as supplementary material. The code will be released as open source upon paper acceptance.

B Appendix – Vocabularies

In this appendix we list the exact vocabularies used to define clarification-associated lexical signals for feature discovery.

Question words vocabulary. This vocabulary is a small, hand-written list of common question-forming tokens. We include it as a simple alternative signal to the dataset-derived vocabulary, to test whether a compact, manually specified list can identify SAE features that steer clarification behavior comparably well or better than a data-driven construction. The vocabulary consists of 11 words:

```
"what"
"where"
"when"
"why"
"who"
"how"
"which"
"could"
"would"
"can"
"will".
```

Clarification vocabulary. This vocabulary is constructed using two-corpus frequency profiling (Rayson and Garside, 2000). We extract candidate n–grams from the ambiguous- instruction corpora and retain those that are substantially more frequent than in a general instruction corpus. The resulting list is intended to capture lexical markers that co-occur with underspecification and clarification contexts. We use this vocabulary as the primary signal for feature discovery. The vocabulary consists of 20 n–grams:

```
"do i need"
"what should i"
"should i take"
"should i use"
"can i use"
"where should i"
"which of my"
"which type of"
"is there a specific"
"to clarify"
"advise me on how"
"advise me on the best"
"you advise me on the"
"can you tell me"
"could you tell me where"
"could you please tell me"
"can you guide me on"
"else i should be aware"
"i should be aware of"
"else i need to know"
```

Algorithm 1 ClarifySAE

Inputs: Instruction x, model M, SAE, strengths $\{\alpha_j\}$
Output: Response y
Part I: Offline feature discovery (run once)
Build clarifying corpus C_{clar} and baseline corpus C_{base}
Extract clarification vocabulary V_{clar}
for each SAE feature j **do**
  Compute ClarifyScore_j (Eq. F.1) and OutputScore_j (Eq. F.2)
end for
Select features F with high scores
Part II: Online inference-time steering (per prompt)
Initialize model state on x
for each decoding step t **do**
  $z_t \leftarrow \text{SAE}(h_t)$
  for each $j \in F$ **do**
    $z_{t,j} \leftarrow z_{t,j} + \alpha_j$
  end for
  $h_t \leftarrow \text{SAE}^{-1}(z_t)$
  Generate next token
end for
return y

C Appendix – ClarifySAE algorithm

Algorithm 1 summarizes the full ClarifySAE pipeline, including (i) vocabulary construction, (ii) feature scoring and filtering (ClarifyScore and OutputScore), and (iii) inference-time feature steering.

D Appendix – AmbiK Metrics

This appendix defines the evaluation metrics we use for AmbiK. For reproducibility, we provide an implementation in our codebase; the definitions below are the metrics reported in the paper.

Notation. Each example i has an ambiguity type $c_i \in \mathcal{C}$, a gold clarifying question g_i, and a (possibly empty) list of model-generated questions $Q_i = \{q_{i1}, \ldots, q_{ik_i}\}$ with $k_i = |Q_i|$.

Best-match similarity and resolution indicators. We define the embedding-based (Reimers and

Gurevych, 2019) best-match similarity as

$$s_i = \max_{q \in Q_i} \mathrm{BM}(q, g_i), \tag{1}$$

where $\mathrm{BM}(\cdot,\cdot)$ denotes the best-match scoring function. We then define the proxy resolution indicator

$$r_i = \mathbb{1}[s_i \geq \tau], \tag{2}$$

where τ is the embedding threshold (`embed_threshold`).

Analogously, we define an NLI-based best-match similarity

$$s_i^{\mathrm{NLI}} = \max_{q \in Q_i} \mathrm{NLI}(q, g_i), \tag{3}$$

and the corresponding resolution indicator

$$r_i^{\mathrm{NLI}} = \mathbb{1}\left[s_i^{\mathrm{NLI}} \geq \tau_{\mathrm{NLI}}\right], \tag{4}$$

where τ_{NLI} is the NLI threshold. If not specified, we set $\tau_{\mathrm{NLI}} = \tau$.

Resolved rates. Let N be the number of examples. The embedding-based resolved rate is

$$\mathrm{ResolvedProxyRate} = \frac{1}{N} \sum_{i=1}^{N} r_i, \tag{5}$$

and the NLI-based variant is

$$\mathrm{ResolvedProxyRate}_{\mathrm{NLI}} = \frac{1}{N} \sum_{i=1}^{N} r_i^{\mathrm{NLI}}. \tag{6}$$

Question count statistics. We report a histogram of k_i and the mean number of questions:

$$\mathrm{AvgNumQuestions} = \frac{1}{N} \sum_{i=1}^{N} k_i. \tag{7}$$

Per-category similarity (asked-only). Let $I_c = \{\, i : c_i = c \wedge k_i > 0 \,\}$ denote the set of indices associated with class c. For each category $c \in \mathcal{C}$, we define the per-category similarity as the average best-match similarity over examples where the model asked at least one question:

$$\mathrm{Sim}(c) = \frac{1}{|I_c|} \sum_{i \in I_c} s_i. \tag{8}$$

and similarly $\mathrm{Sim}_{\mathrm{NLI}}(c)$ using s_i^{NLI}.

Necessity precision/recall and necessity score. AmbiK considers `preferences` as the category in which a clarification question is *necessary*. Define

$$\mathrm{Asked}_i = \mathbb{1}[k_i > 0]. \tag{9}$$

Let $I_{\mathrm{ask}} = \{\, i : \mathrm{Asked}_i = 1 \,\}$ and $I_{\mathrm{pref}} = \{\, i : c_i = \texttt{preferences} \,\}$. We compute true/false positives and false negatives as:

$$\mathrm{TP} = |I_{\mathrm{pref}} \cap I_{\mathrm{ask}}|. \tag{10}$$

$$\mathrm{FP} = |(\overline{I_{\mathrm{pref}}}) \cap I_{\mathrm{ask}}|. \tag{11}$$

$$\mathrm{FN} = |I_{\mathrm{pref}} \cap (\overline{I_{\mathrm{ask}}})|. \tag{12}$$

Then

$$\mathrm{NecessityPrecision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{13}$$

$$\mathrm{NecessityRecall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{14}$$

Let $N_{\texttt{pref}} = |\{\, i : c_i = \texttt{preferences} \,\}|$. We also calculate a *necessity score* (recall on the `preferences` class):

$$\mathrm{NecessityScore} = \frac{\mathrm{TP}}{N_{\texttt{pref}}}. \tag{15}$$

Brevity score. Let b be `brevity_max` (default $b = 1$). We define

$$\mathrm{BrevityScore} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}[k_i \leq b]. \tag{16}$$

Overall similarity (asked-only). We define overall similarity as the mean of s_i over examples where the model asked at least one question:

$$\mathrm{OverallSimilarity} = \frac{1}{|I_{ask}|} \sum_{i \in I_{ask}} s_i. \tag{17}$$

and similarly $\mathrm{OverallSimilarity}_{\mathrm{NLI}}$ using s_i^{NLI}.

Overall weighted score. Finally, we combine the above components into:

$$\begin{aligned} \mathrm{OverallWeighted} = {} & 0.5\,\mathrm{NecessityScore} \\ & + 0.4\,\mathrm{OverallSimilarity} \\ & + 0.1\,\mathrm{BrevityScore}. \end{aligned} \tag{18}$$

D.1 Slot-Alignment Rate

To evaluate whether generated clarification questions target the intended ambiguity, we compute slot-alignment rate (SAR). For each example where the model asks at least one question, we compare each generated question with the gold ambiguity slot description using semantic similarity and retain the maximum similarity score. $SAR@\tau$ is the fraction of asked examples whose best-match similarity is at least threshold τ. We report SAR at $\tau \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ (Table 5). Because SAR is conditional on asking, we report clarification rate alongside SAR to distinguish asking frequency from alignment quality.

E Appendix – AmbiK Evaluation Prompts

We list the prompts used in our AmbiK evaluation. For each example, we fill `<DESCRIPTION>` with the scene description (`environment_full`) and `<TASK>` with the ambiguous instruction (`ambiguous_task`). The model is required to return exactly one JSON object with fields `ambiguous` and `question`.

```
We: You are a robot operating in an office
↪  kitchen. You are in front of
a counter with two closed drawers, a top one and
↪  a bottom one. There is
also a landfill bin, a recycling bin, and a
↪  compost bin.
We: Apart from that, in the kitchen there is
↪  <DESCRIPTION>.
We: You have received the task "<TASK>".
Decide whether this task is ambiguous.
If it is ambiguous, generate one or more
↪  clarifying questions that
would help you perform it correctly.

Return your final answer only as a JSON object in
↪  the following format:

{
  "ambiguous": true or false,
  "question": ["question 1", "question 2", ...]
}

Formatting rules:
- Always output exactly one JSON object.
- If the task is not ambiguous, use "ambiguous":
↪  false and "question": [].
- If the task is ambiguous, use "ambiguous": true
↪  and provide at least
  one question.
- Never output explanations or text outside the
↪  JSON.
- If unsure, make your best judgment and still
↪  output a valid JSON.

Now analyze the following task:
"<TASK>"
```

F Appendix – Scoring Details

F.1 ReasonScore definitions

Let $\{G_1, \ldots, G_K\}$ be vocabulary groups (phrase occurrences) and $z_{t,j}$ be the activation of feature j at token position t.

Group-wise means and entropy. For each feature j and group G_k, define the group-wise mean activation

$$\mu_{j,k} = \frac{1}{|G_k|} \sum_{t \in G_k} z_{t,j}. \tag{19}$$

Normalize the group means into a discrete distribution

$$p_{j,k} = \frac{\mu_{j,k}}{\sum_{k'=1}^{K} \mu_{j,k'}}, \tag{20}$$

and compute normalized entropy

$$H_j = -\frac{1}{\log K} \sum_{k=1}^{K} p_{j,k} \log p_{j,k}, \tag{21}$$

with the convention $H_j = 1$ when $K = 1$.

Normalizers. Let K^{pos} and K^{neg} denote the sets of groups treated as positive and negative, respectively. Define the aggregated mean activations

$$\mu_j^{\text{pos}} = \sum_{k \in K^{\text{pos}}} \mu_{j,k}, \qquad \mu_j^{\text{neg}} = \sum_{k \in K^{\text{neg}}} \mu_{j,k}. \tag{22}$$

We use the global normalizers

$$S^{\text{pos}} = \sum_{j'} \mu_{j'}^{\text{pos}}, \qquad S^{\text{neg}} = \sum_{j'} \mu_{j'}^{\text{neg}}. \tag{23}$$

F.2 OutputScore definitions

For feature i, let $f_i \in \mathbb{R}^d$ denote its SAE decoder vector and $W_U \in \mathbb{R}^{d \times |V|}$ the model unembedding matrix. We compute the logit-lens projection (Belrose et al., 2023)

$$l_i = f_i^\top W_U \in \mathbb{R}^{|V|}, \tag{24}$$

and define the representative token as $v_i^\star = \arg\max_v l_i(v)$.

Feature-level intervention. Let $z(h)$ denote SAE activations for hidden state h, and let $z_{\max,i}$ be the maximum observed activation of feature i over a large corpus. At the intervention layer, we modify the activation

$$z_i' = z_i + \kappa \, z_{\max,i}, \tag{25}$$

Table 4: KnowNo (ask_F1). Str k denotes steering strength k.

Model	Vocab	Base	Str1	Str3	Str5	Str10
2B	C	**0.422**	0.390	0.406	0.388	0.392
2B	Q	**0.422**	0.411	0.390	0.403	0.400
9B	C	0.449	**0.451**	0.332	0.248	0.145
9B	Q	**0.449**	**0.449**	0.420	0.361	0.315

where κ is a fixed coefficient. Decoding the modified activations yields an output distribution $p_i(\cdot)$ at the final generation position. We define

$$p_i^\star = p_i(v_i^\star) \tag{26}$$

and let r_i be the rank of $v_i^\star$ under $p_i(\cdot)$.

G Appendix – KnowNo Evaluation

We additionally evaluate on KnowNo (Ren et al., 2023), which provides mobile manipulation instructions paired with structured environment descriptions and annotated instruction types.

For each example, we run a two-step interaction: (1) a *clarification step* where the model outputs a JSON object indicating whether the instruction is ambiguous and, if so, a list of clarifying questions; and (2) a *final plan step* where the model outputs a short plan conditioned on a simulated user reply. When the model asks a question, we simulate the reply by selecting the first option in the annotated `user_intent` (field values are separated by `|`) and providing the annotated `intent_location`.

We compute ask_F1 for the binary decision to ask a clarification question, treating the following KnowNo types as requiring clarification: `spatial_ambiguous_task`, `multilabel_task`, and `creative_multilabel_task`.

As shown in Table 4, steering does not improve ask_F1 on KnowNo and can degrade performance at higher strengths. Table 6 illustrates that this degradation is driven by shifts in the asking propensity: some steered configurations fail to ask on ambiguous instances, while others introduce extra questions on unambiguous inputs.

H Appendix – Examples

We provide qualitative context for the steering features used in the main experiments. Table 8 lists representative ClarifyScore-selected features (after OutputScore filtering) together with their top-decoded tokens, illustrating the types of lexical cues captured by each feature. Table 7 shows paired generations for the same input under baseline vs. steering settings, highlighting how steering changes question-asking behavior. Table 9 reports the manually selected features used in experiments, including their OutputScore values and top-decoded tokens for interpretability. Some useful steering features have unintuitive top-token projections, especially when the decoder direction affects generation through downstream context rather than directly projecting to an obvious clarification word. We therefore use these tables as qualitative diagnostics, not as proof that each feature is monosemantic or fully interpretable.

I Appendix – Analysis of ClarQ-LLM steering

To determine whether metric improvements correspond to better clarification behavior, we compared baseline and steered dialogues for the same task instances (Table 10). The qualitative evidence is mixed. In some cases, steering makes questions more task-specific, replacing generic requests for more information with questions about missing constraints. In other cases, metric gains are achieved by asking more questions, repeating similar questions, or producing less natural dialogue. For instance, some steered outputs repeatedly ask about the same route choice or use unnatural phrasing, even when step recall improves.

Thus, success rate and step recall should be interpreted as measures of task-aligned information coverage, not as direct measures of clarification quality.

Table 5: Slot-alignment for single-feature steering.

Model	Vocab	Layer	Str	Clar	SAR@0.5	SAR@0.6	SAR@0.7	SAR@0.8	SAR@0.9
2B	C	12	1.0	**0.44**	**0.676**	0.528	0.267	**0.142**	0.028
2B	C	12	3.0	0.26	0.600	0.438	0.200	0.069	0.031
2B	C	12	15.0	0.15	0.607	0.443	0.197	0.066	**0.033**
2B	C	12	30.0	0.15	0.607	0.443	0.197	0.066	**0.033**
2B	Q	12	1.0	**0.38**	**0.665**	0.483	0.281	**0.095**	**0.011**
2B	Q	12	3.0	0.13	0.342	0.184	0.105	0.000	0.000
2B	base	-	0	0.36	0.611	**0.528**	**0.361**	0.083	0.000
9B	C	20	1.0	0.62	**0.649**	0.467	0.245	0.126	0.032
9B	C	20	2.0	0.63	0.616	0.450	0.245	**0.142**	**0.047**
9B	C	20	3.0	0.59	0.291	0.212	0.120	0.063	0.017
9B	C	20	4.0	0.52	0.618	0.437	**0.256**	0.129	0.029
9B	C	31	1.0	**0.64**	0.641	**0.469**	0.254	**0.142**	0.042
9B	C	31	2.0	0.62	0.644	0.451	0.253	**0.142**	0.046
9B	C	31	3.0	0.47	0.628	0.449	0.244	0.133	0.042
9B	C	31	4.0	0.38	0.622	0.435	0.238	0.125	0.045
9B	C	31	10.0	0.43	0.642	0.451	0.232	0.128	0.038
9B	Q	20	1.0	0.64	**0.650**	**0.476**	0.255	0.144	0.038
9B	Q	20	3.0	0.63	0.649	0.466	**0.258**	**0.155**	**0.052**
9B	Q	20	15.0	**0.67**	0.628	0.438	0.243	0.144	0.037
9B	Q	20	30.0	0.57	0.566	0.409	0.240	0.125	0.042
9B	base	-	0	0.61	0.607	0.443	0.197	0.066	0.033

Table 6: Qualitative KnowNo examples. Since Table 4 averages over multiple feature runs per strength, we report the frequency of asking under S_{10} as X/Y runs.

Model	Type	Instruction	Base (ask/question)	S_{10} (ask rate; example)
9B–C	`multilabel_task`	Bring me a cola.	Ask: "Would you like Pepsi or Coke?"	Ask in 5/20 runs; otherwise no question. Example ask: "Pepsi or Coke?"
9B–C	`spatial_ambiguous_task`	Put bottled water in the drawer.	Ask: "Which drawer?"	Ask in 3/20 runs; otherwise no question. Example ask: "Which drawer would you like me to put the water in?"
2B–C	`creative_multilabel_task`	Bring me something with a kick.	Ask: "What is something that can give you a kick?"	Ask in 0/4 runs (never asks).
2B–C	`unambiguous_task`	I need that kettle chips.	No question.	Ask in 4/4 runs. Example: "Where are the kettle chips located in this environment?"
2B–C	`spatial_ambiguous_task`	Put orange beside the bottle.	Ask: "Which item should be placed beside the bottled water?"	Ask in 4/4 runs. Example: "Where should the orange be placed in the environment?"

Table 7: Paired baseline vs. steered examples of clarification questions on AmbiK dataset. Each pair uses the same task instance with different steering settings.

Task ID	Feature ID	Steering strength	Gold question	Baseline questions	Steered questions
Steering improves question quality:					
14	93	1.0	Where should you serve the patties after cooking them?	1. Which pan should I use? 2. Where is the sour cream? 3. Do you have chopped green onions, or should I chop some?	1. Which pan should I use? 2. Do you have sour cream available? 3. Where are the green onions?
51	344	4.0	Where should I place the eggs?	1. What is the specified location?	1. Where should I place the eggs?
59	480	15.0	What should be used to blend all the ingredients together?	1. Where is the 'suitable appliance'?	1. What appliance should I use to blend the ingredients? 2. Where are the bowls located?
127	550	3.0	Which ingredient should I use for the omelette, apart from eggs?	1. Which cheese would you like me to use? 2. Do you have a preference for cheddar or cottage cheese?	1. Which of the available ingredients should I use for the omelette?
309	166	15.0	Which seedless grapes should I use for serving?	1. Which plate should I wash?	1. Which plate should I wash? 2. What kind of seedless grapes should I use?
341	12423	15.0	What specific vegetables should be combined in the bowl?	1. Which bread should I use? 2. What type of spread would you like me to use?	1. What kind of spread should I use on the bread? 2. What vegetables should I combine in the bowl?
Steering makes no improvement:					
113	493	15.0	Which type of bread should be toasted in the toaster for this task?	1. Is there a bowl available? 2. Which type of bread should I use?	1. Is there a bowl available for me to use?
312	2856	3.0	Which type of plate should I use to arrange the donuts?	1. Which plate should I use for the pastries? 2. Where is the table located?	1. Which resource is designated as 'the table'? 2. What type of resource constitutes 'other resources'? 3. Which resource is designated as 'a cup'?

Table 8: Representative features from the **K=50** sets (OutputScore > 0). For each condition, we report the top-ranked feature within each partition (Q-only, Overlap, C-only), ranked by OutputScore within that partition, together with its id and top-tokens. The examples show that question- and clarification-derived feature sets can emphasize different surface cues. Token glosses are not unique identifiers: distinct features can share very similar top tokens while still being different features with different scores.

Model	Layer	Chunk	Set	Feature ID	OutputScore	Top tokens
9B	20	0000	Q-only	232	0.117	["is", "represents", "defaultstate", "verhe"]
9B	20	0000	Overlap	527	0.0061	["", "Ready", "Private", "Practice", "John"]
9B	20	0000	C-only	218	0.0006	["\n\n", "WebServlet", "werb", "DeltaTime"]
2B	12	0000	Q-only	103	0.979	["<bos>", "fta", "ftu", "effe", "desir"]
2B	12	0000	Overlap	210	0.00001	["yourself", "your", "personally"]
2B	12	0000	C-only	439	0.0201	["However", "Therefore", "But", "Yet", "antik"]
2B	12	0001	Q-only	538	0.0063	["wasn", "seemed", "was", "resulted", "reflects"]
2B	12	0001	Overlap	505	0.9995	["<bos>", "GEBURTSDATUM", "expandindo", "betweenstory", "Autoritní"]
2B	12	0001	C-only	492	0.9999	["<bos>", "GEBURTSDATUM", "expandindo", "betweenstory", "Autoritní"]

Table 9: Manually selected features used in experiments and their corresponding OutputScore and top-decoded tokens.

Model	Vocab	Layer	Chunk	Feature ID	OutputScore	Top tokens
2B	C	12	0000	439	0.0201	["However", "Therefore", "But", "Yet", "antik"]
2B	C	12	0001	375	0.0008	["none", "only", "most", "ones", "one"]
2B	C	12	0001	432	0.9736	["physical", "physical", "Physical", "Physical", "PHYSICAL"]
2B	C	12	0001	557	0.2422	["it", "there", "they", "if", "we"]
2B	Q	12	0000	231	0.3613	["you", "your", "please", "youre", "your"]
2B	Q	12	0000	700	0.0006	["options", "providers", "brands", "Options", "options"]
2B	Q	12	0000	791	0.0001	["should", "needs", "solutions", "implementation"]
2B	Q	12	0001	375	0.0008	["none", "only", "most", "ones", "one"]
2B	Q	12	0001	538	0.0062	["wasn", "seemed", "was", "resulted", "reflects"]
2B	Q	12	0001	557	0.2422	["it", "there", "they", "if", "we"]
2B	Q	12	0001	688	0.0005	["strict", "rigid", "precision", "uniform", "disciplined"]
9B	C	20	0002	344	0.0267	["both", "begge", "Both", "ambos", "both"]
9B	C	20	0002	383	0.0008	["everything", "enterOuterAlt", "Everything", "Everything", "Bronnen"]
9B	C	20	0002	565	0.0002	["price", "Price", "priced", "price", "Price"]
9B	C	20	0002	591	≈ 0	["now", "Now", "Now", "now", "ahora"]
9B	C	20	0002	688	0.0002	["system", "système", "systeem", "sistema", "ssystem"]
9B	C	20	0002	771	0.9917	[".", ",", ";", "because", ".."]
9B	C	31	0000	124	0.2031	["we", "you", "it", "they", "I"]
9B	C	31	0000	158	0.6313	["some", "quelques", "some", "einige", "Some"]
9B	C	31	0000	294	0.0007	["tell", "Tell", "tale", "Tell", "TELL"]
9B	C	31	0000	308	0.0875	["how", "why", "cómo", "bagaimana", "how"]
9B	C	31	0000	540	≈ 0	["Java", "java", "Java", "Javan", "JAVA"]
9B	C	31	0000	550	0.0690	["my", "myself", "mijn", "I", "meiner"]
9B	C	31	0000	631	0.6533	["be", "have", "become", "take", "come"]
9B	C	31	0000	778	0.0083	["she", "he", "they", "you", "we"]
9B	C	31	0001	52	0.9976	[",", "highly", "yet", "colourful"]
9B	C	31	0001	93	0.5488	["thorough", "thoroughly", "comprehensive", "Thorough", "exhaustive"]
9B	C	31	0001	124	0.2031	["we", "you", "it", "they", "I"]
9B	C	31	0001	130	0.0528	["empty", "vacant", "missing", "emptied", "Empty"]
9B	C	31	0001	294	0.0007	["tell", "Tell", "tale", "Tell", "TELL"]
9B	C	31	0001	467	0.9956	["-", "-", "chrétien", "-", "Inscrivez"]
9B	C	31	0001	720	0.9551	["{", "({", "{", "",",]
9B	Q	20	0000	229	0.001155	["global", "worldwide", "globally", "weltweit", "global"]
9B	Q	20	0000	480	0.0003	["similar", "other", "same", "another", "same"]
9B	Q	20	0000	748	≈ 0	["I", "1", "\n", "\n\n", "Is"]
9B	Q	20	0001	113	≈ 0	["free", "gratis", "free", "FREE", "gratuite"]
9B	Q	20	0001	166	≈ 0	["also", "numerous", "an", "two", "many"]
9B	Q	20	0001	318	0.0005	["December", "month", "June", "December", "January"]
9B	Q	20	0001	455	≈ 0	["goal", "Goal", "goal", "Goal", "GOAL"]
9B	Q	20	0001	493	≈ 0	["member", "member", "Member", "Member", "members"]
9B	Q	20	0001	711	0.0016	["simple", "tiny", "simple", "piccoli", "semplici"]
9B	Q	20	0001	771	0.9917	[".", ",", ";", "because", ".."]

Table 10: Pairwise qualitative examples on 10 dialogue ClarQ-LLM evaluation. Each row compares baseline and steered generations on the same task instance. Arrows show metric changes from baseline to steered generation. The examples show that improvements in success rate or step recall can correspond to more targeted questions, but can also coexist with repetition, unnatural wording, or role-play artifacts.

Case	**Configuration**	**Task**	**Baseline behavior**	**Steered behavior**	**Interpretation**
Incomplete improvement	Gemma-2B, Q, $f = 538$, $\alpha = -5$	Collecting artifacts	Asks a generic opening question about available shoes; success $0 \rightarrow 1$, step recall $0 \rightarrow 1$.	Asks how to access the cave and mountains that contain the artifacts.	Steering makes the question more specific and shorter than the baseline, but does not cover all missing details.
Mixed improvement	Gemma-9B, C, $f = 344$, $\alpha = 3$	Collecting soul fragments	Asks about mountain order and paths; success $0 \rightarrow 1$, step recall $0.17 \rightarrow 1$.	Asks for collection order, but repeats similar order/path questions.	The metric gain partly reflects broader coverage, but also comes with redundant questioning.
No qualitative gain	Llama-8B, Q, $f = 62124$, $\alpha = 10$	Collecting diamonds	Asks relevant questions about forest choice, protective item, and digging method; success $1 \rightarrow 1$, step recall $1 \rightarrow 1$.	Covers the same information but contains unnatural repetitions and malformed wording.	There is no metric improvement, and question quality appears worse.
Failure	Llama-1B, C, $f = 6230$, $\alpha = -5$	Collecting emeralds	Drifts into narrative role-play and does not recover the required steps; success $0 \rightarrow 0$, step recall $0 \rightarrow 0$.	Steering increases neither coverage nor usefulness.	This illustrates that steering can fail by changing style without improving clarification.