Steering Large Language Models Toward Clarification through Sparse Autoencoders

Anonymous ACL submission

Abstract

Instruction-tuned LLMs often respond to ambiguous instructions by guessing missing details rather than asking clarifying questions. Clarification-seeking improves reliability by aligning responses with user intent and avoiding under-specified assumptions. This is especially important for embodied AI, where misinterpretations can translate into task failure or safety risks. We propose **ClarifySAE**, an inference-time method that steers clarification-seeking by intervening on Sparse Autoencoder (SAE) features. ClarifySAE ranks SAE features using ClarifyScore, which measures association with clarification contexts, and filters them with OutputScore to retain features that measurably affect the model's output distribution. During decoding, we apply additive biases to the selected features, increasing the likelihood of generating a clarifying question without updating model weights. We evaluate our method on two datasets with ambiguous instructions (AmbiK and ClarQ-LLM) and two Gemma instruction-tuned models (2B and 9B) using pretrained 16k-feature SAEs. On AmbiK with Gemma-2-9B-IT, ClarifySAE increases clarification rate from 0.61 to 0.95 and improves task success from 0.06 to 0.21. Our code will be 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; Zitkovich et al., 2023). 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 "chip" options (e.g., jalapeño, potato, or rice chips), and the user does not specify which one they mean. If a robot acts on an incorrect interpretation, the result may range from task failure to unsafe behavior (Zhang et al., 2024a). In safety-critical, user-facing HRI deployments, agents must act in the physical world under underspecified natural-language instructions. Proceeding without clarification can significantly limit the reliability and trustworthiness of LLM-driven agents (Amodei et al., 2016).

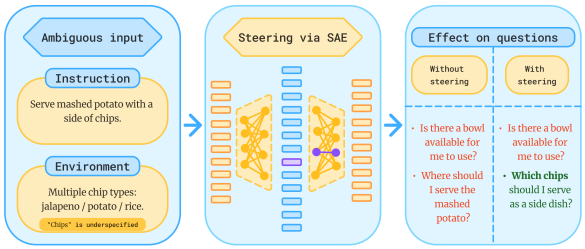

Figure 1: Overview of ClarifySAE. The method consists of constructing a clarification signal, identifying clarification-related SAE features, and applying feature-level steering during inference.

Despite their strong instruction-following performance (Ouyang et al., 2022; Wei et al., 2021), current LLMs rarely ask clarifying questions when ambiguity is present. Instead, they tend to produce fluent and confident responses, often implicitly guessing missing details (Lin et al., 2022; Ji et al., 2023). This tendency is particularly problematic for robots operating in the physical world, where incorrect assumptions may translate directly into irreversible physical actions (Hundt et al., 2025; Ahn et al., 2022). 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 responsible 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.

To identify clarification-related 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 and ClarQ-LLM using Gemma-2B-IT and Gemma-2-9B-IT. Activating a small set of selected SAE features increases clarification question-asking on AmbiK—e.g., for Gemma-2-9B-IT the clarification rate rises from 0.61 to 0.95 while task success improves from 0.06 to 0.21 (Figure 3). Effects are smaller and less stable for Gemma-2B-IT and on ClarQ-LLM (Section 6). 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 datasets across two instruction-tuned model scales (2B and 9B), observing increased clarification behavior relative to prompt-based 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. Closely related, He et al. (2025) 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., 2025). Arad et al. (2025) introduce Output Score, a metric that quantifies how much intervening on an individual SAE feature affects the model's output. They show that selecting features with higher Output Scores yields more reliable and semantically meaningful steering. Relatedly, Wu et al. (2025) introduce AutoSteer, an inference-time intervention framework for safe multimodal LLMs that automates the choice of steering signals.

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 Output Score, 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 targeted 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, our work explores a mechanistic route: we intervene directly on internal model activations to encourage clarification-seeking when ambiguity is detected. 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.

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 (e.g., detective cases and situation puzzles) 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 whose hidden states we analyze and steer. SAEs provide a linear, sparsity-constrained decomposition of model hidden states into interpretable features. Given a hidden representation $h \in \mathbb{R}^d$, the encoder produces nonnegative sparse activations $z = \mathrm{ReLU}(W_{\mathrm{enc}}h + b_{\mathrm{enc}})$ with $z \in \mathbb{R}^m_{\geq 0}$, and the decoder reconstructs $\hat{h} = W_{\mathrm{dec}}z + b_{\mathrm{dec}}$ with $\hat{h} \in \mathbb{R}^d$. Here $W_{\mathrm{dec}} \in \mathbb{R}^{d\times m}$ maps feature activations back into the hidden-state space, and $b_{\mathrm{dec}} \in \mathbb{R}^d$ is a learned bias term added to the reconstruction. SAEs are typically trained by minimizing a reconstruction objective with a sparsity penalty, $\mathcal{L}_{\mathrm{SAE}} = \mathbb{E}_{h\sim\mathcal{H}}\Big[\|h - \hat{h}\|_2^2 + \lambda\|z\|_1\Big]$, where $\mathcal{H}$ denotes the distribution of hidden states extracted from M and λ controls sparsity. Recent work shows that individual SAE features often correspond to meaningful directions and that intervening on selected features can steer model 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 an intervention strength α, we define the steered activation vector as

$$z'_j = \begin{cases} z_j + \alpha, & j \in S, \\ z_j, & j \notin S, \end{cases} \tag{1}$$

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 increase the steering effect but can degrade generation quality, so we tune α on validation data.

To select S, we use two established metrics. Let V denote a target vocabulary and let x range over inputs. ClarifyScore (adapted from ReasonScore (Galichin et al., 2025)) measures how strongly a feature's activation is associated with occurrences of tokens from V. Output Score (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 Output Score to prioritize those with a measurable causal effect on generation.

Following AmbiK (Ivanova et al., 2025), we call an instruction ambiguous if, given the current state of the environment, constructing a plan to satisfy the instruction contains at least one decision point with multiple plausible choices (e.g., selecting the target object or destination), where an incorrect choice can lead to an undesirable outcome. Formally, given an instruction u (and, in robotic tasks, an optional scene description s), the model chooses between producing an answer a or a question q that requests missing information.

4 Method

When an instruction is underspecified or admits multiple interpretations, a reliable robot-assistant should ask for missing information rather than guess. We therefore aim to steer an instruction-tuned language model M toward generating a clarification question in such cases.

We propose ClarifySAE method for steering decoding by intervening in a SAE feature space. At each step t, we map the hidden state h_t to sparse activations z_t, select a small set of features that are associated with clarification contexts and whose amplification affects the output distribution, and then add learned biases to these features at inference time. Decoding the modified activations back to a hidden state nudges next-token probabilities toward clarification while leaving unambiguous cases largely unchanged. The full procedure is illustrated in Figure 2 and specified in Algorithm 1.

4.1 Clarification Vocabulary

We use a lexical signal as a lightweight form of supervision: explicit clarifying turns provide a reliable source of text whose surface form often contains characteristic clarification markers. This is in line with ClarifyScore-style analyses, which link SAE features to behaviors via user-defined vocabularies (originally for reasoning tokens) (Galichin et al., 2025). To obtain a focused set of markers, we extract terms that are statistically overrepresented in clarifying turns relative to non-clarifying text, following standard corpus frequency-profiling methods (Rayson and Garside, 2000). This assumption is supported by prior dialogue work showing that clarification requests exhibit recurring lexical forms (e.g., "what do you mean", "which one") (Purver et al., 2003). From AmbiK, we include all annotated clarifying questions. From ClarQ-LLM, we collect seeker utterances that satisfy simple interrogative heuristics (e.g., ending with a question mark; starting with a wh-word or a modal verb; or containing short clarification cues such as "do you mean" or "could you tell me"). These heuristics provide an automatic filter for isolating seeker turns that are likely to be clarification requests in multi-turn dialogues. We remove provider responses, speaker prefixes, and polite fillers to prevent dataset artifacts from dominating the derived vocabulary.

To form a contrast corpus, we also collect non-clarifying text from the same datasets, including unambiguous instructions, final answers, and provider turns. The two corpora thus approximate contexts where clarification is versus is not required. To identify lexical markers characteristic of clarifying questions, we perform a two-corpus frequency comparison. After normalization and tokenization, we extract all contiguous 3–5-gram sequences and compute, for each phrase, its frequency in the clarifying corpus and the baseline corpus. We then apply the log-likelihood statistic commonly used in corpus linguistics for keyword extraction (Rayson and Garside, 2000), retaining only n–grams that occur significantly more often in clarifying text. Low-frequency, stopword-dominated, or dataset-specific phrases are removed, and candidates are required to contain at least one interrogative or clarification cue.

Finally, we collapse nested or overlapping phrases and select a small set of twenty broadly applicable clarification phrases (e.g., "do I need", "what should I", "which type of", "is there a specific", "can you tell me"). We choose a fixed list of 20 phrases as a practical trade-off between coverage and specificity: a very small list risks missing common clarification forms, while a much larger list tends to introduce dataset-specific or overly narrow n–grams that dilute the signal. This choice is also consistent with prior ReasonScore-style analyses, which use compact, hand-curated vocabularies (e.g., 10 reasoning-related phrases in (Galichin et al., 2025)). This vocabulary provides a compact lexical signal for clarification behavior and is later used in identifying clarification-related SAE features. Full vocabulary lists are provided in Appendix B.

4.2 Identifying SAE Features

To steer clarification behavior, we select a small subset of SAE features. We combine ClarifyScore to retrieve features associated with a clarification

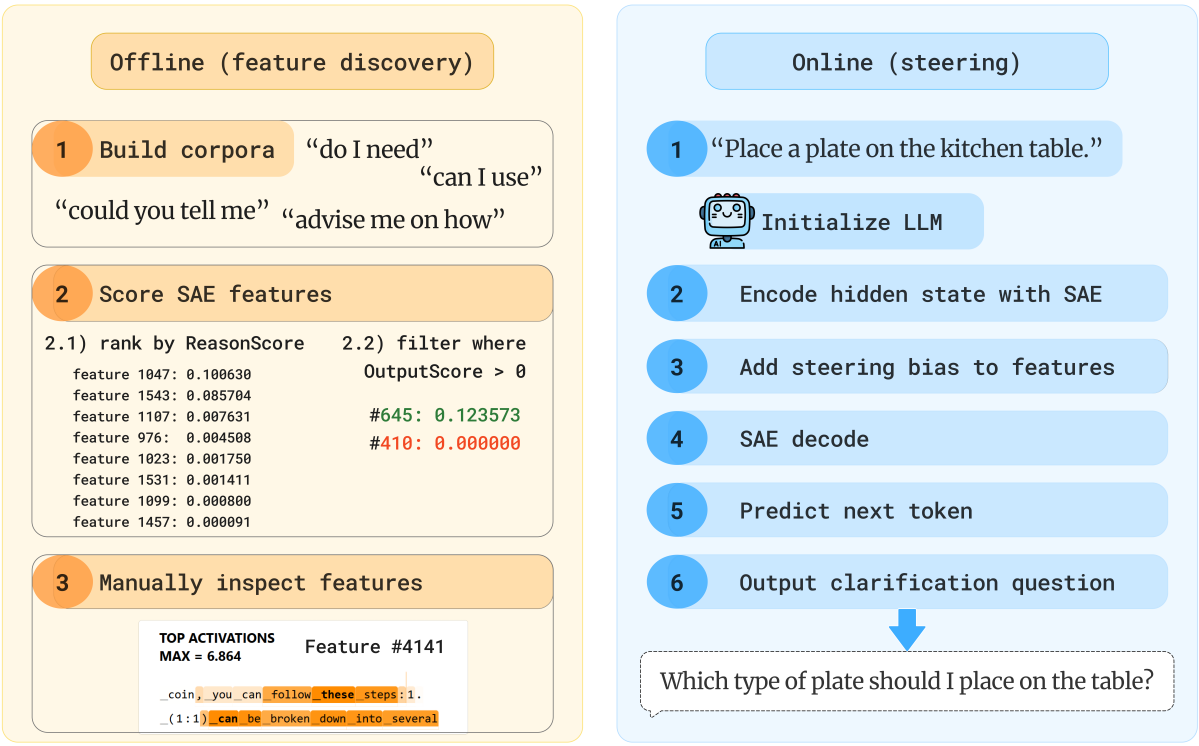

Figure 2: **ClarifySAE method overview.** ClarifySAE induces clarification-seeking by identifying a small set of SAE features that act as an internal "clarification" control knob and amplifying them at inference time. Offline, we construct a lexical signal for clarification from clarifying vs. non-clarifying corpora and use it to find SAE features whose activations co-occur with clarification cues; we then retain features that also have a measurable causal influence on the model's output distribution. Online, we add a bias to these selected features during decoding, nudging generation toward asking a clarifying question.

vocabulary and OutputScore to keep only those with measurable causal influence on generation; their full formal definitions are provided in Appendix E.1 and Appendix E.2.

ClarifyScore. ClarifyScore measures how selectively a feature activates around occurrences of a clarification vocabulary in a clarifying corpus compared to a baseline corpus. Concretely, for each feature j we compute its mean activation on token positions matched by the clarification vocabulary (positive contexts) and on all other positions (negative contexts), optionally weighted by an entropy-based concentration term over vocabulary groups. Features with high ClarifyScore activate strongly in clarification contexts while remaining relatively inactive elsewhere.

OutputScore. ClarifyScore captures correlation but does not guarantee that amplifying a feature changes generation. OutputScore therefore filters features by a simple intervention test: we increase feature j by a fixed amount at a chosen layer during decoding and measure whether this increases the probability of an interpretable representative token for that feature (e.g., defined via a logit-lens projection). We retain features with $\mathrm{OutputScore} > 0$.

Final Feature Set. We take the top-K features by ClarifyScore, filter them by $\mathrm{OutputScore} > 0$, and then apply a lightweight manual coherence check by inspecting each feature's top decoded tokens. Only features judged semantically coherent are included in the final steering set F.

4.3 Feature Steering

We apply feature-level steering during decoding. At each generation step t, the SAE maps the hidden state h_t to a feature vector z_t. For each selected feature $j \in F$, we apply a fixed additive shift:

$$z_{t,j} \leftarrow z_{t,j} + \beta_j, \tag{2}$$

where β_j controls the intervention strength. The modified activations are decoded back into h_t, after which the model proceeds with standard autoregressive generation (Radford et al., 2019).

This intervention provides a simple and interpretable mechanism for biasing the model toward

Algorithm 1 ClarifySAE

Inputs: Instruction I, model M, SAE, strengths $\{\beta_j\}$
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. E.1) and OutputScore_j (Eq. E.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**
        $z_{t,j} \leftarrow z_{t,j} + \beta_j$
    end for
    $h_t \leftarrow \text{SAE}^{-1}(z_t)$
    Generate next token
end for
return y

clarification-seeking behavior without retraining: we modify a small set of SAE features whose semantics are inspected via their representative tokens and activation patterns, and the steering strength is controlled by explicit coefficients. By modifying one feature at a time, we can separately analyze the behavioral impact of individual features, though this simplification does not capture potential interactions between features.

5 Experimental Setup

We conduct experiments with two instruction-tuned language models, `gemma-2b-it` and `gemma-2-9b-it`, and fixed residual-stream SAE feature bases for each model. We evaluate on two benchmarks: AmbiK (single-turn) and ClarQ-LLM (multi-turn). Using this setup, 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 signal—i.e., using question tokens (Q) versus clarification-context tokens (C)?
- **RQ3: Strength sensitivity.** How does steering strength α affect model behavior and performance?
- **RQ4: Random Features.** Are gains specific to discovered features, compared to random-feature steering?

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, common-sense knowledge, and safety. Each example includes an environment description, an ambiguous instruction plus two unambiguous rewrites (direct/indirect), a reference clarifying question with an answer, and step-by-step plans for both the ambiguous and unambiguous versions (with an index 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 are evaluated as seeker agents via interactive dialogue with a provider agent that replicates the original provider behavior.

5.2 Models and SAE Feature Bases

We conduct experiments using two instruction-tuned language models: `gemma-2b-it` and `gemma-2-9b-it`. We use publicly released residual-stream SAE (Bricken et al., 2023; Cunningham et al., 2023) checkpoints with 16k features for both models (community-released SAEs for `gemma-2b-it` and Gemma Scope SAEs (Lieberum et al., 2024) for `gemma-2-9b-it`), ensuring a fixed and reproducible feature basis.

5.3 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`

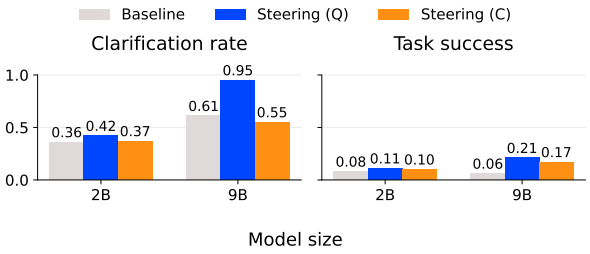

Figure 3: AmbiK evaluation on 100 examples. Clarification rate (left) and task success (right) for Gemma-2B-IT and Gemma-2-9B-IT. "Steering (Q)" and "Steering (C)" use features discovered with the question-words vs. dataset-derived clarification vocabulary. Question-vocabulary steering strongly increases clarification for 9B, whereas clarification-vocabulary steering slightly reduces clarification rate for 9B, despite improving task success over baseline.

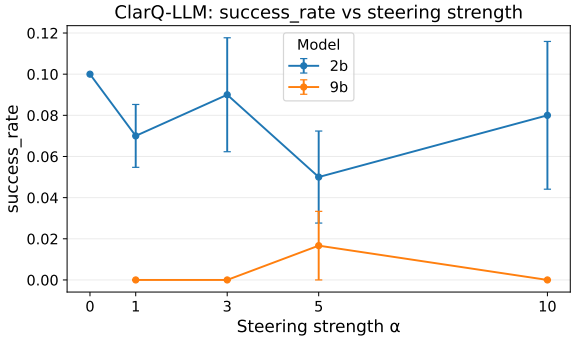

Figure 4: ClarQ-LLM: Success rate vs. steering strength. We report the mean strict success rate over features discovered with clarification vocabulary (C) for each steering strength α ; error bars show $\pm$SEM across features. Steering strength shows no consistent monotonic effect: 2B varies modestly across α, while 9B remains near zero success for most strengths.

vocabulary ($|C| = 20$) and (ii) a minimal `Question-words` vocabulary ($|Q| = 11$); e.g., *what, where, how*). As a control (RQ4), we also evaluate randomly sampled SAE features from the same layer. Appendix B lists the vocabularies.

5.4 Steering and Evaluation Protocol

We steer a single SAE feature by adding its activation 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 α to characterize sensitivity and also report the best-performing configuration per vocabulary. Appendix D contains evaluation prompts, and Appendix C defines metrics.

6 Results and Discussion

In this section, we present and analyze the main findings on two datasets: AmbiK (single-turn) and ClarQ-LLM (multi-turn).

6.1 RQ1: Clarification Behavior

We first evaluate whether feature-level steering increases the model's propensity to ask clarification questions. On AmbiK, we measure clarification rate (fraction of examples where the model asks at least one question) and task success, which is the fraction marked as resolved by the proxy criterion, $\text{Acc} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\texttt{resolved\_proxy}_i]$ (more details in Appendix C). On ClarQ-LLM, we report success rate, i.e. the fraction of dialogues in which the seeker elicits all gold task details from the provider.

Figure 3 shows that steering can substantially increase question-asking, but the effect depends on the vocabulary used for feature discovery. On Gemma-2-9B-IT, Q-vocabulary steering yields a large gain in clarification rate **(0.61 → 0.95)** and also improves task success **(0.06 → 0.21)**. In contrast, C-vocabulary steering slightly reduces clarification rate for 9B while still improving task success over baseline, suggesting a trade-off between eliciting questions and improving downstream resolution under the proxy metric.

On ClarQ-LLM, Figure 4 indicates that success remains low and is not monotonic in α: Gemma-2B-IT varies modestly across strengths, while Gemma-2-9B-IT stays near zero for most α. This suggests that inducing question-asking alone is insufficient for the strict success criterion on this benchmark, and that the intervention may need more targeted feature sets or different calibration for multi-turn settings.

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). Table 1 shows that the resulting feature sets can differ substantially (e.g., low Jaccard overlap in some settings), and Figure 5 shows that these differences translate into different robustness profiles under α sweeps.

6.3 RQ3: Strength Sensitivity

Steering is sensitive to the intervention strength α (Figure 5). On AmbiK, moderate α can improve

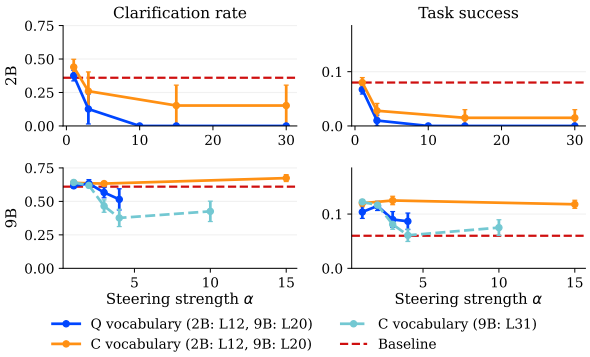

Figure 5: α sweeps comparing Q- vs. C-derived feature discovery on different layers (2B: L12; 9B: Q@L20, C@L20, C@L31) for AmbiK dataset. Dashed red indicates the unsteered baseline and error bars are SEM. Vocabulary choice strongly affects robustness: on 9B, Q features preserve clarification and task success across moderate α, while C features (especially at L31) deteriorate at higher strengths. On 2B, both vocabularies become harmful as α grows, suggesting that effective steering occurs only in a lower-α regime.

Table 1: Overlap of **K=50** ClarifyScore-selected candidates after filtering by OutputScore > 0. Vocabulary choice substantially affects which features survive filtering, but the effect is not uniform across settings. Some conditions show very low agreement between question- and clarification-derived sets (e.g., Jaccard ≈ 0.15), while others show high agreement (e.g., Jaccard ≈ 0.72). This indicates that offline feature discovery can be sensitive to the vocabulary signal in some layers/chunks and relatively robust in others.

M	(L,chunk)	$\lvert F_Q \rvert$	$\lvert F_C \rvert$	$\lvert F_Q \cap F_C \rvert$	Jaccard
9B	(20,0)	12	8	6	0.429
2B	(12,0)	10	13	3	0.150
2B	(12,1)	31	31	26	0.722

task success and/or clarification rate, but larger strengths can saturate or degrade performance, especially for C-derived features (notably at 9B L31). On 2B, both vocabularies become harmful as α increases, indicating that effective steering occurs primarily in a low-α regime.

6.4 RQ4: Random Features

Finally, we compare vocabulary-selected features against a random-feature control: 10 SAE features are uniformly sampled from Gemma-2-9B-IT SAE at layer 20 (IDs: 1250, 1715, 2856, 3529, 8734, 8771, 10885, 11167, 12423, 13344) and steered with the same protocol.

Figure 6 shows that improvements are not explained by generic distribution shift: random-feature steering produces larger degradations at comparable shifts, whereas vocabulary-selected features achieve a better clarification–success trade-off. This supports that the gains are specific to the discovered feature sets rather than an artifact of activating arbitrary directions.

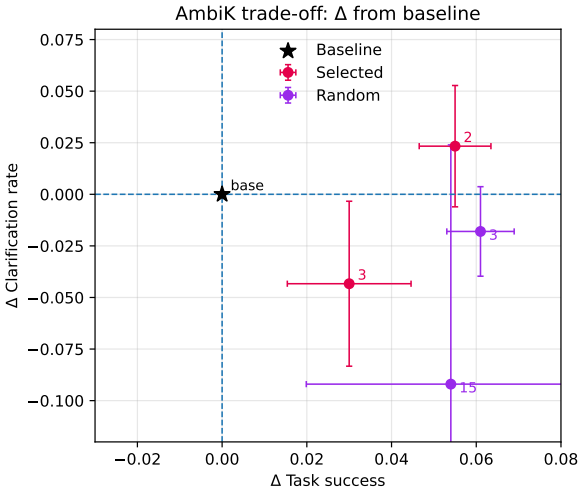

Figure 6: Trade-off between clarification rate and task success on AmbiK, shown as differences from the unsteered baseline. Points correspond to individual runs on 9B model for clarification vocabulary (C) with steering strengths α; error bars are SEM across steered features (selected) or random feature samples (random). Selected features improve clarification without the degradation observed under random steering at comparable distribution shifts.

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* (a vocabulary-driven adaptation of ReasonScore) and Output Score. 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 any additional model training, complementing prompt- or training-based approaches to eliciting question asking.

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 tested our method on two Gemma instruction-tuned models (2B and 9B) because pretrained SAE checkpoints are available for them; training SAEs for additional models or layers was beyond our available compute budget. Extending ClarifySAE to larger models and other architectures as more SAE checkpoints become available is a clear next step.

Dataset coverage. 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). This can be mitigated by adding broader benchmarks and using lightweight human evaluation of question necessity/helpfulness.

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. For simplicity we mostly analyze single-feature interventions; clarification behavior may instead be encoded in interacting or distributed feature patterns. 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*.

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.

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*.

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

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.

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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*.

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. 2025. Automating steering for safe multimodal large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 792–814.

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*.

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*.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, and 1 others. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR.

A Appendix – Reproducibility and Code

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-related 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"
```

C 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

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

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

$$r_i = \mathbb{1}[s_i \geq \tau], \tag{4}$$

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

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

$$s_i^{\mathrm{NLI}} = \max_{q \in Q_i} \mathrm{NLI}(q, g_i), \tag{5}$$

and the corresponding resolution indicator

$$r_i^{\mathrm{NLI}} = \mathbb{1}\left[s_i^{\mathrm{NLI}} \geq \tau_{\mathrm{NLI}}\right], \tag{6}$$

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

$$\mathrm{ResolvedProxyRate} = \frac{1}{N} \sum_{i=1}^{N} r_i, \tag{7}$$

and the NLI-based variant is

$$\mathrm{ResolvedProxyRate}_{\mathrm{NLI}} = \frac{1}{N} \sum_{i=1}^{N} r_i^{\mathrm{NLI}}. \tag{8}$$

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

$$\mathrm{AvgNumQuestions} = \frac{1}{N} \sum_{i=1}^{N} k_i. \tag{9}$$

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:

$$\mathrm{Sim}(c) = \frac{1}{|I_c|} \sum_{i \in I_c} s_i. \tag{10}$$

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

$$\mathrm{Asked}_i = \mathbb{1}[k_i > 0]. \tag{11}$$

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:

$$\mathrm{TP} = |I_{\mathrm{pref}} \cap I_{\mathrm{ask}}|. \tag{12}$$

$$\mathrm{FP} = |(\overline{I_{\mathrm{pref}}}) \cap I_{\mathrm{ask}}|. \tag{13}$$

$$\mathrm{FN} = |I_{\mathrm{pref}} \cap (\overline{I_{\mathrm{ask}}})|. \tag{14}$$

Then

$$\mathrm{NecessityPrecision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{15}$$

$$\mathrm{NecessityRecall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{16}$$

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

$$\mathrm{NecessityScore} = \frac{\mathrm{TP}}{N_{\texttt{pref}}}. \tag{17}$$

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

$$\mathrm{BrevityScore} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}[k_i \leq b]. \tag{18}$$

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

$$\mathrm{OverallSimilarity} = \frac{1}{|I_{ask}|} \sum_{i \in I_{ask}} s_i. \tag{19}$$

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

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

$$\begin{aligned} \mathrm{OverallWeighted} = {} & 0.5\,\mathrm{NecessityScore} \\ & + 0.4\,\mathrm{OverallSimilarity} \\ & + 0.1\,\mathrm{BrevityScore}. \end{aligned} \tag{20}$$

D 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>"
```

E Appendix – Scoring Details

E.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

$$\mu_{j,k} = \frac{1}{|G_k|} \sum_{t \in G_k} z_{t,j}. \tag{21}$$

Normalize the group means into a discrete distribution

$$p_{j,k} = \frac{\mu_{j,k}}{\sum_{k'=1}^{K} \mu_{j,k'}}, \tag{22}$$

and compute normalized entropy

$$H_j = -\frac{1}{\log K} \sum_{k=1}^{K} p_{j,k} \log p_{j,k}, \tag{23}$$

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

$$\mu_j^{\text{pos}} = \sum_{k \in K^{\text{pos}}} \mu_{j,k}, \qquad \mu_j^{\text{neg}} = \sum_{k \in K^{\text{neg}}} \mu_{j,k}. \tag{24}$$

We use the global normalizers

$$S^{\text{pos}} = \sum_{j'} \mu_{j'}^{\text{pos}}, \qquad S^{\text{neg}} = \sum_{j'} \mu_{j'}^{\text{neg}}. \tag{25}$$

E.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)

$$l_i = f_i^\top W_U \in \mathbb{R}^{|V|}, \tag{26}$$

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

$$z_i' = z_i + \kappa\, z_{\max,i}, \tag{27}$$

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

$$p_i^\star = p_i(v_i^\star) \tag{28}$$

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

F Appendix – Examples

We provide qualitative context for the steering features used in the main experiments. Table 3 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 2 shows paired generations for the same input under baseline vs. steering settings, highlighting how steering changes question-asking behavior. Table 4 reports the manually selected features used in experiments, including their OutputScore values and top-decoded tokens for interpretability.

Table 2: 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 3: 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 4: Manually selected features used in experiments and their corresponding Output scores 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", ".."]