DOES REASONING ALWAYS HELP HALLUCINATION DETECTION? A LENS THROUGH STEP SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large reasoning models (LRMs) generate long reasoning traces before producing final answers. While these traces may contain useful signals for hallucination detection, harnessing them is non-trivial because long trajectories often include noisy steps that obscure the cues relevant to truthfulness assessment. In this paper, we identify two prevalent forms of reasoning noises, i.e., irrelevant steps and repetitive steps, and show that both substantially degrade hallucination detection performance. Existing confidence-based scores and naive embedding-based filtering fail to reliably separate noisy from informative steps. To address this challenge, we propose REDE, a novel learning framework for denoising reasoning traces for hallucination detection. Specifically, REDE leverages final-answer attention as an automatic supervision signal to shape the step-level representation space, yielding refined embeddings in which noisy steps can be reliably identified and filtered. REDE can be readily plugged into diverse hallucination detectors by operating on the filtered reasoning trajectory after removing noisy steps. Extensive experiments on multiple reasoning benchmarks show that REDE consistently improves detection performance over competitive baselines.

1 INTRODUCTION

Hallucination detection for large reasoning models (LRMs) is critical for building reliable AI systems. Recent LRMs, such as OpenAI's o series (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025), generate long reasoning traces before producing final answers. While these traces can improve reasoning capability, hallucination remains a serious reliability concern (Lu et al., 2025; Bao et al., 2025; Guo et al., 2025; Hou et al., 2025b). Existing studies have begun to leverage reasoning traces for hallucination detection (Sun et al., 2026; Lu et al., 2026), motivated by the intuition that intermediate reasoning may reveal useful signals about the final answer correctness. However, harnessing reasoning traces is non-trivial. Recent evidence shows that, although long reasoning traces can improve reasoning performance, they can also obscure the cues useful for hallucination detection (Cheng et al., 2025). This suggests that naively incorporating the full reasoning trace does not consistently benefit detection.

A key challenge is that long reasoning traces often contain *noisy steps* that are weakly informative for hallucination detection. By examining LRM trajectories, we identify two prevalent forms of noisy steps: *irrelevant steps*, which do not contribute meaningful problem-specific information, and *repetitive steps*, whose information is already covered by later steps in the trajectory. As shown in Figure 1(a), these noisy steps can substantially degrade hallucination detection performance, while removing them yields clear improvements. This observation suggests that *identifying and filtering noisy reasoning steps is crucial* for effectively using reasoning traces in hallucination detection.

A natural approach is to identify noisy steps using confidence-based signals. Recent work has used confidence for improving reasoning-time generation (Varshney et al., 2023; Zubkova et al., 2025; Santosh et al., 2025; Tang et al., 2026). However, these methods are designed to improve generation quality, rather than to identify which reasoning steps are informative for detecting hallucination in the answer. Confidence reflects how certain the model is about what it generates, not how useful a step is for downstream hallucination detection. Indeed, Figure 1(b) shows the confidence distributions of informative and noisy steps overlap heavily, making confidence-based filtering ineffective.

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

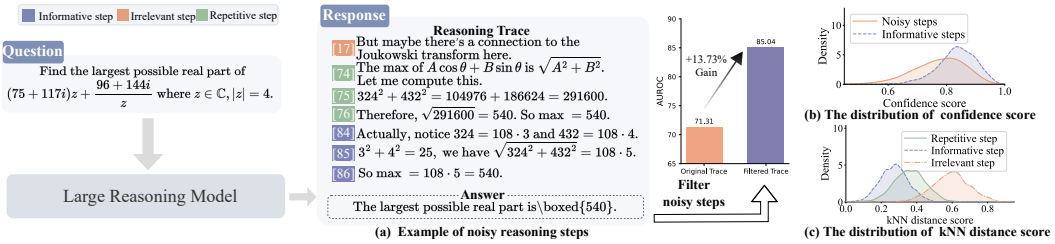


Figure 1: (a) Long reasoning traces contain *irrelevant* and *repetitive* steps, which hurt hallucination detection; removing them yields clear improvements, as validated with supervised probing (Azaria & Mitchell, 2023) using the last-layer embeddings. (b) Confidence-based signals fail to distinguish informative and noisy steps, as their score distributions overlap heavily. (c) Naive kNN-based on raw step embeddings can identify irrelevant steps but fails to separate repetitive from informative ones. Experiments are based on AIME 2024 (Hugging Face H4, 2025) with Qwen3-8B (Yang et al., 2025). We annotate the reasoning steps into informative, irrelevant, and repetitive categories. Annotation details are in Appendix A.3.

Another possible route is to identify noisy steps in the embedding space. While irrelevant steps may behave as outliers and can sometimes be detected by simple representation-similarity methods such as kNN, this strategy remains inadequate for repetitive steps. Unlike irrelevant steps, repetitive steps are often semantically similar to informative ones and therefore remain close in the raw representation space. As shown in Figure 1(c), naive kNN-based filtering can identify some irrelevant steps but fails to distinguish repetitive from informative ones. These limitations suggest that effective noisy-step filtering requires a signal beyond confidence or raw semantic similarity—one that better reflects each step's contribution to the final answer and to hallucination detection.

To address this challenge, we propose **Reasoning Denoiser (REDE)**, a novel framework for filtering noisy reasoning steps by shaping the step-level representation space. Our key idea is to use the attention score between the final answer token and each reasoning step as a supervision signal, which requires no human annotation. Empirically, this attention is associated with step usefulness: *irrelevant steps* tend to receive low attention because they are weakly connected to the answer, while *repetitive steps* can also receive low attention because their information has already been covered by later steps in the reasoning trajectory. Based on this observation, REDE learns a lightweight projection that pulls informative steps closer together while pushing noisy steps away. The resulting refined step embeddings allow noisy steps to be reliably identified and filtered by simple distance-based methods such as kNN.

REDE can be readily plugged into diverse hallucination detection methods by operating on the filtered reasoning trajectory after removing noisy steps, including probing-based (Azaria & Mitchell, 2023; Burns et al., 2023), uncertainty-based (Ren et al., 2023), and verbalized methods (Lin et al., 2022a). Extensive experiments on four representative reasoning tasks show that REDE consistently improves hallucination detection performance across diverse datasets. Specifically, on a representative benchmark TruthfulQA, REDE improves detection performance by up to 18.69% over using the original unfiltered reasoning trace, and achieves state-of-the-art performance of 87.32%, demonstrating that reasoning-step denoising is an effective and general strategy for hallucination detection in LRMs.

Our key contributions are summarized as follows:

- We identify *noisy reasoning steps* as an overlooked obstacle to hallucination detection. We further show that these noisy steps degrade hallucination detection performance, establishing reasoning-step filtering as an important solution for reliable hallucination detection.
- We propose Reasoning Denoiser (REDE), a novel learning framework that leverages final-answer attention to shape the step-level representation space, enabling reliable filtering of noisy reasoning steps without human annotation.
- Extensive experiments (Section 5.2 and Appendix B) evaluate the effectiveness of REDE through comparisons across reasoning tasks, controlled evaluations, and ablations of its design choices. We also demonstrate that REDE can be plugged into diverse hallucination detection methods for consistent gains.

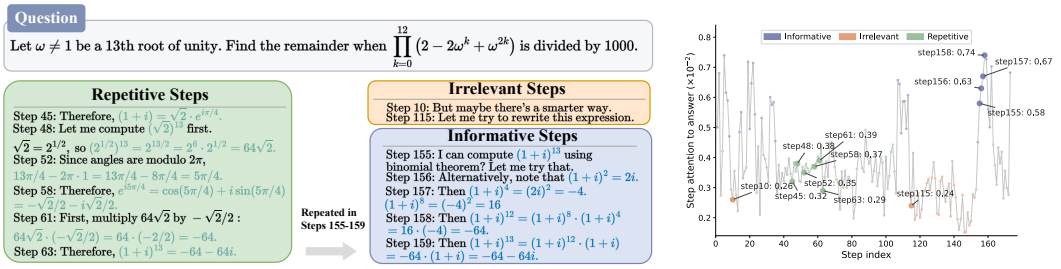


Figure 2: Attention scores from the final-answer token to each reasoning step on a representative AIME 2024 (Hugging Face H4, 2025) example with Qwen3-8B (Yang et al., 2025). Irrelevant and repetitive steps receive low attention, while informative steps receive high attention. The computation of attention scores is detailed in Section 4.1.

2 Problem Setup

We study hallucination detection for large reasoning models that generate reasoning traces before final answers, which have become increasingly prominent in today's foundation model landscape.

Reasoning generation. Let an L-layer causal LRM take a prompt $\mathbf{p}$ and generate a response autoregressively. In reasoning-centric settings, we decompose the generated response into a reasoning trace $\mathbf{C} = (\mathbf{c}_1, \ldots, \mathbf{c}_K)$ and a final answer $\mathbf{a}$, where each $\mathbf{c}_k$ is a contiguous token span corresponding to one reasoning step. Both the number of reasoning steps K and the length of each step can vary across inputs.

Hallucination detection. Given $(\mathbf{p}, \mathbf{C}, \mathbf{a})$, the goal is to determine whether the final answer $\mathbf{a}$ is truthful under the task-specific criterion. Let $y \in \{0, 1\}$ denote the label, where $y = 0$ means the answer is truthful and $y = 1$ means it is hallucinated. We aim to learn a binary detector $G(\mathbf{p}, \mathbf{C}, \mathbf{a}) \in \{0, 1\}$, which predicts the truthfulness of the final answer (Lu et al., 2026; Sun et al., 2026; Wang et al., 2026; Zhang et al., 2026).

Challenge of long reasoning traces. Unlike standard generation, the reasoning trace in an LRM is not uniformly useful for hallucination detection. Long traces often contain noisy steps that are weakly informative about the correctness of the final answer, thereby diluting the signals most relevant to truthfulness assessment (Cheng et al., 2025). Consequently, naively using the original reasoning trace may hurt detection performance. This motivates our goal of denoising the reasoning trace before hallucination detection. We provide analysis on the noisy reasoning steps next.

3 Understanding Noisy Reasoning Steps for Hallucination Detection

Before introducing our method, we first seek to better understand the nature of noisy reasoning steps and to identify a suitable signal for detecting them. Our analysis leads to two main observations. First, the final-answer attention in LRMs provides a meaningful signal for distinguishing informative steps from noisy ones. Second, among semantically similar steps, earlier steps are more likely to be redundant, whereas later steps tend to preserve the information most relevant to the final answer. These observations motivate the design of our framework in Section 4.

Final-answer attention reflects step informativeness. We examine whether the LRM's own internal signals can reveal which reasoning steps are useful for hallucination detection. Our key observation is that the attention score from the final-answer token to each reasoning step provides a meaningful relevance signal: steps that contribute more directly to the final answer tend to receive higher attention.

We verify this on our annotated AIME 2024 (Hugging Face H4, 2025) (*cf.* Appendix A.3) using Qwen3-8B (Yang et al., 2025), where we compute the attention score assigned by the final-answer token to each reasoning step (formally defined in Eq. 1). Figure 2 shows a representative example. Three patterns emerge clearly: (1) *irrelevant steps* receive low attention; (2) *repetitive steps*, whose content is effectively subsumed by later steps, also receive low attention; and (3) *informative steps* that directly support the final answer receive substantially higher attention. This association also holds at the dataset level. Averaged over all examples, informative steps receive a mean attention

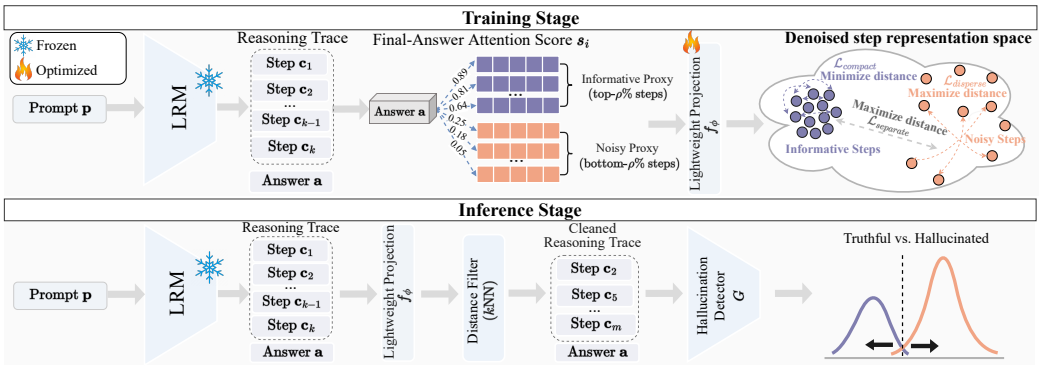


Figure 4: **Overview of the REDE framework for hallucination detection in LRMs.** During training, REDE scores each reasoning step via final-answer attention and learns a lightweight projection f_ϕ that maps step embeddings into a denoised representation space where informative and noisy steps are well separated. During inference, step embeddings are projected by f_ϕ and noisy steps are filtered via distance-based scoring for downstream hallucination detection.

score of 6.5×10^{-3}, compared to 2.1×10^{-3} for irrelevant steps and 3.3×10^{-3} for repetitive steps. These results support final-answer attention as an annotation-free proxy for step usefulness, without establishing a causal attribution of individual steps. We further validate the effectiveness of this signal in Appendix C.1.

Earlier semantically similar steps are often redundant. Our analysis shows low-attention steps are not homogeneous: some are irrelevant, while others are part of repeated reasoning. We next focus on the latter case. When multiple semantically similar steps appear in a reasoning trajectory, an important question is which occurrence preserves the information most relevant to the answer. If later steps consistently retain the essential information better than earlier ones, this suggests earlier occurrences are redundant. To study this, we cluster reasoning steps based on embedding similarity and, from each cluster, retain either the earliest or the latest step; details are in Appendix A.6.

For each retained subset, we concatenate the selected steps with the original question and answer, and then extract the resulting answer embedding from the LRM. We evaluate two criteria: (1) the cosine similarity between the new answer embedding and the original full-trace answer embedding, which measures how well the retained steps preserve the original reasoning signal; and (2) the hallucination detection performance obtained by applying a linear probe to the new answer embedding. Figure 3 shows retaining the latest steps consistently yields higher cosine similarity and stronger detection performance than retaining the earliest steps. This result suggests that *earlier steps are often redundant*, whereas later steps might preserve the information relevant for hallucination detection.

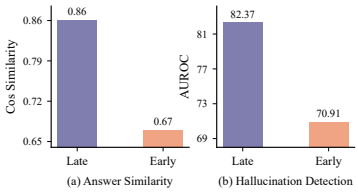


Figure 3: (a) Cosine similarity between the answer embedding from the retained earliest/latest steps and that from the full reasoning trace. (b) Hallucination detection performance of linear probing based on the retained earliest/latest steps. Results on AIME 2024 (Hugging Face H4, 2025) with Qwen3-8B (Yang et al., 2025).

4 METHOD

Our novel framework REDE addresses the challenge of denoising reasoning traces for hallucination detection. The key idea is to learn a step-level representation space in which informative reasoning steps form a compact region, while noisy steps become isolated and hence easy to filter. Rather than using the raw reasoning trace directly, REDE first identifies an annotation-free supervision signal from the LRM itself, then learns a lightweight projection that reshapes the step embeddings, and finally performs filtering in the learned space before applying a downstream hallucination detector. Our framework is illustrated in Figure 4.

Framework overview. Given a prompt $\mathbf{p}$, a reasoning trace $\mathbf{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$, and a final answer $\mathbf{a}$ generated by an LRM, our framework encompasses three components addressing the following

questions: (1) *How can we obtain a reliable supervision signal without human annotation?* We use final-answer attention as a proxy for reasoning-step usefulness, yielding an annotation-free step score (Section 4.1). (2) *How can we learn a representation space favorable for noisy-step filtering?* Using the attention scores as supervision, we train a projection network that maps step embeddings into a new space where informative and noisy steps are more separable (Section 4.2). (3) *How can we perform efficient filtering at test time?* We filter noisy steps in the shaped space and feed the resulting trace to a downstream hallucination detector (Section 4.3). The training and inference procedures are summarized in Algorithms 2 and 3.

4.1 Scoring reasoning steps with final-answer attention

REDE uses final-answer attention as a proxy for reasoning-step usefulness. As shown in Section 3, the internal attention of LRMs provides a natural signal for this purpose.

Let a_T denote the last token of answer $\mathbf{a}$, $\mathbf{x}_{1:N} = (x_1, \ldots, x_N)$ denote the full causal context visible to a_T, and $I_i \subseteq \{1, \ldots, N\}$ denote the set of token indices corresponding to the i-th reasoning step $\mathbf{c}_i$. We define the score of step $\mathbf{c}_i$ as the total attention mass assigned by a_T to the tokens in that step:

$$s_i = \sum_{t \in I_i} \frac{\exp\big(\mathbf{q}(a_T)^\top \mathbf{k}(x_t)/\sqrt{\omega}\big)}{\sum_{u=1}^{N} \exp(\mathbf{q}(a_T)^\top \mathbf{k}(x_u)/\sqrt{\omega})}, \tag{1}$$

where $\mathbf{q}(\cdot)$ and $\mathbf{k}(\cdot)$ are the query and key projections, and ω is the query/key projection dimension.

The score s_i measures how strongly the final answer attends to step $\mathbf{c}_i$. The patterns in Section 3 motivate its use as a coarse supervision signal. If a step is *irrelevant* to the answer, its key representations are weakly aligned with the query of the answer token, resulting in a low score. If a step is *repetitive*, its content is often subsumed by later steps, and the attention tends to concentrate on the later occurrence, again leading to a low score. Informative steps tend to receive higher scores. These are statistical tendencies used to construct proxy sets, rather than causal attributions or ground-truth step labels.

From attention scores to representation shaping. While directly thresholding the attention score s_i at test time and discarding low-attention steps is feasible, this approach has two important limitations. First, it requires recomputing attention scores for every test sample, which incurs nontrivial inference overhead. Second, although attention provides a meaningful signal when aggregated across many examples, per-instance attention scores can vary with reasoning trace length and structure, and prior work has shown that vanilla attention may not reliably transfer to downstream tasks without further learning (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). We verify this empirically in Section 5.2, where direct attention-based filtering consistently underperforms our learning-based approach.

Instead of using attention as a direct filter, we use it as an annotation-free supervision signal to shape the step-level representation space. By learning a lightweight projection over many examples, REDE converts the per-instance attention signal into a more stable embedding structure that reflects step informativeness. We introduce this process next.

4.2 Learning a denoised step representation space

We now describe how REDE extracts step embeddings and learns a shaped representation space for noisy-step filtering.

Step embedding extraction. We run the frozen LRM on the prompt, reasoning trace, and final answer to obtain contextual token representations. For each step $\mathbf{c}_i$ spanning token indices I_i, we aggregate the hidden states $\{\mathbf{h}(x_t)\}_{t \in I_i}$ into a step embedding $\mathbf{e}_i \in \mathbb{R}^d$. We use the same extraction procedure during training and inference; implementation details are provided in Appendix A.5.

Selecting informative and noisy proxies. For each training trace, we rank the steps by the attention scores $\{s_i\}_{i=1}^{K}$ (Eq. 1). We denote by $\mathcal{T}$ the top-$\rho\%$ steps and by $\mathcal{B}$ the bottom-$\rho\%$ steps. The set $\mathcal{T}$ serves as a proxy set of informative steps, whereas $\mathcal{B}$ serves as a proxy set of noisy steps (the effect of ρ is studied in Appendix B.6).

Table 1: **Comparison with competitive hallucination detection methods.** "Single Generation" indicates whether the method requires only one generation during inference. "Training labels" indicates whether the method uses ground-truth annotations for training. Both REDE configurations use 100 labeled validation examples for hyperparameter selection (Appendix A.4). We report the results of REDE with CCS and supervised probing as downstream detectors. The best results are highlighted in **bold**.

Method	Single Generation	Training labels	Qwen3-8B				DeepSeek-R1-Distill-Llama-8B			
			TruthfulQA	MATH	CodeElo	MULTIHOPQA	TruthfulQA	MATH	CodeElo	MULTIHOPQA
Lexical Similarity (Lin et al., 2024)	✗	✗	52.00	49.02	55.44	58.26	44.09	66.28	52.57	54.33
SelfCheckGPT (Manakul et al., 2023)	✗	✗	53.11	47.20	51.85	52.61	58.54	51.81	63.33	50.21
Semantic Entropy (Kuhn et al., 2023)	✗	✗	54.35	40.69	53.76	66.83	49.43	51.11	72.52	50.13
EigenScore (Chen et al., 2024a)	✗	✗	49.50	55.31	53.88	60.27	45.31	51.17	55.20	58.12
P(True) (Kadavath et al., 2022)	✓	✗	64.47	75.16	55.62	75.59	42.93	45.64	53.88	65.25
HaloScope (Du et al., 2024)	✓	✗	67.22	61.43	57.19	61.03	54.19	64.18	54.22	51.31
TSV (Park et al., 2025)	✓	✓	69.91	71.88	59.42	67.43	52.59	68.75	59.28	59.71
CED (Lee et al., 2024)	✓	✗	70.29	68.42	60.23	72.18	54.51	65.30	61.72	57.32
HaMI (Niu et al., 2025)	✓	✓	71.42	70.05	60.44	70.63	58.72	67.41	60.57	61.74
LRM-based										
RHD (Sun et al., 2026)	✓	✗	58.08	64.15	62.21	58.16	57.16	57.63	65.20	63.02
RACE (Wang et al., 2026)	✗	✗	80.82	70.55	77.14	74.58	61.23	62.62	59.02	64.62
HalluGuard (Zeng et al., 2026)	✓	✗	51.67	53.58	56.52	55.73	51.15	57.57	51.86	53.55
ARS (CCS) (Zhang et al., 2026)	✓	✗	81.92	77.03	80.05	70.31	75.42	72.18	73.97	68.40
REDE (CCS)	✓	✗	**87.32**$^{\pm 0.83}$	81.33$^{\pm 2.14}$	82.17$^{\pm 1.76}$	73.41$^{\pm 2.53}$	**78.30**$^{\pm 1.91}$	74.24$^{\pm 2.37}$	79.09$^{\pm 0.62}$	72.85$^{\pm 2.85}$
REDE (Probing)	✓	✓	86.44$^{\pm 0.47}$	**87.06**$^{\pm 1.23}$	**87.19**$^{\pm 0.95}$	**82.94**$^{\pm 1.67}$	74.37$^{\pm 2.08}$	**83.91**$^{\pm 0.71}$	**82.55**$^{\pm 1.42}$	**78.71**$^{\pm 1.89}$

Projection learning. We train a lightweight projection network $f_\phi : \mathbb{R}^d \to \mathbb{R}^{d'}$ to map each step embedding $\mathbf{e}_i$ to a projected embedding $\mathbf{z}_i = f_\phi(\mathbf{e}_i)$. Throughout, the underlying LRM is frozen and only the projection parameters ϕ are optimized.

The goal of the projection is to induce an embedding distribution favorable for distance-based filtering: informative steps should form a compact cluster, noisy steps should remain scattered rather than collapse together, and the two groups should be well separated. To realize this, we define the cosine similarity between projected steps i and j as $\mathrm{sim}_{ij} = \mathbf{z}_i^\top \mathbf{z}_j / (\|\mathbf{z}_i\|_2 \|\mathbf{z}_j\|_2)$, and optimize the following objective:

$$\mathcal{L} = \underbrace{\mathbb{E}_{i \neq j,\, i,j \in \mathcal{T}}\,(1 - \mathrm{sim}_{ij})}_{\mathcal{L}_{\text{compact}}} + \lambda_{\text{disperse}} \cdot \underbrace{\mathbb{E}_{i \neq j,\, i,j \in \mathcal{B}}\,\mathrm{sim}_{ij}}_{\mathcal{L}_{\text{disperse}}} + \lambda_{\text{separate}} \cdot \underbrace{\mathbb{E}_{i \in \mathcal{T},\, j \in \mathcal{B}}\,\mathrm{sim}_{ij}}_{\mathcal{L}_{\text{separate}}}. \tag{2}$$

Each term plays a distinct role. $\mathcal{L}_{\text{compact}}$ pulls informative steps together, forming a compact region. $\mathcal{L}_{\text{disperse}}$ discourages noisy steps from clustering with one another, so that they remain individually identifiable as outliers. $\mathcal{L}_{\text{separate}}$ pushes informative and noisy steps apart, improving their global separability. Overall, these three terms shape a representation space in which noisy steps have large distances to the informative region and can therefore be filtered effectively using simple non-parametric rules (sensitivity to loss weights is analyzed in Appendix B.8).

Discussion. Our design is intentionally lightweight: the projection network operates on frozen step representations and requires only annotation-free supervision from the LRM itself. The filtered trace can be passed to diverse downstream hallucination detectors. The compactness and dispersion objectives are learning targets for the projected space, rather than properties required of the original embeddings. We examine the learned step representations and the contribution of each loss across tasks and model families in Appendix B.3.

4.3 INFERENCE-TIME FILTERING AND HALLUCINATION DETECTION

During inference, REDE performs step filtering without requiring either attention scores or ground-truth labels. Given a test trace $\bar{\mathbf{C}} = \{\bar{\mathbf{c}}_1, \ldots, \bar{\mathbf{c}}_{\bar{K}}\}$, we first extract the step embeddings $\{\bar{\mathbf{e}}_i\}_{i=1}^{\bar{K}}$ as described in Section 4.2 and project them into the learned space: $\bar{\mathbf{z}}_i = f_\phi(\bar{\mathbf{e}}_i)$. We then compute a distance score S_i for each step as its k-nearest neighbor cosine distance with respect to all other steps within the same trace: $S_i = 1 - \bar{\mathbf{z}}_i^\top \bar{\mathbf{z}}_i^{(k)} / (\|\bar{\mathbf{z}}_i\|_2 \|\bar{\mathbf{z}}_i^{(k)}\|_2)$, where $\bar{\mathbf{z}}_i^{(k)}$ denotes the k-th nearest neighbor of $\bar{\mathbf{z}}_i$ among $\{\bar{\mathbf{z}}_j\}_{j \neq i}$. Intuitively, informative steps cluster together in the shaped space and thus have small k-NN distances, while noisy steps are scattered and yield larger distances. We remove the top-$\zeta\%$ steps with the largest distance scores and retain the remaining steps as the filtered trace $\bar{\mathbf{C}}^{\text{fil}} \subseteq \bar{\mathbf{C}}$. The filtered trace is passed to a downstream hallucination detector G, which outputs $\hat{y} = G(\bar{\mathbf{p}}, \bar{\mathbf{C}}^{\text{fil}}, \bar{\mathbf{a}})$. Importantly, REDE is an agnostic denoising module for diverse downstream detection pipelines.

Table 2: **Comparison of detecting hallucination using the original reasoning trace vs. our filtered one.** All values are percentages (AUROC) on different detection methods. The best results are highlighted in **bold**.

Model	Dataset	CCS (Burns et al., 2023)		Supervised Probing (Azaria & Mitchell, 2023)		Perplexity (Ren et al., 2023)		Verbalized Certainty (Lin et al., 2022a)	
		Original	Filtered	Original	Filtered	Original	Filtered	Original	Filtered
Qwen3-8B	TruthfulQA	68.63	**87.32**	80.42	**86.44**	58.17	**63.46**	50.37	**61.10**
	MATH	63.27	**81.33**	81.05	**87.06**	65.16	**71.03**	57.19	**65.13**
	CodeElo	61.64	**82.17**	82.82	**87.19**	61.78	**72.03**	78.89	**83.89**
	MULTIHOPQA	58.19	**73.41**	77.84	**82.94**	52.79	**65.84**	63.75	**69.50**
DeepSeek-R1-Distill-Llama-8B	TruthfulQA	54.14	**78.30**	70.02	**74.37**	52.79	**67.72**	56.61	**61.94**
	MATH	63.62	**74.24**	77.19	**83.91**	65.85	**69.14**	69.71	**76.13**
	CodeElo	55.16	**79.09**	75.07	**82.55**	69.35	**78.49**	79.13	**84.49**
	MULTIHOPQA	60.77	**72.85**	70.05	**78.71**	55.13	**64.45**	50.87	**59.56**

5 EXPERIMENTS

Datasets and Models. We evaluate REDE on four reasoning benchmarks: TruthfulQA (Lin et al., 2022b), MATH, constructed by combining MATH500 (Hendrycks et al., 2021), AIME 2024 (Hugging Face H4, 2025), and AIME 2025 (OpenCompass Team, 2025), CODEELO (Quan et al., 2025), and MULTIHOPQA, which is built following prior work (Sun et al., 2026) by sampling from HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). These datasets cover open-domain question answering, mathematical reasoning, code generation, and multi-hop question answering, respectively. For all datasets, we use 25% of the available data for testing, reserve 100 examples for validation, and use the remaining data for training. We evaluate all methods using AUROC, and all reported results are averaged over three random seeds.

We evaluate our method on two widely used LRM families: Qwen3-8B/32B (Yang et al., 2025) and DeepSeek-R1-Distill-Llama-8B/DeepSeek-R1-Distill-Qwen-32B (Guo et al., 2025). Correctness labels are obtained using a strong external judge model Qwen3-32B following prior work (Yao et al., 2025) (robustness to alternative labeling methods is verified in Appendix C.6). By default, model outputs are generated with greedy decoding. Additional prompt, dataset, sampling strategies, and implementation details are provided in Appendices A.1, A.2, C.5, and A.4, respectively. We also report qualitative case studies, results on GSM8K (Cobbe et al., 2021), the impact of filtering on task accuracy, and additional evaluation metrics in Appendices C.4, C.7, C.9, and C.10, respectively.

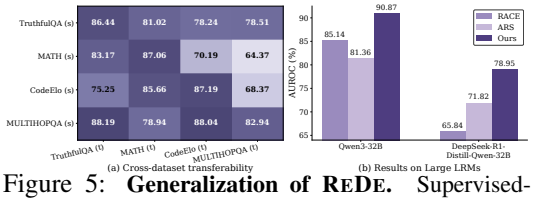
Baselines. We first compare detection performance using the original unfiltered reasoning trace and our filtered one under four representative hallucination detectors: probing-based detector (Supervised Probing (Azaria & Mitchell, 2023) and Contrast-Consistent Search, CCS (Burns et al., 2023)), uncertainty-based detector (Perplexity (Ren et al., 2023)), and verbalized detector (Verbalized Certainty (Lin et al., 2022a)). We then compare our method with a comprehensive set of baselines, including: (1) embedding-based methods: Truthfulness Separator Vector (TSV) (Park et al., 2025), HaloScope (Du et al., 2024), CED (Lee et al., 2024), and HaMI (Niu et al., 2025); (2) consistency-based methods: Lexical Similarity (Lin et al., 2024), SelfCheckGPT (Manakul et al., 2023), Semantic Entropy (Kuhn et al., 2023) and EigenScore (Chen et al., 2024a); and (3) verbalized methods: P(True) (Kadavath et al., 2022). We further include methods specifically designed for LRMs, including Reasoning and Answer Consistency Evaluation (RACE) (Wang et al., 2026), Reasoning Hallucination Detection (RHD) (Sun et al., 2026), HalluGuard (Zeng et al., 2026), and Answer-agreement Representation Shaping (ARS) (Zhang et al., 2026). For fair comparison, all baselines are evaluated on the same test sets under the default settings reported in their original papers. For each filtering strategy, trainable downstream detectors are trained on the corresponding traces using the same training questions, splits, and detector hyperparameters. Projection learning uses no human step annotations or correctness labels; hyperparameters are selected using AUROC on the 100 labeled validation examples. Implementation details are provided in Appendix A.4.

5.1 MAIN RESULTS

Effect of reasoning step filtering. Table 2 compares hallucination detection using the original unfiltered reasoning trace versus our filtered trace across four representative detectors. Across all datasets and two LRMs, our step selection consistently yields substantial improvements, indicating that the benefit stems from filtering noisy steps rather than from any particular downstream detector.

For example, on Qwen3-8B over TruthfulQA, AUROC improves from 68.63 to 87.32 for CCS, and from 80.42 to 86.44 for supervised probing. Similar trends hold on DeepSeek-R1-Distill-Llama-8B. A frozen-detector control also improves on all four datasets (Appendix B.2), showing that gains persist when the detector is not retrained.

Comparison with baselines. Table 1 compares REDE with embedding-based, consistency-based, verbalized, and LRM-specific detectors. REDE achieves the highest AUROC in nearly all settings. On Qwen3-8B, supervised probing achieves 87.06% on MATH, 87.19% on CodeElo, and 82.94% on MULTIHOPQA. Unsupervised CCS achieves 87.32% on TruthfulQA, outperforming other unsupervised baselines. It also exceeds the representation-shaping baseline ARS (Zhang et al., 2026), including 81.33% versus 77.03% on MATH with CCS. These results support step-level denoising across detectors and tasks; Appendix A.8 analyzes efficiency.



(a) Cross-dataset transferability (b) Results on Large LRMs

Figure 5: **Generalization of REDE.** Supervised-probing AUROC (%): (a) Qwen3-8B transfer (rows: source; columns: target); (b) 32B LRMs on TruthfulQA.

Cross-dataset transfer. Following prior work (Park et al., 2025), we train the projection f_ϕ on a source dataset and apply it to a different target dataset without retraining the projection. Figure 5(a) reports results with Qwen3-8B and supervised probing. Training on MULTIHOPQA and testing on CodeElo yields 88.04% AUROC, slightly exceeding the in-domain result of 87.19%. The projection transfers across diverse reasoning tasks, suggesting that the learned denoising signal extends beyond the training distribution. The filter thus remains effective without dataset-specific projection training.

Scaling to larger LRMs. Figure 5(b) evaluates REDE on 32B LRMs using TruthfulQA and supervised probing. REDE outperforms both RACE and ARS: it achieves 90.87% AUROC on Qwen3-32B and 78.95% on DeepSeek-R1-Distill-Qwen-32B, exceeding RACE by 5.73 and 13.11 percentage points, respectively. These gains show that step-level denoising remains effective at larger model scales. Appendix C.3 reports results on additional benchmarks.

5.2 ANALYSIS

Ablation on selection strategies. Figure 6 compares REDE with random selection, four existing selectors (Liang et al., 2026; Cui et al., 2025; Zeng et al., 2025; Li et al., 2026), and Qwen3-32B as an LLM-as-Judge (Yang et al., 2025). Original keeps all reasoning steps; Remove all reasoning keeps only the question and original final answer. REDE achieves the highest CCS AUROC. Random selection and removing all reasoning (64.01%) underperform Original (68.63%), indicating that useful reasoning should be retained. Existing selectors target generation quality or efficiency; REDE learns to identify noisy steps for detection. LLM-as-Judge also gives smaller gains. Rankings vary by detector: Step Entropy achieves higher probing AUROC on TruthfulQA. Appendix B.1 reports other detectors and three additional datasets; Appendix B.1.1 distinguishes significant gains from statistically comparable results.

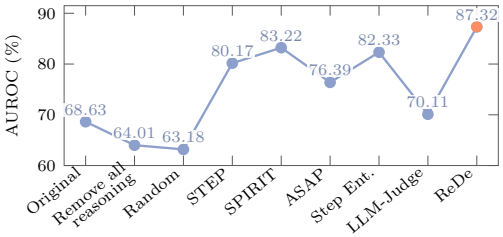


Figure 6: **Step-selection strategies.** CCS AUROC (%) on TruthfulQA with Qwen3-8B.

Role of attention supervision. On TruthfulQA, direct attention filtering underperforms REDE across all four detectors (Table 3), including 80.51% versus 87.32% AUROC with CCS. Learning from attention improves filtering without requiring attention computation at inference; Appendix C.1 discusses this distinction.

Table 3: Direct attention vs. REDE.

Detector	Attention	Ours
CCS	80.51	**87.32**
Supervised Probing	84.97	**86.44**
Perplexity	60.11	**63.46**
Verbalized Certainty	59.73	**61.10**

Effect of individual loss components. Removing any term from Equation 2 lowers supervised-probing AUROC on TruthfulQA with Qwen3-8B. Removing $\mathcal{L}_{\text{disperse}}$ causes the largest drop, from 86.44% to 80.06%. Appendix B.3 reports the full ablation, t-SNE visualizations, and comparisons across tasks and model families.

Denoising with a frozen detector. To separate denoising gains from detector adaptation, we train a probing detector on original traces and hold its parameters fixed. We change only the test-time trace, keeping the question and original final answer unchanged. With Qwen3-8B, REDE improves AUROC by 3.05, 3.36, 2.24, and 2.78 points on TruthfulQA, MATH, CodeElo, and MULTIHOPQA, respectively (Table 10, Appendix B.2). These gains across four tasks show that removing noisy steps improves the detector's input even without retraining it on filtered traces.

Additional analysis. We provide further analyses of (1) selection strategies and statistical significance (Appendix B.1); (2) representations and loss components (Appendix B.3); (3) representation and inference choices (Appendices B.4–B.11); (4) attention supervision (Appendix C.1); (5) human-annotated selection accuracy (Appendix C.2); (6) larger LRMs (Appendix C.3); (7) qualitative examples (Appendix C.4); (8) sampling strategies (Appendix C.5); (9) truthfulness labeling (Appendix C.6); (10) evaluation on GSM8K (Appendix C.7); (11) black-box applicability (Appendix C.8); (12) task accuracy (Appendix C.9); and (13) additional detection metrics (Appendix C.10).

6 RELATED WORK

Hallucination detection has attracted broad interest across multiple paradigms, including logit- and probability-based uncertainty (Kadavath et al., 2022; Ren et al., 2023; Varshney et al., 2023), consistency-based scoring (Burns et al., 2023; Kuhn et al., 2023; Lin et al., 2024; Manakul et al., 2023), verbalized confidence (Lin et al., 2022a), embedding-based approaches (Azaria & Mitchell, 2023; Chen et al., 2024a; Du et al., 2024; Park et al., 2025; Hu et al., 2026; Bhatnagar et al., 2026; Niu et al., 2025), and methods that reveal mechanistic distinctions between factual and reasoning-driven hallucinations (Luo et al., 2026). While effective for standard LLMs, these methods are not designed for LRMs, whose long reasoning traces introduce additional complexity. Empirical studies show that the reasoning trace actively obscures hallucination signals (Yao et al., 2025; Cheng et al., 2025). Recent LRM-specific detectors exploit reasoning traces more directly (Lu et al., 2025; 2026; Sun et al., 2026; Srey et al., 2025; Wang et al., 2026; Zeng et al., 2026), but none address the noise steps embedded within the trace itself. The most relevant work is ARS (Zhang et al., 2026), which shapes answer-level representations via latent perturbations at the trace boundary but treats the reasoning trace as a monolithic context, without considering the reasoning noise steps. REDE fills this gap by filtering noisy steps before hallucination detection.

Reasoning step selection has been explored recently to select reasoning steps for better generation quality and efficiency (Xia et al., 2025; Wu et al., 2025; Hou et al., 2025a; Luo et al., 2025; Hong et al., 2025; Tu et al., 2025; Fu et al., 2025; Do et al., 2025; Hong et al., 2026). Representative approaches include measuring perplexity shifts upon step removal (SPIRIT (Cui et al., 2025)), using first-token surprisal to identify logical pivots (ASAP (Zeng et al., 2025)), compressing traces via step-level entropy (Step Entropy (Li et al., 2026)), training hidden-state probes to evaluate step contributions (STEP (Liang et al., 2026)), and using various step-level importance signals to identify critical or redundant steps (Wang et al., 2025; Yuan et al., 2025; An et al., 2025). All of them target generation rather than hallucination detection. As we show in Section 5.2, their selection signals do not reliably identify noisy steps for detection. More broadly, attention scores have been used for token-level filtering and KV cache compression in LLMs (Zhang et al., 2023; Li et al., 2024; Chen et al., 2024b). Prior work applies attention patterns to identify pivotal reasoning sentences for interpretability (Bogdan et al., 2025). We explore using final-answer attention as a training and filtering signal to shape a representation space tailored for hallucination detection.

7 CONCLUSION

We propose REDE, a novel framework for filtering noisy reasoning steps to detect hallucination in LRMs. Our framework leverages final-answer attention to shape the step-level representation space, enabling reliable identification and removal of noisy steps. REDE is lightweight, uses annotation-free projection learning, and can be readily plugged into diverse downstream hallucination detectors. Extensive experiments show that REDE consistently improves hallucination detection performance across datasets, models, and detector families. We hope our work can inspire future research on reasoning-trace denoising and more broadly on building reliable methods for assessing LRMs.

AI USE STATEMENT

We used generative AI tools to polish the manuscript. In the experiments, LRMs generate reasoning traces, and Qwen3-32B provides automated correctness judgments and LLM-as-Judge step selections. The prompts and evaluation protocols are described in Appendices A.1 and B.1; robustness to alternative labeling methods is evaluated in Appendix C.6. The authors take responsibility for the final content of this work, including AI-assisted text, claims, and artifacts.

ETHICS STATEMENT

This work studies hallucination detection using public reasoning benchmarks and pre-trained models, whose licenses are documented in Appendix A.9. Human annotations concern model-generated reasoning steps; the annotation procedure and inter-annotator agreement are described in Appendix A.3. While REDE aims to improve the reliability of LRMs, filtering may discard informative steps, detection may miss hallucinations, and model-based evaluation may inherit biases or labeling errors. The method should therefore complement independent verification and human oversight, especially in high-stakes applications.

REPRODUCIBILITY STATEMENT

Section 4 describes the method, and Section 5 specifies the datasets, models, data splits, and evaluation protocol. The appendix provides the prompts and dataset details (Appendices A.1 and A.2), annotation guidelines (Appendix A.3), detector implementations and hyperparameter selection (Appendix A.4), embedding extraction (Appendix A.5), clustering and training/inference algorithms (Appendices A.6 and A.7), and software, hardware, and timing information (Appendix A.8). Results are averaged over three random seeds; Appendix B.1.1 reports additional uncertainty estimates and significance tests for step-selection comparisons.

REFERENCES

Sohyun An, Ruochen Wang, Tianyi Zhou, and Cho-Jui Hsieh. Don't think longer, think wisely: Optimizing thinking dynamics for large reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, 2023.

Forrest Bao, Chenyu Xu, and Ofer Mendelevitch. DeepSeek-R1 hallucinates more than DeepSeek-V3. https://www.vectara.com/blog/deepseek-r1-hallucinates-more-than-deepseek-v3, January 2025. Vectara Blog, accessed 2025-04-02.

Rohan Bhatnagar, Youran Sun, Chi Andrew Zhang, Yixin Wen, and Haizhao Yang. HALT: Hallucination assessment via latent testing. *arXiv preprint arXiv:2601.14210*, 2026.

Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. Thought anchors: Which LLM reasoning steps matter? *arXiv preprint arXiv:2506.19143*, 2025.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *The Eleventh International Conference on Learning Representations*, 2023.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. INSIDE: LLMs' internal states retain the power of hallucination detection. In *The Twelfth International Conference on Learning Representations*, 2024a.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024b.

Jiahao Cheng, Tiancheng Su, Jia Yuan, Guoxiu He, Jiawei Liu, Xinqi Tao, Jingwen Xie, and Huaxia Li. Chain-of-thought prompting obscures hallucination cues in large language models: An empirical evaluation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association for Computational Linguistics, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zhen Li, et al. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18581–18597, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Heejin Do, Jaehui Hwang, Dongyoon Han, Seong Joon Oh, and Sangdoo Yun. What defines good reasoning in LLMs? dissecting reasoning steps with multi-aspect evaluation. *arXiv preprint arXiv:2510.20603*, 2025.

Xuefeng Du, Chaowei Xiao, and Yixuan Li. HaloScope: Harnessing unlabeled LLM generations for hallucination detection. *Advances in Neural Information Processing Systems*, 37:102948–102972, 2024.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. *arXiv preprint arXiv:2508.15260*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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

Bin Hong, Jiayu Liu, Kai Zhang, Jianwen Sun, Mengdi Zhang, and Zhenya Huang. Pruning long chain-of-thought of large reasoning models via small-scale preference optimization. In *The Fourteenth International Conference on Learning Representations*, 2026.

Colin Hong, Xu Guo, Anand Chaanan Singh, Esha Choukse, and Dmitrii Ustiugov. Slim-SC: Thought pruning for efficient scaling with self-consistency. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 34488–34505, 2025.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. ThinkPrune: Pruning long chain-of-thought of LLMs via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025a.

Zhenyu Hou, Xin Lv, Rui Lu, Jiajie Zhang, Yujiang Li, Zijun Yao, Juanzi Li, Jie Tang, and Yuxiao Dong. T1: Advancing language model reasoning through reinforcement learning and inference scaling. In *Forty-second International Conference on Machine Learning*, 2025b.

Junjie Hu, Gang Tu, Cheng Shengyu, JinXin Li, Jinting Wang, Rui Chen, Zhilong Zhou, and Dongbo Shan. HARP: Hallucination detection via reasoning subspace projection. In *The Fourteenth International Conference on Learning Representations*, 2026.

Hugging Face H4. Huggingfaceh4/aime_2024, 2025. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024. Dataset.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556, 2019.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023.

J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pp. 159–174, 1977.

Hakyung Lee, Keon-Hee Park, Hoyoon Byun, Jeyoon Yeom, Jihee Kim, Gyeong-Moon Park, and Kyungwoo Song. CED: Comparing embedding differences for detecting out-of-distribution and hallucinated text. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14866–14882, 2024.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970, 2024.

Zeju Li, Jianyuan Zhong, Ziyang Zheng, Xiangyu Wen, Zhijian Xu, Yingying Cheng, Fan Zhang, and Qiang Xu. Making slow thinking faster: Compressing LLM chain-of-thought via step entropy. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zhixiang Liang, Beichen Huang, Zheng Wang, and Minjia Zhang. Hidden states as early signals: Step-level trace evaluation and pruning for efficient test-time scaling. In *Findings of the Association for Computational Linguistics: ACL 2026*, San Diego, California, USA, 2026. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*, 2022a. ISSN 2835-8856.

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

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Haolang Lu, Yilian Liu, Jingxin Xu, Guoshun Nan, Yuanlong Yu, Zhican Chen, and Kun Wang. Auditing meta-cognitive hallucinations in reasoning large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Haolang Lu, Minghui Pan, Ripeng Li, Guoshun Nan, Jialin Zhuang, Zijie Zhao, Zhongxiang Sun, Kun Wang, and Yang Liu. Streaming hallucination detection in long chain-of-thought reasoning. *arXiv preprint arXiv:2601.02170*, 2026.

Wen Luo, Guangyue Peng, Wei Li, Shaohang Wei, Feifan Song, Liang Wang, Nan Yang, Xingxing Zhang, Jing Jin, Furu Wei, et al. Two pathways to truthfulness: On the intrinsic encoding of LLM hallucinations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, San Diego, California, USA, 2026. Association for Computational Linguistics.

Yijia Luo, Yulin Song, Xingyao Zhang, Jiaheng Liu, Weixun Wang, GengRu Chen, Wenbo Su, and Bo Zheng. Deconstructing long chain-of-thought: A structured reasoning optimization framework for long cot distillation. *arXiv preprint arXiv:2503.16385*, 2025.

Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 9004–9017, 2023.

Pablo Miralles-González, Javier Huertas-Tato, Alejandro Martín, and David Camacho. Not all tokens are created equal: Perplexity attention weighted networks for ai generated text detection. *arXiv preprint arXiv:2501.03940*, 2025.

Mengjia Niu, Hamed Haddadi, and Guansong Pang. Robust hallucination detection in LLMs via adaptive token selection. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

OpenCompass Team. AIME 2025 dataset, 2025. URL https://huggingface.co/datasets/opencompass/AIME2025. Problems from the American Invitational Mathematics Examination (AIME) 2025-I & II.

Seongheon Park, Xuefeng Du, Min-Hsuan Yeh, Haobo Wang, and Yixuan Li. Steer LLM latents for hallucination detection. In *International Conference on Machine Learning*, pp. 47971–47990. PMLR, 2025.

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

Shanghaoran Quan, Jiaxi Yang, Bowen Yu, Bo Zheng, Dayiheng Liu, An Yang, Xuancheng Ren, Bofei Gao, Yibo Miao, Yunlong Feng, et al. CodeElo: Benchmarking competition-level code generation of LLMs with human-comparable elo ratings. *arXiv preprint arXiv:2501.01257*, 2025.

Jie Ren, Jiaming Luo, Yao Zhao, Kundan Krishna, Mohammad Saleh, Balaji Lakshminarayanan, and Peter J Liu. Out-of-distribution detection and selective generation for conditional language models. In *The Eleventh International Conference on Learning Representations*, 2023.

TYSS Santosh, Youssef Tarek Elkhayat, Oana Ichim, Pranav Shetty, Dongsheng Wang, Zhiqiang Ma, Armineh Nourbakhsh, and Xiaomo Liu. CoCoLex: Confidence-guided copy-based decoding for grounded legal text generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 19002–19018, 2025.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7881–7892, 2020.

Ponhvoan Srey, Xiaobao Wu, and Luu Anh Tuan. Unsupervised hallucination detection by inspecting reasoning processes. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 22128–22140, 2025.

Zhongxiang Sun, Qipeng Wang, Haoyu Wang, Xiao Zhang, and Jun Xu. Mechanistic detection and mitigation of hallucination in large reasoning models. In *The Fourteenth International Conference on Learning Representations*, 2026.

Lexiang Tang, Weihao Gao, Bingchen Zhao, Lu Ma, Bang Yang, Yuexian Zou, et al. Thinking by subtraction: Confidence-driven contrastive decoding for LLM reasoning. *arXiv preprint arXiv:2602.18232*, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Shangqing Tu, Yaxuan Li, Yushi Bai, Lei Hou, and Juanzi Li. DeepPrune: Parallel scaling without inter-trace redundancy. *arXiv preprint arXiv:2510.08483*, 2025.

Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of LLMs by validating low-confidence generation. *arXiv preprint arXiv:2307.03987*, 2023.

Changyue Wang, Weihang Su, Qingyao Ai, and Yiqun Liu. Joint evaluation of answer and reasoning consistency for hallucination detection in large reasoning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33377–33385, 2026.

Siyuan Wang, Enda Zhao, and Xiang Ren. Stepwise informativeness search for improving LLM reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25291–25309, 2025.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 11–20, 2019.

Canhui Wu, Qiong Cao, Chang Li, Zhenfang Wang, Chao Xue, Yuwei Fan, Wei Xi, and Xiaodong He. Beyond token length: Step pruner for efficient and accurate reasoning in large language models. *arXiv preprint arXiv:2510.03805*, 2025.

Heming Xia, Chak Tou Leong, Wenjie Wang, Yongqi Li, and Wenjie Li. TokenSkip: Controllable chain-of-thought compression in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3351–3363, 2025.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, Shuohang Wang, Weijian Xu, Jianfeng Gao, and Weizhu Chen. Phi-4-Mini-Reasoning: Exploring the limits of small reasoning language models in math. *arXiv preprint arXiv:2504.21233*, 2025. URL https://arxiv.org/abs/2504.21233.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Zijun Yao, Yantao Liu, Yanxu Chen, Jianhui Chen, Junfeng Fang, Lei Hou, Juanzi Li, and Tat-Seng Chua. Are reasoning models more prone to hallucination? *arXiv preprint arXiv:2505.23646*, 2025.

Hang Yuan, Bin Yu, Haotian Li, Shijun Yang, Christina Dan Wang, Zhou Yu, Xueyin Xu, Weizhen Qi, and Kai Chen. Not all tokens are what you need in thinking. *arXiv preprint arXiv:2505.17827*, 2025.

Wenhao Zeng, Yaoning Wang, Chao Hu, Yuling Shi, Chengcheng Wan, Hongyu Zhang, and Xiaodong Gu. Pruning the unsurprising: Efficient code reasoning via first-token surprisal. *arXiv preprint arXiv:2508.05988*, 2025.

Xinyue Zeng, Junhong Lin, Yujun Yan, Feng Guo, Liang Shi, Jun Wu, and Dawei Zhou. HalluGuard: Demystifying data-driven and reasoning-driven hallucinations in LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jianxiong Zhang, Bing Guo, Yuming Jiang, Haobo Wang, Bo An, and Xuefeng Du. Harnessing reasoning trajectories for hallucination detection via answer-agreement representation shaping. *International Conference on Machine Learning*, 2026.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2O: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Hanna Zubkova, Ji-Hoon Park, and Seong-Whan Lee. SUGAR: leveraging contextual confidence for smarter retrieval. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2025.

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 INPUT PROMPTS

We provide the detailed prompts used in our experiments for two purposes: (1) generating the original reasoning traces and final answers, and (2) evaluating the correctness of model-generated answers with an external judge model.

Generation prompts. For each dataset, we prepend a task-specific system instruction before the question. The system instructions are as follows:

- **TruthfulQA:** "*You are a factual question answering expert. Provide one concise and direct final answer.*"
- **MATH:** "*You are a mathematical reasoning expert. Solve the following problem step by step and give the final answer in the format \boxed{YOUR_ANSWER}. Answer concisely.*"
- **CodeElo:** "*You are a competitive programming expert. Solve the following Codeforces problem and provide a correct and efficient C++17 solution. Return only one final C++ code block. Answer concisely.*"
- **MULTIHOPQA:** "*You are a multi-hop question answering expert. Reason across the evidence*[1] *and provide one concise final answer.*"

These system instructions are combined with the model-specific prompt templates below.

Qwen Prompt

```
<|im_start|>user
{system_instruction}
Question: {question}
<|im_end|>
<|im_start|>assistant
```

Figure 7: Prompt used to generate reasoning traces and final answers for Qwen models.

DeepSeek-R1 Prompt

```
<|begin_of_sentence|><|User|>
{system_instruction}
Question: {question}
<|Assistant|><think>
```

Figure 8: Prompt used to generate reasoning traces and final answers for DeepSeek-R1 models.

Correctness evaluation prompt. Following prior work (Zhang et al., 2026; Yao et al., 2025; Zheng et al., 2023), we use Qwen3-32B as an external judge to assess whether a model-generated answer is correct, using the prompt shown in Figure 9. Answers graded as `INCORRECT` or `NOT_ATTEMPTED` are treated as hallucinations.

[1] Here, "evidence" refers to the multiple pieces of supporting information that must be connected to answer a multi-hop question. For example, answering "Was the director of *Jaws* born in the same country as the author of *Hamlet*?" requires combining evidence about the film and its director, the play and its author, and their birth countries.

Judge Prompt

```
Your job is to look at a question, multiple acceptable gold targets, and a
    predicted answer, and then assign a grade of either "CORRECT",
    "INCORRECT", or "NOT_ATTEMPTED".

IMPORTANT: The question has multiple acceptable correct answers provided as
    gold targets. The predicted answer is CORRECT if it matches any one of the
    gold targets according to the criteria below.

A predicted answer is CORRECT if:
- It fully contains the important information from at least one gold target.
- It does not contain any information that contradicts any gold target.
- Only semantic meaning matters; capitalization, punctuation, grammar, and
    order do not matter.
- Hedging or guessing is allowed as long as at least one gold target is fully
    included and the response contains no incorrect or contradictory
    statements.

A predicted answer is INCORRECT if:
- Any factual statement in the answer contradicts all the gold targets.
- Hedging statements are still considered incorrect if they introduce
    contradictory content.

A predicted answer is NOT_ATTEMPTED if:
- The important information from any gold target is not included.
- No statements in the answer contradict any of the gold targets.

Additional notes:
- Numerical answers must match the last significant figure of the gold target.
- If the gold target contains more information than the question requires, the
    predicted answer only needs to match the information requested by the
    question.
- Do not penalize omissions that are clearly inferable from the question.
- Do not penalize small typos in names if the intended meaning is clear.

Question: {question}
Gold targets: {targets}
Predicted answer: {predicted_answer}

Return only one of the following:
A
B
C

where:
A = CORRECT
B = INCORRECT
C = NOT_ATTEMPTED
```

Figure 9: Prompt used to evaluate the correctness of model-generated answers with Qwen3-32B.

Verbalized Uncertainty: Qwen

```
<|im_start|>user
A language model was asked: {question}
The model's answer was: {answer}

a. Very Certain
b. Fairly Certain
c. Moderately Certain
d. Somewhat Certain
e. Not Certain
f. Very Uncertain

Analyze its answer. How certain are you of the model's answer?
<|im_end|>
<|im_start|>assistant
```

Figure 10: Prompt used for verbalized uncertainty estimation with Qwen models.

Verbalized Uncertainty: DeepSeek-R1

```
<|begin_of_sentence|><|User|>
A language model was asked: {question}
The model's answer was: {answer}

a. Very Certain
b. Fairly Certain
c. Moderately Certain
d. Somewhat Certain
e. Not Certain
f. Very Uncertain

Analyze its answer. How certain are you of the model's answer?
<|Assistant|><think>
</think>
Answer:
```

Figure 11: Prompt used for verbalized uncertainty estimation with DeepSeek-R1 models.

A.2 Dataset details

We evaluate ReDe on four tasks: TruthfulQA, Math, CodeElo, and MultiHopQA.

TruthfulQA. We use TruthfulQA (Lin et al., 2022b), which contains 817 questions covering factual knowledge across diverse topics.

Math. We construct the evaluation set by combining MATH500 (Hendrycks et al., 2021), AIME 2024 (Hugging Face H4, 2025), and AIME 2025 (OpenCompass Team, 2025), resulting in 560 problems in total.

CodeElo. We use CodeElo (Quan et al., 2025), which contains 408 Codeforces problems requiring competition-level code generation.

MultiHopQA. We construct the evaluation set following prior work (Sun et al., 2026) by sampling 1,000 questions from HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023).

For all datasets, we reserve 25% of the available examples for testing, use 100 examples for validation, and use the remaining examples for training. Unless otherwise specified, model outputs are generated with greedy decoding.

Reasoning trace statistics. Table 4 reports the average number of reasoning tokens and reasoning steps per sample across datasets and models. Reasoning trace length varies substantially across tasks: CodeElo produces the longest traces (up to 7,808 tokens and 254.3 steps on average), while TruthfulQA and MULTIHOPQA are considerably shorter. Math problems fall in between, reflecting the iterative nature of symbolic reasoning. These statistics motivate the need for effective step filtering, as longer traces contain more opportunities for noisy steps to accumulate.

Table 4: Average reasoning trace length (tokens) and number of reasoning steps per sample.

Model	Dataset	Avg. Tokens	Avg. Steps
Qwen3-8B	TruthfulQA	2,294	65.8
	MATH	5,606	145.0
	CodeElo	7,808	254.3
	MULTIHOPQA	2,969	86.6
DeepSeek-R1-Distill-Llama-8B	TruthfulQA	2,090	54.5
	MATH	4,251	112.4
	CodeElo	7,656	267.2
	MULTIHOPQA	2,210	50.4

A.3 ANNOTATION DETAILS ON AIME 2024

To better understand the structure of noisy reasoning traces, we annotate the reasoning steps on AIME 2024 generated by Qwen3-8B. These annotations are used for analysis, including the examples in Figure 1 and the step-selection accuracy evaluation in Section 5.2.

Step segmentation. Before annotation, we segment each reasoning trace into step-level units. Following prior work on reasoning hallucination analysis (Sun et al., 2026), we adopt a hybrid segmentation strategy that combines discourse-marker cues with formatting-based boundaries. Specifically, we split the trace at explicit discourse markers that typically indicate a transition in reasoning, such as "Wait", "But", "However", "Hmm", and "Alternatively". We further use formatting-based boundaries, such as paragraph breaks, when they correspond to coherent changes in reasoning. This procedure yields a sequence of contiguous reasoning steps for subsequent annotation.

Annotation labels. Each reasoning step is assigned exactly one of three labels:

- **Informative**, if it provides non-redundant, problem-specific information that directly supports the final answer.
- **Irrelevant**, if it does not contribute meaningful problem-specific information for solving the problem or verifying the answer.
- **Repetitive**, if its main content is semantically redundant with later reasoning steps and is expressed more completely or more decisively afterwards.

Annotation statistics. Table 5 summarizes the distribution of the final labels across the three categories. Overall, repetitive steps form the largest category (2,127 steps, 48.8%), while informative steps account for 1,178 steps (27.1%) and irrelevant steps for 1,050 steps (24.1%). Taken together, irrelevant and repetitive steps make up 3,177 steps (73.0%) of the annotated traces, so we treat them as a single noisy class for the accuracy evaluation in Section 5.2.

Table 5: Manual step annotations on AIME 2024 with Qwen3-8B.

Step Category	Count
Informative	1,178
Irrelevant	1,050
Repetitive	2,127

Annotation procedure. We adopt a two-stage procedure to ensure annotation quality. In the first stage, we conduct three successive rounds of labeling on the segmented reasoning steps, where the label guidelines are iteratively refined based on observed ambiguous cases before producing an initial label for each step. In the second stage, three graduate students with relevant expertise in mathematical reasoning and NLP *independently* verify the initial labels on a randomly sampled subset without access to each other's judgments; disagreements with the initial label are resolved by

majority vote among the verifiers, and the result serves as the final label for the sampled subset. The three verifiers achieve a Fleiss' κ of 0.76 on this subset, indicating substantial agreement (Landis & Koch, 1977), which supports the reliability of the initial labels.

A.4 Implementation details

ReDe. For each reasoning trace, we first segment the trace into reasoning steps and extract a step-level embedding from the hidden representations of the backbone model. Unless otherwise specified, we use the final-layer hidden states and compute each step embedding using the aggregation in Appendix A.5. The projection module f_ϕ is implemented as a two-layer MLP with ReLU activation. We train it using Adam with learning rate $1e{-}4$, weight decay $1e{-}5$, cosine learning rate decay, and batch size 128. The projection dimension, loss weights, top/bottom selection ratio, and step drop ratio are selected by AUROC on the held-out set of 100 labeled validation examples. Correctness labels and human step annotations are not used to train the projection; the label-free CCS training objective likewise uses no correctness labels. Thus, annotation-free projection learning and unsupervised CCS training are combined with limited labeled validation for model selection. At test time, noisy steps are identified using kNN distance in the projected space with $k = 15$, and the remaining steps are used to construct the final detection representation. For the step score in Eq. 1, we uniformly average the per-head attention probabilities over all heads in the chosen layer.

Detector-training protocol. For every filtering strategy, we train supervised probing and CCS on representations constructed from that strategy's traces over the same training questions. The data splits, extraction procedure for the detector input, classifier architecture, optimization settings, and detector hyperparameters are identical across ReDe and all step-selection baselines. For probing, we concatenate the retained steps in their original order with the question and original answer, then extract the answer's last-token representation. The detector paired with ReDe is therefore trained on ReDe-filtered traces, while the original-trace baseline is trained on unfiltered traces. Appendix B.2 separately evaluates a probe trained on original traces and held fixed, isolating the effect of changing the test-time trace.

Downstream detectors. We evaluate ReDe with four downstream detectors: supervised probing (Azaria & Mitchell, 2023), CCS (Burns et al., 2023), verbalized uncertainty (Lin et al., 2022a), and perplexity (Ren et al., 2023; Cheng et al., 2025). For supervised probing, we follow prior work (Azaria & Mitchell, 2023) and adopt an MLP classifier with a 512-dimensional hidden layer. We use dropout 0.6, Gaussian input noise with $\sigma = 0.008$, and BatchNorm momentum 0.05. The classifier is trained with `BCEWithLogitsLoss` using SGD with learning rate 8×10^{-3}, momentum 0.9, and weight decay 5×10^{-2} for 100 epochs with batch size 128. A ReduceLROnPlateau scheduler is used with factor 0.5, patience 7, and minimum learning rate 10^{-4}. Unless otherwise specified, we extract representations from layers 0–35 and use the last-token representation for classification. For CCS, we follow its original setup (Burns et al., 2023) and train a lightweight CCS classifier instantiated as a single linear layer, optimized with AdamW (learning rate 1e-3, weight decay 1e-2). The model is trained on balanced positive–negative embedding pairs constructed via difference vectors for 1000 epochs and with a logical consistency loss. We repeat training for 10 random initializations and retain the best-performing checkpoint. For verbalized uncertainty, we follow prior work (Lin et al., 2022a) and prompt the backbone model to self-assess its confidence using a six-level certainty scale and use the resulting score for AUROC evaluation. For perplexity-based detection, we follow prior work (Ren et al., 2023; Cheng et al., 2025) and use the mean perplexity over answer tokens as the detection score.

Baselines. For lexical similarity (Lin et al., 2024) and semantic entropy (Kuhn et al., 2023), we reproduce them based on the original papers using multiple sampled responses from the same backbone model. SelfCheckGPT (Manakul et al., 2023) is implemented with its NLI-based variant, which uses a fine-tuned DeBERTa-v3-large model[2] to estimate the contradiction or entailment relation between the most-likely answer and sampled responses. For P(True) (Kadavath et al., 2022), we reproduce it following the original paper. For TSV (Park et al., 2025), we follow the default settings described in the original paper and train it on the same dataset using embeddings extracted from the same layer as in our main experiments. For HaloScope (Du et al., 2024), we train it on the same unlabeled training set as our method and use pseudo-labels generated via PCA to train the lightweight

[2] https://huggingface.co/MoritzLaurer/DeBERTa-v3-large-mnli-fever-anli-ling-wanli

MLP classifier. For EigenScore (Chen et al., 2024a), we search the optimal number of sampled generations on the validation set and report the best AUROC. For RACE (Wang et al., 2026), we follow the original implementation and adopt the setting without pre-extracted chains of thought, where reasoning steps are summarized automatically by the built-in extractor. For RHD (Sun et al., 2026), we reproduce it based on the official code. For HalluGuard (Zeng et al., 2026), we follow the default configuration described in the original paper. For ARS (Zhang et al., 2026), we use the officially released codebase with the default hyperparameters reported in the paper. All baselines are evaluated on the same test sets under the default settings reported in their respective papers.

A.5 Step embedding extraction

We provide the aggregation details for the step embeddings introduced in Section 4.2. Given the full prompt, reasoning trace, and final answer, we run the frozen LRM to obtain the contextual hidden state $\mathbf{h}(x_t) \in \mathbb{R}^d$ for each token. For step $\mathbf{c}_i$ with token indices I_i, we follow Miralles-González et al. (2025) and use perplexity-weighted pooling:

$$\mathbf{e}_i = \frac{\sum_{t \in I_i} \mathrm{PPL}(x_t)\, \mathbf{h}(x_t)}{\sum_{t \in I_i} \mathrm{PPL}(x_t)}, \qquad \mathrm{PPL}(x_t) = \frac{1}{p_\theta(x_t \mid x_{<t})}, \tag{3}$$

Here $p_\theta(x_t \mid x_{<t})$ is the next-token probability assigned by the frozen LRM. The same aggregation is used during training and inference. All filtering decisions are made from distances between projected step embeddings, as described in Section 4.3.

This pooling rule gives less predictable tokens greater influence in the step representation. It is an empirical aggregation choice, rather than a measure of token informativeness: surprising but unhelpful tokens, including unusual symbols or formatting artifacts, can also receive large weights. We compare it with uniform averaging and last-token extraction on TruthfulQA and on math tasks across three model families in Appendix B.5.

A.6 Clustering details for Section 3

We describe the procedure for clustering semantically similar reasoning steps and retaining representative steps from each cluster. The full procedure is summarized in Algorithm 1.

Given a reasoning trace $\mathbf{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$, we first extract step embeddings $\{\mathbf{e}_1, \ldots, \mathbf{e}_K\}$ using perplexity-weighted averaging (Eq. 3). We then iteratively assign each step to an existing cluster if its cosine similarity to the cluster centroid exceeds a threshold τ_c, or create a new cluster otherwise. After all steps are assigned, we retain either the earliest or latest step from each cluster based on its position in the trace. The retained steps are concatenated with the original question and answer, and the resulting sequence is fed through the LRM to extract the last-token hidden state of the answer as the answer embedding. We set $\tau_c = 0.7$.

A.7 Training and inference algorithms of our ReDe

We provide the complete training and inference procedures of ReDe in Algorithm 2 and Algorithm 3, respectively. During training, ReDe computes step-level attention scores from the LRM's own attention mechanism and uses them as supervision to train a lightweight projection module f_ϕ. During inference, ReDe applies the learned projection to identify and remove noisy steps without requiring attention scores or ground-truth labels, and passes the filtered reasoning trace to a downstream hallucination detector.

A.8 Compute resources and time

Software and hardware. We conduct all experiments using Python 3.10 and PyTorch 2.3.1 on NVIDIA A100 GPUs with 80 GB memory.

Training and inference time. We report the training and inference time of ReDe on TruthfulQA with Qwen3-8B. Based on tracked runs, training ReDe takes 681 seconds, and inference on the TruthfulQA test set takes 49 seconds. For comparison, Semantic Entropy (Kuhn et al., 2023) requires 1141 seconds, RACE (Wang et al., 2026) requires 1507 seconds, and SelfCheckGPT (Manakul et al., 2023) requires 2077 seconds under the same setting. These results show that ReDe

Algorithm 1: Step Clustering and Retention

Input: Reasoning trace $\mathbf{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_K\}$, similarity threshold τ_c, retention mode $m \in \{\texttt{earliest}, \texttt{latest}\}$
Output: Retained reasoning trace $\mathbf{C}^{\text{ret}}$

```
Extract step embeddings {e_1, ..., e_K} via Eq. 3
Initialize cluster list G ← ∅
for i = 1, ..., K do
    assigned ← False
    for each cluster G ∈ G do
        Compute centroid ē_G = (1/|G|) Σ_{j∈G} e_j
        if cos(e_i, ē_G) ≥ τ_c then
            G ← G ∪ {i}
            assigned ← True
            break
    if assigned = False then
        Append singleton cluster {i} to G
Initialize retained index set I^ret ← ∅
for each cluster G ∈ G do
    if m = earliest then
        I^ret ← I^ret ∪ {min(G)}
    else
        I^ret ← I^ret ∪ {max(G)}
Sort the indices in I^ret in ascending order
Construct C^ret by retaining {c_i | i ∈ I^ret} in the original trace order
return C^ret
```

Algorithm 2: Training Procedure of REDE

Input: Training set $\mathcal{D}_{\text{train}} = \{(\mathbf{p}^{(i)}, \mathbf{C}^{(i)}, \mathbf{a}^{(i)})\}_{i=1}^{N}$, frozen LRM with parameters θ, selection ratio ρ, loss weights $\lambda_{\text{disperse}}$ and $\lambda_{\text{separate}}$, number of epochs E
Output: Trained projection module f_ϕ

```
Initialize projection module f_φ: R^d → R^{d'}
// Step 1: Construct proxy sets from final-answer attention
for each training sample (p, C, a) in D_train do
    Let C = {c_1, ..., c_K}
    for i = 1, ..., K do
        Compute PPL-weighted step embedding e_i via Eq. 3
        Compute step-level attention score s_i via Eq. 1
    T ← indices of the top ρ% steps ranked by {s_i}_{i=1}^K
    B ← indices of the bottom ρ% steps ranked by {s_i}_{i=1}^K
    Store ({e_i}_{i=1}^K, T, B) for training
// Step 2: Train the projection module
for epoch = 1, ..., E do
    for each mini-batch of stored ({e_i}, T, B) do
        Compute projected embeddings z_i = f_φ(e_i) for all steps in the mini-batch
        Compute L_compact, L_disperse, and L_separate from the projected embeddings and proxy sets
        Compute total loss L = L_compact + λ_disperse L_disperse + λ_separate L_separate via Eq. 2
        Update φ by gradient descent on L
return f_φ
```

Algorithm 3: Inference Procedure of REDE

Input: Test sample $(\bar{\mathbf{p}}, \bar{\mathbf{C}}, \bar{\mathbf{a}})$, trained projection f_ϕ, number of nearest neighbors k, drop ratio ζ, downstream detector G
Output: Hallucination prediction $\hat{y} \in \{0, 1\}$
Let $\bar{\mathbf{C}} = \{\bar{\mathbf{c}}_1, \ldots, \bar{\mathbf{c}}_{\bar{K}}\}$
`// Step 1: Extract and project step embeddings`
for $i = 1, \ldots, \bar{K}$ **do**
  Compute PPL-weighted step embedding $\bar{\mathbf{e}}_i$ via Eq. 3
  Compute projected embedding $\bar{\mathbf{z}}_i = f_\phi(\bar{\mathbf{e}}_i)$
`// Step 2: Compute within-trace` k`NN distance scores`
for $i = 1, \ldots, \bar{K}$ **do**
  Let $\bar{\mathbf{z}}_i^{(k)}$ denote the k-th nearest neighbor of $\bar{\mathbf{z}}_i$ among $\{\bar{\mathbf{z}}_j\}_{j \neq i}$ within the same trace
  Compute distance score $S_i = 1 - \bar{\mathbf{z}}_i^\top \bar{\mathbf{z}}_i^{(k)} / (\|\bar{\mathbf{z}}_i\|_2 \|\bar{\mathbf{z}}_i^{(k)}\|_2)$
`// Step 3: Filter noisy steps`
$\mathcal{I}_{\text{drop}} \leftarrow$ indices of the top-$\zeta\%$ steps with the largest scores in $\{S_i\}_{i=1}^{\bar{K}}$
Construct $\bar{\mathbf{C}}^{\text{fil}}$ by retaining $\{\bar{\mathbf{c}}_i \mid i \notin \mathcal{I}_{\text{drop}}\}$ in the original trace order
`// Step 4: Downstream hallucination detection`
$\hat{y} \leftarrow G(\bar{\mathbf{p}}, \bar{\mathbf{C}}^{\text{fil}}, \bar{\mathbf{a}})$
return $\hat{y}$

has a clear efficiency advantage over these baselines while achieving better hallucination detection performance.

A.9 LICENSES OF EXISTING ASSETS

We list the licenses of the datasets and pre-trained models used in this work. All assets are used in accordance with their respective licenses and intended research use.

Datasets. TruthfulQA (Lin et al., 2022b), 2WikiMultihopQA (Ho et al., 2020), and CodeElo (Quan et al., 2025) are released under Apache License 2.0; MATH (Hendrycks et al., 2021), AIME 2024 (Hugging Face H4, 2025), AIME 2025 (OpenCompass Team, 2025), Bamboogle (Press et al., 2023), and GSM8K (Cobbe et al., 2021) are released under MIT License; HotpotQA (Yang et al., 2018) is released under CC BY-SA 4.0; and MuSiQue (Trivedi et al., 2022) is released under CC BY 4.0.

Pre-trained models. Qwen3-8B and Qwen3-32B (Yang et al., 2025) are released under Apache License 2.0. DeepSeek-R1-Distill-Llama-8B and DeepSeek-R1-Distill-Qwen-32B (Guo et al., 2025) are released under MIT License, with their base models additionally subject to the Llama 3.1 Community License and Apache License 2.0, respectively. The DeBERTa-v3-large-mnli-fever-anli-ling-wanli model used for the NLI variant of SelfCheckGPT (Manakul et al., 2023) is released under MIT License.

B ADDITIONAL EXPERIMENTS AND ABLATION STUDIES

We first compare step-selection methods across tasks and examine statistical significance, then isolate the effect of denoising with a frozen detector. We next evaluate step representations and loss components before examining the remaining design choices. Unless otherwise specified, all experiments are conducted on TruthfulQA using Qwen3-8B with supervised probing as the downstream detector. Detector columns in this section refer to CCS (Burns et al., 2023), supervised probing (Azaria & Mitchell, 2023), perplexity (Ren et al., 2023), and verbalized certainty (Lin et al., 2022a).

B.1 Additional results for selection strategies

Table 6 complements Figure 6 with supervised-probing, PPL, and verbalized-certainty results. REDE improves all three detectors over Original, with the best PPL and verbalized-certainty results. Step Entropy obtains the highest probing AUROC (87.31 versus 86.44), so the benefit is not uniform across individual configurations. The detector baselines are supervised probing (Azaria & Mitchell, 2023), perplexity (Ren et al., 2023), and verbalized certainty (Lin et al., 2022a); the alternative selectors are STEP (Liang et al., 2026), SPIRIT (Cui et al., 2025), ASAP (Zeng et al., 2025), Step Entropy (Li et al., 2026), and LLM-as-Judge (Yang et al., 2025).

Table 6: **Additional detector results for step selection strategies.** AUROC (%) on TruthfulQA using Qwen3-8B. The best result in each column is highlighted in **bold**.

Selection Strategy	Probing	PPL	Verbalized
Original	80.42	58.17	50.37
Remove all reasoning	76.12	57.03	48.23
Random	74.57	54.23	47.12
STEP	84.92	60.52	58.33
SPIRIT	78.14	57.93	56.78
ASAP	79.18	54.33	52.57
Step Entropy	**87.31**	61.49	60.03
LLM-as-Judge	74.81	59.17	50.94
REDE	86.44	**63.46**	**61.10**

B.1.1 Statistical comparison with Step Entropy

Step Entropy is the strongest competing selector in several settings, so we examine the differences in more detail. Tables 8 and 9 present AUROC across three random seeds and paired bootstrap p-values computed by resampling matched test examples, rather than testing only the three seed-level means. The latter table also includes 95% confidence intervals for the AUROC difference, measured in percentage points.

The paired bootstrap comparison of filtered and original traces in Table 2 is significant across all model–dataset–detector settings. On TruthfulQA, REDE improves over Step Entropy for CCS, PPL, and verbalized certainty with $p < 0.05$. Across the three additional datasets, the gains for CCS, PPL, and verbalized certainty also have $p < 0.05$. For probing, the improvement on CodeElo is significant under the pairwise test ($p = 0.0437$), whereas the small differences on MATH and MULTIHOPQA are not ($p = 0.6240$ and 0.7500). These results support improvements across detector families while identifying settings in which the two selectors are statistically comparable.

B.2 Denoising with a frozen downstream detector

The main comparison trains each detector on the traces produced by its paired filtering strategy (Appendix A.4). To isolate the effect of denoising from detector adaptation, we train a probing detector only on the original traces and keep all of its parameters fixed. At test time, we change only the input trace, comparing the original trace with the REDE-filtered trace. The question, original final answer, and detector remain the same.

Table 10 shows improvements on all four datasets with Qwen3-8B: AUROC increases by 3.05 points on TruthfulQA, 3.36 on MATH, 2.24 on CodeElo, and 2.78 on MULTIHOPQA. These gains show that denoising can improve the detector's input even without retraining it. Training the detector on filtered traces yields further gains in Table 2, so the main results reflect both the changed trace and detector adaptation to that input.

B.3 Step representations and loss ablations

Loss components on TruthfulQA. We remove each term from the full objective (Equation 2) and evaluate supervised probing on TruthfulQA with Qwen3-8B. Every removal lowers AUROC (Figure 13(a)). Removing $\mathcal{L}_{\text{disperse}}$ causes the largest drop, from 86.44% to 80.06%, as noisy steps cluster

Table 7: **Step-selection strategies across reasoning tasks.** AUROC (%) with Qwen3-8B at a common drop ratio of 70%. The best result for each dataset and detector is highlighted in **bold**.

Dataset	Selector	CCS	Probing	PPL	Verbalized
MATH	STEP	74.26	85.41	67.83	62.17
	SPIRIT	77.08	79.34	65.72	60.46
	ASAP	70.61	80.12	61.58	56.91
	Step Entropy	76.47	86.93	68.94	63.78
	REDE	**81.33**	**87.06**	**71.03**	**65.13**
CodeElo	STEP	75.42	85.93	69.27	80.24
	SPIRIT	77.61	79.48	66.14	79.91
	ASAP	71.08	80.37	63.18	74.83
	Step Entropy	77.46	86.71	70.24	82.46
	REDE	**82.17**	**87.19**	**72.03**	**83.89**
MULTIHOPQA	STEP	65.94	81.28	62.61	67.28
	SPIRIT	69.57	75.73	60.11	64.49
	ASAP	62.83	75.39	57.02	61.77
	Step Entropy	68.71	**83.02**	64.07	68.62
	REDE	**73.41**	82.94	**65.84**	**69.50**

Comparison across reasoning tasks. We extend the comparison to MATH, CodeElo, and MULTIHOPQA with Qwen3-8B, using the same drop ratio $\zeta = 70\%$ for all selectors. Table 7 reports results for STEP, SPIRIT, ASAP, Step Entropy, and REDE. All trainable detectors follow the common protocol in Appendix A.4. REDE achieves the highest CCS, PPL, and verbalized-certainty AUROC on all three datasets. Probing results are closer: REDE leads on MATH and CodeElo, while Step Entropy is slightly higher on MULTIHOPQA (83.02 versus 82.94).

Table 8: **Comparison with Step Entropy on TruthfulQA.** Qwen3-8B AUROC (%) across three seeds and pairwise p-values. The best mean for each detector is highlighted in **bold**.

Detector	Step Entropy	REDE	p-value
CCS	82.33 ± 1.06	$\mathbf{87.32} \pm 0.83$	0.0143
PPL	61.49 ± 1.18	$\mathbf{63.46} \pm 0.97$	0.0271
Verbalized	60.03 ± 1.37	$\mathbf{61.10} \pm 1.23$	0.0426

together and become harder to distinguish using kNN. Removing $\mathcal{L}_{\text{compact}}$ spreads informative steps into noisy regions; removing $\mathcal{L}_{\text{separate}}$ reduces the distance between the two groups. Below, we provide t-SNE plots, measurements of step separation, and loss ablations on math and coding tasks across three model families.

The individual loss-component results summarized in Section 5.2 are shown in Figure 13(a); the remaining panels are discussed in Appendix B.4. Figure 12 provides t-SNE visualizations corresponding to the loss ablation in Section 5.2. In the original embedding space, informative and noisy steps are heavily mixed. The full model produces a compact informative cluster clearly separated from scattered noisy steps, while removing any single loss leads to increased overlap.

Quantitative evaluation of step separation. We complement the visualization with measurements on the annotated AIME 2024 traces generated by Qwen3-8B (Appendix A.3). We group irrelevant and repetitive steps as noisy and compute the silhouette score, each group's average within-trace kNN distance, and noisy-step identification AUROC. As shown in Table 11, the learned projection increases the silhouette score from 0.08 to 0.34 and noisy-step identification AUROC from 0.71 to 0.84. Informative steps become closer to their neighbors (0.40 to 0.24), while noisy steps remain farther away (0.42 to 0.45), increasing the distance gap from 0.02 to 0.21. These measurements describe the structure induced by training; they do not require the raw embeddings to exhibit the same separation.

Table 9: **Statistical comparison across reasoning tasks.** Qwen3-8B AUROC (%) across three seeds. Δ is the mean AUROC of REDE minus that of Step Entropy; brackets give the 95% confidence interval for this difference. The best mean per row is highlighted in **bold**.

Detector	Step Entropy	REDE	Δ [95% CI]	p-value
MATH				
CCS	76.47 ± 1.71	$\mathbf{81.33} \pm 2.14$	$+4.86\ [+1.58, +8.94]$	0.0107
Probing	86.93 ± 1.08	$\mathbf{87.06} \pm 1.23$	$+0.13\ [-0.37, +0.71]$	0.6240
PPL	68.94 ± 1.21	$\mathbf{71.03} \pm 1.03$	$+2.09\ [+0.42, +3.93]$	0.0208
Verbalized	63.78 ± 1.42	$\mathbf{65.13} \pm 1.29$	$+1.35\ [+0.20, +2.84]$	0.0314
CodeElo				
CCS	77.46 ± 1.52	$\mathbf{82.17} \pm 1.76$	$+4.71\ [+1.74, +8.16]$	0.0058
Probing	86.71 ± 0.86	$\mathbf{87.19} \pm 0.95$	$+0.48\ [+0.04, +0.98]$	0.0437
PPL	70.24 ± 1.31	$\mathbf{72.03} \pm 1.12$	$+1.79\ [+0.16, +3.59]$	0.0256
Verbalized	82.46 ± 1.44	$\mathbf{83.89} \pm 1.36$	$+1.43\ [+0.18, +2.94]$	0.0189
MULTIHOPQA				
CCS	68.71 ± 2.12	$\mathbf{73.41} \pm 2.53$	$+4.70\ [+0.82, +9.11]$	0.0272
Probing	$\mathbf{83.02} \pm 1.48$	82.94 ± 1.67	$-0.08\ [-0.62, +0.39]$	0.7500
PPL	64.07 ± 1.14	$\mathbf{65.84} \pm 0.94$	$+1.77\ [+0.24, +3.48]$	0.0157
Verbalized	68.62 ± 1.33	$\mathbf{69.50} \pm 1.13$	$+0.88\ [+0.10, +1.82]$	0.0364

Table 10: **Frozen-detector control.** AUROC (%) with Qwen3-8B. The probing detector is trained on original traces and held fixed for both test-time inputs. The best result per dataset is highlighted in **bold**.

Test-time trace	TruthfulQA	MATH	CodeElo	MULTIHOPQA
Original	80.42	81.05	82.82	77.84
REDE-filtered	**83.47**	**84.41**	**85.06**	**80.62**

Table 11: **Step separation before and after projection learning.** Results on annotated AIME 2024 with Qwen3-8B. The distance gap is the noisy-group mean minus the informative-group mean. Identification AUROC is on a 0–1 scale.

Space	Silhouette ↑	Avg. kNN distance		Gap ↑	Identification AUROC ↑
		Informative ↓	Noisy ↑		
Original	0.08	0.40	0.42	0.02	0.71
Learned	**0.34**	**0.24**	**0.45**	**0.21**	**0.84**

Loss ablations across tasks and model families. We extend the loss ablation beyond TruthfulQA to MATH and CodeElo, using Qwen3-8B, DeepSeek-R1-Distill-Llama-8B, and Phi-4-mini-reasoning (Xu et al., 2025). Table 12 reports AUROC with supervised probing. Removing any loss component lowers AUROC in every setting, and removing $\mathcal{L}_{\text{disperse}}$ produces the largest drop in all six cases. Together with the measurements of step separation, these results support learning a compact informative region while discouraging noisy steps from forming a competing dense cluster across both mathematical and coding traces.

B.4 SENSITIVITY TO INFERENCE AND REPRESENTATION CHOICES

We examine the inference-time drop ratio and the choices of projection dimension and embedding layer. Figure 13 collects these analyses alongside the loss-component comparison discussed in Appendix B.3. All results use TruthfulQA, Qwen3-8B, and supervised probing.

Effect of the drop ratio ζ. The drop ratio controls the fraction of steps removed at inference time. At the default $\zeta = 70\%$, AUROC is 86.44%, compared with 80.42% without filtering (Figure 13(b)). More aggressive removal reduces performance, supporting a balance between removing noise and retaining informative steps.

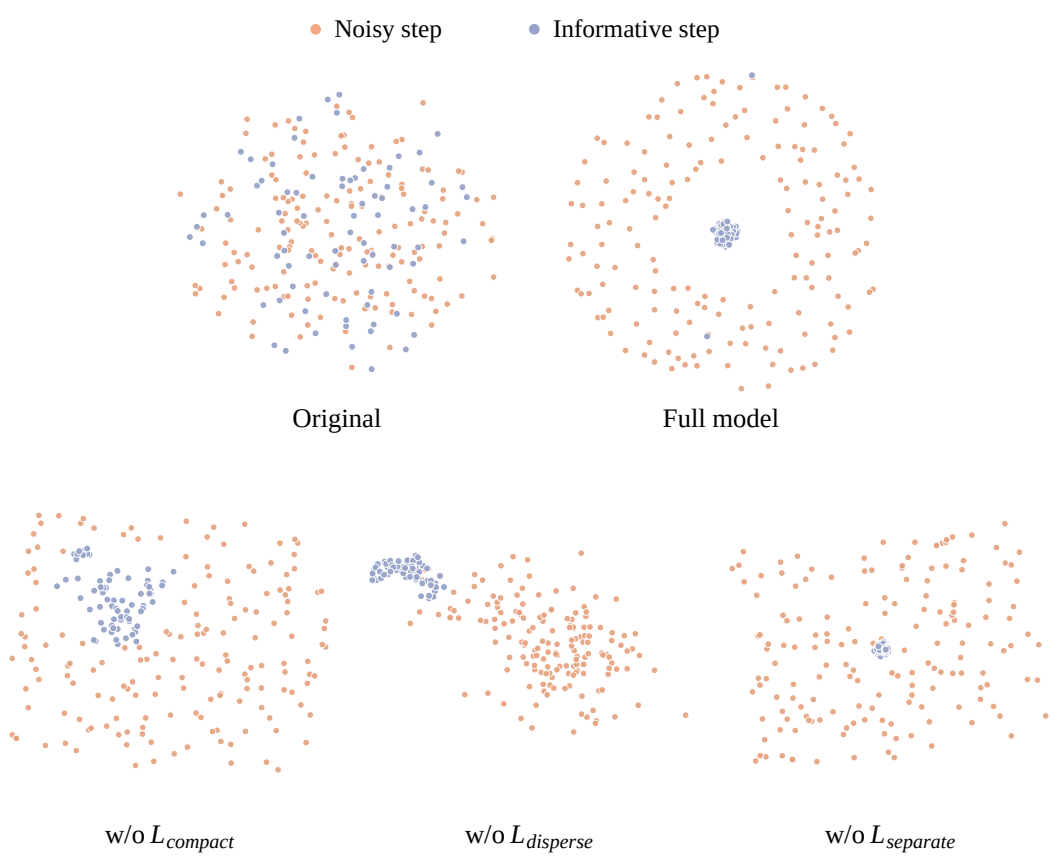


Figure 12: t-SNE visualization of step embeddings under each ablated loss variant on AIME 2024 (Hugging Face H4, 2025) with Qwen3-8B. Orange and blue points denote noisy and informative steps, respectively.

Table 12: **Loss ablations across tasks and model families.** AUROC (%) with supervised probing. Each ablation removes one term from Eq. 2. The best result for each model and dataset is highlighted in **bold**.

Model	Dataset	Full	w/o $\mathcal{L}_{\text{compact}}$	w/o $\mathcal{L}_{\text{disperse}}$	w/o $\mathcal{L}_{\text{separate}}$
Qwen3-8B	MATH	**87.06**	84.83	81.72	85.91
	CodeElo	**87.19**	84.41	82.33	85.37
DeepSeek-R1-Distill-Llama-8B	MATH	**83.91**	81.74	78.62	82.26
	CodeElo	**82.55**	79.62	77.31	80.94
Phi-4-mini-reasoning	MATH	**81.50**	77.96	76.31	79.72
	CodeElo	**79.84**	78.26	75.41	76.88

Effect of the projection dimension d'. We vary the output dimension of f_ϕ. With $d' = 1024$, AUROC is 86.44% (Figure 13(c)). The tested dimensions all outperform the unfiltered baseline, indicating that the denoising signal can be retained at different representation sizes.

Effect of the embedding layer. We extract step embeddings from different LRM layers and train a projection for each choice (Figure 13(d)). Intermediate-to-late layers give the highest AUROC, consistent with prior work (Zhang et al., 2026; Chen et al., 2024a). AUROC is at least 82.58 across the tested layers and 86.44 at the final layer, compared with 80.42 for the unfiltered trace, indicating that the gains do not depend on a single extraction layer.

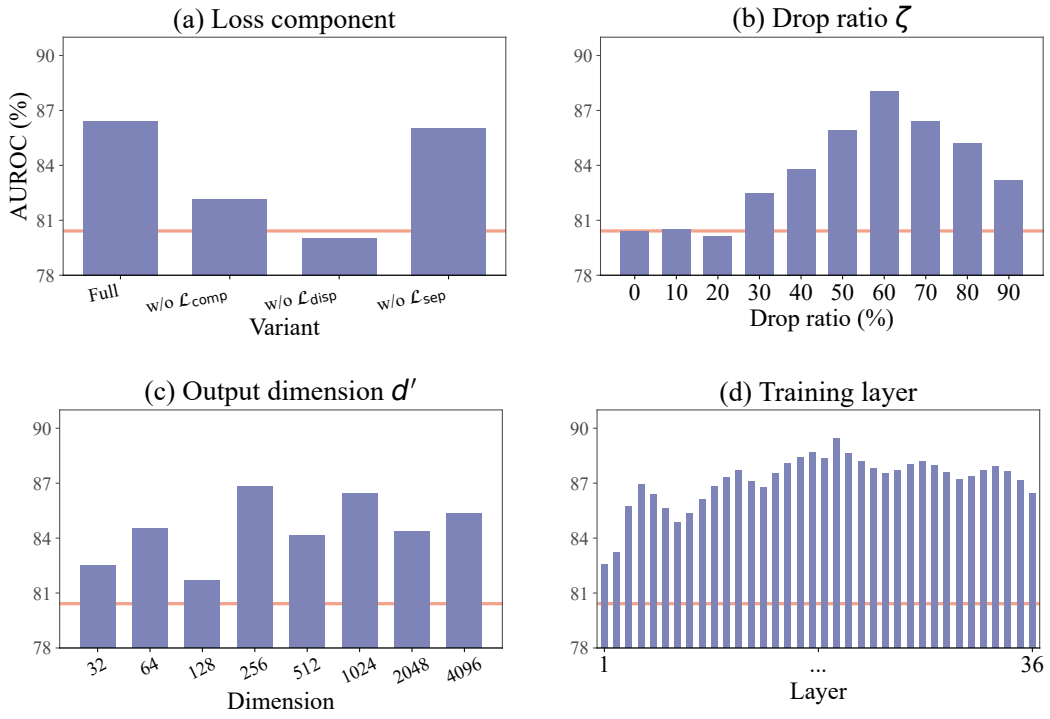


Figure 13: **Ablations of REDE.** Effects of (a) loss components, (b) drop ratio ζ, (c) projection dimension d', and (d) embedding layer. Results are AUROC (%) on TruthfulQA with Qwen3-8B and supervised probing. The orange line marks the unfiltered-trace baseline.

B.5 EFFECT OF STEP EMBEDDING WEIGHTING

We compare three ways to construct a step embedding from token hidden states: uniform averaging, last-token extraction, and the PPL-weighted aggregation described in Appendix A.5. Table 13 reports the comparison on TruthfulQA with Qwen3-8B. PPL-weighted pooling gives the highest AUROC across all four detectors, although its margins over last-token extraction are small in several settings. The comparison evaluates an aggregation choice; token perplexity itself is not the step-selection score.

Table 13: Effect of step embedding weighting strategy (AUROC, %) on TruthfulQA with Qwen3-8B. The best result in each column is highlighted in **bold**.

Weighting Strategy	CCS	Probing	PPL	Verbalized
Uniform average	82.56	85.41	60.88	59.08
Last token	86.18	86.05	63.31	60.09
PPL-weighted (Ours)	**87.32**	**86.44**	**63.46**	**61.10**

Comparison on math across model families. To evaluate the aggregation choice on symbolic reasoning and additional backbones, we repeat the comparison on MATH with Qwen3-8B, DeepSeek-R1-Distill-Llama-8B, and Phi-4-mini-reasoning (Xu et al., 2025). Table 14 shows that PPL-weighted pooling gives the highest AUROC for each detector in all three settings. Uniform averaging and last-token extraction also yield competitive results, indicating that the filtering framework can operate with different step representations. These results support the chosen pooling rule empirically while retaining the sensitivity to surprising but unhelpful tokens discussed in Appendix A.5.

B.6 EFFECT OF TRAINING SELECTION RATIO

During training, we select the top $\rho\%$ and bottom $\rho\%$ of steps by attention score to form the informative and noisy sets, respectively. Table 15 reports detection performance under varying ρ across

Table 14: **Step embedding aggregation on math across model families.** AUROC (%) on MATH. The best result for each model and detector is highlighted in **bold**.

Aggregation	CCS	Probing	PPL	Verbalized
Qwen3-8B				
Uniform average	75.84	85.72	67.91	62.48
Last token	79.92	86.51	70.54	64.37
PPL-weighted	**81.33**	**87.06**	**71.03**	**65.13**
DeepSeek-R1-Distill-Llama-8B				
Uniform average	69.73	82.26	65.88	73.54
Last token	72.61	83.14	68.72	75.41
PPL-weighted	**74.24**	**83.91**	**69.14**	**76.13**
Phi-4-mini-reasoning				
Uniform average	68.11	79.83	64.28	69.76
Last token	70.74	80.96	66.98	71.88
PPL-weighted	**72.18**	**81.50**	**67.64**	**72.93**

four downstream detectors. A small ρ provides insufficient contrast between the two groups, weakening the contrastive signal; a large ρ reduces the separation between the two ends of the attention distribution, leading to less discriminative contrastive samples. Performance consistently peaks in the range $\rho \in [20, 25]$, confirming that well-separated contrastive samples are important for learning a discriminative projection. The training-time ratio ρ complements the inference-time drop ratio $\zeta = 70\%$: ρ selects the most contrastive samples at the two ends of the attention distribution for effective supervision, while ζ enables thorough denoising at test time.

Table 15: Effect of training selection ratio ρ (%) on hallucination detection (AUROC, %) on TruthfulQA with Qwen3-8B. The best result in each column is highlighted in **bold**.

ρ (%)	CCS	Probing	PPL	Verbalized
10	82.19	84.06	59.11	54.42
15	84.52	82.29	61.05	57.82
20	**87.32**	83.60	**63.46**	**61.10**
25	87.05	**86.44**	62.74	59.18
30	85.53	85.51	63.02	53.67
35	84.41	86.07	60.31	57.93
40	86.07	84.05	60.85	54.13

B.7 EFFECT OF STEP SCORE AGGREGATION

In Eq. 1, we compute the step-level attention score s_i by aggregating attention weights from the final answer token to each reasoning step. We ablate two design choices for this aggregation: the position of the answer token used as the query, and the layer from which the attention is extracted.

Answer token position. We compare four choices: the first token, the middle token (1/2), mean pooling over all answer tokens, and the last token. As shown in Table 16, using the last answer token yields the best performance across all downstream detectors, suggesting that it accumulates the most complete contextual information from the answer sequence and therefore provides the most informative relevance signal for each reasoning step.

Layer. Fixing the query to the last answer token, we further vary the transformer layer from which the attention score is computed. As shown in Figure 14, early layers (0–7) are generally less effective ($\sim$ 83% AUROC), while intermediate-to-late layers yield stronger results, with local peaks around layer 10, 20, and 22. Layers 27–28 show a mild drop to around 84.5%, after which performance recovers and peaks at the final layer (86.44%). This supports our default choice of computing the step score from the last answer token at the final transformer layer.

Table 16: Effect of answer token position for step score computation on TruthfulQA with Qwen3-8B. The best result in each column is highlighted in **bold**.

Token Position	CCS	Probing	PPL	Verbalized
First token	86.17	82.15	54.09	52.96
Middle token (1/2)	85.09	85.35	57.48	55.72
Mean pooling	86.71	86.34	61.30	60.27
Last token (Ours)	**87.32**	**86.44**	**63.46**	**61.10**

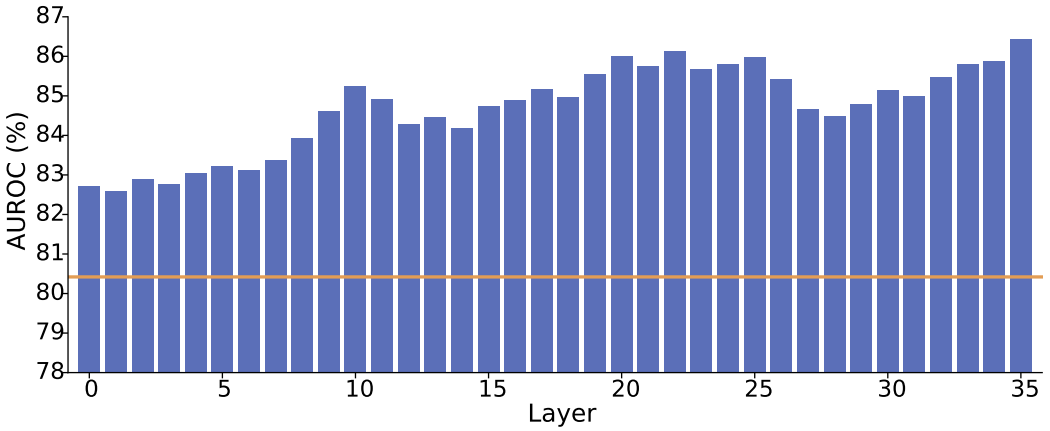


Figure 14: Effect of the layer used to compute the attention score in Eq. 1. Results are on TruthfulQA with Qwen3-8B, evaluated with supervised probing. The orange horizontal line denotes the performance of using the unfiltered reasoning trace.

B.8 Effect of loss weights

REDE uses two weighting hyperparameters $\lambda_{\text{disperse}}$ and $\lambda_{\text{separate}}$ to balance the three loss components (Eq. 2). We independently vary each weight while fixing the other to its default value, and report detection performance with supervised probing on TruthfulQA with Qwen3-8B in Figure 15. Note that $\lambda = 0$ corresponds to removing the respective loss component entirely. Performance remains stable across a moderate range of both weights, indicating that REDE is not overly sensitive to the precise choice of loss weights. A small weight weakens the corresponding training objective, reducing the quality of the learned projection; a large weight dominates the objective and suppresses the contributions of the other two losses.

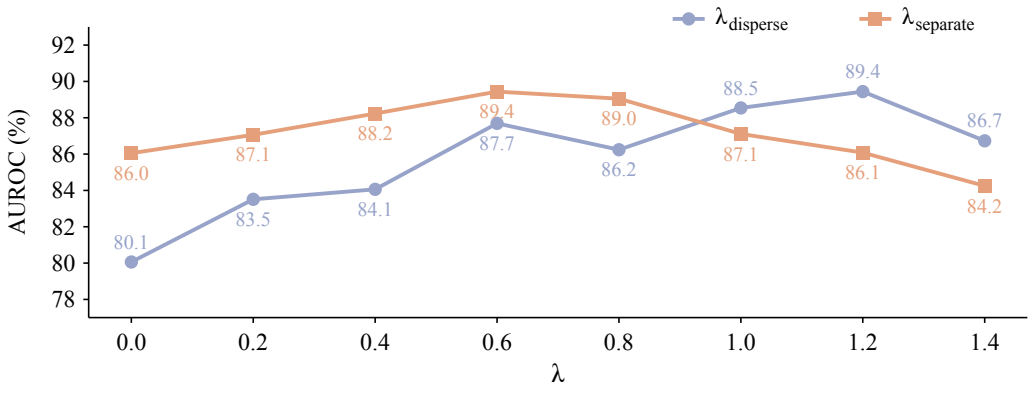


Figure 15: Effect of loss weights $\lambda_{\text{disperse}}$ and $\lambda_{\text{separate}}$ on hallucination detection on TruthfulQA with Qwen3-8B, evaluated with supervised probing. $\lambda = 0$ indicates the corresponding loss is removed.

B.9 Effect of different distance metrics

We compare cosine, Euclidean, and Mahalanobis distances for test-time filtering on TruthfulQA with Qwen3-8B and supervised probing (Table 17). Their AUROCs are 86.44%, 83.02%, and 81.51%, respectively; each exceeds the 80.42% unfiltered-trace baseline. Cosine is the default and performs best. The gains therefore persist across distance metrics, although their magnitude depends on the metric.

Table 17: Distance metrics (AUROC %). The best result is in bold.

Distance Metric	**AUROC (%)**
Cosine (Ours)	**86.44**
Mahalanobis	81.51
Euclidean	83.02

B.10 Effect of the number of nearest neighbors k

At inference time, we compute each step's distance score as its k-NN cosine distance with respect to all other steps within the same trace. Table 18 reports detection performance under varying k across four downstream detectors. Across all values of k, REDE consistently outperforms the original unfiltered reasoning trace, demonstrating that the learned projection produces a stable density structure that is robust to the choice of k. Performance peaks at $k = 15$ across all four detectors, confirming that this setting best captures the local density signal induced by the learned projection.

Table 18: Effect of k (AUROC %) on TruthfulQA with Qwen3-8B.

k	**CCS**	**Probing**	**PPL**	**Verbalized**
5	83.97	82.61	60.42	57.13
10	85.34	85.78	61.89	60.27
15	**87.32**	**86.44**	**63.46**	**61.10**
20	86.18	85.39	62.07	59.45
25	85.91	84.02	62.83	58.74
30	84.05	84.71	60.51	58.06

B.11 Effect of embedding extraction location

Following prior work (Du et al., 2024; Park et al., 2025), we investigate the effect of the multi-head attention (MHA) architecture on step-level representation quality. Specifically, the MHA can be conceptually expressed as:

$$f_{i+1} = f_i + Q_i \,\mathrm{Attn}_i(f_i), \tag{4}$$

where f_i represents the output of the i-th transformer block, $\mathrm{Attn}_i(f_i)$ denotes the output of the self-attention module in the i-th block, and Q_i is the output projection matrix of the attention sub-block. We evaluate hallucination detection performance using step embeddings extracted from three different locations within the MHA architecture, as shown in Table 19. The results show that the block output f is a favorable choice for detecting hallucinations across both LRM architectures.

Table 19: Effect of embedding extraction location on hallucination detection (AUROC, %) based on supervised probing (Azaria & Mitchell, 2023). The best result in each column is highlighted in **bold**.

Embedding Location	**Qwen3-8B**		**DeepSeek-R1-Distill Llama-8B**	
	TruthfulQA	Math	TruthfulQA	Math
$\mathrm{Attn}(f)$	82.40	84.06	72.08	80.05
$Q\,\mathrm{Attn}(f)$	85.50	83.44	73.18	78.52
f (Ours)	**86.44**	**87.06**	**74.37**	**83.91**

C Additional analysis

C.1 Validating the attention signal

We further investigate (i) whether final-answer attention reflects a meaningful notion of step usefulness, and (ii) the advantage of REDE over simple position-based filtering strategies. Experiments are conducted on all four benchmarks with Qwen3-8B, using supervised probing as the downstream detector. All filtering operations use the same drop ratio $\zeta = 70\%$ as REDE.

Learning from attention versus direct filtering. Direct attention-based filtering underperforms REDE across all four detectors (Table 3), including 80.51% versus 87.32% AUROC with CCS. Attention provides a useful training signal, but its sensitivity to trace length and structure can limit direct filtering. REDE uses this signal to learn a projection in which informative steps form a compact region and noisy steps become outliers. This enables reliable distance-based filtering without attention computation at inference.

Final-answer attention reflects step usefulness. We compare two filtering strategies against the original unfiltered trace: dropping the top-70% and the bottom-70% of reasoning steps ranked by their final-answer attention score. For each strategy, we evaluate the cosine similarity between the answer embedding of the remaining trace and that of the full trace, together with the downstream detection AUROC. As shown in Table 20, dropping high-attention steps substantially degrades both metrics compared with the original trace (e.g., AUROC drops from 80.42 to 57.18 on TruthfulQA), whereas dropping low-attention steps preserves and often improves detection quality (e.g., 84.97 vs. 80.42 on TruthfulQA) while maintaining a high cosine similarity to the full-trace answer embedding. This asymmetry supports an empirical association between final-answer attention and step usefulness for detection. It does not establish that the attention score is a causal attribution of each step to answer correctness.

Comparison with position-based filtering strategies. We compare REDE against two position-based baselines under the same drop ratio: Drop-Latest removes the latest 70% of steps, and Drop-Earliest removes the earliest 70% of steps. As shown in Table 21, Drop-Latest performs worst across all datasets, and while Drop-Earliest is stronger, it still falls well below REDE (e.g., 86.44% vs. 77.53% on TruthfulQA). This demonstrates that our attention-based supervision captures step-level usefulness beyond simple positional patterns, and REDE effectively leverages this signal to identify informative steps throughout the reasoning trace.

Table 20: Effect of dropping steps ranked by final-answer attention score on Qwen3-8B, compared against the original unfiltered trace. "Cos Sim" denotes the cosine similarity between the answer embedding of the remaining trace and that of the full trace (1.000 for the original trace by definition); AUROC is computed with supervised probing (Azaria & Mitchell, 2023).

	Original trace (no filtering)		**Drop top-70% attention**		**Drop bottom-70% attention**	
Dataset	Cos Sim	AUROC	Cos Sim	AUROC	Cos Sim	AUROC
TruthfulQA	1.000	80.42	0.427	57.18	0.872	84.97
MATH	1.000	81.05	0.361	52.84	0.815	83.62
CodeElo	1.000	82.82	0.483	61.07	0.846	85.11
MULTIHOPQA	1.000	77.84	0.394	50.62	0.829	79.48

Table 21: Comparison of REDE against position-based filtering baselines, all using drop ratio $\zeta = 70\%$ on Qwen3-8B. Drop-Latest removes the latest 70% of steps (retaining the earliest 30%), and Drop-Earliest removes the earliest 70% of steps (retaining the latest 30%). AUROC is computed with supervised probing (Azaria & Mitchell, 2023). The best result in each column is highlighted in **bold**.

Method	**TruthfulQA**	**MATH**	**CodeElo**	**MULTIHOPQA**
Drop-Latest	68.42	65.71	70.83	66.94
Drop-Earliest	77.53	75.68	79.14	74.29
REDE (Ours)	**86.44**	**87.06**	**87.19**	**82.94**

C.2 Accuracy of step selection

We further evaluate whether REDE can accurately identify noisy steps. On AIME 2024 (Hugging Face H4, 2025) with Qwen3-8B, we compare the steps flagged as noisy by REDE against human annotations that categorize each step as informative, irrelevant, or repetitive. REDE achieves an accuracy of 86.37% in classifying steps as informative versus noisy, confirming that the learned projection reliably identifies genuinely noisy steps and that the downstream detection gains stem from accurate step filtering.

C.3 Additional results on larger LRMs

Figure 5(b) demonstrates the scalability of REDE on TruthfulQA. Here we extend this evaluation to the other three benchmarks. As shown in Table 22, REDE consistently outperforms two competitive LRM-specific baselines, RACE (Wang et al., 2026) and ARS (Zhang et al., 2026), across all benchmarks on both Qwen3-32B and DeepSeek-R1-Distill-Qwen-32B. For example, compared with ARS, REDE improves by 14.63%, 4.27%, and 8.41% on MATH, CodeElo, and MULTIHOPQA with Qwen3-32B, respectively. Similar trends hold on DeepSeek-R1-Distill-Qwen-32B, where REDE surpasses ARS by 2.99% on MATH, 3.55% on CodeElo, and 5.47% on MULTIHOPQA. These results confirm that reasoning-step denoising remains effective across diverse reasoning tasks as LRMs scale up.

Table 22: Additional results on larger LRMs across three benchmarks. All values are AUROC with supervised probing as the downstream detector. The best result for each model and dataset is highlighted in **bold**.

	Qwen3-32B			DeepSeek-R1-Distill Qwen-32B		
Method	MATH	CodeElo	MULTIHOPQA	MATH	CodeElo	MULTIHOPQA
RACE	77.14	80.96	77.83	67.66	63.72	67.91
ARS	70.09	78.94	73.26	82.44	78.63	70.47
REDE	**84.72**	**83.21**	**81.67**	**85.43**	**82.18**	**75.94**

C.4 Qualitative analysis

We present step-level case studies and trace-level embedding visualizations to illustrate how REDE improves hallucination detection in practice. The step-level examples use reasoning traces generated by Qwen3-8B (Yang et al., 2025) on the MATH benchmark with greedy decoding, and detection scores are computed using supervised probing (Azaria & Mitchell, 2023).

Answer embedding visualization. We further examine whether step-level denoising improves the separability of truthful and hallucinated traces. Figure 16 shows the answer-level embeddings on TruthfulQA with Qwen3-8B. In the original space (left), the two classes are heavily mixed. After REDE filtering (right), they become more clearly separated, with only limited overlap near the decision boundary. This visualization complements the frozen-detector control in Appendix B.2, providing empirical evidence that filtering can improve the representations used for downstream detection.

Step-level filtering examples. Figures 17 and 18 show two representative traces after REDE filtering, where each reasoning step is marked as either kept or filtered. In Figure 17, the model gives an incorrect answer due to an arithmetic error. The original 74-step trace contains substantial repetitive and tangential content, leading to a low detection score of 0.41. After REDE retains 26 of the 74 steps (64.9% filtered), the score increases to 0.86, making the error more salient to the detector.

Figure 18 shows the complementary case, where the model reaches the correct answer through a verbose but valid derivation. The original trace receives a score of 0.77, incorrectly suggesting hallucination. After REDE retains 24 of the 78 steps (69.2% filtered), the score drops to 0.28, reducing this false positive. Together, these examples show that REDE improves detection for both hallucinated and truthful responses by filtering noisy reasoning steps.

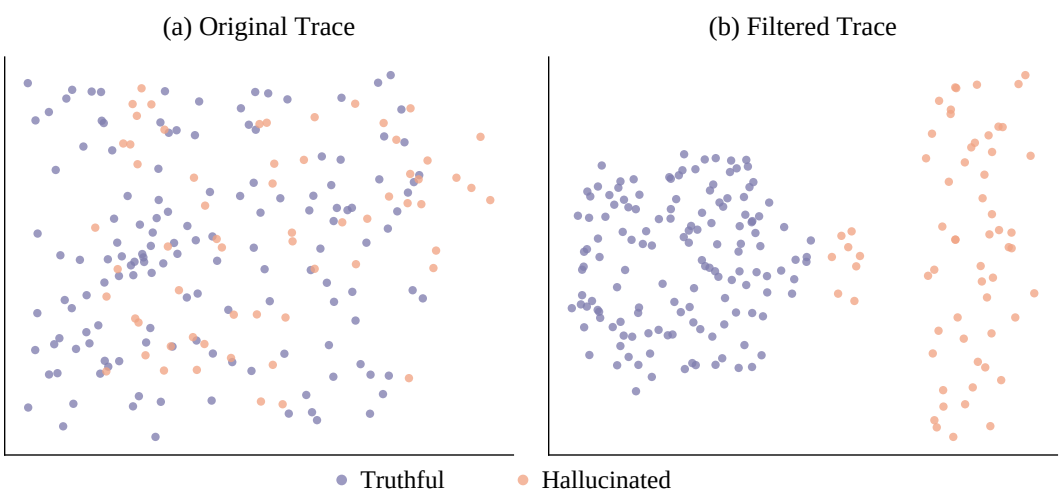


Figure 16: Answer embeddings on TruthfulQA with Qwen3-8B via supervised probing (Azaria & Mitchell, 2023). Left: original full traces, where truthful and hallucinated responses overlap heavily. Right: after REDE denoising, the two groups become clearly more separable.

Sample 1 — Kept / Filtered

Problem. Tetrahedron $ABCD$: $AB=CD=\sqrt{41}$, $AC=BD=\sqrt{80}$, $BC=AD=\sqrt{89}$. Point I inside equidistant to all faces at distance $\frac{m\sqrt{n}}{p}$. Find $m+n+p$. **GT: 104**

Pred: 102 ✗

S1 Equidistant point = incenter, distance = inradius r.	**S26** Area $= 2\sqrt{189}$.	**S51** Minor 2: $5696 - 6400 = -704$.
S2 In a triangle, incenter equidistant from sides.	S27 $\sqrt{445}$ cancels in product.	S52 $32\times178 = 5696$. ✓
S3 2D: $r = A/s$ for area A, semiperimeter s.	**S28** $\sqrt{189} = 3\sqrt{21}$. Area $= 6\sqrt{21}$.	**S53** Minor 3: $4096 - 8000 = -3904$.
S4 Incenter splits tet into 4 sub-tetrahedra.	S29 Alt verify: $16A^2 = 12096$. $A^2 = 756$. ✓	S54 $32\times128 = 4096$. ✓
S5 Each has base = face, height = r.	S30 $6\sqrt{21} \approx 27.5$. Reasonable.	S55 $160\times50 = 8000$. ✓
S6 $V = \frac{1}{3}Bh$. Basic formula.	**S31** $S = 4\times6\sqrt{21} = 24\sqrt{21}$.	**S56** $\det(N) = -82(12096) - 32(704) + 50(3904)$.
S7 Sub-volumes: $\frac{1}{3}rS_i$.	S32 Now compute V via Cayley–Menger.	**S57** $= -991872 - 22528 + 195200 = -819200$.
S8 $V = \frac{r}{3}S_{\text{tot}}$, so $r = 3V/S$.	**S33** $288V^2 = \lvert\text{CM}\rvert$, 5×5 det.	S58 Verify sub-products. ✓
S9 Need volume V and surface area S.	**S34** CM with $AB^2=41,\ldots,CD^2=41$.	S59 $-991872 - 22528 = -1014400$. ✓
S10 Opposite edges equal ⇒ disphenoid.	S35 Write out full matrix.	S60 CM $= 819200$.
S11 All 4 faces: $\sqrt{41}, \sqrt{80}, \sqrt{89}$.	S36 Use row reduction.	S61 $819200 = 2^{14}\times50$. Clean.
S12 $\triangle ABC$: $\sqrt{41}, \sqrt{89}, \sqrt{80}$.	**S37** Row ops: R_2', R_3', R_4'.	**S62** $V^2 = 25600/9$. $V = 160/3$.
S13 $\triangle ABD$: same. $\triangle ACD$: same.	S38 Verify R_2'. ✓	S63 $819200/16 = 51200$, $/18 = 25600/9$. ✓
S14 $\triangle BCD$: same. Confirmed.	S39 Verify R_3'. ✓	S64 $\sqrt{25600} = 160$. ✓
S15 $S = 4\times$area(one face).	S40 Verify R_4'. ✓	S65 $V \approx 53.3$. Plausible.
S16 Check right triangle? $41, 80, 89$.	**S41** CM $= -\det(M)$, 4×4 minor.	**S66** $r = 3V/S = 160/(24\sqrt{21})$.
S17 $41+80=121\neq89$. Not right.	S42 Antisymmetry in rows. Disphenoid.	**S67** $\gcd(160,24) = 8$: $r = 20/(3\sqrt{21})$.
S18 $41+80 = 11^2$, $80+89 = 13^2$. Unhelpful.	**S43** Reduce to 3×3: N.	S68 $160 = 2^5\times5$, $24 = 2^3\times3$. ✓
S19 Need other method for area.	S44 Verify R_1''. ✓	**S69** Rationalize: $r = 20\sqrt{21}/(3\times21) = 20\sqrt{21}/61$.
S20 Law of cosines at C.	S45 Verify R_2''. ✓	S70 $3\times21 = 60+1 = 61$.
S21 $41 = 169 - 2\sqrt{7120}\cos C$.	S46 Verify R_3''. ✓	S71 61 prime. $\gcd(20,61) = 1$. ✓
S22 $\cos C = 16/\sqrt{445}$.	S47 N symmetric as expected.	S72 $21 = 3\times7$, square-free. ✓
S23 $\sqrt{7120} = 4\sqrt{445}$. ✓	**S48** Expand $\det(N)$ row 1.	**S73** $m=20, n=21, p=61$. $m+n+p = 102$.
S24 $\cos^2 C = 256/445 < 1$, acute.	**S49** Minor 1: $28480 - 16384 = 12096$.	S74 Alt coords: $a=2, b=5/2, c=4$. Same V. ✓
S25 $\sin^2 C = 189/445$.	S50 $160\times178 = 28480$. ✓	

	Steps	Redundancy Ratio	Score	
Original Trace	74	0% filtered	**0.41**	
REDE-filtered	26 (of 74)	64.9% filtered	**0.86**	+0.45 ↑

Figure 17: **Sample 1 (incorrect answer).** REDE step-level filtering on a tetrahedron inradius problem from MATH with Qwen3-8B. Green: kept; gray: filtered. Detection scores are from supervised probing (Azaria & Mitchell, 2023). After filtering 64.9% of the steps, the score increases from 0.41 to 0.86, correctly flagging the hallucinated response.

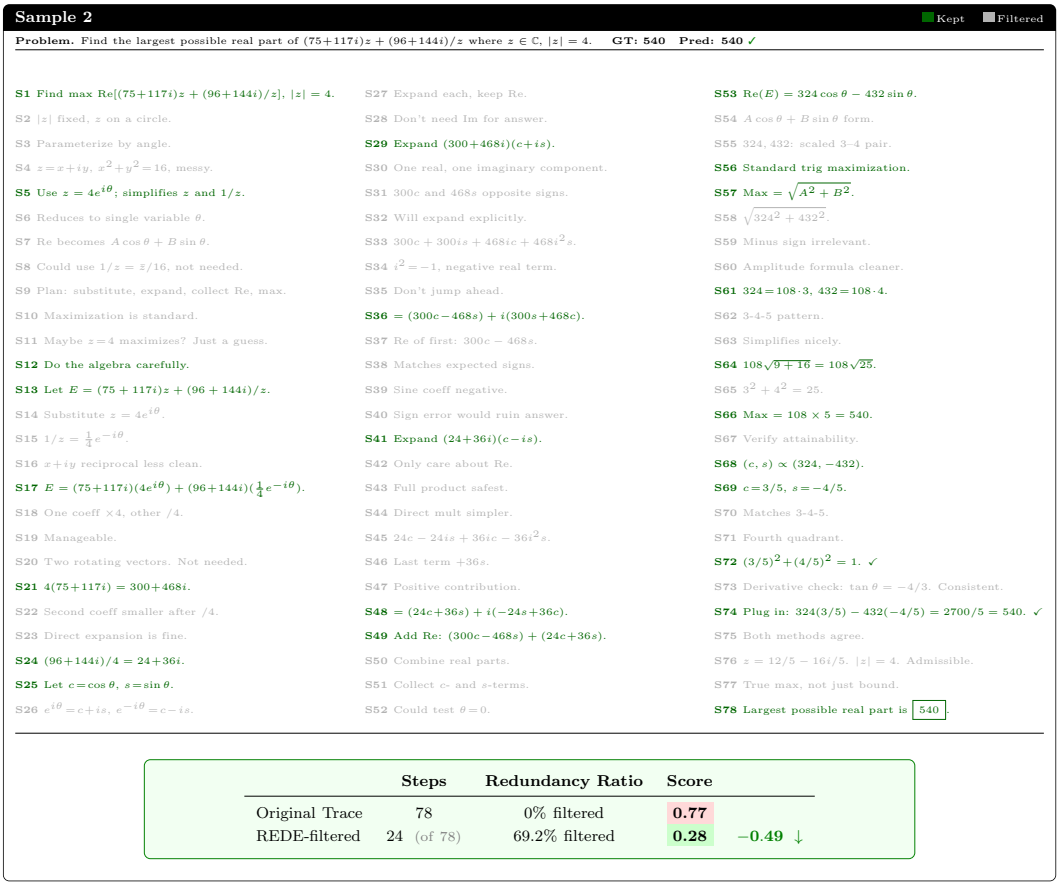


Figure 18: **Sample 2 (correct answer).** REDE step-level filtering on a complex-number optimization problem from MATH with Qwen3-8B. Green: kept; gray: filtered. Detection scores are from supervised probing (Azaria & Mitchell, 2023). After filtering 69.2% of the steps, the score decreases from 0.77 to 0.28, correctly reducing the false-positive rate.

C.5 Effect of Sampling Strategies

We evaluate hallucination detection performance when REDE generates reasoning traces under different sampling strategies. Our main results are obtained with greedy decoding, which selects the most likely next token at each step. We additionally compare it with multinomial sampling using a temperature of 0.5. We evaluate CCS (Burns et al., 2023), supervised probing (Azaria & Mitchell, 2023), perplexity (Ren et al., 2023), and verbalized certainty (Lin et al., 2022a). As shown in Table 23, greedy decoding consistently outperforms multinomial sampling across all detectors. The improvement is particularly notable for the Perplexity and Verbalized detectors, suggesting that more deterministic reasoning traces provide more reliable signals for downstream hallucination detection.

Table 23: Sampling strategies on TruthfulQA with Qwen3-8B (AUROC, %). The best result for each detector is in **bold**.

Detector	Multinomial	Greedy (Ours)
CCS	86.49	**87.32**
Probing	85.07	**86.44**
PPL	58.11	**63.46**
Verbalized	57.14	**61.10**

C.6 Effect of Truthfulness Labeling Methods

In our main experiments, the correctness of model-generated answers is judged using a strong external model Qwen3-32B. In this ablation, we show that the results remain robust under different judgment methods, including ROUGE-L, BLEURT (Sellam et al., 2020), and a different judge model,

DeepSeek-R1-Distill-Qwen-32B (Guo et al., 2025). For ROUGE-L, a generation is deemed truthful when the similarity score between the generation and the ground truth exceeds a threshold of 0.3. For BLEURT, we use the `bleurt-base-128` variant, a learned metric built upon BERT (Devlin et al., 2019) and augmented with diverse lexical and semantic-level supervision signals. As shown in Table 24, REDE consistently achieves strong detection performance across all labeling methods, confirming that our approach does not rely on a specific judge model or evaluation metric.

Table 24: Hallucination detection results (AUROC, %) under different truthfulness labeling methods on TruthfulQA and MATH. For REDE, we report results using probing as the downstream detector. The best result for each labeling method and dataset is highlighted in **bold**.

	ROUGE-L		BLEURT		DeepSeek-R1-Distill Qwen-32B	
Detector	TruthfulQA	MATH	TruthfulQA	MATH	TruthfulQA	MATH
RACE	85.82	77.34	86.21	76.95	65.21	69.02
ARS (CCS)	93.56	81.71	90.18	84.75	80.07	65.49
REDE (Probing)	**95.08**	**83.77**	**92.77**	**85.04**	**85.13**	**78.41**

C.7 RESULTS ON AN ADDITIONAL DATASET

To further evaluate the effectiveness of REDE beyond the datasets used in the main paper, we conduct experiments on GSM8K (Cobbe et al., 2021). For mathematical reasoning, we use the train split, which contains 7,473 problems. Following the same experimental setup as in our main experiments, we reserve 25% of the data for testing, use 100 examples for validation, and use the remaining examples for training. As shown in Table 25, using REDE-selected steps consistently outperforms using the original reasoning trace across all detectors and both model families. The gains are particularly substantial for CCS and remain consistent for probing, perplexity, and verbalized confidence, showing that reasoning-step denoising remains effective on this additional mathematical reasoning dataset. The four detector columns correspond to CCS (Burns et al., 2023), supervised probing (Azaria & Mitchell, 2023), perplexity (Ren et al., 2023), and verbalized certainty (Lin et al., 2022a).

Table 25: **Comparison of hallucination detection using the original reasoning trace versus REDE-selected steps on GSM8K.** All values are percentages (AUROC). The best results for each model and detector are highlighted in **bold**.

Model	**Trace**	**CCS**	**Probing**	**PPL**	**Verbalized**
Qwen3-8B	Original	60.08	87.50	55.21	50.07
	Filtered	**91.01**	**92.44**	**60.03**	**60.09**
DeepSeek-R1-Distill-Llama-8B	Original	58.31	78.61	52.31	53.81
	Filtered	**84.22**	**84.66**	**58.07**	**62.30**

C.8 APPLICABILITY IN A BLACK-BOX SETTING

Our main setup assumes white-box access to the target LRM in order to compute final-answer attention and step embeddings. We further study whether REDE can be applied when the target LRM is a black box, by using a separate white-box LRM as a proxy. Specifically, for each input generated by the black-box target, we feed the same prompt and its generated reasoning trace into the proxy to compute the final-answer attention and step embeddings, and then apply REDE to obtain the filtered trace. The filtered trace is passed back to the target model for verbalized certainty estimation, which does not require internal access. We use Qwen3-32B (Yang et al., 2025) as the proxy and evaluate on two target models: DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025) and Qwen3-8B (Yang et al., 2025).

As shown in Table 26, the proxy-based variant consistently improves detection over the original trace across all four benchmarks and both target models, improving over the original by 5.92% on MultiHopQA with DeepSeek and 8.04% on TruthfulQA with Qwen3-8B, recovering most of

the gain obtained with white-box REDE. This suggests that REDE remains effective in black-box settings where a suitable proxy LRM is available.

Table 26: Applicability of REDE in a black-box setting with Qwen3-32B as the white-box proxy. Detection uses verbalized certainty, which does not require internal access to the target. "REDE (proxy)" uses Qwen3-32B for attention scoring and step embedding, while "REDE (white-box)" has direct access to the target and is shown as an upper-bound reference. All values are AUROC (%). The best result among black-box methods is highlighted in **bold**.

Setting	TruthfulQA	MATH	CodeElo	MULTIHOPQA
Target: DeepSeek-R1-Distill-Llama-8B				
Original trace	56.61	69.71	79.13	50.87
REDE (proxy)	**59.85**	**74.26**	**82.07**	**56.79**
REDE (white-box)	61.94	76.13	84.49	59.56
Target: Qwen3-8B				
Original trace	50.37	57.19	78.89	63.75
REDE (proxy)	**58.41**	**62.83**	**82.14**	**67.62**
REDE (white-box)	61.10	65.13	83.89	69.50

C.9 IMPACT OF STEP FILTERING ON DOWNSTREAM TASK ACCURACY

Our main experiments measure hallucination detection performance (AUROC). Here we examine a complementary aspect: whether the filtered reasoning trace preserves the LRM's ability to produce correct answers. This evaluation targets *task-level accuracy*, which is distinct from hallucination detection. Concretely, for each reasoning trace generated by Qwen3-8B on MATH, we remove the steps identified as noisy by REDE and regenerate the final answer conditioned on the retained steps. Task accuracy is evaluated using the same judge model as in our main experiments. After filtering, the reasoning traces are reduced by approximately 70% in length. Despite this substantial reduction, the filtered traces achieve an accuracy of 80.00%, compared to 81.43% for the original traces, maintaining a comparable level of task performance. This confirms that the vast majority of discarded steps are uninformative for solving the task, and that REDE's filtering preserves the reasoning capability of the LRM even with a significantly compressed trace.

C.10 RESULTS WITH ADDITIONAL METRICS

Our main experiments use AUROC as the primary evaluation metric, as it measures discrimination ability across all decision thresholds. To provide a more complete evaluation, we additionally report Accuracy and F1 score. As shown in Table 27, using REDE-Filtered steps consistently improves over using the Original reasoning trace across all three metrics for both CCS (Burns et al., 2023) and probing (Azaria & Mitchell, 2023). Notably, beyond the substantial gains in AUROC, REDE also achieves strong improvements in ACC and F1, further confirming the effectiveness of reasoning-step denoising.

Table 27: **Hallucination detection results under additional evaluation metrics on TruthfulQA with Qwen3-8B.** We compare using the Original reasoning trace versus REDE-Filtered steps. The best result in each group is highlighted in **bold**.

Setting	CCS			Probing		
	AUROC	ACC	F1	AUROC	ACC	F1
Original	68.63	58.91	65.03	80.42	66.57	63.08
Filtered	**87.32**	**70.47**	**73.95**	**86.44**	**71.09**	**68.82**