RFS-Guard: Detecting Reasoning Hallucinations via Cross-Phase Routing Focus in Large Reasoning Models

Zihang Liu[1], Zhouhua Fang[2], Hui Liu[2], Zhiwei Liu[2], Yong Li[2,*], Haishuai Wang[1,*]
[1]Zhejiang Key Laboratory of Accessible Perception and Intelligent Systems,
College of Computer Science and Technology, Zhejiang University
[2]Ant Group

Abstract

Large reasoning models (LRMs) achieve strong performance on complex tasks by generating intermediate reasoning before the final answer, yet they remain prone to reasoning hallucinations such as subtle arithmetic or constraint-violation errors. Prior hallucination detectors often rely on external verification or local token-level signals, which are limited for LRMs and largely overlook whether the cross-phase information flow from reasoning to answering is structurally robust. We propose Routing Focus Score (RFS), a step-level indicator that measures how strongly cross-step attention routing aligns with semantic proximity derived from hidden-state cosine similarity. We further design RFS-Guard, a lightweight hallucination detection framework based on RFS. Empirically, we observe that higher reasoning–answer RFS is consistently associated with higher hallucination risk, suggesting a routing-collapse failure mode where models might prefer self-confirmation loops and suppress the ability to audit their own generations. Experimental results across multiple domains and models demonstrate the superiority of RFS-Guard for detecting and localizing hallucinations in LRMs without requiring external tools or repeated sampling.

1 Introduction

Large Reasoning Models (LRMs, DeepSeek-AI, 2025; Team, 2025), typically trained with reinforcement learning or process-supervision signals, have demonstrated strong performance on multi-step tasks such as mathematical problem solving, code generation, and scientific question answering. By producing intermediate reasoning traces before the final answer, LRMs appear to offer improved compositional generalization and interpretability compared with standard instruction-tuned Large Language Models (LLMs, Wang et al., 2025).

*Corresponding authors:
liyong.liy@antgroup.com, haishuai.wang@zju.edu.cn

However, despite these advances, LRMs remain vulnerable to reasoning hallucinations (Wang et al., 2025): the model's intermediate steps and final response are fluent and seemingly coherent, yet contain invalid deductions or incorrect factual statements. In mathematical domains, this often manifests as subtle calculation errors that propagate through subsequent steps, yielding an answer that is stylistically confident but objectively wrong (Huang et al., 2025). Such errors are particularly challenging because they are harder to detect by surface-level plausibility and can degrade trust in safety-critical applications.

A substantial body of existing work has studied hallucination in LLMs (Rahman et al., 2026). They propose detectors based on external verification (Min et al., 2023; Wei et al., 2024), self-consistency sampling (Miao et al., 2024; Mündler et al., 2024), uncertainty estimation (Malinin and Gales, 2021a; Farquhar et al., 2024), or token-level signals (Malinin and Gales, 2021b; Kadavath et al., 2022). Nonetheless, these methods are limited when applied to LRMs. First, many LLM-oriented approaches implicitly assume short-form generations and are not available for the structured, multi-stage reasoning characteristic of LRMs. Second, verification-heavy solutions (e.g., tool use, retrieval, or ensemble sampling) can be computationally expensive and may not capture the position where an error emerges within a multi-step reasoning chain. Third, token-level uncertainty or probability heuristics can be misleading in LRMs: a model may assign high likelihood to a plausible continuation even when it has already deviated from the correct computational path (Yao et al., 2025).

Meanwhile, existing studies specifically targeting LRMs often focus on reasoning trajectories. For example, RHD (Sun et al., 2026), CoRE-Eval (Li et al., 2025a), and G-Detector (Zhang et al., 2026) track the models' state (thinking depth or logic) during reasoning and reveal underlying pat-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 19371–19385

tern shifts for hallucination detection. Wang et al. (2026) and Xiong et al. (2025) uncover the unfaithfulness or inconsistencies of reasoning and answer procedures for hallucinations. However, few of the existing methods explicitly ask whether the information flow between reasoning and answering is robust. Intuitively, a correct solution requires the final answer to be grounded in the right evidence and intermediate thoughts. A model may fail when its internal routing mechanism prioritizes semantical continuations over evidence-critical dependencies, allowing early mistakes to be concealed by coherent narration (Wang et al., 2025).

Based on this insight, we introduce RFS-Guard, a lightweight hallucination detection method for open-source LRMs that uses the Routing Focus Score (RFS) as its core signal. RFS quantifies the degree to which attention routing between thinking and answer steps collapses into simple semantic proximities. RFS-Guard further enhances RFS with backtracking-based information flow and efficient semantic modeling to improve detection effectiveness and efficiency. Experiments across multiple domains and modern LRMs demonstrate that our method achieves state-of-the-art hallucination detection and localization performance while maintaining high efficiency. Our contributions can be summarized as follows:

- We propose RFS, a step-level measure that aligns attention-based routing with semantic proximity, capturing routing collapses that correlate with reasoning hallucinations.
- We present RFS-Guard, which models the multi-hop information flow across steps and leverages RFS to localize potential hallucinations, enabling effective and fine-grained hallucination detection.
- We conduct comprehensive experiments across multiple domains and modern LRMs, demonstrating the effectiveness and efficiency of RFS-Guard to identify and localize reasoning hallucinations.

2 Preliminaries

2.1 Task Formulation

Given a question Q, the response generated by an LRM consists of a reasoning phase R and an answer A, denoted as (Q, R, A). Following Sun et al. (2026), we split R into reasoning steps $s_0^r, s_1^r, \cdots, s_{N_R-1}^r$ and split A into answer steps $s_0^a, s_1^a, \cdots, s_{N_A-1}^a$. N_R and N_A denote the number of reasoning and answer steps, respectively. Each step s_i comprises a token sequence $c_{i,0}, c_{i,1}, \cdots, c_{i,n_i-1}$, where n_i is the length of s_i.

Hallucination Detection determines whether the answer A is hallucinated, given Q and (R, A) generated by model $\mathcal{M}$, i.e., $f : (Q, R, A, \mathcal{M}) \to Y$.

Hallucination Localization identifies hallucination at the step level, classifying each step s_i as hallucinated or not, i.e., $f : (Q, R, A, \mathcal{M}, s_i) \to y_i$.

2.2 Inner-state of Transformer-based Models

Transformer-based language models consist of a stack of transformer blocks, each comprising multi-head self-attention and a position-wise feed-forward network (Vaswani et al., 2017). Self-attention assigns weights that determine how strongly each token attends to other tokens in the input sequence. Given tokens $c_0, c_1, \cdots, c_{n-1}$, let $\mathbf{w}_{i,j}^{l,h}$ be the contribution of token c_j to the updated representation of token c_i (typically $j \leq i$) at layer l and head h. The hidden representation of token c_i after layer l is denoted by $\mathbf{z}_i^l \in \mathbb{R}^d$, where d is the model's hidden dimension.

3 Empirical Study on Cross-Phase Routing Focus

Our empirical study investigates the relationship between the reasoning phase and the answer phase of open-source LRMs. The study reveals that the over-focus phenomenon of attention patterns between reasoning and answer steps is a strong indication of reasoning hallucinations.

Attention Correlation quantifies the association between reasoning and answer steps using the model's self-attention. Prior work shows that attention weights can capture long-range dependencies between tokens (Sun et al., 2025; Chen et al., 2025; Qiu et al., 2025). We extend this idea from token-level to step-level links by aggregating token-wise attention within each step. Let $\mathbf{W}^{l,h}$ denote the step-level attention at layer l and head h:

$$\mathbf{W}_{i,j}^{l,h} = \mathrm{StepPool}_{c_{i'} \in s_i} \left(\mathrm{StepPool}_{c_{j'} \in s_j} (\mathbf{w}_{i',j'}^{l,h}) \right), \tag{1}$$

where $\mathrm{StepPool}$ is a pooling operator (e.g., mean). $\mathbf{w}_{i',j'}^{l,h}$ is the attention weight from token $c_{i'}$ to $c_{j'}$.

In multi-head self-attention layers, different heads encode different attentive patterns. To iden-

tify the most informative heads for discerning semantically significant steps, we select informative heads using entropy. Given matrix $\mathbf{W}^{l,h}$, we compute $H^{l,h} = -\sum_{i,j}\left(\mathbf{W}^{l,h}_{i,j}\log \mathbf{W}^{l,h}_{i,j}\right)$. A lower entropy indicates a sharper attention distribution and a more concentrated, informative attention head (Li et al., 2025b). Then, we dynamically choose the k_1 attention heads with the smallest entropy for each sample, i.e.,

$$\mathcal{H} = \text{argmin}^{(l,h)}_{k_1} H^{l,h}. \tag{2}$$

We then aggregate the selected heads to obtain a single step-level attention matrix, i.e.,

$$\hat{\mathbf{W}} = \text{HeadPool}_{(l,h)\in\mathcal{H}}\mathbf{W}^{l,h}, \tag{3}$$

which captures directional attention across steps. HeadPool is a pooling operator (e.g., mean). Finally, following Li et al. (2025b), we adjust positional biases to reduce recency effects and normalize the scores to obtain $\mathbf{W} \in \mathbb{R}^{(N_R+N_A)\times(N_R+N_A)}$.

Semantic Correlation captures the semantic alignment between reasoning steps and answer steps. Reasoning traces generated by LRMs often contain exploratory, speculative, or redundant content that does not directly contribute to the final answer (Wang et al., 2026). To reduce this noise, we use Qwen3-235B-A22B-Instruct-2507 (Qwen3-235B, Team, 2025) as an annotator: for each answer step $s_i \in A$, it selects the subset of reasoning steps in R that are directly relevant. We then encode these selections as multi-hot vectors to form a semantic correlation matrix $\mathbf{M} \in \mathbb{R}^{N_A\times N_R}$:

$$\begin{aligned} R^{\text{LLM}}_i &= \text{LLM}(\mathcal{P}, Q, R, s_i) \subseteq R, \\ \mathbf{M}_i &= \text{MultiHot}(R^{\text{LLM}}_i), \end{aligned} \tag{4}$$

where $\mathcal{P}$ is the prompt shown in Appendix G and MultiHot maps the selected reasoning-step indices into a multi-hot vector of length N_R.

Routing Focus Score measures how well attention-based correlations align with semantic correlations. Specifically, for each answer step $s_i \in A$, we compare the attention vector $\mathbf{W}_i$ with the semantic vector $\mathbf{M}_i$ by quantifying the accumulated attention weights of semantically relevant entries and obtain a routing focus score:

$$\begin{aligned} \text{RFS}(\mathbf{W}_i, \mathbf{M}_i) &= \sum_{j\in\mathcal{I}(\mathbf{M}_i)} \mathbf{W}_{i,j}, \\ \mathcal{I}(\mathbf{M}_i) &= \{j | \mathbf{M}_{i,j} > \phi\}, \end{aligned} \tag{5}$$

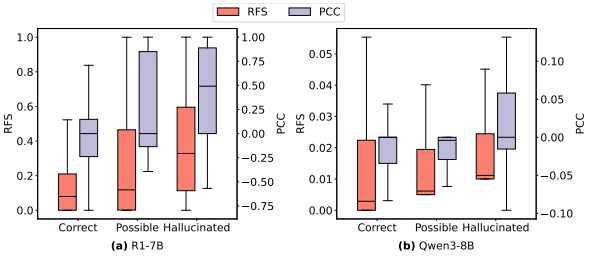

Figure 1: Routing focus score of steps with different categories.

where ϕ thresholds semantic relevance. Larger RFS indicates a stronger focus of attention map on semantically relevant steps.

Experimental Settings. We validate the effectiveness of the routing focus score on a composite mathematical dataset, which includes MATH500, AIME25, and minervamath, with DeepSeek-R1-Distill-Qwen-7B (R1-7B, DeepSeek-AI, 2025) and Qwen3-8B (Team, 2025) models. We first sample model solutions and retain those with an incorrect final answer as hallucinated generations. We then label the correctness of each answer step using a strong annotator model (Qwen3-235B). To reduce single-run variance and prompt sensitivity, we run the judge 5 times independently with an instruction shown in Appendix G. Based on how frequently a step is judged incorrect, we partition answer steps into three groups: (i) *hallucinated* (marked incorrect ≥ 3 times), (ii) *possible-hallucinated* (marked incorrect 1-2 times), and (iii) *correct* (never marked incorrect).

Across 128 (R1-7B) and 30 (Qwen3-8B) hallucinated solutions, we obtain 356/59 hallucinated steps, 176/66 possible-hallucinated steps, and 726/366 correct steps, respectively. To assess label quality and mitigate bias from relying on a single judge model, we additionally perform a human spot-check. Three postgraduate annotators independently audit these steps using the same instructions, without access to the judge model's outputs. The majority vote is conducted when annotators disagree. We report 86% agreement between the model-based labels and the human annotations. In addition, among 100 randomly sampled hallucinated steps, 92% stem from concrete arithmetic mistakes (e.g., incorrect operations on specific numbers) or explicit constraint violations.

Results. Figure 1 reports the step-level score distributions computed with either RFS or Pearson Correlation Coefficient (PCC) as the scoring func-

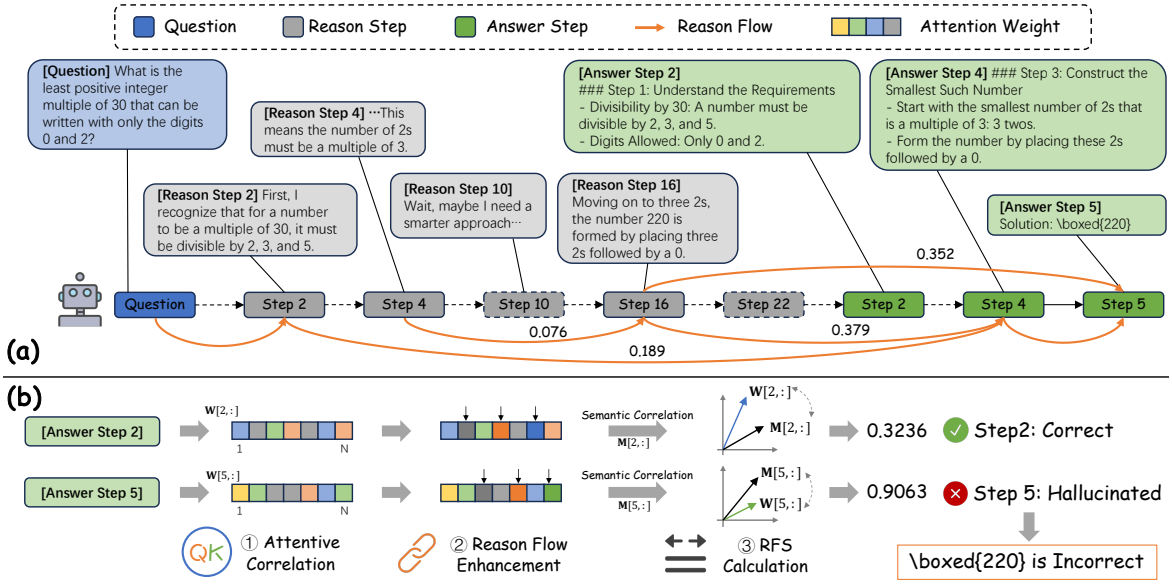

Figure 2: Framework overview. **(a)** Illustration of a reason flow. **(b)** Schema of RFS-Guard to detect and locate hallucinations. Arrows point from source steps to destination steps with attention weights.

tion. Both function choices consistently differentiate correct from hallucinated steps: correct steps exhibit a substantially lower score than hallucinated steps. One-sided Mann-Whitney U tests confirm that the correct-step scores are significantly smaller than the hallucinated-steps (R1-7B: u=9610/p=1.5e-16 for RFS and u=10040/p=2.9e-15 for PCC. Qwen3-8B: u=6135/p=9.7e-10 for RFS and u=8230/p=4.6e-3 for PCC). We provide a closer look at a boundary case of valid repetition in Appendix D.

These results suggest that RFS, which captures the alignment between attention-based routing and semantic similarity, serves as an effective indicator of step-level hallucination in open-source LRMs. Steps with a higher score exhibit a markedly higher risk of hallucination, while a lower score is associated with evidence-grounded transitions and logical deduction. We owe that higher RFS might reflect a collapse phenomenon in the attention pattern, where attention increasingly follows superficial semantic neighbors rather than selectively retrieving task-critical evidence (e.g., numeric quantities or constraints) from the context.

A higher RFS suggests that the model might enter a *self-confirmation loop*: the attention mechanism majorly retrieves information from similar reasoning steps, making the representations of the reasoning and answering phases increasingly similar. As a result, the answer behaves more like a semantic paraphrase of the ongoing reasoning trajectory, rather than an independent verification process. Once a local error occurs in reasoning (e.g., an arithmetic slip), the answer is more likely to expand along the same erroneous semantic path, producing outputs that appear well-structured yet are factually incorrect.

This collapse might also lead to *missing corrective jumps*. Effective error correction typically requires attention to make counter-intuitive transitions, i.e., jumping away from the current semantic neighborhood to retrieve less similar but decision-critical evidence, such as rereading the problem statement or revisiting a key intermediate quantity. When routing is dominated by semantic proximity (high RFS), these cross-semantic jumps are suppressed, making early mistakes more likely to propagate through subsequent steps (Zhang et al., 2024a) and ultimately contaminate the final answer.

4 Methodology

Building on the findings in Section 3, we introduce RFS-Guard, a training-free and lightweight hallucination detection algorithm for open-source LRMs. It consists of three components: (1) Attention Correlation Module extracts cross-phase attention patterns between reasoning and answer steps, (2) Reason Flow Module refines the attention to better approximate multi-hop information propagation along the reasoning flow, and (3) Routing Focus Scoring Module quantifies the degree of routing focus. Figure 2 shows the overview.

4.1 Obtaining Attention Correlation Map

The attention correlation map is designed to expose how information is routed between the reasoning phase and the final answer. A direct use of raw attention (as discussed in Section 3) is often insufficient for LRMs because it primarily captures one-hop step-to-step dependencies and thus underestimates the multi-hop information flows that arise in long-form reasoning.

Specifically, the enhanced robustness of LRMs on complex problem-solving scenarios is closely attributed to richer reasoning behaviors, such as exploration of alternative solution paths and self-verification of intermediate conclusions (Wang et al., 2025). These behaviors make the dependency structure inherently non-local: an answer step may be grounded in evidence distributed across multiple earlier reasoning steps. Consequently, a faithful representation of reasoning-to-answer routing requires aggregating and propagating attention beyond immediate links.

To this end, we construct an enhanced attention correlation map by constructing a compact and semantically relevant *reason flow* to precisely track the information flow from each intermediate reasoning step to the final answer step, which then supports the subsequent routing focus scoring stages.

Reasoning Flow. Our goal is to trace how the final answer A is supported by intermediate reasoning steps R. We assume that each answer step s_i depends on only a small subset of influential prior steps, while many other steps (e.g., misguided exploration or redundant self-verification) contribute little. To emphasize the truly supportive steps, we adapt the backtracking mechanism of Li et al. (2025b) to construct a compact reasoning flow that approximates the dominant information flow into each answer step $s_i \in A$.

Let $\mathbf{W}$ denote the step-level attention matrix from Equation 3. Starting from $\mathcal{S}^0_{src} = \{s_i\}$, at iteration p we collect candidate links from each current source step $s_j \in \mathcal{S}^p_{src}$ to its top-k_2 most attended predecessors:

$$\hat{\mathcal{L}}^p = \{(s_k, s_j) \mid s_k \in \text{argmax}_{k_2}(\mathbf{W}_j), s_j \in \mathcal{S}^p_{src}\}. \tag{6}$$

We then score each candidate predecessor s_k by its accumulated influence:

$$\theta(s_k) = \sum_{(s_k,s_j)\in\hat{\mathcal{L}}^p} \mathbf{W}_{j,k}, \tag{7}$$

and keep the top-k_3 predecessors with $\theta(s_k) > \theta_{\text{min}}$:

$$\mathcal{S}^p_{dst} = \text{argmax}_{k_3}\left(\{s_k : \theta(s_k) > \theta_{\text{min}}\}\right), \quad \mathcal{L}^p = \{(s_k, s_j) \in \hat{\mathcal{L}}^p \mid s_k \in \mathcal{S}^p_{dst}\}. \tag{8}$$

We set $\mathcal{S}^{p+1}_{src} \leftarrow \mathcal{S}^p_{dst}$ and iterate until reaching the input prompt or a maximum depth. The resulting reason flow for s_i is $\mathcal{C}_i = \{\mathcal{L}^p_i\}_{p=1}^{P_i}$, where P_i is the number of iterations conducted.

Attention Enhancement. Given $\mathcal{C}_i$, we calibrate $\mathbf{W}$ to reflect multi-hop support from earlier steps. For each iteration p, we propagate attention along links in $\mathcal{L}^p$, i.e.,

$$\mathbf{E}^p_{i,k} = \begin{cases} \mathbf{W}_{i,j} \cdot \mathbf{W}_{j,k}, & (s_k, s_j) \in \mathcal{L}^p_i, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

We aggregate these signals to obtain the calibrated attention vector for s_i with factor α:

$$\bar{\mathbf{W}}_i = \mathbf{W}_i + \sum_{p=1}^{P_i} \alpha_p \mathbf{E}^p_i, \tag{10}$$

and stack across steps to form the calibrated map $\bar{\mathbf{W}}$, which captures multi-hop correlations between all reasoning and answer step pairs.

4.2 Obtaining Semantic Correlation Map

The semantic correlation map captures step-level semantic relatedness (e.g., shared equations, quantities, or atomic facts) between reasoning and answer steps. While this map could be obtained via LLM-based or manual annotations, such approaches are typically costly and difficult to scale.

Instead, we approximate semantic correlation using embedding similarity. Concretely, we represent each step s_j by aggregating token hidden states from transformer layer l:

$$\mathbf{z}^l_{s_j} = \text{EmbPool}_{c\in s_j}(\mathbf{z}^l_c), \tag{11}$$

where $\mathbf{z}^l_c$ is the hidden state of token c at layer l and EmbPool is a pooling operator (e.g., mean). For a sample (Q, R, A), we compute the semantic correlation map $\mathbf{M}$ via cosine similarity between each answer step $s_i \in A$ and each reasoning step $s_j \in R$, i.e., $\mathbf{M}_{i,j} = \text{cosine}\left(\mathbf{z}^l_{s_i}, \mathbf{z}^l_{s_j}\right)$.

4.2.1 Measuring Routing Focus Score

For a generation consisting of reasoning steps R and answer steps A, the calibrated attention map $\bar{\mathbf{W}}$ encodes how information is routed from R to each $s_i \in A$, while the semantic map $\mathbf{M}$ provides semantic proximity between these steps.

Category	Method	R1-7B			R1-14B			Qwen3-8B			Qwen3-14B		
		AUROC	AUPRC	F1	AUROC	AUPRC	F1	AUROC	AUPRC	F1	AUROC	AUPRC	F1
MATH													
Rule	LengthScore	0.5589	0.5420	0.5057	0.4943	0.5084	0.4795	0.5462	0.5710	0.5439	0.5045	0.5267	0.5333
Ensemble	SINdex	0.5362	0.5534	0.5747	0.5238	0.5271	0.5728	0.5248	0.5262	0.6142	0.5172	0.5356	0.3488
	SelfCheckGPT	0.5274	0.5390	0.5321	0.5922	0.5710	0.0800	0.5423	0.5463	0.5935	0.5286	0.5235	0.6585
	RACE	0.5510	0.5537	0.5806	0.5150	0.4982	0.5208	0.5277	0.5398	0.5873	0.4983	0.5570	0.5985
Uncertainty	P(True)	0.5492	0.5461	0.6497	0.5166	0.5474	0.3830	0.4967	0.5261	0.5071	0.5371	0.5275	0.5821
	LNPE	0.5154	0.5148	0.6468	0.5129	0.5545	0.5674	0.5282	0.5549	0.5839	0.5069	0.5437	0.6415
	PPL	0.5412	0.5391	0.6052	0.5351	0.5635	0.5728	0.5645	0.5850	0.5049	0.4340	0.4909	0.6323
	CCP	0.4298	0.4467	0.6436	0.5561	0.5342	0.6441	0.4623	0.5001	0.6411	0.4494	0.4533	0.6588
Self-Aware	EigenScore	0.5325	0.5315	0.6443	0.5418	0.5307	0.5445	0.5558	0.5623	0.5983	0.5380	0.5426	0.6015
	AttentionScore	0.5393	0.5430	0.6299	0.5174	0.5222	0.6009	0.5392	0.5461	0.5920	0.5015	0.4952	0.6456
	UQAC	0.6033	0.5932	0.6099	0.5225	0.5367	0.6102	0.5377	0.5512	0.6230	0.4641	0.5034	0.6065
	RHD	0.5659	0.5730	0.6313	0.5686	0.5251	0.5697	0.5788	0.5535	0.6067	0.5380	0.5702	0.6116
	RFS-Guard	**0.6370**	**0.6215**	**0.6666**	**0.6404**	**0.6257**	**0.6520**	**0.6115**	**0.6187**	**0.6579**	**0.6224**	**0.5771**	**0.6626**
	Improvement	5.59%	4.77%	2.60%	8.14%	9.58%	1.23%	5.65%	5.76%	2.62%	15.69%	1.21%	0.58%
Science													
Rule	LengthScore	0.4935	0.5013	0.4723	0.5763	0.5308	0.6160	0.5287	0.5341	0.4286	0.5122	0.4990	0.5150
Ensemble	SINdex	0.5031	0.5290	0.5053	0.5234	0.5233	0.6055	0.5164	0.4984	0.5176	0.5246	0.5204	0.5057
	SelfCheckGPT	0.5744	0.5954	0.4267	0.5131	0.5174	0.5714	0.4967	0.5155	0.0465	0.5304	0.5185	0.3704
	RACE	0.5250	0.5435	0.5111	0.5479	0.5374	0.5550	0.4990	0.4981	0.5699	0.5383	0.5244	0.5700
Uncertainty	P(True)	0.5347	0.5275	0.6567	0.5122	0.5033	0.5116	0.4942	0.5046	0.5771	0.5365	0.5254	0.5591
	LNPE	0.5050	0.5249	0.5200	0.5234	0.5113	0.5376	0.5511	0.5382	0.6449	0.5193	0.5624	0.6250
	PPL	0.5532	0.5777	0.6154	0.4801	0.5641	0.4561	0.5279	0.5429	0.6584	0.5314	0.5470	0.6420
	CCP	0.5737	0.5634	0.5455	0.5091	0.5460	0.6415	0.4909	0.4797	0.6612	0.4669	0.4661	0.6667
Self-Aware	EigenScore	0.5139	0.5376	0.6478	0.5251	0.5113	0.6372	0.4997	0.5141	0.5729	0.5040	0.5024	0.5755
	AttentionScore	0.5803	0.5936	0.6019	0.5153	0.5154	0.6195	0.5199	0.5179	0.6449	0.4742	0.4879	0.6475
	UQAC	0.4381	0.4567	0.6333	0.5016	0.4900	0.6213	0.4387	0.4776	0.6553	0.5247	0.5429	0.5579
	RHD	0.6575	0.6186	0.6588	0.6094	0.5697	0.6344	0.5136	0.5043	0.5952	0.5577	0.5293	0.5943
	RFS-Guard	**0.6738**	**0.6222**	**0.6670**	**0.6511**	**0.5805**	**0.6490**	**0.5760**	**0.5619**	**0.6667**	**0.5860**	**0.5848**	**0.6720**
	Improvement	2.48%	0.58%	1.24%	6.84%	1.90%	1.17%	4.52%	3.50%	0.83%	5.07%	3.98%	0.79%
MultiHopQA													
Rule	LengthScore	0.4801	0.4908	0.4520	0.5404	0.5311	0.4295	0.5237	0.5122	0.4762	0.6361	0.5854	0.5325
Ensemble	SINdex	0.5321	0.5316	0.4466	0.5296	0.5294	0.5707	0.5261	0.5218	0.5010	0.5232	0.5287	0.5629
	SelfCheckGPT	0.5527	0.5416	0.5796	0.5896	0.5939	0.6194	0.5761	0.5913	0.5069	0.5317	0.5144	0.6452
	RACE	0.5293	0.5342	0.4706	0.5513	0.5484	0.6071	0.5530	0.5287	0.5079	0.5413	0.5392	0.5081
Uncertainty	P(True)	0.5023	0.5088	0.6504	0.5211	0.5332	0.4809	0.5034	0.4955	0.6469	0.4928	0.5129	0.3605
	LNPE	0.5471	0.5136	0.6505	0.5352	0.5150	0.6347	0.6170	0.5842	0.6480	0.6351	0.5938	0.6247
	PPL	0.5090	0.5019	0.6344	0.4819	0.4971	0.5178	0.6341	0.5830	0.6489	0.6158	0.6241	0.6528
	CCP	0.5142	0.5053	0.4165	0.5569	0.5523	0.6178	0.5754	0.5407	0.6409	0.4065	0.4997	0.6560
Self-Aware	EigenScore	0.4863	0.5005	0.6576	0.4892	0.5033	0.6549	0.5345	0.5117	0.5759	0.5238	0.5231	0.6692
	AttentionScore	0.5548	0.5580	0.6435	0.5398	0.5329	0.6562	0.5335	0.5247	0.6493	0.5519	0.5243	0.6716
	UQAC	0.5563	0.5482	0.5913	0.4815	0.4906	0.6327	0.6247	0.5896	0.5706	0.6515	0.6532	0.5125
	RHD	0.5610	0.5567	0.4063	0.5381	0.5597	0.6555	0.5199	0.5451	0.6321	0.6288	0.6289	0.6188
	RFS-Guard	**0.6096**	**0.5996**	**0.6638**	**0.6173**	**0.6139**	**0.6568**	**0.6549**	**0.6248**	**0.6624**	**0.6680**	**0.6863**	**0.6849**
	Improvement	8.66%	7.46%	0.94%	4.70%	3.37%	0.09%	3.28%	5.67%	2.02%	2.53%	5.07%	1.98%

Table 1: Hallucination detection results. **Bold**: best. Underlined: second best.

We compute the RFS (Equation 5) for each answer step by measuring their alignment, i.e., $y_i = \mathrm{RFS}(\bar{\mathbf{W}}_i, \mathbf{M}_i)$, and use y_i as the hallucination score of step s_i. To obtain an overall score for the entire sample, we take the maximum score over answer steps, i.e., $Y = \max_{s_i \in A} y_i$.

5 Experiments

5.1 Experimental Setup

Dataset Construction. Following Sun et al. (2026), we construct hallucination detection datasets by prompting LRMs to generate reasonings and answers to questions spanning Math, Science, and MultiHopQA. We use four strong open-source models: DeepSeek-R1-Distill-Qwen-7B (R1-7B), DeepSeek-R1-Distill-Qwen-14B (R1-14B, DeepSeek-AI, 2025), Qwen3-8B, and Qwen3-14B (Team, 2025). For each question, we generate 20 solutions from each model with temperature=0.7 and check their correctness with Qwen3-235B given ground-truths and prompts in Appendix G. We keep one correct and one hallucinated solution for each question for balanced datasets. We

report AUPRC↑, AUROC↑, and F1↑ scores to measure hallucination detection performance. Appendix B shows the statistics of the datasets.

Baselines. We include four categories of baselines. (1) **Rule**-based method (Zeng et al., 2025) directly uses length of the reasoning trace as its hallucination score. (2) **Ensemble**-based methods use sampling and voting for confidence estimation, including SelfCheckGPT (Manakul et al., 2023), SINdex (Abdaljalil et al., 2025), and RACE (Wang et al., 2026). (3) **Uncertainty**-based methods rely on logit distributions at the top layer, including P(True) (Kadavath et al., 2022), LNPE (Ren et al., 2023), Perplexity (PPL, Malinin and Gales, 2021b), and CCP (Fadeeva et al., 2024). (4) **Self-Aware**-based methods investigate specific patterns within hidden states or attention maps, including EigenScore (Chen et al., 2024a), AttentionScore (Sriramanan et al., 2024), UQAC (Li et al., 2025b), and RHD (Sun et al., 2026). More implementation setups are described in Appendix C.

5.2 Main Results

Table 1 reports the hallucination detection results. Overall, uncertainty-based methods outperform rule-based and ensemble-based baselines. We hypothesize this is because LRM generations are typically long, logically organized, and rhetorically convincing (Wang et al., 2025), making surface-level semantic comparison and voting-based aggregation less reliable. Self-Aware approaches are also competitive, suggesting that signals extracted from internal model states can effectively capture hallucination-related patterns. Across all settings, RFS-Guard achieves the best performance, improving over the strongest baseline by an average of 5.28%/2.74%/3.81% on MATH/Science/MultiHopQA, respectively. In particular, compared with RACE (Wang et al., 2026), which also evaluates reasoning-answer consistency, RFS-Guard yields gains of 16.63%/16.72%/21.09% on the three datasets.

5.3 Efficiency

Figure 3 and Appendix F.1 compare detection performance with inference time for representative methods. Ensemble-based approaches (e.g., RACE) require multiple forward passes to produce diverse generations, leading to substantial computational overhead. In contrast, uncertainty-based and self-aware-based methods are markedly more efficient because they rely on intrinsic model signals, such as hidden-state dynamics or attention patterns, without additional sampling. Compared with all baselines, RFS-Guard provides favorable accuracy-efficiency trade-offs, therefore, more suited for latency-sensitive or resource-constrained deployment.

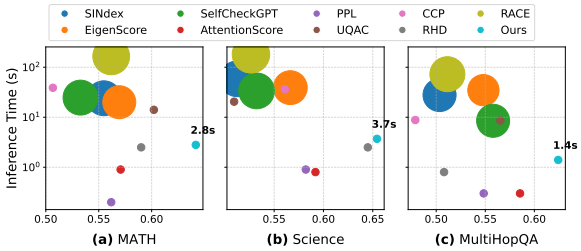

Figure 3: Hallucination detection efficiency. `x`: average performance. `y`: inference time per sample (logarithm). `Scale`: number of tokens cost. Bold numbers indicate the inference time of RFS-Guard.

5.4 Hallucination Localization

Beyond sample-level hallucination detection, we further evaluate whether methods can localize hallucinations within an answer. To construct step-level ground truth, we label the correctness of each answer step using Qwen3-235B and then manually verify the annotations, following the protocol in Section 3. We report AUROC and AUPRC over all answer steps, as well as Hit@k over samples, which measures whether the top-k highest-scored steps overlap with the ground-truth erroneous steps. To enable step-level evaluation, we adapt several baselines to output a binary score for each answer step, including LengthScore, SelfCheckGPT, P(True), PPL, CCP, EigenScore, and AttentionScore (details in Appendix C).

Table 2 summarizes the results. Among the baselines, SelfCheckGPT performs best overall, consistent with the effectiveness of step-wise checking via external evidence; however, it is also substantially less efficient, which limits practical use. Interestingly, LengthScore is competitive, supporting the intuition that longer reasoning traces tend to be more error-prone (Zeng et al., 2025). In contrast, uncertainty-based methods, which are strong at sample-level detection, degrade noticeably at the step level, suggesting that local uncertainty can be dominated by linguistic variability or syntactic ambiguity rather than factual correctness. RFS-Guard achieves the strongest step-level localization performance, likely because it explicitly models

Model	Method	MATH				Science				MultiHopQA			
		AUPRC	AUROC	H@1	H@3	AUPRC	AUROC	H@1	H@3	AUPRC	AUROC	H@1	H@3
R1-7B	LengthScore	0.3960	0.6231	0.4931	0.4398	0.6591	0.5404	0.8140	0.6589	0.9354	0.7855	0.9625	0.8222
	SelfCheckGPT	0.4175	**0.6452**	0.5317	0.4573	0.6872	0.5663	0.8437	0.6798	0.9346	0.8039	0.9708	0.8181
	P(True)	0.3227	0.5533	0.4097	0.3472	0.6579	0.5739	0.7907	0.6899	0.7979	0.4386	0.8250	0.7375
	PPL	0.2708	0.4812	0.3403	0.3356	0.6487	0.5638	0.7442	0.6512	0.7783	0.3926	0.7667	0.7208
	CCP	0.2437	0.4165	0.3056	0.2870	0.5939	0.4902	0.5814	0.5969	0.8434	0.5296	0.8917	0.7639
	EigenScore	0.3375	0.5858	0.3819	0.3773	0.5263	0.4005	0.4884	0.5504	0.8956	0.6654	0.9500	0.7986
	AttentionScore	0.2943	0.5204	0.3056	0.2986	0.6183	0.5043	0.6744	0.6202	0.8323	0.5288	0.8875	0.7736
	RFS-Guard	**0.4251**	0.6413	**0.5512**	**0.4671**	**0.7231**	**0.5939**	**0.8881**	**0.7227**	**0.9401**	**0.8181**	**0.9791**	**0.8240**
	Improvement	1.8%	-0.6%	3.7%	2.1%	5.2%	3.5%	5.3%	4.8%	0.5%	1.8%	0.9%	0.2%
Qwen3-8B	LengthScore	0.1547	0.5833	0.1909	0.1674	0.3578	0.6407	0.4018	0.3951	0.8761	0.5738	0.9686	0.7442
	SelfCheckGPT	0.1528	0.5651	0.1909	0.1630	0.3207	0.6007	0.3853	0.3588	0.9156	0.7225	0.9308	**0.8071**
	P(True)	0.1387	0.4904	0.2174	0.1304	0.2733	0.5842	0.2941	0.2500	0.7488	0.3496	0.7673	0.7191
	PPL	0.1133	0.4214	0.0870	0.1014	0.2505	0.5679	0.1176	0.1716	0.7901	0.4379	0.8553	0.7463
	CCP	0.1119	0.4753	0.0160	0.1159	0.1746	0.3583	0.1471	0.1569	0.8658	0.5804	0.9245	0.7254
	EigenScore	0.1466	0.5302	0.1739	0.1449	0.3330	0.6657	0.2941	0.3627	0.7504	0.4143	0.7736	0.7275
	AttentionScore	0.1143	0.4729	0.0870	0.1014	0.2861	0.6203	0.2647	0.3284	0.8934	0.6943	0.9686	0.7904
	RFS-Guard	**0.1671**	**0.5981**	**0.2231**	**0.1771**	**0.3819**	**0.6813**	**0.4281**	**0.4005**	**0.9356**	**0.7625**	**0.9723**	0.8060
	Improvement	8.0%	2.5%	2.6%	5.8%	6.7%	2.3%	6.5%	1.4%	2.2%	5.5%	0.4%	-0.1%

Table 2: Hallucination localization performance. **Bold**: best. Underlined: second best.

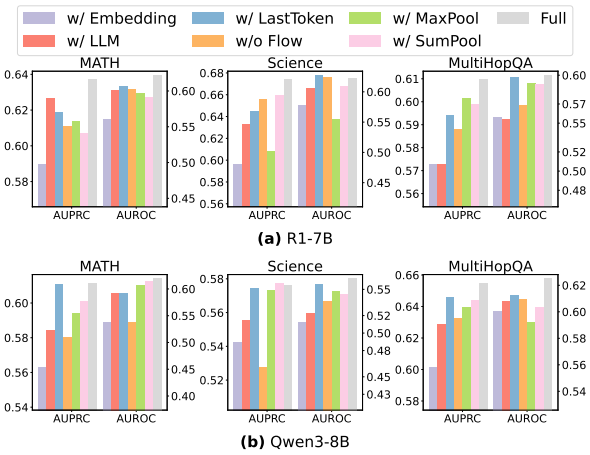

Figure 4: Ablation results.

multi-hop information flow to each answer step, enabling fine-grained identification of where attentions collapse into self-confirming routing.

5.5 Ablation Study

We perform an ablation study to quantify the contribution of each component in RFS-Guard. Concretely, we ablate (1) Semantic Correlation Module by using the last-token hidden state for EmbPool (*w/ LastToken*), Qwen3-Embedding-8B embeddings (*w/ Embedding*), or LLM-judged semantic correlations obtained by prompting Qwen3-235B (*w/ LLM*); and (2) the Attention Correlation Module by removing the reasoning-flow backtracking (*w/o Flow*) or altering token-to-step aggregation StepPool using max (*w/ MaxPool*) or sum (*w/ SumPool*). Figure 4 and Appendix F.2 report the results.

Replacing the semantic module with external models consistently degrades performance, which might be because embedding models are less sensitive to subtle differences in long reasoning steps. The reasoning flow is particularly important: removing it causes a clear drop in accuracy (averaging -2.5%/-4.6% for R1-7B/Qwen3-8B). The effect is more salient on datasets with longer traces (e.g., Qwen3-8B MATH and Science with ~100 steps on average), where the drop increases to -5.9%/-5.3%. This pattern highlights the necessity to model long-range, non-local information flow in LRMs for reliable hallucination detection. Finally, alternative aggregation strategies (*w/ LastToken*, *w/ MaxPool*, and *w/ SumPool*) yield smaller but consistent declines (-1.3%, -2.7%, and -1.9% on average) relative to the full RFS-Guard.

6 Conclusion

We propose RFS-Guard, a lightweight and training-free framework for hallucination detection in large reasoning models. It monitors cross-phase information flow from reasoning to answering via the Routing Focus Score (RFS), which quantifies the extent to which attention routing collapses toward semantic-neighbor connections. Experiments spanning domains and model families further show that RFS-Guard achieves state-of-the-art detection and localization performance while remaining computationally efficient. Overall, our results suggest that routing collapse provides a useful structural signal for diagnosing and localizing reasoning failures, helping improve the reliability of LRMs.

Limitations

RFS provides a correlational signal and does not, by itself, establish a causal relationship between routing collapse and hallucinations. Its performance may depend on the quality of step segmentation and may vary with model families, prompting styles, and decoding configurations. Moreover, RFS may also be useful beyond detection, serving as an optimization signal for hallucination mitigation (e.g., as a reward or constraint in reinforcement learning). We leave a systematic investigation of this direction to future work. Finally, RFS-Guard requires access to internal activations (attention and hidden states), which limits its applicability to closed-source systems exposed only through black-box APIs.

Acknowledgement

This work was supported by the National Key Research and Development Program of China (2025YFC2708700), Zhejiang Province Major Science and Technology Plan Project (2026C01021), the Central Government-Guided Fund for Local Science and Technology Development (2025ZY01034), the Provincial Key Laboratory Program (2025E10048), and Zhejiang Province Science and Technology Plan Project (2025C01128).

References

Samir Abdaljalil, Hasan Kurban, Parichit Sharma, Erchin Serpedin, and Rachad Atat. 2025. Sindex: Semantic inconsistency index for hallucination detection in llms. *CoRR*, abs/2503.05980.

Amos Azaria and Tom M. Mitchell. 2023. The internal state of an LLM knows when it's lying. In *EMNLP (Findings)*, Findings of ACL, pages 967–976. Association for Computational Linguistics.

Jakub Binkowski, Denis Janiak, Albert Sawczyn, Bogdan Gabrys, and Tomasz Kajdanowicz. 2025. Hallucination detection in llms using spectral features of attention maps. In *EMNLP*, pages 24354–24385. Association for Computational Linguistics.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024a. INSIDE: llms' internal states retain the power of hallucination detection. In *ICLR*. OpenReview.net.

Jianghao Chen, Junhong Wu, Yangyifan Xu, and Jiajun Zhang. 2025. LADM: long-context training data selection with attention-based dependency measurement for llms. In *ACL (1)*, pages 3076–3090. Association for Computational Linguistics.

Shiqi Chen, Miao Xiong, Junteng Liu, Zhengxuan Wu, Teng Xiao, Siyang Gao, and Junxian He. 2024b. In-context sharpness as alerts: An inner representation perspective for hallucination mitigation. In *ICML*, Proceedings of Machine Learning Research, pages 7553–7567. PMLR / OpenReview.net.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2024. Chain-of-verification reduces hallucination in large language models. In *ACL (Findings)*, Findings of ACL, pages 3563–3578. Association for Computational Linguistics.

Ekaterina Fadeeva, Aleksandr Rubashevskii, Artem Shelmanov, Sergey Petrakov, Haonan Li, Hamdy Mubarak, Evgenii Tsymbalov, Gleb Kuzmin, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, and Maxim Panov. 2024. Fact-checking the output of large language models via token-level uncertainty quantification. In *ACL (Findings)*, Findings of ACL, pages 9367–9385. Association for Computational Linguistics.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nat.*, 630(8017):625–630.

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

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.*, 43(2):42:1–42:55.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, and 17 others. 2022. Language models (mostly) know what they know. *CoRR*, abs/2207.05221.

Haoqiang Kang, Juntong Ni, and Huaxiu Yao. 2023. Ever: Mitigating hallucination in large language models through real-time verification and rectification. *CoRR*, abs/2311.09114.

Haoxi Li, Sikai Bai, Jie Zhang, and Song Guo. 2025a. Core: Enhancing metacognition with label-free self-evaluation in lrms. *CoRR*, abs/2507.06087.

Kenneth Li, Oam Patel, Fernanda B. Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *NeurIPS*.

Yinghao Li, Rushi Qiang, Lama Moukheiber, and Chao Zhang. 2025b. Language model uncertainty quantification with attention chain. *CoRR*, abs/2503.19168.

Zihang Liu, Jiawei Guo, Hao Zhang, Hongyang Chen, Jiajun Bu, and Haishuai Wang. 2025. Long-form hallucination detection with self-elicitation. In *ACL (Findings)*, Findings of ACL, pages 4082–4100. Association for Computational Linguistics.

Haolang Lu, Yilian Liu, Jingxin Xu, Guoshun Nan, Yuanlong Yu, Zhican Chen, and Kun Wang. 2025. Auditing meta-cognitive hallucinations in reasoning large language models. *CoRR*, abs/2505.13143.

Andrey Malinin and Mark J. F. Gales. 2021a. Uncertainty estimation in autoregressive structured prediction. In *ICLR*. OpenReview.net.

Andrey Malinin and Mark J. F. Gales. 2021b. Uncertainty estimation in autoregressive structured prediction. In *ICLR*. OpenReview.net.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. In *EMNLP*, pages 9004–9017. Association for Computational Linguistics.

Ning Miao, Yee Whye Teh, and Tom Rainforth. 2024. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. In *ICLR*. OpenReview.net.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. In *EMNLP*, pages 12076–12100. Association for Computational Linguistics.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin T. Vechev. 2024. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *ICLR*. OpenReview.net.

Yifu Qiu, Varun R. Embar, Yizhe Zhang, Navdeep Jaitly, Shay B. Cohen, and Benjamin Han. 2025. Eliciting in-context retrieval and reasoning for long-context large language models. In *ACL (Findings)*, Findings of ACL, pages 3176–3192. Association for Computational Linguistics.

Subhey Sadi Rahman, Md. Adnanul Islam, Md. Mahbub Alam, Musarrat Zeba, Md. Abdur Rahman, Sadia Sultana Chowa, Mohaimenul Azam Khan Raiaan, and Sami Azam. 2026. Hallucination to truth: a review of fact-checking and factuality evaluation in large language models. *Artif. Intell. Rev.*, 59(2):70.

Jie Ren, Jiaming Luo, Yao Zhao, Kundan Krishna, Mohammad Saleh, Balaji Lakshminarayanan, and Peter J. Liu. 2023. Out-of-distribution detection and selective generation for conditional language models. In *ICLR*. OpenReview.net.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. 2024. Llm-check: Investigating detection of hallucinations in large language models. In *NeurIPS*.

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

Zhongxiang Sun, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. 2025. Redeep: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability. In *ICLR*. OpenReview.net.

Qwen Team. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Changyue Wang, Weihang Su, Qingyao Ai, and Yiqun Liu. 2026. Joint evaluation of answer and reasoning consistency for hallucination detection in large reasoning models. In *AAAI*, pages 33377–33385. AAAI Press.

Lei Wang. 2025. Seredeep: Hallucination detection in retrieval-augmented models via semantic entropy and context-parameter fusion. *CoRR*, abs/2505.07528.

Yanbo Wang, Yongcan Yu, Jian Liang, and Ran He. 2025. A comprehensive survey on trustworthiness in reasoning with large language models. *CoRR*, abs/2509.03871.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. 2024. Long-form factuality in large language models. In *NeurIPS*.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2025. Retrieval head mechanistically explains long-context factuality. In *ICLR*. OpenReview.net.

Zidi Xiong, Shan Chen, Zhenting Qi, and Himabindu Lakkaraju. 2025. Measuring the faithfulness of thinking drafts in large reasoning models. *CoRR*, abs/2505.13774.

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

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

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Yunhua Zhou, and Xipeng Qiu. 2025. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? In *ACL (1)*, pages 4651–4665. Association for Computational Linguistics.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. 2025a. Reasoning models know when they're right: Probing hidden states for self-verification. *CoRR*, abs/2504.05419.

Guibin Zhang, Yuxiang Zhang, Moayad Aloqaily, Haolang Lu, Kun Wang, Jing Liang, Xingjun Ma, Yu-Gang Jiang, and Qingsong Wen. 2026. Unraveling hallucination in large reasoning models: A topological perspective.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. 2024a. How language model hallucinations can snowball. In *ICML*, Proceedings of Machine Learning Research, pages 59670–59684. PMLR / OpenReview.net.

Wenqi Zhang, Yongliang Shen, Linjuan Wu, Qiuying Peng, Jun Wang, Yueting Zhuang, and Weiming Lu. 2024b. Self-contrast: Better reflection through inconsistent solving perspectives. In *ACL (1)*, pages 3602–3622. Association for Computational Linguistics.

Zhenliang Zhang, Xinyu Hu, Huixuan Zhang, Junzhe Zhang, and Xiaojun Wan. 2025b. ICR probe: Tracking hidden state dynamics for reliable hallucination detection in llms. In *ACL (1)*, pages 17986–18002. Association for Computational Linguistics.

Xinran Zhao, Hongming Zhang, Xiaoman Pan, Wenlin Yao, Dong Yu, Tongshuang Wu, and Jianshu Chen. 2024. Fact-and-reflection (far) improves confidence calibration of large language models. In *ACL (Findings)*, Findings of ACL, pages 8702–8718. Association for Computational Linguistics.

A Related Works

A.1 Hallucination in Large Language Models

The leading strategies for hallucination in LLMs can be broadly categorized into two complementary directions: looking outward to ground the models' generation process in external, verifiable knowledge, and looking inward to the models' own internal states for signals of uncertainty and failure.

Model-based strategies treat the LLM as a self-reflective agent capable of verifying its own outputs through techniques. For example, Zhang et al. (2024b); Zhao et al. (2024); Liu et al. (2025) ask the model to iteratively generate and critique its reasoning. Chain-of-verification (Miao et al., 2024; Mündler et al., 2024; Dhuliawala et al., 2024) structure self-correction via planned verification questions. Retrieval-augmented verification (Kang et al., 2023; Min et al., 2023; Wei et al., 2024) utilizes retrieved information from external databases to verify the accuracy of the models' generated content. Complementarily, inner-state-based methods adopt a "white-box" perspective by analyzing internal model signals during generation, for example, uncertainty quantification via probability-based metrics (Malinin and Gales, 2021a; Farquhar et al., 2024; Kadavath et al., 2022). More advanced probing techniques examine hidden states (Azaria and Mitchell, 2023; Li et al., 2023), attention patterns (Wu et al., 2025; Binkowski et al., 2025), or layer-wise dynamics (Sun et al., 2025; Wang, 2025). For example, using eigen values (Chen et al., 2024a) from response embeddings, sharpness of hidden state distributions (Chen et al., 2024b), spectral features of attention maps (Binkowski et al., 2025), or tracking feed-forward versus attention contributions (Zhang et al., 2025b) to identify subtle indicators of hallucination. Together, these approaches aim to enhance the reliability and trustworthiness of LLM-generated content, yet they are less effective for large reasoning models.

A.2 Hallucination in Large Reasoning Models

Hallucination in LLMs refers to model generating content that is superficially coherent and plausible but inconsistent with the context or factuality (Wang et al., 2025). Since LRMs often generate content that is more logically coherent and persuasive, hallucinations are prone to spread (Zhang et al., 2024a) and harder for users to identify, especially in high-stake senarios such as law and healthcare. For example, Zhang et al. (2025a) adopts

linear probing to identify errors early in reasoning. HARP (Hu et al., 2026) decomposes LLM hidden states into semantic and reasoning subspaces using SVD on the unembedding layer, then projects hidden states onto the reasoning subspace. RACE (Wang et al., 2026) jointly evaluates intra and inter consistencies of reasoning and answer procedures. Xiong et al. (2025) proposes to evaluate the faithfulness of thinking drafts along intra-draft and draft-to-answer faithfulness. RHD (Sun et al., 2026) introduced reasoning scores based on divergence between intermediate hidden states and final logits. Lu et al. (2025) reveals that reasoning LLMs amplify hallucinations through overconfident, prompt-aligned reflection in long chain-of-thought reasoning, making errors persistent and hard to detect or correct. HalluGuard (Zeng et al., 2026) jointly identifies both data-driven (from flawed training knowledge) and reasoning-driven (from unstable inference dynamics) hallucinations by leveraging Neural Tangent Kernel-induced geometry and representation stability. CoRE-Eval (Li et al., 2025a) and G-Detector (Zhang et al., 2026) leverage geometric analysis of latent reasoning trajectories to detect redundant thinking or hallucinations, improving both efficiency and accuracy in large reasoning models.

B Datasets

For the MATH dataset, we include questions from MATH-500[1], AIME25[2], and minervamath[3] datasets. For the Science dataset, we include questions from GPQA-extended[4] dataset. For the MultiHopQA dataset, we include questions from HotpotQA[5], 2WikiMultihopQA[6], MuSiQue[7], and bamboogle[8] datasets.

Table 3 shows the statistics of the datasets. To ensure stability, we sample responses to construct the datasets offline. During hallucination detection inference, we subsequently employ the same model to encode the token sequences of these responses, thereby preserving consistency between the model's state at inference and its state during response generation. During dataset generation, we only keep solutions with no more than 200 steps, and each step has a maximum token length of 512.

[1] https://huggingface.co/datasets/HuggingFaceH4/MATH-500

[2] https://huggingface.co/datasets/math-ai/aime25

[3] https://huggingface.co/datasets/math-ai/minervamath

[4] https://huggingface.co/datasets/Idavidrein/gpqa

[5] https://huggingface.co/datasets/hotpotqa/hotpot_qa

[6] https://huggingface.co/datasets/voidful/2WikiMultihopQA

[7] https://huggingface.co/datasets/dgslibisey/MuSiQue

[8] https://huggingface.co/datasets/chiayewken/bamboogle

C Implementation Details

Hyperparameters. We implement StepPool, HeadPool, and EmbPool with the mean operation. Following Li et al. (2025b), all stop tokens (e.g., pronouns) are skipped. k_2 in Equation 6 is set to 3 and k_3 in Equation 8 is set to 5, and minimal attention weight θ_{min} is 0.1. k_1 in Equation 3 is set to 8 for R1-7B and Qwen-8B and 16 for R1-14B and Qwen3-14B. We set α_p in Equation 10 to 1 for all iterations. ϕ in Equation 5 is set to 0.9. We search for the best layer index l individually for each model in Equation 11 from $\{0, 1, N/2, N-1\}$, where N is the number of hidden layers for the model, and set $l = 1$ across all our experiments. This configuration consistently yields competitive results across most settings (shown in Appendix E.1), though task-specific tuning may offer marginal gains.

Software and Hardware. Experiments are conducted on Linux machines with Python 3.10.15, PyTorch 2.4.0, Transformers 4.51.3, VLLM 0.11.0, and CUDA 12.4, equipped with Intel Xeon Platinum 8369B CPUs and NVIDIA H20 96G GPUs.

Threshold Decision. For each method, we individually normalize all predicted scores to $[0, 1]$ before applying thresholding. We then set a fixed threshold of 0.5 for all methods. Predictions with scores equal to or higher than the threshold are considered positive (hallucinated) predictions, while scores lower than the threshold are considered negative (correct) predictions.

Localization Experiments. Since most baselines are designed for answer-wise hallucination detection, they are not originally suitable for localizing the specific hallucinated steps from the answer. To this end, we modify the following baselines such that they can predict a binary score individually for each answer step:

- **LengthScore** uses the length of each step as the prediction.

Model	MATH			Science			MultiHopQA		
	#Sample	#Step (correct)	#Step (hallu)	#Sample	#Step (correct)	#Step (hallu)	#Sample	#Step (correct)	#Step (hallu)
R1-7B	192	37.32	30.23	46	41.24	43.43	240	17.92	17.94
R1-14B	93	62.62	63.59	90	108.57	119.89	184	14.17	15.08
Qwen3-8B	52	94.94	100.25	82	134.16	137.7	160	41.03	41.74
Qwen3-14B	57	94.21	90.72	89	126.26	124.85	189	26.44	31.46

Table 3: Dataset statistics. #Step(correct)/#Step(hallu): average number of steps per correct/hallucinated sample.

- **SelfCheckGPT** checks consistencies of each answer step over sampled documents.
- **P(True)** checks the normalized confidence of `False` token for each step.
- **PPL** and **CCP** aggregate token features (entropy or probability) within each step.
- **EigenScore** generates multiple next-steps and computes their EigenScore at the end of each answer step as the prediction.
- **AttentionScore** computes the determinant score of the attention map at the end of each answer step.

D Boundary Case of Valid Repetition

In Section 3, we reveal that the over-focus phenomenon of attention patterns between reasoning and answer steps is a strong indication of reasoning hallucinations. We take a closer look at a boundary case of valid repetition behaviors where models copy outputs from semantically identical preceding content.

Our empirical analysis is conducted with R1-7B on the MATH domain by examining 1,033 instances of valid repetition (where reasoning and answer steps exhibit high lexical similarity judged by Qwen3-235B) and further classifying them into correct repetitions versus hallucinated repetitions (whether the step contains factual errors).

Results show that hallucinated repetitions exhibit significantly higher attention weights compared to correct repetitions, which is consistent with conclusions in Section 3. Specifically, the sum/maximum/average attention scores assigned to repeated reasoning steps are statistically higher in hallucinated cases (one-sided t-tests: p=1.25e-07/8.26e-06/1.92e-04). Moreover, we observe that the attention allocated to the initial several answer steps is significantly lower in hallucinated responses than correct responses (p=5.30e-16 for the first answer step, p=8.65e-07 for the second answer step), suggesting that hallucinated responses show weakened contextual coherence and evidence retrieval.

k_1	k_2	k_3	MATH	Science	QA	$\Delta(\%)$
4	3	5	0.6165	0.6321	0.6166	-2.86
8*	3*	5*	0.6417	**0.6543**	0.6243	-
16	3	5	0.6371	0.6415	0.6126	-1.52
32	3	5	0.6361	0.6355	0.6040	-2.34
64	3	5	0.6355	0.6335	0.5968	-2.86
8	1	5	0.6171	0.6389	0.6192	-2.34
8	5	5	0.6367	0.6451	0.6188	-1.03
8	3	3	0.6367	0.6436	**0.6300**	-0.51
8	3	7	0.6332	0.6456	0.6288	-0.65
8	3	9	**0.6425**	0.6429	0.6219	-0.67

Table 4: Sensitivity analysis of k_1, k_2, and k_3 with R1-7B. **Bold**: best. Asterisk*: default. Metrics refer to average AUROC, AUPRC, and F1 scores.

Model	k_1	MATH	Science	QA	$\Delta(\%)$
Qwen3-8B	4	0.5603	0.5771	0.6428	-5.25
	8*	**0.6294**	**0.6015**	0.6474	-
	16	0.6201	0.5868	0.6511	-1.11
	32	0.5902	0.6014	0.6537	-1.75
	64	0.5966	0.5926	**0.6588**	-1.64
R1-14B	4	0.6123	**0.6492**	0.6127	-1.11
	8	0.6306	0.6451	0.6291	+0.50
	16*	0.6394	0.6269	0.6293	-
	32	**0.6606**	0.6076	**0.6431**	+0.81
	64	0.6388	0.6043	0.6195	-1.75
Qwen3-14B	4	0.6253	0.5756	0.6406	-3.42
	8	0.6134	0.5944	0.6651	-1.82
	16*	0.6207	**0.6143**	**0.6721**	-
	32	0.6381	0.5996	0.6624	-0.34
	64	**0.6573**	0.5686	0.6370	-2.26

Table 5: Sensitivity analysis of k_1 with different models. **Bold**: best. Asterisk*: default. Metrics refer to average AUROC, AUPRC, and F1 scores. Results of R1-7B are shown in Table 4.

E Sensitivity Analysis

E.1 Hyperparameters

Table 4 and Table 5 show the results of a comprehensive hyperparameter sweep. While our default setting is slightly suboptimal compared to the best-performing combination, it provides a robust and convenient out-of-the-box solution.

Model	Seg.	MATH	Science	MultiHopQA
R1-7B	\n\n	0.6417	0.6543	0.6243
	spaCy	0.6283 (-2.1%)	0.6261 (-4.4%)	0.6240 (-0.1%)
	NLTK	0.6256 (-2.6%)	0.6596 (+0.9%)	0.6391 (+2.4%)
Qwen3-8B	\n\n	0.6294	0.6015	0.6474
	spaCy	0.6276 (-0.3%)	0.6085 (+1.4%)	0.6577 (+1.6%)
	NLTK	0.6114 (-3.0%)	0.6167 (+2.8%)	0.6594 (+1.9%)

Table 6: Results with different segmentation methods. Metrics refer to average AUROC, AUPRC, and F1 scores.

Evaluate whether the expression inside the \boxed{} in the provided answer exactly matches the given ground truth. Ignore differences in formatting, whitespace, or stylistic notation, but consider semantic equivalence and symbolic identity.
Return "Correct" if they are equivalent, and "Incorrect" otherwise.
Answer: {answer}
Ground Truth: {gold}

Table 7: Prompt for answer checking.

E.2 Segmentation

In default settings, we adopt a step-separation instruction for \n\n format, following prior work (Sun et al., 2026). This simple delimiter effectively prompts the model to structure its reasoning into distinct steps during generation.

To assess the sensitivity of our method to segmentation strategies, we conducted an ablation study using alternative approaches. Specifically, sentences are split via NLTK (with the punkt tokenizer) and spaCy (with the en_core_web_sm model). The average number of steps increases from 61.5 to 102.4 (with NLTK) and 105.1 (with spaCy). As shown in Table 6, different segmentation methods yield performance comparable to the default \n\n separator. However, they introduce additional dependencies and computational overhead. Given the simplicity and effectiveness of the \n\n approach, we retain it as our default while acknowledging that other segmentation schemes are viable.

F Detailed Experimental Results

F.1 Efficiency

Table 10 shows the full results of the performance and inference time of representative methods.

 # Instruction
You are an expert in analyzing the incorrectness of a mathematical reasoning chain. Your task is to identify which steps are incorrect.
 # Query
{query}
 # True Answer
{answer}
 # Candidate Reasoning Steps
{chain}
Please list all incorrect steps in the following JSON format.
```json
{
  "error_steps": [0, 1, 2, 3...]
}
```

Table 8: Prompt for annotating hallucinated steps.

F.2 Ablation Study

Table 11 shows the full results of the ablated variants.

G Prompt

Table 7 shows the prompt for checking the correctness of a generated answer. Table 8 shows the prompt for annotating hallucinated steps given an incorrect answer. Table 9 shows the prompt for mapping semantically similar reasoning to answer steps.

 # Instruction
You are given a reasoning trace that consists of two parts: "think" and "response".
Each part is a list of steps (sentences). Your task is to map each step in the "response" to the most relevant step(s) in the "think" that directly support or lead to it.
Format your answer as a JSON object with the key "mapping", whose value is a list of objects.
Each object corresponds to one "response" step (in order) and has:
- "response_step_index": the 0-based index of the response step,
- "think_step_indices": a list of 0-based indices from the "think" steps that justify or produce this response step.
Only include direct and clear correspondences. If a response step draws from multiple think steps, list all relevant indices.
If none clearly correspond, use an empty list.
 # Think Steps
{thinking_steps}
 # Response Steps
{response_steps}
Now output the mapping in the specified JSON format.

Table 9: Prompt for semantic correlation between reasoning and answer steps.

Model	Dataset	Ensemble						Uncertainty		Self-Aware					
		SINdex		SelfCheckGPT		RACE		PPL	CCP	EigenScore		AttnScore	UQAC	RHD	RFS-Guard
		Time	#Token	Time	#Token	Time	#Token	Time	Time	Time	#Token	Time	Time	Time	Time
R1-7B	MATH	24.0	35,657	24.7	37,510	165.2	124,372	0.2	38.9	20.0	18,213	0.9	14.1	2.5	2.8
	Science	58.7	60,633	33.7	65,227	181.3	198,922	0.9	36.5	39.1	19,123	0.8	20.5	2.5	3.7
	MultiHopQA	27.8	12,019	8.5	11,836	72.6	39,844	0.3	8.8	34.8	4,218	0.3	8.6	0.8	1.4
R1-14B	MATH	67.3	63,320	71.4	66,774	284.8	207,675	0.8	28.3	51.1	21,412	2.8	129.9	5.7	12.6
	Science	82.9	88,840	91.1	94,302	332.6	286,582	1.4	48.1	81.4	29,914	4.6	257.2	120.0	21.1
	MultiHopQA	10.1	10,577	10.4	11,068	57.9	36,651	0.2	7.3	10.0	3,591	0.5	17.7	0.9	2.7
Qwen3-8B	MATH	47.8	70,209	49.0	73,008	209.9	225,261	0.7	70.2	29.3	23,948	3.4	54.1	6.8	11.0
	Science	67.0	102,972	75.2	108,365	294.8	329,593	1.0	109.2	63.9	35,968	4.9	97.8	13.8	16.6
	MultiHopQA	12.2	20,025	13.0	21,016	66.7	66,131	0.2	16.9	10.9	7,749	1.4	25.8	2.5	5.7
Qwen3-14B	MATH	64.4	73,258	69.7	77,597	272.9	237,983	1.1	70.4	49.8	23,841	4.0	120.0	9.5	15.7
	Science	96.9	112,476	72.4	117,814	326.7	359,259	1.6	108.4	80.6	40,713	5.7	98.8	10.4	22.0
	MultiHopQA	13.0	16,186	11.0	17,291	65.2	54,852	0.3	20.1	8.6	5,121	1.1	28.0	2.0	3.7

Table 10: Hallucination detection efficiency. `Time`: inference time (seconds) per sample. `#Token`: number of tokens generated per sample.

Model	Variant	MATH				Science				MultiHopQA			
		AUROC	AUPRC	F1	Δ	AUROC	AUPRC	F1	Δ	AUROC	AUPRC	F1	Δ
R1-7B	Full RFS-Guard	0.6370	0.6215	0.6666	-	0.6738	0.6222	0.6670	-	0.6096	0.5996	0.6638	-
	w/ LastToken	0.6187	0.6069	0.6537	-2.4%	0.6443	0.6268	0.6666	-1.2%	0.5939	0.5977	0.6618	-1.1%
	w/ Embedding	0.5896	0.5604	0.5752	-10.3%	0.5961	0.5769	0.6504	-7.1%	0.5727	0.5556	0.6460	-5.4%
	w/ LLM	0.6265	0.6012	0.6668	-1.6%	0.6323	0.6058	0.6484	-3.9%	0.5728	0.5540	0.6510	-5.2%
	w/o Flow	0.6107	0.6019	0.6655	-2.5%	0.6559	0.6235	0.6470	-1.8%	0.5878	0.5680	0.6571	-3.3%
	w/ MaxPool	0.6134	0.5968	0.6537	-3.2%	0.6081	0.5546	0.6569	-7.4%	0.6016	0.5910	0.6638	-0.9%
	w/ SumPool	0.6069	0.5914	0.6537	-3.8%	0.6596	0.6096	0.6417	-2.6%	0.5987	0.5901	0.6638	-1.1%
Qwen3-8B	Full RFS-Guard	0.6115	0.6187	0.6579	-	0.5760	0.5619	0.6667	-	0.6549	0.6248	0.6624	-
	w/ LastToken	0.6107	0.5906	0.6581	-1.5%	0.5744	0.5555	0.6612	-0.7%	0.6456	0.6125	0.6681	-0.8%
	w/ Embedding	0.5627	0.5370	0.6343	-8.3%	0.5425	0.5119	0.6368	-6.4%	0.6011	0.6003	0.6268	-5.8%
	w/ LLM	0.5843	0.5917	0.6422	-3.7%	0.5551	0.5220	0.6572	-4.1%	0.6284	0.6079	0.6476	-3.0%
	w/o Flow	0.5799	0.5370	0.6622	-5.9%	0.5273	0.5361	0.6475	-5.3%	0.6321	0.6093	0.6505	-2.6%
	w/ MaxPool	0.5937	0.6072	0.6667	-1.1%	0.5733	0.5476	0.6087	-3.9%	0.6394	0.5918	0.6638	-2.5%
	w/ SumPool	0.6009	0.6132	0.6714	-0.2%	0.5775	0.5436	0.6010	-4.3%	0.6441	0.6030	0.6652	-1.6%

Table 11: Ablation study results. Δ: difference compared with the full method.