CORTEX: Token-Level Hallucination Detection in RAG via Comparative Internal Representations

Anonymous ACL submission

Abstract

In this paper, we propose CORTEX, a token-level hallucination detection method for Retrieval-Augmented Generation (RAG). In long-form RAG outputs, hallucinations often arise in localized spans rather than throughout an entire response. CORTEX therefore identifies ungrounded content at the token level, enabling fine-grained localization of hallucinations. The key intuition behind CORTEX is that tokens grounded in retrieved documents should be more strongly influenced by those documents than hallucinated tokens. To capture this document-induced effect, CORTEX compares internal representations of a large language model (LLM) under two conditions: with and without the retrieved documents. Instead of relying solely on each token's immediate sensitivity to the retrieved documents, CORTEX also leverages the propagation of document-grounded information through preceding tokens, reducing false positives for tokens whose evidence has already been absorbed into the context. Finally, CORTEX applies post-processing smoothing step that models the tendency of hallucination labels to persist over contiguous spans, reducing local noise and encouraging span-consistent predictions. Experiments on two RAG benchmarks and three LLMs show that CORTEX substantially improves token-level hallucination detection, with each component consistently contributing to performance gains.

1 Introduction

Retrieval-Augmented Generation (RAG) has been widely adopted to improve the factuality of large language models (LLMs) by grounding generation in external references (Lewis et al., 2020). Although RAG mitigates hallucinations to some extent, LLMs may still generate groundless or inconsistent content even when references are provided (Fan et al., 2026). Accurately detecting such hallucinations within generated outputs therefore remains a critical challenge (Niu et al., 2024; Liu et al., 2025; Bang et al., 2025).

Various hallucination detection methods have been proposed to address this challenge. Self-consistency-based approaches estimate reliability by measuring agreement across multiple outputs from the same prompt (Manakul et al., 2023). Prompt-based methods use LLMs as fact-checkers to explicitly verify generated content (Zheng et al., 2023; Es et al., 2024; Furumai et al., 2024). While these approaches leverage LLMs' strong language understanding capabilities, they may incur high computational cost due to repeated sampling, and their reliability can depend on prompt design (Ganesh et al., 2026).

More recently, hallucination detection methods based on the internal representations of LLMs have been increasingly studied (Sriramanan et al., 2024; Zhang et al., 2025; Xiong et al., 2026). These methods seek hallucination-related signals in model activations and can offer insight into the internal conditions under which hallucinations arise. This line of work has analyzed attention and feedforward-network states, quantified component contributions to generation, and extracted hallucination-related features from internal representations.

However, many existing approaches are not designed specifically for RAG and primarily assess the generated output as a whole, i.e., at the answer level. This is limiting for long-form RAG outputs, where faithful and hallucinated content often coexist. For practical use, detecting hallucinations in such mixed outputs requires a finer-grained formulation than answer-level detection. The appropriate granularity can vary across tasks and annotation protocols, from words and phrases to sentences or paragraphs. Given this variability, token-level detection can serve as a practical approach for localizing ungrounded content and later adapting predictions to the required granularity.

In this paper, we propose **CORTEX**

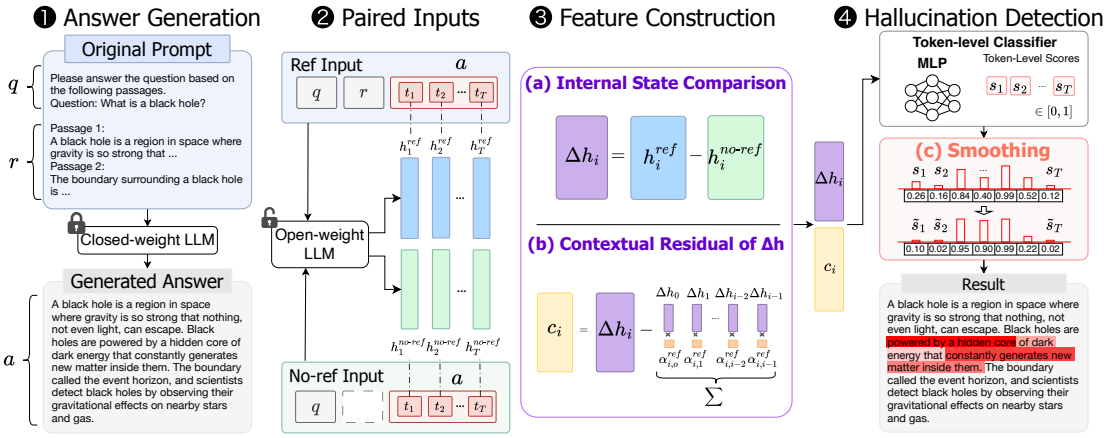

Figure 1: Overview of CORTEX for token-level hallucination detection in RAG, using reference-conditioned internal representation comparisons, contextual residual features, and label-persistence smoothing.

(**C**omparative **O**bserved **R**eference-based **T**oken-level **EX**pressions), a post-hoc token-level hallucination detection method for RAG. Figure 1 illustrates the overall framework. CORTEX builds on the intuition that faithful tokens should exhibit coherent representational changes when references are provided, whereas hallucinated tokens should show weaker or less coherent changes. To measure this effect, CORTEX constructs a paired counterfactual view of the same answer under reference-conditioned and no-reference inputs, rather than probing a single input as in conventional representation-based detectors. The resulting reference-induced changes are then encoded as token-level delta features for hallucination detection.

CORTEX further introduces an attention-based contextual residual feature. In long-form generation, reference influence may propagate indirectly through preceding answer tokens, as in reasoning-style or self-referential generation. CORTEX captures this context-mediated reference influence by combining token-level delta features with attention patterns, helping the classifier distinguish tokens that are indirectly grounded in the reference from groundless tokens and thereby reducing false positives.

Finally, CORTEX uses post-processing smoothing step to adjust the granularity of token-level scores. Raw token-level predictions can be sensitive to local noise and may produce isolated high-score tokens, whereas human hallucination annotations often appear as contiguous spans. We therefore introduce *label-persistence smoothing*, which treats raw scores as token-level confidence and models the tendency of hallucination labels to persist across neighboring tokens. This reduces scattered noise and adapts token-level predictions to span-level annotation structure while still retaining token-level scores.

Experiments on two RAG benchmarks and three LLMs demonstrate that CORTEX substantially improves token-level hallucination detection, with all components contributing consistently to performance gains.

Our contributions are summarized as follows:

- We propose CORTEX, which, to the best of our knowledge, is the first token-level hallucination detection method specifically designed for RAG.
- CORTEX is a practical post-hoc framework that is easy to implement and can detect hallucinations in outputs from arbitrary LLMs, including API-based closed-weight models.

2 Preliminaries

We consider a practical setting in which the answer is produced by a closed-weight LLM whose parameters and internal representations are not accessible, as is the case for many API-based models. Such models are often attractive in deployed applications because of their strong generation quality. We denote this closed-weight LLM by M_{close}. Given a question q^{text}, the closed-weight LLM produces an answer as

$$a^{\text{text}} = M_{\text{close}}(q^{\text{text}}). \tag{1}$$

Our goal is to identify hallucinated tokens in the generated answer a^{text}. Although a direct way to

obtain hallucination-related features would be to analyze the internal representations of the closed-weight LLM, we cannot do so for the aforementioned reason. We therefore use a separate open-weight LLM, denoted by M_{open}, as a post-hoc analysis model. Unlike M_{close}, M_{open} exposes internal representations, which allows us to extract features from the answer in relation to the question.

Let $\text{Tok}(\cdot)$ denote the tokenizer of M_{open}. We tokenize the answer as

$$a = \text{Tok}(a^{\text{text}}) = (t_1, t_2, \ldots, t_i, \ldots, t_T), \quad (2)$$

where T denotes the number of answer tokens and t_i denotes the i-th answer token under the tokenizer of M_{open}. We assume that M_{open} processes an input containing the question and the answer as $M_{\text{open}}(q^{\text{text}} \| a^{\text{text}})$, where $\|$ denotes text concatenation with the appropriate prompt format. M_{open} produces token-level internal representations over the entire input sequence, which capture the relationships between q^{text} and a^{text}. Since hallucination detection is performed over the answer, we use only the internal representations corresponding to the answer tokens. We denote the representation aligned with answer token t_i by $h_i \in R^d$. The specific input construction and the extraction of answer-token representations are described in Section 3.

The token-level hallucination detection task is to estimate whether each answer token is hallucinated. Let $y_i \in \{0, 1\}$ denote the token-level label, where $y_i = 1$ indicates that token t_i is hallucinated and $y_i = 0$ indicates that it is faithful.

In our setting, we focus on RAG applications, where references are provided as grounding evidence for answers. This setting reflects common practical deployments in which retrieved documents are used to reduce hallucination and the answer is expected to be faithful to those references.

3 CORTEX

We propose CORTEX (**C**omparative **O**bserved **R**eference-based **T**oken-level **EX**pressions), a post-hoc hallucination detection framework for RAG.

CORTEX is built on three key ideas, illustrated in Figure 1(a)–(c): (a) a reference-induced delta representation for capturing how each answer token's internal representation changes when the references are provided; (b) an attention-based contextual residual for distinguishing direct reference sensitivity from reference influence mediated by preceding answer tokens; and (c) label-persistence smoothing for reducing isolated token-level noise and obtaining span-consistent hallucination scores while preserving token-level predictions.

3.1 Reference-Induced Delta Representation

CORTEX constructs two conditioned inputs for the open-weight LLM, differing only in whether the references are included. Let r^{text} denote the references, i.e., the retrieved documents used as grounding evidence in the RAG setting. The reference-conditioned input is defined as $x_{\text{ref}}^{\text{text}} = q^{\text{text}} \| r^{\text{text}} \| a^{\text{text}}$, whereas the no-reference input is defined as $x_{\text{no-ref}}^{\text{text}} = q^{\text{text}} \| a^{\text{text}}$. With this construction, the answer span corresponds to the same token sequence $(t_1, \ldots, t_T)$ in both conditions, allowing CORTEX to compare internal representations aligned with the same answer tokens.

When M_{open} processes $x_{\text{ref}}^{\text{text}}$ and $x_{\text{no-ref}}^{\text{text}}$, it produces token-level internal representations over the entire input sequence. For an input $x = \text{Tok}(x^{\text{text}})$, let $\text{Rep}_{M_{\text{open}}}(x)$ denote the sequence of internal representations obtained from M_{open}. We define an extraction function f that selects the representations aligned with the answer tokens:

$$\begin{aligned} (h_1^{\text{ref}}, \ldots, h_T^{\text{ref}}) &= f(\text{Rep}_{M_{\text{open}}}(x_{\text{ref}})), \\ (h_1^{\text{no-ref}}, \ldots, h_T^{\text{no-ref}}) &= f(\text{Rep}_{M_{\text{open}}}(x_{\text{no-ref}})). \end{aligned} \quad (3)$$

Here, $h_i^{\text{ref}}, h_i^{\text{no-ref}} \in R^d$ denote the internal representations aligned with the same answer token t_i under the reference-conditioned and no-reference conditions, respectively, and d denotes the representation dimensionality.[1]

CORTEX builds on the intuition that faithful tokens should exhibit coherent representational changes when references are provided, whereas hallucinated tokens should show weaker or less coherent changes. Based on this intuition, CORTEX computes the reference-induced delta representation for each answer token t_i as

$$\Delta h_i = h_i^{\text{ref}} - h_i^{\text{no-ref}}. \quad (4)$$

This vector is the core representation in CORTEX. Since the same answer is included in both inputs, Δh_i captures how the representation of token t_i changes when references are provided. In other words, Δh_i is intended to emphasize the reference-induced change in how M_{open} contextualizes that

[1] Specifically, we use the final transformer layer output corresponding to each answer token as its token-level representation h_i.

token, rather than merely representing the token content itself.

3.2 Attention-Based Contextual Residual

The delta representation Δh_i captures how the representation of token t_i changes when references are added. However, in reasoning-style outputs such as chain-of-thought, facts grounded in the references may first be stated in earlier parts of the answer, and later tokens may continue the reasoning based on those facts. In this case, the influence of the references can reach t_i not only through the direct path $r \to t_i$, but also through the context-mediated path $r \to t_{<i} \to t_i$. As a result, even tokens that are indirectly grounded by the references may appear to have a weak relationship with the references if we only use Δh_i.

To address this issue, we introduce a feature that represents how much reference influence is contained in the preceding tokens that the current token t_i relies on as

$$\bar{\Delta h}_i = \sum_{j<i} \alpha_{ij}^{\text{ref}} \Delta h_j, \tag{5}$$

where $\alpha_{ij}^{\text{ref}} \in [0, 1]$ denotes the attention weight from t_i to a preceding answer token t_j under the reference-conditioned input. We further subtract this context-mediated influence from the current token's own change and define the contextual residual as

$$c_i = \Delta h_i - \bar{\Delta h}_i. \tag{6}$$

This residual represents the token-specific change that remains after removing the reference influence explainable through preceding tokens. We use both Δh_i and c_i for token-level hallucination detection.

3.3 Token-Level Hallucination Detection

For each answer token t_i, CORTEX predicts a raw token-level hallucination score s_i by feeding $[\Delta h_i; c_i]$ into a multilayer perceptron (MLP) classifier g_θ:

$$s_i = \sigma(g_\theta([\Delta h_i; c_i])), \qquad s_i \in [0, 1], \tag{7}$$

where σ denotes the sigmoid function. The classifier is trained with binary cross-entropy loss using token-level hallucination labels. To address class imbalance, the positive class is weighted according to the ratio of negative to positive tokens in the training set.

3.4 Label-Persistence Smoothing

We define a span as a contiguous sequence of tokens annotated as a single hallucination unit. Human hallucination annotations are typically span-based: once a token is marked as hallucinated, neighboring tokens in the same span are likely to receive the same label. Based on this property, we apply *label-persistence smoothing* as a post-processing step, rather than using moving-average smoothing.

Given the raw token-level scores $s_{1:T}$, we introduce an unobserved binary smoothing-label variable $z_i \in \{0, 1\}$ to represent the span-consistent label underlying the post-processed score, where $z_i = 0$ denotes a faithful token and $z_i = 1$ denotes a hallucinated token. To obtain span-consistent post-processed scores, we model the smoothed label sequence by combining two components around positions i and $i + 1$: the tendency of neighboring labels to persist and information from the raw scores at each position.

We first model span-level label persistence by defining the pairwise persistence term between neighboring labels as

$$\rho(z_i, z_{i+1}) = \begin{cases} p_{\text{stay}}, & z_{i+1} = z_i, \\ 1 - p_{\text{stay}}, & z_{i+1} \neq z_i. \end{cases} \tag{8}$$

The parameter $p_{\text{stay}} \in [0, 1]$ controls the degree of label persistence: smaller values allow finer-grained label changes, whereas larger values favor longer same-label spans. This provides a deliberately simple approximation to span-level label continuity, rather than modeling the full variability of human annotation patterns. We then define the token-level confidence induced by the raw score s_i at each position:

$$\phi_i(z_i = 1) = s_i, \qquad \phi_i(z_i = 0) = 1 - s_i. \tag{9}$$

This quantity represents how the raw score at position i is converted into token-level confidence for each value of the smoothing-label variable.[2]

Combining Eq. (8) and Eq. (9), we define the local compatibility term for neighboring positions i and $i + 1$ as

$$\phi_i(z_i)\rho(z_i, z_{i+1})\phi_{i+1}(z_{i+1}). \tag{10}$$

[2] In implementation, we apply a small amount of clipping to the raw scores so that the token-level confidence does not become exactly 0 or 1.

This term measures the compatibility of the neighboring label assignment (z_i, z_{i+1}) with both the raw scores and the label-persistence assumption.

We therefore define the normalized distribution over sequence labels as

$$P(z_{1:T} \mid s_{1:T}) = \frac{1}{Z(s_{1:T})} \pi(z_1) \prod_{i=1}^{T} \phi_i(z_i) \times \prod_{i=1}^{T-1} \rho(z_i, z_{i+1}), \tag{11}$$

where $Z(s_{1:T})$ is the normalizing constant and $\pi(z_1)$ is the initial distribution over the first smoothing label. We use a uniform initial distribution, $\pi(0) = \pi(1) = 1/2$.

Although $z_{1:T}$ is unobserved, the desired token-level smoothed score can be obtained by marginalizing over all smoothing-label sequences:

$$\tilde{s}_i = P(z_i = 1 \mid s_{1:T}). \tag{12}$$

This posterior can be computed efficiently using the forward–backward algorithm (Rabiner, 1990).We provide the full recursions and posterior marginal formula in Appendix A.

4 Experiments

We evaluate CORTEX on two publicly available RAG hallucination benchmarks, RAGTruth (Niu et al., 2024) and HalluRAG (Ridder and Schilling, 2024), using three LLMs: Llama-3.1-8B-Instruct (Grattafiori et al., 2024), Qwen3-8B (Yang et al., 2025), and Mistral-7B-Instruct-v0.2 (Mistral AI, 2023). We use these open-weight LLMs to obtain internal representations for CORTEX and the relevant baselines. Evaluation is performed at the token and answer levels. Token-level evaluation assesses whether a method can identify hallucinated tokens in a generated answer, while answer-level evaluation assesses whether the answer contains any hallucinated content. For CORTEX, answer-level evaluation is performed by aggregating token-level scores, as it is trained for token-level detection and does not use answer-level supervision. Specifically, we use the maximum token-level hallucination score as the answer-level score.

For label-persistence smoothing, we use $p_{\text{stay}} = 0.993$ at the token level and $p_{\text{stay}} = 0.930$ at the answer level. We provide a sensitivity analysis of p_{stay} in Appendix B. Additional details on the experimental setup are provided in Appendix C. The following subsections describe the baselines and datasets used in our experiments.

4.1 Baselines

We compare CORTEX with five baselines. **NLL** uses negative log-likelihood as an uncertainty-based signal, following prior work on uncertainty estimation from next-token predictive distributions (Malinin and Gales, 2021). **SAPLMA** trains a probe on intermediate transformer representations, based on the observation that hidden states encode hallucination-related signals (Azaria and Mitchell, 2023). **LLM-Check** extracts features from internal states produced by attention mechanisms and feedforward neural networks, including spectral properties such as eigenvalues (Sriramanan et al., 2024). **ICR Probe** uses internal component attribution to quantify the contribution of attention and feedforward networks and identify hallucinated content (Zhang et al., 2025). **RAGLens** applies sparse autoencoders to token embeddings and selects hallucination-related sparse features via token aggregation and mutual-information-based feature selection (Xiong et al., 2026).

For answer-level evaluation, we follow the original implementation of baselines. However, their answer-level implementation cannot be directly used for token-level evaluation. We modify parts of the baseline implementations to adapt them to token-level detection while preserving their original mechanisms as much as possible.

4.2 Datasets

Datasets for hallucination detection in RAG settings remain limited, especially those that include both references and fine-grained hallucination annotations. To the best of our knowledge, RAGTruth is the only publicly available RAG benchmark with human-annotated hallucination spans that are sufficiently fine-grained to support token-level evaluation. Although RAGTruth contains multiple task types, we use only its QA subset in this work.

HalluRAG consists of answers generated by multiple LLMs under two RAG prompt settings using either relevant or irrelevant Wikipedia chunks, with sentence-level hallucination annotations by GPT-4o (OpenAI et al., 2024). Since token-level labels are unavailable, we derive token-level pseudo-labels by marking all tokens in hallucinated sentences as hallucinated.

The two datasets provide complementary settings, differing in annotation granularity, labeling procedure, retrieval quality, and QA distribution. RAGTruth provides fine-grained human span anno-

Statistic	RAGTruth	HalluRAG
# Samples	5,934	2,243
Train samples	4,530	1,662
Validation samples	504	185
Test samples	900	396
Avg. prompt len.	1,660.0	1,778.9
Avg. answer len.	686.5	164.0

Table 1: Dataset statistics.

tations, whereas HalluRAG provides sentence-level GPT-4o annotations and includes cases where references may be irrelevant to the question. They also differ in construction and topical coverage, with RAGTruth's QA subset is based on daily-life questions, whereas HalluRAG derives questions from Wikipedia passages. Together, these differences allow us to assess robustness across conditions.

Both datasets contain answers generated by various LLMs, together with the references provided for generating those answers. We treat these answers as outputs from closed-weight LLMs, regardless of which model generated them. To ensure a fair comparison, all methods receive the same answer and corresponding references as input.

4.3 Results

Token-level Detection

Table 2 reports token-level hallucination detection results in terms of average precision (AP) and the area under the receiver operating characteristic curve (AUROC). Across both RAGTruth and HalluRAG, CORTEX achieves the best performance in all settings. The gains are particularly substantial on RAGTruth, where human fine-grained annotations are available, suggesting that CORTEX is well aligned with token-level hallucination detection. Even on HalluRAG, where token-level labels are derived from sentence-level annotations, CORTEX remains the best-performing method, indicating robustness to coarser and noisier supervision.

All baselines receive inputs with references, but rely on a single reference-conditioned view of model representations. SAPLMA directly uses transformer layer outputs as features, while RAGLens extracts hallucination-related sparse features from token embeddings using a trained sparse encoder. In contrast, CORTEX derives its signal from the contrast between paired representations of the same tokens with and without references. The consistent gains suggest that this comparative formulation captures reference-induced hallucination signals that are not readily available from a single internal representation or features derived from it.

Label-persistence smoothing is a post-processing module applicable to any token-level predictions. In Appendix D, we further apply it to baselines to demonstrate its general effectiveness.

Answer-level Detection

Table 3 reports the answer-level hallucination detection results in terms of AP and AUROC. Unlike the baselines, which are trained directly for answer-level detection, CORTEX does not use answer-level supervision. Instead, CORTEX reuses the token-level hallucination scores and assigns each answer the maximum score among its tokens. Thus, this setting evaluates whether the token-level signals captured by CORTEX can also indicate hallucination at the answer level.

Despite this simple aggregation strategy, CORTEX achieves competitive performance against the baselines. This suggests that its token-level predictions provide useful signals for both localized detection and answer-level reliability assessment. At the same time, the strong performance of some answer-level baselines suggests that answer-level detection may require global signals beyond localized token-level signals. Taken together, these results indicate that token-level and answer-level hallucination detection are both important and should be studied as complementary problems.

4.4 Ablation Study

To analyze the contribution of each CORTEX component, we conduct an ablation study. Table 4 reports token- and answer-level performance for each configuration. Both the contextual residual c and label-persistence smoothing (Smooth) improve performance. Removing label-persistence smoothing shows that raw token-level scores are informative but locally unstable, while removing c shows that context-mediated reference influence provides information complementary to Δh.

Figures 2 and 3 illustrate these effects qualitatively using cases with and without hallucinations. In the heatmaps, color intensity reflects the predicted hallucination score, with redder tokens indicating higher hallucination likelihood. In Figure 2, the model without label-persistence smoothing produces interleaved high- and low-score tokens across semantically coherent regions, yielding predictions less consistent with human span-level annotations. In contrast, full CORTEX produces smoother and more contiguous high-score

Method	RAGTruth						HalluRAG					
	Llama		Qwen		Mistral		Llama		Qwen		Mistral	
	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC
NLL	0.0476	0.4676	0.0495	0.4686	0.0479	0.4666	0.2648	0.4952	0.2658	0.4934	0.2685	0.4945
LLM-Check	0.0769	0.6079	0.0951	0.6480	0.0743	0.6189	0.2994	0.5411	0.2874	0.5342	0.2944	0.5467
ICR Probe	0.2665	0.5380	0.2464	0.5067	0.2621	0.5311	0.3292	0.5285	0.3131	0.5106	0.3197	0.5143
SAPLMA	0.3894	0.8879	0.4182	0.8820	0.4027	0.8822	0.6875	0.8024	0.6584	0.8129	0.6193	0.7578
RAGLens	0.4271	0.8953	0.4575	0.9075	0.4244	0.8884	0.5725	0.7425	0.6380	0.7639	0.5596	0.7393
CORTEX	**0.5686**	**0.9275**	**0.5940**	**0.9372**	**0.5473**	**0.9244**	**0.7690**	**0.8360**	**0.7495**	**0.8426**	**0.7308**	**0.8233**

Table 2: Token-level hallucination detection results on RAGTruth and HalluRAG. Bold and underline indicate the best and second-best results, respectively.

Method	RAGTruth						HalluRAG					
	Llama		Qwen		Mistral		Llama		Qwen		Mistral	
	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC	AP	AUROC
NLL	0.1994	0.5621	0.2019	0.5227	0.2313	0.6088	0.2055	0.4827	0.3184	0.5806	0.2969	0.5600
LLM-Check	0.3375	0.7100	0.3330	0.7198	0.3240	0.7170	0.3052	0.5711	0.3956	0.6638	0.4059	0.6481
ICR Probe	0.1286	0.3459	0.2301	0.6162	0.1831	0.5182	0.4077	0.6640	0.3286	0.6031	0.2157	0.4310
SAPLMA	0.6708	0.8863	0.5901	0.8600	0.4897	0.8077	0.7922	0.8482	0.7867	**0.8695**	0.7529	0.8628
RAGLens	**0.7329**	**0.9011**	0.6679	0.8812	0.6781	0.8819	**0.7994**	**0.8905**	0.7669	0.8380	0.8006	**0.8835**
CORTEX	0.6893	0.8957	**0.7596**	**0.9077**	**0.6787**	**0.8907**	0.7795	0.8481	**0.7907**	0.8627	**0.8010**	0.8718

Table 3: Answer-level hallucination detection results. Bold and underline indicate the best and second-best results, respectively.

Method	Token-level		Answer-level	
	AP	AUROC	AP	AUROC
Δh	0.4732	0.9074	0.6040	0.8853
$\Delta h + c$	0.5335	0.9156	0.6910	0.9011
$\Delta h + c + \text{Smooth}$	0.5940	0.9372	0.7596	0.9077

Table 4: Ablation results on RAGTruth using Qwen.

regions, better matching the ground-truth hallucination spans. Figure 3 illustrates the role of the contextual residual c. Without c, the model can overemphasize the absence of direct reference influence and assign high scores to tokens indirectly grounded through the preceding answer context. With c, CORTEX accounts for reference influence propagated through previous tokens and reduces false positives caused by indirect grounding.

Overall, these ablation results support the design of CORTEX. The combination of Δh, the contextual residual c, and label-persistence smoothing enables CORTEX to achieve robust localized hallucination detection. Further ablation case studies are presented in Appendix E.

5 Related Work

The problem of hallucination in LLMs has been extensively studied, yet remains unresolved (Huang et al., 2025; Kalai et al., 2025). It undermines the reliability of LLM-based AI agents and hinders real-world deployment. RAG mitigates hallucinations by incorporating references into prompts (Gao et al., 2024; Rackauckas, 2024), but LLMs may still generate hallucinated content even with references, motivating the detection of groundless or inconsistent RAG outputs (Fan et al., 2026).

Many hallucination detection methods rely on external verification or repeated generation. Self-consistency methods estimate reliability by comparing multiple outputs from the same prompt (Manakul et al., 2023), while prompt-based approaches use LLMs as judges or fact-checkers (Li et al., 2025; Es et al., 2024; Furumai et al., 2024). These methods exploit LLMs' language understanding but depend on prompt design and verifier capability, and often require additional inference.

Another line of work uses model-internal signals, including uncertainty from next-token distributions (Malinin and Gales, 2021), hallucination-related information in intermediate transformer representations (Azaria and Mitchell, 2023), spectral features of attention and feedforward states (Sriramanan et al., 2024), component-level attribution for tracing information sources (Zhang et al., 2025),

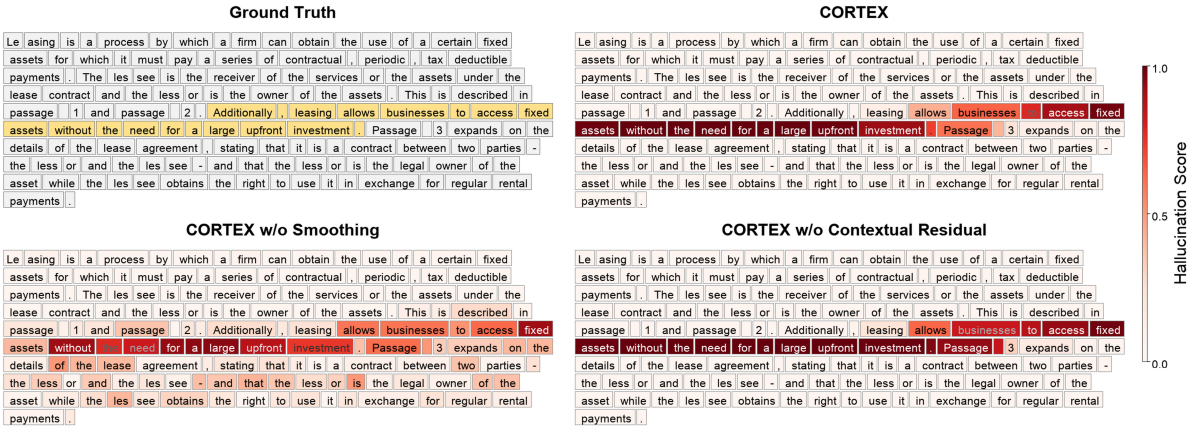

Figure 2: Detection example for an answer containing hallucinated content. Label-persistence smoothing suppresses noisy token-level score patterns and highlights the hallucinated span more accurately and coherently.

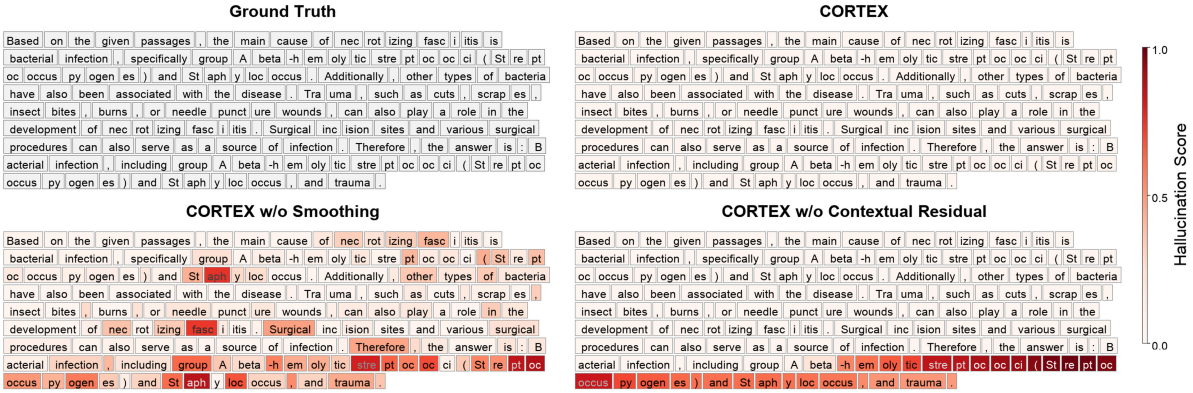

Figure 3: Detection example for an answer without hallucinated content. The contextual residual c reduces false positives by accounting for reference influence mediated through the preceding answer context.

and sparse autoencoder features for RAG hallucination analysis (Xiong et al., 2026). These studies show that hallucination-related signals are present in model computations and representations, motivating representation-based detection.

CORTEX differs from prior representation-based approaches by comparing internal representations obtained from reference-conditioned and no-reference inputs, rather than analyzing a single internal state in isolation. It further uses label-persistence smoothing to reduce isolated noise and better align predictions with span-based hallucination annotations.

6 Conclusion

We proposed CORTEX, a post-hoc token-level hallucination detection method for RAG. CORTEX constructs a paired counterfactual view of the same answer by analyzing its internal representations with and without references, and encodes reference-induced changes as token-level delta features. It combines these features with attention patterns to capture context-mediated reference influence and uses label-persistence smoothing to reduce local noise while preserving span-consistent scores.

Experiments show that CORTEX substantially outperforms token-level baselines and achieves competitive answer-level performance through simple score aggregation. Ablations confirm the contributions of both the attention-based contextual residual and label-persistence smoothing. These results demonstrate the effectiveness of comparing internal representations for hallucination detection in RAG. By using an open-weight LLM to analyze outputs from closed-weight LLMs without modifying generation, CORTEX provides a practical approach to reliability assessment. Beyond detection, these comparative internal-representation signals may provide a basis for future hallucination mitigation, including reward modeling for tuning generators and objectives that encourage stronger reference-induced representations.

Limitations

CORTEX is designed for RAG settings and therefore assumes the presence of references against which generated outputs can be assessed. It is not intended to verify claims that are generated solely from the parametric knowledge of an LLM. This restriction may be acceptable, or even desirable, in controlled applications such as enterprise customer support, where answers should be grounded only in approved documents. In more open-ended assistant scenarios, however, this may be limiting because useful information that is not explicitly grounded in the provided references may be treated as groundless.

Another limitation is that CORTEX requires access to an open-weight LLM that exposes internal representations, including hidden states and attention weights. Although the original generator can be API-based or otherwise inaccessible, the post-hoc analysis model must provide sufficient internal signals. The quality of detection may therefore depend on the choice of the open-weight LLM and its ability to interpret the generated answer and references.

A further limitation is the need for fine-grained supervision during training. Although RAGTruth provides human span-level annotations, such annotations remain scarce for RAG hallucination detection. Improving supervision under limited annotation resources remains an important direction for future work.

Ethical Considerations

CORTEX is intended to support reliability assessment in RAG systems by identifying potentially groundless tokens in generated outputs. Such detection can help reduce the risk of users relying on hallucinated information, especially in applications where answers are expected to be grounded in specific references. However, CORTEX should not be interpreted as a guarantee of factual correctness. Its predictions are probabilistic and may include both false positives and false negatives; therefore, human review or additional verification may still be necessary in high-stakes domains.

There is also a risk that hallucination detection tools could be over-relied upon or used to present generated outputs as more reliable than they actually are. We encourage practitioners to communicate the limitations of detection results clearly and to avoid using CORTEX as the sole safety mechanism for critical decision-making.

It is also important to distinguish the scope of CORTEX from broader safety evaluation. CORTEX is designed to detect ungrounded content in reference-grounded generation, not to determine broader notions of truth, fairness, or social harm. Future work should examine how token-level reliability signals can be combined with other safeguards, including checks for bias, toxicity, privacy risks, and domain-specific safety issues.

References

Amos Azaria and Tom Mitchell. 2023. The internal state of an LLM knows when it's lying. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 967–976. Association for Computational Linguistics.

Yejin Bang, Ziwei Ji, Alan Schelten, Anthony Hartshorn, Tara Fowler, Cheng Zhang, Nicola Cancedda, and Pascale Fung. 2025. HalluLens: LLM hallucination benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 24128–24156. Association for Computational Linguistics.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158. Association for Computational Linguistics.

Dongyang Fan, Sebastien Delsad, Nicolas Flammarion, and Maksym Andriushchenko. 2026. Halluhard: A hard multi-turn hallucination benchmark. *Computing Research Repository*, arXiv:2602.01031.

Kazuaki Furumai, Roberto Legaspi, Julio Cesar Vizcarra Romero, Yudai Yamazaki, Yasutaka Nishimura, Sina Semnani, Kazushi Ikeda, Weiyan Shi, and Monica Lam. 2024. Zero-shot persuasive chatbots with LLM-generated strategies and information retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11224–11249. Association for Computational Linguistics.

Prakhar Ganesh, Reza Shokri, and Golnoosh Farnadi. 2026. Rethinking hallucinations: Correctness, consistency, and prompt multiplicity. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6959–6978, Rabat, Morocco. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Computing Research Repository*, arXiv:2312.10997.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Computing Research Repository*, arXiv:2407.21783.

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 Transactions on Information Systems*, 43(2).

Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. 2025. Why language models hallucinate. *Computing Research Repository*, arXiv:2509.04664.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. 2025. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 2757–2791. Association for Computational Linguistics.

Qiang Liu, Xinlong Chen, Yue Ding, Bowen Song, Weiqiang Wang, Shu Wu, and Liang Wang. 2025. Attention-guided self-reflection for zero-shot hallucination detection in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21005–21021. Association for Computational Linguistics.

Andrey Malinin and Mark John Francis Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *Proceedings of the 2021 International Conference on Learning Representations*.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. 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*, pages 9004–9017. Association for Computational Linguistics.

Mistral AI. 2023. Mistral-7B-Instruct-v0.2. Hugging Face model card.

Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, KaShun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10862–10878. Association for Computational Linguistics.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card.

Lawrence R. Rabiner. 1990. A tutorial on hidden markov models and selected applications in speech recognition. page 267–296.

Zackary Rackauckas. 2024. Rag-fusion: A new take on retrieval augmented generation. *International Journal on Natural Language Computing*, 13(1):37–47.

Fabian Ridder and Malte Schilling. 2024. The hallurag dataset: Detecting closed-domain hallucinations in rag applications using an llm's internal states. *Computing Research Repository*, arXiv:2412.17056.

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 *Proceedings of the 38th International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Guangzhi Xiong, Zhenghao He, Bohan Liu, Sanchit Sinha, and Aidong Zhang. 2026. Toward faithful retrieval-augmented generation with sparse autoencoders. In *The Fourteenth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Computing Research Repository*, arXiv:2505.09388.

Zhenliang Zhang, Xinyu Hu, Huixuan Zhang, Junzhe Zhang, and Xiaojun Wan. 2025. ICR probe: Tracking hidden state dynamics for reliable hallucination detection in LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pages 17986–18002. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Proceedings of the 37th International Conference*

on Neural Information Processing Systems. Curran Associates Inc.

A Details of Label-Persistence Smoothing

This appendix gives the forward–backward recursions and posterior marginal derivation for label-persistence smoothing. The method defines non-negative weights over smoothing-label sequences by combining token-level confidence with label persistence between neighboring labels, so that posterior marginals can be computed efficiently without enumerating all 2^T possible label sequences.

A.1 Forward Recursion

The forward message $F_i(\ell)$ is the unnormalized total weight of all partial smoothing-label sequences ending with $z_i = \ell$:

$$F_i(\ell) = \sum_{z_{1:i}:z_i=\ell} \pi(z_1) \prod_{j=1}^{i} \phi_j(z_j) \times \prod_{j=2}^{i} \rho(z_{j-1}, z_j). \tag{13}$$

The initialization is

$$F_1(\ell) = \pi(\ell)\phi_1(\ell). \tag{14}$$

For $i = 2, \ldots, T$, the recursion is obtained by grouping the partial sequences according to the previous label $\ell' = z_{i-1}$. Starting from the definition,

$$\begin{aligned} F_i(\ell) &= \sum_{z_{1:i}:z_i=\ell} \pi(z_1) \prod_{j=1}^{i} \phi_j(z_j) \prod_{j=2}^{i} \rho(z_{j-1}, z_j) \\ &= \sum_{\ell'\in\{0,1\}} \sum_{z_{1:i-1}:z_{i-1}=\ell'} \pi(z_1) \prod_{j=1}^{i-1} \phi_j(z_j) \\ &\quad \times \prod_{j=2}^{i-1} \rho(z_{j-1}, z_j) \rho(\ell', \ell) \phi_i(\ell). \end{aligned} \tag{15}$$

The inner sum is exactly $F_{i-1}(\ell')$. Therefore,

$$F_i(\ell) = \phi_i(\ell) \sum_{\ell'\in\{0,1\}} F_{i-1}(\ell')\rho(\ell', \ell). \tag{16}$$

Since the smoothing label is binary, the recursion can be written explicitly. Let $p = p_{\text{stay}}$. Using $\phi_i(1) = s_i$ and $\phi_i(0) = 1 - s_i$, we obtain

$$F_i(1) = s_i \left[pF_{i-1}(1) + (1-p)F_{i-1}(0)\right], \tag{17}$$

and

$$F_i(0) = (1-s_i) \left[pF_{i-1}(0) + (1-p)F_{i-1}(1)\right]. \tag{18}$$

The corresponding initial values are

$$F_1(1) = \frac{1}{2}s_1, \qquad F_1(0) = \frac{1}{2}(1-s_1). \tag{19}$$

A.2 Backward Recursion

The backward message $B_i(\ell)$ is the unnormalized total weight of all suffix smoothing-label sequences from positions $i+1$ to T, conditioned on $z_i = \ell$:

$$B_i(\ell) = \sum_{z_{i+1:T}} \prod_{j=i+1}^{T} \phi_j(z_j) \times \prod_{j=i+1}^{T} \rho(z_{j-1}, z_j), \qquad z_i = \ell. \tag{20}$$

The initialization is

$$B_T(\ell) = 1, \tag{21}$$

which corresponds to the empty product beyond the final token.

For $i = T-1, \ldots, 1$, the recursion is obtained by grouping the suffix sequences according to the next label $\ell' = z_{i+1}$. Starting from the definition,

$$\begin{aligned} B_i(\ell) &= \sum_{z_{i+1:T}} \prod_{j=i+1}^{T} \phi_j(z_j) \prod_{j=i+1}^{T} \rho(z_{j-1}, z_j) \\ &= \sum_{\ell'\in\{0,1\}} \sum_{z_{i+2:T}} \phi_{i+1}(\ell')\rho(\ell, \ell') \\ &\quad \times \prod_{j=i+2}^{T} \phi_j(z_j) \prod_{j=i+2}^{T} \rho(z_{j-1}, z_j). \end{aligned} \tag{22}$$

The inner sum is exactly $B_{i+1}(\ell')$. Therefore,

$$B_i(\ell) = \sum_{\ell'\in\{0,1\}} \rho(\ell, \ell')\phi_{i+1}(\ell')B_{i+1}(\ell'). \tag{23}$$

For the binary case, this recursion is equivalently

$$B_i(1) = ps_{i+1}B_{i+1}(1) + (1-p)(1-s_{i+1})B_{i+1}(0), \tag{24}$$

and

$$B_i(0) = p(1-s_{i+1})B_{i+1}(0) + (1-p)s_{i+1}B_{i+1}(1). \tag{25}$$

The terminal values are

$$B_T(1) = 1, \qquad B_T(0) = 1. \tag{26}$$

A.3 Posterior Marginal

The posterior marginal for smoothing label ℓ at position i is

$$q_i(\ell) = P(z_i = \ell \mid s_{1:T}) = \sum_{z_{1:T}:z_i=\ell} P(z_{1:T} \mid s_{1:T}). \tag{27}$$

Substituting the normalized sequence weight gives

$$q_i(\ell) = \frac{1}{Z(s_{1:T})} \sum_{z_{1:T}:z_i=\ell} \pi(z_1) \prod_{j=1}^{T} \phi_j(z_j) \times \prod_{j=2}^{T} \rho(z_{j-1}, z_j). \tag{28}$$

Because the sequence weight has a first-order linear-chain factorization, fixing $z_i = \ell$ separates the unnormalized weight into prefix and suffix terms:

$$\sum_{z_{1:T}:z_i=\ell} \pi(z_1) \prod_{j=1}^{T} \phi_j(z_j) \prod_{j=2}^{T} \rho(z_{j-1}, z_j) = F_i(\ell) B_i(\ell). \tag{29}$$

Thus,

$$q_i(\ell) = \frac{F_i(\ell) B_i(\ell)}{Z(s_{1:T})}. \tag{30}$$

The normalizing constant can be decomposed at position i:

$$Z(s_{1:T}) = \sum_{\ell' \in \{0,1\}} F_i(\ell') B_i(\ell'). \tag{31}$$

Therefore,

$$q_i(\ell) = \frac{F_i(\ell) B_i(\ell)}{\sum_{\ell' \in \{0,1\}} F_i(\ell') B_i(\ell')}. \tag{32}$$

The final smoothed hallucination score is

$$\tilde{s}_i = q_i(1) = \frac{F_i(1) B_i(1)}{F_i(0) B_i(0) + F_i(1) B_i(1)}. \tag{33}$$

B Sensitivity Analysis of p_{stay}

We analyze the sensitivity of CORTEX to the self-loop probability p_{stay} used in label-persistence smoothing. This parameter controls the strength of span-level smoothing: larger values encourage neighboring tokens to remain in the same smoothing-label variable, whereas smaller values keep the smoothed scores closer to the raw token-level classifier outputs.

Figures 4 and 5 show the results for token-level and answer-level evaluation, respectively. The dashed horizontal lines indicate the raw scores before label-persistence smoothing. Across both datasets and all open-weight LLMs, label-persistence smoothing improves over the raw scores for a wide range of p_{stay}, indicating that the gains of CORTEX are not tied to a single finely tuned value.

For token-level detection, performance is generally stable when p_{stay} is large. On both RAGTruth and HalluRAG, AP increases as p_{stay} becomes larger and then forms a broad plateau around high values such as 0.99, 0.993, and 0.995. AUROC shows a similar trend, with improvements over the raw scores maintained across a wide range of high p_{stay} values. This behavior is consistent with the role of label-persistence smoothing in token-level hallucination detection: hallucinated content typically appears as contiguous spans, and therefore stronger self-transition probabilities help suppress isolated noisy predictions while preserving coherent hallucinated regions.

For answer-level detection, the optimal p_{stay} tends to be smaller than in token-level evaluation. This is because answer-level scores are obtained by aggregating token-level scores with the maximum operator. If p_{stay} is too large, smoothing may overly spread high scores across neighboring tokens or make span boundaries too persistent, which can affect the maximum score used for answer-level classification. Nevertheless, label-persistence smoothing remains beneficial across a broad range of values, and the selected setting $p_{\text{stay}} = 0.93$ provides strong and stable answer-level performance.

Based on these results, we use $p_{\text{stay}} = 0.993$ for token-level evaluation and $p_{\text{stay}} = 0.93$ for answer-level evaluation in the main experiments. We use a single value for each evaluation granularity rather than tuning p_{stay} separately for each dataset and open-weight LLM. This setting keeps the evaluation protocol simple while still capturing the main benefit of label-persistence smoothing: converting locally noisy token-level scores into more span-consistent hallucination estimates.

C Experimental Details

MLP classifier. For all methods that require a supervised classifier, including CORTEX and MLP-based baselines, we use the same three-layer MLP architecture to isolate the effect of the input fea-

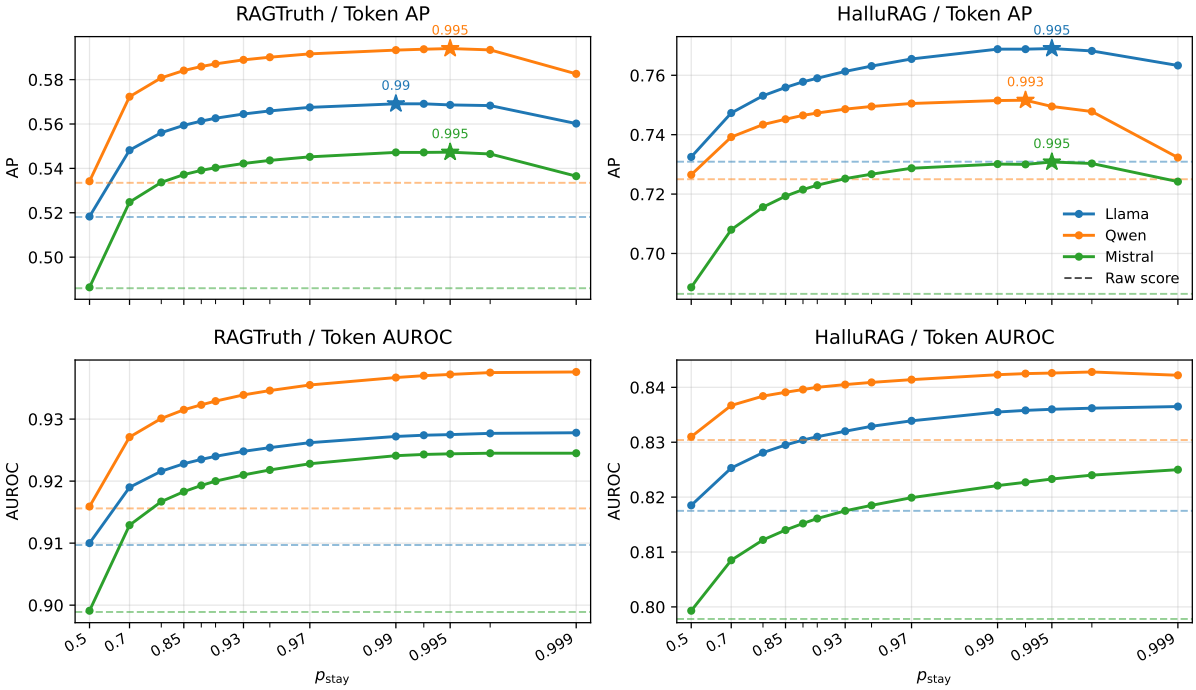

Figure 4: Sensitivity of token-level hallucination detection performance to p_{stay}. Dashed horizontal lines indicate raw scores before label-persistence smoothing.

tures. The classifier consists of two hidden linear layers with 256 and 128 hidden units, respectively, followed by a final linear layer that produces a scalar output logit. Each hidden layer is followed by a ReLU activation and dropout with a rate of 0.1. We train the classifier using AdamW with a learning rate of 1×10^{-3} and a weight decay of 1×10^{-4}. The batch size is set to 4096, and the model is trained for 10 epochs. The same hyperparameters are used across all datasets, models, and methods unless otherwise stated.

Representation extraction. To obtain internal representations from the open-weight analysis model M_{open}, we do not perform autoregressive generation. Instead, we feed the constructed input sequence to M_{open} and extract the internal representations computed in a single forward pass. This implementation matches the post-hoc setting considered in this work: the answer text has already been generated by the closed-weight model, and M_{open} is used only to analyze the given answer.

Computational cost. All experiments are conducted on a single NVIDIA A100 GPU. CORTEX is lightweight in practice: on RAGTruth, feature extraction, classifier training, and evaluation take approximately 20 minutes in total, while on HalluRAG they take approximately 5 minutes in total. This indicates that CORTEX incurs only modest computational overhead while providing token-level hallucination scores.

Artifact Licenses and Use. We use RAGTruth and HalluRAG as existing public evaluation benchmarks and do not redistribute the datasets. RAGTruth is released under the MIT License. HalluRAG is made publicly available by its authors through their repository and dataset DOI, but we did not find an explicit dataset license in the available documentation. All datasets are used solely for research evaluation, and we cite their original sources.

Dataset Documentation. We use only publicly available English-language RAG hallucination benchmarks and do not collect any new data. Our experiments focus on hallucination detection in generated RAG outputs; we do not use, infer, or analyze demographic attributes.

D Effect of Label-persistence Smoothing on Token-Level Baselines

Label-persistence smoothing is a post-processing module that can be applied to any method that produces token-level hallucination scores. To examine whether the gains of CORTEX are due only to label-persistence smoothing procedure to two

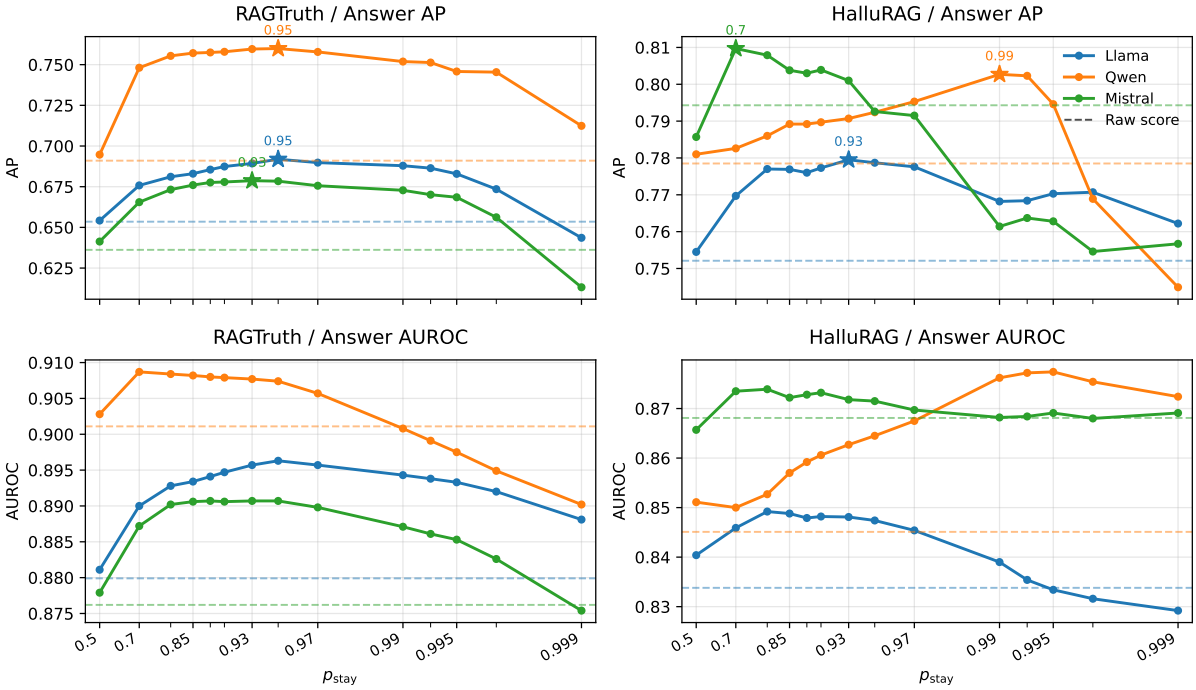

Figure 5: Sensitivity of answer-level hallucination detection performance to p_{stay}. Answer-level scores are obtained by taking the maximum over token-level hallucination scores. Dashed horizontal lines indicate raw scores before label-persistence smoothing.

strong token-level baselines, SAPLMA and RAGLens, and compare them with CORTEX under the same settings. Tables 5 and 6 report the results on RAGTruth and HalluRAG, respectively.

Before applying label-persistence smoothing, CORTEX achieves the best performance in all settings across both datasets and all open-weight LLMs. This result indicates that the comparative representation features of CORTEX already provide a stronger token-level signal than the single-view representation features used by the baselines. After label-persistence smoothing is applied, the performance of SAPLMA and RAGLens improves substantially, confirming that span-level post-processing is broadly useful for reducing local prediction noise. Nevertheless, CORTEX remains strongest in AP across all RAGTruth settings and remains competitive or superior in most HalluRAG settings. These results suggest that label-persistence smoothing is beneficial as a generic post-processing step.

In particular, the comparison before smoothing isolates the effect of the underlying token-level scoring function: CORTEX outperforms the baselines without relying on span-level post-processing. The comparison after smoothing further shows that CORTEX can benefit from the same generic smoothing procedure while retaining the advantage of its paired reference-conditioned and no-reference representation comparison. Thus, label-persistence smoothing and the comparative representation features play complementary roles: the former improves span consistency, whereas the latter provides a stronger reference-grounded hallucination signal.

E Additional Heatmap-Based Ablation Case Studies

We present additional heatmap-based case studies to qualitatively analyze the behavior of each component in CORTEX. The heatmaps compare the ground-truth hallucination spans with the predictions of the full CORTEX model, CORTEX without label-persistence smoothing, and CORTEX without the contextual residual c. Darker colors indicate higher hallucination scores.

Figure 6 shows an example in which the annotated hallucination spans vary in granularity. The ground-truth annotation contains both a broad hallucinated region spanning multiple sentences and more compact hallucinated fragments. Without label-persistence smoothing, CORTEX assigns high scores at relatively fine-grained units. This behavior suggests that the raw token-level classi-

Method	RAGTruth					
	Llama		Qwen		Mistral	
	AP	AUROC	AP	AUROC	AP	AUROC
SAPLMA	0.3894	0.8879	0.4182	0.8820	0.4027	0.8822
+ Smoothing ($P_{\text{stay}} = 0.993$)	0.4910	0.9225	0.5078	0.9163	0.4965	0.9205
+ Smoothing ($P_{\text{stay}} = 0.995$)	0.4925	0.9228	0.5088	0.9168	0.4972	0.9210
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.4938	0.9232	0.5089	0.9174	0.4982	0.9216
RAGLens	0.4271	0.8953	0.4575	0.9075	0.4244	0.8884
+ Smoothing ($P_{\text{stay}} = 0.993$)	0.5191	0.9300	0.5068	0.9286	0.5264	0.9268
+ Smoothing ($P_{\text{stay}} = 0.995$)	0.5191	0.9302	0.5057	0.9286	0.5276	0.9274
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.5191	**0.9305**	0.5026	0.9286	0.5290	**0.9282**
CORTEX	0.5181	0.9097	0.5335	0.9156	0.4860	0.8989
+ Smoothing ($P_{\text{stay}} = 0.993$)	**0.5691**	0.9274	0.5937	0.9370	0.5472	0.9243
+ Smoothing ($P_{\text{stay}} = 0.995$)	0.5686	0.9275	**0.5940**	0.9372	**0.5473**	0.9244
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.5683	0.9277	0.5934	**0.9375**	0.5465	0.9245

Table 5: Effect of label-persistence smoothing on token-level baselines on RAGTruth. Improvements for both SAPLMA and RAGLens indicate that span-level post-processing is broadly useful.

Method	HalluRAG					
	Llama		Qwen		Mistral	
	AP	AUROC	AP	AUROC	AP	AUROC
SAPLMA	0.6875	0.8024	0.6584	0.8129	0.6193	0.7578
+ Smoothing ($P_{\text{stay}} = 0.993$)	0.7527	0.8291	0.7125	0.8378	0.7132	0.8108
+ Smoothing ($P_{\text{stay}} = 0.995$)	0.7539	0.8295	0.7119	0.8383	0.7160	0.8122
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.7546	0.8301	0.7128	0.8391	0.7173	0.8140
RAGLens	0.5725	0.7425	0.6380	0.7639	0.5596	0.7393
+ Smoothing ($P_{\text{stay}} = 0.993$)	0.7146	0.8007	0.7668	0.8138	0.7420	0.8214
+ Smoothing ($P_{\text{stay}} = 0.995$)	0.7169	0.8019	0.7695	0.8148	0.7459	0.8232
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.7202	0.8035	**0.7732**	0.8163	**0.7513**	**0.8256**
CORTEX	0.7309	0.8175	0.7250	0.8304	0.6864	0.7978
+ Smoothing ($P_{\text{stay}} = 0.993$)	0.7688	0.8358	0.7516	0.8425	0.7300	0.8227
+ Smoothing ($P_{\text{stay}} = 0.995$)	**0.7690**	0.8360	0.7495	0.8426	0.7308	0.8233
+ Smoothing ($P_{\text{stay}} = 0.997$)	0.7682	**0.8362**	0.7478	**0.8428**	0.7303	0.8240

Table 6: Effect of applying smoothing to token-level baselines on HalluRAG. Label-persistence smoothing improves all methods substantially, including SAPLMA and RAGLens.

fier can capture localized hallucination signals, but its predictions are fragmented and locally unstable. After label-persistence smoothing, these fragmented high-score regions are connected into a more coherent span, producing predictions that better reflect the span-level nature of hallucination annotations. At the same time, this example also illustrates a trade-off introduced by smoothing: when hallucinated evidence appears in several nearby but distinct regions, label-persistence smoothing may merge them into a broader continuous span. Thus, label-persistence smoothing improves span consistency, but may reduce boundary precision in cases where hallucinated content is interleaved with faithful tokens.

The same example also illustrates the role of the contextual residual c. When c is removed, the model tends to assign high scores more broadly in regions where the current token is influenced by preceding generated context. This behavior is consistent with the motivation for the residual feature: Δh alone captures reference-induced changes at the current token, but it does not explicitly account for reference influence that has already been expressed through earlier answer tokens. By incorporating c, CORTEX can distinguish direct token-level reference sensitivity from deviations relative to the preceding context, leading to more controlled localization.

Figure 7 shows a different type of error case in which the answer includes additional advice that is not supported by the provided references. The

Figure 6: Ablation case study with hallucination spans of different granularities. Without label-persistence smoothing, hallucination scores are more fragmented and localized. With label-persistence smoothing, nearby high-score regions are connected into broader span-consistent predictions, although this can also merge distinct hallucinated fragments into a wider continuous region.

advice is informative and may be reasonable from the model's parametric knowledge, but it is not grounded in the supplied passages. CORTEX assigns high hallucination scores to this region because its objective is to detect whether the answer is supported by the references, not whether the content is generally plausible or useful.

This case highlights an important ambiguity in reference-grounded hallucination detection. In controlled RAG applications, such as enterprise customer support, advice that is not grounded in references can be undesirable or risky even when it is factually plausible, because the system is expected to answer only from those documents. In such settings, flagging unsupported advice is a desirable behavior. In contrast, in open-ended assistant scenarios, suppressing all useful but reference-unsupported content may overly restrict the capabilities of the LLM. Therefore, the appropriate treatment of such content depends on the application: CORTEX should be interpreted as a detector of reference support, rather than a general judge of factual correctness or utility.

This example also clarifies the scope of CORTEX. Because the method compares internal representations under inputs with and without references, it is sensitive to whether a token is grounded in the provided evidence. Consequently, content generated from parametric knowledge alone can receive high hallucination scores if it is not supported by the references. This behavior is aligned with the intended design of CORTEX for reference-grounded generation, but it should be considered when applying the method to settings where answers are allowed to go beyond the retrieved references.

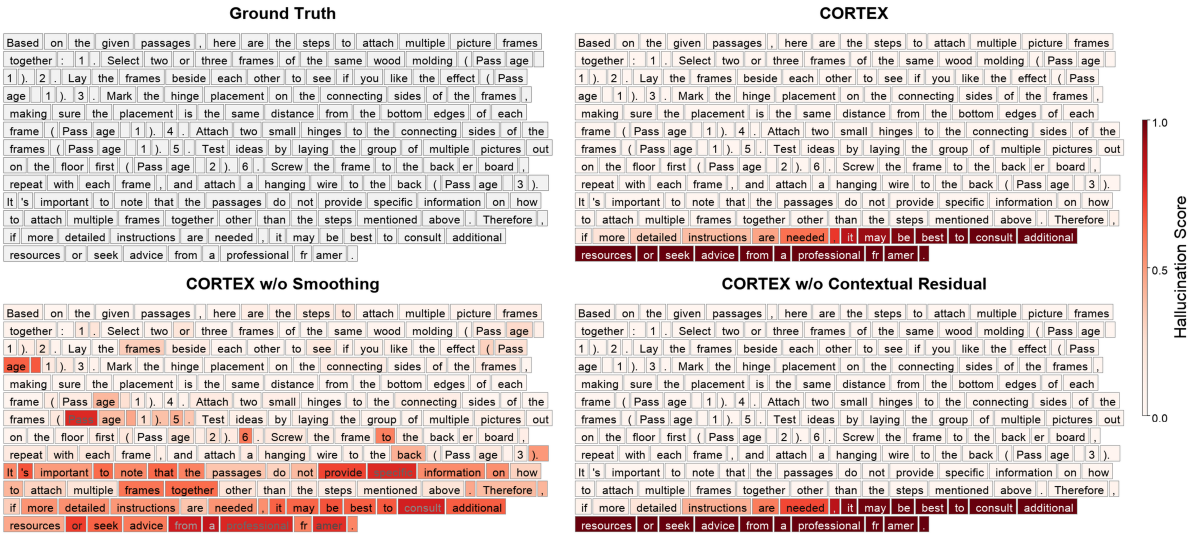

Figure 7: Ablation case study involving groundless advice. The answer includes additional advice that may be informative but is not grounded in the provided references. CORTEX assigns high hallucination scores to this region, reflecting its role as a detector of reference support rather than a general factuality or usefulness judge.