Detecting Hallucinations in Retrieval-Augmented Generation via Semantic-level Internal Reasoning Graph

Anonymous ACL submission

Abstract

The Retrieval-augmented generation (RAG) system based on Large language model (LLM) has made significant progress. It can effectively reduce factuality hallucinations, but faithfulness hallucinations still exist. Previous methods for detecting faithfulness hallucinations either neglect to capture the models' internal reasoning processes or handle those features coarsely, making it difficult for discriminators to learn. This paper proposes a semantic-level internal reasoning graph-based method for detecting faithfulness hallucination. Specifically, we first extend the layer-wise relevance propagation algorithm from the token level to the semantic level, constructing an internal reasoning graph based on attribution vectors. This provides a more faithful semantic-level representation of dependency. Furthermore, we design a general framework based on a small pre-trained language model to utilize the dependencies in LLM's reasoning for training and hallucination detection, which can dynamically adjust the pass rate of correct samples through a threshold. Experimental results demonstrate that our method achieves better overall performance compared to state-of-the-art baselines on RAGTruth and Dolly-15k. Implementation available here: https://anonymous.4open.science/r/SIRG-1022.

1 Introduction

LLMs easily generate grammatically coherent but factually incorrect outputs, a phenomenon commonly referred to as "hallucination" (Mishra et al.; Zhang et al., 2024; Li et al., 2023). Post-learning for downstream tasks or introducing the RAG system (Lewis et al., 2020) can mitigate the factuality hallucinations to some extent, which refer to the tendency of LLMs to produce outputs that are inconsistent with real-world facts. However, due to the inherent knowledge bias of the internal parameters of LLMs, the generated content of RAG may be inconsistent with the context provided by users (Lyu et al., 2024), leading to faithfulness hallucinations. He et al. (2022) has found that faithfulness hallucinations arise primarily from the inconsistency between LLMs' word-level output and true thought process, meaning that LLMs only utilize surface knowledge, such as entity popularity (Lehmann et al., 2025), during reasoning.

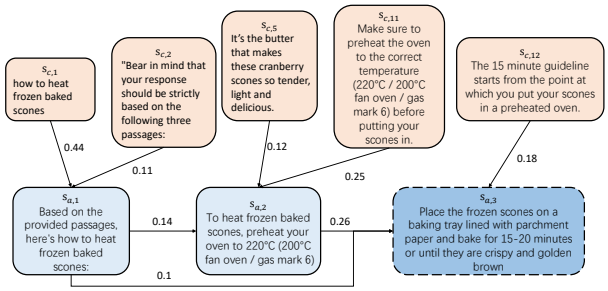


Figure 1: Example of a semantic-level internal reasoning graph. Yellow nodes represent contextual semantic fragments, blue nodes represent semantic fragments of the model's response, and the weight on the edge indicates the contribution degree of the source semantic fragment to the target (with an upper boundary of 1). The dashed box indicates a hallucinated semantic fragment.

To detect faithfulness hallucinations, both Manakul et al. (2023) and Zhao and Zhang (2025) propose post-processing methods with LLMs. However, multiple invocations of the LLM system can lead to significant resource consumption and amplification of model bias. Other scholars (Chen et al., 2025; Wu et al., 2024; Burns et al., 2022) assess hallucinations based on the internal embedding of the model, which often relies on heuristic discrimination trained on these abstract features, resulting in poor interpretability. Hu et al. (2024); Chuang et al. (2024) detect hallucinations from the perspective of output attribution, but their direct accumulation of all token-level attribution vectors for LLMs' responses will introduce substantial noise

(as shown in Appendix A).

To better explain the origin of faithfulness hallucinations, motivated by Phukan et al. (2024), we divide the tokens generated during the autoregressive reasoning of LLMs into linking and substantive tokens. Unlike the original concept, we define linking tokens as non-substantive text in LLM-generated responses that serve to connect contextual information, whereas substantive tokens refer to text that utilizes the contextual information provided by users and reflects the semantic content of the responses. We consider that **hallucinations originate from LLM mistakenly generating substantive tokens as linking**, which manifests on the surface as generation based on entity popularity. Visual analysis from the perspective of attribution score distribution can reveal which substantive tokens LLMs treat as linking tokens, and these differences are difficult to detect from a human perspective because of semantic drift. Specifically, linking tokens rely more on words generated earlier within the same sentence, whereas substantive tokens also depend on words in the long-distance context (detailed in Appendix B).

Since faithfulness hallucinations typically occur at the semantic level, we extend the aforementioned concepts to this level and detect faithfulness hallucinations by our proposed semantic-level internal inference graphs, which can faithfully construct the dependency of the linking and substantive fragments. Specifically, we first employ Layer-wise relevance propagation (LRP) to calculate the score vector attributed to each token during the autoregressive process. This attribution method, which utilizes internal model parameters and predefined rules, faithfully reflects the true computational process within the model. According to the token-level attribution vectors, we model the attribution relationship between contextual semantic fragments and those of LLM's response, forming the internal reasoning graph, as illustrated in Fig. 1. We can observe that the semantic fragments with hallucinations assign higher attribution scores to previously generated semantic fragments in the model's autoregressive reasoning, while showing weaker dependency on the context provided by the user. LLM mistakenly treats substantive fragments as linking during the reasoning process, leading to the hallucination phenomenon.

Based on the above observation, we linearize each response node and attribution dependency in the internal reasoning graph, concatenate them into a prompt, and input it to downstream pretrained language models (PLM) for fine-tuning in binary classification tasks. In the hallucination detection phase, we determine hallucination based on the binary classification label distribution of all semantic fragments in the LLM-generated text. The entire process relies solely on the model with a small number of parameters.

In summary, our contributions mainly include:

- We extend token-level LRP to the semantic level and propose a method for constructing semantic-level internal reasoning graph of LLMs.
- We analyze faithfulness hallucinations from the distribution differences between linking and substantive fragments and utilize internal reasoning graphs to detect them.
- The experimental results on two general datasets demonstrate that the performance of our framework surpasses previous baselines.

2 Related Work

Hallucination Detection LLMs often produce content that is grammatically correct but semantically conflicts with real-world or contextual knowledge, known as hallucinations (Huang et al., 2025). Hallucinations widely exist in downstream reasoning (Vu et al., 2024; Li et al., 2024). Existing hallucination detection methods mainly include: (1) LLMs-based post-verification methods, which detect hallucinations by using multi-agent systems (Nguyen et al., 2025), multiple rounds of self-criticism (Mündler et al., 2024), or consistency of multiple generations (Kuhn et al.); (2) representation-based methods, which identify abnormal states through model hidden states (Bhatia et al., 2025), outputs of attention modules (Simhi et al., 2024), semantic alignment rates (Huang et al., 2023), etc.; (3) task-specific methods, including fine-tuning (Bergeron et al., 2025) or designing task-specific features (Guerreiro et al., 2023). Most of the aforementioned methods rely on explicit reasoning or uninterpretable feature spaces, which may fail to detect semantic biases within the model. Our framework focuses more on faithfully modeling the true reasoning dependencies within LLMs and training a lightweight detector based on them.

Faithful Attribution Our method, grounded in additive interpretability theory (Agarwal et al.,

2021), decomposes model predictions into a sum of contributions from each input, analyzing the significance of source tokens to target tokens based on contribution scores. Perturbation-based method is the most classic attribution algorithm, including SHAP (Lundberg and Lee, 2017), LIME (Ribeiro et al., 2016), AtMan (Deiseroth et al., 2023), etc. These methods exhibit significant computational complexity in the application of LLMs. For the transformer architecture, some scholars utilize attention mechanisms (Abnar and Zuidema, 2020; Chefer et al., 2021) to capture causal relationships. However, these methods lack category specificity and cannot faithfully interpret the final predictions. Based on an improved backpropagation method, the propagation rules are customized between layers to trace back from the model output to the input. Classic methods include Input $\times$ Gradient (Simonyan et al., 2013), LRP (Bach et al., 2015; Voita et al., 2021), SmoothGrad (Smilkov et al., 2017), etc. AttenLRP (Achtibat et al., 2024) employed in this paper is an improved backpropagation method that can effectively handle nonlinear relationships.

3 Methodology

3.1 Task Formulation

Let $D = (Q_i, A_i)_{i=1}^{|D|}$ denote a RAG dataset consisting of $|D|$ samples, where each sample pair comprises a user query Q_i and its corresponding answer A_i. Each answer is based on contexts retrieved from a knowledge base $C = \{C_j\}_{j=1}^{|C|}$, where C_j represents a text block. Given a question Q_i, a retrieval model parameterized by ϕ first retrieves the most relevant text blocks from C to form a prompt, which is then input into a generative model parameterized by θ to generate the answer A_i. The complete process of RAG is defined as follows:

$$P(A_i|Q_i) = P_\phi(C_j|Q_i)P_\theta(A_i|Q_i, C_j)$$
$$P_\theta(A_i|Q_i, C_j) = \prod_{k=1}^{n} P_\theta(a_k|a_1, \cdots, a_{k-1}; C_j; Q_i) \tag{1}$$

where n is the total number of tokens in the target answer A_i. We introduce a discriminator parameterized by γ to determine whether A_i contains hallucinations:

$$P(L_i|Q_i, C_j) = P_\theta(A_i|Q_i, C_j)P_\gamma(L_i|Q_i, C_j, A_i) \tag{2}$$

where label L_i indicates whether the sample exhibits hallucinations. The framework of our method is shown in Fig. 2.

3.2 Contribution Score Calculation

The LRP algorithm is used to calculate the contribution score. The basic assumption of LRP is that a function f_j with N input features $\boldsymbol{x} = \{x_i\}_{i-1}^{N}$ can be decomposed into independent contributions of a single input variable $R_{i\leftarrow j}$, representing the amount of output j that can be attributed to input i. When these contributions are added up, they are proportional to the original function value.

$$f_j(\boldsymbol{x}) \propto R_j = \sum_{i}^{N} R_{i\leftarrow j} \tag{3}$$

The decomposition property of LRP leads to an important conservation property (Achtibat et al., 2024), which ensures that the sum of all contribution scores in each layer remains constant. This feature allows for meaningful and faithful attribution, as the scale of each contribution score can be associated with the output of the original function.

In the generation stage, when generating each token a_k, the generator simultaneously outputs a correlation vector $\vec{r}_k$ based on internal gradients and predefined LRP rules (see Appendix C for details), which is used to quantify the correlation between a_k and each contextual token C_j. By aggregating all correlation scores $\vec{r}_k$ of a token during the generation process, we can obtain a correlation matrix R_i.

3.3 Internal Reasoning Graph Construction

Due to potential semantic biases in the model during training, which is caused by the solidification of pre-trained corpora and the prejudice of the model, the semantic meaning of certain words understood by LLMs may differ from their meaning in the real world. This results in a gap between the reasoning process understood by humans based on chain-of-thought and the reasoning process actually intended by LLMs. That is, there are some thinking outputs that cater to human preferences, while the model's true reasoning process does not rely on these thinking outputs or follow thinking shortcuts. The correlation matrix R_i calculated in the previous step using the LRP is a token-level attribution matrix that is faithful to the internal reasoning process of the model. Therefore, the internal reasoning graph constructed based on R_i can

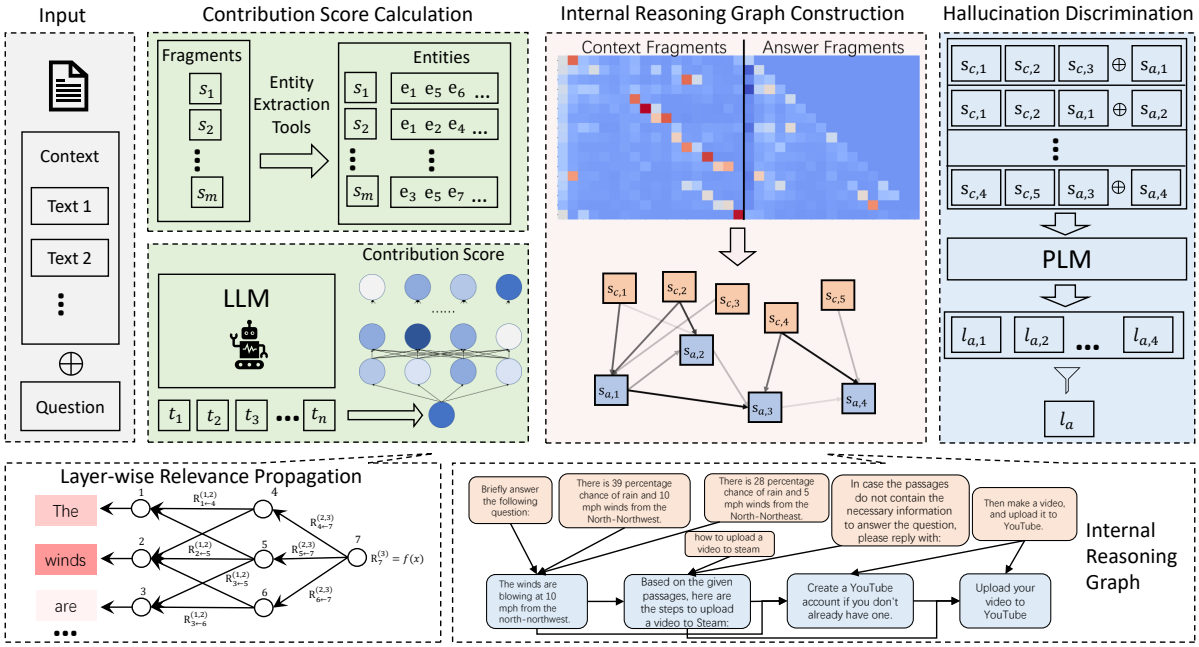


Figure 2: The framework of our method. LRP is first applied to derive a relevance distribution R_i based on the parameters of the LLM. Based on R_i of entities and semantic fragments of input and output content, a semantic-level internal reasoning graph of LLMs is constructed. Subsequently, a PLM is used to determine whether each fragment exhibits semantic conflicts or omissions. Finally, the degree of hallucination occurring in the reasoning graph units is used to determine whether the model's overall response exhibits hallucinations.

faithfully reflect which contextual semantic fragments a certain semantic fragment originates from during the internal inference of the model.

We first recursively use "\n" and sentence tool of Spacy to segment the input context into individual semantic fragments, thereby obtaining a set $S_c = \{s_{c,1}, s_{c,2}, \cdots\}$, where each $s_{c,i}$ represents the ith semantic fragment in that context. For the model's output, we similarly use "\n" to segment, obtaining a set of semantic fragments $S_a = \{s_{a,1}, s_{a,2}, \cdots\}$ contained in one response from the model. The union of these semantic fragments, denoted as $S = S_c \cup S_a$, serves as the node set of the model's internal reasoning graph, that is, the set of atomic steps for model inference.

As mentioned above, the text within a semantic fragment includes both linking content used to connect contexts and make them grammatically correct, and substantive content that specifically reflects the actual meaning and thought expressed in the text. When calculating attribution at the fragment level based on LRP, the attribution scores calculated for these linking fragments introduce a significant amount of noise. To distinguish between semantic fragments that contain rich semantic information and meaningless content, we use a general named entity extraction tool (Spacy and Stanze) to extract entities from the document to the greatest extent possible. The extracted content includes nouns, verbs, noun phrases, negation words, and named entities, which can maximize the reflection of the semantic meaning expressed in the text while filtering out linking content. The set of extracted results, after removing duplicates, is considered the core content $E = \{e_1, e_2, \cdots\}$ that expresses semantic information in the text. Finally, by mapping the set E to each element in the semantic fragment set S, we can obtain the subset of actual meaning $E_{s_i} = \{e\}$ contained in each semantic fragment.

In each target semantic fragment s_i, we only select tokens contained in E_{s_i} to calculate its attribution vector relative to the preceding text. Then, we average all the selected correlation vectors element by element to obtain the token-level attribution vector of the semantic segment relative to the preceding text. For the attributed semantic fragment s_j in the preceding text, we select the maximum value of tokens contained in E_{s_j} in the attribution vector of s_i as the score of s_i's attribution to s_j. When attributing, s_i often only attaches to a small number of tokens with actual meaning in the preceding text. Using an average function can dilute the high correlation information in s_j due to fragment length. Therefore, we use a maximum function to aggre-

gate the correlation vectors:

$$W_{s_{a,i},s_{*,j}} = \begin{cases} \max\limits_{w \in s_{c,j}} (\frac{1}{|E_{s_{a,i}}|} \sum\limits_{e \in E_{s_{a,i}}} \vec{r}_{e,w}) \\ \qquad s_{a,i} \in S_a, s_{c,j} \in S_c \\ \max\limits_{w \in s_{a,j}} (\frac{1}{|E_{s_{a,i}}|} \sum\limits_{e \in E_{s_{a,i}}} \vec{r}_{e,w}) \\ \qquad s_{a,i}, s_{a,j} \in S_a, i > j \\ 0 \quad s_{a,i}, s_{a,j} \in S_a, i \leq j \end{cases} \tag{4}$$

where $\vec{r}_{e,w}$ represents the element associated with token w in the vector corresponding to entity e in the matrix R_i. If the entity consists of multiple tokens, the average of the vectors corresponding to these multiple entities is taken. Through this step, we can obtain the semantic-level correlation matrix $W \in \mathbb{R}^{n_a \times (n_c + n_a)}$, where each element $W_{i,j}$ represents the influence degree of the jth semantic fragment on the ith. As shown in Fig. 2, since subsequent semantic segments do not have an attributive influence on preceding semantic segments, this matrix is a lower triangular matrix from column n_a to column $n_c + n_a$.

The internal reasoning graph $G = \{V, E\}$ is a directed graph that faithfully reflects the dependency relationships between semantic fragments during the internal inference process of the model. The nodes of the graph, $V = S_c \cup S_a$, represent the semantic fragments obtained from the previous context. The edges are associated through the attribution scores between semantic fragments. The weight of each edge is the normalized attribution score. Based on the semantic-level correlation matrix, we propose two methods for constructing intra-inference graphs:

Top k method This strategy first ranks the attribution scores calculated for the target semantic fragments from highest to lowest, then selects the top k fragments as the source, and inserts edges from the source to the target semantic fragments in the graph. The set of incoming edges for node $s_{a,j}$ can be represented as:

$$E_{a,j} = \{(s_{*,i}, s_{a,j}) | Topk(W_{s_{a,j},s_{*,i}}), s_{*,i} \in V\} \tag{5}$$

Adaptive Method The distribution of attribution scores calculated often exhibits a long-tail characteristic. To select edges adaptively, we arrange the attribution scores in descending order and calculate the discrete gradient of the sequence. The maximum discrete gradient point is used to distinguish between important and unimportant source semantic fragments. Assuming that $v_1 \geq v_2 \geq \cdots \geq v_{n_c+n_a}$ is the non-increasing ordering of elements in $W_{s_{a,i}}$, where each v_1 is mapped one-to-one with $s_{*,i}$ using the function $f(\cdot)$, then the set of incoming edges for node $s_{a,j}$ can be represented as follows:

$$m = \arg \max_{1 \leq i \leq n_c + n_a - 1} (W_{s_{a,j}, f(v_i)} - W_{s_{a,j}, f(v_{i+1})})$$
$$E_{a,j} = \{(s_{*,i}, s_{a,j}) | s_{*,i} \in f(v_k), 1 \leq k \leq m\} \tag{6}$$

The union of the incident edge sets of all nodes forms the edge set $E = \{E_{a,j} | 1 \leq j \leq n_a\}$.

3.4 Hallucination Discrimination

As depicted in Fig. 1, in the attribution nodes of hallucinated semantic fragments, a higher proportion of attribution is allocated to the previous semantic fragment answered by the model; whereas the attribution nodes of non-hallucinated semantic fragments tend to be more related to the contextual corpus provided by the user, indicating that they are more faithful to the context provided by humans. Therefore, an important reason for the occurrence of hallucination is that the model treats the next generated semantic fragment as linking content, rather than as substantive content.

To enable the model to discover the attribution distribution differences and semantic differences in these contextual dependencies, we linearize the inference graph into multiple semantic combinations. Specifically, for each semantic fragment $s_{a,j}$ answered by the model, we concatenate all its incoming edges to form a prompt, and feed it into a pre-trained language model (PLM) to obtain the label $l_{a,j}$:

$$l_{a,j} = PLM(\{s_{*,i} | s_{*,i} \in E_{a,j}\} \oplus s_{a,j}) \tag{7}$$

In this paper, we utilize the ALIGNSCORE (Zha et al., 2023), which is based on the RoBERTa architecture (Liu et al., 2019), as the PLM for training, and employ its binary classification inference head for hallucination discrimination. ALIGNSCORE has undergone pretraining on a vast amount of data, focusing on the degree of information alignment between two arbitrary segments. Therefore, by simply fine-tuning it with the downstream cross-entropy loss function, it can exhibit strong hallucination detection capabilities. Ultimately, we employ a flexible threshold α to determine whether

the entire model response exhibits hallucinations. If the proportion of semantic fragments containing hallucinations in the model response exceeds α, then the model response is considered to exhibit hallucinations. Formally, it can be expressed as follows:

$$l_a = \mathbb{I}[\frac{\sum_{j=1}^{n_a} \mathbb{I}[l_{a,j} = 0]}{n_a} \leq \alpha] \quad (8)$$

where $\mathbb{I}$ denotes an indicator function. This paper aligns with previous research, where $l_a = 1$ signifies a correct model response, indicating the absence of hallucination, and $l_a = 0$ denotes an incorrect model response, indicating the presence of hallucination. When $\alpha = 0$, it means that any semantic fragment in the model response that is suspected of being hallucinatory will result in the entire response being classified as hallucination. In practice, the value of α can be adjusted based on the required level of model reliability for specific scenarios.

4 Experiments

4.1 Datasets

We conducted experiments using RAGTruth (Niu et al., 2024) and Dolly-15k (Conover et al., 2023). RAGTruth is a manually annotated RAG sample set generated by various LLMs. The Llama-7B part we used contains 510 hallucination samples and 479 normal samples, while the Llama-13B part contains 399 hallucination samples and 590 normal samples. Dolly-15k is a large model question-answering dataset covering multiple scenarios. We only used the closed question-answering scenarios oriented towards the RAG framework and filtered out samples with empty contexts. Consistent with Hu et al. (2024), GPT-4 is used to compare model outputs with standard answers to annotate the dataset.

4.2 Baselines

Detailed implementation of our method (SIRG) is provided in the Appendix D. We compare SIRG with the following baselines:

Prompt (Niu et al., 2024) Through prompt engineering, we manually design LLM (Llama-7b and GPT-3.5-turbo) prompts to identify hallucinations.

SelfCheckGPT (Manakul et al., 2023) SelfCheckGPT is employed to assess the consistency between sampled responses, calculating the probability of hallucination.

Fine-tune (Niu et al., 2024) We fine-tune Llama-7b and Qwen-7b on the corresponding dataset to detect hallucinations.

EigenScore (Chen et al., 2024) This method utilizes the eigenvalues of the response covariance matrix to measure semantic consistency in the embedding space.

SEP (Kossen et al., 2024) A linear probe trained on the hidden states of LLMs is utilized to detect hallucinations.

LRP4RAG (Hu et al., 2024) This is a method based on LRP, which directly feeds the contribution scores to SVM classifiers or LLMs for hallucination detection.

4.3 Main Results

We employ 3 evaluation metrics to compare SIRG with 9 state-of-the-art baselines on RAGTruth and Dolly-15k, with some results directly sourced from Hu et al. (2024).

Table 1 shows the comparison results on RAGTruth. SIRG has strong performance across all metrics, even outperforming high-resource-consumption methods such as LLMs fine-tuning. On $\text{RAGTruth}_{\text{Llama-7B}}$ and $\text{RAGTruth}_{\text{Llama-13B}}$, SIRG ranks first with the highest F1, achieving improvements of 3.07% and 5.78% over the currently most advanced methods respectively. Prompt-based and self-validation-based methods (SelfCheckGPT) rely on pre-trained LLMs by designing prompts to enable single or multiple rounds of self-correction, which makes them highly unstable. Switching LLMs or prompts can significantly impact downstream tasks. For example, using different prompts with the same gpt-3.5-turbo resulted in a 44.89% difference in recall. Although gpt-3.5-turbo achieves 92.54% recall in the SelfCheckGPT framework, its precision drops to 53.27%, indicating that LLMs blindly classify most samples as correct. This demonstrates that relying solely on pre-training knowledge is insufficient for accurate hallucination detection. The Fintune approach trains LLMs using specific RAGTruth data samples, yet its average performance remains only 62.74% and 36.75%. We attribute this to the insufficient training data scale, which may inadvertently disrupt the general knowledge acquired through fine-tuning. Consequently, the fine-tuning outcomes are inferior to those of direct Prompt-based methods. Both EigenScore and SEP are methods based on

Model	Precision	Recall	F1
RAGTruth$_{\text{Llama-7B}}$			
Prompt$_{\text{llama-7b}}$	52.64%	76.08%	62.23%
Prompt$_{\text{gpt-3.5-turbo}}$	56.91%	47.65%	51.87%
SelfCheckGPT$_{\text{llama-7b}}$	53.32%	83.53%	65.09%
SelfCheckGPT$_{\text{gpt-3.5-turbo}}$	53.27%	**92.54%**	67.62%
Fintune$_{\text{llama-7b}}$	62.50%	65.75%	63.58%
Fintune$_{\text{qwen2-7b}}$	61.76%	64.34%	61.90%
EigenScore	–	74.69%	66.82%
SEP	–	74.77%	66.27%
LRP4RAG$_{\text{LLM}}$	71.18%	75.78%	73.54%
SIRG (Ours)	**73.64%**	79.83%	**76.61%**
RAGTruth$_{\text{Llama-13B}}$			
Prompt$_{\text{llama-7b}}$	41.02%	56.64%	47.58%
Prompt$_{\text{gpt-3.5-turbo}}$	47.58%	44.36%	45.91%
SelfCheckGPT$_{\text{llama-7b}}$	43.66%	75.94%	55.44%
SelfCheckGPT$_{\text{gpt-3.5-turbo}}$	43.01%	**89.47%**	58.10%
Fintune$_{\text{llama-7b}}$	62.50%	27.92%	37.62%
Fintune$_{\text{qwen2-7b}}$	63.55%	25.93%	35.89%
EigenScore	–	67.15%	66.37%
SEP	–	65.80%	71.59%
LRP4RAG$_{\text{LLM}}$	77.14%	74.58%	75.86%
SIRG (Ours)	**78.48%**	85.51%	**81.84%**

Table 1: Overall precision, recall, and F1-score on RAGTruth with Llama-7B and Llama-13B.

Model	Precision	Recall	F1
Dolly$_{\text{Qwen2.5-3B}}$			
Prompt	58.41%	24.98%	34.99%
SelfCheckGPT	67.32%	32.47%	43.88%
EigenScore	68.88%	64.58%	66.66%
SEP	77.94%	79.19%	78.56%
LRP4RAG$_{\text{LLM}}$	**80.55%**	82.91%	81.71%
SIRG (Ours)	72.10%	**95.49%**	**82.17%**
Dolly$_{\text{Qwen2.5-7B}}$			
Prompt	61.46%	47.23%	53.36%
SelfCheckGPT	67.32%	32.47%	43.88%
EigenScore	58.57%	70.03%	63.79%
SEP	76.36%	77.59%	76.97%
LRP4RAG$_{\text{LLM}}$	79.60%	82.20%	80.80%
SIRG (Ours)	**84.21%**	**96.00%**	**89.71%**

Table 2: Overall precision, recall, and F1-score on Dolly-15k with Qwen2.5-3B and Qwen2.5-7B.

vector space discriminators that lack direct contextual semantic information, making it difficult to adequately identify hallucination. LRP4RAG employs token-level aggregation of attribution vectors generated by the LRP algorithm to derive contextual relevance for model responses, representing a coarse-grained approach. This method introduces excessive noise of linking text, resulting in suboptimal performance during classifier training or discriminative tasks using LLMs. Our approach also employs the LRP algorithm, but it enhances the processing of substantive information in response texts by filtering out semantic noise and formally modeling it as a reasoning graph. This facilitates easier training of the downstream discriminator while helping humans understand the decision-making process of LLMs. See Fig.9 in Appendix B for details of the example.

Table 2 presents the comparative results of the Dolly-15k dataset. For the threshold-based benchmark model, we provide its optimal threshold parameters. Since the content generated by the LLM each time is random, fine-tuning to fit fixed responses is pointless in this scenario. On Dolly$_{\text{Qwen2.5-3B}}$ and Dolly$_{\text{Qwen2.5-7B}}$, SIRG achieved F1 scores of 82.17% and 89.17% respectively, surpassing the state-of-the-art method LRP4RAG. Due to the more unstable response of Qwen2.5-3B compared to Qwen2.5-7B, SIRG performs better on Qwen2.5-7B than on Qwen2.5-3B.

4.4 Faithfulness of LRP-based Internal Reasoning Graph

To verify the faithfulness of the internal inference graph constructed by SIRG, the same perturbation tests as (Bakish et al., 2025) are employed (detailed in Appendix E).

We implement token-level blocking of semantic fragments in ten sequential steps based on their relevance, with results presented in Fig. 3. When adding the semantic fragment deemed most relevant by LRP, the most significant changes are observed in the decrease of $(y_0 - y_p)^2$ and the increase of $loghts_k$, indicating that this semantic fragment plays a pivotal role in LLM's computational process. After pruning semantic fragments based on relevance scores, those with lower relevance to the target exhibit negligible impact on both $(y_0 - y_p)^2$ and $loghts_k$, whereas highly relevant fragments demonstrate substantial effects post-pruning. This shows that our algorithm can effectively identify the source semantic fragment, which has a significant impact on the target semantic fragment. If the importance of contributions is randomly assigned, the result curve should change gradually with addition or pruning. Compared with the standard curve of random addition or pruning, our method demonstrates a significant AUC advantage. For

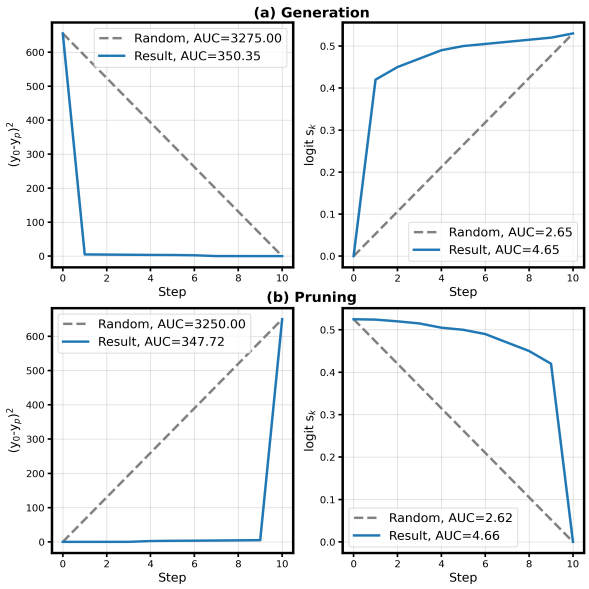


Figure 3: Perturbation tests on RAGTruth with Llama-7B. $(y_0 - y_p)^2$ indicates the change of final embeddings before and after perturbation, while $logits_k$ represents the average probability of the target semantic fragment. The dashed line shows the curve state after random addition or pruning. Perturbation tests are conducted on 100 samples and mean of the above indicators are took.

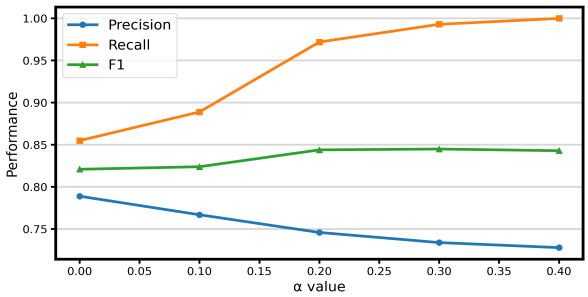


Figure 4: Overall precision, recall, and F1 score of Llama-13B on RAGTruth are evaluated by setting different α values.

Top	1	5	10	15	20	Ada
Precision	79.35%	82.46%	85.71%	83.12%	82.27%	78.98%
Recall	84.82%	87.58%	91.03%	91.72%	89.65%	85.51%
F1	82.00%	84.94%	85.71%	87.21%	85.80%	82.11%

Table 3: Overall precision, recall, and F1 score of Llama-13B on RAGTruth are evaluated by setting different $Topk$. Ada denotes the gradient-based adaptive construction strategy referred in Equation 6.

quantitative comparison of this curve, please refer to the work of Bakish et al. (2025).

4.5 The Impact of Hyperparameters

For the classifier of SIRG, we focus on exploring the impact of α in Equation 8. When $\alpha = 0$, as long as there is one hallucinated semantic fragment, this response will be judged as a hallucinated sample, representing the strictest hallucination detection strategy. As shown in Fig. 4, at this point, SIRG has a relatively low recall for correct samples but a high precision in identifying hallucinations. As the value of α increases, the detection strategy becomes increasingly lenient, so the pass rate for correct samples rises. When $\alpha = 0.4$, the recall for correct samples reaches 100%, but the precision decreases to 72.86%. Although adjusting α greatly impacts recall and precision, it has a relatively weak effect on the F1 score. In different scenarios, we can dynamically adjust α based on the desired pass rate for correct samples.

For the construction of the internal reasoning graph, we regulate the number of source semantic fragments by setting different $Topk$ values. The performance of discriminators trained using graphs generated by various construction strategies is demonstrated in Table 3. As the number of source semantic fragments increases, the discriminator can obtain more semantic information. Consequently, the discriminators' F1 score progressively improves during the initial training phase, reaching its peak at $k = 15$. An increase in k will introduce edge contribution noise into the target semantic fragment, meaning many low-contribution semantic fragments are mistakenly fed to the discriminator for training. The discriminator may misinterpret the conflict between insignificant semantic fragments and target semantic fragments as hallucination, resulting in inferior performance at $k = 20$ compared to $k = 15$. For the adaptive discrete gradient strategy, it tends to only select 1 or 2 source semantic fragments. Without sufficient information, the discriminator's performance is limited.

5 Conclusion

This paper first extends the token-level LRP algorithm to the semantic level within the autoregressive inference paradigm. Then we construct internal reasoning graphs using semantic fragments from RAG contexts and LLMs' responses, which faithfully model the dependencies of the internal reasoning process. Based on that, we propose a framework, SIRG, for identifying faithfulness hallucinations in RAG. SIRG achieves the performance of LLM-based detection frameworks using only a lightweight parameterized discriminator, demonstrating the effectiveness of our approach.

Limitations

For the internal reasoning graph construction of SIRG, since LRP requires computing internal model gradients for each token generation, this results in a high time complexity for the attribution score calculation phase. Future work will optimize LRP's computational objects from semantic fragment perspectives to reduce graph construction time. We will also evaluate the semantic-level faithfulness of various attribution methods in capturing internal inference processes.

For the hallucination detection module of SIRG, although the linearization method has achieved great results, it is a naive way of using topological relation, which ignores the multi-hop dependency information and the subtle error propagation information in the internal reasoning graph. Future efforts will explore multi-angle applications of this graph, including adaptive node relationship aggregation via graph neural networks. Additionally, developing low-resource hallucination discriminators remains a key research focus.

Acknowledgments

We would like to appreciate anonymous reviewers for their valuable comments that help us to improve this manuscript.

Ethics Statement

Based on the research presented in this paper, we acknowledge the ethical implications of developing hallucination detection methods for LLMs. While our work aims to enhance the reliability and trustworthiness of AI-generated content, we recognize that such techniques could potentially be misused to conceal model limitations or manipulate outputs in ways that undermine transparency. We affirm our commitment to responsible AI research by ensuring our method, SIRG, is designed to improve factual faithfulness rather than to deceive. All experiments were conducted using publicly available datasets with proper citations, and we have openly disclosed the limitations of our approach to avoid overstating its capabilities. We encourage the community to utilize this work for promoting accountability and interpretability in LLMs, and we emphasize the importance of continued ethical scrutiny as hallucination detection technologies evolve.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Reduan Achtibat, Sayed Mohammad Vakilzadeh Hatefi, Maximilian Dreyer, Aakriti Jain, Thomas Wiegand, Sebastian Roland Lapuschkin, and Wojciech Samek. 2024. Attnlrp: Attention-aware layer-wise relevance propagation for transformers. In *International Conference on Machine Learning 2024*.

Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. 2021. Neural additive models: Interpretable machine learning with neural nets. *Advances in neural information processing systems*, 34:4699–4711.

Ameen Ali, Thomas Schnake, Oliver Eberle, Grégoire Montavon, Klaus-Robert Müller, and Lior Wolf. 2022. Xai for transformers: Better explanations through conservative propagation. In *International conference on machine learning*, pages 435–451. PMLR.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.

Yarden Bakish, Itamar Zimerman, Hila Chefer, and Lior Wolf. 2025. Revisiting lrp: Positional attribution as the missing ingredient for transformer explainability. *arXiv preprint arXiv:2506.02138*.

Loris Bergeron, Ioana Buhnila, Jérôme François, and Radu State. 2025. Halluguard: Evidence-grounded small reasoning models to mitigate hallucinations in retrieval-augmented generation. *arXiv preprint arXiv:2510.00880*.

Gagan Bhatia, Somayajulu G Sripada, Kevin Allan, and Jacobo Azcona. 2025. Distributional semantics tracing: A framework for explaining hallucinations in large language models. *arXiv preprint arXiv:2510.06107*.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 782–791.

Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024. Inside: Llms' internal states retain the power of hallucination detection. *arXiv preprint arXiv:2402.03744*.

Kang Chen, Yaoning Wang, Kai Xiong, Zhuoka Feng, Wenhe Sun, Haotian Chen, and Yixin Cao. 2025. Do llms signal when they're right? evidence from neuron agreement. *arXiv preprint arXiv:2510.26277*.

Yung-Sung Chuang, Linlu Qiu, Cheng-Yu Hsieh, Ranjay Krishna, Yoon Kim, and James Glass. 2024. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1419–1436.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Björn Deiseroth, Mayukh Deb, Samuel Weinbach, Manuel Brack, Patrick Schramowski, and Kristian Kersting. 2023. Atman: Understanding transformer predictions through memory efficient attention manipulation. *Advances in Neural Information Processing Systems*, 36:63437–63460.

Nuno M Guerreiro, Duarte M Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and André FT Martins. 2023. Hallucinations in large multilingual translation models. *Transactions of the Association for Computational Linguistics*, 11:1500–1517.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Haichuan Hu, Congqing He, Xiaochen Xie, and Quanjun Zhang. 2024. Lrp4rag: Detecting hallucinations in retrieval-augmented generation via layer-wise relevance propagation. *arXiv preprint arXiv:2408.15533*.

Kung-Hsiang Huang, Hou Pong Chan, and Heng Ji. 2023. Zero-shot faithful factual error correction. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

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

Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. 2024. Semantic entropy probes: Robust and cheap hallucination detection in llms. *arXiv preprint arXiv:2406.15927*.

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

Hans Hergen Lehmann, Jae Hee Lee, Steven Schockaert, and Stefan Wermter. 2025. Knowing the facts but choosing the shortcut: Understanding how large language models compare entities. *arXiv preprint arXiv:2510.16815*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024. The dawn after the dark: An empirical study on factuality hallucination in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10879–10899.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2024. Towards faithful model explanation in nlp: A survey. *Computational Linguistics*, 50(2):657–723.

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.

Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. Fine-grained hallucination detection and editing for language models. In *First Conference on Language Modeling*.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. 2024. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*. OpenReview.

Vy Nguyen, Ziqi Xu, Jeffrey Chan, Estrid He, Feng Xia, and Xiuzhen Zhang. 2025. Hallucinate less by thinking more: Aspect-based causal abstention for large language models. *arXiv preprint arXiv:2511.17170*.

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.

Anirudh Phukan, Shwetha Somasundaram, Apoorv Saxena, Koustava Goswami, and Balaji Vasan Srinivasan. 2024. Peering into the mind of language models: An approach for attribution in contextual question answering. In *ACL (Findings)*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Adi Simhi, Jonathan Herzig, Idan Szpektor, and Yonatan Belinkov. 2024. Constructing benchmarks and interventions for combating hallucinations in llms. *arXiv preprint arXiv:2404.09971*.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1126–1140.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and 1 others. 2024. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720.

Di Wu, Jia-Chen Gu, Fan Yin, Nanyun Peng, and Kai-Wei Chang. 2024. Synchronous faithfulness monitoring for trustworthy retrieval-augmented generation. *arXiv preprint arXiv:2406.13692*.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348.

Dongxu Zhang, Varun Gangal, Barrett Lattimer, and Yi Yang. 2024. Enhancing hallucination detection through perturbation-based synthetic data generation in system responses. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13321–13332.

Yaxin Zhao and Yu Zhang. 2025. Halluclean: A unified framework to combat hallucinations in llms. *arXiv preprint arXiv:2511.08916*.

A Coarse-grained Processing

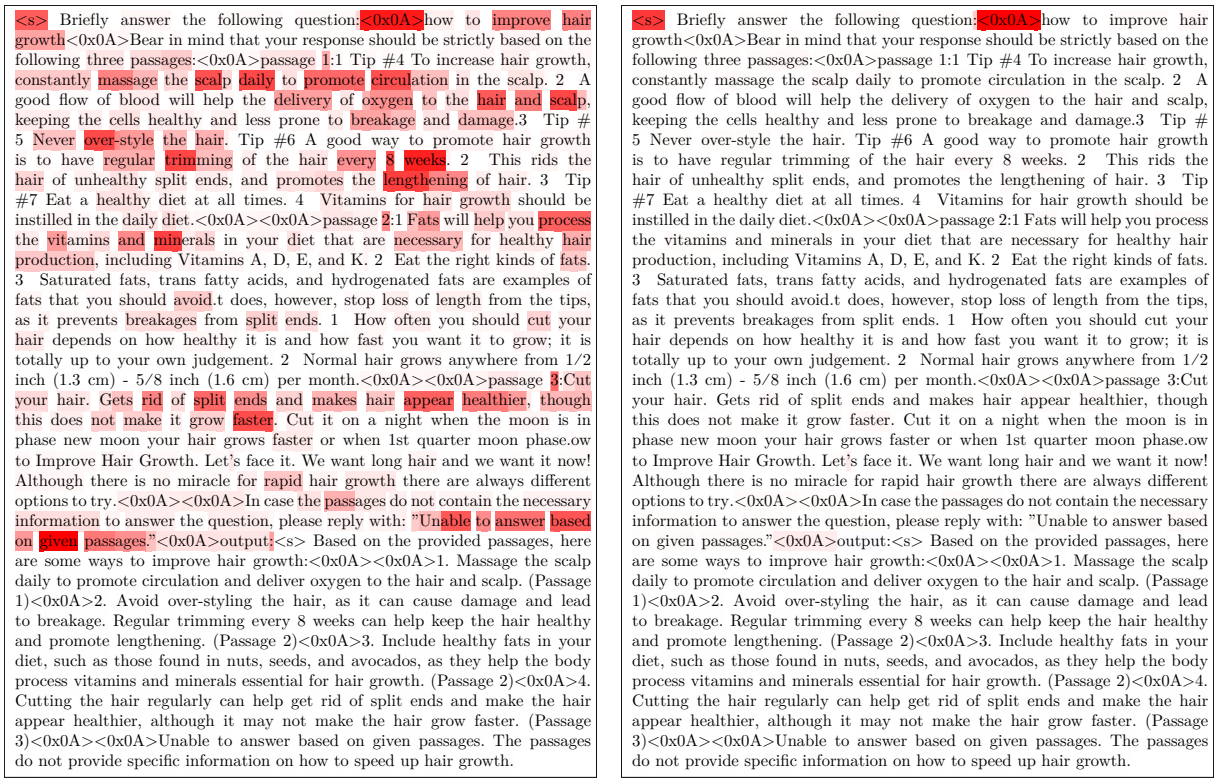
Hu et al. (2024) employs the maximum and average value of the LRP attribution vector for all tokens to get the attribution distributions of the whole response. As shown in Fig. 5, using the maximum value method accumulates the context that contributes the most to each response token, resulting in a noisy contribution distribution. Using the average value method dilutes the context with a high contribution to the substantive word with a large number of linking words, leading to a significantly lower contribution distribution.

B Linking Tokens and Substantive Tokens

In the sentence “The 15 minute guideline starts from the point at which you put your scones in a preheated oven”, “15 minute”, “preheated oven”, etc., contain rich semantic information and are considered as substantive tokens, while “The”, “which”, etc., serve to enhance sentence fluency in the document and are therefore considered linking tokens.

The above description of the linking and substantive token is identified from a human perspective. From an LLM’s standpoint, due to semantic drift and dataset bias during training, the linking and substantive tokens inferred by the LLM may differ from human perceptions. When LLMs generate tokens by treating human-considered substantive content as linking, it results in the faithfulness hallucination. The cognitive gap between humans and LLMs makes this phenomenon hard to detect directly.

We employ the LRP to perform token-level attribution of the next token generated by the LLM, and visualize the results using a heat map. As shown in Fig.6, linking tokens rely more on words generated earlier within the same sentence, whereas substantive tokens also depend on words in the long-distance context. When the model generates hallucination tokens, the heat map in Fig.7 reveals that their attribution distribution exhibits characteristics of linking tokens. From a human perspective,



<s> Briefly answer the following question:<0x0A>how to improve hair growth<0x0A>Bear in mind that your response should be strictly based on the following three passages:<0x0A>passage 1:1 Tip #4 To increase hair growth, constantly massage the scalp daily to promote circulation in the scalp. 2 A good flow of blood will help the delivery of oxygen to the hair and scalp, keeping the cells healthy and less prone to breakage and damage.3 Tip # 5 Never over-style the hair. Tip #6 A good way to promote hair growth is to have regular trimming of the hair every 8 weeks. 2 This rids the hair of unhealthy split ends, and promotes the lengthening of hair. 3 Tip #7 Eat a healthy diet at all times. 4 Vitamins for hair growth should be instilled in the daily diet.<0x0A><0x0A>passage 2:1 Fats will help you process the vitamins and minerals in your diet that are necessary for healthy hair production, including Vitamins A, D, E, and K. 2 Eat the right kinds of fats. 3 Saturated fats, trans fatty acids, and hydrogenated fats are examples of fats that you should avoid.t does, however, stop loss of length from the tips, as it prevents breakages from split ends. 1 How often you should cut your hair depends on how healthy it is and how fast you want it to grow; it is totally up to your own judgement. 2 Normal hair grows anywhere from 1/2 inch (1.3 cm) - 5/8 inch (1.6 cm) per month.<0x0A><0x0A>passage 3:Cut your hair. Gets rid of split ends and makes hair appear healthier, though this does not make it grow faster. Cut it on a night when the moon is in phase new moon your hair grows faster or when 1st quarter moon phase.ow to Improve Hair Growth. Let's face it. We want long hair and we want it now! Although there is no miracle for rapid hair growth there are always different options to try.<0x0A><0x0A>In case the passages do not contain the necessary information to answer the question, please reply with: "Unable to answer based on given passages."<0x0A>output:<s> Based on the provided passages, here are some ways to improve hair growth:<0x0A><0x0A>1. Massage the scalp daily to promote circulation and deliver oxygen to the hair and scalp. (Passage 1)<0x0A>2. Avoid over-styling the hair, as it can cause damage and lead to breakage. Regular trimming every 8 weeks can help keep the hair healthy and promote lengthening. (Passage 2)<0x0A>3. Include healthy fats in your diet, such as those found in nuts, seeds, and avocados, as they help the body process vitamins and minerals essential for hair growth. (Passage 2)<0x0A>4. Cutting the hair regularly can help get rid of split ends and make the hair appear healthier, although it may not make the hair grow faster. (Passage 3)<0x0A><0x0A>Unable to answer based on given passages. The passages do not provide specific information on how to speed up hair growth.

Figure 5: The attribution of LLM responses is obtained by processing attribution features at a coarse granularity. The left figure shows the maximum value of the attribution vector for all tokens in the response, while the right figure shows the average value. The deeper the red color, the greater the token's contribution to the target response.

however, these tokens should be substantive tokens in the RAG system that strictly depend on contextual content. In other words, LLMs process substantive tokens as linking, resulting in faithfulness hallucination.

Faithfulness hallucination is a semantic phenomenon where LLM responses contain content that is semantically inconsistent or contextually absent. When isolated from context, it is challenging to determine whether LLMs exhibit faithfulness hallucination at the token level alone. As shown in Fig.8, the sample's golden label is "Have Hallucination" and the reason is "LOW INTRODUCTION OF NEW INFORMATION. Original: This might be correct, however, the exact way to make sesame milk is not directly stated in the passages. Generative: ...by grinding the seeds into a fine powder and mixing them with water or other liquids. (Passage 2)". However, direct analysis of the superimposed token-level attribution distribution makes it difficult to identify the hallucination fragment. By constructing the target fragment as a semantic-level reasoning graph using our method (as shown in Fig.9), we can easily identify that the information in the target fragment primarily originates from Passage 1, Passage 3, and the LLM's previous generation, indicating a high probability of the faithfulness hallucination.

C LRP Rules

To obtain token-level correlations within LLM, we backpropagate output correlations through each transformer layer.

For the densely connected modules within the transformer architecture, we utilize the local Jacobian matrix $J_{ji} = \frac{\partial z_j}{\partial x_i}$, which represents the gradient of the layer's output z_j with respect to its input x_i, as the weight for assigning correlations. Specifically, for the multi-layer perceptrons within the model, they typically consist of a linear layer followed by an additional nonlinear function:

$$\begin{aligned} z_j &= \sum_i \boldsymbol{W_{ji}} x_i + b_j \\ a_j &= \sigma(z_j) \end{aligned} \tag{9}$$

where W_{ji} represents the model's parameter and σ denotes the nonlinear function. By linearizing the linear layer in Equation 9 at any point $x \in \mathbb{R}^N$, the

Figure 6: Contribution scores of the next generated token. The upper three boxes show the distribution of attribution scores for the next generated linking token, while the lower three boxes display the substantive token. The deeper the red color, the greater the token's contribution to the last token.

basic LRP rule can be derived:

$$R_i^{l-1} = \sum_j \boldsymbol{W_{ji}} x_i \frac{R_j^l}{z_j(\boldsymbol{x}) + \varepsilon} \tag{10}$$

Since element-wise nonlinearity has only a single input variable and output variable, the decomposition of attribution results is the operation itself. Therefore, the total input correlation R_i^l can only be allocated to a single input variable.

$$R_i^{l-1} = R_i^l \tag{11}$$

We apply the Equation 11 to all element-wise operations of single-input and single-output variables.

For the multi-head attention module in the transformer architecture, let Q, K, and V represent the query, key, and value matrices, respectively. The scaled dot product attention calculates the attention weight A and output value O as follows:

$$\begin{aligned} \boldsymbol{A} &= softmax(\frac{\boldsymbol{Q} \cdot \boldsymbol{K}^T}{\sqrt{d_k}}) \\ \boldsymbol{O} &= \boldsymbol{A} \cdot \boldsymbol{V} \\ softmax_j(\boldsymbol{x}) &= \frac{e^{x_j}}{\sum_k e^{x_k}} \end{aligned} \tag{12}$$

Taylor expansion of the softmax function at reference point x yields the following correlation propagation rules:

$$R_i^{l-1} = x_i(R_i^l - s_i \sum_j R_j^l) \tag{13}$$

Among them, s_i represents the i-th output of the softmax function. For the double matrix multiplication part in self-attention computation, since $f(0,0) = 0$ holds, it is necessary to decompose the matrix multiplication into a form without bias terms. For this purpose, we decompose matrix multiplication into affine operations containing summation and bilinear parts containing element-wise multiplication.

$$\boldsymbol{O}_{jp} = \sum_i \boldsymbol{A}_{ji} \boldsymbol{V}_{ip} \tag{14}$$

For the bilinear multiplication summation part mentioned above, we decompose it into element-level multiplication operations with N input variables:

$$f_j(x) = \prod_i^N x_{ji} \tag{15}$$

The following uniform correlation propagation rules can be obtained at the reference point x using

<s> Briefly answer the following question:<0x0A>how do i get copies of my income tax returns<0x0A>Bear in mind that your response should be strictly based on the following three passages:<0x0A>passage 1:Step 4: Enter form used. Enter the form you used to file your federal income tax return on line 6. If you are requesting a personal income tax return, it's likely you filed on Form 1040, 1040A or 1040EZ. You must also enter the tax years you are requesting copies for on line 7.<0x0A><0x0A>passage 2:1 This transcript includes your marital status, the type of return you filed, your adjusted gross income and taxable income. You can get transcripts on the web, by phone or by mail. To request transcripts online, go to IRS.gov and use the Order a Transcript tool.<0x0A><0x0A>passage 3:You may also request a copy of your California tax return by submitting a Request for Copy of Income Tax or Fiduciary Return Form FTB 3516 or by submitting your request on plain paper and provide the following: 1 Your Name (and Spouse if joint return) Social security numbers.<0x0A><0x0A>In case the passages do not contain the necessary information to answer the question, please reply with: "Unable to answer based on given passages."<0x0A>output:<s> Based on the provided passages, here is how you can obtain copies of your income tax returns:<0x0A><0x0A>Passage 1 states that to obtain copies of your income tax returns, you will need to complete Form 4506

<s> Briefly answer the following question:<0x0A>how to heat frozen baked scones<0x0A>Bear in mind that your response should be strictly based on the following three passages:<0x0A>passage 1:There should be little bits of cold butter scattered around the flour. The moment the cold butter pieces bake, they create little pockets of steam and cause the dough to rise – this is similar to what happens when you bake pie crust. It's the butter that makes these cranberry scones so tender, light and delicious.<0x0A><0x0A>passage 2:Scones should be light and flaky, not heavy and flat. We stick to all-purpose flour in our kitchen – it's pretty much our go-to flour. For this recipe, we prefer to use soft, low protein all-purpose flour. Protein levels vary by flour brand. Protein content can affect the tenderness of baked goods.<0x0A><0x0A>passage 3:Make sure to preheat the oven to the correct temperature (220°C / 200°C fan oven / gas mark 6) before putting your scones in. The 15 minute guideline starts from the point at which you put your scones in a preheated oven. It may also take a bit longer to bake them through in a standard (non-fan) or gas oven. You will also need to extend the time if you're baking more than 12 tea-time or 24 canapé scones in the oven at the same time.<0x0A><0x0A>In case the passages do not contain the necessary information to answer the question, please reply with: "Unable to answer based on given passages."<0x0A>output:<s> Based on the provided passages, here's how to heat frozen baked scones:<0x0A><0x0A>To heat frozen baked scones, preheat your oven to 220°C (200°C fan oven / gas mark 6). Place the frozen scones on a baking tray lined with parchment paper and bake for 15-2

Figure 7: The contribution score of the next generated token (labeled as hallucination). The deeper the red color, the greater the token's contribution to the last token.

<s> Briefly answer the following question:<0x0A>how to use sesame seeds<0x0A>Bear in mind that your response should be strictly based on the following three passages:<0x0A>passage 1:1 Add raw sesame seeds to any side dish to add crunch and flavor. 2 For instance, you can add to vegetable or bean dishes. 3 Sesame oil is a great option to try to incorporate in your cooking.4 Like almond milk and hemp milk, you can make your own sesame milk. You can also make a light salad dressing out of tahini. 2 Sesame seeds can also be roasted, crushed and then sprinkled over salads. 3 This will give you more variety and flavor than your typical salad. 4 Add raw sesame seeds to any side dish to add crunch and flavor.<0x0A><0x0A>passage 2:1. Toast over the stovetop. If you don't see any dust or small pebbles in your sesame seeds, you can put them directly in a skillet or frying pan. Heat over medium-low heat, stirring occasionally, for two to three minutes or until the seeds turn brown, glisten, and occasionally pop or jump.1 Do not add oil to the skillet.. Toast over the stovetop. If you don't see any dust or small pebbles in your sesame seeds, you can put them directly in a skillet or frying pan. Heat over medium-low heat, stirring occasionally, for two to three minutes or until the seeds turn brown, glisten, and occasionally pop or jump.<0x0A><0x0A>passage 3:Sesame seeds are tiny, flat oval seeds with a nutty taste and a delicate, almost invisible crunch. They come in a host of different colors, depending upon the variety, including white, yellow, black and red. Sesame seeds are highly valued for their high content of sesame oil, an oil that is very resistant to rancidity.he hulls of sesame seeds contain oxalates. In fact, most of the calcium found in the seed hull comes in the form of calcium oxalate. The sesame seed paste (tahini) found in grocery stores is most often made with seed kernels—the part of the sesame seed that remains after the hull has been removed.<0x0A><0x0A>In case the passages do not contain the necessary information to answer the question, please reply with: "Unable to answer based on given passages."<0x0A>output:<s> Based on the provided passages, here are some ways to use sesame seeds:<0x0A><0x0A>1. Add raw sesame seeds to any side dish to add crunch and flavor. (Passage 1)<0x0A>2. Toast sesame seeds over the stovetop by heating them over medium-low heat, stirring occasionally, for two to three minutes or until they turn brown, glisten, and occasionally pop or jump. (Passage 2)<0x0A>3. Use sesame seeds as a topping for salads, adding a nutty flavor and crunchy texture. (Passage 1)<0x0A>4. Make sesame milk by grinding the seeds into a fine powder and mixing them with water or other liquids. (Passage 2)<0x0A>5. Roast sesame seeds and use them as a topping for salads or other dishes. (Passage 1)<0x0A>6. Use sesame oil in cooking for added flavor and health benefits. (Passage 1)<0x0A>7. Make a light salad dressing using tahini, a sesame seed paste. (Passage 2)<0x0A>8. Add sesame seeds to baked goods, such as breads and muffins, for added flavor and nutrition. (Passage 1)<0x0A>9. Use sesame seeds as a garnish for soups and other dishes. (Passage 1)<0x0A>10. Experiment with different types of sesame seeds, such as white, yellow, black, and red, to find the one that best suits your taste preferences. (Passage 3)<0x0A><0x0A>Unable to answer based on given passages. The passages do not provide information on how to use sesame seeds in specific dishes or recipes.

Figure 8: The superimposed token-level attribution distribution of the response (start with "<s> Based on"). The deeper the red color, the greater the token's contribution to the last token. Based solely on the overlapping attribution distribution, it is difficult to determine whether the response contains hallucination fragments.

the Shapley method (with a baseline of zero) or the Taylor decomposition method:

$$R_{ji}^{l-1} = \frac{1}{N} R_j^l \tag{16}$$

Therefore, based on the correlation propagation calculation formulas of each sub-part in the self-attention module mentioned above, we can obtain the following correlation propagation rules in the self-attention layer:

$$R_{ji}^{l-1} = \sum_p \boldsymbol{A}_{ji} \boldsymbol{V}_{ip} \frac{R_{jp}^l}{2\boldsymbol{O}_{jp} + \varepsilon} \tag{17}$$

There is no bias term for absorption correlation in this rule, and the amount of ε absorption can be ignored. By adopting this rule, we strictly adhere to the conservation property in the process of correlation propagation.

Regarding the correlation propagation rule of LayerNorm layer, when using common $\varepsilon = 10^{-6}$ and $Ver[x] = 1$, it actually absorbs 99% of the correlation. Therefore, linearizing at x is meaningless. By using Taylor expansion to decompose LayerNorm or RMSNorm with reference point 0, we can obtain the same identity association propagation rule as Equation 11.

By using the correlation backpropagation based on the above rules, we can obtain the contribution score of the model input to the probability value output. Intuitively, the higher the contribution score, the greater the impact of the input label on a specific output label. By combining these contribution vectors into a matrix $\boldsymbol{R}_i$, we can obtain a correlation matrix that is faithful to the internal inference process of the model. The proof is detailed in (Achtibat et al., 2024).

D Implementation Details

We employ LXT 2.0 to compute LRP attribution scores for both Llama and Qwen. For substantive word extraction, we utilize Spacy and Stanza to

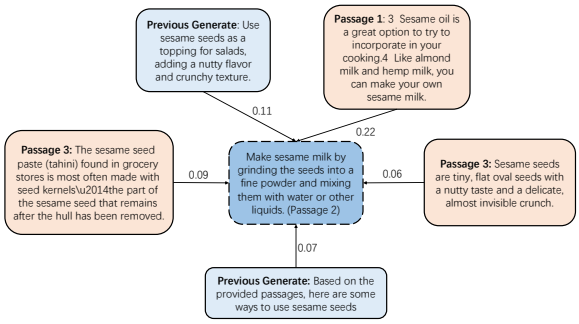


Figure 9: The internal reasoning graph constructed based on the target fragment in the response shown in Fig. 8. The target fragment clearly originates from passage 1, passage 3 and previous generation, contradicting the annotation that it comes from passage 2. Thus, the fragment within the dashed box is most likely a case of faithfulness hallucination.

identify nouns, named entities, noun phrases, and negations in the text. During training, the AlignScore pre-trained Roberta model with 124M parameters is used. The batch size is set to 16, and we perform 100 iterations on the training set using the Adam optimizer at a learning rate of $1e-5$. The model with the best F1 score is selected for the final result.

E Perturbation Tests

The perturbation tests employed by Bakish et al. (2025) are divided into two types of disturbance: generation and pruning. The generation process progressively incorporates semantic fragments starting from null, ordered by relevance from highest to lowest. An approach that faithfully replicates the real decision-making process of LLMs will identify the most impactful semantic fragments. When these fragments are added, the embedding distance of this model is significantly reduced compared to the previous one, and the logit (calculated against the predicted target semantic fragment) shows a marked increase. The pruning method starts by masking the semantic fragments with the lowest relevance and gradually progresses towards the semantic fragments with higher importance. Removing low-impact semantic fragments should preserve model prediction stability. Following Ali et al. (2022), the final metrics were quantified using Area-Under-Curve (AUC), capturing model accuracy relative to the percentage of masked semantic fragments, from 0% to 100%.