SALSA: SALiency-based Source Attribution for RAG Systems

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) systems are being rapidly adopted to provide LLMs with access to up-to-date external knowledge without the need to constantly re-train. A major challenge with the adoption of RAG systems is *user trust* - users need to be able to quickly verify that system responses are factually correct given the retrieved knowledge. Attribution (citation) systems address this need. However, most implementations either rely on hallucination-prone prompting methods or post-hoc analysis which may not reflect the actual information used by the LLM during response generation. We propose an attribution method for RAG systems that requires no special prompting or external evaluation - instead, it relies only on the LLM itself with the original context presented in the user query and the subsequently generated response. Specifically, we use the response loss to compute a saliency map over the entire context, including the retrieved documents. We then derive sets of context spans likely to support or conflict with each sentence in the response. Experiments with end-to-end RAG pipelines show that the proposed saliency-based approach outperforms prompting on granular span attribution while being orders of magnitude more efficient. Additionally, by deriving saliency measurements directly from the LLM, we maximize the likelihood that the cited text actually influenced the response, providing better explainability.

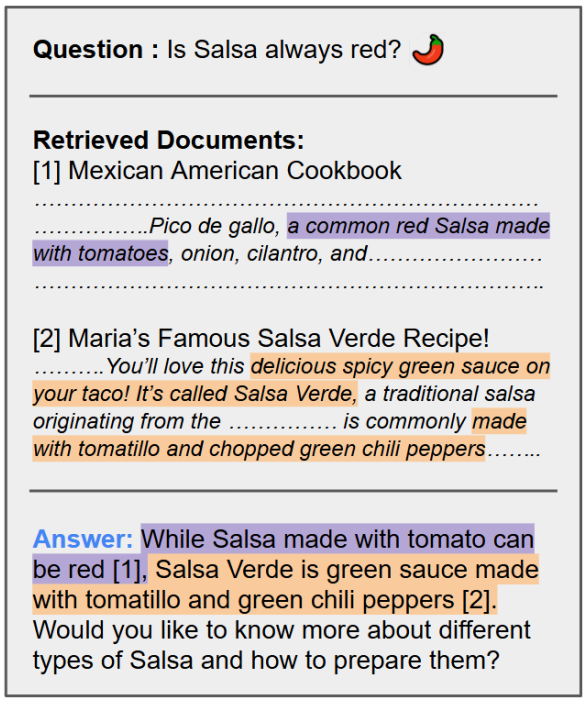


Figure 1: An example of granular attribution in a RAG system. The system not only indicates which source documents ([1] or [2]) support each part of the answer, but also provides token-level highlighting showing the precise correspondence between information in the source documents (highlighted in purple and yellow) and the generated answer text. This granular attribution allows users to trace exactly which parts of each source document contribute to specific segments of the response.

1 Introduction

Retrieval-Augmented Generation (RAG) systems are an important application of Large Language Models (LLM), allowing existing models access to fresh, up-to-date knowledge beyond the scope of their original training data. Originally introduced by Lewis et al. (2020) as an encoder-decoder model jointly trained with a vector-based retriever, RAG has since been widely adopted as a general framework for pairing LLMs with search capability (Borgeaud et al., 2022). A standard modern RAG implementation will include an LLM (e.g., Llama3.1 (Dubey et al., 2024), GPT-4 (OpenAI, 2023)) that is instructed to issue a query to a search engine based on the user's request. The results are ranked by similarity to the request and inserted into the LLM context, after which the LLM can condition on them while generating its response. Search engines typically used by RAG systems include internet search (e.g., Google, Bing), structured knowledge graphs (e.g., Wikidata), and vector-

based retrievers (e.g., ChromaDB Chroma (2024), FAISS Douze et al. (2024)), bringing a wide variety of external knowledge sources within reach of the LLM.

A key challenge that remains with RAG systems is the fact that the LLM is free to use its context in any capacity. This means that it can ignore retrieved information in favor of generating a response conditioned by its own pre-training, or it can mistakenly combine unrelated facts from different retrieved chunks. Additionally, if the retrieved results are not relevant to the user's query, the LLM may use them to generate a misinformed response. In order for users to *trust* the response from a RAG system, they must be able to quickly and effortlessly verify that: (a) the LLM grounded its response to a relevant passage within the retrieved documents, and (b) the grounding passage is relevant and from a trustworthy source. While (b) is up to user discretion and can be aided by user-centric interface design, (a) presents the technical challenge of **response attribution** to the LLM's context. Specifically, the user needs to know which spans of text in the context were referenced by the LLM while generating its response, and which retrieved documents those spans originate from.

To address this need, RAG system developers typically employ post-hoc attribution (citation) systems that attempt to match each response sentence to their most likely source in the retrieved context. Typical post-hoc attribution methods include prompt-based approaches, textual similarity metrics, and Natural Language Inference (NLI). Prompting approaches include asking the LLM to rank each response-document pair for attributability or to directly generate in-line citations (Gao et al., 2023). Textual similarity metrics include n-gram overlap metrics (e.g., BLEU, ROUGE) or sentence embedding cosine similarity to identify the documents that contain the highest content overlap with the response. Natural Language Inference (NLI) models predict whether each retrieved document is likely to entail the response(Honovich et al., 2022).

However, none of these post-hoc methods show *what exact information* was likely used by the LLM during response generation; rather, they make a "best-guess" attempt at identifying sources that could reasonably support the response. Furthermore, post-hoc attribution methods that rely on prompting are prone to hallucination in the same manner as the RAG systems they are there to explain, leading to the potential for compounding hallucinations.

To address this, we turn to **saliency-based** methods for context attribution, a paradigm that has long been used for explainability of neural models in NLP and Computer Vision. Specifically, we keep the original context and generated response presented during the user's interaction with the LLM and compute a saliency map over the context with respect to each sentence in the response. The saliency map shows the **positive or negative** impact of each context token on the LLM-assigned likelihood of the response sentence, intuitively indicating whether the presence of each token **supports** or **conflicts with** the prediction of that sentence. If the salient tokens fall within the bounds of the a document, this indicates that we can cite that document as a supporting or conflicting source (Fig. 1); if the salient tokens fall outside the bounds of any retrieved document, this indicates a **missing citation**.

Importantly, unlike current RAG attribution benchmarks that simply require the correct documents to be cited, we present the first (to our knowledge) formulation of the RAG attribution task that requires the ***correct documents to be cited correctly***: that is, salient token spans within a document must sufficiently overlap with and not exceed ground-truth spans for the citation to count.

Thus we propose **SALSA: SALiency-based Source Attribution for RAG Systems**[1], with the following contributions:

1. **Sliding-Window saliency method** with dynamic Z-thresholding for supporting and conflicting span extraction.

2. **Human-annotated dataset** of supporting and conflicting document spans corresponding to the ELI5 (Fan et al., 2019) portion of the ALCE RAG attribution benchmark (Gao et al., 2023).

3. **Span-level and document-level evaluation** of an end-to-end RAG pipeline using LLMs from different families and at different scales.

2 Methods

We now describe our saliency-based method as applied to a RAG system. Let $\mathbf{c} = (c_1, \ldots, c_n)$ be

[1]Our code and utilized datasets are available on GitHub for reproducibility: https://anonymous.4open.science/r/salsa-citation

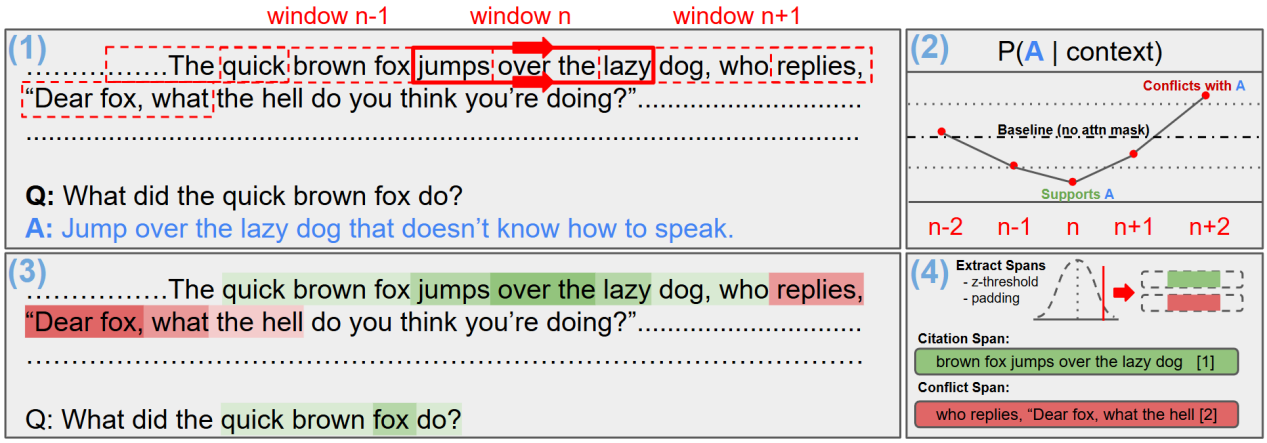


Figure 2: Overview of the sliding-window saliency (§2.1) and span extraction (§2.2) methods. (1) A sliding attention mask window with configurable size and overlap traverses the context. For illustration purposes, we show a window size of 4 with an overlap of 2. (2) Each window position receives a saliency score by measuring how masking its text affects $P(A \mid \text{context})$: a decrease in probability indicates the masked text supports answer A, while an increase indicates it conflicts with A. (3) Token-level saliency scores are computed by averaging across all windows that contain each token and smoothing using a 1d convolution. (4) Finally, attribution spans are extracted by separating saliency scores into positive (supporting) and negative (conflicting) bins, selecting the tokens from each bin with saliency scores exceeding the z-threshold. The tokens are padded into contiguous citation [1] and conflict [2] spans.

the context tokens which include all retrieved documents $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_z)$. Let $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_m)$ be a set of response **sentences** generated by the language model M. For each sentence $\mathbf{a}_s \in \mathbf{A}$, our objective is to identify spans from $\mathbf{D}$ that **support** or **conflict with** $\mathbf{a}_s$.

2.1 Sliding-Window Saliency

For context $\mathbf{c}$ with n tokens, let $\mathbf{w_k} = (c_k, \ldots, c_{\min(k+w-1,n)})$ be a sliding window of size $w \ll n$ tokens with an overlap of $o < w$ tokens. This yields $l = 1 + \left\lceil \frac{n-w}{w-o} \right\rceil$ overlapping windows over $\mathbf{c}$ such that

$$k \in \{j(w-o)+1 : j \in \mathbb{Z}; 0 \leq j < l\}. \quad (1)$$

Also, let $\mathcal{L}(\mathbf{a}_s, \mathbf{a}_{t<s}, \mathbf{c})$ be the negative log-likelihood loss of language model M for response sentence $\mathbf{a}_s$ given response sentences $\mathbf{a}_{t<s}$ and context $\mathbf{c}$. [2]

First, we compute the *relative loss* δ_k for each window $\mathbf{w_k}$ as:

$$\delta_k = \mathcal{L}(\mathbf{a}_s, \mathbf{a}_{t<s}, \mathbf{c} \setminus \mathbf{w_k}) - \mathcal{L}(\mathbf{a}_s, \mathbf{a}_{t<s}, \mathbf{c}) \quad (2)$$

where $\mathbf{c} \setminus \mathbf{w_k}$ denotes the context with window $\mathbf{w_k}$ hidden using an attention mask. Intuitively, δ_k represents the impact of hiding the tokens in $\mathbf{w_k}$ on the likelihood of response sentence $\mathbf{a}_s$: A positive δ_k value indicates that hiding $\mathbf{w_k}$ makes $\mathbf{a}_s$ less likely, meaning that the tokens in $\mathbf{w_k}$ are important context for the generation of $\mathbf{a}_s$. Likewise, a negative δ_k value indicates that $\mathbf{w_k}$ harbors distracting context which possibly conflicts with $\mathbf{a}_s$. A δ_k value near zero indicates that $\mathbf{w_k}$ is likely inconsequential to $\mathbf{a}_s$.

Next, the saliency score s_i for each context token c_i is defined as the average of the relative losses for all windows containing c_i:

$$s_i = \frac{1}{|\{k : c_i \in \mathbf{w_k}\}|} \sum_{k:c_i \in \mathbf{w_k}} \delta_k \quad (3)$$

Note that when overlap o is zero, each token c_i will only appear in one window and thus Eq. 3 simplifies to $s_i = \delta_k : c_i \in \mathbf{w_k}$.

Finally, we smooth the token-level saliency scores using a one-dimensional convolution with a constant kernel of size $\omega > 1$, where the constant $\lambda = \frac{1}{\omega}$ averages each token's saliency with its $\omega - 1$ nearest neighbors. Smoothing helps avoid span fragmentation in areas with multiple close-by but non-contiguous high-saliency tokens.

2.2 Salient Span Extraction

To extract the most relevant contiguous regions of the context for the generated response, we normalize all saliency scores to have a zero mean and unit variance (z-score) and select tokens with saliency

[2] For a detailed explanation of how the loss function $\mathcal{L}(\mathbf{a}_s, \mathbf{a}_{t<s}, \mathbf{c})$ is computed for a specific sentence $\mathbf{a}_s$ as opposed to the entire response $\mathbf{A}$, see Appendix A.

above a given z-threshold to form the supporting and conflicting attribution spans. To avoid the need to select a good z-threshold, we implement **dynamic z-thresholding**: we set $z = 2\exp\left(\frac{S_s}{|\mathbf{c}|}\right)$ where S_s is the shannon entropy of the saliency scores and $|\mathbf{c}|$ is the total number of tokens in the context. Intuitively, this scales the baseline z-threshold of 2 by the entropy of the saliency scores normalized by the context length: tokens in contexts with dispersed saliency (e.g. multiple candidate attribution spans) require higher saliency values to qualify for attribution than contexts with focused saliency (e.g., one clear candidate span).

2.3 Document Attribution

We attribute each response sentence $\mathbf{a}_s$ to zero or more documents in $\mathbf{D}$ based on the candidate supporting spans $\mathbf{S}^+$, and zero or more documents in $\mathbf{D}$ based on the candidate conflicting spans $\mathbf{S}^-$. The set of supporting documents $\mathbf{D_a}$ for a response statement $\mathbf{a}_s$ is then:

$$\mathbf{D_a} = \bigcup_{\mathbf{r} \in \mathbf{S}^+} \mathbf{d} \in \mathbf{D} : \mathbf{r} \subseteq \mathbf{d} \tag{4}$$

Similarly, the set of conflicting documents $\mathbf{D_c}$ for a response statement $\mathbf{a}_s$ is:

$$\mathbf{D_c} = \bigcup_{\mathbf{r} \in \mathbf{S}^-} \mathbf{d} \in \mathbf{D} : \mathbf{r} \subseteq \mathbf{d} \tag{5}$$

3 Experimental Setup

The goal of our experiments are to: (a) determine if saliency-based attribution is competitive with a traditional prompt-based attribution approach; (b) validate that saliency-based attribution is robust across different LLM families and scales; and (c) compare these approaches from the lens of computational efficiency, since RAG systems are typically deployed to many concurrent users at scale.

When measuring performance we adhere to a strict interpretation of the RAG attribution task: it is not enough that the correct documents are cited; rather each document must also be cited *correctly*, e.g. the system can identify exactly what content *inside* the document enables its use as a supporting source. Thus, we construct a new dataset of character-level ground-truth citation spans for documents used in the ALCE RAG benchmark (Gao et al., 2023) for our experiments.

3.1 Data Annotation

We construct our dataset based on the ELI5 Q&A corpus (Fan et al., 2019), using its RAG-adapted version (Gao et al., 2023) which pairs questions with their top-5 retrieved documents from filtered Common Crawl. ELI5 is particularly suitable for RAG evaluation due to its open-ended questions that require multi-sentence answers. We select a sample of 100 question, answer, document tuples from the ELI5 dataset to annotate. Annotations were collected using a web application built using the Streamlit library. [3] We recruited two computer science students from a 4-year institution as annotators. Each annotator was trained to identify supporting and conflicting spans within source documents that either substantiate or contradict given answer statements. Following the guidelines (see Appendix F), annotators were instructed to select minimal contiguous text spans that conveyed the necessary semantic meaning, allowing for multiple span selections when information was distributed across documents.

3.2 Evaluation Metrics

We develop a comprehensive evaluation pipeline with two main components: granular character-level evaluation and document-level attribution assessment.

Character-level span evaluations measure the overlap between candidate spans $\mathbf{S}$ and ground truth spans $\mathbf{G}$ using the binary classification metrics precision, recall, and F1:

$$\text{Precision} = \frac{|\mathbf{S} \cap \mathbf{G}|}{|\mathbf{S}|} \tag{6}$$

$$\text{Recall} = \frac{|\mathbf{S} \cap \mathbf{G}|}{|\mathbf{G}|} \tag{7}$$

where $|\mathbf{S} \cap \mathbf{G}|$ represents the number of characters that overlap between the predicted and ground truth spans, $|\mathbf{S}|$ is the total number of characters in the predicted spans, and $|\mathbf{G}|$ is the total number of characters in the ground truth spans.

Document-level evaluations assess the quality of document attribution by comparing the predicted document set $\mathbf{D_a}$ with the ground truth document set $\mathbf{D_g}$ using Precision, Recall, and F1 metrics. Importantly, we only consider a document citation as a true positive if both: (1) the document is correctly cited, and (2) the corresponding text span

[3] https://streamlit.io/

Model	Method	Document-level ↑			Character-level ↑			Avg. Time (sec) ↓
		Prec.	Rec.	F1	Prec.	Rec.	F1	
Mistral-7B-Instruct-v0.1	Prompt-based	0.00	0.00	0.00	0.00	0.00	0.00	64.4
	SALSA$_{\text{BASE}}$	0.52	0.37	0.44	0.39	0.38	0.38	**6.5**
	SALSA$_{+\text{T}}$	0.57	0.36	0.44	0.41	0.37	0.39	
	SALSA$_{+\text{S}}$	0.53	0.37	0.44	0.39	0.38	0.39	
	SALSA$_{+\text{TS}}$	0.61	0.37	**0.46**	0.43	0.38	**0.40**	
Llama-3.1-8B-Instruct	Prompt-based	0.38	0.13	0.19	0.26	0.15	0.19	41.0
	SALSA$_{\text{BASE}}$	0.51	0.38	0.43	0.38	0.40	0.39	**6.0**
	SALSA$_{+\text{T}}$	0.56	0.37	**0.44**	0.40	0.39	0.39	
	SALSA$_{+\text{S}}$	0.54	0.38	**0.44**	0.38	0.40	0.39	
	SALSA$_{+\text{TS}}$	0.58	0.35	**0.44**	0.41	0.39	**0.40**	
Llama-3.1-70B-Instruct	Prompt-based	0.26	0.53	0.35	0.23	0.48	0.31	175.7
	SALSA$_{\text{BASE}}$	0.51	0.41	0.45	0.35	0.42	**0.39**	**22.8**
	SALSA$_{+\text{T}}$	0.54	0.40	**0.46**	0.37	0.42	**0.39**	
	SALSA$_{+\text{S}}$	0.52	0.40	0.45	0.35	0.42	0.38	
	SALSA$_{+\text{TS}}$	0.55	0.39	0.45	0.37	0.41	**0.39**	

Table 1: Answer attribution quality comparison showing both document-level and span-level metrics (Precision, Recall, F1). Unless otherwise specified, all SALSA experiments use sliding window size $w = 7$, overlap $o = 2$, z-threshold $z = 4.0$, and padding $p = 7$. Ablations: SALSA$_{+\text{T}}$: adds dynamic z-thresholding; SALSA$_{+\text{S}}$: adds smoothing ($\omega = 7$); SALSA$_{+\text{TS}}$: adds dynamic z-thresholding and smoothing ($\omega = 7$). Prompt-based baseline generates up to 2048 tokens with temp= 0.6 and top_p= 0.9. Best F1 scores and Times for each model are in **bold**.

achieves an F1 score above a threshold of 0.5 with the annotated ground truth span.

We note that it is also appropriate to consider using Intersection over Union (IoU) as a metric for this task as described in Appendix C; however, we select character-level P, R, and F1 since over IoU as it allows us to examine whether the systems over-cite or under-cite on the span-level.

3.3 RAG Pipelines

We run our experiments in an end-to-end RAG setting, where a language model generates an answer based on the retrieved documents. Specifically, we generate all answers ahead of time using the `Llama-3.1-70B-Instruct` language model. We select this language model since it's the best LLM we can run given our GPU constraints. Retrieved documents are provided along with each question in the ALCE ELI5 dataset - we use these for our evaluation instead of implementing a full retrieval pipeline (e.g., chunking, embedding, storage, approximate-nearest-neighbor search).

We evaluate three LLMs as the RAG model: `Mistral-7B-Instruct-v0.1`, `Llama-3.1-8B-Instruct`, and `Llama-3.1-70B-Instruct`. These models represent two families and scales of LLM. Each LLM is evaluated in two modes: (a) as a **Prompt-based baseline** and (b) as the **SALSA** system.

When running as the prompt-based baseline, the LLM is prompted with each question/document/answer sentence triple and instructed to rewrite the document verbatim, placing tags around text that supports or conflicts with the answer (the exact prompt is shown in Appendix E.2). The tags are then matched with a Regex to extract the attribution spans. Importantly, prompt-based baseline must generate **2 $\times$ number of documents $\times$ number of answer sentences** (approx. 50) document rewrites to predict supporting and conflicting attributions for each answer sentence.

When running as the SALSA system, the LLM is prompted as a RAG system for answer generation (the expact prompt is shown in Appendix E.1). However, the pre-generated answer is appended to the prompt and no actual generation is done. Instead, the sliding window saliency method described in Section 2 is applied to extract all supporting and conflicting attribution spans in one shot (requiring only $l+1$ model forward passes; see Section 2.1).

All of the experiments were run using a Linux computer system with 2xA100 80GB GPUs and 2xA6000 GPUs. The experiments in total took $\sim$ 24 hours on this system.

4 Experimental Results

We evaluate our Saliency-based attribution method on the annotated ELI5 dataset.

Table 1 shows the performance of our Saliency-based method compared to the prompt-based base-

line. Experimental hyperparameters and ablations are described beneath the results. Specifically, each SALSA pipeline is run with static thresholding and no smoothing (**BASE**), with dynamic z-thresholding (**+T**), with smoothing (**+S**), and with both dynamic z-thresholding and smoothing (**+TS**).

We find that SALSA consistently outperforms the prompt-based baseline for all LLMs we test, and by a wide margin: for example, SALSA achieves a document-level F1 of 0.44 vs. 0.19 from the prompt-based baseline using `Llama-3.1-8B-Instruct`. We observe that `Mistral-7B-Instruct-v0.1` completely fails to follow the prompt-based instructions, yielding no valid document rewrites with extractable spans and a consequent F1 score of 0 at all levels. `Llama-3.1-70B-Instruct` was the only LLM able to reasonably follow the document-rewriting prompt, however was still not competitive with SALSA (0.46 F1 vs. 0.35 document-level F1).

Overall, we find that the effectiveness of the prompt-based baseline is directly tied to the scale class of the LLM (e.g., 70B scale vs 8B scale). In contrast, **SALSA provides approximately the same robust performance across all LLMs**.

In terms of computation, SALSA is orders-of-magnitude more efficient than the prompt-based baseline (7-9x faster), a benefit for at-scale deployment.

4.1 Effect of Hyperparameters

We measure the effect of a range of choices for sliding window size w, overlap o, smoothing kernel size ω, z-threshold z, and padding length p. Each range ablation only varies one hyperparameter, keeping all others constant. We find that SALSA gives consistent results across most values of w, o, and ω, while adjusting z and p involves a precision-recall trade-off (Figures 3, 4). Despite this trade-off, we show that dynamic z-thresholding yields near-optimal performance without needing to manually select a value for z.

5 Related Work

Attribution in language models refers to the task of identifying and verifying which portions of source documents support or conflict with model-generated text. This capability is crucial for building trustworthy AI systems, as it allows users to verify claims and trace information to its origins. Prior approaches to this challenge broadly fall into three categories: fine-tuning models for attribution capabilities, developing prompting strategies, and applying saliency techniques to identify influential context.

► **Fine-tuning Based Attribution.** Tahaei et al. (2024) introduces FiDCiter, showing that targeted fine-tuning of a FLAN-T5 model with a Fusion-in-Decoder (FiD) architecture (Izacard and Grave, 2020) can enhance both answer quality and citation verification capabilities enabling a much smaller FLAN-T5 model (3B) to perform comparably with a much larger Llama-13B. Xia et al. (2024) proposed ReClaim, a more granular approach that uses two fine-tuned language models that interleave reference and answer generation at the sentence level, enabling finer attribution control. However, a key challenge in developing these systems is the scarcity of high-quality training data for multi-source attribution. Patel et al. (2024) addressed this challenge by introducing MultiAttr, a method for transforming existing QA datasets into attribution-focused training data. Their work demonstrated that fine-tuning on such transformed data yields significant improvements across multiple attribution benchmarks compared to domain-specific training alone. While fine-tuning can improve attribution capabilities, these methods typically require substantial training data and may not generalize well across domains. Additionally, Yue et al. (2023) showed that reliable evaluation of attribution quality remains an open challenge that warrants further investigation.

► **Prompt-based Attribution.** Prompt engineering offers a more flexible alternative to fine-tuning by enabling citation capabilities without specialized training data. Recent work has explored various prompting strategies for eliciting attribution to sources from language models (Hu et al., 2024; Li et al., 2024), while Yue et al. (2023) demonstrated prompting's utility for evaluating attribution quality through natural language inference. Gao et al. (2023) demonstrated the viability of few-shot prompting for citation generation and introduced ALCE, a benchmark for evaluating citation quality, correctness, and fluency. Their experiments revealed significant challenges, including the generation of plausible but incorrect citations. Press et al. (2024) further demonstrated these limitations through chain-of-thought prompting experiments, where even state-of-the-art LLMs performed very poorly and achieved only 4.2-18.5% accuracy compared to human performance of 69.7%. While

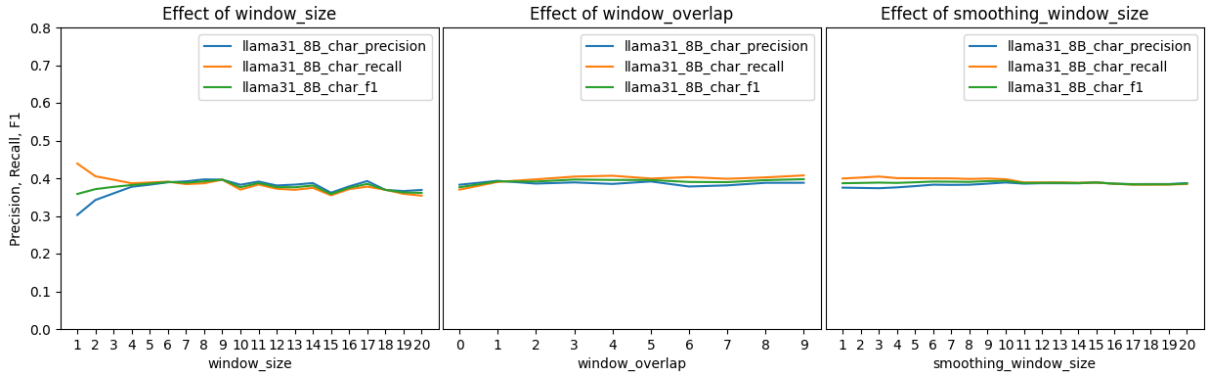


Figure 3: Effect of adjusting sliding window size w, and overlap o, and smoothing kernel size ω hyperparameters on character-level citation F1 (Llama-3.1-8B-Instruct). The system gives consistent results across most choices for these hyperparameters, however smoothing can yield performance benefits when combined with dynamic z-thresholding as shown for Mistral-7B in Table 1.

prompt-based methods offer deployment flexibility, their current reliability remains insufficient for high-stakes attribution tasks.

► **Saliency-based Attribution.** Saliency methods identify which input tokens most strongly influence a model's predictions by analyzing the model's internal representations and gradients. Unlike prompting or fine-tuning approaches, these methods require no modifications to the base model. Yin and Neubig (2022) introduced contrastive explanations that identify influential input tokens by comparing the model's behavior with and without specific context. Sarti et al. (2023) developed this further with their Inseq toolkit, providing a unified framework for extracting and visualizing token-level attributions in sequence generation tasks. Feldhus et al. (2023) compared different approaches for representing feature importance, contrasting model-free saliency methods with instruction-based approaches. Despite their potential for faithful attribution, saliency-based approaches remain relatively unexplored for RAG systems. While Cohen-Wang et al. (2024) demonstrated saliency methods for analyzing model behavior in general text generation, their approach was not investigated specifically for RAG attribution. Concurrent work by Qi et al. (2024) introduced MIRAGE, which employs feature attribution to detect context-sensitive answer tokens and match them with retrieved documents in RAG systems. However, MIRAGE relies on calibration data for optimal performance. Our work advances this direction through a sliding-window approach that dynamically identifies both supporting and conflicting spans without requiring calibration data, prompting, nor fine-tuning and outperforms prompting-based approaches.

6 Discussion & Conclusion

We presented SALSA, a saliency-based attribution approach for RAG systems that identifies supporting and conflicting spans in retrieved documents without requiring special prompting or training. Our experiments demonstrate that SALSA achieves superior performance compared to prompt-based approaches while being much faster. The method is robust across various LLMs and their scales, and provides granular span-level attribution that helps users quickly verify factual claims.

By deriving saliency measurements directly from the LLM's internals, SALSA provides more faithful attributions that reflect actual context usage during generation.

7 Limitations

While SALSA shows promising results for attribution in RAG systems, several important limitations should be noted. First, the sliding window approach may become less effective with very long contexts. While the window size can be adjusted, there is an inherent trade-off between computational efficiency and the ability to capture long-range dependencies. Very large windows increase computational overhead, while smaller windows might miss important contextual relationships that span across longer distances in the text.

Third, SALSA can be sensitive to token-level similarity rather than purely semantic relationships. The system may attribute spans based on lexical overlap or stylistic similarities even when the semantic content is not actually being referenced by

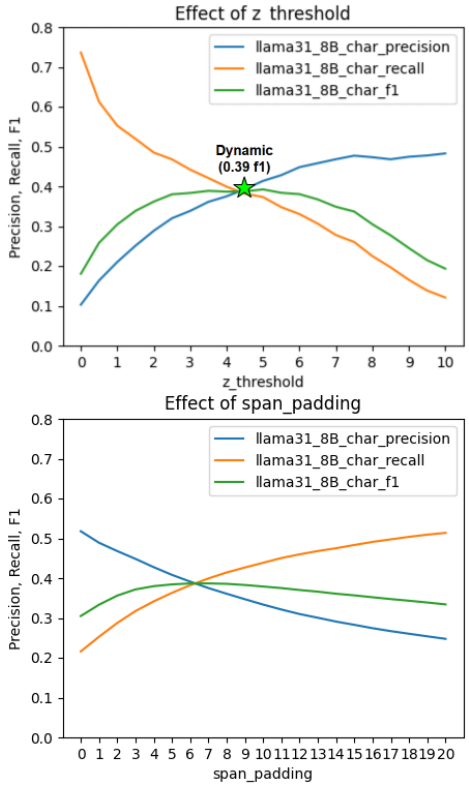


Figure 4: Effect of adjusting z-threshold z and span padding p hyperparameters on character-level citation F1 (Llama-3.1-8B-Instruct). Both z and p present a precision-recall trade-off. However, note that dynamic z-thresholding yields near-optimal performance without needing to manually select a value for z.

the model. This makes it challenging to definitively distinguish between factual citations and cases where the model is simply picking up on shared vocabulary or writing patterns.

Finally, while computing saliency maps does introduce additional computational overhead compared to basic RAG inference, this cost can be significantly mitigated through parallelization and batching. In contrast, prompt-based attribution approaches require multiple additional generation steps, which typically constitute a more substantial computational burden. The relative efficiency of SALSA makes it more practical for production deployment, though careful attention should still be paid to performance optimization.

8 Ethical Considerations

The development and deployment of attribution systems like SALSA raises several important ethical considerations. A primary concern is that detailed attribution patterns could potentially be exploited by malicious actors to develop more sophisticated prompt injection attacks. By understanding exactly how models use their context to generate responses, attackers might be able to craft more effective adversarial inputs or carefully generated misinformation posing as documents that are more likely to be cited by attribution systems. Bad actors could potentially use this capability to generate large volumes of seemingly well-cited but actually misleading content.

Another critical consideration is the risk of false confidence in model outputs. While SALSA's granular span-level attribution aims to make verification more accessible and less cognitively demanding for users, there is still a danger that users might overrely on the system without properly examining the highlighted spans. This risk is particularly acute in high-stakes domains like healthcare or policy-making where incorrect attributions could have serious consequences. However, we believe that by making the verification process more streamlined and transparent, SALSA can actually encourage more users to validate model outputs rather than accepting them blindly.

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. *Preprint*, arXiv:2112.04426.

Chroma. 2024. Chroma. `https://github.com/chroma-core/chroma`.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. 2024. ContextCite: Attributing Model Generation to Context. In *ICML 2024 Workshop on Foundation Models in the Wild*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long Form Question Answering. In *Proceedings of*

the 57th Annual Meeting of the Association for Computational Linguistics, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Nils Feldhus, Leonhard Hennig, Maximilian Nasert, Christopher Ebert, Robert Schwarzenberg, and Sebastian Mller. 2023. Saliency Map Verbalization: Comparing Feature Importance Representations from Model-free and Instruction-based Methods. In *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*, pages 30–46, Toronto, Canada. Association for Computational Linguistics.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling Large Language Models to Generate Text with Citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating factual consistency evaluation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3905–3920, Seattle, United States. Association for Computational Linguistics.

Xiangkun Hu, Dongyu Ru, Lin Qiu, Qipeng Guo, Tianhang Zhang, Yang Xu, Yun Luo, Pengfei Liu, Yue Zhang, and Zheng Zhang. 2024. RefChecker: Reference-based Fine-grained Hallucination Checker and Benchmark for Large Language Models. *arXiv preprint*. ArXiv:2405.14486 [cs].

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint*.

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.

Yifei Li, Xiang Yue, Zeyi Liao, and Huan Sun. 2024. AttributionBench: How Hard is Automatic Attribution Evaluation? *arXiv preprint*. ArXiv:2402.15089 [cs].

OpenAI. 2023. Gpt-4 technical report. *arXiv*.

Nilay Patel, Shivashankar Subramanian, Siddhant Garg, Pratyay Banerjee, and Amita Misra. 2024. Towards improved multi-source attribution for long-form answer generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3906–3919, Mexico City, Mexico. Association for Computational Linguistics.

Ori Press, Andreas Hochlehnert, Ameya Prabhu, Vishaal Udandarao, Ofir Press, and Matthias Bethge. 2024. CiteME: Can Language Models Accurately Cite Scientific Claims? *arXiv preprint*. ArXiv:2407.12861 [cs].

Jirui Qi, Gabriele Sarti, Raquel Fern'andez, and Arianna Bisazza. 2024. Model internals-based answer attribution for trustworthy retrieval-augmented generation.

Gabriele Sarti, Nils Feldhus, Ludwig Sickert, and Oskar Van Der Wal. 2023. Inseq: An Interpretability Toolkit for Sequence Generation Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 421–435, Toronto, Canada. Association for Computational Linguistics.

Marzieh Tahaei, Aref Jafari, Ahmad Rashid, David Alfonso-Hermelo, Khalil Bibi, Yimeng Wu, Ali Ghodsi, Boxing Chen, and Mehdi Rezagholizadeh. 2024. Efficient Citer: Tuning Large Language Models for Enhanced Answer Quality and Verification. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4443–4450, Mexico City, Mexico. Association for Computational Linguistics.

Sirui Xia, Xintao Wang, Jiaqing Liang, Yifei Zhang, Weikang Zhou, Jiaji Deng, Fei Yu, and Yanghua Xiao. 2024. Ground Every Sentence: Improving Retrieval-Augmented LLMs with Interleaved Reference-Claim Generation. *arXiv preprint*. ArXiv:2407.01796 [cs].

Kayo Yin and Graham Neubig. 2022. Interpreting Language Models with Contrastive Explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 184–198, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiang Yue, Boshi Wang, Ziru Chen, Kai Zhang, Yu Su, and Huan Sun. 2023. Automatic Evaluation of Attribution by Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4615–4635, Singapore. Association for Computational Linguistics.

A Sentence-Level Loss Computation

The loss function $\mathcal{L}(\mathbf{A}, \mathbf{c})$ for the entire response is typically computed as the average cross-entropy loss over all tokens in the response. Let $\mathbf{a}_r = \bigcup_{\mathbf{a} \in \mathbf{A}} a \in \mathbf{a}$ be the set of all tokens in all response sentences. The cross-entropy loss of the full response is then:

$$\mathcal{L}(\mathbf{A}, \mathbf{c}) = -\frac{1}{|\mathbf{a}_r|} \sum_{i=1}^{|\mathbf{a}_r|} \log P(\mathbf{a}_{r_i} | \mathbf{c}, \mathbf{a}_{r_{t<i}}) \tag{8}$$

where $P(\mathbf{a}_{r_i} | \mathbf{c}, \mathbf{a}_{r_{t<i}})$ is the model's predicted probability for token $\mathbf{a}_{r_i}$ given the context and preceding tokens.

For computing the loss for a specific sentence $\mathbf{a}_s$ within the response, we modify this approach. Let $\mathbf{a}_s = (\mathbf{a}_{s_1}, \ldots, \mathbf{a}_{s_k})$ be the tokens of the sentence we're focusing on, where s_1 is the index of the first token of the sentence. The loss function for this specific sentence is then:

$$\mathcal{L}(\mathbf{a}_s, \mathbf{a}_{t<s}, \mathbf{c}) = -\frac{1}{k} \sum_{i=1}^{k} \log P(\mathbf{a}_{s_i} | \mathbf{c}, \mathbf{a}_{t<s}, \mathbf{a}_{s_{t<i}}) \tag{9}$$

This formulation focuses on the tokens in the specific sentence $\mathbf{a}_s$ while maintaining the autoregressive nature of the model by conditioning on the previous tokens in the response denoted by $\mathbf{a}_{t<s}$. Using this sentence-specific loss in Equation 2 of the main text, we can compute saliency scores specifically tailored to identify important spans in the context that support or conflict with the sentence in question, rather than the entire response.

B Sliding Window Example

This section provides a worked-out example of the sliding window approach, demonstrating how we identify spans that support or conflict with a specific response sentence, and how we determine the relevant documents. We use mock scores and documents for illustration and skip the final scaling step for simplicity.

Let's consider a context $\mathbf{c}$ of 10 tokens, divided into three documents:

$$\mathbf{c} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}) \tag{10}$$

$$\begin{aligned} \mathbf{d}_1 &= (c_1, c_2, c_3) \\ \mathbf{d}_2 &= (c_4, c_5, c_6, c_7) \\ \mathbf{d}_3 &= (c_8, c_9, c_{10}) \end{aligned}$$

We'll use a window size $w = 3$ with an overlap of 1 token ($o = 1$) and a padding of 1 ($p = 1$).

B.1 Saliency Score Computation

First, we compute the *relative loss* δ_k for each window $\mathbf{w_k}$ using Equation 2. Let's assume we get the following scores:

$$\begin{aligned} \delta_1 &= 0.5 && (\mathbf{w_1} = (c_1, c_2, c_3)) \\ \delta_2 &= -0.2 && (\mathbf{w_2} = (c_3, c_4, c_5)) \\ \delta_3 &= 0.8 && (\mathbf{w_3} = (c_5, c_6, c_7)) \\ \delta_4 &= 0.3 && (\mathbf{w_4} = (c_7, c_8, c_9)) \\ \delta_5 &= -0.7 && (\mathbf{w_5} = (c_9, c_{10})) \end{aligned}$$

Next, we compute the saliency score s_i for each token using Equation 3:

$$s_1 = s_2 = 0.5$$
$$s_3 = \frac{0.5 + (-0.2)}{2} = 0.15$$
$$s_4 = -0.2$$
$$s_5 = \frac{-0.2 + 0.8}{2} = 0.3$$
$$s_6 = 0.8$$
$$s_7 = \frac{0.8 + 0.3}{2} = 0.55$$
$$s_8 = 0.3$$
$$s_9 = \frac{0.3 + -0.7}{2} = -0.2$$
$$s_{10} = -0.7$$

B.2 Supporting Salient Span Identification

We'll use three non-negative uniform bins for discretization ($\eta = 3$): low (< 0.33), medium (0.33 to 0.67), and high ($>= 0.67$). The binary sequence $\mathbf{b}$ (where 1 represents scores in the highest bin) is:

$$\mathbf{b} = (1, 1, 0, 0, 0, 1, 1, 0, 0, 0) \tag{11}$$

By identifying consecutive sequences of 1's and applying padding, we get two spans:

$$\mathbf{r}_1 = (c_1, \ldots, c_3)$$
$$\mathbf{r}_2 = (c_5, \ldots, c_8)$$

B.3 Attribution to Documents

We match these spans to their containing documents:

$$(c_1, \ldots, c_3) \subseteq \mathbf{d}_1 \tag{12}$$
$$(c_5, \ldots, c_7) \subseteq \mathbf{d}_2 \cup (c_8) \subseteq \mathbf{d}_3 \tag{13}$$
$$\tag{14}$$

Thus, the set of documents that support the sentence is:

$$\mathbf{D_a} = \{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3\} \tag{15}$$

B.4 Conflicting Salient Span Identification

For conflicts, we focus on the spans with the lowest scores, so the three uniform bins are: high (> -0.33), medium (-0.33 to -0.67), and low ($<= -0.67$). The binary sequence $\mathbf{b}$ (where 1 represents scores in the lowest bin) is:

$$\mathbf{b} = (0, 0, 0, 0, 0, 0, 0, 0, 0, 1) \tag{16}$$

By identifying consecutive sequences of 1's and applying padding, we get one span:

$$\mathbf{r}_1 = (c_9, c_{10})$$

Matching this span to its containing document:

$$(c_9, c_{10}) \subseteq \mathbf{d}_3 \tag{17}$$

Thus, the set of documents that potentially conflict with the sentence is:

$$\mathbf{D_c} = \{\mathbf{d}_3\} \tag{18}$$

In this example, our sliding window approach has identified that the sentence is supported by information from all three documents, but may conflict with information in document $\mathbf{d}_3$.

C Alternative Evaluation Metrics

In addition to character-level P, R, F1, it is appropriate to evaluate systems using Intersection over Union (IoU) for **granular span-level evaluations**. IoU measures the overlap between candidate spans $\mathbf{S}$ and ground truth spans $\mathbf{G}$ using the Intersection over Union (IoU) metric:

$$\text{IoU} = \frac{|\mathbf{S} \cap \mathbf{G}|}{|\mathbf{S} \cup \mathbf{G}|} \tag{19}$$

Document-level evaluations are computed similarly but instead of using character-level F1 as a threshold, we use the IoU score as a threshold. In other words, we only consider a document citation as a true positive if both: (1) the document is correctly cited, and (2) the corresponding text span achieves a span-level IoU above a threshold of 0.4 with the annotated ground truth span.

D Hyperparameter Analysis

To ensure robust performance and provide guidance for practitioners implementing SALSA, we conduct extensive hyperparameter sensitivity analyses. We evaluate five key parameters that control different aspects of our attribution pipeline: (1) sliding window size, which determines the granularity of context masking, (2) window overlap, which affects how smoothly the saliency scores transition between adjacent windows, (3) span padding size, which influences how much surrounding context is included in extracted spans, (4) saliency threshold (z-score), which controls how selective the method is in identifying salient tokens, and (5) smoothing window size, which determines the degree of post-processing applied to raw saliency scores.

For each parameter, we conduct sweeps across reasonable ranges while holding other parameters fixed at their default values. We evaluate performance on the same test set from the annotated ELI5 dataset using both document-level and character-level span metrics. The following subsections present detailed analyses for each parameter.

D.1 Sliding Window Size Analysis

[Placeholder]

D.2 Window Overlap Analysis

[Placeholder]

D.3 Span Padding Analysis

[Placeholder]

D.4 Saliency Threshold Analysis

[Placeholder]

D.5 Smoothing Window Analysis

[Placeholder]

E Language Model Prompts

This section lists the prompts used for both answer generation and source attribution in our experiments.

E.1 Answer Generation Prompt Template

The following prompt template was used by the language model `Llama-3.1-70B-Instruct` to generate answers for ELI5 questions in the end-to-end RAG experimental setting:

SYSTEM: You are a question answering assistant. Your job is to search for documents with relevant information and then use them to answer the user's question. To search, respond with `` `search("Q")` `` where `` `Q` `` is a search query based on the user's question.
USER: Question: `Q`
ASSISTANT: search("Q")
USER: Search Results:
———
Document [1] (Title: `T`): `P`
———
[Additional documents from ELI5 in the same format...]
———
ASSISTANT: Answer:

where:

- `{Q}` is the user's question
- `{T}` is the document title
- `{P}` is the document text
- `{ID}` is the document identifier. Used for IEEE-style inline citations.

E.2 Source Attribution Prompt Template

For the prompt-based attribution approach, we used the following template:

SYSTEM: You are a source attribution assistant. You will be given a question, an answer, and a document. For each sentence in the answer, your job is to tag all text in the document that supports the answer. You will only be given one answer sentence at a time to analyze. When you are given an answer sentence, you must write out the document verbatim. Surround text that supports the answer sentence with `<|support|>...</|support|>` tags. If no supporting text is found, do not add any tags.
USER: Question: `Q`
Answer: `A`
Document `[ID]` (Title: `T`): `P`
———
[Additional documents from ELI5 in the same format...]
———
Remember, you must write out the document verbatim. Surround text that supports the answer sentence with `<|support|>...</|support|>` tags. If no supporting text is found, do not add any tags.
ASSISTANT: I am ready for the first answer sentence!
USER: Sentence: `S`
ASSISTANT: Document `[ID]` (Title: `T`):

For identifying conflicting information, the same template is used but with:

- `<|conflict|>` tags instead of `<|support|>`
- "conflicts with" instead of "supports" in the instructions
- "conflicting" instead of "supporting" in the reminder

F Annotation Guidelines

Annotators were on-boarded with the following guidelines for the tool and task.

SALSA Span Annotation Guidelines

Given a question, its context (if shown), and an answer sentence, your task is to annotate zero or more **citation** and **conflict** spans in each of the five retrieved documents.

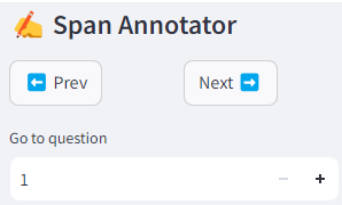
A citation span is text within the document that supports an answer sentence, while a conflict span is text within the document that conflicts with (possibly contradicts) an answer sentence.

The following sections outline the span annotation procedure and provide guidelines for what makes high quality, relevant citation/conflict spans.

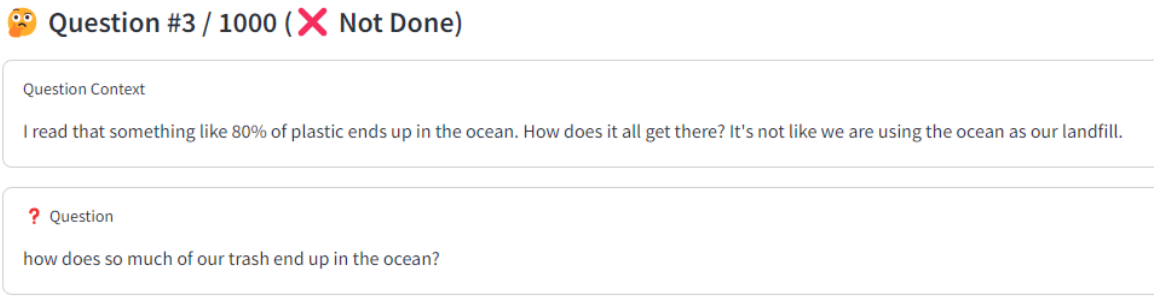
Annotation Procedure

Step 1: Navigation

To navigate to the previous or next question, use the buttons on the left-hand sidebar. You may also use the numeric input box to jump directly to a question.



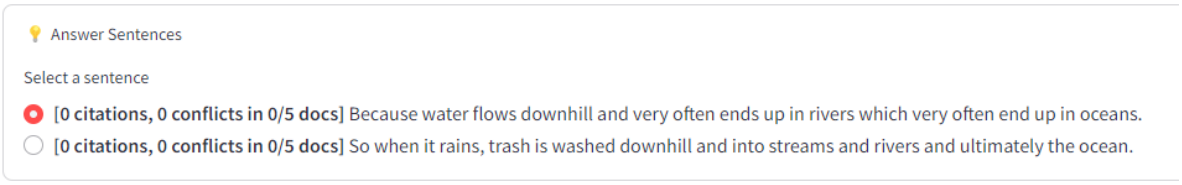
Step 2: Read the question and its context



Note: sometimes there is no context provided and you will only see a question.

Step 3: Select an answer sentence

Each answer sentence gets its own distinct set of citation and conflict span annotations. Use the selection box to choose a sentence:

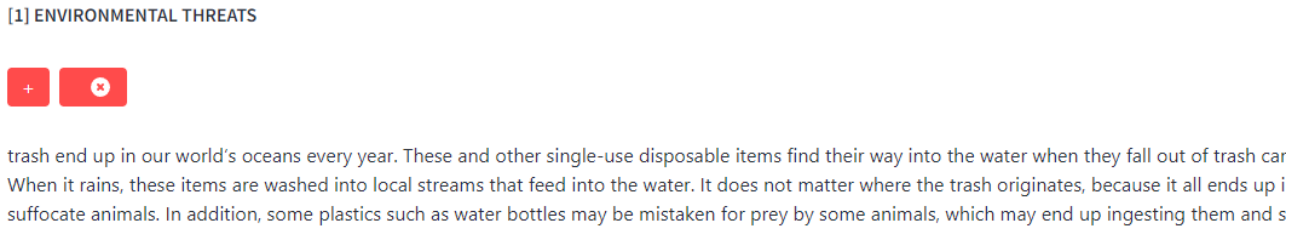


Step 4: Select a span type

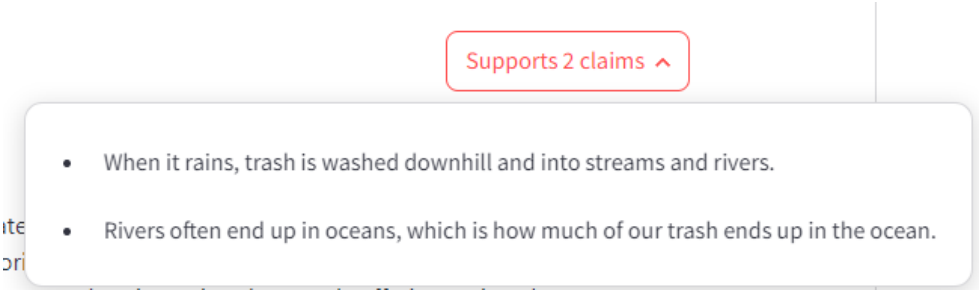
Select if you want to annotate **citation spans** (supporting text) or **conflict spans** (possibly contradicting text):

Step 5: Carefully read the first document

Make sure to actually read the document text and not just superficially skim it.



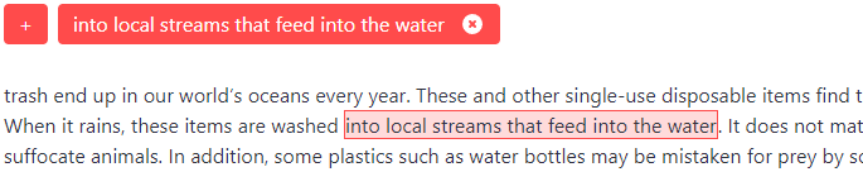
If the document was detected by the ALCE NLI model to support one or more “claims” extracted from the answer by GPT-3.5, you can see them here on the right-hand side of the document:



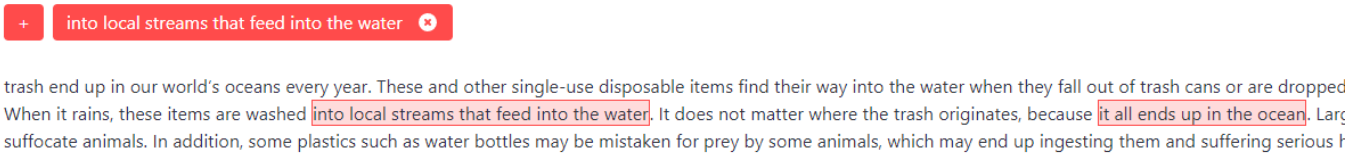
Note: these “supported claims” were generated automatically during the ALCE ELI5 dataset creation process, and don’t necessarily correspond to the presence or absence of valid spans in the document. If you see 0 claims supported, still make sure to read the document completely before moving on.

Step 6: Annotate the spans

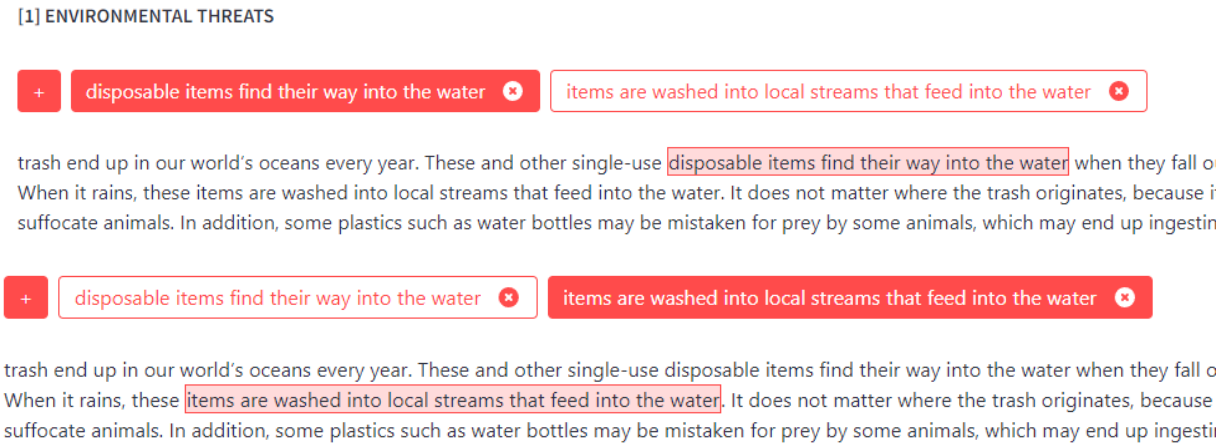
To annotate a span, simply select the text with your cursor:



You may select multiple spans within the same document to indicate that multiple text passages support (or conflict with) the answer together. This means a system should identify ***both*** of these spans within the document:



Alternatively, you may select multiple spans within the same document where ***either one*** can be independently cited; i.e., they both independently support (or conflict with) the answer sentence. To do this, use the [+] button to create additional ***span sets***:



As before, each span set can contain more than one span when multiple passages should be cited together.

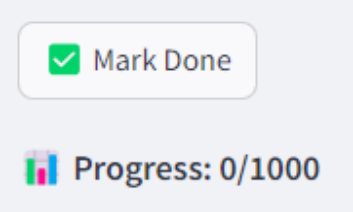
Step 7: Repeat Steps 5-6 for each of the remaining documents.

If no supporting or conflicting spans are identified (i.e. the documents are completely irrelevant to the answer), it is okay to annotate nothing.

Step 8: Repeat Steps 3-7 for each of the remaining answer sentences.

Step 9: Mark the question as Done

When all documents have been annotated for all answer sentences, mark the question as "Done" with this button:



This will increment the progress indicator.

Span Annotation

What makes a good span?

A span should cover the **minimum amount of contiguous text** needed to support (or refute) a statement in the answer sentence. Irrespective of the truthfulness of the text and whether the text entails the statement.

For example, consider the following question and answer:

Question: *What color is the sky?*
Answer: *The sky is blue on clear days and grey on cloudy days.*

The following represents a good minimal contiguous span (✅ **do this!**):

Document: *You can always tell it will be a great day to hike if you look out the window and see a nice, clear day with a bright blue sky. If the sky is grey and cloudy, you might want to take a look at the forecast to see if it's going to rain.*

The following represents minimal but non-contiguous spans (❌ **don't do this!**):

Document: *You can always tell it will be a great day to hike if you look out the window and see a nice, clear day with a bright blue sky. If the sky is grey and cloudy, you might want to take a look at the forecast to see if it's going to rain.*

The following represents a non-minimal contiguous span (❌ **don't do this!**)

Document: *You can always tell it will be a great day to hike if you look out the window and see a nice, clear day with a bright blue sky. If the sky is grey and cloudy, you might want to take a look at the forecast to see if it's going to rain.*

If information is spread out in the document (more than 10-15 words apart) it is okay to annotate separate spans that work together to support (or conflict with) the answer (✅ **do this!**):

Document: *Have you ever really looked up at the sky? Every now and then you should try it, it's good for your well-being and sense of self-purpose. It doesn't matter if it's clear, sunny and blue, or if it's one of those days where your bones ache and you don't want to get out of bed because it is cloudy, rainy, and grey.*

In the case above, a single minimal contiguous span would include too much irrelevant information, so we break it up into three independent minimal contiguous spans.

What makes a relevant span?

Often, an answer will summarize or paraphrase the supporting documents. Thus it is unnecessary for a span to contain the exact word-for-word phrasing that is in the answer. A relevant citation span should **share semantic meaning** with the answer or **entail** the answer. A relevant conflict span should **contradict** the answer or make it **ambiguous**.

For example, consider the following question and answer:

Question: *What color is the sky?*
Answer: *The sky is blue on clear days and grey on cloudy days.*

The following represents a citation span that shares semantic meaning with the answer (turquoise is a shade of blue):

Document: *On clear evenings in the summer you can watch the sun set in the turquoise sky over the Hotel Calabria on Flamingo Island… It is one of those rare photo opportunities that you don't want to miss while you're in town.*

The following represents a citation span that entails the answer given other context in the document (the lake reflects the sky):

Document: *When the wind is calm the lake makes a mirror-like reflection of the sky. On bright clear days the lake shines in a deep radiant blue, while on cloudy days the lake becomes a dark grey portal among the hills.*

The following represents a conflict span that directly contradicts the answer:

Document: *Polar night: when you're up that far north, it is night time for 24 hours. Regardless of the weather, the sky is always black at all times of day. This is because the sun is always below the horizon at that latitude.*

The following represents a conflict span that makes the answer ambiguous (blue is not the only color it can be on a clear day):

Document: *Often during a sunset the sky can be painted deep shades of orange and red, especially on clear days! You've heard the saying, "red sky at night, sailors delight…" which means that there is clear weather ahead!*

Handling duplicate citations

A duplicate citation happens when the same (or similar) information shows up in: (a) multiple places within the same document, or (b) in multiple documents.

Duplicate citations within the same document

If the same information shows up in multiple places in the same document, use the [+] button to create multiple span sets (see Step 6: Annotate the spans), one set for each instance of the duplicated information. This way, we indicate that only one citation is needed in that document, referring to the spans from any one of the sets.

Duplicate citations across multiple documents

If the same information shows up in multiple documents, multiple span sets are not needed, since a citation system would be expected to cite each document in which that information exists. Simply annotate spans for all instances of the duplicated information.

For example:

Answer: *The sky is blue on clear days and grey on cloudy days.*

Document [1]: *The sky is blue on clear days and grey on rainy or cloudy days because…*

Document [2]: *Blue skies on clear days are nice to see…*

Document [3]: *I love grey skies on a cloudy day. Call me a pessimist, but…*

Annotating spans for long answers

Ideally, answer sentences are short and concise, making only one or two statements. For example:

Answer: *The sky is blue on clear days and grey on cloudy days.*

However, sometimes the answer authors allow their sentences to run on, packing many statements within the same sentence. For example:

Answer: *The sky is blue on clear days and grey on cloudy days, but it doesn't really matter if you're far enough north because of the Polar night where the sky is black all the time, or if you're watching a sunset where even a clear day can have deep orange skies.*

The answer above contains four separate claims and would have ideally been split into multiple sentences, but unfortunately it was not. In this case, a single span is rarely enough to fully support the sentence, and the supporting information is likely to be scattered across many documents. When faced with an answer sentence like this, make sure to annotate spans over all of the available evidence for each statement:

Document [1]: *When the wind is calm the lake makes a mirror-like reflection of the sky. On bright clear days the lake shines in a deep radiant blue, while on cloudy days the lake becomes a dark grey portal among the hills.*

Document [2]: *Polar night: when you're up that far north, it is night time for 24 hours. Regardless of the weather, the sky is always black at all times of day. This is because the sun is always below the horizon at that latitude.*

Document [3]: *Often during a sunset the sky can be painted deep shades of orange and red, especially on clear days! You've heard the saying, "red sky at night, sailors delight…" which means that there is clear weather ahead!*