Contributive Attribution for Question Answering via Tree-based Context Pruning

Anonymous ACL submission

Abstract

The development of large language models for question answering has benefited from understanding which context sentences are responsible for their answer. These sentences are commonly called contributive attribution. Recent works use the probability drop of the answer for a modified context to estimate how well sentences in the context match the attribution. Unfortunately, this metric does not convey the necessity and sufficiency qualities that the natural language processing community has defined in previous works. We propose a metric composed of a necessary and a sufficiency score based on probability drops to fill this gap. Then, to illustrate the soundness of the metric in practice, we develop a hierarchical method, TreeFinder, which progressively selects finer parts of the context through tree-based pruning using the metric. It begins with a few coarse-grained chunks and iteratively narrows the top k chunks according to our metric down to sentence-level granularity. At each iteration, we calculate our metric using ablation-based log-probability differences and filter out irrelevant chunks. Experimental results on HotpotQA demonstrate that TreeFinder outperforms ContextCite and TracLLM in contributive attribution quality when it is composed of a few sentences. Further experiments on Loogle and LongBench-v2 show that TreeFinder ranks sentences for attribution score better than ContextCite in long contexts.

1 Introduction

Revealing which sentences are determinant to the answer of Large Language Models (LLMs) in Question-Answering (QA) systems is an important step to building trusted agents. In an ideal QA system, users should be able to trace the answers effortlessly to eliminate doubts, and methods should be built to automatically spot uncertainty.

Currently, LLMs are still challenged by language comprehension benchmarks, especially when contexts are long (Lee et al., 2024), due to the sparsity of rewards (Su et al., 2025) and limited data (Villalobos et al., 2022). This may be, in part, because current models are prone to missing key information in their context (Yang et al., 2025), especially as the number of tokens increases (Hsieh et al., 2024). Another part of the problem is that LLMs are prone to distractions, such as irrelevant information (Yoon et al., 2024) or adversarial attacks (Zou et al., 2023).

In this work, we focus on contributive attributions (Worledge et al., 2024a). Those are the key sentences that the model identifies as important to create its answer. Identifying these attributions enables, for example, the creation of robust fact-checking solutions by verifying that the answer is entailed by the attributions. They can also be used to generate a second answer that should match the first or enhance it (Huang et al., 2024). As a last example, they can show the biases of the model when presented with contradictory statements. Indeed, not having all variations equally weighed could indicate a preference.

These attributions should not be confused with corroborative attributions, which are the sentences supporting the correct answer (Worledge et al., 2024a,b). By contrast, contributive attributions (or rationales, citations, context traceback (Wang et al., 2025b)) fundamentally attempt to explain the answer provided by the LLM. They are thus very valuable, since

their mismatch with corroborative ones may help to understand the failures of LLM when their answers are not satisfactory.

Methods for finding contributive attributions often leverage the innards of LLM, such as using attention scores or gradients similar to GradCAM (Selvaraju et al., 2019). However, these depend on many choices, such as the particular head or layer computing these scores, that vary with architecture (Pirenne et al., 2025). Beyond making users lost in choices, these methods could become unusable as new LLM architectures, possibly incompatible, are developed.

In this work, we first propose a metric that defines the quality of a set of sentences as the contributive attribution using probability drops. This metric consists of a sufficiency score and a necessity score. Afterwards, we detail an algorithm that segments a context, measures the scores of the segments, selects the best ones with a Top-k filter, and repeats greedily until the desired sentence-level coarseness. Our algorithm avoids enforcing a linear model, such as with LIME (Ribeiro et al., 2016) or SHAP (Lundberg and Lee, 2017) and thus does not make assumptions on the linearity of the metric under context ablations.

The paper is structured as follows. In Section 1.3, we give our motivation to use our metric and we formally introduce the problem statement. Then, we go over related works. Afterwards, in Section 2, we explain our methodology and write the pseudocode to implement our algorithm. We continue in Section 3 with our experiments, followed by the results and in Section 4, we conclude. Finally, for completeness, we provide four appendices. Appendix A provides details of how we divide the context in each iteration. In Appendix B, we display the plots for more datasets. In Appendix C, we identify the top five sentences given by each compared algorithm for two specific samples. And, in Appendix D, we show how contributive and corroborative contributions diverge in practical cases.

1.1 Contributive attributions

The literature often defines multiple qualities that attributions should possess. Two qualities, a compactness and a sufficiency condition, have been described by Lei et al. (2016). Then, a necessity condition has been used by Yu et al. (2019) and DeYoung et al. (2020). These were used in subsequent works such as (Brinner and Zarrieß, 2023) and (Zhang et al., 2023) for finding attributions in two label problems, but not for full sentence answers.

In this section, we want to define contributive attributions through probability drops to satisfy the three qualities of necessity, sufficiency, and compactness. Using this definition, we derive a single metric to optimize for. By construction, this metric only finds contributive attributions, since the probabilities and the answer are derived from the same model.

From this point on, every piece of text will be considered a sequence $C = [s_1, \ldots, s_n]$ of atoms, which are sentences in our experiments. We use $|C| = n$ to denote the length of such a sequence. In addition, we use $R \subseteq C$ to denote that R is a subsequence of C. Note that a subsequence is not necessarily contiguous, but the relative order of its elements is conserved.

We consider a closed QA setting, where a question Q is associated with a context C made up of a number of $|C|$ separable elements (e.g., sentences). Let M be a generative model from which we can generate an answer A and derive its probability knowing Q and C giving $P_M(A|Q, C)$.

We want to find $R \subseteq C$ defined as

$$R := \arg\min_{r \subseteq C} |r| \quad s.t. \begin{cases} \text{Equation (2)} \\ \text{Equation (3)} \end{cases} \tag{1}$$

$$\log P_M(A \mid Q, r) = \log P_M(A \mid Q, C) \tag{2}$$

$$\log P_M(A \mid Q, C \setminus r) = \log P_M(A \mid Q, \varnothing) \tag{3}$$

In practice, the only solution of Equation (1) with equality constraints is $r = C$ because the probability of the answer changes even by adding irrelevant sentences. We therefore relax these requirements to add small tolerances. In this context, we adopt the following formal definition of R:

$$R := \arg\min_{r \subseteq C} |r| \quad s.t. \begin{cases} \text{Equation (5)} \\ \text{Equation (6)} \end{cases} \tag{4}$$

$$\log P_M(A \mid Q, C) - \log P_M(A \mid Q, r) \leq \varepsilon_{\text{suf}} \tag{5}$$

$$\log P_M(A \mid Q, C \setminus r) - \log P_M(A \mid Q, \varnothing) \leq \varepsilon_{\text{nec}} \tag{6}$$

where ε_{suf} and ε_{nec} are small constants that control tolerance for the sufficiency and necessity constraints, respectively.

We add the left-hand sides of the sufficiency and necessity inequalities, weighed respectively by $1 - \alpha$ and α, to produce a single metric. We get

$$\begin{aligned} a_M(Q, C, A, r) = \alpha\big(&\log P_M(A \mid Q, C \setminus r) \\ &- \log P_M(A \mid Q, \varnothing)\big) \\ + (1-\alpha)\big(&\log P_M(A \mid Q, C) \\ &- \log P_M(A \mid Q, r)\big). \end{aligned} \tag{7}$$

1.2 Probability drop

Imposing modifications in a context C by removing some parts to create a subsequence $C' \subseteq C$ modifies the probability distribution of the answer. We can infer the loss or gain of information by analyzing the relationships between the distributions $P_M(\cdot|Q, C)$, $P_M(\cdot|Q, C')$, $P_M(\cdot|Q, C \setminus C')$, and $P_M(\cdot|Q, \emptyset)$. The first and fourth are constant with respect to the choice of C'. Together with the second and third, they produce the necessary and sufficient conditions to produce A when they are equal.

When selecting the candidate subsequence $C' \subseteq C$, we can work at different levels of granularity. In our case, the finest granularity would correspond to the sentence-level, where a sentence is defined by NLTK (Bird et al., 2009). But we can also consider coarser levels, where sentences are grouped by paragraphs, pages, etc. We propose a hierarchical approach that will start from a coarse subdivision of the context, and then, after selecting the best C' by removing complete groups of sentences, it will iteratively consider finer and finer granularity levels.

In this paper, we rank the chunks with the metric of Equation (7) and then choose to keep the t_k best ones to create C'.

1.3 Related Work

In the QA setting, multiple fundamental questions arise: Does the answer come from internal or external knowledge? Is it reliable? Can it be explained or traced back? When we limit ourselves to the context, the main interest in explainability is corroborative (elements supporting an answer) and contributive (elements responsible for the answer) attributions (Worledge et al., 2024a).

Corroborative Attribution Corroborative attributions can exist without a reference model because they correspond to the rationale for the construction of the answer. Although interesting, they fundamentally achieve a different goal from ours. While we want to discover what the LLM is using as support, they want to support the ideal answer. Both attributions can be found, and their overlap is an indicator of the correctness of the model. We provide an intuition for this in Appendix D.

Contributive Attribution Contributive attributions can be given by the model itself. For example, GopherCite (Menick et al., 2022) uses guided generation to provide exact quotes. SelfCite (Chuang et al., 2025) refines this through Reinforcement Learning using probability drops as reward signals. Both methods rely heavily on the model being trusted, which is what we want to avoid. We therefore focus on properties that are impartial, such as probability distributions or attention weights, without dependence on the statistical generation.

Reducing the context or prompt to eliminate redundant information can be done by paraphrasing, filtering (Hou et al., 2024) or encoding it (Cheng et al., 2024) so that its size is reduced but the content is largely retained in meaning (Li et al., 2025). (Feng et al., 2018) uses an iterative filtering in which they keep removing the word that has the least impact. In contrast to us, they operate at a fixed grain and do not incorporate a tree approach.

The search for contributive attributions through the alteration of the context can be

found in the literature as a perturbation-based method for feature attribution (Zhao et al., 2024). Perturbations can be random, as in LIME or generated by models such as variational autoencoders (Alvarez-Melis and Jaakkola, 2017).

One recent work leveraging perturbations to find attributions is ContextCite (Cohen-Wang et al., 2025). To avoid computing the probability of a particular answer after the removal of an atom of text, ContextCite generates random ablation vectors and extrapolates the scores of each sentence with a linear model. In contrast, we avoid using such a model by finding important chunks of successively smaller size directly from the answer probabilities. ContextCite does not compute probabilities directly; instead, they compute the logit function $\log \frac{p}{1-p}$ of the difference of the chosen logits with a smooth approximation of the maximum of the rest of the logits ($LogSumExp$). This slightly modifies the objective.

TracLLM (Wang et al., 2025b) uses a tree-based method. They create chunks and remove the ones that have the lowest metric permanently from the context before continuing deeper. To compute their metric, they sample and aggregate a portion of all possible marginal probability drops. Similarly to ContextCite, they use linear models to compute a score for each chunk at every depth.

As emphasized in (Su, 2025), these approaches leveraging probability drop, working as a consequence of the answer being modeled probabilistically, are fundamentally robust to architectural modifications that are currently the focus of research for stronger LLMs (Assran et al., 2025; Wang et al., 2025a).

2 TreeFinder

The complete algorithm is described in Algorithm 1. Here is a short summary of how it works. It first divides the context into chunks. Afterwards, it computes both scores for each individual chunk in the running as R, and filters them to keep only the most promising ones. It then repeats with smaller chunks until they all reach the size of one sentence. This process is summarized in Figure 1.

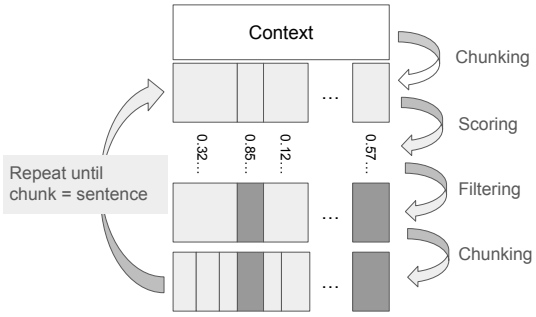


Figure 1: TreeFinder overview. The three repeating steps are represented in light gray. Dark-gray boxes represents the filtered out sentences that will not be subdivided or re-scored in further repetitions.

We have defined two intermediary functions in Algorithms 2 and 3: Score and Chunkify. The first computes our metric $a(Q, C, A, r)$ for each given chunk r. The second simply cuts the context into approximately k chunks, returning a list of chunks and a mapping from the indices of sentences in the global context to the indices of the chunks that contain them. Since it is a straightforward implementation, it has been moved to the Appendix. A Top-k filter is present to ensure a logarithmic time complexity with the size of the context $|C|$. The filter can be modified to be more restrictive for faster code execution.

3 Experiments

In this section, we first choose three datasets to use in our experiments. We then describe the LLM models and parameters used for the experiments. Next, we compare our algorithm with ContextCite and TracLLM on the chosen datasets and explore the influence of the necessity weight parameter α of Algorithm 1.

3.1 Datasets

We work with the following three datasets. The first is HotpotQA (Yang et al., 2018), a widely used dataset for corroborative attribution in QA. Its purpose is to benchmark LLM for their ability to make multi-hop reasoning. Its "distractor" mode includes various paragraphs of related articles, which makes it quite challenging for LLMs.

The other two, LooGLE (Li et al., 2024) and LongBench-v2 (Bai et al., 2025), are made up of long contexts. We use two subdivi-

Algorithm 1 TreeFinder

Input: M (Model), C (Context), Q (Question), A (Answer), k (Initial Chunking Factor), t_k (Top-k), α (Necessity weight)
Output: X (Per sentence score)
$V \leftarrow \mathbf{1}^{|C|}$ ▷ Ablation
$X \leftarrow \mathbf{0}^{|C|}$ ▷ Score
$V_c \leftarrow \mathbf{1}^{|k|}$ ▷ Chunk ablation
$X_c \leftarrow \mathbf{0}^{|k|}$ ▷ Chunk score
while $k < t_k|C|$ and $V_c \neq \mathbf{0}$ **do**
    $S, map \leftarrow \text{Chunkify}(C, k)$
    for $0 \leq i < |S|$ **do**
        for $j \in map[i]$ **do**
            $V[j] \leftarrow V[j] \wedge V_c[i]$
            $X[i] \leftarrow X_c[i]$ if $V_c[i]$
        end for
    end for

    $X_c \leftarrow \text{Score}\quad (C, Q, A, M, V, S, map, \alpha)$
    $V_c \leftarrow \text{Top-k}(X_c, map, t_k)$
    $k \leftarrow k * t_k$
end while
return X

Algorithm 2 Score

Input: C (Context), Q (Question), A (Answer), M (Model), V (Ablation vector), S (Chunks), map (Local to global indices), α (Necessity weight)
Output: $scores$ (Per chunk score)
$P_{total} \leftarrow \log P_M(A|Q, C)$
$P_\emptyset \leftarrow \log P_M(A|Q, \emptyset)$
for $0 \leq i < |S|$ **do**
    if $\forall j \in \text{map}[i], V[j] = 1$ **then**
        $P_{rem}[i] \leftarrow \log P_M(A|Q, C \setminus S_i)$
        $P_{unique}[i] \leftarrow \log P_M(A|Q, S_i)$
    end if
end for
$scores = \alpha(P_{rem} - P_\emptyset) + (1-\alpha)(P_{total} - P_{unique})$
return $scores$

sions (named 2a and 2b hereafter) of this LooGLE: short dependencies and long dependencies. These measure the distance between the different elements needed to answer in the context.

All datasets have been truncated to allow the algorithms to run on a maximum of two A100 40GB (one for HotpotQA). Loogle and Longbench-v2 have had their maximum context length set to 20000 tokens, with a maximum of a thousand sample per for time and budget constraints.

3.2 Model and Parameters

The algorithm has 3 hyperparameters: the chunking factor k, the Top-k value $t_k = 3$ and the necessity weight α. We chose $k = 6$ with $t_k = 3$ and $\alpha = 0.25$ after having tried $k = 2$ and $k = 3$ which gave poor results.

The model M chosen for all experiments is `Qwen-2.5-7B-Instruct-1M` (Yang et al., 2025) for its stated long context understanding. Comparisons are carried out using the Transformers Python library (Wolf et al., 2020).

Corroborative attribution tests are run using the vLLM engine (Kwon et al., 2023) for fast inference in long contexts.

3.3 Results

Evaluation criteria We compute the necessity and sufficiency of the first sentences taken together in a given group of size $w \in [1, \ldots, 5]$ with $\alpha = 0.5$ and report the average and standard error in the Figure 2 for HotpotQA. Since this is a greedy approach, "Ground Truth" is an upper bound.

Lastly, in Figures 3 and 4, we search for the first five "Ground Truth" sentences and report the median positions each method has assigned to them. Unfortunately, since TracLLM does not provide a score for all sentences, we cannot include them.

It might seem surprising that the evaluation does not use human annotations, but this is intended since the goal is to identify the sentences M relies on.

Comparison We now compare the rankings given by our algorithm with those provided by ContextCite and TracLLM. We can see on Fig-

ure 2, that for the first sentence, TracLLM and ContextCite reach lower average scores than TreeFinder. However, as more sentences are included, TreeFinder improves the average score, showing the improved quality of the broader contributive attribution.

With the selected hyperparameters, we achieve a similar time of 5.72s as ContextCite (5.84s) despite an increase in the number of forward calls (46.9 against 32). The reason is that $P(A|Q, r)$ becomes cheaper in terms of tokens used compared to $P(A|Q, C \setminus r)$ as r shortens. This leads to about half of the calls that account for the entire compute time. Since TracLLM discards chunks from the context and never uses them again, they also have a disproportionate amount of calls (293.3) to the compute time (25.6s) compared to ContextCite. Their default parameters have three score estimations that compute many more marginal probabilities for the same initial chunking factor (their $K = 6$).

For the score-based criterion, no method shows a clear advantage across all datasets (others in Appendix). In Appendix B, we provide results for Loogle (short) and Longbench where TracLLM is better, as well as Loogle (long) where the result depends on α. In general, while TracLLM is slower, its attributions are closer in value to the ground truth.

Since the metric values are close, we provide a last angle for comparison in Figure 3, where we take the positions at which the first k ground-truth sentences can be found in each method and report the median over the dataset. We find that our method misses the ranking of the most important sentence, but the following are consistently classified better than ContextCite. Increasing the number of ablations (calls) for ContextCite from 32 to match our methods' number of calls, we find no significant improvement. This, we believe, highlights the need to incorporate both the necessity and sufficiency scores in attribution finding algorithms to properly encompass all the desired properties of an attribution.

We note that our slice of HotpotQA has a median of 2 sentences as support, while Loogle (long) has 3 and Loogle (short) has 1 (and Longbench is not labeled). The ranking difference, therefore, can be argued to be only relevant on Loogle (long). However, contributive attributions capture more than the support, and thus a greater span should be considered relevant.

Influence of the scores necessity weight parameter In this section, we modify the weights of the necessity and sufficiency scores α from 0 to 1. Choosing 0 or 1 nearly halves the number of forward calls to the LLM since we do not compute one of the two metrics, any other repartition does not have this benefit.

From Figure 4, we see that the best value for α is between 0 and 1 since our original value of $\alpha = 0.25$ achieves a lower sentence ranking in all cases. However, in HotpotQA and Loogle (long), $\alpha = 0$ is much faster than $\alpha = 0.25$ (2x and 8x, respectively) and barely modifies the ranking, so we could sacrifice a bit of accuracy for speed in certain contexts.

3.4 Discussions

When using probability drop to find contributive attribution, we must consider an inherent issue coming from the method itself. That is, the alteration of the context can modify its meaning. Indeed, many sentences refer to previous ones, such that, if a sentence modifying the subject is removed, then the other sentences referring to it might imply wrong facts.

Additionally, the LLM can miss important facts in the initial long context that can be used when it becomes truncated. According to our definition, these should not be included in the attribution, but our relaxation can include them. If r contains them and the answer is correct, then the left-hand-side of the sufficiency condition in Equation (4) can become negative.

The Top-k filter t_k plays a strategic role in the rapid termination of the algorithm. The value and softmax thresholds can sometimes prune the tree but cannot always do so at every step, and thus do not bind the time complexity of TreeFinder. Every node has a maximum of t_k children, so the maximum is reached at $t_k * k$. The total number of nodes explored is this result multiplied by the depth of the tree, which is the logarithm of the number of sentences, and the two calls per iteration with the two initial calls for the empty and complete contexts. This leads to the number of forward calls

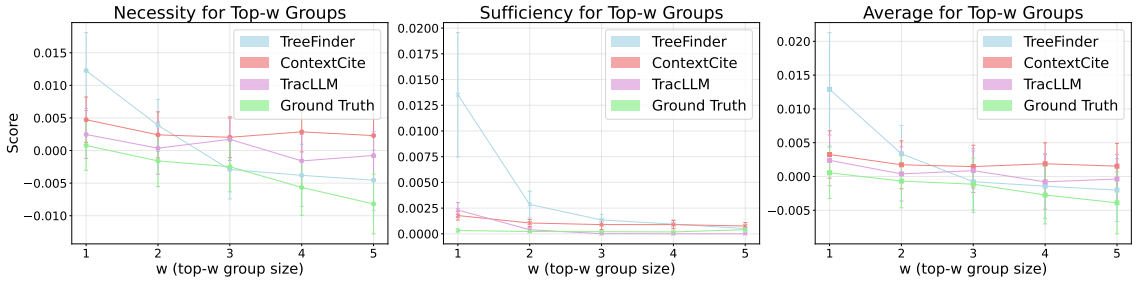


Figure 2: Sufficiency and necessity for the top sentences jointly in HotpotQA. Lower is better. TreeFinder lags behind for the first three sentences but quickly achieves a better overall attribution quality when more sentences are taken into account.

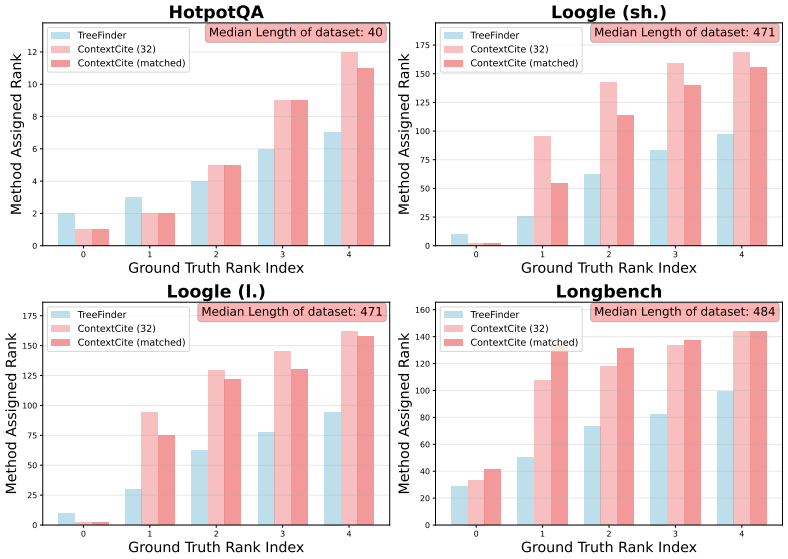


Figure 3: Median positions the k^{th} ground truth sentences (α=0.25). Median context length in sentences shown per dataset. Lower median positions are better. The sentences beyond the first are almost always found earlier in the ranking of TreeFinder. Increasing the number of ablations in ContextCite from 32 to match TreeFinder (50 for HotpoQA, 100 for the others) does not close the gap.

$\#M_{forward} = \mathcal{O}(4 + 2(k * t_k) * \log_{t_k}(|C|))$.

There can be an order of magnitude between the values of necessity and sufficiency scores. Therefore, we believe that balancing α such that both scores contribute meaningfully to the choice of attribution is difficult. A possible remedy to this issue could be the normalization of the scores at each depth in the tree.

4 Conclusion

This work presented a metric to measure the quality of contributive attributions with probability drops. Using this metric, we have developed a method, namely TreeFinder, to iteratively extract contributive attribution in question-answer environments using out-of-the-box LLMs without requiring task-specific training or surrogate models. TreeFinder employs a tree-based context pruning strategy, progressively refining the contributive attribution from coarse-grained chunks to individual sentences. All code is available at: https://anonymous.4open.science/r/TreeFinder-F3FD

Experimental results on HotpotQA demonstrate that TreeFinder ranks attributions, as commonly defined in previous works, better than ContextCite when they are composed of multiple sentences. This highlights the benefits of enforcing the necessity and sufficiency criteria in the search process.

Future work can focus on improving the chunking to group complementary sentences or incorporate meaningful structures of the context. Indeed, syntactically meaningful tree rep-

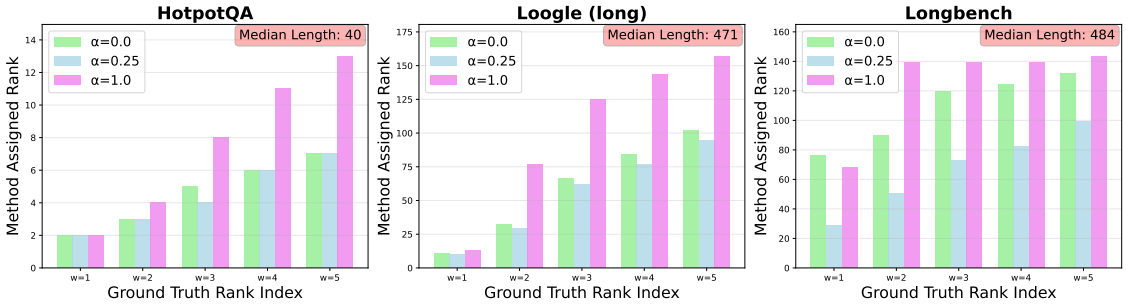


Figure 4: Ablation study of Tree Finder across $\alpha = 0.0, 0.25, 1.0$. Median context length in sentences shown per method. Lower median positions are better. $\alpha = 0$ can be enough to find attributions but Longbench necessitates both necessity and sufficiency.

resentations have been shown to improve the results of textual classification tasks (Nallapati and Allan, 2002). We believe that more accuracy and speed can be achieved by using textual segmentation techniques (Ghinassi et al., 2024). Alternative pruning decisions and node score estimations can potentially guide the process to be faster and more accurate.

Additionally, dropping some sentences X from the context for the remainder of the algorithm, as does TracLLM, can also improve termination speed. However, this changes the marginal probability from $P(A|Q, C)$ to $P(A|Q, C \setminus X)$ and could have unforeseen consequences on the attribution found.

As a final word, we believe that TreeFinder represents a step towards building more trustworthy and explainable QA systems, enabling users to not only receive reliable answers but also understand why those answers are generated, fostering confidence, and facilitating fact-checking.

Limitations

Due to the time and compute required to produce certain graphs, our work only explores results for a single model. This can impact the generalization of our findings.

Since TracLLM and ContextCite already compare themselves to traditional methods such as attention, shapley and others, we only compared ourselves to them to save time, compute and paper length. There may be cases where these traditional methods would have added insights to our analysis.

Acknowledgments

Omitted for submission.

References

David Alvarez-Melis and Tommi S. Jaakkola. 2017. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 412–421. Association for Computational Linguistics.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, and 11 others. 2025. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv: 2506.09985*.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 3639–3664. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. "O'Reilly Media, Inc.".

Marc Brinner and Sina Zarrieß. 2023. Model interpretability and rationale extraction by input

mask optimization. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13722–13744, Toronto, Canada. Association for Computational Linguistics.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xrag: Extreme context compression for retrieval-augmented generation with one token. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Yung-Sung Chuang, Benjamin Cohen-Wang, Shannon Zejiang Shen, Zhaofeng Wu, Hu Xu, Xi Victoria Lin, James Glass, Shang-Wen Li, and Wentau Yih. 2025. SelfCite: Self-supervised alignment for context attribution in large language models. *arXiv preprint arXiv:2502.09604*.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. 2025. ContextCite: Attributing model generation to context. *Advances in Neural Information Processing Systems*, 37:95764–95807.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.

Iacopo Ghinassi, Lin Wang, Chris Newell, and Matthew Purver. 2024. Recent trends in linear text segmentation: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3084–3095, Miami, Florida, USA. Association for Computational Linguistics.

Haowen Hou, Fei Ma, Binwen Bai, Xinxin Zhu, and Fei Yu. 2024. Enhancing and accelerating large language models via instruction-aware contextual compression. *arXiv preprint arXiv: 2408.15491*.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv: 2404.06654*.

Lei Huang, Xiaocheng Feng, Weitao Ma, Yuxuan Gu, Weihong Zhong, Xiachong Feng, Weijiang Yu, Weihua Peng, Duyu Tang, Dandan Tu, and Bing Qin. 2024. Learning fine-grained grounded citations for attributed large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14095–14113, Bangkok, Thailand. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Jinhyuk Lee, Anthony Chen, Zhuyun Dai, Dheeru Dua, Devendra Singh Sachan, Michael Boratko, Yi Luan, Sébastien M. R. Arnold, Vincent Perot, Siddharth Dalmia, Hexiang Hu, Xudong Lin, Panupong Pasupat, Aida Amini, Jeremy R. Cole, Sebastian Riedel, Iftekhar Naim, Ming-Wei Chang, and Kelvin Guu. 2024. Can long-context language models subsume retrieval, RAG, SQL, and more? *arXiv preprint arXiv: 2406.13121*.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2024. LooGLE: Can long-context language models understand long contexts? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 16304–16333. Association for Computational Linguistics.

Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2025. Prompt compression for large language models: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 7182–7195. Association for Computational Linguistics.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, and Nat McAleese. 2022. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv: 2203.11147*.

Ramesh Nallapati and James Allan. 2002. Capturing term dependencies using a language model based on sentence trees. In *Proceedings of the eleventh international conference on Information and knowledge management*, pages 383–390.

Lize Pirenne, Samy Mokeddem, Damien Ernst, and Gilles Louppe. 2025. Exploration of rationale-extraction methods for closed-domain question answering with a new sentence-level rationale dataset. In *Natural Language Processing and Information Systems*, pages 3–13, Cham. Springer Nature Switzerland.

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*, KDD '16, page 1135–1144, New York, NY, USA. Association for Computing Machinery.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2019. Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128:336–359.

Weijie Su. 2025. Do large language models (really) need statistical foundations? *arXiv preprint arXiv: 2505.19145*.

Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. 2025. Crossing the reward bridge: Expanding RL with verifiable rewards across diverse domains. *arXiv preprint arXiv: 2503.23829*.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. 2022. Will we run out of data? Limits of LLM scaling based on human-generated data. *arXiv preprint arXiv: 2211.04325*.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. 2025a. Hierarchical reasoning model. *arXiv preprint arXiv: 2506.21734*.

Yanting Wang, Wei Zou, Runpeng Geng, and Jinyuan Jia. 2025b. TracLLM: A generic framework for attributing long context LLMs. *arXiv preprint arXiv: 2506.04202*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Theodora Worledge, Judy Hanwen Shen, Nicole Meister, Caleb Winston, and Carlos Guestrin. 2024a. Unifying corroborative and contributive attributions in large language models. In *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 665–683, Los Alamitos, CA, USA. IEEE Computer Society.

Theodora Worledge, Judy Hanwen Shen, Nicole Meister, Caleb Winston, and Carlos Guestrin. 2024b. Unifying corroborative and contributive attributions in large language models. In *2024 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 665–683.

An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, and 9 others. 2025. Qwen2.5-1M technical report. *arXiv preprint arXiv: 2501.15383*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. Compact: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 21424–21439. Association for Computational Linguistics.

Mo Yu, Shiyu Chang, Yang Zhang, and Tommi Jaakkola. 2019. Rethinking cooperative rationalization: Introspective extraction and complement control. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-*

IJCNLP), pages 4094–4103, Hong Kong, China. Association for Computational Linguistics.

Wenbo Zhang, Tong Wu, Yunlong Wang, Yong Cai, and Hengrui Cai. 2023. Towards trustworthy explanation: On causal rationalization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 41715–41736. PMLR.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv: 2307.15043*.

A Algorithms

Algorithm 3 separates the context, seen as a list of sentences, into a maximum of k chunks, each of which contains a minimum of $total\_length/k$ characters except the last. It also produces a mapping from the block indices to the set of sentences indices they contain.

Algorithm 3 Chunkify

Input: Context $C = (s_1, \ldots, s_n)$, character lengths $\text{len}(s_i)$, maximum number of chunks $k > 1$
Output: $(\mathcal{C}_k, \text{map})$ where $\mathcal{C}_k = [C^{(1)}, \ldots, C^{(m)}]$ and map $= [I^{(1)}, \ldots, I^{(m)}]$ with $I^{(j)} = [i \mid s_i \in C^{(j)}]$
$L \leftarrow \sum_{i=1}^{n} \text{len}(s_i)$
$B \leftarrow \lceil \frac{L}{k} \rceil$ ▷ Char. per chunk
$\mathcal{C}_k \leftarrow [\,]$, map $\leftarrow [\,]$
$C' \leftarrow [\,]$, $I' \leftarrow [\,]$, $\ell \leftarrow 0$
for $i \leftarrow 1$ **to** n **do**
    append s_i **to** C'
    append i **to** I'
    $\ell \leftarrow \ell + \text{len}(s_i)$
    if $\ell > B$ **then**
        append C' **to** $\mathcal{C}_k$
        append I' **to** map
        $C' \leftarrow [\,]$, $I' \leftarrow [\,]$, $\ell \leftarrow 0$
    end if
end for
if C' **is not** empty **then**
    append C' **to** $\mathcal{C}_k$
    append I' **to** map
end if
return $(\mathcal{C}_k, \text{map})$

B Results on other datasets

In this section are provided the necessity and sufficiency scores for the first sentences, grouped and not grouped, of the remaining datasets (Longbench, Loogle).

We compute our metric from Equation (7) with $\alpha = 0.5$ for every sentence in a dataset and get a "Ground Truth" ranking. These values are also computed for the first ten sentences given by each method. The box plots for these values are given in Figures 5, 6, 8 and 10 for datasets 1, 2a, 2b, and 3, respectively.

B.1 Hotpot-QA

When the sentences are scored individually for HotpotQA in Figure 5, we can see that the first is ranked better by TracLLM, then the following ones are ranked better by TreeFinder.

B.2 LongBench-v2

For the sufficiency in Figure 7, we can observe that ContextCite is leading for the first sentence,

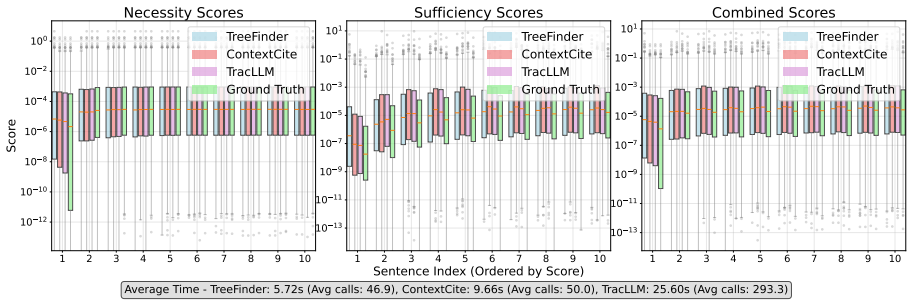


Figure 5: Sufficiency, necessity and average box plots for the top sentences individually in HotpotQA. Lower is better. The first sentence found with ContextCite and TracLLM is of high quality, while the following ones tend to bring little. TreeFinder misses the first but ranks the following ones better.

then TracLLM takes the lead. Afterwards, the curves interlace and the differences become too small to interpret meaningfully. In Figure 6, TracLLM has a pretty clear advantage in sufficiency.

B.3 Loogle Short

TracLLM consistently finds groups and individual sentences with lower necessity and sufficiency, as can be seen in Figures 8 and 9. When seen as individual sentences, the sufficiency scores remain far from the ground truth for all methods. Interestingly, when taken as a group, by the fourth sentence TracLLM reaches the ground-truth value.

We note that despite TreeFinder being worse in these graphs, it outperforms ContextCite in Figure 3.

B.4 Loogle Long

In Figure 10, TracLLM seems to perform better. In Figure 11, TreeFinder retains its slight advantage in necessity scores while TracLLM leads in sufficiency scores.

C Comparison with a specific example

In this section, we provide a sample from HotpotQA where the answer provided by the model is correct along with the support sentences provided in the dataset. We highlight the five best sentences as given by each method and match them with the support if possible in Tables 1 to 3. The support sentences should not necessarily be matching the contributive attributions since they are corroborative attributions.

We observe that all methods find contributive attributions that are aligned with the corroborative attributions in these cases.

Question: All: "What Golden Globe nominated American actor from Juilliard School played a role in the 1986 romantic drama film, Children of a Lesser God, directed by Randa Haines?"

Support: "Children of a Lesser God is a 1986 American romantic drama film directed by Randa Haines and written by Hesper Anderson and Mark Medoff.", "An adaptation of Medoff's Tony Award–winning stage play of the same name, the film stars Marlee Matlin (in an Oscar-winning performance) and William Hurt as employees at a school for the deaf: a deaf custodian and a hearing speech teacher, whose conflicting ideologies on speech and deafness create tension and discord in their developing romantic relationship.", "William McChord Hurt (born March 20, 1950) is an American actor.", "He received his acting training at the Juilliard School and began acting on stage in the 1970s.", "Hurt made his film debut in 1980 as a troubled scientist in Ken Russell's science-fiction feature "Altered States", for which he received a Golden Globe nomination for New Star of the Year."

Answers: TreeFinder and TracLLM: "The Golden Globe nominated American actor from the Juilliard School who played a role in the 1986 romantic drama film *Children of a Lesser God*, directed by Randa Haines, is **William Hurt**. William Hurt received his acting training at the Juilliard School and made his film debut". ContextCite: "The Golden Globe nominated American actor from the Juilliard School who played a role in the 1986 romantic drama

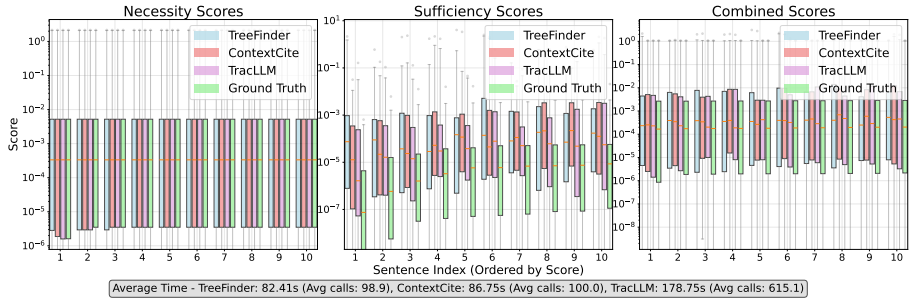


Figure 6: Sufficiency, necessity and Average box plots for the top sentences individually in LongBench-v2. Lower is better. The large error bars obscure any conclusion for the necessity. The sufficiency is better for TracLLM.

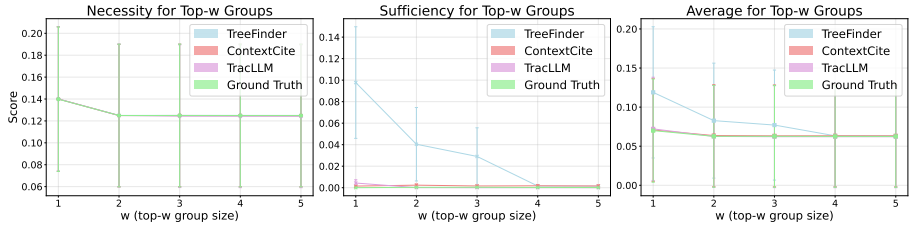


Figure 7: Sufficiency and necessity for the top sentences jointly in LongBench-v2. Lower is better. The large error bars obscure any conclusion for the necessity. The sufficiency of the rationale by TreeFinder is lacking a first but quickly converges to the same values as ContextCite and TracLLM.

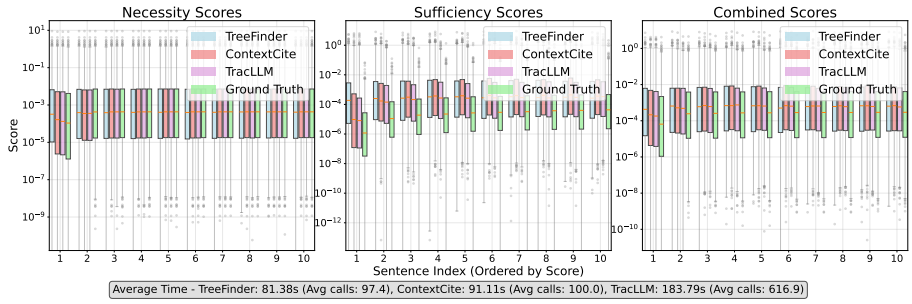


Figure 8: Sufficiency, necessity and average box plots for the top sentences individually in Loogle (short). Lower is better. TrackLLM consistently outperforms other methods.

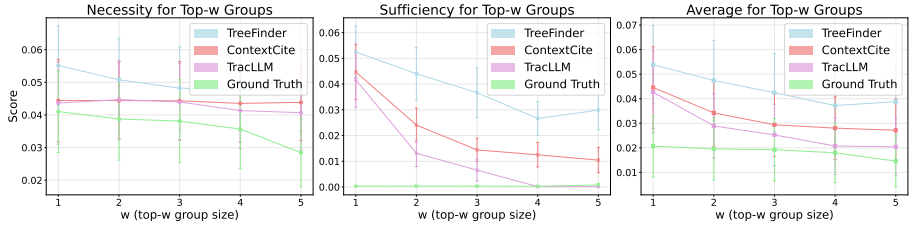


Figure 9: Sufficiency and necessity for the top sentences jointly in Loogle (short). Lower is better. TrackLLM consistently outperforms other methods.

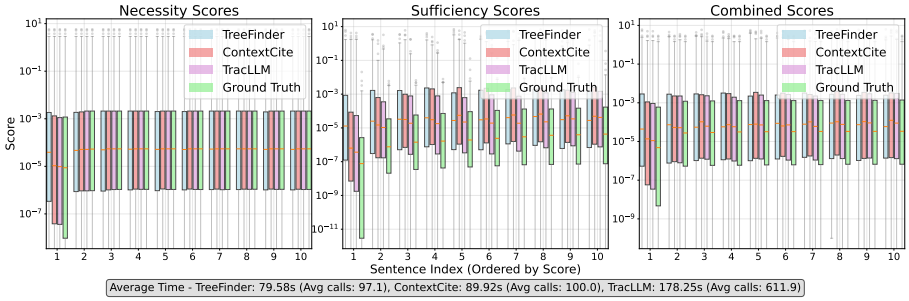


Figure 10: Sufficiency, necessity and average box plots for the top sentences individually in Loogle (long). Lower is better. TracLLM leads for the eight first sentences then is tied with TreeFinder. Both remain far from the ground truth.

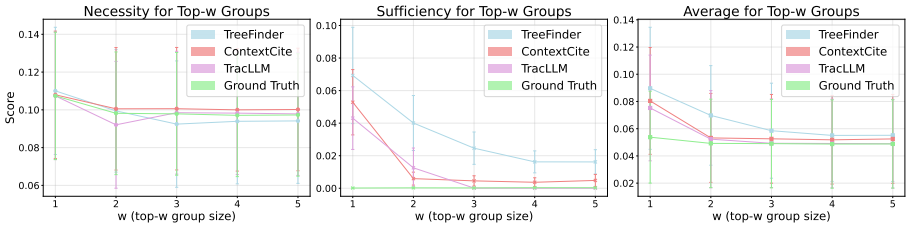


Figure 11: Sufficiency and necessity for the top sentences jointly in Loogle (long). Lower is better. There is a slight advantage in necessity for TreeFinder, but TracLLM has a clear lead in sufficiency.

film *Children of a Lesser God*, directed by Randa Haines, is **William Hurt**.### Explanation: - **William Hurt** is mentioned as having received his acting training at the”

Rank	Sentence
1	#2
2	#3
3	#4
4	Two years later he made his breakthrough by starring in the romantic comedy ”Risky Business” (1983), which garnered Cruise his first nomination for the Golden Globe Award for Best Actor – Motion Picture Musical or Comedy.
5	In 1986, Cruise played a fighter pilot in the Tony Scott-directed action drama ”Top Gun” (the highest-grossing film that year), and also starred opposite Paul Newman in the Martin Scorsese-directed drama ”The Color of Money”.

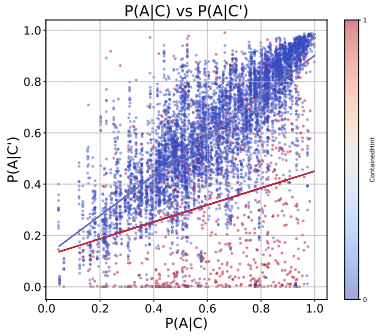
Table 1: Top 5 sentences from TreeFinder.

Rank	Sentence
1	#3
2	#2
3	#1
4	The film was directed by Randa Haines, and was released directly on television.
5	#4

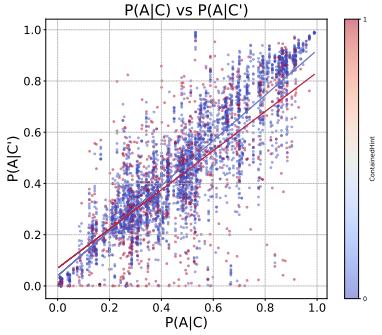
Table 2: Top 5 sentences from ContextCite.

Rank	Sentence
1	#3
2	#4
3	He subsequently played a leading role, as a lawyer who succumbs to the temptations of Kathleen Turner, in the neo-noir "Body Heat" (1981), and, as Arkady Renko, in Gorky Park (1983).
4	Tom Cruise is an American actor and producer who made his film debut with a minor role in the 1981 romantic drama "Endless Love".
5	She is perhaps most famous for directing the critically acclaimed feature film "Children of a Lesser God" (1986), which starred William Hurt and Marlee Matlin, for which Matlin won the 1987 Academy Award as Best Actress.

Table 3: Top 5 sentences from TracLLM.



(a) Loogle (short)



(b) Loogle (long)

Figure 12: $P(A|C')$ against $P(A|C)$ in the case where $C' \subset C$ has a chunk removed. It is colored red if it contained part of the corroborative attribution and blue in the opposite. The chunk length is $|C'| = 8000$ characters. The difference of distribution shows the ability of the probability drop method to find corroborative attributions for a given model and dataset. Linear interpolations are drawn for reference.

D Proximity to corroborative attribution

We provide here an illustration of why the corroborative and contributive attributions are not the same, especially as the model has more difficulties responding correctly. To do this, we plot the probabilities of dropped chunks in Figure 12 against their original probabilities for each data point ($x = const$ in the figure) and check that we can separate the chunks that possess a corroborative attribution (red) from those that do

not (blue).

In Loogle (short), the chunks that do not contain a corroborative attribution have their probabilities only slightly modified and remain close to $x = y$, while the other are shifted toward $y = 0$. This means that corroborative attributions are aligned with contributive attributions.

However, in Loogle (long), the two distributions are too similar, as indicated by both trendlines following $x = y$; removing a corroborative attribution no longer changes the probability of the answer. This can be the result of the model missing these sentences in the global context in the first place. Here, corroborative and contributive attributions are no longer aligned.

We highlight that discrepancies between the two types of attributions are key to understanding model uncertainty and should not be confused for each other.