Attention with Dependency Parsing Augmentation for Fine-Grained Attribution

Qiang Ding[1,2,3], Lvzhou Luo[1,2,3], Yixuan Cao[1,2,3*], Ping Luo[1,2,3,4*]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China
[2]State Key Lab of AI Safety, Beijing 100094, China
[3]University of Chinese Academy of Sciences, Beijing 100049, China
[4]Peng Cheng Laboratory, Shenzhen 518066, China
{dingqiang22z,luolvzhou23s,caoyixuan,luop}@ict.ac.cn

Abstract

To assist humans in efficiently validating RAG-generated content, developing a fine-grained attribution mechanism that provides supporting evidence from retrieved documents for every answer span is essential. Existing fine-grained attribution methods rely on model-internal similarity metrics between responses and documents, such as saliency scores and hidden state similarity. However, these approaches suffer from either high computational complexity or coarse-grained representations. Additionally, a common problem shared by the previous works is their reliance on decoder-only Transformers, limiting their ability to incorporate contextual information after the target span. To address the above problems, we propose two techniques applicable to all model-internals-based methods. First, we aggregate token-wise evidence through set union operations, preserving the granularity of representations. Second, we enhance the attributor by integrating dependency parsing to enrich the semantic completeness of target spans. For practical implementation, our approach employs attention weights as the similarity metric. Experimental results demonstrate that the proposed method consistently outperforms all prior works.

1 Introduction

Retrieval-augmented generation (RAG) enhances the factual recall of LLMs and has been applied in knowledge-intensive NLP tasks such as open-domain question answering (Lewis et al., 2020). However, the generated content may still deviate from the retrieved documents (Niu et al., 2024), necessitating careful verification, especially when used in safety-critical domains like finance. To assist users in validating LLM-generated responses, QA systems must provide supporting evidence, also referred to as attribution or citation (Li et al., 2023a). Furthermore, developing a fine-grained attribution mechanism that supplies evidence for arbitrary answer spans (as illustrated in Fig. 1) is particularly beneficial, as it allows users to efficiently verify the accuracy of individual segments within complex, long-form answers.

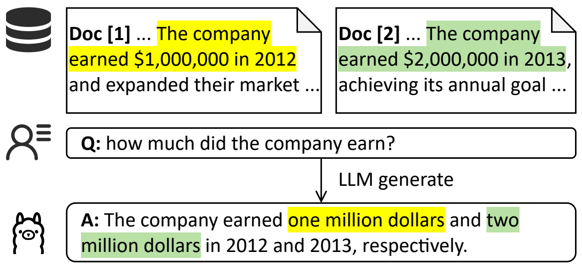

Doc [1] ... The company earned $1,000,000 in 2012 and expanded their market ...
Doc [2] ... The company earned $2,000,000 in 2013, achieving its annual goal ...
Q: how much did the company earn?
LLM generate
A: The company earned one million dollars and two million dollars in 2012 and 2013, respectively.

Figure 1: An example of fine-grained attribution, i.e., finding evidence from the retrieved documents for arbitrary target spans. Each highlighted span in the answer is a target, with evidence in the same background color.

Fine-grained attribution has been addressed by only two model-internals-based approaches: CCI (Yin and Neubig, 2022; Sarti et al., 2023; Qi et al., 2024) and HSSAvg (Average Hidden State Similarity method, Phukan et al., 2024). CCI leverages saliency scores for attribution but requires gradient back-propagation for each target token, resulting in high computational complexity. HSSAvg operates by (1) measuring the similarity between the average hidden states of the token span and those of a sliding window with a fixed size W over the documents, and (2) selecting the highest-scoring window as the evidence. However, the averaging operation over the target span introduces coarse granularity, limiting the precision of its representations. A further challenge common to both methods lies in their reliance on the internal representations of decoder-only Transformers, which lack access to tokens that appear after the target span, constraining their contextual understanding (see Sec. 3.3 for details).

[*]Corresponding author: Yixuan Cao and Ping Luo.

Building on insights into the limitations of previous methods, we propose a novel, effective, and efficient model-internals-based approach. This approach consists of two techniques applicable to all model-internals-based approaches. The first technique involves aggregating token-wise evidence using set unions, addressing the coarse-granularity issue in averaging hidden states while eliminating the need to recompute these averages for each new target span (see Sec. 3.2 for details). The second technique incorporates dependency parsing to enhance the attributor by integrating related tokens into the attribution of the target token, as illustrated by Fig. 2 (see Sec 3.3 for details). For practical implementation, we utilize attention weights as the similarity metric due to their faster computation compared to gradient back-propagation and superior empirical performance over hidden state similarity. Our experiments demonstrate that the proposed method surpasses all baseline approaches and generalizes effectively to sentence-level attribution, highlighting its practical value.

We also address two practical challenges in implementing our method: (1) the inaccessibility of attention weights from black-box sources and (2) the high GPU memory consumption required for their computation. To tackle the first challenge, we approximate attention weights using open-source LLMs. For the second, we apply engineering optimizations to enable more efficient attention weight calculations, achieving significantly faster performance than prior approaches.

In summary, our contributions are:

1. We propose a novel model-internals-based fine-grained attributor that aggregates token-wise evidence through set unions, addressing the coarse granularity induced by averaging hidden states.
2. We propose leveraging dependency parsing to enhance decoder-based attributors by enriching the semantics of the target span.
3. The proposed method utilizes LLM attention weights for attribution and incorporates an optimized routine for efficient computation.
4. Our method sets a new state-of-the-art in fine-grained attribution and demonstrates strong generalization to sentence-level attribution.

2 Related Work

This section reviews related work across four aspects: (1) attribution, (2) non-faithfulness detection, (3) fact verification, and (4) other works using attention weights.

Attribution. Attribution methods can be categorized into three classes: self-generated attribution (or self-citation) (Thoppilan et al., 2022; Nakano et al., 2021; Menick et al., 2022; Slobodkin et al., 2024), retrieval-based attribution (Gao et al., 2023a; Sancheti et al., 2024), and model-internals-based attribution (Qi et al., 2024; Cohen-Wang et al., 2024; Phukan et al., 2024). The self-generated approach prompts or fine-tunes the LLM to produce citations during answer generation, but it does not guarantee that each statement has adequate citations (Gao et al., 2023b). The retrieval-based approach provides sentence-level citations by retrieving relevant results from external documents, ensuring adequate citations for each sentence. Model-internals-based attribution relies on similarity metrics such as saliency scores or hidden state similarities to align the response with the prompt, retrieving evidence from prompt tokens with the highest similarity. Of these three approaches, the first two primarily target sentence-level attribution, while some methods from the third category explore fine-grained attribution (Phukan et al., 2024; Qi et al., 2024), which has been discussed in the Introduction.

Non-Faithfulness Detection. Attribution plays a valuable role in detecting non-faithfulness (He et al., 2022; Niu et al., 2024), a type of hallucination detection (Wang et al., 2020; Muhlgay et al., 2024) that assesses whether the generated text is inconsistent with the *input documents*. Many non-faithfulness detection methods are black-box (Feng et al., 2023; Yue et al., 2023; Bazaga et al., 2024; Mishra et al., 2024; Zhang et al., 2024a), resembling traditional Natural Language Inference (NLI) models but operating with longer contexts and greater complexity. In these methods, attribution can be integrated into the detection pipeline to break down the complex NLI task, enhancing interpretability by providing supporting evidence for each statement. An alternative approach to hallucination detection involves the model-internals methods (Li et al., 2023b; Hu et al., 2024; Chuang et al., 2024), which offer the potential to unify attribution and non-faithfulness detection within a single model. However, these methods have so far overlooked attribution.

Fact Verification. Fact verification integrates hallucination detection with attribution, requiring not only an assessment of whether a claim is hallucinated but also evidence that supports or contradicts the claim (Guo et al., 2022). In this domain, retrieval-based attribution is commonly used (Min et al., 2023). While early studies (before the LLM era) employed attention weights from GNNs or other small models for attribution (Yang et al., 2019; Liu et al., 2020; Chen et al., 2022), using LLM attention weights for attribution remains unexplored in fact verification literature.

Other Works Using Attention Weights. In attribution, attention weights have been utilized in small models like BERT (Clark et al., 2019; Kobayashi et al., 2020). However, with the advent of LLMs, their use for attribution remains unexplored due to high memory costs and the lack of support from popular inference frameworks and proprietary models. For hallucination mitigation, Huang et al. (2024) introduces a method that retrospects LLM generation whenever a hallucination-related "aggregate pattern" appears in the attention weights.

3 Method

We begin by outlining the problem settings in Sec. 3.1, followed by a description of the attention-based method in Sec. 3.2 and the dependency parsing augmentation in Sec. 3.3. Finally, we introduce our approach to addressing challenges in real-world applications in Sec. 3.4.

3.1 Problem Settings

Suppose an LLM generates a response $\mathbf{r}$ given the prompt composed of the retrieved documents $\mathbf{d}$ and the question $\mathbf{q}$. The goal of *fine-grained attribution* is, given an arbitrary target span $\mathbf{t} \subset \mathbf{r}$, to identify evidence from $\mathbf{d}$ that supports the *atomic facts* (short statements that each contain one piece of information, Min et al., 2023) that involves $\mathbf{t}$. An example is provided in Fig. 1, where for target span "one million dollars", the atomic fact is "The company earned one million dollars in 2012", and the evidence is "The company earned \$1,0000,00 in 2012" in the first document.

3.2 The Basic Algorithm

To solve fine-grained attribution, we propose attributing each target token in the span and then aggregating the attributions of tokens by the union of sets. Formally, let the documents, question, and response of the LLM be three sequences of tokens $\mathbf{d} = (d_1, d_2, ..., d_c)$, $\mathbf{q} = (q_1, q_2, ..., q_m)$, and $\mathbf{r} = (r_1, r_2, ..., r_n)$, respectively. Assume the LLM can output a similarity metric $\mathbf{S} \in \mathbb{R}^{n\times(c+m)}$ between response tokens and prompt tokens (e.g., attention weights, hidden state similarity, and saliency scores). Then the attribution scores are defined as[1]

$$w(r_i, d_j) := \begin{cases} \mathbf{S}_{i,j}, \text{if } \mathbf{S}_{i,j} \geq \text{top-}k(\mathbf{S}_i) \\ 0, \text{otherwise} \end{cases} \tag{1}$$

where $1 \leq i \leq n$, $1 \leq j \leq c$, top-$k(\mathbf{x})$ is the k-th largest component in vector $\mathbf{x}$, and k is a hyper-parameter. With these attribution scores, each response token r_i is attributed to $e(r_i) := \{d_j | w(r_i, d_j) > 0\}$, and the aggregated evidence for the target span $\mathbf{t}$ is $e(\mathbf{t}) := \bigcup_{r_i\in\mathbf{t}} e(r_i) = \{d_j | \sum_{r_i\in\mathbf{t}} w(r_i, d_j) > 0\}$ with each token $d_j \in e(\mathbf{t})$ associated with an attribution score $w(\mathbf{t}, d_j) := \sum_{r_i\in\mathbf{t}} w(r_i, d_j)$. Furthermore, to avoid noisy and fragmented attribution[2], we remove the isolated evidence tokens that are at least τ tokens away from other evidence tokens. The pseudo-code of the whole algorithm is provided in Algorithm 1. In addition, the token-wise attributions $e(r_i)$ with scores $w(r_i, \cdot)$, $i = 1, 2, ..., n$, can be reused by any target span in the same response, saving computational cost. We utilize this trick but omit it in the pseudo-code for brevity.

3.3 Dependency Parsing Augmentation

We observe a common defect in previous works: the representations of decoder-only models cannot see the context that follows the target span, potentially missing relevant factual information that appears later. For instance, as shown in Fig. 1, decoder-only attributors cannot see "million dollars" or "in 2012" at the position of "one", increasing the difficulty of fine-grained attribution. A straightforward solution is to allow the attributor to access subsequent context. However, extending the accessible scope to the entire context or full sentence is empirically shown to be ineffective in fully addressing the issue (see Sec. 4.2.1).

To address this issue, we propose **DEpendency Parsing augmentation** (DEP). This method first recognizes key elements — such as subject, object, and predicate — within atomic facts that contain the target span using dependency parsing. For example, as illustrated in Fig. 2, given the target token "one", the method collects atomic fact elements as

[1]Here we assume the documents are located at the start of the prompt. If not, we need to replace $\mathbf{S}_{ij}$ with $\mathbf{S}_{i,j+o}$, where o is the offset of the documents in the prompt.

[2]An example of noisy attribution is the widely-known attention sink (Kobayashi et al., 2020; Xiao et al., 2024; Huang et al., 2024; Zhang et al., 2024b) where a large amount of response tokens distribute a lot of attention weights to several isolated meaningless prompt tokens (e.g., BOS and punctuation marks), hindering the word alignment of attention weights

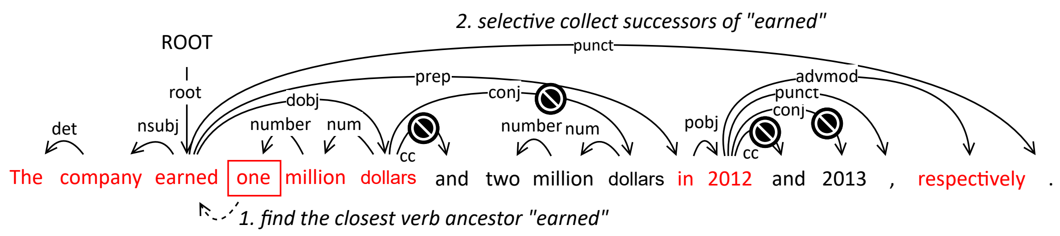

2. selective collect successors of "earned"
ROOT
root
punct
prep
dobj
conj
advmod
punct
conj
det
nsubj
number
num
number
num
pobj
cc
cc
The company earned one million dollars and two million dollars in 2012 and 2013 , respectively .
1. find the closest verb ancestor "earned"

Figure 2: An illustration of dependency parsing augmentation. Suppose the target span is the token "one". The method first finds the closest verb ancestor of "one", i.e., "earned", and then collects successors of "earned", excluding unrelated coordinating constituents "two million dollars" and "2013". The resulting augmentation tokens are in red. Lastly, the attribution of "one" is updated by summing the attributions of the augmentation tokens.

shown in red: "The company earned one million dollars ... in 2012 ... respectively." It then leverages the attributions of these atomic fact elements to enhance the attribution of the target span. The detailed algorithm proceeds as follows.

1. Recognizing Atomic Fact Elements. We assume that the elements of atomic facts can be extracted from the text. For each token r_i, let these elements be $\mathcal{A}(r_i) \subset \mathbf{r}$. This extraction is approximated by leveraging the dependency parse tree, constructed using the LAL-Parser (Mrini et al., 2020), applied to the local sentence, as shown in Fig. 2. Since each node in this tree corresponds to a word while the attribution pertains to tokens, we need to align the two. For simplicity, we temporarily assume they are the same, with alignment details provided in Appendix B. Initially, we set $\mathcal{A}(r_i) \leftarrow \{r_i\}$. Then, the algorithm appends $\mathcal{A}(r_i)$ with r_i's closest verb ancestor v and all v's successors except punctuation marks. If $\mathcal{A}(r_i)$ contains coordinating structures (e.g., tokens connected by "and" or "or"), we use a rule-based subroutine (described in Appendix C) to eliminate irrelevant coordinating constituents. For example, in Fig. 2, $\mathcal{A}$("one") includes the closest verb ancestor "earned" and all its successors except the irrelevant coordinating constituents "two million dollars" and "2013."

2. Augmenting Attribution. Using $\mathcal{A}(r_i)$, the attribution of token r_i is augmented as $e(r_i) \leftarrow \bigcup_{a \in \mathcal{A}(r_i)} e(a)$, with attribution scores updated to $w(r_i, d_j) \leftarrow \sum_{a \in \mathcal{A}(r_i)} w(a, d_j)$. The aggregate attribution is then computed based on the updated token-wise attribution, following the same approach as the basic algorithm.

3.4 Attention and its Computational Challenges

We choose attention weights as similarity scores due to their strong empirical performance. The method is outlined as follows. Let the attention weights between the response and the prompt at the l-th layer and the h-th head of the LLM be $\mathbf{A}^{(l,h)} \in \mathbb{R}^{n \times (c+m)}$, where the LLM has L layers and H heads. Specifically, $\mathbf{A}_i^{(l,h)}$ is the attention weights used to predict r_i. For a Transformer decoder, $\mathbf{A}_{ij}^{(l,h)}$ is the attention weight from r_{i-1} to d_j. For a Transformer encoder, $\mathbf{A}_{ij}^{(l,h)}$ is the attention weight from r_i to d_j. Based on these attention weights, the similarity score is computed by averaging the attention weights from a selected layer L^* across all heads: $\mathbf{S} := \frac{1}{H} \sum_{h=1}^{H} \mathbf{A}^{(L^*,h)}$.

Another potential similarity metric is cosine similarity among hidden states. Let $\mathbf{h}^{\mathrm{p}} \in \mathbb{R}^{(c+m) \times d}$ and $\mathbf{h}^{\mathrm{r}} \in \mathbb{R}^{r \times d}$ denote the hidden states of the prompt and the response, respectively, where d is the dimension of hidden states. The hidden state cosine similarity is computed as: $\mathbf{S}_{ij} := \frac{\mathbf{h}_i^{\mathrm{r}} \cdot \mathbf{h}_j^{\mathrm{p}}}{\|\mathbf{h}_i^{\mathrm{r}}\| \cdot \|\mathbf{h}_j^{\mathrm{p}}\|}$. We will empirically show that attention-based similarity is better than hidden state cosine similarity.

However, the attention-based approach has the following two challenges in real-world application.

Inaccessible Attention Weights. The attention weights are inaccessible under the following conditions: (1) the LLM is proprietary, such GPT-4 (Achiam et al., 2023); (2) the inference framework, such as vLLM (Kwon et al., 2023), does not support outputting attention weights; (3) the response may be generated by other black-box sources, e.g., humans. To address this, we leverage open-sourced LLMs to approximate the attention weights. Empirical results demonstrate that these approximate attention weights offer reliable attribution.

High Memory Overload. In the Huggingface (Wolf et al., 2020) implementation[3], attention weights and the KV cache are stored simultaneously in GPU memory when calculating attention weights from the response to the prompt. This dual memory load can quickly exhaust the GPU memory. Therefore, we propose a specialized routine for calculating attention weights. We calculate attention weights after the generation process. The model processes the concatenation of the response and the prompt with FlashAttention (Dao et al., 2022) and NF4 quantization (Dettmers et al., 2023), exiting early after the $(L^* - 1)$-th layer without producing attention weights or storing the KV cache. From the resulting hidden states, we calculate the attention weights from the response to the prompt, yielding the desired similarity metric.

4 Experiments

We begin by evaluating our methods on fine-grained attribution in Sec. 4.1, with the corresponding ablation study presented in Sec. 4.2. We then assess the faithfulness of our methods to the generators in Sec. 4.3. The evaluation of sentence-level attribution is in Sec. 4.4. Lastly, we compare the latency of all fine-grained attributors in Sec. 4.5.

4.1 Evaluating Fine-Grained Attribution

Benchmark. Following Phukan et al. (2024), we use two datasets QuoteSum (Schuster et al., 2024) and VERI-GRAN (Phukan et al., 2024) to evaluate fine-grained attribution. Each instance of these two datasets contains a question, retrieved passages, and an answer, where the question and the retrieved passages combine to form the prompt[4]. The task is to identify the evidence passage (or evidence sentence in the case of VERI-GRAN) corresponding to each context-sensitive span in the answer, where the span is determined based on the similarity of LLM hidden states (Phukan et al., 2024), and the evidence passage is labeled by human annotators. Additional details about the datasets are provided in Table 6. The evaluation metric is the accuracy of the predicted evidence passage for these spans. To apply our method and CCI to this benchmark, we predict the passage with the highest cumulative attribution score, i.e., $\arg\max_{\mathbf{d}_i} \sum_{d \in \mathbf{d}_i} w(\mathbf{t}, d)$, where $\mathbf{d}_i$ is the i-th passage and $\mathbf{t}$ is the target span. For HSSAvg, we select the passage containing the highest-scoring sliding window as the prediction.

[3]The implementation details are provided in Appendix D.

[4]We use the same prompt template as Phukan et al. (2024).

Baselines. We include the only two prior works on fine-grained attribution as our baselines, CCI (Sarti et al., 2023; Qi et al., 2024), also known as contrastive feature attribution (Yin and Neubig, 2022), and HSSAvg (Phukan et al., 2024). Additionally, we include the reported results of GPT-4 from Phukan et al. (2024).

Models. We conduct experiments using two models: Llama2 7B Chat(Touvron et al., 2023) and Qwen2 7B Instruct (Yang et al., 2024). For HSSAvg, hidden states are extracted from the $\lfloor \frac{L}{2} \rfloor$-th layer, as Phukan et al. (2024) report that this method performs well across different models when using hidden states from the middle layers. For our methods, the attention weights are from the $(\lfloor \frac{L}{2} \rfloor + 1)$-th layer — just above the hidden states used for HSSAvg — due to their strong empirical performance on validation sets. To address GPU memory limitations, we apply NF4 quantization to CCI[5].

Hyperparameters. We set the token-wise evidence size $k = 2$ and the threshold for recognizing isolated tokens $\tau = 2$ for our method (experiments on sensitivity to hyperparameters are presented in Appendix F). For HSSAvg, we configure the window size $W = 8$, as this setting yields good performance on validation sets. For CCI, following the original paper, we choose the top-scoring three tokens as the evidence for each target token.

Results. The accuracy of fine-grained attribution is reported in Table 1, where we refer to our basic algorithm and its augmented version as AttnUnion and AttnUnionDep, respectively. Across both datasets, AttnUnion achieves comparable performance to HSSAvg, while AttnUnionDep consistently outperforms HSSAvg and GPT-4, setting a new SOTA in fine-grained attribution and demonstrating the effectiveness of Dep.

4.2 Ablation Study

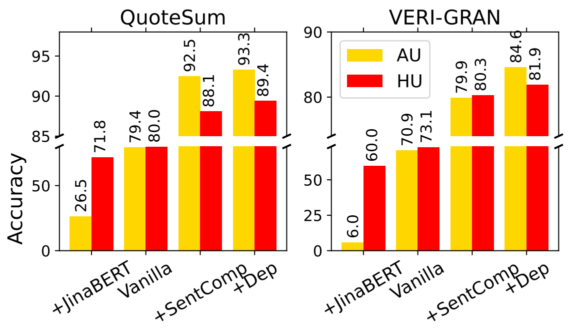
In this section, we conduct ablation studies about Dep, Attn, and Union.

4.2.1 Ablating Dep

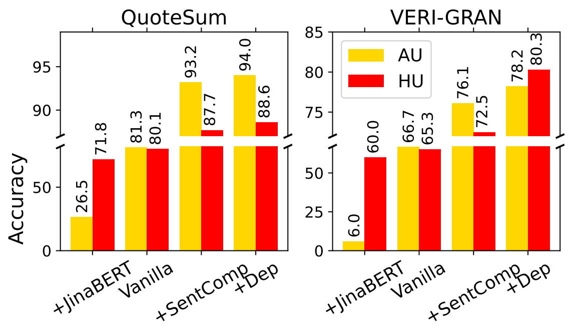
Research Question. Previous results have confirmed the effectiveness of Dep. However, an important question remains: *Are there any simpler alternatives to* Dep *that allow the model representations to see the context following the target span?*

Experiment Setting. We empirically compare two

[5]Our methods also use NF4 quantization, as Sec. 3 states.



(a) Qwen2 7B



(b) Llama2 7B

Figure 3: Results of ablating DEP. Here AU and HU represent ATTNUNION and HSSUNION, respectively.

	Model	QuoteSum	VERI-GRAN
	Baselines		
2-SHOT	GPT-4†	90.6	62.1
HSSAVG	Qwen2	80.4	67.1
	Llama2	77.1	64.5
	Llama2†	87.5	77.3
CCI	Qwen2	71.3	64.5
	Llama2	72.2	59.0
	Our Methods		
ATTNUNION	Qwen2	79.4	70.9
	Llama2	81.3	66.7
ATTNUNIONDEP	Qwen2	<u>93.3</u>	**84.6**
	Llama2	**94.0**	<u>78.2</u>

Table 1: Accuracy (%) of fine-grained attribution on QuoteSum and VERI-GRAN. The best and the second-best entries are marked in bold and underlined, respectively. Cited results are marked by †.

	Model	QuoteSum	VERI-GRAN
	UNION VS. AVG w/o DEP		
HSSUNION	Qwen2	80.0	73.1
	Llama2	80.1	65.3
HSSAVG	Qwen2	80.4	67.1
	Llama2	77.1	64.5
	UNION VS. AVG w/ DEP		
HSSUNIONDEP	Qwen2	89.4	81.9
	Llama2	88.6	80.3
HSSAVGDEP	Qwen2	79.3	62.8
	Llama2	72.2	60.7

Table 2: Accuracy (%) of UNION vs. AVG on QuoteSum and VERI-GRAN.

alternatives with DEP[6]: 1. extending the span to the entire local sentence (SENTCOMP); 2. leveraging representations from bidirectional attention models, such as BERT, for attribution. The benchmarks, backbones, and hyper-parameters remain consistent with those used in the previous experiment.

Specifically, the second alternative leverages JinaBERT's (Günther et al., 2023) attention weights or hidden state similarity (HSS) as the similarity metric. We refer to the HSS-based method as HSSUNION. The attention weights are from the 5th layer, while the hidden state similarities are from the final layer, as these configurations demonstrated strong performance on the validation sets.

Results & Insights. The results are shown in Fig. 3, demonstrating that all alternatives consistently underperform compared to DEP. The results indicate that recognizing atomic facts via DEP is more effective than extending the accessible context to the entire context or full sentence.

4.2.2 ATTN vs. HSS

Research Question. While calculating attention weights is significantly faster than gradient backpropagation, it incurs a similar computational cost to computing hidden state similarity. This raises the question: *Is using attention weights more effective than hidden state similarity?*

Results. We analyze the results from Fig. 3 by comparing adjacent bars of ATTNUNION and HSSUNION. Excluding the underperforming JINABERT and VANILLA settings, ATTNUNION outperforms HSSUNION in most cases, highlighting the effectiveness of using attention weights as attribution scores.

4.2.3 UNION vs. AVG

Research Question. The final component to examine is the aggregation by union. We pose the question: *Is* UNION *more effective than* AVG*?*

Experiment Setting. We compare UNION and

[6]Another more complicated alternative of DEP is discussed in Appendix I, which is empirically not as good as DEP.

	Model	QuoteSum		VERI-GRAN	
		`Qw.`	`Ll.`	`Qw.`	`Ll.`
Baselines					
Random		0.81	1.09	0.13	0.30
HSSAvg	Qwen2	5.85	7.94	3.47	5.69
	Llama2	5.97	8.05	**3.50**	<u>5.76</u>
HSSAvgDep	Qwen2	5.77	7.74	3.26	5.29
	Llama2	5.39	7.62	3.18	5.52
CCI	Qwen2	5.70	7.39	3.47	5.39
	Llama2	5.43	7.76	3.19	5.66
Oracle		6.41	8.49	4.03	6.60
Our Methods					
AttnUnion	Qwen2	4.08	5.53	2.39	4.40
	Llama2	4.60	5.96	2.61	4.51
AttnUnionDep	Qwen2	<u>6.17</u>	<u>8.09</u>	<u>3.48</u>	**5.79**
	Llama2	**6.18**	**8.13**	3.47	<u>5.76</u>

Table 3: Log probability drops on QuoteSum and VERI-GRAN with generator Qwen2 (`Qw.`) and Llama2 (`Ll.`). The best and the second best (except Oracle) entry is marked in bold and underlined, respectively.

Avg both with and without the integration of Dep. To incorporate Dep with HSSAvg, we first expand the target span using Dep, i.e., $\mathbf{t} \leftarrow \bigcup_{r_i \in \mathbf{t}} \mathcal{A}(r_i)$, and then apply HSSAvg to the new span. All other experimental settings remain consistent with those used in the previous experiment.

Results. The results, presented in Table 2, show that while Union underperforms Avg without Dep, it surpasses Avg by a significant margin — at least 9.0 percent points — when Dep is applied.

Insights. The results suggest that Union is most effective when used in conjunction with Dep. Conversely, combining the results of Table 1 and Table 2, we conclude that Dep also works best with Union, as HSSAvgDep performs worse than HSSAvg. This outcome is reasonable since the expanded target spans in HSSAvgDep introduce more context semantics, resulting in averaged hidden states that dilute fine-grained information.

4.3 Faithfulness of fine-grained Attribution

Research Question. We have shown that the proposed method accurately identifies the human-labeled evidence passages, indicating a strong alignment between our approach and human annotations. However, it remains to be verified whether our method is faithful to the generation model — specifically, *does the attributed evidence directly influence the generator to produce the target span?*

Experiment Setting. To quantify the causal effect, we follow the approach of Cohen-Wang et al. (2024), which involves removing the evidence from the prompt, rerunning the generator, and measuring the log predictive probability drop of the target span before and after the removal. First, We use Llama2 or Qwen2 to generate answers with greedy decoding on QuoteSum and VERI-GRAN. Next, in each generated answer, we apply CTI (Qi et al., 2024) to identify context-sensitive tokens, which serve as target spans. We then use the attributor to locate evidence for each target span. Finally, we calculate the log probability drops as follows. Let the documents be $\mathbf{d}_1, ..., \mathbf{d}_C$ with the attributed document $\mathbf{d}_i$. The *log probability drop* is

$$\log \frac{p_{\text{LLM}}(\mathbf{t}|\mathbf{d}_1, ..., \mathbf{d}_C, \mathbf{q}, \mathbf{r}_{\prec \mathbf{t}})}{p_{\text{LLM}}(\mathbf{t}|\mathbf{d}_1, ..., \mathbf{d}_{i-1}, \mathbf{d}_{i+1}, ..., \mathbf{d}_C, \mathbf{q}, \mathbf{r}_{\prec \mathbf{t}})},$$

where p_{LLM} is the output probability of the LLM, $\mathbf{t}$ is the target span, and $\mathbf{r}_{\prec \mathbf{t}}$ is the response prefix preceding $\mathbf{t}$. A larger log probability drop indicates greater faithfulness of the attribution to the generation process.

We include HSSAvg, HSSAvgDep, and CCI as baselines. Additionally, we introduce two extra baselines, Random and Oracle. Random randomly selects an evidence passage, with the experiment repeated three times to calculate the average performance. Oracle, on the other hand, performs a brute-force search to identify the passage that results in the highest log-probability drop, using that passage as the evidence.

Results & Insights. The results in Table 3 show that Dep significantly enhances the faithfulness of AttnUnion. In most cases, AttnUnionDep outperforms baselines except Oracle and closely approaches its performance, indicating that AttnUnionDep offers greater faithfulness than previous methods. Interestingly, all methods maintain their faithfulness even when using different models from the generator, showing their flexibility and independence from the generator's backbone.

4.4 Sentence-level Attribution

Fine-grained attribution can be easily applied to sentence-level attribution, as long as we select sentences as the target spans. In the following, we evaluate the performance of sentence-level attribution of AttnUnion on commonly used benchmarks and compare it with the current SOTA[7].

Benchmarks. Following Gao et al. (2023b), we conduct experiments on datasets ASQA (Stelmakh

[7]We do not use AttnUnionDep because Dep does not change the output evidence of AttnUnion in this experiment.

	ELI5			ASQA		
	R	**P**	**F1**	**R**	**P**	**F1**
			Qwen2			
SelfCitation	31.1	30.1	30.6	57.4	53.3	55.3
+ HSSAvg	24.5	22.4	23.4	35.0	25.9	29.8
+ CCI	35.7	18.9	24.7	59.4	33.6	42.9
+ AttnUnion	39.4	29.3	33.6	65.7	49.0	56.1
AttrFirst	69.3	**69.1**	69.2	63.4	**69.9**	66.5
+ AttnUnion	**81.9**	64.8	**72.3**	**87.2**	60.7	**71.6**
			Llama2			
SelfCitation	20.1	15.3	17.4	55.4	51.9	53.6
+ HSSAvg	18.0	16.1	17.0	33.7	22.6	27.1
+ CCI	26.2†	**29.1**†	**27.6**†	42.3	22.6	29.5
+ AttnUnion	**28.1**	26.9	27.5	**62.1**	**52.6**	**57.0**

Table 4: The citation quality of sentence-level attribution (**R** and **P** are citation recall and precision, respectively). "+X" means using X to attribute the generation of the SelfCitation/AttrFirst with the original citations removed. Cited results are marked by †.

	QuoteSum	VERI-GRAN	ASQA
		Baselines	
CCI	2921.8	7213.7$_{(3)}$	21780.5$_{(3)}$
HSSAvg	84.5	538.5	184.4
AttnUnion♣	141.9	1679.5	790.5
		Our Methods	
AttnUnion	**22.7**	**265.0**	**123.4**
AttnUnionDep	54.0	427.4	-

Table 5: Average Latency per target span (ms) with backbone of Qwen2 7B. By default, we ran this experiment on a single GPU, except for entries that met the OOM problem. The OOM entries were run on multiple GPUs, marked by a subscript indicating the number of GPUs used. The best entry is marked in bold and the second best entry is underlined. The Hugginface implementation of AttnUnion is marked by ♣.

et al., 2022) and ELI5 (Fan et al., 2019). In these datasets, each question is attached with 100 documents, and the top 5 documents are used as the retrieved documents (the vanilla setting of Gao et al. (2023b)). Following ALCE (Gao et al., 2023b), we use metrics MAUVE (Pillutla et al., 2021), EM/claim recall, citation recall, and citation precision. The first two measure the generation quality, and the last two measure the citation quality.

Baselines. The baselines include all previously introduced baselines and two baselines of sentence-level attribution, the vanilla version of self-citation (Gao et al., 2023b) and Attribute First Then Generate (AttrFirst for short, Slobodkin et al., 2024). To rule out the effect of different prompts, following Qi et al. (2024), all fine-grained baselines share the prompt and the generated response of self-citation (with the original citations removed) to attribute. To compare our method with AttrFirst, we separately evaluate our method on the generated results of AttrFirst.

Backbones. The backbones are by default Qwen2 7B and Llama2 7B. However, due to the context length required by AttrFirst exceeds the max context length of Llama2 7B, we only evaluate AttrFirst on Qwen2 7B.

Hyperparameters. For fine-grained methods, because the benchmarks allow outputting multiple citations for a target sentence, AttnUnion, HSSAvg, and CCI output all passages possessing attribution scores above a threshold. The threshold is zero for AttnUnion and CCI, and is 0.55 for HSSAvg. The other hyperparameters of these fine-grained attribution methods are the same as the previous experiments.

For sentence-level methods, self-citation generates with 2-shot, temperature of 1.0, and top-p of 0.95; AttrFirst generates with 1-shot for content selection, 4-shot for fusion in context, temperature of 0.3, max retry number of 5.

Results. We repeat all experiments three times with different seeds (except AttrFirst and CCI due to their heavy computational overload) and take the average results. AttrFirst failed on 355 and 448 instances on ELI5 and ASQA, respectively, and we report the results measured on the successful instances. The generation quality and citation quality results are shown in Table 4 and Table 7, respectively. As the results show, AttnUnion consistently outperforms other fine-grained attributors and improves the citation quality of SelfCitation and AttrFirst, suggesting its application for improving various attributors.

4.5 Attribution Latency

Research Question. From a practical point of view, we ask: *is our method faster than previous works?*

Experiment Setting. We compare our method with all fine-grained attribution baselines and AttnUnion based on Huggingface implementation of calculating attention weights. The experiments are conducted on QuoteSum, VERI-GRAN, and ASQA with the backbone of an NF4-quantized Qwen2 7B and the device of a single NVIDIA RTX 3090 24GB GPU. HSSAvg uses the same early exit as AttnUnion for a fair comparison. We input instances of the datasets[8] one by one to the methods. The latency is measured by the average time consumed per target span.

Results. The results are shown in Table 5. Our implementation of AttnUnion largely outperforms other methods, with an average latency of 265.0 ms on the long-context dataset VERI-GRAN, and AttnUnionDep is the second.

Insights. Considering the early exits of AttnUnion and HSSAvg are the same, AttnUnion should have a similar latency to HSSAvg. However, the latency of AttnUnion is much less than HSSAvg. This is because the AttnUnion does not need to recompute average hidden states for each new target token as HSSAvg; instead, AttnUnion reuses token-wise attribution for all target spans in the same response.

5 Conclusion

This work proposes a novel fine-grained attribution method, leveraging attention weights and dependency parsing. The experiments show that our method is the new SOTA fine-grained attributor and generalizes well to sentence-level attribution. Moreover, our method is much faster than previous works, showing its potential to be applied in real-time attribution systems.

6 Limitations

A limitation of our evaluation is that the target spans of QuoteSum and VERI-GRAN are selected by models, which might not reflect the real application scenario where target spans are selected by human users. This limitation could be resolved by collecting user-selected target spans in the future. Another limitation is that QuoteSum and VERI-GRAN benchmarks focus on attributing verbatim spans from the documents. Although Phukan et al. (2024) observe that LLMs tend to produce verbatim answers, it is interesting to evaluate attribution with more abstractive answers in future work. In addition, in dependency parsing augmentation, the expansion on the dependency parse tree is rule-based (e.g., recognizing coordinating components by relation "conj"), which could be improved by machine learning in the future. Finally, our work only considers attribution in English. Although the attention-based method can be easily adapted to LLMs of other languages, it may take effort (but not difficult, as we show in Appendix H) to adapt our dependency parsing augmentation to other languages since dependency parsing is language-specific.

[8]On ASQA, the response is the generation of self-citation with citations removed.

Acknowledgments

This work has been supported by National Natural Science Foundation of China (No. 62206265, 62076231) and National Key Research and Development Program of China (No.2022YFB2702502).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Adrián Bazaga, Pietro Lio, and Gos Micklem. 2024. Unsupervised pretraining for fact verification by language model distillation. In *The Twelfth International Conference on Learning Representations*.

Jiangjie Chen, Qiaoben Bao, Changzhi Sun, Xinbo Zhang, Jiaze Chen, Hao Zhou, Yanghua Xiao, and Lei Li. 2022. Loren: Logic-regularized reasoning for interpretable fact verification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10482–10491.

Yung-Sung Chuang, Linlu Qiu, Cheng-Yu Hsieh, Ranjay Krishna, Yoon Kim, and James Glass. 2024. Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps. *arXiv preprint arXiv:2407.07071*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. 2024. Contextcite: Attributing model generation to context. *arXiv preprint arXiv:2409.00729*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115.

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.

Shangbin Feng, Vidhisha Balachandran, Yuyang Bai, and Yulia Tsvetkov. 2023. FactKB: Generalizable factuality evaluation using language models enhanced with factual knowledge. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 933–952.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2023a. RARR: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023b. 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.

Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, Maximilian Werk, Nan Wang, and Han Xiao. 2023. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents. *arXiv preprint arXiv:2310.19923*.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

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

Haichuan Hu, Yuhan Sun, and Qunjun Zhang. 2024. Lrp4rag: Detecting hallucinations in retrieval-augmented generation via layer-wise relevance propagation. *arXiv preprint arXiv:2408.15533*.

Qidong Huang, Xiaoyi Dong, Pan Zhang, Bin Wang, Conghui He, Jiaqi Wang, Dahua Lin, Weiming Zhang, and Nenghai Yu. 2024. Opera: Alleviating hallucination in multi-modal large language models via over-trust penalty and retrospection-allocation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13418–13427.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Dongfang Li, Zetian Sun, Xinshuo Hu, Zhenyu Liu, Ziyang Chen, Baotian Hu, Aiguo Wu, and Min Zhang. 2023a. A survey of large language models attribution. *arXiv preprint arXiv:2311.03731*.

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

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351.

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*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.

Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. 2024. Fine-grained hallucination detection and editing for language models. *arXiv preprint arXiv:2401.06855*.

Khalil Mrini, Franck Dernoncourt, Quan Hung Tran, Trung Bui, Walter Chang, and Ndapa Nakashole. 2020. Rethinking self-attention: Towards interpretability in neural parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 731–742.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2024. Generating benchmarks for factuality evaluation of language models. In *Proceedings of the 18th*

Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 49–66.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

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

Anirudh Phukan, Shwetha Somasundaram, Apoorv Saxena, Koustava Goswami, and Balaji Vasan Srinivasan. 2024. Peering into the mind of language models: An approach for attribution in contextual question answering. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11481–11495.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. In *Advances in Neural Information Processing Systems*, volume 34, pages 4816–4828.

Jirui Qi, Gabriele Sarti, Raquel Fernández, and Arianna Bisazza. 2024. Model internals-based answer attribution for trustworthy retrieval-augmented generation. *arXiv preprint arXiv:2406.13663*.

Abhilasha Sancheti, Koustava Goswami, and Balaji Srinivasan. 2024. Post-hoc answer attribution for grounded and trustworthy long document comprehension: Task, insights, and challenges. In *Proceedings of the 13th Joint Conference on Lexical and Computational Semantics (*SEM 2024)*, pages 49–57.

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.

Tal Schuster, Adam Lelkes, Haitian Sun, Jai Gupta, Jonathan Berant, William Cohen, and Donald Metzler. 2024. SEMQA: Semi-extractive multi-source question answering. 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 1363–1381.

Aviv Slobodkin, Eran Hirsch, Arie Cattan, Tal Schuster, and Ido Dagan. 2024. Attribute first, then generate: Locally-attributable grounded text generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3344.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Pranesh Srinivasan, Laichee Man, Kathleen Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael, Smith Ranjan, Subramanian Xiaoqing, Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 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.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, Zhihao Fan, Qwen Team, and Alibaba Group. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Fan Yang, Shiva K. Pentyala, Sina Mohseni, Mengnan Du, Hao Yuan, Rhema Linder, Eric D. Ragan, Shuiwang Ji, and Xia (Ben) Hu. 2019. Xfake: Explainable fake news detector with visualizations. In *The World Wide Web Conference*, WWW '19, page 3600–3604.

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.

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.

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

Xiaofeng Zhang, Yihao Quan, Chaochen Gu, Chen Shen, Xiaosong Yuan, Shaotian Yan, Hao Cheng, Kaijie Wu, and Jieping Ye. 2024b. Seeing clearly by layer two: Enhancing attention heads to alleviate hallucination in lvlms. *arXiv preprint arXiv:2411.09968*.

Algorithm 1 The Basic Algorithm

```
Input: Document tokens $\mathbf{d} = (d_1, ..., d_c)$, question tokens $\mathbf{q} = (q_1, ..., q_m)$, Response tokens $\mathbf{r} = (r_1, ..., r_n)$, the target span $\mathbf{t} \subset \mathbf{r}$, and the model $\mathcal{M}$ that outputs similarity metric.
Output: The evidence tokens of the target span $T$ and their scores.
$\mathbf{S} \leftarrow \mathcal{M}(\mathbf{d} + \mathbf{q} + \mathbf{r}) \in \mathbb{R}^{r \times (c+m)}$
$w \leftarrow$ dictionary()                ▷ The attribution scores
for $r_i \in \mathbf{t}$ do                ▷ Aggregate evidence and scores
    for $j \in$ range($\mathbf{d}$) $\cap$ arg top-$k(\mathbf{S}_i)$ do
        if $j \in w.keys()$ then
            $w[j] \leftarrow w[j] + \mathbf{S}_{ij}$
        else
            $w[j] \leftarrow \mathbf{S}_{ij}$
        end if
    end for
end for
$new\_w \leftarrow$ dictionary()
for $i \in w.keys()$ do                ▷ Remove isolated tokens
    $isolated \leftarrow True$
    for $j$ in $i - \tau$ to $i + \tau$ do
        if $j \neq i$ and $j \in w.keys()$ then
            $isolated \leftarrow False$
            break
        end if
    end for
    if $\neg isolated$ then
        $new\_w[i] \leftarrow w[i]$
    end if
end for
return $new\_w$
```

A Pseudo-code for AttnUnion

The pseudo-code for AttnUnion is Algorithm 1.

B Word-Token Alignment in Dep

With words and tokens distinguished, the atomic-fact recognition method $\mathcal{A}$ should be updated to

$$\mathcal{A}(r_i) \leftarrow \psi \left(\bigcup_{w \in \phi(r_i)} \widetilde{\mathcal{A}}(w)) \right),$$

where $\widetilde{\mathcal{A}}$ is the atomic fact recognition method introduced in Sec. 3.3, ϕ is the token-to-words mapping, and ψ is words-to-tokens mapping. The ϕ

	Target Granularity	Evidence Granularity	#instance	#target per instance	#evidence per instance	Document Len.	Question Len.	Answer Len.
Fine-grained Attribution								
QuoteSum	span	passage	1319	4.6	3.4	1958.1	43.6	275.4
VERI-GRAN	span	sentence	197	1.2	69.1	8211.3	52.9	398.9
Sentence-level Attribution								
ELI5	sentence	passage	1000	6.7	5.0	2906.3	92.3	699.2
ASQA	sentence	passage	948	3.5	5.0	3005.6	47.2	427.0

Table 6: Statistics of datasets used in experiments, where lengths are measured by the numbers of characters.

	ELI5		ASQA	
	MAUVE	Claim Rec.	MAUVE	EM Rec.
Qwen2				
SELFCITATION	23.5	12.4	80.5	39.6
ATTRFIRST	55.0	4.5	55.9	35.2
Llama2				
SELFCITATION	32.9	12.3	56.9	30.1

Table 7: The generation quality results of SELFCITATION and ATTRFIRST.

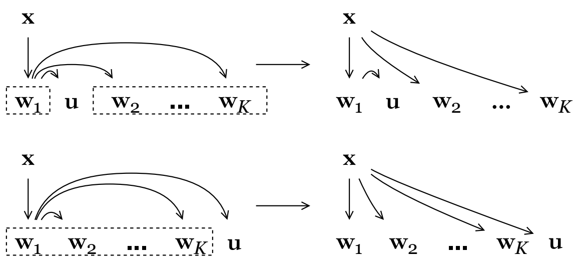


Figure 4: An illustration of reforming the coordinate structures, where the words framed by dash lines are coordinate components.

maps a token to the minimum set of words that cover the token. The ψ maps a set of words to the minimum set of tokens that covers these words.

C Details of Excluding Irrelevant Coordinating Constituents

This section describes how to exclude irrelevant coordinating constituents, given that $\mathcal{A}(r_i)$ has included all successors of v (the closest verb ancestor of r_i). In a dependency parse tree, a *coordinating structure* is multiple words $\mathbf{w}_1, \mathbf{w}_1, ..., \mathbf{w}_K, K > 1$ that satisfy

$$\begin{cases} \text{label}(\mathbf{w}_1) = ... = \text{label}(\mathbf{w}_{K-1}), \\ \text{label}(\mathbf{w}_K) = \text{“conj”}, \\ \text{head}(\mathbf{w}_2) = ... = \text{head}(\mathbf{w}_K) = \mathbf{w}_1, \end{cases} \quad (2)$$

where head($\mathbf{w}_i$) represent the parent of $\mathbf{w}_i$, and label($\mathbf{w}_i$) represent the corresponding relation type between the parent and $\mathbf{w}_i$. Considering that the first component is special, we name the first component as the *leader* of the coordinating structure. The algorithm is outlined as follows.

Input. r_i: the word to augment; v: the closest verb ancestor of r_i; $\mathcal{A}(r_i)$: the atomic fact elements that have collected v and its successors; the heads and labels of all words.

Output. The $\mathcal{A}(r_i)$ with irrelevant coordinate components removed.

1. Identify coordinating structures. The coordinating structures are searched by enumerating the potential leaders (shown as follows).

```
def find_coordinations(
    dep_head,
    dep_label
):
    coordinations = []
    in_coordination_words = set()
    for j in range(len(dep_head)-1):
        if j in in_coordination_words:
            continue
        new_coordination = [j]
        for k in range(j+1, len(dep_head)):
            case1 = get_head(dep_head, k) == j
                and dep_label[k] == dep_label[j]
            case2 = get_head(dep_head, k) == j
                and dep_label[k] == 'conj'
            if case1 or case2:
                new_coordination.append(k)

        if len(new_coordination) > 1:
            coordinations.append(sorted(
                new_coordination))
            in_coordination_words.update(
                new_coordination)

    return coordinations
```

2. Reform the tree. For the convenience of the following process, we temporarily reform the local structures for all coordinate structures, as Fig. 4 shows. We replace the heads of non-leader components with the head of the leader, ending the asymmetric relationship between the leader and the other components. For other children of the leader, we retain its head if it precedes the first non-leader component (as the upper half of Fig. 4 shows); otherwise, we replace its head with the head of the leader (as the lower half of Fig. 4 shows).

3. Identify the path from v to r_i.

4. Process Coordinating Structures that intersect with $v \to r_i$. For coordinating structures that intersect with $v \to r_i$, the tree retains the intersection and removes all other components.

5. Process Coordinating Structures that do not intersect with $v \to r_i$. For non-intersecting coordinating structures $\mathcal{G}$, the algorithm first determines whether there is a parallel coordinating structure of it, where parallel coordinating structures are those that have the same number of constituents, e.g., ("one million dollars," "two million dollars") and ("2012," "2013"). If so, denoting the parallel coordinating structure as $\mathcal{G}'$ and its i-th component is retained, then the algorithm deletes all components of $\mathcal{G}$ except the i-th component from the dependency parse tree.

6. Recollect. The algorithm recollects all v's successors (except punctuation marks) with the new tree, yielding the final $\mathcal{A}(r_i)$.

D Details of Huggingface Implementation of AttnUnion

```python
def forward_stage1(self, batch):
    """stage 1: do not output attention, output
        kv cache on prompt_ids
    """
    outputs = self.model(
        input_ids=batch['prompt_ids'][:,:-1].to(
            self.device), # leave the last prompt
             token to forward in stage2
        attention_mask=batch['prompt_mask'
            ][:,:-1].to(self.device),
        output_attentions=False,
        use_cache=True,
        return_dict=True,
    )
    return outputs.past_key_values

def forward_stage2(self, batch, past_key_values)
    :
    """stage2: output response-to-prompt
        attentions
    """
    attention_mask = torch.cat([batch['
        prompt_mask'][:,:-1], batch['
        response_mask']], dim=1).to(self.device)
    input_ids = torch.cat([batch['prompt_ids'
        ][:,-1:], batch['response_ids'][:,:-1]],
         dim=1).to(self.device)
    outputs = self.model(
        input_ids=input_ids,
        attention_mask=attention_mask,
        past_key_values=past_key_values,
        output_attentions=True,
        use_cache=True,
        return_dict=True,
    )
    return outputs.attentions
```

```python
def forward(self, batch):
    """output response-to-prompt attentions for
        causal LMs
    """
    with torch.no_grad():
        past_key_values = self.forward_stage1(
            batch)
        attentions = self.forward_stage2(batch,
            past_key_values)
        attentions = attentions[self.
            output_attentions_layer]
    return attentions
```

Using Huggingface Transformers to output self-attention weights on the concatenation of the prompt and the response is impractical since this approach will incur tremendous memory overload of $\mathcal{O}((c+m+n)^2)$.

A more memory-efficient approach is to use the response as the query and the concatenation of prompt and response as the key, which incurs a memory overload of $\mathcal{O}(n \times (c+m))$ and requires the KV cache. We choose the latter memory-efficient approach as our baseline in Sec. 4.5, which is illustrated by the `forward` method in the code above.

E Statistics of Datasets

The statistics of datasets are shown in Table 6.

F Experiments on AttnUnionDep's Sensitivity to Hyperparameters

Here, we consider the following hyperparameters:

1. L^*: the layer to extract attention weights;

2. k: how many top-scored prompt tokens are selected as evidence for each response token;

3. τ: the threshold for recognizing isolated tokens.

The experiments are conducted on validation sets of QuoteSum and VERI-GRAN, with the metric of accuracy (%), the attributor of AttnUnionDep, and the models of Qwen2 7B and Llama 7B. The results are shown in Fig. 5, 6, and 7. The performance is relatively stable across a range of hyperparameters. Note that our choice of hyperparameters may not be optimal, because we did not extensively tune them but qualitatively chose them by human evaluation. We found under these hyperparameters, the proposed method provides the most human-friendly (neat and fragmentless) attribution.

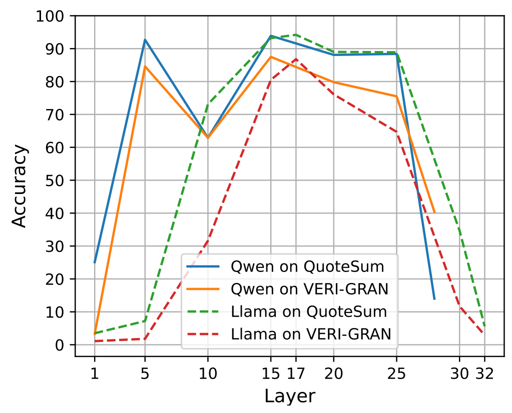


Figure 5: The validation accuracy of AttnUnionDep against the layer from which the attention weights are extracted (fixing $k = 2, \tau = 2$). For Qwen2 7B, $L = 28$, and $\lfloor L/2 \rfloor + 1 = 15$. For Llama2 7B, $L = 32$, , and $\lfloor L/2 \rfloor + 1 = 17$.

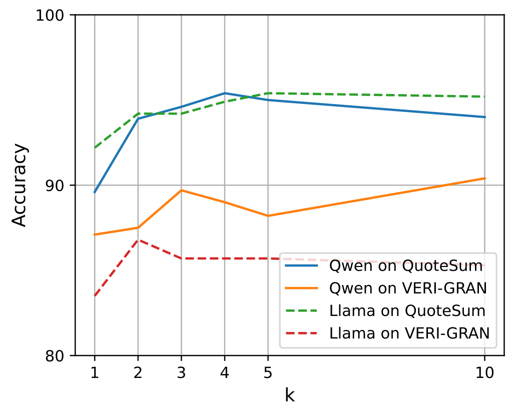


Figure 6: The validation accuracy of AttnUnionDep against k (fixing $\tau = 2$, $L^* = 15$ and 17 for Qwen2 7B and Llama2 7B, respectively).

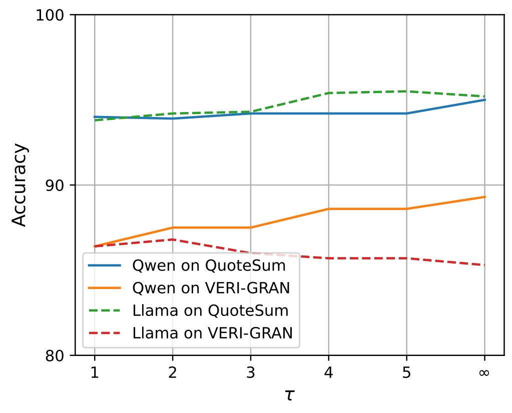


Figure 7: The validation accuracy of AttnUnionDep against τ (fixing $k = 2$, $L^* = 15$ and 17 for Qwen2 7B and Llama2 7B, respectively). Here, $\tau = \infty$ means no filtering out isolated evidence tokens.

G Generation Qualities of SelfCitatoin and AttrFirst

The generation qualities of SelfCitation and AttrFirst are shown in Table 7.

H Evaluating Attribution on Chinese Synthetic Datasets

We conducted experiments on Chinese synthetic datasets to show that our method can be migrated to languages other than English. The synthetic Chinese datasets are constructed by translating the answers from QuoteSum and VERI-GRAN to Chinese and maintaining the questions and the passages in English. Specifically, each answer is processed as follows (during this process, the annotations of the target spans are maintained):

1. segmenting the answer by the boundaries of target spans;
2. separately translating each segment into Chinese;
3. concatenating all translated segments to build the translated answer.

We adapt the dependency parsing augmentation to Chinese without modifying the rules (except for mapping Chinese dependency parsing labels to the English counterpart). We compare HSSAvg, AttnUnion, and AttnUnionDep using the backbone of Qwen2 7B. For all methods, we use the same hyperparameters as in Sec. 4.1. The results are shown in Table 8, which shows that our method AttnUnionDep still outperforms HSSAvg and AttnUnion.

Method	Translated QuoteSum	Translated VERI-GRAN
HSSAvg	76.1	61.6
AttnUnion	76.0	65.3
AttnUnionDep	**87.0**	**72.5**

Table 8: Accuracy (%) on the synthetic Chinese datasets, where the best entries are marked in bold.

I An Additional Alternative of Dep

During the review of this paper, a reviewer proposed an interesting alternative of Dep: considering the self-attention among response tokens, the latter response tokens can be attributed to previous response tokens and this might also reveal some relationship between them (as what Dep does). For instance, “2012” might be attributed to “one million dollars”. Then the attribution of “one million dollars” could be updated with the attribution scores from “2012”.

We implemented the attribution among the response tokens using the same hyperparameters (except τ; no filtering out isolated evidence tokens in this attribution) as the attribution between response and prompt tokens. The results are shown in Table 9. Here we name the alternative AugmentByAttn. We also evaluate a variant of AugmentByAttn that limits the augmentation tokens in the target span's local sentence, to ensure the augmentation tokens are more relevant to the target span. As the results show, AugmentByAttn and its variant improve AttnUnion but are not as good as dependency parsing augmentation, indicating dependency parsing augmentation is more accurate in recognizing semantically relevant tokens to the target span.

	QuoteSum	**VERI-GRAN**
AttnUnion	79.4	70.9
AugmentByAttn	86.0	78.4
AugmentByAttn variant	86.2	77.8
AttnUnionDep	**93.3**	**84.6**

Table 9: Evaluation results for addition alternatives of Dep, i.e., AugmentByAttn and its variant, where the backbone is Qwen2 7B.