Decomposition-Enhanced Training for Post-Hoc Attributions in Language Models

Sriram Balasubramanian[1,2*], **Samyadeep Basu**[1], **Koustava Goswami**[1], **Ryan Rossi**[1], **Varun Manjunatha**[1], **Roshan Santhosh**[3], **Ruiyi Zhang**[1], **Soheil Feizi**[2], **Nedim Lipka**[1]
[1]Adobe Research , [2]University of Maryland, College Park, [3]Adobe Systems
samyadeepb@adobe.com, sriramb@umd.edu

Abstract

Large language models (LLMs) are increasingly used for long-document question answering, where reliable attribution to sources is critical for trust. Existing post-hoc attribution methods work well for extractive QA but struggle in multi-hop, abstractive, and semi-extractive settings, where answers synthesize information across passages. To address these challenges, we argue that post-hoc attribution can be reframed as a *reasoning* problem, where answers are decomposed into constituent units, each tied to specific context. We first show that prompting models to generate such decompositions alongside attributions improves performance. Building on this, we introduce DECOMPTUNE, a post-training method that teaches models to produce answer decompositions as intermediate reasoning steps. We curate a diverse dataset of complex QA tasks, annotated with decompositions by a strong LLM, and post-train Qwen-2.5 (7B and 14B) using a two-stage SFT + GRPO pipeline with task-specific curated rewards. Across extensive experiments and ablations, DECOMPTUNE substantially improves attribution quality, outperforming prior methods and matching or exceeding state-of-the-art frontier models.

1 Introduction

Large Language Models (LLMs) have demonstrated impressive capabilities across a range of natural language tasks, including question answering (Kamalloo et al., 2023), summarization (Liu et al., 2024), and open-ended generation (OpenAI et al., 2024). As their outputs increasingly influence downstream applications in real-world products such as AI Assistants (Mialon et al., 2023; Gabriel et al., 2024), the need for trustworthy and transparent generation becomes critical. One key aspect of this is attribution—the ability to cite supporting evidence from input sources that justifies each part of the model's output. Although attributions can in principle be generated alongside answer generation, doing so often requires significant modifications to existing pipelines that already deliver strong answer-generation performance. To alleviate this problem, recent works (Phukan et al., 2024, 2025; Ramu et al., 2024) generate these attributions in a post-hoc manner i.e., given a document, question and a generated answer, they attribute the generated answer to relevant parts of the document. These post-hoc attribution methods are often based on various forms of semantic search (Robertson and Zaragoza, 2009) or identifying signals from the model internals (e.g., attention-heads or token embeddings) (Hirsch et al., 2025). These approaches perform well on purely extractive QA, but often might underperform in more complex scenarios, particularly in multi-hop, semi-extractive and abstractive QA. In such cases, the surface-level answer often obscures the underlying reasoning steps or the multiple sources that were fused during generation. Without access to these intermediate computations, existing attribution methods frequently retrieve incomplete or irrelevant citations.

We argue that this limitation arises from a fundamental mismatch: post-hoc attribution is treated as a *retrieval* problem (Phukan et al., 2024; Hirsch et al., 2025), when in fact it is often a *reasoning* problem. For instance, a multi-hop question may require the model to infer intermediate facts (e.g., the birth country of a historical figure, followed by the laws of that country), each grounded in distinct pieces of evidence. Similarly, abstractive summaries often interleave paraphrased and synthesized content from several documents. In both settings, identifying appropriate citations necessitates reasoning by decomposing the model's output answer into logical units that can each be grounded in specific source content. In our paper, we first

[**]Research work conducted during internship at Adobe Research

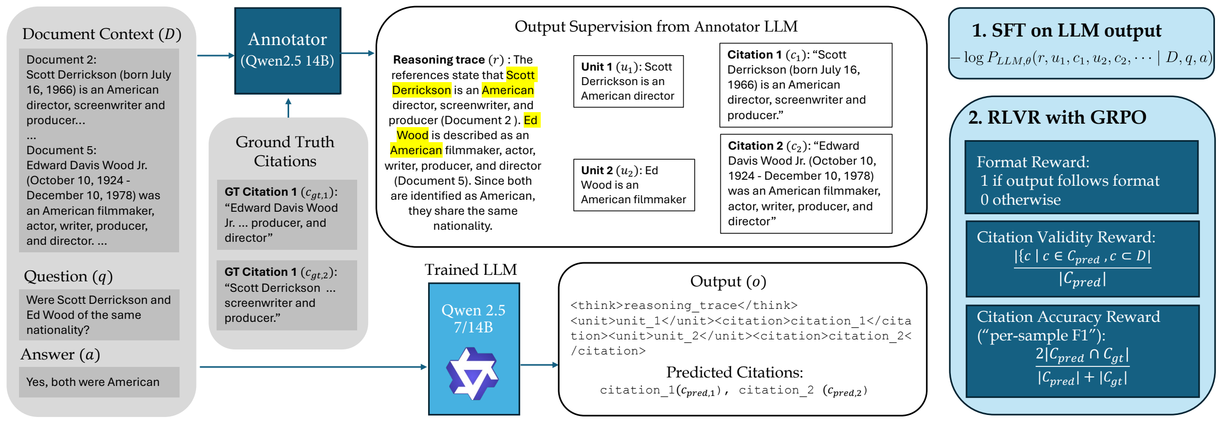
Document Context (D)
Document 2: Scott Derrickson (born July 16, 1966) is an American director, screenwriter and producer...
...
Document 5: Edward Davis Wood Jr. (October 10, 1924 - December 10, 1978) was an American filmmaker, actor, writer, producer, and director. ...
Question (q)
Were Scott Derrickson and Ed Wood of the same nationality?
Answer (a)
Yes, both were American
Annotator (Qwen2.5 14B)
Ground Truth Citations
GT Citation 1 ($c_{gt,1}$): "Edward Davis Wood Jr. ... producer, and director"
GT Citation 1 ($c_{gt,2}$): "Scott Derrickson ... screenwriter and producer."
Output Supervision from Annotator LLM
Reasoning trace (r) : The references state that Scott Derrickson is an American director, screenwriter, and producer (Document 2). Ed Wood is described as an American filmmaker, actor, writer, producer, and director (Document 5). Since both are identified as American, they share the same nationality.
Unit 1 (u_1): Scott Derrickson is an American director
Unit 2 (u_2): Ed Wood is an American filmmaker
Citation 1 (c_1): "Scott Derrickson (born July 16, 1966) is an American director, screenwriter and producer."
Citation 2 (c_2): "Edward Davis Wood Jr. (October 10, 1924 - December 10, 1978) was an American filmmaker, actor, writer, producer, and director"
Trained LLM
Qwen 2.5 7/14B
Output (o)
<think>reasoning_trace</think><unit>unit_1</unit><citation>citation_1</citation><unit>unit_2</unit><citation>citation_2</citation>
Predicted Citations:
citation_1($c_{pred,1}$), citation_2 ($c_{pred,2}$)
1. SFT on LLM output
$-\log P_{LLM,\theta}(r, u_1, c_1, u_2, c_2, \cdots \mid D, q, a)$
2. RLVR with GRPO
Format Reward: 1 if output follows format 0 otherwise
Citation Validity Reward: $\frac{|\{c \mid c \in C_{pred}, c \subset D\}|}{|C_{pred}|}$
Citation Accuracy Reward ("per-sample F1"): $\frac{2|C_{pred} \cap C_{gt}|}{|C_{pred}| + |C_{gt}|}$

Figure 1: **Overview of our method DECOMPTUNE.** We train small- to medium-sized language models to incorporate *answer decomposition as an intermediate reasoning step* prior to generating final attributions. To enable this, we leverage decomposition traces produced by an annotator LLM and adopt a two-stage training strategy: SFT to instill decomposition-aware reasoning, followed by GRPO with task-specific continuous and verifiable rewards to further refine attribution quality.

verify that prompting a strong language model to generate these decompositions as an intermediate reasoning step and then generating the attributions together leads to strong improvements in attribution quality for cases involving multi-hop QA and complex semi-extractive QA setups.

Using this observation, we introduce DECOMPTUNE, a post-training method and framework which trains small to intermediate sized language models to generate post-hoc attributions with a *reasoning-based decomposition step* as an intermediate step, rather than directly generating the attributions. To this end, we first curate a data-mixture encompassing various multi-hop, semi-extractive and abstractive QA where each QA pair is annotated with the intermediate decompositions using an annotator LLM (e.g. Qwen-2.5-14B). We then train Qwen-2.5-7B, and 14B using a two-stage sequential training approach. In the first stage, we train the language model on our data-mixture using SFT. In the second stage, we use GRPO with a set of continuous rewards (including format reward, accuracy reward and retrieval or validity reward). With this two stage training approach, we find that the post-trained Qwen series of models using DECOMPTUNE outperform existing open-source post-hoc attribution methods by a significant margin of at least 17 percentage points and also obtains a performance improvement of 23 points over the base model. Our post-trained models also obtain performance at-par with the best performing frontier models for most datasets and also outperform them in a few cases.

Overall, in summary our paper makes the following contributions:

- We find that generating *answer decompositions as an intermediate reasoning step* improves attribution - all in one forward pass.
- We curate a large-scale dataset mixture annotated with answer decompositions and their corresponding source attributions. This dataset serves as a valuable resource for post-training language models to perform attribution as a structured reasoning task.
- We post-train small to intermediate sized language models using a two stage SFT + GRPO (with continuous rewards) training on our curated dataset, outperforming existing post-hoc attribution methods and obtaining performance at-par with closed-source models.

2 Related Works

Attributing the output of a neural network to its input has been a long-standing problem in NLP (Madsen et al., 2022; Rogers et al., 2020) and computer vision (Kazmierczak et al., 2025; Shrikumar et al., 2019), and has historically been approached from the lens of interpretability. In the pre-LLM era, the tasks that neural networks were capable of solving were small-scale, with input/output size at most a few paragraphs (in NLP) or images (in vision). The primary unit for attribution for these tasks were similarly small, with gradient-based and activation-based interpretability methods attributing the output to tokens or image patches. With the advent of powerful closed-source LLMs (Liu et al., 2025) and new paradigms like multi-doc or long-context QA, several attribution methods (Nakano et al., 2022; Qin et al., 2023; Menick et al., 2022; Batista et al., 2025) utilize the LLM itself for providing citations at a higher level, typically documents or paragraphs. Many of these works aim to prompt or train LLMs produce citations *jointly with the answer* so that all factual sentences can be grounded in some external document. Zhang et al. (2024) finetune open-source LLMs (via SFT) on their curated dataset LongCite which consists of automatically generated long-context QA instances with precise sentence-level citations. Chuang et al. (2025) propose SelfCite which is trained via RL with an ablation-based reward to cite sentences in the context. Companies like (Anthropic, 2025) and (Perplexity, 2025) have also exposed LLM-generated citations API in their respective APIs.

In **post-hoc attribution**, citations are produced after the answer is generated in a separate step. Formally speaking, given a **document** D, **question** q and a **generated answer** a, we seek the **citations** in D that justify and ground the answer a. Prior work has focused on strategies for prompting an LLM to return citations given the context, question and answer (Ramu et al., 2024), interpretability techniques to identify relevant circuits (Basu et al., 2025), internal layers (Phukan et al., 2024; Hirsch et al., 2025; Qi et al., 2024) or information retrieval (IR) based approach (Ramu et al., 2024) in which the sentence most relevant to the answer is retrieved from the documents in the context. However, these methods do not properly handle cases where multiple sources of information are synthesized into one answer part. To the best of our knowledge, we are the first to propose a decomposition based prompting strategy to generate data and train LLMs on this data, outperforming other methods in this domain. Ramu et al. (2024) also employs answer decompositions; however, these are treated as separate, independent steps for decomposition and attribution, rather than as part of an intermediate reasoning process, nor do they train LLMs using this decomposition.

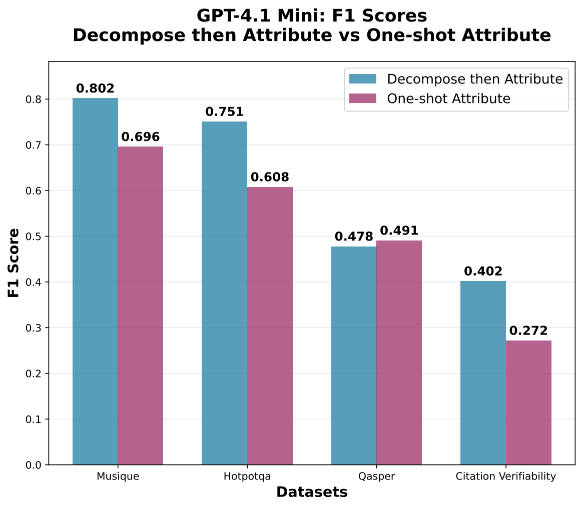

Figure 2: **Decomposition as intermediate step helps in attribution.** Prompting LLMs to decompose an answer as an intermediate step, then attribute, performs better than solely prompting them to only generate attributions, even with chain-of-thought prompting.

3 Initial Observation: Decomposition As an Intermediate Step Improves Attribution

Earlier works (Ramu et al., 2024) use answer decomposition as a pre-processing strategy where the decomposed answer units are then fed to an attributor separately (can be a LLM or a semantic search) for improved attribution. In our work, we prompt the language model to perform the decomposition step as an *intermediate reasoning step* and the attribution generation together. Given a document D, question q and a generated answer a, we prompt an annotator language model to first decompose the answer a into logical sub-units with a reasoning trace (see Fig.(1) for an example of decomposition) and then generate the attributions for each of these sub-units. To improve the quality of the traces, we filter out sentences unrelated to the ground truth citations from the document context. From Fig. 2, we observe that prompting the language model to first generate the intermediate decomposition and then the final attributions yields substantial gains in attribution performance. Here, *both* methods are instructed to think step-by-step and output their reasoning (exact DSPy (Khattab et al., 2024) CoT prompts in Appendix C), which implies that the difference in performance is solely due to decomposition into information units. In multi-hop and semi-extractive settings, relative performance improves by 13% and 12%, respectively. These results confirm that introducing a decomposition step as an intermediate step prior to attribution generation, rather than producing attributions directly, leads to more accurate attributions. In the following sections, we describe the curation of training data and methods to train small-to-medium sized language models for post-hoc attribution using these answer decompositions as intermediate steps.

4 DECOMPTUNE: Training Language Models to Decompose and Attribute

Our method, in brief, consists of curating and processing open-source datasets covering our targeted domains (4.1), and the details of the training algorithms employed for training the models (4.2).

4.1 Training Dataset Curation and Processing

We construct our training dataset covering the (a) **multi-hop extractive** setting in which in answer is fairly short (only a few tokens) but is synthesized from multiple source citations; and (b) **abstractive** or **semi-extractive** settings wherein the answers are longer and may contain multiple distinct parts, each of which have to be disentangled and then attributed back to the context. We choose MuSiQue (Trivedi et al., 2022) and HotPotQA (Yang et al., 2018) as our multi-hop datasets, and QASPER (Dasigi et al., 2021), Verifiability Judgements (Liu et al., 2023) and LongCite (Zhang et al., 2024) for the semi-extractive and abstractive setting. However, not all of these datasets have sufficiently long context or sentence level ground truth attributions. We also require detailed reasoning traces containing the decompositions to finetune our models. We thus process the datasets in three stages:

Step 1 - Context Extension: Certain datasets (MuSiQue and HotpotQA) contain documents that are very short, often limited to just a few sentences or a single paragraph. In these cases, attribution may be too trivial. To create a more challenging benchmark, we generate extended versions of these documents (containing at least 500 words). To perform this extension any strong LLM (open-source or closed-source) can be used. The extended text must (a) include all original sentences verbatim, and (b) avoid restating the information from the original sentences elsewhere in the document. These constraints ensure that the ground-truth citations remain valid for the longer documents.

Step 2 - Intermediate Decompositions: Training our LLMs via supervised fine-tuning (SFT) requires high-quality reasoning traces that break down answers into smaller components, each of which can be attributed to a specific sentence in the source text. To generate these traces, we prompt the annotator LLM to first *decompose* answers into logical sub-parts. We only give the ground truth citations as the input document to the annotator. The annotator LLM is primarily responsible for matching the decomposed unit with the ground-truth citations. Since we have access to the ground truth during this step, we validate whether the citations generated are identical to the ground truth, and discard the rare samples in which they are not. We note that the decomposition step is agnostic to a specific model and any capable strong open-source or closed-source model can carry out this step. In our experiments, we find that using stronger models can improve performance after SFT, but these differences vanish after further finetuning with GRPO. Therefore, we can use relatively weaker annotator LLMs such as Qwen2.5 14B without any significant performance loss.

Step 3 - Attribution Matching: Post intermediate decompositions of the answer sentence, we attribute each decomposed sub-unit to the relevant source sentence. To enhance precision, we provide the model with only the ground-truth citations as its source material during this process. We note that the annotator LLM is primarily responsible for matching the decomposed unit with the ground-truth citations (already present). In particular, the annotator LLM is not responsible for generating the citations.

We provide the prompts used for each of the processing stages in the Appendix (Section A). We obtain a dataset of around 90,000 samples after this processing, each containing the relevant documents D, the question q and full answer a, the answer units $\{u_i\}$, the corresponding ground-truth citations $\{c_i\}$ and LLM-generated reasoning trace containing the decomposition r.

4.2 Training Details

We adopt a two-stage training procedure: first, we fine-tune the LLM via SFT on the intermediate decompositions; then, we further train the SFT-tuned LLM via GRPO with specifically curated rewards for the task of post-hoc attribution. We explain the details of these two stages below:

SFT on reasoning traces: We first use direct supervision and fine-tune the LLM to generate the reasoning trace r, the decomposition units $\{u_i\}$ and the corresponding citations $\{c_i\}$ we obtained from the annotator LLM (Section 4.1). Formally, the LLM is trained to generate r, $\{u_i\}$ and $\{c_i\}$ given documents D, question q, answer a by minimizing the standard cross-entropy loss for next token prediction $\mathbb{E}\left[-\log P_{LLM,\theta}\left(r, u_1, c_1, , \ldots u_n, c_n \mid D, q, a, \right)\right]$, where P_θ represents the conditional likelihood of the data according to the LLM with weights θ. We format the SFT completions containing the reasoning trace, units and citations as a single string o: `<think> reasoning trace </think> <unit> unit 1 </unit> <citation> citation 1 </citation> ... <unit> unit n </unit> <citation> citation n </citation>`

During this stage, the model learns to output its reasoning containing the decomposition and the citations in the required format, greatly reducing the computational effort in the GRPO stage. We find that simply prompting the base models is not very effective due to the relatively poor instruction following capabilities of these models, thus necessitating an SFT stage. The reasoning trace r also serves as a rich, low-noise, signal in the SFT optimization process, thus preparing the model for RL training with only scalar rewards.

RL with continuous verifiable rewards: After a round of SFT, we train the model further via GRPO (Shao et al., 2024) with continuous verifiable rewards. We use a **format** reward, **validity** reward and a **weighted accuracy** reward for each sample. We describe each of them below:

Format Reward: We give a reward of 1 to the model for following the format required for parsing the reasoning, units, and citations from the output; and 0 if the output does not follow the format.

Validity Reward: Also, we give a reward if the citations output by the model are valid, that is, present in the document verbatim. The reward is the fraction of predicted citations (C_{pred}) that is a sub-string of the document context D. Formally, $R_{\text{validity}} = \frac{|\{c|c\in C_{pred}, c\subset D\}|}{|C_{pred}|}$. Here, we use $c \subset D$ to denote that c is a substring of D.

Weighted Accuracy Reward: Most importantly, we incentivize accurate citations by using a "per-sample weighted" F1 score as reward. Given predicted citations C_{pred}, ground-truth citations C_{gt}, and a recall weight α, the accuracy reward is $R_{\text{accuracy}} = \frac{|C_{pred}\cap C_{gt}|}{(1-\alpha)|C_{pred}|+\alpha|C_{gt}|}$. This can be interpreted as the weighted harmonic mean of the precision and recall.

All the above rewards are moderately sparse, and can drop to zero even with minor changes to the citations. We thus shape the rewards by utilizing a string similarity function to give non-zero reward when the citations are close but not exactly correct. Let $S(a, b)$ denote the similarity between two strings a and b as a number between 0 and 1 . Then,

$$R_{\text{format}}(o) = \frac{\max_{s\subset o, s \text{ satisfies format}} |s|}{|o|}$$

$$R_{\text{validity}}(C_{pred}, D) = \frac{\sum_{c\in C_{pred}} \max_{s\subset D, |s|=|c|} S(c, s)}{|C_{pred}|}$$

$$R_{\text{accuracy}}(C_{pred}, C_{gt}) = \frac{\max_M \sum_{(i,j)\in M} S(c_{pred,i}, c_{gt,j})}{(1-\alpha)|C_{pred}| + \alpha|C_{gt}|}$$

where M can be any bipartite matching between C_{pred} and C_{gt}.

In practice, we use fuzzy string matching based on Levenshtein distance for calculating the similarity scores $S(a, b)$, which makes for efficient calculation of all these rewards. We also set $\alpha = 0.75$ rather than the usual F1 score with $\alpha = 0.5$, as this broadly improves recall while maintaining precision (see Section 7 for details).

5 Experiments and Results

Using DECOMPTUNE, we train two open-source language models: (i) Qwen2.5-7B-1M, and (ii) Qwen2.5-14B-1M (Qwen et al., 2025). By training across these representative model sizes, we aim to assess the effectiveness and scalability of DECOMPTUNE under different resource regimes. We now provide a detailed description of the baseline methods as well as the adopted evaluation metrics.

5.1 Baselines

We compare our approach against three broad families of baselines, chosen to represent both strong language models and established post-hoc attribution methods. (i) *Frontier LLMs*. We include GPT-4o, GPT-4.1-mini, Claude-Sonnet 4 and DeepSeek-R1 as representative state-of-the-art frontier models. These models have consistently demonstrated strong performance across a wide range of reasoning and QA tasks, and therefore provide a competitive upper bound for attribution quality in closed settings. We use the decomposition-based prompting strategy we used in Section 3. (ii) *Smaller LLMs*. To establish comparisons with smaller LLMs comparable to , we also evaluate against Qwen2.5-3B, Qwen2.5-7B-1M, Qwen2.5-14B-1M (Qwen et al., 2025) and

	Multi-hop extractive						Abstractive/Semi-extractive								
	MusiQue			HotpotQa			Qasper			Verifiability			Overall		
Methods	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Frontier Models															
GPT-4o	.979	.236	.380	.713	.413	.523	.533	.361	.430	.343	.314	.328	.642	.331	.415
GPT-4.1-mini	.941	.699	**.802**	.724	.781	.751	.531	.435	**.478**	.349	.474	.402	.636	.597	.608
Claude-Sonnet 4	.964	.351	.514	.869	.413	.560	.611	.355	.449	.491	.474	.483	.734	.398	.501
DeepSeek-R1	.963	.428	.592	.809	.481	.603	.478	.304	.372	.418	.411	.414	.667	.406	.495
Ours: 7B	.885	.716	.792	.834	.815	**.825**	.455	.468	.462	.500	.484	**.492**	.669	.621	**.642 (+.034)**
Other Methods															
Qwen2.5-3B *	.746	.215	.334	.486	.315	.382	.174	.143	.157	.119	.204	.150	.381	.219	.256
Qwen2.5-7B *	.869	.393	.542	.587	.482	.529	.282	.324	.302	.209	.334	.257	.487	.383	.407
Qwen2.5-14B *	.934	.376	.537	.695	.520	.595	.415	.354	.382	.364	.387	.375	.602	.410	.472
Llama3.1-8B *	.832	.563	.671	.534	.571	.552	.178	.258	.211	.137	.404	.204	.420	.449	.409
Embeddings	.711	.263	.384	.631	.282	.390	.391	.285	.330	.381	.360	.370	.520	.297	.370
Attention-Head	.753	.282	.410	.612	.297	.400	.425	.291	.345	.393	.362	.377	.545	.308	.394
Ours: 7B	.885	.716	**.792**	.834	.815	**.825**	.455	.468	**.462**	.500	.484	**.492**	.669	.621	**.642 (+.170)**
Ours: 7B															
SFT	.848	.273	.414	.779	.445	.566	.516	.239	.326	.405	.249	.308	.637	.301	.404
SFT + GRPO $_{(\alpha = 0.5)}$	.904	.709	.795	.835	.776	.805	.564	.351	.433	.525	.380	.441	.707	.554	.618
SFT + GRPO	.885	.716	.792	.834	.815	.825	.455	.468	.462	.500	.484	.492	.669	.621	.642 **(+.238)**
Ours: 14B															
SFT	.888	.303	.452	.819	.463	.592	.547	.270	.362	.458	.279	.346	.678	.329	.438
SFT + GRPO	.918	.743	.821	.861	.764	.809	.518	.406	.455	.529	.513	.520	.706	.606	**.650 (+.212)**

Table 1: **Evaluation Results using exact string match with ground-truth attributions.** We find that with exact-string matches to ground-truth attributions as an evaluation - our post-trained models surpasses frontier models and also obtain gains over other post-hoc attribution methods by a strong margin. We find that post-training with our curated GRPO rewards is very crucial in obtaining the best performance. The best performing open-source variant is marked with Blue. Improvement over the best performing method in each section is shown in parenthesis (Note that asterisked methods use a more involved prompting strategy with multiple LLM calls; see Section 5.1 and Appendix C)

Llama 3.1 8B (Grattafiori et al., 2024). This selection allows us to cover models of different parameter scales and training regimes, providing insight into how attribution performance varies across model size and capacity in publicly available systems. For these models, however, directly prompting them to decompose then attribute yields poor results because of inadequate instruction following capabilities. Therefore, we use a more involved strategy — first prompt to decompose into individual units, then in separate LLM calls, prompt to attribute each unit to a sentence in the reference — for much better performance across the board (see Appendix C for exact prompts). However, this involves many more LLM calls per sample, with much larger inference time overhead making it less practical for deployment.

Finally, we evaluate two white-box attribution approaches: (a) Token Embeddings (Phukan et al., 2024), which leverage token-level representation alignment; and (b) AttnAttribute (Basu et al., 2025) which uses attribution heads. For AttnAttribute, we follow the original formulation but adapt it by providing the generated answer as input within the assistant tag, along with the question and document context, and then extract attributions using the attention heatmap from the attribution head.

5.2 Evaluations

We evaluate our trained models on the test partitions of MusiQue (Trivedi et al., 2022), HotPotQA (Yang et al., 2018), Qasper (Dasigi et al., 2021), Verifiability (Liu et al., 2023), as well as on the GovReport (Huang et al., 2021) and arXiv-summarization datasets (Cohan et al., 2018). This collection of benchmarks spans a diverse range of evaluation settings, covering multi-hop reasoning, semi-extractive and abstractive question answering, and long-document summarization. Among these, MusiQue, HotPotQA, Qasper, and Verifiability provide ground-truth citations which enables us to measure attribution quality via an exact string match. We report the **precision**, **recall**, and **F1 score** for these datasets. The precision is the fraction of output citations that match the ground truth, the recall is the fraction of ground truth citations that are returned by the model, and the F1 score is the harmonic mean of the precision and recall.

However, not all benchmarks (GovReport and arxiv-summarization) contain such reference annotations. For consistency, we further adopt an **LLM-as-a-judge framework** to evaluate the at-

	Multi-hop extractive				Abstractive/Semi-extractive				Summarization					
	MusiQue		HotpotQa		Qasper		Verifiability		GovReport		Arxiv-Summary		Overall	
Methods	IS	CS	IS	CS	IS	CS	IS	CS	IS	CS	IS	CS	IS	CS
Frontier Models														
GPT-4o	.432	.347	.571	.545	.736	.723	.691	.620	.825	.660	.733	.440	.665	.556
GPT-4.1-mini	.616	**.616**	.776	**.846**	.715	.756	.819	.812	.812	.698	.733	.597	.745	**.721**
Claude-Sonnet 4	**.679**	.455	.800	.588	**.813**	**.779**	**.932**	**.862**	**.887**	**.780**	**.801**	**.687**	**.819**	.692
DeepSeek-R1	.673	.509	.817	.680	.762	.707	.892	.810	.848	.733	.722	.580	.786	.670
Ours: 7B	.632	.583	**.821**	.839	.666	.768	.887	.839	.699	.716	.563	.583	.712 (-.107)	**.721** (+0)
Other Methods														
Qwen2.5-3B *	.320	.229	.541	.390	.396	.331	.614	.462	.476	.555	.336	.265	.447	.372
Qwen2.5-7B *	.511	.393	.715	.578	.674	.623	.769	.680	.589	.614	.456	.506	.619	.565
Qwen2.5-14B *	.571	.423	.790	.653	**.741**	.671	**.892**	.724	**.758**	.566	**.668**	.425	**.737**	.577
Llama3.1-8B *	.461	.486	.630	.650	.346	.465	.590	.689	.342	**.741**	.307	.286	.446	.553
Embeddings	.531	.341	.789	.531	.685	.651	.812	.689	.610	.620	.468	.524	.649	.559
Attention-Head	.546	.330	.810	.546	.683	.677	.831	.699	.604	.628	.488	.531	.660	.568
Ours: 7B	**.632**	**.583**	**.821**	**.839**	.666	**.768**	.887	**.839**	.699	.716	.563	**.583**	.712 (-.016)	**.721 (+.144)**
Ours:7B														
Only SFT	.468	.344	.731	.596	.580	.521	.709	.545	.717	.390	.570	.237	.629	.439
SFT + GRPO $(\alpha = 0.5)$	.611	.547	.819	.818	.659	.647	.837	.705	.734	.489	.602	.381	.710	.598
SFT + GRPO	.632	.583	.821	.839	.666	.768	.887	.839	.699	.716	.563	.583	.712 (**+.083**)	.721 (**+.273**)
Ours: 14B														
Only SFT	.517	.377	.770	.623	.666	.607	.763	.616	.767	.438	.596	.252	.680	.485
SFT + GRPO	.620	.569	.825	.837	.688	.771	.891	.873	.768	.730	.649	.651	**.736** (**+.056**)	**.738** (**+.253**)

Table 2: **Evaluation Results with LLM-as-a-Judge**. We find that our post-trained small language models closely match frontier models and outperform existing post-hoc attribution methods. Even in this evaluation setup, we find that post-training with our curated GRPO rewards is very crucial in obtaining the best performance. The best performing open-source variant is marked with Blue.

tributions. This approach allows us to evaluate and compare the methods over all benchmarks in a unified manner. Given a question, answer, and a list of output citations, we prompt GPT 4.1 to give scores according to two rubrics: (a) **collective support score (CS)**: how well does the list of citations (taken collectively) support the answer (analogous to recall), and (b) **individual support score (IS)**: does each individual citation support any part of the answer to the question (analogous to precision). We report the average CS and IS over the entire dataset after normalizing the scores between 0 to 1. The exact prompt template employed in our LLM-as-a-judge evaluations is in the appendix (see Appendix B). While a few methods (such as AttnAttribute) always output citations that are single sentences present in the document, LLM-powered methods may not always obey this condition. In such cases, we may encounter invalid or multi-sentence citations. Thus, we first filter out the citations that cannot be found in the context, and then break each of them into individual sentences if necessary. As these steps can be feasibly performed during inference time, we may view them as part of the methods themselves.

6 Main Results

6.1 Improvements Over Strong Baselines

We find that our proposed approach, DECOMPTUNE, yields substantial improvements over a wide range of baseline models. The SFT stage alone improves instruction following to the extent that attribution becomes possible in a single LLM call, with performance close to the more involved prompting strategy. In particular, our two-stage training framework, which combines supervised fine-tuning (SFT) with GRPO using our task-specific curated rewards, consistently enhances attribution quality across different evaluation settings. Specifically, we observe improvements in F1 score of upto 30 percentage points for multi-hop QA attribution tasks, 23 points for semi-extractive and abstractive attribution tasks, and upto 11 points of improvement in IS (and 23 points in CS) for summarization attribution tasks over the original models. These gains are especially notable for models in the 7B parameter range, where resource constraints typically limit performance. Overall, our findings demonstrate that decomposition-enhanced training provides a reliable and generalizable recipe for strengthening attribution in small- to medium-scale models. This highlights the potential of carefully designed training strategies to unlock grounding capabilities in open-source language models. We

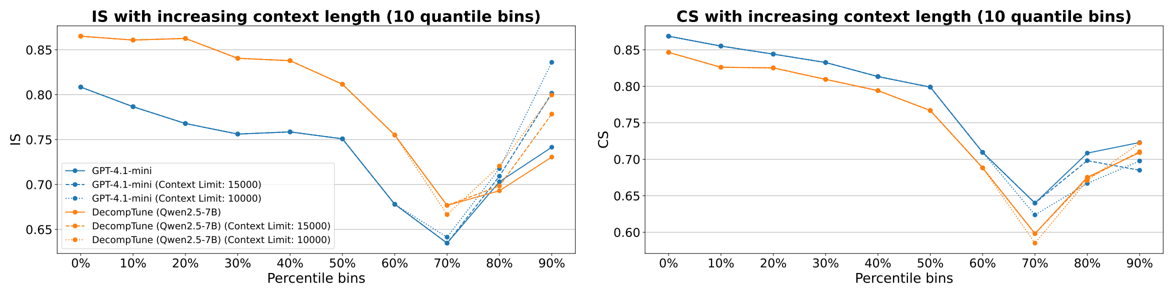

Figure 3: **Effect of Context-Length on Post-Hoc Attribution**. IS and CS score on samples from all datasets of different lengths, grouped into 10 equal quantile bins. DECOMPTUNE (7B) outperforms GPT 4.1 mini on IS, and is closely matched on CS. When coupled with BM25 based search to reduce context, all models gain significantly on IS, but CS for GPT-4.1 drops while DECOMPTUNE improves.

also benchmark against Token-Embeddings and AttnAttribute, both of which are strong baselines that leverage white-box access to language model internals. These methods represent a more challenging comparison point, as they are specifically designed to exploit internal model signals for attribution. Nevertheless, we find that our trained models outperform both by a significant margin of at least 28 points, highlighting the robustness of decomposition-enhanced training.

6.2 Comparisons with Frontier Models

We also compare the performance of our trained models against strong frontier models. In particular, following our initial observations in Fig. 1, we prompt the frontier models to first perform answer decomposition as an intermediate reasoning step before generating the final attributions. To ensure a fair comparison, we apply the same prompting strategy to both our trained models and the frontier systems. As shown in Table 1 and Table 2, our trained models achieve performance that is on par with, and in some cases even exceeds, that of the frontier baselines. These results demonstrate not only the effectiveness of decomposition-enhanced training, but also its potential to narrow the gap between open-source and frontier models in the task of post-hoc attribution. Overall, our results show that our training strategy leads to strong attribution performance with 7B and 14B sized models, outperforms existing post-hoc attribution methods and obtains performance at-par with frontier models, providing an efficient alternative.

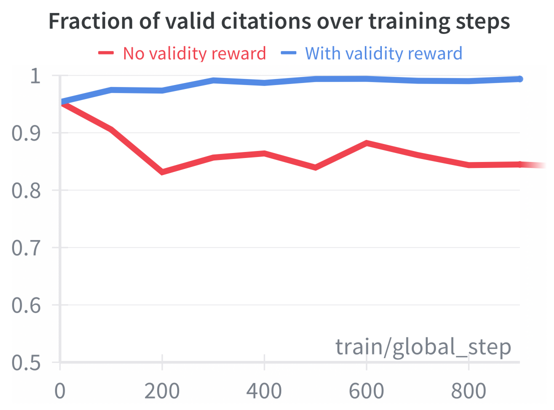

Figure 4: **Validity reward** ensures that the output citations are present verbatim in the context.

Main Takeaway: For post-hoc attribution, SFT alone yields *instruction following improvements*, whereas applying GRPO with task-specific curated rewards yields *more substantial gains* for attribution.

7 Ablations and Further Analysis

Ablation on rewards: We ablate the validity and recall-weighted accuracy reward to show their utility in performance of the model. In Figure 4, we plot the fraction of citations that are *valid* (present verbatim in the document) as the RL training progresses. While initially high, the fraction quickly drops in the absence of a validity reward. With such a reward, however, the fraction of valid sentences is approximately 1. We also show the overall performance (IS, CS, precision, recall, and F1) of models trained with standard F1 score ($\alpha = 0.5$) and trained with recall weighted F1 score ($\alpha = 0.75$) in Table 1 and Table 2. The recall and collective support scores, which measure the extent to which the answer is supported by the citations, improve dramatically, while maintaining precision and IS.

Error analysis on input context length: We analyze model performance across varying context lengths to assess how accuracy depends on reference size. Since very long documents exceed the context window, we also test whether performance holds when reducing context using a lightweight BM25-based sentence selection. Figure 3 shows IS and CS for GPT-4.1-mini and DECOMPTUNE (7B) across 10 context-length quantiles, as well as when contexts are capped at 10k and 15k characters. DECOMPTUNE (7B) achieves notably higher IS than GPT-4.1-mini on short contexts and matches it on the top quantiles, though it slightly lags in CS across all bins. With BM25-based search, all models show strong IS gains on long contexts; however, GPT-4.1-mini's CS declines, while DECOMPTUNE maintains or improves it. We thus believe LLM-based attribution methods can scale well to longer contexts when coupled with search.

8 Conclusion

In this paper, we introduced DECOMPTUNE, to train small models for post-hoc attribution. Instead of generating attributions directly from answers, DECOMPTUNE guides models to decompose answers into constituent units that can be more reliably aligned with supporting evidence. Our training pipeline combines SFT for instruction-following followed by GRPO with task-specific rewards. Across diverse benchmarks, DECOMPTUNE consistently outperforms existing methods and matches/surpasses frontier model performance.

9 Limitations

While our work introduces a new approach to attribution in post-training language models, it does not pose direct ethical risks. On the contrary, attribution methods have the potential to mitigate harms by grounding model outputs in identifiable sources and making the generative process more interpretable. By connecting generated content back to inputs, our approach can support accountability, reduce misinformation, and provide users with greater transparency into model behavior. Nonetheless, ethical challenges in this area remain broader than our scope: attribution alone cannot fully address issues such as dataset bias, privacy, or intellectual property. We emphasize, however, that our contribution is directed toward enhancing interpretability and grounding, and therefore serves as a positive step toward more trustworthy language model deployment.

Acknowledgements

Soheil Feizi and Sriram Balasubramanian were supported in part by a grant from an NSF CAREER AWARD 1942230, the ONR PECASE grant N00014-25-1-2378, ARO's Early Career Program Award 310902-00001, Army Grant No. W911NF2120076, the NSF award CCF2212458, NSF Award No. 2229885 (NSF Institute for Trustworthy AI in Law and Society, TRAILS), and a MURI grant 14262683, DARPA AIQ grant HR00112590066.

References

Anthropic. 2025. Introducing citations on the anthropic api. *Anthropic News*.

Samyadeep Basu, Vlad I Morariu, Ryan A. Rossi, Nanxuan Zhao, Zichao Wang, Soheil Feizi, and Varun Manjunatha. 2025. On mechanistic circuits for extractive question-answering. In *Second Conference on Language Modeling*.

João Eduardo Batista, Emil Vatai, and Mohamed Wahib. 2025. Think before you attribute: Improving the performance of llms attribution systems. *arXiv preprint arXiv:2505.12621*.

Yung-Sung Chuang, Benjamin Cohen-Wang, Zejiang Shen, Zhaofeng Wu, Hu Xu, Xi Victoria Lin, James R. Glass, Shang-Wen Li, and Wen tau Yih. 2025. Selfcite: Self-supervised alignment for context attribution in large language models. In *Forty-second International Conference on Machine Learning*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

Iason Gabriel, Arianna Manzini, Geoff Keeling, Lisa Anne Hendricks, Verena Rieser, Hasan Iqbal, Nenad Tomašev, Ira Ktena, Zachary Kenton, Mikel Rodriguez, Seliem El-Sayed, Sasha Brown, Canfer Akbulut, Andrew Trask, Edward Hughes, A. Stevie Bergman, Renee Shelby, Nahema Marchal, Conor Griffin, and 38 others. 2024. The ethics of advanced ai assistants. *Preprint*, arXiv:2404.16244.

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

Eran Hirsch, Aviv Slobodkin, David Wan, Elias Stengel-Eskin, Mohit Bansal, and Ido Dagan. 2025. Laquer: Localized attribution queries in content-grounded generation. *Preprint*, arXiv:2506.01187.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5591–5606, Toronto, Canada. Association for Computational Linguistics.

Rémi Kazmierczak, Eloïse Berthier, Goran Frehse, and Gianni Franchi. 2025. Explainability for vision foundation models: A survey. *Preprint*, arXiv:2501.12203.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2024. Dspy: Compiling declarative language model calls into self-improving pipelines. In *The Twelfth International Conference on Learning Representations*.

Jiaheng Liu, Dawei Zhu, Zhiqi Bai, Yancheng He, Huanxuan Liao, Haoran Que, Zekun Wang, Chenchen Zhang, Ge Zhang, Jiebin Zhang, Yuanxing Zhang, Zhuo Chen, Hangyu Guo, Shilong Li, Ziqiang Liu, Yong Shan, Yifan Song, Jiayi Tian, Wenhao Wu, and 18 others. 2025. A comprehensive survey on long context language modeling. *Preprint*, arXiv:2503.17407.

Nelson Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7001–7025, Singapore. Association for Computational Linguistics.

Yixin Liu, Kejian Shi, Katherine S He, Longtian Ye, Alexander R. Fabbri, Pengfei Liu, Dragomir Radev, and Arman Cohan. 2024. On learning to summarize with large language models as references. *Preprint*, arXiv:2305.14239.

Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-hoc interpretability for neural nlp: A survey. *ACM Computing Surveys*, 55(8):1–42.

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. *Preprint*, arXiv:2203.11147.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. *Preprint*, arXiv:2311.12983.

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. 2022. Webgpt: Browser-assisted question-answering with human feedback. *Preprint*, arXiv:2112.09332.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Perplexity. 2025. Getting started — overview. `https://docs.perplexity.ai/getting-started/overview`. Developer documentation, accessed September 1, 2025.

Anirudh Phukan, Divyansh, Harshit Kumar Morj, Vaishnavi, Apoorv Saxena, and Koustava Goswami. 2025. Beyond logit lens: Contextual embeddings for robust hallucination detection & grounding in vlms. *Preprint*, arXiv:2411.19187.

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. *Preprint*, arXiv:2405.17980.

Jirui Qi, Gabriele Sarti, Raquel Fernández, and Arianna Bisazza. 2024. Model internals-based answer attribution for trustworthy retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, page 6037–6053. Association for Computational Linguistics.

Yujia Qin, Zihan Cai, Dian Jin, Lan Yan, Shihao Liang, Kunlun Zhu, Yankai Lin, Xu Han, Ning Ding, Huadong Wang, Ruobing Xie, Fanchao Qi, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2023. WebCPM: Interactive web search for Chinese long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8968–8988, Toronto, Canada. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Pritika Ramu, Koustava Goswami, Apoorv Saxena, and Balaji Vasan Srinivasan. 2024. Enhancing post-hoc attributions in long document comprehension via coarse grained answer decomposition. *Preprint*, arXiv:2409.17073.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Preprint*, arXiv:2002.12327.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2019. Learning important features through propagating activation differences. *Preprint*, arXiv:1704.02685.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. 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 (EMNLP)*, pages 2369–2380.

Jiajie Zhang, Yushi Bai, Xin Lv, Wanjun Gu, Danqing Liu, Minhao Zou, Shulin Cao, Lei Hou, Yuxiao Dong, Ling Feng, and Juanzi Li. 2024. Longcite: Enabling llms to generate fine-grained citations in long-context qa. *arXiv preprint arXiv:2409.02897*.

A Prompts For Dataset Construction

Prompt to extend context

Seamlessly extend this passage by around `{num_words}` words:

`{passage}`

Instructions:

1. In your output, you must include ALL the sentences in the original passage.
2. Instead of repeating the sentence word-to-word, just insert the corresponding tag instead. For example, the output can be "This is a new sentence. `<s0>` This is another new sentence. `<s1>` And yet another one."
3. The original sentences should be scattered throughout the output, they should not appear in one place.
4. VERY IMPORTANT: The information in the original sentences (in the `<si>` tags) should not be repeated in the scattered sentences, they should appear only once. In other words, you should NOT be able to infer the information in the original sentences by reading the rest of the document.
5. Ensure that ALL tags `<s0>`, `<s1>`, ..., `<sn>` are present in the output, corresponding to the original sentences.
6. Your output should JUST be the extended passage, with no additional text or formatting.
7. Again, ensure that you DO NOT GENERATE any sentences that contain the information in the original sentences! Simply use the tags `<s0>`, `<s1>`, ..., `<sn>` to refer to the original sentences.

Prompt to generate sentence level attributions for Musique

You are an expert at extracting sentence attributions from a context based on a question and answer. You are given the following:

Context: `{context}`

Question: `{question}`

Answer: `{answer}`

Your task is to extract the EXACT sentences from the context that are used for producing the answer to the question. Each sentence is separated by a newline in the context.
You should return the sentences separated by newlines, with no additional text or formatting, in the order they appear in the context.
To derive the answer from the question, you will need at least one sentence from each paragraph in the context.
Do not include any sentences that are not necessary for answering the question.
VERY IMPORTANT: You should include at least one sentence from each paragraph in the context.

Prompt to generate decompositions

You will be given a set of input references, a question, an `accumulated_text`, and a `latest_text_chunk`. You need to break down the `latest_text_chunk` for a given question into information units. Give only those information units that are attributable to the documents.

Instruction on what good information units are:

1. Give information units that are relevant to the answer.
2. Information units should be meaningful.
3. All information units should be necessary for the answer to derived from the question and documents.
4. The list of information units should be complete, the answer should be derivable solely from the listed information units.
5. Each information unit should be attributable to a SINGLE sentence in the references. It should be worded in a form such that it can be easily searched in the references.
6. Some extra information may need to be added to the information units so that they can be easily attributed to the references. For example, references may contain the sentences "The Great Gatsby is a famous 1925 novel written by F. Scott Fitzgerald" and "The Great Gatsby was narrated by Nick Carraway.", but the question and answer may not have the name of the novel (example: "Who wrote the famous 1925 novel narrated by Nick Carraway? F. Scott Fitzgerald."). In this case, the information units should contain the name of the novel too ("The Great Gatsby") so that it can be attributed easily to the references.
7. The decomposition may be trivial, that is, the `latest_text_chunk` may already be a well-defined information unit.

Instruction on what bad information units are:

1. Information units that convey duplicate information.
2. Information units that are non statements.
3. Information units that are not meaningful to the question.
4. Information units that are not attributable to the references.
5. Information units that are not atomic, i.e, combine information from multiple sentences in the references.

B Judge LLM Prompt

Prompt to score output citations

You are an expert judge. Given a question, a part of the answer, and a list of citations, determine whether the citations SUPPORT the answer. Output consists of two parts, collective support score and individual support score.
Collective support score is defined as follows for the entire citation list:

1. If the citations FULLY support the answer part, then 2.
2. If the citations only PARTIALLY support the answer part, then 1.
3. If the citations do NOT support the answer part at all, then 0.

Individual support score is defined as follows for each citation: If the citation supports (fully or partially) the answer part, then 1. If the citation does NOT support the answer part at all, then 0.

Examples:
Question: What is the capital of the largest country in the world (by area)?
Answer part: The capital of the largest country, Russia, is Moscow.
Citation list: ["Moscow is the capital of Russia.", "Russia is the largest country in the world by area."]
Output: "collective_support_score": 2, "individual_support_scores": [1, 1]
Explanation: The citation list fully supports the answer part because together they confirm that Moscow is the capital of Russia and that Russia is the largest country by area. Individually, both citations support the answer part.
If citation list is: ["Moscow is the capital of Russia."]
Output: "collective_support_score": 1, "individual_support_scores": [1]
Explanation: The citation list partially supports the answer part because while it confirms that Moscow is the capital of Russia, it does not provide information about Russia being the largest country by area. Individually, the single citation supports the answer part.
If citations are: ["The capital of France is Paris.", "The largest country by area is Russia.", "Moscow is the capital of Russia."]
Output: "collective_support_score": 2, "individual_support_scores": [0, 1, 1]
Explanation: The citation list fully supports the answer part because two of the citations confirm that Moscow is the capital of Russia and that Russia is the largest country by area. Individually, the first citation does not support the answer part, while the other two do.

Now evaluate the following:

Question: `{question}`

Answer part: `{answer_part}`

Citation list (this is what you are evaluating): `{citation_list}`

Instructions:

1. Ensure that you do not use any information or knowledge outside of the snippet when evaluating.
2. If the citation list is empty, `collective_support_score` should be 2 if there is no factual information in the answer part, else 0. `individual_support_scores` should be an empty list.
3. Your output should be in the following JSON format without any additional text, no backticks or code blocks.

```
{ "collective_support_score": 0, 1, or 2, "individual_support_scores":
[list of 0s and 1s corresponding to each citation]}
```

C Prompts For Evaluation

DSPY Prompt to decompose, then attribute in the same call (with CoT)

You will be given a set of input references, a question, an `accumulated_text`, and a `latest_text_chunk`. You need to break down the `latest_text_chunk` for a given question into information units. Then for each unit, find ONE citation from the references that supports it. Each citation should be verbatim from the references. You should output a dictionary where each key is an information unit and the value is a citation from the references that supports it.
Instruction on what good information units are:

1. Give information units that are relevant and meaningful to the answer.
2. All information units should be necessary for the answer to derived from the question and documents.
3. The list of information units should be complete, the answer should be derivable solely from the listed information units.
4. Each information unit should be attributable to a SINGLE sentence in the references. It should be worded in a form such that it can be easily searched in the references.
5. Some extra information may need to be added to the information units so that they can be easily attributed to the references. For example, references may contain the sentences "The Great Gatsby is a famous 1925 novel written by F. Scott Fitzgerald" and "The Great Gatsby was narrated by Nick Carraway.", but the question and answer may not have the name of the novel (example: "Who wrote the famous 1925 novel narrated by Nick Carraway? F. Scott Fitzgerald."). In this case, the information units should contain the name of the novel too ("The Great Gatsby") so that it can be attributed easily to the references.
6. The decomposition may be trivial, that is, the `latest_text_chunk` may already be a well-defined information unit.

Instruction on what bad information units are:

1. Information units that convey duplicate information. 2. Information units that are non statements. 3. Information units that are not meaningful to the question. 4. Information units that are not attributable to the references. 5. Information units that are not atomic, i.e, combine information from multiple sentences in the references.

Format instructions:

- Remember to add ## at the end of each field name in the output, the format of '## `<field_name>` ##' is IMPORTANT.
- The name of each field in the output should EXACTLY match the name in the instructions.

DSPY Prompt to only decompose into units (with CoT)

You will be given a set of input references, a question, an `accumulated_text`, and a `latest_text_chunk`. You need to break down the `latest_text_chunk` for a given question into information units. Give only those information units that are attributable to the documents.
Instruction on what good information units are:

1. Give information units that are relevant and meaningful to the answer.
2. All information units should be necessary for the answer to derived from the question and documents.
3. The list of information units should be complete, the answer should be derivable solely from the listed information units.
4. Each information unit should be attributable to a SINGLE sentence in the references. It should be worded in a form such that it can be easily searched in the references.
5. Some extra information may need to be added to the information units so that they can be easily attributed to the references. For example, references may contain the sentences "The Great Gatsby is a famous 1925 novel written by F. Scott Fitzgerald" and "The Great Gatsby was narrated by Nick Carraway.", but the question and answer may not have the name of the novel (example: "Who wrote the famous 1925 novel narrated by Nick Carraway? F. Scott Fitzgerald."). In this case, the information units should contain the name of the novel too ("The Great Gatsby") so that it can be attributed easily to the references.
6. The decomposition may be trivial, that is, the `latest_text_chunk` may already be a well-defined information unit.

Instruction on what bad information units are:

1. Information units that convey duplicate information. 2. Information units that are non statements. 3. Information units that are not meaningful to the question. 4. Information units that are not attributable to the references. 5. Information units that are not atomic, i.e, combine information from multiple sentences in the references.

Format instructions:

- Remember to add ## at the end of each field name in the output, the format of '## `<field_name>` ##' is IMPORTANT.
- The name of each field in the output should EXACTLY match the name in the instructions.

DSPY Prompt to attribute each unit

You will be given a set of input references and a query sentence. Your job is to find the sentence in the references that supports the query most closely and output it verbatim.

Follow these rules:

1. The output sentence should be only a SINGLE SENTENCE from the references and should match EXACTLY.
2. DO NOT output multiple sentences.

Prompt used for training in DECOMP-TUNE

You need to produce attributions to ground an answer to a question based on relevant documents. The inputs are:

Question
`{question}`

Relevant documents
`{documents}`

Answer generated till now
`{answer_till_now}`

Answer part
`{answer_part}`

You need to ground the answer part to the relevant documents by first decomposing the answer part into reasoning, constituent units and citations.

DSPY Prompt to directly attribute (with CoT)

You will be given a set of input references, a question, an `accumulated_text`, and a `latest_text_chunk`. You need to output all the citations that support the `latest_text_chunk`. Each citation should be verbatim from the references. Each citation should be necessary to support the `latest_text_chunk`. All citations collectively should be sufficient to support the `latest_text_chunk`.

Format instructions:

- Remember to add ## at the end of each field name in the output, the format of '## `<field_name>` ##' is IMPORTANT.
- The name of each field in the output should EXACTLY match the name in the instructions.

D Hyperparameters for Training

We describe the hyperparameters for the SFT and GRPO training stages below:

SFT: For all models, we use a batch size of 64 packed sequences of length 32768 tokens and learning rate is 4×10^{-5} with Adam optimizer. We select the best model based on the validation loss on a held out validation dataset.

GRPO: For all models, we use a batch size of 288 with group size 8. The maximum prompt length was 16384 with maximum completion length as 2048 tokens. All rewards were weighted equally, and the recall weight (α) was 0.75. We select the best model based on the validation accuracy reward on a held out validation dataset. We use the corrected GRPO algorithm with no length or difficulty bias. Rollouts were generated using the vLLM engine.

All experiments were performed using at most 2 nodes of 8 A100 GPUs.

E Hyperparameters for AttnAttribute and Token-Embeddings

For adapting the AttnAttribute method from (Basu et al., 2025), we use the Qwen-2.5-7B-Instruct model with the attention layer, head of [15,30] which we find to be the attribution head. We use the same algorithm for span selection as in (Basu et al., 2025). For Token-Embeddings from (Phukan et al., 2024), we use the same layer of 15th in Qwen-2.5-7B-Instruct, but use the distance between the hidden representations to select the relevant span above a high-threshold (we use 0.7 in our experiments).