ATTRIBOT: A BAG OF TRICKS FOR EFFICIENTLY APPROXIMATING LEAVE-ONE-OUT CONTEXT ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

The influence of contextual input on the behavior of large language models (LLMs) has prompted the development of *context attribution* methods that aim to quantify each context span's effect on an LLM's generations. The leave-one-out (LOO) error, which measures the change in the likelihood of the LLM's response when a given span of the context is removed, provides a principled way to perform context attribution, but can be prohibitively expensive to compute for large models. In this work, we introduce AttriBoT, a series of novel techniques for efficiently computing an approximation of the LOO error for context attribution. Specifically, AttriBoT uses cached activations to avoid redundant operations, performs hierarchical attribution to reduce computation, and emulates the behavior of large target models with smaller proxy models. Taken together, AttriBoT can provide a $>300\times$ speedup while remaining more faithful to a target model's LOO error than prior context attribution methods. This stark increase in performance makes computing context attributions for a given response $30\times$ faster than generating the response itself, empowering real-world applications that require computing attributions at scale. We release a user-friendly and efficient implementation of AttriBoT to enable efficient LLM interpretability as well as encourage future development of efficient context attribution methods.

1 INTRODUCTION

The use of large language models (LLMs) has proliferated in recent years including the integration of OpenAI's GPT-4 (OpenAI, 2024) and Google's Gemini (Team, 2024) into Apple and Android-based products with billions of users. As LLMs become more widely used, their influence on information access, decision-making, and social interactions will grow, as will with the consequences of incorrect or problematic outputs. The risks and impact of this widespread adoption spur the need for a deeper understanding of *how* and *why* LLMs generate their outputs. Indeed, a great deal of recent work on LLM interpretability aims to uncover and elucidate their inner workings, including determining the influence of pre-training data (Koh & Liang, 2020; Grosse et al., 2023; Kandpal et al., 2023) and mechanistically understanding their underlying architecture (Cammarata et al., 2020).

A common usage pattern for LLMs involves providing relevant contextual information alongside a query. For example, in retrieval-augmented generation (RAG) (Lewis et al., 2021), documents from an external datastore that are relevant to a given query are retrieved and are provided as part of the LLM's input. This allows LLMs to process data or make decisions based on information that is not available in their pre-training dataset, which has proven critical to making LLMs applicable to a wide range of use cases. While inspecting the documents retrieved by a RAG system can provide a form of interpretability, LLMs generally provide no direct insight into which part of the augmented context influenced the model's generation. To address this shortcoming, *context attribution* methods (Cohen-Wang et al., 2024; Yin & Neubig, 2022; Gao et al., 2023b) aim to quantify the influence of each span of text in an LLM's context on its generated output.

A natural approach for context attribution is to remove a span of text from the context and measure the ensuing change in the likelihood of the model's original response. This notion of importance, known as the Leave-One-Out (LOO) error, is a common idea used in training data attribution (Koh

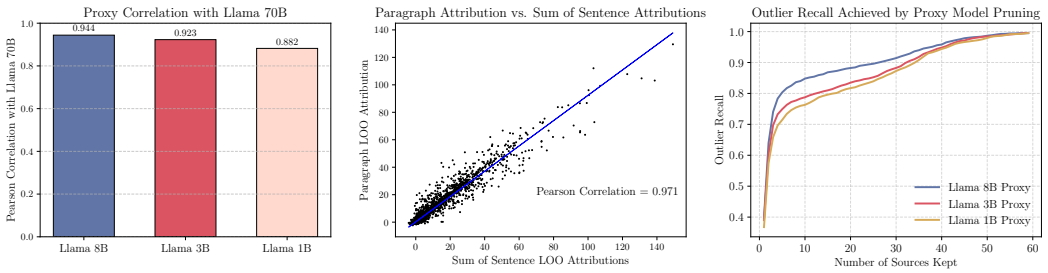

Figure 1: We empirically test the assumptions underlying the AttriBoT's underlying methods on examples from Hotpot QA. **Left:** The attribution scores of small proxy models ranging from 1B to 8B parameters have high correlation with the attribution scores of a 70B-parameter target model, implying that the attributions from smaller models can be a reliable proxy for those from a target model. **Middle:** Paragraph-level attribution scores correlate extremely well ($R = 0.97$) with the sum of the sentence-level attribution scores in a given paragraph, suggesting that hierarchical attribution can provide an effective means of pruning a large amount of irrelevant context. **Right:** Proxy models can effectively prune contexts of unnecessary sources, achieving recall of 90% when keeping only half of the sources in a context.

& Liang, 2020), data valuation (Choe et al., 2024), feature attribution (Li et al., 2017), and recently has been explored for context attribution (Cohen-Wang et al., 2024). While the LOO error produces meaningful and interpretable context attribution scores, it is often viewed as impractical due to the need to perform an independent forward pass to score each text span in the context. This is particularly problematic for modern LLMs whose forward passes are computationally expensive and realistic use cases often involve long contexts with many text spans (e.g., sentences or paragraphs) to score.

In this work, we aim to show that LOO context attributions can be efficiently approximated at LLM scale. To do so, we leverage the following observations:

1. Approximately half of the FLOPs needed to naïvely compute LOO attributions are redundant and can be avoided by caching the attention key and value tensors at each layer (Pope et al., 2022).
2. The sum of the LOO attributions for k contiguous text spans (e.g., sentences in a paragraph or paragraphs in a section) are well-approximated by a single Leave-k-Out attribution score (Figure 1, middle).
3. The LOO attribution scores for a large model (e.g., 70B parameters) are well-approximated by smaller models (e.g., 8B parameters) in the same model family (Figure 1, left and right).

These findings naturally lend themselves to efficient key-value caching schemes, novel hierarchical approaches that prune low-information context chunks before performing attribution, and methods that leverage small proxy LMs to approximate the LOO attributions of a larger target model. We develop and explore the practical application of these schemes to develop an aggregate system called AttriBoT (a **B**ag **o**f **T**ricks for efficient context attribution). When evaluating AttriBoT in the open-book question answering setting, where a model is presented a question with one or more related documents provided in its context, we find that our methods significantly reduce the cost of computing LOO attributions – at times by $>300\times$ – while remaining more faithful to the original model's LOO error than a wide range of baselines. In addition, the cascade of approaches underlying AttriBoT can be naturally composed to attain a Pareto-optimal trade-off between efficiency and accuracy over efficiencies ranging multiple orders of magnitude.

To enable better intepretability of context-augmented LLMs and facilitate future work on context attribution, we release an efficient and user-friendly implementation of AttriBoT.[1]

In the following section, we formalize the problem of context attribution, define the LOO error in detail, and discuss the metrics we use to evaluate a given approximate attribution method's faithfulness. Then, in Section 3, we detail our novel techniques for efficiently approximating LOO error,

[1] https://see-supplementary.com

ultimately producing AttriBoT. Experimental results across multiple model families and datasets are provided in Section 4.1, with related work in Section 5 and a conclusion in section 6.

2 Problem Statement

In this section, we introduce the problem of context attribution (Section 2.1), define the simple and principled Leave-One-Out (LOO) error method for context attribution (Section 2.2), and define how we evaluate approximate LOO attribution methods (Section 2.3).

Notation Suppose that an autoregressive Transformer (Vaswani et al., 2023) language model (e.g. Llama (Dubey et al., 2024), GPT-4 (OpenAI, 2024), etc.) generates a response R conditioned on a query Q and a context C. For the purposes of this paper, we view a language model with parameters θ as a function $p_\theta(R|Q, C)$ that returns the probability of generating a response given a particular query and context. Furthermore, we assume the context can be partitioned into a sequence of sources with a partitioning function $\Pi(C) = [s_i]_{i=1}^N$, where each source s_i is a contiguous chunk of text in the context, like a sentence, paragraph, or document. For notational convenience, we define $|C|$ to be the number of sources in $\Pi(C)$.

2.1 Context Attribution

A large body of work on feature attribution has studied the relationship between a model's predictions and input features (Li et al., 2016; Wu et al., 2021). More recently, the problem of *context attribution*, coined by Cohen-Wang et al. (2024), has been introduced as a special case of feature attribution, where a response generated by an LLM is attributed back to parts of the LLM's contextual information.

Formally, we follow Cohen-Wang et al. (2024) and define a context attribution method as a function $\tau(\theta, R, s_1, ..., s_{|C|}) \in \mathbb{R}^{|C|}$ that maps a language model's parameters θ, response R, and sources $s_i \in \Pi(C)$ to a vector of real-valued scores indicating each source's importance to the model's response. While Cohen-Wang et al. (2024) leave the dependence of τ on θ and R implicit, we make this dependence explicit because we will later explore algorithms where multiple models are used and may be fed other models' responses.

2.2 Leave-One-Out Attribution

There are many notions of what makes a source important that each lead to different choices of τ. Perhaps the simplest and most interpretable notion of importance is that important sources lead to large changes in the likelihood of the model's response when they are removed from the original context. This quantity, often referred to as the Leave-One-Out (LOO) error, leads to the following choice of context attribution function, which we refer to as the "LOO attribution" in the remainder of the paper:

$$\tau_{LOO}(\theta, R, s_1, ..., s_{|C|})_i = \log p_\theta(R|Q, C) - \log p_\theta(R|Q, C \setminus \{s_i\}) \quad (1)$$

In practice, LOO attributions can be impractical to compute for particularly large language models since scoring all $|C|$ sources in the context requires $|C| + 1$ forward passes of the language model – one pass to compute the likelihood of the response given the full context and $|C|$ passes to compute the likelihood with each of the $|C|$ sources individually removed from the context. In realistic settings where models are provided with a large amount of contextual information (i.e. $|C|$ is very large), computing LOO attributions for all sources in a context can be orders of magnitude more expensive than generating the response itself. Thus, the remainder of this paper explores a variety of methods for efficiently approximating τ_{LOO}.

2.3 Evaluating Approximate Leave-One-Out Attributions

Much past work on intrepretability and attribution measures performance in terms of how well a given method matches human annotations of source importance. Our focus is instead on efficiently approximating the attributions of a "target" LLM whose LOO attributions are prohibitively expensive to compute. We therefore introduce a straightforward evaluation procedure for measuring effi-

Method	Method Parameters	Theoretical FLOPs	Speedup over LOO
LOO	N/A	$2PT\|C\|(\|C\|-1)$	1
KV Caching	N/A	$PT\|C\|(\|C\|-1)$	2
Proxy	P' : Proxy model size	$2P'T\|C\|(\|C\|-1)$	$\frac{P}{P'}$
Pruning	P' : Proxy model size α : Fraction of sources	$2PT\|C\|((\alpha^2+\frac{P'}{P})\|C\|-\alpha-\frac{P'}{P})$	$\frac{P(\|C\|-1)}{(\alpha^2P+P')\|C\|-\alpha P-P'}$
Hierarchical	H : Sources per group β : Fraction of groups	$2PT\|C\|((\beta^2+\frac{1}{H})\|C\|-\beta-1)$	$\frac{H(\|C\|-1)}{(\beta^2H+1)\|C\|-\beta H-H}$

Table 1: The theoretical number of floating-point operations (FLOPs) needed by different methods to compute LOO attributions expressed in terms of P, the number of target model parameters, T, the number of tokens per source, and C, the number of context sources.

ciency and determining how faithful an approximate method's attributions are to those produced by the original target model.

Approximation Error Past work has found that for many real-world tasks like summarization and question answering, attributions tend to be sparse, meaning that only a small number of sources in the context significantly influence a model's response (Liu et al., 2024; Kuratov et al., 2024; Yang et al., 2018). Thus, when evaluating how well an algorithm approximates LOO attributions, we are primarily interested in how well the algorithm recovers these few highly contributive sources.

To evaluate approximate LOO attribution methods we first compute the LOO attributions of the target model and identify the most contributive sources. We consider a source to be highly contributive if it has an outlyingly large score. To find outlier scores, we apply the Generalized Extreme Studentized Deviate (ESD) test (Rosner, 1983), a statistical test that returns the outlier values in a collection of data (for more details, see Appendix B.2). We denote the number of detected outlier sources for a given example as n_{out}.

We then test the extent to which approximate LOO attribution methods recover these n_{out} outlier sources using evaluation metrics from information retrieval. In particular, we rank the sources according to their approximate LOO attribution scores and measure the mean Average Precision (mAP) over examples in a dataset (Manning et al., 2008).

Efficiency For each approximate LOO algorithm, we report its practical and theoretical efficiency. To measure practical efficiency, we report the average GPU-seconds needed to compute attribution scores for all sources in a context. For theoretical efficiency, we estimate the number of floating-point operations (FLOPs) needed to compute attributions as a function of the number of model parameters P, the number of sources in the context $|C|$, the number of tokens in each source T, and any other method-specific parameters. To estimate FLOPs, we use the simplifying assumptions from Kaplan et al. (2020) and Hoffmann et al. (2022), i.e. that performing a forward pass for a P-parameter language model on T tokens requires approximately $2PT$ FLOPs.

3 Accelerating Leave-One-Out Attribution with AttriBoT

In this section, we describe the **B**ag **o**f **T**ricks that AttriBoT uses to efficiently compute approximate LOO attributions. For each method, we provide intuition for why it serves as an effective approximation to the LOO error as well as discussion on its theoretical speedup over standard LOO attribution using the target model. Full details on the efficiency gains for each method can be found in Table 1 with derivations available in Appendix A.

3.1 Key-Value Caching

For a context containing $|C|$ sources, computing τ_{LOO} requires $|C|+1$ forward passes (including the first forward pass to generate the response). However, much of this computation is redundant when the underlying language model is an autoregressive Transformer. At each self-attention layer

in an autoregressive Transformer, the key and value tensors for a given position in the sequence are only a function of previous tokens due to autoregressivity or causal masking (Vaswani et al., 2023). Thus, if two inputs share a prefix, the key and value tensors at each position in the prefix will be identical. This redundant computation can be avoided by caching the key and value tensors at each layer and reusing the cached values for inputs that have a previously computed prefix (Pope et al., 2022).

Specifically, when computing LOO attributions, the keys and values can be cached for the full sequence of tokens while computing $p_\theta(R|Q, C)$. Then, for each subsequent forward pass computing $p_\theta(R|Q, C \setminus \{s_i\})$, the cached keys and values for Q and the first $i-1$ sources in C can be reused. In aggregate over all sources, this avoids computation for a total of $(|C|-1)/2$ sources, ultimately saving approximately half of the FLOPs used to compute τ_{LOO}. Notably, ignoring differences due to numerical error of floating point operations, we should expect KV caching to be lossless – i.e., LOO attributions should be the same regardless of whether KV caching is used.

3.2 Hierarchical Attribution

In many settings, we expect that the context has a hierarchical nature – for example, the context might comprise a sequence of paragraphs that can each be broken down into a sequence of sentences. This structure can be leveraged to efficiently identify sources that are likely to have high LOO attributions, allowing us to avoid computing τ_{LOO} for every source in the context.

Imagine we want to compute LOO attributions at the sentence level. The key assumption underlying our hierarchical attribution algorithm is that the sum of the LOO attributions for k sentences in a paragraph can be closely approximated by a single Leave-k-out attribution score computed by removing the paragraph as a whole. In cases where this holds, paragraphs whose removal incurs a large drop in the response's likelihood are also likely to contain highly contributive sentences.

Specifically, to perform hierarchical attribution we assume the context can be partitioned into a sequence of "source groups" (e.g., paragraphs) $\Pi_g(C) = [G_i]_{i=1}^{M}$ and each of these groups can be further decomposed into its constituent sources (e.g., sentences) $\Pi_s(G_i) = [s_j]_{j=1}^{M_i}$, where M_i is the number of sentences in group G_i. We first compute the LOO error for each source group, $\tau_{LOO}(\theta, R, G_i, \ldots, G_M)$, and keep only a fraction, β, of the groups with the highest scores. From these high-attribution groups, we construct a shortened context $\hat{C}$ comprising only the retained groups and compute $\tau_{LOO}(\theta, R, s_1, \ldots, s_{|\hat{C}|})$ for the sources $s_i \in \Pi_s(\hat{C})$.

For example, when processing a document with 10 paragraphs and using $\beta = 0.2$, we would first compute the LOO error corresponding to the removal each of the paragraphs and retain the two paragraphs with the highest LOO score. Finally, to compute attributions for the sentences in these two paragraphs, we concatenate the paragraphs into a truncated context and compute the LOO error incurred by removing each sentence from the truncated context.

This hierarchical approach achieves a speedup over LOO by (1) first doing fewer forward passes over the full context when computing attributions at the source group level and (2) doing fewer forward passes on a shortened context after the low-attribution source groups are removed. With the simplifying assumption that each source group contains H sources, if we keep only a constant number source groups irrespective of the total number of sources (i.e. $\beta \sim 1/|C|$), then this method's speedup over LOO is approximately H for long contexts.

3.3 Proxy Modeling

The high cost of computing the LOO error stems from having to compute $|C|$ forward passes through a large and computationally expensive language model. The cost of each of these forward passes is naturally cheaper for smaller models. If a smaller "proxy" model's attributions are faithful to a larger target model whose attributions we hope to obtain, this raises the possibility of reducing the cost of computing LOO attributions by using the proxy model instead. In particular, we might hope that a smaller model from the same model family (i.e. sharing a model architecture, training dataset, and training objective but differing in its parameter count) produces similar attributions to a target model. Specifically, we take the response R generated by the target model, and use the approximation $\tau_{LOO}(\theta_{proxy}, R, s_1, \ldots, s_{|C|}) \approx \tau_{LOO}(\theta, R, s_1, \ldots, s_{|C|})$. The proxy model does

not need to produce the response, but rather emulate the conditional likelihood of the target model's response given a context. In practice, as further discussed in section 4.2.1, we find that when using the target model's response to compute attribution scores, the LOO errors from a target model are highly correlated with those from a small proxy model from the target model.

3.4 Proxy Model Pruning

While small proxy models can approximate the context attributions of larger target models reasonably well, one way to improve the fidelity of this approximation is to use the proxy model to prune away low-attribution sources and then re-score the remaining sources with the target model. Specifically, we take the response generated by the target model, use it to compute $\tau_{LOO}(\theta_{proxy}, R, s_1, \ldots, s_{|C|})$, and keep only a fraction, α, of the highest scoring sources. Then we reconstitute the context keeping only these top sources and recompute their LOO attributions using the target model.

Compared to simply using a proxy model, this method is more expensive since it requires forward passes of the target model for each of the non-pruned spans. However, this extra compute is spent improving the approximation error for the high-attribution sources whose LOO attributions are most important to recover accurately. If the number of sources kept after pruning is constant (i.e. $\alpha \sim 1/|C|$), then this method provides a speedup of roughly P/P' for long contexts, the same speedup achieved by proxy modeling alone.

3.5 Composing Methods in the AttriBoT Bag of Tricks

The methods described above improve the efficiency of computing LOO attributions by avoiding redundant calculations during forward passes, reducing the size of the model used to perform the forward passes, and/or reducing the total number of forward passes. As different methods target different factors that make exact LOO computation expensive, these methods can often be composed together to achieve even greater efficiency. Concretely, in addition to the individual methods described previously, we test the following composite methods in our experiments: KV Caching + Hierarchical Attribution, KV Caching + Proxy Modeling, KV Caching + Proxy Model Pruning, and KV Caching + Proxy Modeling + Hierarchical Attribution.

4 Experiments

To validate AttriBoT, we compare its efficiency and faithfulness to a range of baseline context attribution methods. Our focus is on confirming that our approximations provide a Pareto-optimal trade-off between speed and faithfulness to the target model.

4.1 Setup

In this section, we describe the models, datasets, and baseline methods used to evaluate AttriBoT, as well as the computational resources used to run our experiments.

4.1.1 Models

To ensure that AttriBoT is valuable in state-of-the-art settings, we focus on recent performant open LLMs. We therefore experiment with two target models whose LOO context attributions we aim to efficiently approximate: the 70B-parameter instruction-tuned LLM from the Llama 3.1 model family (Dubey et al., 2024) and the 72B-parameter instruction-tuned LLM from the Qwen 2.5 model family (Yang et al., 2024). As proxy models, we use smaller instruction-tuned models from each of these model families. For Llama 70B, this includes the 8B-parameter model from the Llama 3.1 family and the 1B and 3B-parameter models from the Llama 3.2 family. For Qwen 72B, we use the 0.5B, 1.5B, 3B, 7B, and 32B-parameter models from the Qwen 2.5 model family as proxies. Each model within a given family shares an architecture, pre-training objective, and pre-training dataset. We report results for the Llama models in the main text and replicate select experiments with the Qwen models in Appendix C.

4.1.2 DATASETS

There are a number of applications of context attribution that range from verifying faithfulness of model summaries to detecting malicious prompts. In this study, we focus on the well-studied and widespread setting of open-book question answering (OBQA), which involves prompting a model to answer a question that is supported by a provided context. Since many tasks can be formulated as answering a question about contextual information (McCann et al., 2018), we expect results on OBQA to be a reliable indicator of performance in other settings. Given that each method's efficiency depends on the context length, we consider datasets covering a range of context lengths. To this end, we focus on three open-book QA datasets:

1. SQuAD 2.0 (Rajpurkar et al., 2018): A reading comprehension benchmark where a model is presented a question along with a Wikipedia article containing the answer. For this dataset, we consider sources to be individual sentences in the context and source groups for hierarchical attribution to be paragraphs.
2. HotpotQA (Yang et al., 2018): A multi-hop question answering benchmark that requires reasoning across paragraphs from multiple Wikipedia pages, as well as ignoring distractor paragraphs, in order to answer a question. For this dataset, we consider sources to be individual sentences in the context and source groups for hierarchical methods to be paragraphs.
3. QASPER (Dasigi et al., 2021): A document-grounded, information-seeking question answering dataset where each context is a natural language processing paper. Answering questions in the QASPER dataset requires complex reasoning about claims made in multiple parts of a paper. For this dataset, we consider sources to be paragraphs and source groups for hierarchical methods to be sections of the paper provided in the context.

For more details on these datasets, see Appendix B.1.

4.1.3 BASELINES

We compare AttriBoT's performance to a diverse set of context attribution methods:

1. Attention Weights: The self-attention operation in the Transformer architecture provides an implicit way of determining which input entries are influential (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). Concretely, we compute the total attention weight for each source by summing the attention weights of a source's tokens across all attention heads and layers.
2. Gradient Norm: The gradient of the model's response likelihood with respect to its input provides a first-order approximation of the model's sensitivity input perturbations, and thus is a useful interpretation technique (Yin & Neubig, 2022; Mohebbi et al., 2021). Specifically, we calculate the Frobenius norm of the gradient of the response's likelihood with respect to the token embeddings of each source.
3. Sentence embedding similarity: As a simple model-agnostic baseline, we compute the similarity between sentence embeddings for the generated response and each one of the sources. We generate sentence embeddings using the all-MiniLM-L6-v2 SentenceBERT model (Reimers & Gurevych, 2019) and compute similarity via cosine similarity.
4. ContextCite: This method estimates the importance of each context source by training a linear surrogate model to estimate the likelihood of a response given a set of context sources. The trained linear model's weights for each source in the context act as an attribution score (Cohen-Wang et al., 2024). More background on ContextCite is provided in in Appendix B.3.

For more details on baseline methods, see Appendix B.3.

4.1.4 COMPUTATIONAL RESOURCES

All experiments were run on servers with 4 NVIDIA A100 SXM4 GPUs with 80GB of VRAM. For experiments involving models with fewer than 15 billion parameters, we use only a single GPU. For models with more parameters, we use model parallelism across multiple GPUs.

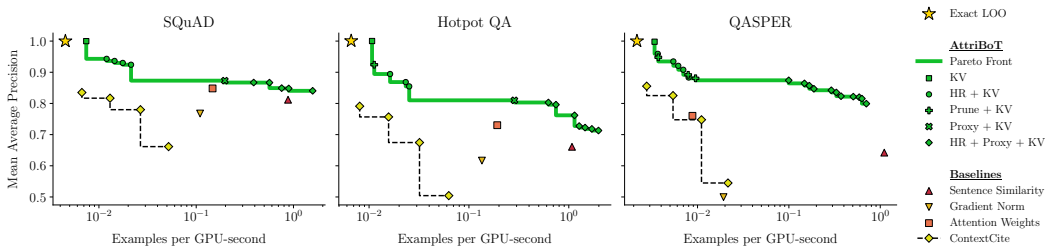

Figure 2: We plot the mean average precision compared to attributions of the target model against the GPU time for AttriBoT and a variety of baselines using Llama 3.1 70B Instruct as the target model and smaller Llama instruct variants as proxy models. Across all three datasets, AttriBoT is consistently Pareto-optimal over multiple orders of magnitude.

4.2 Results

We now provide empirical results that motivate the approximations made in AttriBoT and demonstrate its effectiveness in the OBQA setting.

4.2.1 Validating AttriBoT's Approximations

Hierarchical Attribution The assumption underlying our hierarchical attribution method is that the sum of the LOO scores for k sources in a source group (e.g., sentences in a paragraph) can be estimated by computing the Leave-k-Out attribution of the source group as a whole. We empirically test this assumption for Llama 70B by comparing LOO attributions for all paragraphs in HotpotQA with the sum each paragraph's sentence-level LOO attributions. Figure 1 (Middle) shows that the these two quantities have a very high Pearson correlation ($R = 0.97$), meaning that the removal of low-attribution source groups in our proposed hierarchical attribution method is likely to keep most high-attribution sources.

Proxy Modeling Proxy modeling assumes that two models from the same model family will have similar LOO attributions. Thus, if the target model is large, its LOO attributions for a particular response can be efficiently approximated by attributing the same response with a smaller proxy model. We test this claim by measuring the correlation between the HotpotQA sentence-level attribution scores from Llama 70B and smaller proxy models from the Llama model family, ranging from 1B to 8B parameters. We find that LOO attributions from each of the proxy models are very well correlated with the Llama 70B's LOO attributions. Shown in Figure 1 (Left) the Pearson correlation is as high as 0.94 for the largest proxy model with 8B parameters and only decreases to 0.88 for the smallest 1B-parameter proxy model.

Proxy Model Pruning Like proxy modeling, proxy model pruning assumes that small models can capably approximating a large model's attributions. However, pruning additionally requires the proxy model to rank outlier sources as one of the top-$\alpha|C|$ sources. In practice we find this to be the case. Figure 1 (Right) plots the outlier recall of different proxy models as a function of the number of sources kept when pruning a context. In practice, we find that Llama 8B achieves 85% recall of sources with outlier Llama 70B attributions while only keeping the top-10 spans.

4.2.2 AttriBoT's Pareto-Optimality

Our main results, comparing AttriBoT to the baselines listed in Section 4.1.3 on the datasets described in Section 4.1.2 for the Llama model family, are provided in Figure 2. We find that the techniques composed in AttriBoT produce a Pareto-optimal method across many orders of magnitude of costs on all three of our evaluation datasets. Compared to ContextCite, the only other method we are aware of that computes attributions via repeated forward passes on modified contexts, AttriBoT can provide attributions that are equally faithful to a target model's LOO attributions with orders of magnitude lower cost. Compared to cheaper attribution baselines (sentence similarity, gradient norm, and attention weights), AttriBoT remains Pareto-optimal and can even be more efficient than

using sentence embeddings (which are produced by a small model and do not require any forward passes of a target or proxy model). These results hold true across all datasets, demonstrating that AttriBoT scales well to datasets of varying context length.

KV Caching We find that, in practice, KV caching offers about a $1.6\times$ speedup over computing exact LOO attributions while, as expected, nearly perfectly recovering outlier sources (see Figure 3. While in theory computing attributions with and without KV caching should be identical, in practice we observe a small difference in the attribution values due to accumulated numerical error of floating-point operations.

Proxy Modeling Shown in Figure 3, proxy modeling can be an effective approach for greatly speeding up LOO computation. We find that as the size of the proxy model decreases, its faithfulness to the exact LOO attributions also decreases. However, the faithfulness of attributions from smaller models still remains better than many baselines that use orders of magnitude more compute.

Hierarchical Attribution In Figure 4, we show the accuracy vs. efficiency trade-off of hierarchical attribution. We find that modulating the fraction of source groups retained, β, is an effective method for trading off accuracy for efficiency. Additionally, we find that using the target model yields the most accurate attributions, while substituting the target model for a smaller proxy model achieves greater efficiency.

Proxy Model Pruning Like hierarchical attribution, we show in Figure 5 that modulating the fraction of sources to retain, α, and the proxy model size effectively trades off accuracy for efficiency.

4.2.3 AttriBoT's Impact on Matching Human Annotations

Our main focus has been on efficiently approximating the attributions of a target model. However, it can also be valuable to measure how closely an attribution method matches human-annotated "important" spans. Such annotations are available for the HotpotQA dataset, so we additionally evaluated the impact of using AttriBoT on agreement with human annotations. Full results are in Appendix C.4; as a short summary, we find that the methods introduced in AttriBoT actually remain *more* faithful to the human-annotated attributions than they do to those of the target model. Concretely, we observe that AttriBoT can achieve a $300\times$ speed up with only 10% drop in mAP.

5 Related Work

Generating Citations Computing LOO context attributions involves repeated inference of the target model. In contrast, past work has explored whether it is possible to train LLMs to directly provide citations to sources or spans from their context (Weller et al., 2024; Gao et al., 2023b). Some approaches leverage pipelines for retrieving external facts to ground and verify a generated response while providing explicit attribution (Asai et al., 2023; Li et al., 2024b; Sun et al., 2023; Li et al., 2024a; Chen et al., 2024; Huo et al., 2023; Gao et al., 2023a). Other models are explicitly trained to produce citations, including GopherCite (Menick et al., 2022), which generates quotes from supporting documents after generating a response; LaMDA (Thoppilan et al., 2022), which provides URLs from the information retrieval system used; and WebGPT (Nakano et al., 2022), which provides extracted context from the source webpages. Such citation generation methods only provide insight how each statement is possibly supported by the corresponding source instead of measuring how each source contributes to a response via a causal intervention like source removal. Therefore, while useful for providing evidence for supporting or verifying an LLM's response, these methods cannot necessarily provide insight into *why* an LLM produced a particular output.

While one goal of this line of work is to improve the factuality and reduce hallucinations of LLMs, whether such citations can themselves be considered trustworthy is unclear (Peskoff & Stewart, 2023; Zuccon et al., 2023; Gravel et al., 2023). In fact, previous work has explored generative search engines like Bing Chat, NeevaAI, perplexity.ai, and YouChat that provide inline citations, and find that they frequently contain unsupported statements and inaccurate citations (Liu et al., 2023).

Feature Attribution-Based Explanations The field of neural network interpretability has put forth a huge number of techniques that aim to measure the importance or relevance of each input feature on a model's prediction. Perturbation-based techniques including leave-one-out and masking have been applied on features like words and phrases (Li et al., 2016), input word-vector dimensions, intermediate hidden units (Li et al., 2017), and token spans (Wu et al., 2021). The gradient of a model's output with respect to an input feature indicates the sensitivity of output to input changes and has therefore been used to compute attribution scores (Yin & Neubig, 2022; Mohebbi et al., 2021; Sanyal & Ren, 2021; Sikdar et al., 2021; Enguehard, 2023). We adapted this simple gradient-based attribution approach as one of our baselines. Since attention computes an adaptively weighted average of activations, attention weights can be seen as capturing the correlation between inputs. Attention weights have therefore also been a popular tool to identify important information to the model (Pruthi et al., 2020a; Serrano & Smith, 2019; Wiegreffe & Pinter, 2019) and we therefore use attention weight as one of our baselines. Local interpretable surrogate models are also developed to interpret model outputs (Ribeiro et al., 2016; Lundberg & Lee, 2017).

Attributing Predictions to Training Data Apart from attributing an LLM's predictions to input features that correspond to elements in its context, it can be of practical interest to uncover the training data that influenced a particular prediction. Such is the goal of influence functions (Koh & Liang, 2020), which estimate the impact of removing a particular training example by computing the alignment of gradients through a bilinear form with the inverse Hessian. The high cost of computing and inverting the Hessian, as well as the cost of computing gradients for every in example within a potentially gargantuan training dataset, has led to a great deal of work on computing influence functions more efficiently, particularly for LLMs (Grosse et al., 2023; Choe et al., 2024; Pruthi et al., 2020b). Apart from costs, the reliability of influence functions has been questioned (Basu et al., 2021; Bae et al., 2022). As exact LOO for dataset attribution requires a full training run for each data point, most training data attribution methods resort to evaluating with proxy metrics like task accuracy as a function of the number of high-attribution training examples removed, instead of ground truth LOO attributions as in our work. Beyond directly measuring influence, other works have sought to understand the influence of pre-training data on factual knowledge internalized by the model (Kandpal et al., 2023; Chang et al., 2024; Razeghi et al., 2022; Antoniades et al., 2024). More broadly, the impact of pre-training data on LLM behavior and performance is an important notion for the well-studied problem of data selection; see (Albalak et al., 2024).

6 Conclusion

In this paper, we introduced AttriBoT, a **B**ag **o**f **T**ricks for efficiently attributing an LLM's predictions to spans of text in its input context. AttiBoT leverages a series of novel techniques for approximating the attributions of a large target model, including reusing computations through key-value caching, performing hierarchical attributions to reduce the number of forward passes, and leveraging a smaller proxy model whose attributions reliably approximate those of the target model. Taken together, AttriBoT can provide a $>300\times$ speedup, making attributing a response $30\times$ more efficient than even generating a response in realistic settings, while remaining more faithful to the target model's attributions than prior methods for efficient context attribution. In addition, the components of AttriBoT can be included, excluded, and tuned to produce a Pareto-optimal trade-off between faithfulness and speed. While our experimental results primarily focus on the general setting of open-book question answering, we anticipate AttriBoT will also be useful for detecting malicious prompts and model hallucinations. We hope that our efficient and easy-to-use implementation of AttriBoT[1] ensures that it has real-world impact and also enables future work on efficient context attribution.

References

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, et al. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*, 2024.

Antonis Antoniades, Xinyi Wang, Yanai Elazar, Alfonso Amayuelas, Alon Albalak, Kexun Zhang, and William Yang Wang. Generalization vs memorization: Tracing language models' capabilities

back to pretraining data. *arXiv preprint arXiv:2407.14985*, 2024.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection, 2023. URL https://arxiv.org/abs/2310.11511.

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If influence functions are the answer, then what is the question? *Advances in Neural Information Processing Systems*, 2022.

Samyadeep Basu, Phil Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *International Conference on Learning Representations*, 2021.

Nick Cammarata, Shan Carter, Gabriel Goh, Chris Olah, Michael Petrov, Ludwig Schubert, Chelsea Voss, Ben Egan, and Swee Kiat Lim. Thread: Circuits. *Distill*, 2020. doi: 10.23915/distill.00024. https://distill.pub/2020/circuits.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. How do large language models acquire factual knowledge during pretraining? *arXiv preprint arXiv:2406.11813*, 2024.

Jifan Chen, Grace Kim, Aniruddh Sriram, Greg Durrett, and Eunsol Choi. Complex claim verification with evidence retrieved in the wild. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3569–3587, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.196. URL https://aclanthology.org/2024.naacl-long.196.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Minsoo Kang, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, Jeff Schneider, Eduard Hovy, Roger Grosse, and Eric Xing. What is your data worth to gpt? llm-scale data valuation with influence functions, 2024. URL https://arxiv.org/abs/2405.13954.

Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. Contextcite: Attributing model generation to context, 2024. URL https://arxiv.org/abs/2409.00729.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4599–4610, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.365. URL https://aclanthology.org/2021.naacl-main.365.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak,

Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li,

Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Joseph Enguehard. Sequential integrated gradients: a simple but effective method for explaining language models, 2023. URL https://arxiv.org/abs/2305.15853.

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. RARR: Researching and revising what language models say, using language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16477–16508, Toronto, Canada, July 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.910. URL https://aclanthology.org/2023.acl-long.910.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations, 2023b. URL https://arxiv.org/abs/2305.14627.

Jocelyn Gravel, Madeleine D'Amours-Gravel, and Esli Osmanlliu. Learning to fake it: Limited responses and fabricated references provided by chatgpt for medical questions. *Mayo Clinic Proceedings: Digital Health*, 1(3):226–234, 2023. ISSN 2949-7612. doi: https://doi.org/10.1016/j.mcpdig.2023.05.004. URL https://www.sciencedirect.com/science/article/pii/S2949761223000366.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1): 1–21, 1969. doi: 10.1080/00401706.1969.10490657. URL https://www.tandfonline.com/doi/abs/10.1080/00401706.1969.10490657.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Siqing Huo, Negar Arabzadeh, and Charles Clarke. Retrieving supporting evidence for generative question answering. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP '23, pp. 11–20, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400704086. doi: 10.1145/3624918.3625336. URL https://doi.org/10.1145/3624918.3625336.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Proceedings of NAACL-HLT*, 2019.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, 2023.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions, 2020. URL https://arxiv.org/abs/1703.04730.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *arXiv preprint arXiv:2406.10149*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL https://arxiv.org/abs/2005.11401.

Dongfang Li, Zetian Sun, Baotian Hu, Zhenyu Liu, Xinshuo Hu, Xuebo Liu, and Min Zhang. Improving attributed text generation of large language models via preference learning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics ACL 2024*, pp. 5079–5101, Bangkok, Thailand and virtual meeting, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.301. URL https://aclanthology.org/2024.findings-acl.301.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp, 2016. URL https://arxiv.org/abs/1506.01066.

Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure, 2017. URL https://arxiv.org/abs/1612.08220.

Xiaonan Li, Changtai Zhu, Linyang Li, Zhangyue Yin, Tianxiang Sun, and Xipeng Qiu. LLatrieval: LLM-verified retrieval for verifiable generation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5453–5471, Mexico City, Mexico, June 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.305. URL https://aclanthology.org/2024.naacl-long.305.

Nelson F. Liu, Tianyi Zhang, and Percy Liang. Evaluating verifiability in generative search engines, 2023. URL https://arxiv.org/abs/2304.09848.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017. URL https://arxiv.org/abs/1705.07874.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, USA, 2008. ISBN 0521865719.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, and Nat McAleese. Teaching language models to support answers with verified quotes. *ArXiv*, abs/2203.11147, 2022. URL https://api.semanticscholar.org/CorpusID:247594830.

Hosein Mohebbi, Ali Modarressi, and Mohammad Taher Pilehvar. Exploring the role of BERT token representations to explain sentence probing results. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 792–806, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.61. URL https://aclanthology.org/2021.emnlp-main.61.

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. Webgpt: Browser-assisted question-answering with human feedback, 2022. URL https://arxiv.org/abs/2112.09332.

OpenAI. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Denis Peskoff and Brandon Stewart. Credible without credit: Domain experts assess generative language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 427–438, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-short.37. URL https://aclanthology.org/2023.acl-short.37.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference, 2022. URL https://arxiv.org/abs/2211.05102.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. Learning to deceive with attention-based explanations. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4782–4793, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.432. URL https://aclanthology.org/2020.acl-main.432.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33: 19920–19930, 2020b.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://aclanthology.org/P18-2124.

Yasaman Razeghi, Robert L Logan IV, Matt Gardner, and Sameer Singh. Impact of pretraining term frequencies on few-shot numerical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 840–854, 2022.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016. URL https://arxiv.org/abs/1602.04938.

Bernard Rosner. Percentage points for a generalized esd many-outlier procedure. *Technometrics*, 25 (2):165–172, 1983. ISSN 00401706. URL http://www.jstor.org/stable/1268549.

Soumya Sanyal and Xiang Ren. Discretized integrated gradients for explaining language models. In *Conference on Empirical Methods in Natural Language Processing*, 2021. URL https://api.semanticscholar.org/CorpusID:237363853.

Sofia Serrano and Noah A. Smith. Is attention interpretable?, 2019. URL https://arxiv.org/abs/1906.03731.

Sandipan Sikdar, Parantapa Bhattacharya, and Kieran Heese. Integrated directional gradients: Feature interaction attribution for neural NLP models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 865–878, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.71. URL https://aclanthology.org/2021.acl-long.71.

Hao Sun, Hengyi Cai, Bo Wang, Yingyan Hou, Xiaochi Wei, Shuaiqiang Wang, Yan Zhang, and Dawei Yin. Towards verifiable text generation with evolving memory and self-reflection. *ArXiv*, abs/2312.09075, 2023. URL https://api.semanticscholar.org/CorpusID:266210156.

Gemini Team. Gemini: A family of highly capable multimodal models, 2024. URL https://arxiv.org/abs/2312.11805.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam M. 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, Yanqi Zhou, Chung-Ching Chang, I. A. Krivokon, Willard James Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Hartz Søraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Díaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, V. O. Kuzmina, Joseph Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Rogers Croak, Ed Huai hsin Chi, and Quoc Le. Lamda: Language models for dialog applications. *ArXiv*, abs/2201.08239, 2022. URL https://api.semanticscholar.org/CorpusID:246063428.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Orion Weller, Marc Marone, Nathaniel Weir, Dawn Lawrie, Daniel Khashabi, and Benjamin Van Durme. "according to . . . ": Prompting language models improves quoting from pre-training data. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2288–2301, St. Julian's, Malta, March 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.eacl-long.140.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 11–20, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1002. URL https://aclanthology.org/D19-1002.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. Perturbed masking: Parameter-free probing for analyzing and interpreting bert, 2021. URL https://arxiv.org/abs/2004.14786.

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, 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, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Kayo Yin and Graham Neubig. Interpreting language models with contrastive explanations, 2022. URL https://arxiv.org/abs/2202.10419.

Guido Zuccon, Bevan Koopman, and Razia Shaik. Chatgpt hallucinates when attributing answers. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, SIGIR-AP '23, pp. 46–51, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400704086. doi: 10.1145/3624918.3625329. URL https://doi.org/10.1145/3624918.3625329.

A THEORETICAL EFFICIENCY DERIVATIONS

In this section, we provide derivations for the theoretical efficiencies reported in Table 1.

Preliminaries We assume a target model with P parameters, a context C containing $|C|$ sources, and that each source in the context contains exactly T tokens. Following Kaplan et al. (2020) and Hoffmann et al. (2022), we approximate the number of FLOPs for a forward pass of a P-parameter model on a sequence of T tokens to be $2PT$ FLOPs.

To simplify the analysis, we ignore the fact the query and response are also part of the model's input, as these are typically negligible compared to the length of the context. Accounting for the query and response would simply add a small constant factor to each method's theoretical runtime. Finally, we assume that the likelihood of the full sequence under the target model, $p_\theta(R|Q, C)$ can be computed while generating R, and is thus not accounted for in the analysis of each attribution method.

A.1 EXACT LOO ATTRIBUTION

Exactly calculating LOO attributions under the target model requires computing $p_\theta(R|Q, C \setminus \{s_i\}$ for all in $i \in 1, \ldots, |C|$. In total, this is $|C|$ forward passes each on sequences containing $T(C-1)$ tokens. Thus, the number of FLOPs needed to compute attributions is $2PT|C|(|C|-1)$.

A.2 KV CACHING

KV caching allows us to avoid FLOPs spent on prefixes for which keys and values have already been computed. Since keys and values can be cached for the full sequence while computing $p_\theta(R|Q, C)$, we can avoid recomputing the keys and values for the first $i-1$ sources while computing $p_\theta(R|Q, C \setminus \{s_i\})$. Thus the total number of FLOPs for computing all LOO attributions can be written as:

$$\sum_{i=1}^{|C|} 2PT(|C|-i) = PT|C|(|C|-1)$$

This provides a speedup of a factor of 2 compared to computing LOO attributions with no caching.

A.3 PROXY MODELING

Proxy modeling is identical to computing exact LOO attributions, except with a model with fewer parameters. If the number of proxy model parameters is P', then the number of FLOPs needed to compute attributions is $2P'T|C|(|C|-1)$ and the speedup gained by using a proxy model is P/P'.

A.4 HIERARCHICAL ATTRIBUTION

Our analysis of hierarchical attribution makes the simplifying assumption that each source group in the context contains exactly H sources.

We start by first considering the initial source group-scoring step in the hierarchical attribution algorithm. This step requires performing $|C|/H$ forward passes (one for each source group) each on a sequence of $T(|C|-H)$ tokens. Thus, the initial source group scoring step uses $2PT(|C|/H)(|C|-H)$ FLOPs.

The second step in hierarchical attribution computes scores at the source level on a shortened context containing only $\beta|C|$ sources. Thus, the second step of hierarchical attribution uses $2PT(\beta|C|)(\beta|C|-1)$.

Combining these steps together, we can compute the total number of FLOPs for hierarchical attribution:

$$\begin{aligned}\text{Total FLOPs} =& 2PT\left(\frac{|C|}{H}\right)(|C|-H) + 2PT(\beta|C|)(\beta|C|-1) \\ =& 2PTC\left(\frac{|C|}{H} - 1 + \beta^2|C| - \beta\right) \\ =& 2PTC\left(\left(\beta^2 + \frac{1}{H}\right)|C| - \beta - 1\right)\end{aligned}$$

By comparing to the number of FLOPs used for exact LOO attribution we get the following speedup factor:

$$\frac{|C|-1}{\left(\beta^2 + \frac{1}{H}\right)|C| - \beta - 1} = \frac{H(|C|-1)}{(\beta^2 H + 1)|C| - \beta H - H}$$

Next, we consider the speedup as the number of contexts grows and the number of source groups kept after the first group-level scoring pass is kept constant, i.e., $\beta = k/|C|$ for some constant k.

$$\begin{aligned}\frac{H(|C|-1)}{(\beta^2 H + 1)|C| - \beta H - H} =& \frac{H(|C|-1)}{\left(\left(\frac{k}{|C|}\right)^2 H + 1\right)|C| - \frac{k}{|C|}H - H} \\ =& \frac{H(|C|-1)}{\frac{k^2}{|C|}H + |C| - \frac{k}{|C|}H - H} \\ \approx& \frac{H|C|}{|C|} \quad \text{for large } |C| \\ =& H\end{aligned}$$

Thus, for contexts with many sources, hierarchical attribution's speedup approaches H, the number of sources per source group.

A.5 Proxy Model Pruning

The first stage of proxy model pruning simply uses a proxy model to score each source. Using the result from Appendix A.3, this operation uses $2P'T|C|(|C|-1)$ FLOPs, where P' is the number of parameters in the proxy model. The second stage of the proxy model pruning algorithm uses the target model to score the remaining $\alpha|C|$ sources, requiring $2PT(\beta|C|)(\beta|C|-1)$ FLOPs. Thus the total number of FLOPs can be written as:

$$\begin{aligned}\text{Total FLOPs} =& 2P'T|C|(|C|-1) + 2PT(\beta|C|)(\alpha|C|-1) \\ =& 2PT|C|\left(\left(\frac{P'}{P}\right)(|C|-1) + \alpha(\alpha|C|-1)\right) \\ =& 2PT|C|\left(\frac{P'}{P}|C| - \frac{P'}{P} + \alpha^2|C| - \alpha\right) \\ =& 2PT|C|\left(\left(\alpha^2 + \frac{P'}{P}\right)|C| - \alpha - \frac{P'}{P}\right)\end{aligned}$$

When compared to exact LOO attribution with the target model, this method provides a speedup of:

$$\frac{|C|-1}{\left(\alpha^2 + \frac{P'}{P}\right)|C| - \alpha - \frac{P'}{P}} = \frac{P(|C|-1)}{(\alpha^2 P + P')|C| - \alpha P - P'}$$

Next, we consider the speedup as the number of contexts grows and the number of sources kept after proxy model pruning is kept constant, i.e. $\alpha = k/|C|$ for some constant k.

$$\begin{aligned}\frac{P(|C|-1)}{(\alpha^2 P + P')|C| - \alpha P - P'} &= \frac{P(|C|-1)}{\left(\left(\frac{k}{|C|}\right)^2 P + P'\right)|C| - \frac{k}{|C|}P - P'} \\ &= \frac{P(|C|-1)}{\frac{k^2}{|C|}P + P'|C| - \frac{k}{|C|}P - P'} \\ &\approx \frac{P|C|}{P} \quad \text{for large } |C| \\ &= P\end{aligned}$$

Thus, as the number of sources in the context grows, the speedup of proxy model pruning approaches P/P'.

B Further Details on Experimental Setup

In this section, we provide more details on the experimental setup we use to evaluate context attribution methods.

B.1 Datasets

Preprocessing Before evaluating context attribution methods on SQuAD 2.0, HotpotQA, and QASPER, we first take the following preprocessing steps to make them suitable for running context attribution.

SQuAD2.0:

1. Filter dataset to only include examples that are labeled as possible to answer given the context (SQuAD 2.0 contains both questions where the answer is and is not present in the contextual information).
2. Truncate each example's supporting Wikipedia page context to 10 paragraphs and filter dataset to only examples where the answer lies in the first 10 paragraphs of the context. This is meant to filter out examples with a very large number of context sources, as computing exact LOO attributions on these examples is impractical.
3. Randomly sample 1000 examples from the remaining set of examples.

Hotpot QA:

1. Sample the first 1000 examples of the dataset.

QASPER:

1. Filter dataset to only examples with contexts containing fewer than 60 paragraphs. This limits the context length to make exact LOO attribution tractable and also removes improperly parsed contexts where every line in a LaTeXmath environment is listed as its own paragraph.
2. Randomly sample 1000 examples from the remaining set of examples. In our experiment, there are 2 empty examples and they are ignored, resulting in 998 examples.

We report summary statistics for each dataset after preprocessing in Table 2.

Prompting For each dataset, we use the following prompt templates to format a query and context into a prompt suitable for instruction-tuned models.

SQuAD 2.0 and HotpotQA:

```
Answer the question based on the provided context
```

Dataset	Examples	Source Type	Sources/ Example	Source Groups/ Example	Tokens/ Example	Generation GPU-seconds/Example
HotpotQA	1000	Sentence	41.2	9.9	1,249	18
SQuAD2.0	1000	Sentence	50.1	10.0	1,592	19.6
QASPER	998	Paragraph	38.2	12.7	4,116	51.7

Table 2: Summary statistics for each of the evaluation datasets.

```
Context:
{context}
Question: {question}
```

QASPER:

```
Answer the question based on the following scientific paper:
{context}
Question: {question}
```

B.2 Outlier Detection with Extreme Studentized Deviate Test

The generalized Extreme Studentized Deviate (ESD) test (Rosner, 1983) is a statistical test for detecting an unknown number of outliers in a univariate set of data. This test is an iterated form of Grubbs' test for detecting a single outlier in a set of data (Grubbs, 1969).

The ESD test is parameterized by the maximum possible number of outliers, k, and the significance value α. It works by iteratively selecting the largest value in the dataset, computing that value's Grubbs' statistic G, testing if G is larger than the critical value G_{crit}, and removing the tested value from the dataset. This process stops when the selected value no longer has a Grubbs' statistic greater than the critical value or k values are tested.

The Grubbs' statistic and critical value are defined as is defined as follows for a collection of data $Y_1, \ldots, Y_N$:

$$G = \frac{\max_{i=1,\ldots,N} Y_i - \bar{Y}}{s}$$

$$G_{crit} = \frac{(N-1) \times t}{\sqrt{N} \times \sqrt{N-2+t^2}}$$

where $\bar{Y}$ and s are the mean and standard deviation of $Y_1, \ldots, Y_N$, and t is the upper critical value of the t-distribution with $N-2$ degrees of freedom and a significance level of $\frac{\alpha}{2n}$.

For our experiments we set the significance level $\alpha = 0.05$ and the maximum number of outliers $k = 50$.

B.3 Further Details on ContextCite Baseline

ContextCite works by training a linear surrogate model to approximate the LOO attributions of a target model. Given context sources $S = \{s_1, \ldots, s_{|C|}\}$, ContextCite samples n ablation vectors $\vec{v_1}, \ldots, \vec{v_n} \in \mathbb{R}^{|C|} \sim \text{Bernoulli}(p)$ that act as masks indicating which context sources to remove/keep for each forward pass. Based on these ablation vectors, ContextCite performs a single forward pass on each ablated context and records the change in logit-scaled response probability, τ_i, caused by the removal of the ablated sources. Next, a Lasso regression surrogate model is fit to map each ablation vector $\vec{v_i}$ to the corresponding change in response probability τ_i. Finally, the attribution scores for context source j is simply the weight that the trained Lasso model assigns to the j'th source.

C Additional Experimental Results

C.1 Efficiency vs. Accuracy Tradeoff for AttriBoT Methods

We provide plots illustrating the efficiency vs. accuracy tradeoff for each AttriBoT acceleration method in Figure 3, Figure 4, and Figure 5. Overall, we find that each method provides parameters that effectively vary the accuracy-efficiency tradeoff and are Pareto optimal with respect to baseline context attribution methods.

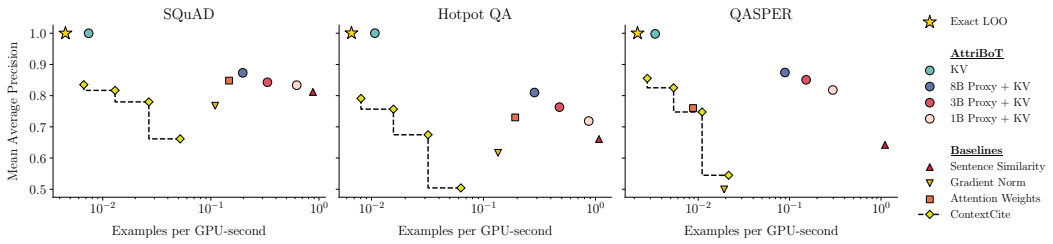

Figure 3: Plot showing accuracy vs. efficiency tradeoff of proxy modeling. We find that smaller proxy models produce attributions that are less faithful to the target model's attributions.

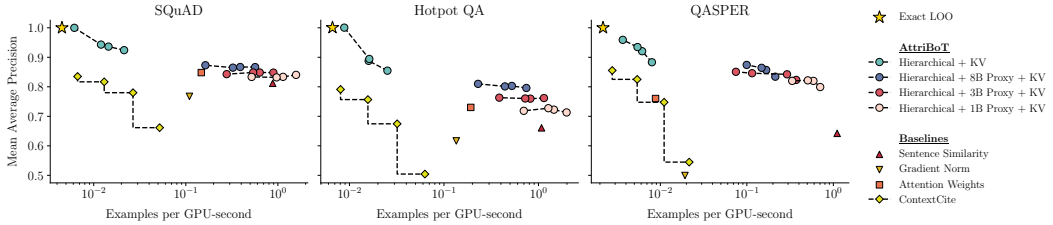

Figure 4: Plot showing accuracy vs. efficiency tradeoff of hierarchical attribution with and without the use of a proxy model. Both varying the size of the proxy model and tuning the number of source groups to retain, β effectively trades attribution faithfulness for speed.

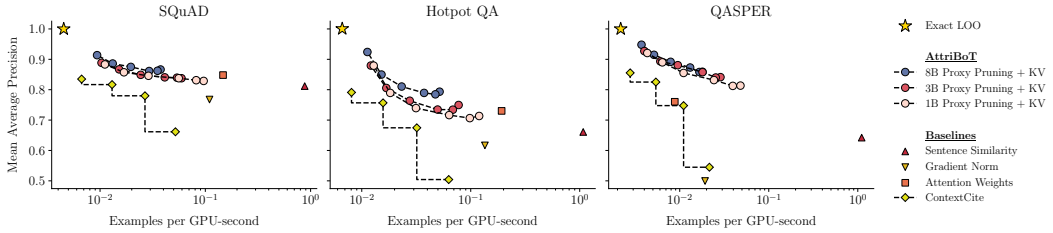

Figure 5: Plot showing the accuracy vs. efficiency tradeoff of proxy model pruning. We find that varying the size of the proxy model and the fraction of sources to retain, α effectively trades attribution faithfulness for speed.

C.2 AttriBoT's Effectiveness Across Different Target Models

To demonstrate AttriBoT's effectiveness for approximating LOO attributions for different target models, we replicate the experiments on HotpotQA using Qwen 72B as the target model. Shown in Figure 6, we observe a nearly identical Pareto front as seen in the Llama model family. Aside from the attention weights baseline, AttriBoT is Pareto optimal at all points on the Pareto front. We hypothesize that the lack of Pareto optimality compared to the attention weights baseline comes from surprisingly good performance of the baseline method compared to its performance in other experiments, achieving a mAP about 10 points higher than when used with Llama 70B.

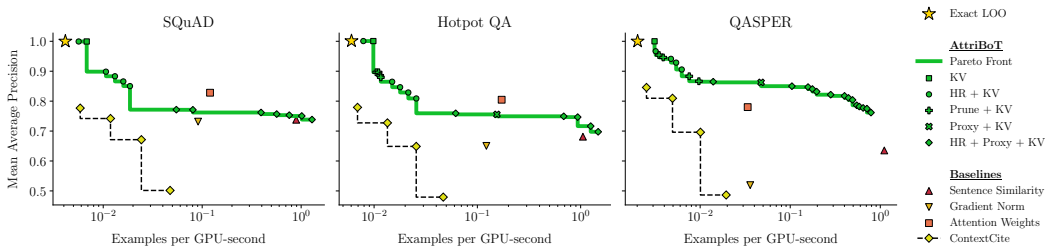

Figure 6: We replicate the experiments on SQuAD, HotpotQA and QASPER using Qwen 72B as the target model to demonstrate the AttriBoT's effectiveness across different target models.

C.3 Affect of Training Data Distribution of Proxy Model

To further investigate the potential impact of proxy model selection and their training data distribution, we considered mismatched proxy models including Mistral 7B Instruct v0.3, as shown in Figure 7, Figure 8, Figure 9, and Figure 10. While our bag of tricks are still effective, using a mismatched proxy model can degrade performance somewhat compared with proxy models trained with same data or distilled from the larger target models.

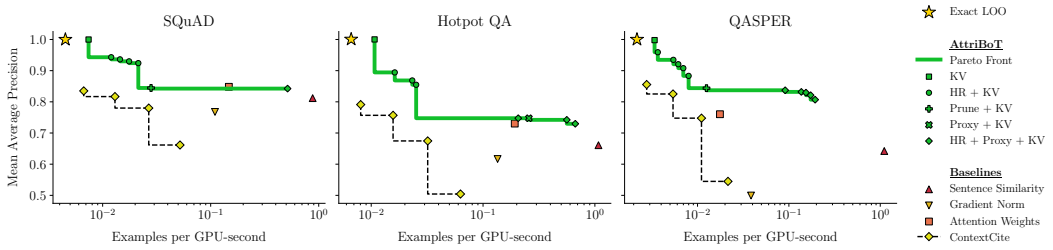

Figure 7: Using Llama 3.1 70B Instruct as the target model and Mistral 7B Instruct v0.3 as the proxy model.

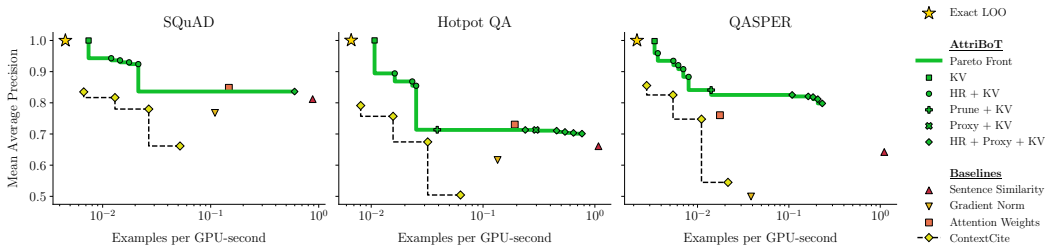

Figure 8: Using Llama 3.1 70B Instruct as the target model and Qwen 2.5 7B Instruct as the proxy model.

C.4 Mean Average Precision Comparing with Human Annotations

The HotpotQA dataset comes with human-annotated "important" text spans in each example's context. While the primary purpose of AttriBoT is to approximate a target model's LOO attributions, here we evaluate the agreement between AttriBoT attributions and human-annotated gold sources. We find that AttriBoT's context attributions closely align with human annotations, achieving only a 10% drop in mAP compared to exact LOO attributions, while achieving a 300× speedup. Results are shown for multiple target models in Figure 11.

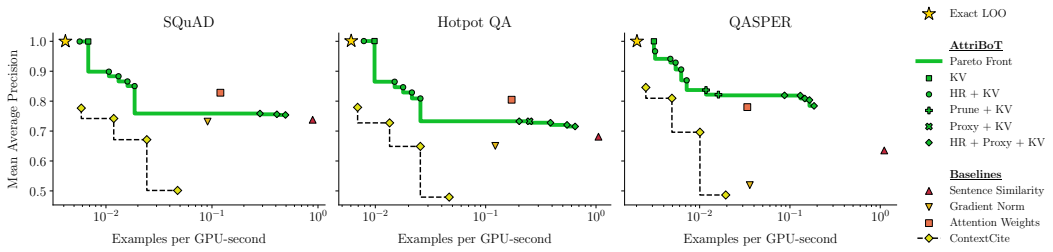

Figure 9: Using Qwen 2.5 72B Instruct as the target model and Mistral 7B Instruct v0.3 as the proxy model.

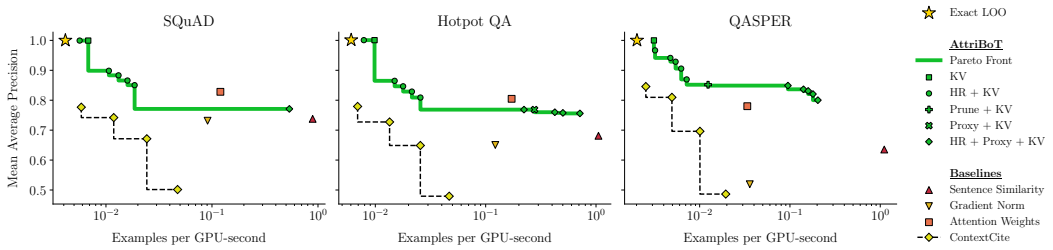

Figure 10: Using Qwen 2.5 72B Instruct as the target model and Llama 3.1 8B Instruct as the proxy model.

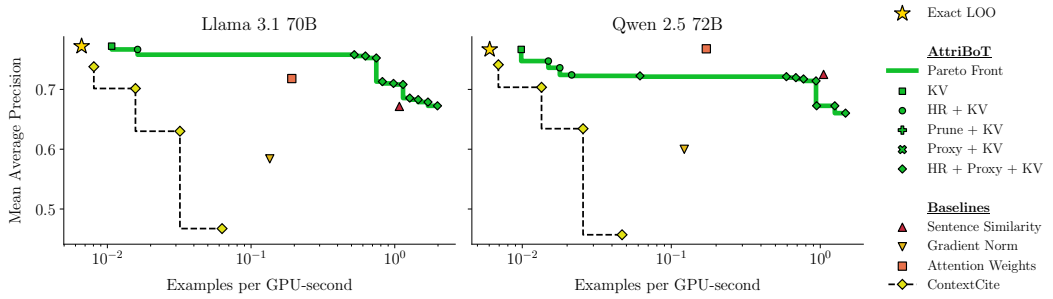

Figure 11: Mean average precision comparing with human annotation on HotpotQA dataset with Llama 3 family and Qwen 2.5 family.

D Practitioner's Guide to using AttriBoT

D.1 Method and Hyperparameter Selection

KV caching should always be employed because is effectively lossless and can be combined with other methods. Whenever feasible, float16 should be used to avoid possible floating point errors. To further reduce LOO costs, we recommend starting with hierarchical attribution and pruning. For tasks like open-domain QA where only a small subset of the context is relevant, retaining 3 to 5 paragraphs or sentences is generally sufficient to maintain faithfulness while achieving significant acceleration. For further speedup, consider using smaller models within the same model family or distilled versions of the target model. Models in the same family with sizes as small as 1B or 0.5B parameters can remain faithful even for complex tasks. Additionally, combining proxy methods with hierarchical attribution is advisable, as their performance degradation does not compound linearly.

D.2 Input preprocessing

In settings such as RAG and in-context learning, input information may be duplicated due to the retrieval process. Such duplication poses challenges for LOO context attribution or any attribution

method that aims to approximate the LOO error, as removing one instance of that information would likely not change the model's response likelihood. We strongly recommend deduplicating input before performing LOO context attribution.

In scenarios where input information is unstructured, we suggest chunking the input by parsing sentences and grouping a few (e.g., 10) sentences together to facilitate hierarchical context attribution.