CONTEXTCITE: Attributing Model Generation to Context

Benjamin Cohen-Wang [*1] **Harshay Shah** [*1] **Kristian Georgiev** [*1] **Aleksander Madry** [1]

Abstract

How do language models actually *use* information provided as context when generating a response? Can we infer whether a particular generated statement is actually grounded in the context, a misinterpretation, or fabricated? To help answer these questions, we introduce the problem of *context attribution*: pinpointing the parts of the context (if any) that *led* a model to generate a particular statement. We then present CONTEXTCITE, a simple and scalable context attribution method that can be applied on top of any existing language model.

1. Introduction

Suppose that we would like to use a language model to learn about recent news. We would first need to provide it with relevant articles as *context*. We would then expect the language model to interact with this context to answer questions. Upon seeing a generated response, we might ask: is everything accurate? Did the model misinterpet anything or make something up? Is the response actually *grounded* in the provided context?

Answering these questions manually might be tedious—we would need to first read the articles ourselves and then verify the statements. To automate this process, prior work has focused on teaching models to generate *citations*: references to parts of the context that *support* a response (Nakano et al., 2021; Menick et al., 2022; Thoppilan et al., 2022; Gao et al., 2022; 2023). They typically do so by explicitly training or prompting language models to produce citations.

In this work, we explore a different type of citation: instead of teaching a language model to cite its sources, can we directly identify the pieces of information that it actually *uses*? Specifically, we ask:

Can we pinpoint the parts of the context (if any) that <u>led</u> to a particular generated statement?

We refer to this problem as *context attribution*. Suppose, for example, that language model misinterprets a piece of information and makes an inaccurate statement. Context attribution would reveal the misinterpreted part of the context. On the other hand, suppose that a language model uses knowledge that it learned from pre-training to generate a statement, rather than the context. In this case, context attribution would indicate this by not attributing the statement to any sources. Unlike citations generated by language models, which can be difficult to validate (Rashkin et al., 2023; Liu et al., 2023), it is easy to evaluate the efficacy of context attributions. Specifically, if a part of the context actually led to a particular generated response, then removing it should substantially affect this response.

1.1. Our contributions

We begin this work by formalizing the task of *context attribution* (Section 2). We then present CONTEXTCITE, a simple and scalable method for context attribution that can be applied on top of any existing language model (see Figure 1). Specifically, we propose to learn a *surrogate model* that approximates how a language model's response is affected by including or excluding different parts of the context (Section 3). We find that it is possible to learn a *linear* surrogate model that (1) faithfully models the language model's behavior and (2) can be efficiently computed using a small number of inference passes. Because the surrogate model is linear, its weights can be used to attribute the generated statement. We benchmark CONTEXTCITE against natural baselines on a diverse set of generation tasks and find that it is indeed effective at identifying the parts of the context responsible for a given generated response (Section 4).

2. Problem statement

In this section, we will introduce the problem of context attribution (Section 2.1) and define metrics for evaluating context attribution methods (Section 2.2). Here, we will consider attributing an entire generated response—we extend this to attributing specific statements in Appendix C.4.

Setup. Suppose that we use a language model to generate a response to a particular query given a context. First, let p_{LM} be an *autoregressive* language model: a model that defines a probability distribution over the next token given a sequence

[*]Equal contribution [1]Massachusetts Institute of Technology. Correspondence to: Benjamin Cohen-Wang <bencw@mit.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

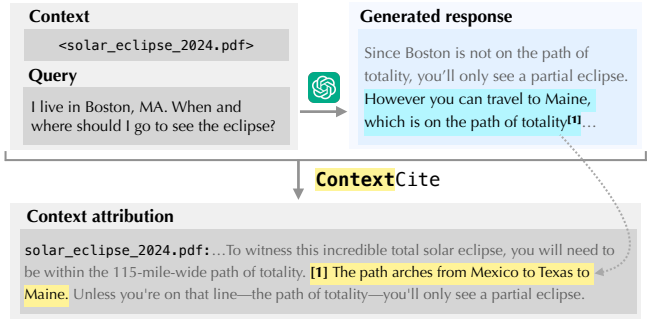


Figure 1. **CONTEXTCITE.** Our context attribution method, CONTEXTCITE, traces any specified generated statement back to the parts of the context that are responsible for it.

of preceeding tokens. We write $p_{\text{LM}}(t_i \mid t_1, \ldots, t_{i-1})$ to denote the probability of the next token being t_i given the preceeding tokens $t_1, \ldots, t_{i-1}$. Next, let C be a context consisting of tokens $c_1, \ldots, c_{\ell_C}$ and Q be a query consisting of tokens $q_1, \ldots, q_{\ell_Q}$. We generate a response R consisting of tokens $r_1, \ldots, r_{\ell_R}$ by sampling from the model conditioned on the context and query. More formally, we generate the ith token r_i of the response as follows:

$$r_i \sim p_{\text{LM}}(\cdot \mid c_1, \ldots, c_{\ell_C}, q_1, \ldots, q_{\ell_Q}, r_1, \ldots, r_{i-1})^1.$$

We write $p_{\text{LM}}(R \mid C, Q)$ to denote the probability of generating the entire response R—the product of the probabilities of generating its individual tokens—given the tokens of a context C and the tokens of a query Q.

2.1. Context attribution

The goal of context attribution is to attribute a generated response back to specific pieces of the context. We refer to these "pieces of the context" as *sources*–each source is just a subset of the tokens in the context. For example, each source might be a document, paragraph, sentence, or even a word. The choice of granularity depends on the application—in this work, we primarily focus on *sentences* as sources and use an off-the-shelf sentence tokenizer to partition the context into sources.

A *context attribution method* τ accepts a list of d sources $s_1, \ldots, s_d$ and assigns a score to each source indicating its "importance" to the response. We formalize this task in the following definition:

Definition 2.1 (*Context attribution*)**.** Suppose that we are given a context C with sources $s_1, \ldots, s_d \in \mathcal{S}$, a query Q, a language model p_{LM} and a generated response R. A *context attribution method* $\tau(s_1, \ldots, s_d)$ is a function $\tau : \mathcal{S}^d \to \mathbb{R}^d$ that assigns a score to each of the d sources. Each score is intended to signify the "importance" of the source to generating the response R.

What do context attribution scores signify? So far, we have only stated that scores should signify how "important" a source is for generating a particular statement. But what does this actually mean? There are two types of attribution that we might be interested in: *contributive* and *corroborative* (Worledge et al., 2023). *Contributive* attribution identifies the sources that *cause* a model to generate a statement. Meanwhile, *corroborative* attribution identifies sources that support or imply a statement. There are several existing methods for corroborative attribution of language models (Nakano et al., 2021; Menick et al., 2022; Gao et al., 2023). These typically involve explicitly training or prompting models to produce citations along with each statement they make.

In this work, we study *contributive* attribution. These attributions give rise to a diverse and distinct set of use cases and applications compared to corroborative attributions. Specifically, if a statement is accurate, then its corroborative and contributive sources may very well be the same. However, if a statement is inaccurate, corroborative and contributive attribution methods would likely behave differently. Indeed, suppose that a model misinterprets a fact in the context. A corroborative method might not find any attributions (because nothing in the context supports its statement). On the other hand, a contributive method would identify the fact that the model misinterpreted.

2.2. Evaluating the quality of context attributions

How might we evaluate the quality of a (contributive) context attribution method? Intuitively, if a source is important, then removing this source should change the response significantly. We introduce the *top-k log-probability drop* as a metric capturing this intuition. Specifically, it measures the effect of excluding the highest-scoring sources on the probability of generating the original response.

To formalize this metric, we first define a *context ablation* as a modification of the context that excludes certain sources. The precise details of a context ablation depend on the nature of the sources—when the sources are sentences, we can directly omit the specified sentences from the context. We write $\textsc{Ablate}(C, v)$ to denote a context C ablated according to a vector $v \in \{0, 1\}^d$ (with zeros specifying the sources to exclude). We are now ready to define the *top-k log-probability drop*:

Definition 2.2 (*Top-k log-probability drop*)**.** Suppose that we are given a context attribution method τ. Let $v_{\text{top-}k}(\tau)$ be an ablation vector that excludes the k highest-scoring sources according to τ and let

$$C' = \textsc{Ablate}(C, v_{\text{top-}k}(\tau))$$

[1] In practice, we may include additional tokens, e.g., to specify the beginning and end of a user's message.

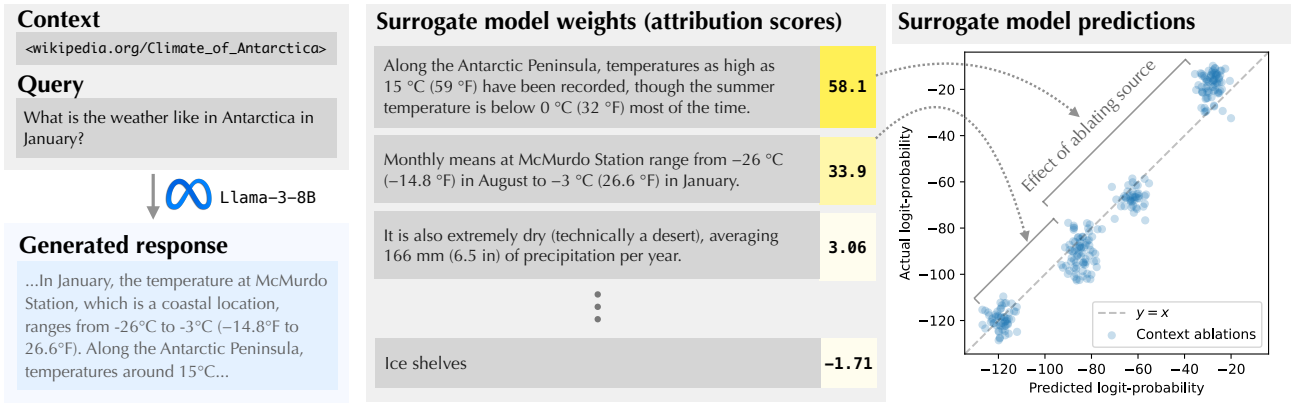


Figure 2. **An example of the *linear* surrogate model used by CONTEXTCITE.** On the left, we consider a context, query, and response about the weather in Antarctica. In the middle, we list the weights of a linear surrogate model that estimates the logit-scaled probability of the response as a function of the context ablation vector (2); CONTEXTCITE casts these weights as attribution scores. On the right, we plot the surrogate model's predictions against the actual logit-scaled probabilities for random context ablations. Two sources appear to be primarily responsible for the response, resulting in four "clusters" corresponding to whether each of these sources is included or excluded. These sources appear to interact *linearly*—the effect of removing both sources is close to the added effects of removing each source individually. As a result, the linear surrogate model is quite faithful.

be the context ablated according to $v_{\text{top-}k}(\tau)$. Then the *top-k log-probability drop* is defined as

$$\text{Top-}k\text{-drop}(\tau) = \underbrace{\log p_{\text{LM}}(R \mid C, Q)}_{\text{original log-}p} - \underbrace{\log p_{\text{LM}}(R \mid C', Q)}_{\text{log-}p\text{ without top sources}}. \tag{1}$$

The top-k log-probability drop is a useful metric for *comparing* methods for context attribution. In particular, if removing the highest-scoring sources of one attribution method causes a larger drop than removing those of another, then we consider the former method to be identifying sources that are more important (in the contributive sense). For a more fine-grained evaluation, in Appendix D.3 we also consider whether attribution scores can accurately *rank* the effects of ablating different sets of sources on the log-probability of the response.

3. Context attribution with CONTEXTCITE

In the previous section, we established that a context attribution method is effective insofar as it can predict the effect of including or excluding certain sources. In other words, given an ablation vector v, a context attribution method should inform how the probability of the original response,

$$f(v) := p_{\text{LM}}(R \mid \textsc{Ablate}(C, v), Q), \tag{2}$$

changes as a function of v. The design of CONTEXTCITE is driven by the following question: can we find a simple *surrogate model* $\hat{f}$ that approximates f well? If so, we could use the surrogate model $\hat{f}$ to understand how including or excluding subsets of sources would affect the probability of the original response (assuming that $\hat{f}$ is simple enough). Indeed, surrogate models have previously been used in this way to attribute predictions to training examples (Ilyas et al., 2022; Park et al., 2023b; Nguyen & Wong, 2023; Chang & Jia, 2022), model internals (Shah et al., 2024; Kramár et al., 2024), and input features (Ribeiro et al., 2016; Lundberg & Lee, 2017; Sokol et al., 2019). At a high-level, our approach consists of the following steps:

Step 1: Sample a "training dataset" of ablation vectors $v_1, \ldots, v_n$ and compute $f(v_i)$ for each v_i.

Step 2: Learn $\hat{f} : \{0,1\}^d \to \mathbb{R}$ that approximates f by training on the pairs $(v_i, f(v_i))$.

Step 3: Attribute the behavior of $\hat{f}$ to individual sources.

For the surrogate model $\hat{f}$ to be useful, it should (1) faithfully model f, (2) be efficient to compute, and (3) offer a method for attributing its outputs to the individual sources. To satisfy these desiderata, we find the following design choices to be effective:

- **Predict *logit-scaled* probabilities**: Fitting a regression model to predict probabilities directly might be problematic because probabilities are bounded in $[0, 1]$. The logit function ($\sigma^{-1}(p) = \log \frac{p}{1-p}$) is a mapping from $[0, 1]$ to $(-\infty, \infty)$, making logit-probability a more natural target for regression.
- **Learn a *linear* surrogate model**: Despite their simplicity, we find that linear surrogate models are often quite faithful. With a linear surrogate model, each weight signifies

the effect of ablating a source on the output. As a result, we can directly cast the weights of the surrogate model as attribution scores. We illustrate an example depicting the effectiveness of a linear surrogate model in Figure 2 and provide additional randomly sampled examples in Appendix D.1.

- **Learn a *sparse* linear surrogate model**: We find that a generated statement can often be explained well by just a handful of sources. Motivated by this observation, we induce sparsity in the surrogate model via LASSO (Tibshirani, 1994). This enables learning a faithful linear surrogate model even with a small number of ablations (see Appendix D.2 for details). For example, the surrogate model in Figure 2 uses just 32 ablations even though the context consists of 98 sources (in this case, sentences).
- **Sample ablation vectors uniformly**: To create the surrogate model's training dataset, we sample ablation vectors uniformly from the possible subsets of the sources in the context.

We summarize the resulting method, CONTEXTCITE, in Algorithm 1. See Figure 2 for an example of CONTEXTCITE attributions for a response generated by `Llama-3-8B` (AI, 2024).

Algorithm 1 CONTEXTCITE

Input: Autoregressive language model p_{LM}, context C consisting of d sources $s_1, \ldots, s_d$, query Q, response R, number of ablations n, regularization parameter λ
Output: Attribution scores $\hat{w} \in \mathbb{R}^d$
$f(v) \coloneqq p_{\text{LM}}(R \mid \textsc{Ablate}(C, v), Q)$ ▷ Ablated prob.
$g(v) \coloneqq \sigma^{-1}(f(v))$ ▷ Logit-scaled version of f
for $i \in \{1, \ldots, t\}$ **do**
  Sample an ablation vector v_i uniformly from $\{0, 1\}^d$
  $y_i \leftarrow g(v_i)$
end for
$\hat{w}, \hat{b} \leftarrow \textsc{Lasso}(\{(v_i, y_i)\}_{i=1}^n, \lambda)$
return $\hat{w}$

4. Evaluating CONTEXTCITE

In this section, we evaluate whether CONTEXTCITE can effectively identify sources that cause the language model to generate a particular response. Specifically, we use measyre the top-k log-probability drop (1) to benchmark CONTEXTCITE against a few natural baselines. See Appendix C.6 for the exact setup.

Datasets and models. Generation tasks can differ in terms of (1) context properties (e.g., length, complexity) and (2) how the model uses in-context information to generate a response (e.g., summarization, question answering). We evaluate CONTEXTCITE using three representative benchmarks:

1. *TyDi QA* (Clark et al., 2020): a question-answering dataset in which the context is an entire Wikipedia article.
2. *Hotpot QA* (Yang et al., 2018): a *multi-hop* question-answering dataset where answering the question requires combining information from multiple documents.
3. *CNN DailyMail* (Nallapati et al., 2016): a news article summarization dataset.

We use instruction-tuned versions of `Llama-3-8B` (AI, 2024) and `Phi-3-mini` (Abdin et al., 2024), and evaluate on up to $1,000$ randomly-sampled validation examples from each benchmark.

Baselines. We consider three natural baselines adapted from prior work on model explanations. We defer details and additional baselines that we found to be less effective to Appendix C.6.1.

1. *Attention*: A line of work on explaining language models leverages attention weights (Lee et al., 2017; Ding et al., 2017; Serrano & Smith, 2019; Jain & Wallace, 2019; Wiegreffe & Pinter, 2019; Abnar & Zuidema, 2020). We use a simple but effective baseline that computes an attribution score for each source by summing the average attention weight of individual tokens in the source across all heads in all layers.
2. *Gradient norm*: Other explanation methods rely on input gradients (Simonyan et al., 2013; Li et al., 2015; Smilkov et al., 2017). Here, following Yin & Neubig (2022), we estimate the attribution score of each source by computing the ℓ_1-norm of the log-probability gradient of the response with respect to the embeddings of tokens in the source.
3. *Semantic similarity*: Finally, we consider attributions based on semantic similarity. We employ a pre-trained sentence embedding model (Reimers & Gurevych, 2019) to embed each source and the generated statement. We treat the cosine similarities between these as attribution scores.

Experiment setup. Each example on which we evaluate consists of a context, a query, a language model, and a generated response. As discussed in Appendix C.4, rather than attributing the entire response to the context, we consider attributing individual *statements* in the response to the context. Specifically, given an example, we (1) split the response into sentences using an off-the-shelf tokenizer (Bird

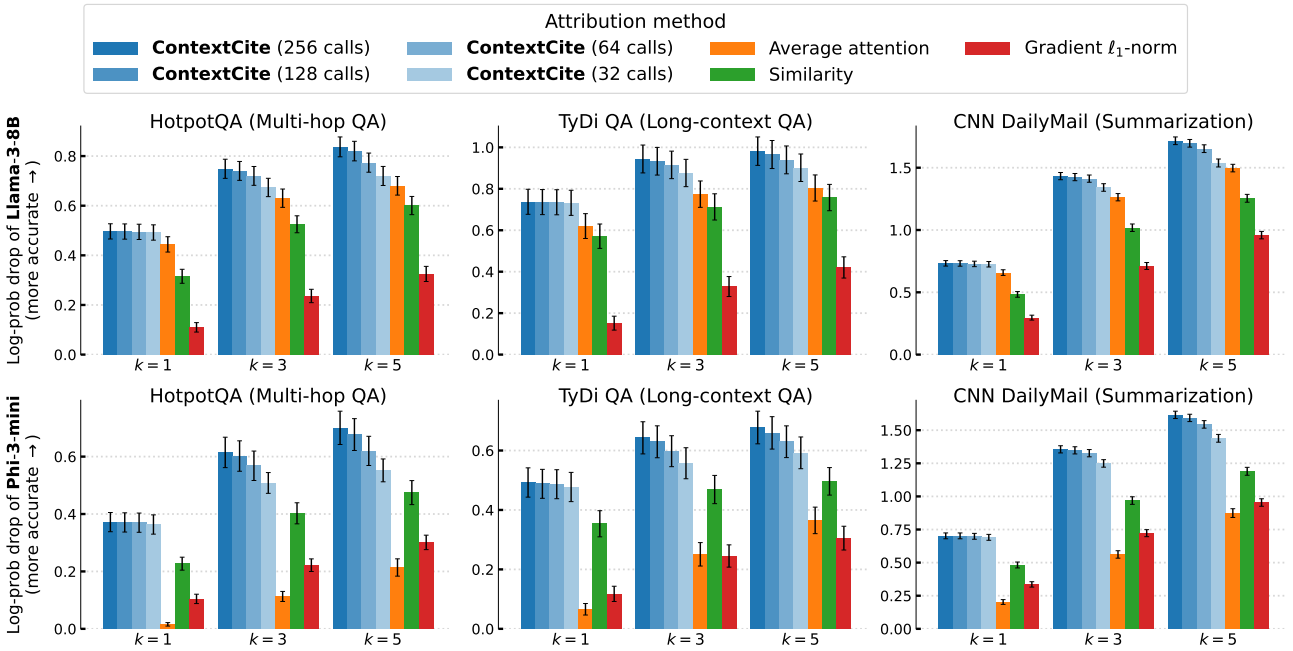


Figure 3. We report the top-k log-probability drop (1), which measures the effect of ablating top-scoring sources on the generated response. A higher drop indicates that the context attribution method identifies more relevant sources. We evaluate attributions of responses generated by `Llama-3-8B` and `Phi-3-mini` on up to $1,000$ randomly sampled validation examples from each of three benchmarks. We find that CONTEXTCITE using just 32 context ablations consistently outperforms the baselines—attention, gradient norm, and semantic similarity—across benchmarks and models. Increasing the number of context ablations to $\{64, 128, 256\}$ can further improve the quality of CONTEXTCITE attributions.

et al., 2009), and (2) compute attribution scores for each sentence. Then, to evaluate the attribution scores, we measure the top-k log-probability drop for $k = \{1, 3, 5\}$ (1) for each sentence separately, and then average performances across sentences. Our experiments perform this evaluation for every combination of context attribution method, dataset, and language model. We evaluate CONTEXTCITE with $\{32, 64, 128, 256\}$ context ablations.

Results. In Figure 3, we find that CONTEXTCITE consistently outperforms baselines, even when we only use 32 context ablations to compute its surrogate model. While the attention baseline approaches the performance of CONTEXTCITE with `Llama-3-8B`, it fares quite poorly with `Phi-3-mini` suggesting that attention is not consistently reliable for context attribution. We report results for additional metrics, models, datasets, and baselines in Appendices D.3 and D.4.

5. Discussion

We introduce the problem of *context attribution*, where the goal is to trace responses generated by language models back to the specific parts of the context. Our proposed method, CONTEXTCITE, uses linear surrogate models to accurately attribute statements generated by any language model in a scalable and sample-efficient manner.

In Appendix A, we showcase the utility of CONTEXTCITE through two case studies:

1. *Verifying generated statements* (Appendix A.1): We hypothesize that if attributed sources do not also *support* a generated statement, then it is less likely to be accurate. We use CONTEXTCITE in conjunction with an off-the-shelf textual entailment classifier to verify statements in this way. We find that a statement being supported by its sources is indeed a strong indicator of its accuracy.

2. *Improving response quality by selecting query-relevant information from the context* (Appendix A.2): Language models often struggle to correctly use information hidden within long contexts (Liu et al., 2024; Peysakhovich & Lerer, 2023). We use CONTEXTCITE to extract the parts of the context relevant for a particular query, and then use this "pruned" context to regenerate the response. We find that doing so improves question answering performance on multiple benchmarks.

We provide related work in Appendix B.

References

Abdin, M., Jacobs, S. A., Awan, A. A., Aneja, J., Awadallah, A., Awadalla, H., Bach, N., Bahree, A., Bakhtiari, A., Behl, H., et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024.

Abnar, S. and Zuidema, W. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Aher, G. V., Arriaga, R. I., and Kalai, A. T. Using large language models to simulate multiple humans and replicate human subject studies. In *International Conference on Machine Learning*, pp. 337–371. PMLR, 2023.

AI, M. Introducing meta llama 3: The most capable openly available llm to date, 2024. URL https://ai.meta.com/blog/meta-llama-3/.

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

Bohnet, B., Tran, V. Q., Verga, P., Aharoni, R., Andor, D., Soares, L. B., Eisenstein, J., Ganchev, K., Herzig, J., Hui, K., et al. Attributed question answering: Evaluation and modeling for attributed large language models. In *Arxiv preprint arXiv:2212.08037*, 2022.

Chang, T.-Y. and Jia, R. Data curation alone can stabilize in-context learning. *arXiv preprint arXiv:2212.10378*, 2022.

Chen, J., Kim, G., Sriram, A., Durrett, G., and Choi, E. Complex claim verification with evidence retrieved in the wild. *arXiv preprint arXiv:2305.11859*, 2023.

Chern, I., Chern, S., Chen, S., Yuan, W., Feng, K., Zhou, C., He, J., Neubig, G., Liu, P., et al. Factool: Factuality detection in generative ai–a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*, 2023.

Clark, J. H., Choi, E., Collins, M., Garrette, D., Kwiatkowski, T., Nikolaev, V., and Palomaki, J. Tydi qa: A benchmark for information-seeking question answering in ty pologically di verse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020.

Ding, Y., Liu, Y., Luan, H., and Sun, M. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1150–1159, 2017.

Enguehard, J. Sequential integrated gradients: a simple but effective method for explaining language models. *arXiv preprint arXiv:2305.15853*, 2023.

Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., Zhao, V. Y., Lao, N., Lee, H., Juan, D.-C., et al. Rarr: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*, 2022.

Gao, T., Yen, H., Yu, J., and Chen, D. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*, 2023.

Grosse, R., Bae, J., Anil, C., Elhage, N., Tamkin, A., Tajdini, A., Steiner, B., Li, D., Durmus, E., Perez, E., et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*, 2023.

Ilyas, A., Park, S. M., Engstrom, L., Leclerc, G., and Madry, A. Datamodels: Predicting predictions from training data. In *International Conference on Machine Learning (ICML)*, 2022.

Jain, S. and Wallace, B. C. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. d. l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Kalai, A. T. and Vempala, S. S. Calibrated language models must hallucinate. *arXiv preprint arXiv:2311.14648*, 2023.

Kokalj, E., Škrlj, B., Lavrač, N., Pollak, S., and Robnik-Šikonja, M. Bert meets shapley: Extending shap explanations to transformer-based classifiers. In *Proceedings of the EACL hackashop on news media content analysis and automated report generation*, pp. 16–21, 2021.

Kramár, J., Lieberum, T., Shah, R., and Nanda, N. Atp*: An efficient and scalable method for localizing llm behaviour to components. *arXiv preprint arXiv:2403.00745*, 2024.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Kwon, Y., Wu, E., Wu, K., and Zou, J. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. *arXiv preprint arXiv:2310.00902*, 2023.

Lee, J., Shin, J.-H., and Kim, J.-S. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 121–126, 2017.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Li, J., Chen, X., Hovy, E., and Jurafsky, D. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*, 2015.

Liu, N. F., Zhang, T., and Liang, P. Evaluating verifiability in generative search engines. *arXiv preprint arXiv:2304.09848*, 2023.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Lundberg, S. and Lee, S.-I. A unified approach to interpreting model predictions. In *Neural Information Processing Systems (NeurIPS)*, 2017.

Menick, J., Trebacz, M., Mikulik, V., Aslanides, J., Song, F., Chadwick, M., Glaese, M., Young, S., Campbell-Gillingham, L., Irving, G., et al. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*, 2022.

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.-t., Koh, P. W., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.

Mohammadi, B. Wait, it's all token noise? always has been: Interpreting llm behavior using shapley value. *arXiv preprint arXiv:2404.01332*, 2024.

Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Nallapati, R., Zhou, B., Gulcehre, C., Xiang, B., et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

Nguyen, T. and Wong, E. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*, 2023.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. Ms marco: A human-generated machine reading comprehension dataset, 2016. URL https://openreview.net/forum?id=Hk1iOLcle.

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–22, 2023a.

Park, S. M., Georgiev, K., Ilyas, A., Leclerc, G., and Madry, A. Trak: Attributing model behavior at scale. In *Arxiv preprint arXiv:2303.14186*, 2023b.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. volume 12, pp. 2825–2830, 2011.

Peysakhovich, A. and Lerer, A. Attention sorting combats recency bias in long context language models. *arXiv preprint arXiv:2310.01427*, 2023.

Rashkin, H., Nikolaev, V., Lamm, M., Aroyo, L., Collins, M., Das, D., Petrov, S., Tomar, G. S., Turc, I., and Reitter, D. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840, 2023.

Reimers, N. and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?": Explaining the predictions of any classifier. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. Design and analysis of computer experiments. In *Statistical Science*, volume 4, pp. 409–423. Institute of Mathematical Statistics, 1989. URL http://www.jstor.org/stable/2245858.

Serrano, S. and Smith, N. A. Is attention interpretable? *arXiv preprint arXiv:1906.03731*, 2019.

Shah, H., Ilyas, A., and Madry, A. Decomposing and editing model predictions. In *arXiv preprint*, 2024.

Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. SmoothGrad: removing noise by adding noise. In *ICML workshop on visualization for deep learning*, 2017.

Sokol, K., Hepburn, A., Santos-Rodriguez, R., and Flach, P. blimey: Surrogate prediction explanations beyond lime. In *Arxiv preprint arXiv:1910.13016*, 2019.

Spearman, C. The proof and measurement of association between two things. In *The American Journal of Psychology*, 1904.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. Lamda: Language models for dialog applications. In *ArXiv preprint arXiv:2201.08239*, 2022.

Tibshirani, R. Regression shrinkage and selection via the lasso. In *Journal of the Royal Statistical Society, Series B*, 1994.

Wainwright, M. J. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Wang, R. E., Wirawarn, P., Khattab, O., Goodman, N., and Demszky, D. Backtracing: Retrieving the cause of the query. *arXiv preprint arXiv:2403.03956*, 2024.

Wiegreffe, S. and Pinter, Y. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019.

Williams, A., Nangia, N., and Bowman, S. R. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Worledge, T., Shen, J. H., Meister, N., Winston, C., and Guestrin, C. Unifying corroborative and contributive attributions in large language models. *arXiv preprint arXiv:2311.12233*, 2023.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., and Manning, C. D. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

Yin, K. and Neubig, G. Interpreting language models with contrastive explanations. *arXiv preprint arXiv:2202.10419*, 2022.

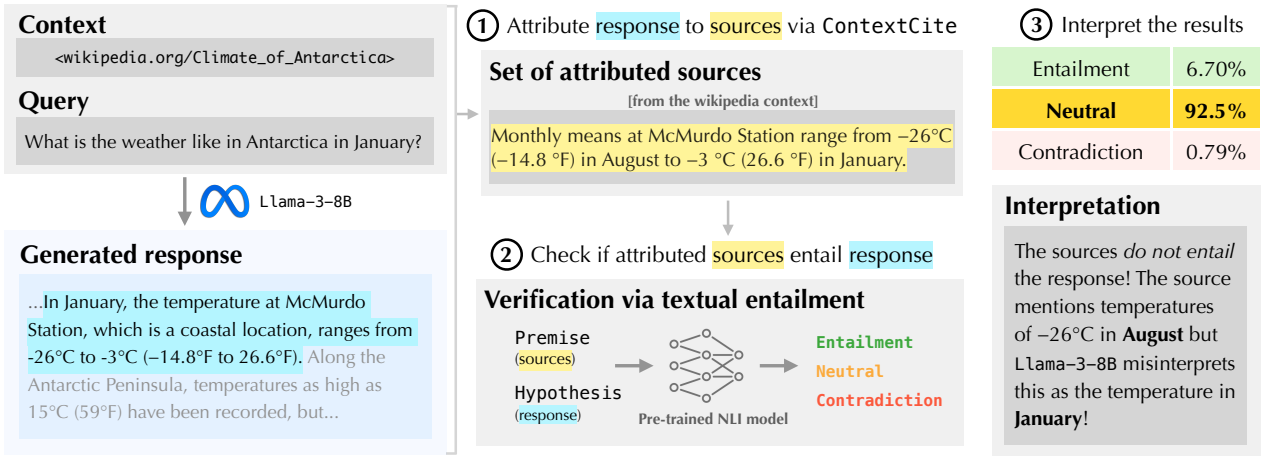


Figure 4. **Verifying generated statements using CONTEXTCITE and textual entailment**. On the left, we consider a context, query, and response about the weather in Antarctica. The response claims that the temperature in Antarctica ranges from -26 to -3 degrees Celsius in January. We (1) attribute this claim to sources in the context using CONTEXTCITE and (2) check whether these sources entail the response using a textual entailment classifier. We (3) find that the sources actually do *not* entail the response. In this case, the temperature is actually -26 degrees in *August* but the model inteprets this as being part of the range for *January*.

A. Applications of CONTEXTCITE

In Section 4, we found that CONTEXTCITE is an effective (contributive) context attribution method. In other words, it identifies the sources in the context that *cause* the model to generate a particular statement. In this section, we present two applications of context attribution: verifying generated statements (Appendix A.1) and improving response quality by extracting query-relevant information from the context (Appendix A.2).

A.1. Verifying generated statements

It can be difficult to know when to *trust* statements generated by language models (Huang et al., 2023; Chen et al., 2023; Chern et al., 2023; Min et al., 2023; Kalai & Vempala, 2023). We investigate whether CONTEXTCITE can help in verifying the accuracy of generated statements. See Appendix C.7 for the exact setup.

Approach. Our approach builds on the following intuition: if the sources identified by CONTEXTCITE for a particular statement do not *support* it, then the statement might be inaccurate. To operationalize this, we (1) use CONTEXTCITE to identify a set of relevant sources and (2) check whether these sources support the statement with an off-the-shelf textual entailment classifier (Lewis et al., 2019). This classifier takes as input a *premise* (the set of attributed sources) and a *hypothesis* (the statement) and predicts whether the premise entails, contradicts, or is neutral with respect to the hypothesis. Based on the prediction, we mark the statement as *verified* if the sources entail it, a *potential misinterpretation* if they contradict it, and *unverified* if the model predicts "neutral" or if the set of sources is empty.

Experiments. We apply our verification pipeline to a statement generated by `Llama-3-8B` in Figure 4. In this case, our pipeline surfaces a specific part of the context that the model misinterpreted. To quantitiatively evaluate these verifications, we consider the CNN DailyMail benchmark consisting of news articles to be summarized (Nallapati et al., 2016). We ask `Llama-3-8B` to summarize each of $1,000$ randomly sampled articles. We partition each generated summary into sentences and run our verification pipeline on each sentence. Following Park et al. (2023a); Aher et al. (2023), we ask GPT-4 (Achiam et al., 2023) whether each sentence is accurate as a proxy for human annotations. Among statements marked as verified, just 2.4% are inaccurate according to GPT-4. For statements marked as unverified, 5.1% are inaccurate. Finally, for statements marked as misinterpretations, 30.7% are inaccurate. This suggests that the verifications produced by CONTEXTCITE can indeed be used to help identify inaccurate statements.

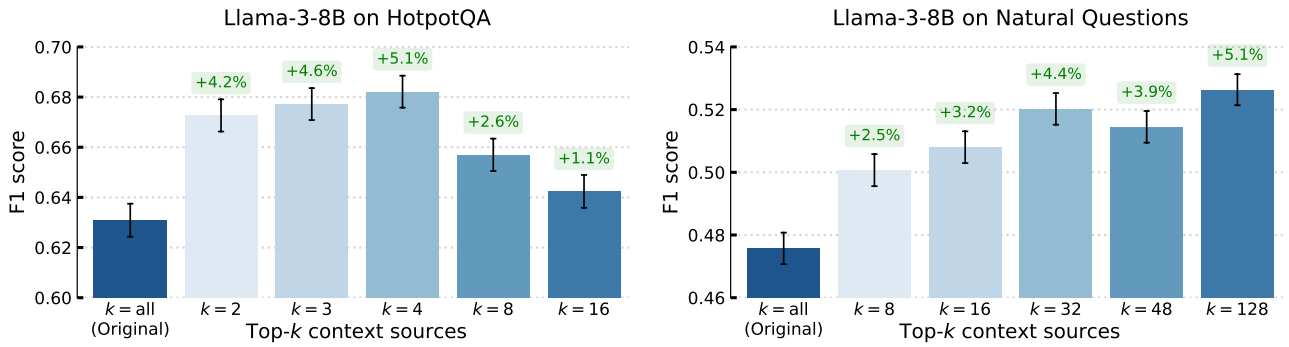


Figure 5. **Improving response quality by constructing query-specific contexts.** On the left, we show that filtering contexts by selecting the top-$\{2, \ldots, 16\}$ query-relevant sources (via CONTEXTCITE) improves the average F_1-score of `Llama-3-8B` on $1,000$ randomly sampled examples from the Hotpot QA dataset. Similarly, on the right, simply replacing the entire context with the top-$\{8, \ldots, 128\}$ query-relevant sources boosts the average F_1-score of `Llama-3-8B` on $1,000$ randomly sampled examples from the Natural Questions dataset. In both cases, CONTEXTCITE improves response quality by extracting the most query-relevant information from the context.

A.2. Improving response quality by extracting query-relevant information from the context

Our second application is motivated by the observation that language models often struggle to correctly use relevant in-context information hidden within long contexts (Liu et al., 2024; Peysakhovich & Lerer, 2023). For example, Liu et al. (2024) show that relevant in-context information can be "lost in the middle", i.e., performance is heavily influenced by the *location* of relevant information within the context.

Approach. To mitigate this issue, we leverage CONTEXTCITE to produce a "query-specific" context that contains only the information relevant for a particular query. This process consists of three steps: (1) generate a response using the entire context, (2) use CONTEXTCITE to compute attribution scores for sources in the context, and (3) construct a query-specific context using only the top-k sources, which can be used to regenerate a response.

Experiments. We assess the effectiveness of this approach on two question-answering datasets: HotpotQA (Yang et al., 2018) and Natural Questions (Kwiatkowski et al., 2019). In both datasets, the provided context typically includes a lot of irrelevant information in addition to the answer to the question. In Figure 5, we report the average F_1-score of `Llama-3-8B` on $1,000$ randomly sampled examples from each dataset (1) when it is provided with the entire context and (2) when it is provided with only the top-k sources according to CONTEXTCITE. We find that simply selecting the most relevant sources can consistently improve question answering capabilities. See Appendix C.8 for the exact setup and Appendix D.5 for additional experiments with `Llama-3-70B`.

B. Related Work

Citations for language models. Prior work on citations for language models has focused on *teaching* models to generate responses with citations (Nakano et al., 2021; Gao et al., 2022; Menick et al., 2022; Thoppilan et al., 2022; Gao et al., 2023). These citations are intended to be *corroborative* (Worledge et al., 2023) in nature; citations are evaluated on whether they *support* or imply a generated statement (Bohnet et al., 2022; Rashkin et al., 2023; Liu et al., 2023; Wang et al., 2024). In contrast, CONTEXTCITE—a *contributive* attribution method—identifies sources that *cause* a language model to generate a given response.

Explaining language models. Related to context attribution is the (more general) problem of explaining language model behavior. Methods for explaining language models have used attention weights (Wiegreffe & Pinter, 2019; Abnar & Zuidema, 2020), similarity metrics (Reimers & Gurevych, 2019) and input gradients (Yin & Neubig, 2022; Enguehard, 2023), which we adapt as baselines. The explanation approaches that are closest in spirit to CONTEXTCITE are ablation-based methods such as LIME (Ribeiro et al., 2016) and methods relying on the Shapley value (Lundberg & Lee, 2017; Kokalj et al., 2021; Mohammadi, 2024).

Understanding model behavior via surrogate modeling. Several prior works use *surrogate modeling* (Sacks et al., 1989) to study different aspects of model behavior. For example, data attribution methods use linear surrogate models to trace model predictions back to individual training examples (Ilyas et al., 2022; Park et al., 2023b; Grosse et al., 2023; Kwon et al., 2023) or in-context learning examples (Nguyen & Wong, 2023; Chang & Jia, 2022). Similarly, methods for identifying input features that drive a model prediction (Ribeiro et al., 2016; Lundberg & Lee, 2017; Sokol et al., 2019) or for attributing predictions back to internal model components (Shah et al., 2024; Kramár et al., 2024) have also leveraged surrogate modeling.

C. Experiment details

C.1. Implementation details

We run all experiments on a cluster of A100 GPUs. We use the `scikit-learn` (Pedregosa et al., 2011) implementation of LASSO for CONTEXTCITE, always with the regularization parameter `alpha` set to `0.01`.

C.2. Models

The language models we consider in this work are `Llama-3-{8/70}B` (AI, 2024), `Mistral-7B` (Jiang et al., 2023) and `phi-3-mini` (Abdin et al., 2024). We use instruction-tuned variants of these models. We use the implementations of language models from HuggingFace's `transformers` library (Wolf et al., 2020). Specifically, we use `meta-llama/Meta-Llama-3-{8/70}B-Instruct`, `mistralai/Mistral-7B-Instruct-v0.2`, and `microsoft/Phi-3-mini-128k-instruct`. When generating responses with these models, we use their standard chat templates, treating the prompt formed from the context and query as a user's message.

C.3. Datasets

We consider a variety of datasets to evaluate CONTEXTCITE spanning question answering and summarization tasks and different context structures and lengths. We provide details about these datasets and preprocessing steps in this section. Some of the datasets, namely Natural Questions and TyDi QA, contain contexts that are longer than the maximum context window of the models we consider. In particular, `Llama-3-8B` has the shortest context window of $8,192$ tokens. When evaluating, we filter datasets to include only examples that fit within this context window (with a padding of 512 tokens for the response).

CNN DailyMail (Nallapati et al., 2016) is a news summarization dataset. The contexts consists of a news article and the query asks the language model to briefly summarize the articles in up to three sentences. We use the following prompt template:

```
Context: {context}

Query: Please summarize the article in up to three sentences.
```

Hotpot QA. (Yang et al., 2018) is a *multi-hop* question-answering dataset in which the context consists of multiple short documents. Answering the question requires combining information from a subset of these documents—the rest are "distractors" containing information that is only seemingly relevant. We use the following prompt template:

```
Title: {title_1}
Content: {document_1}
...
Title: {title_n}
Content: {document_n}

Query: {question}
```

MS MARCO (Nguyen et al., 2016) is question-answering dataset in which the question is a Bing search query and the context is a passage from a retrieved web page that can be used to answer the question. We use the following prompt template:

```
Context: {context}

Query: {question}
```

Natural Questions (Kwiatkowski et al., 2019) is a question-answering dataset in which the questions are Google search

queries and the context is a Wikipedia article. The context is provided as raw HTML; we include only paragraphs (text within `<p>` tags) and headers (text within `<h[1-6]>` tags) and provide these as context. We filter the dataset to include only examples where the question can be answered just using the article. We use the same prompt template as MS MARCO.

TyDi QA (Clark et al., 2020) is a multilingual question-answering dataset. The context is a Wikipedia article and the question about the topic of the article. We filter the dataset to include only English examples and consider only examples where the question can be answered just using the article. We use the same prompt template as MS MARCO.

C.3.1. DATASET STATISTICS.

In Table 1, we provide the average and maximum numbers of sources in the datasets that we consider.

Dataset	Average number of sources	Maximum number of sources
MS MARCO	36.0	95
Hotpot QA	42.0	94
Natural Questions	103.3	353
TyDi QA	165.8	872
CNN DailyMail	32.4	172

Table 1. The average and maximum numbers of sources (in this case, sentences) among the up to $1,000$ randomly sampled examples from each of the datasets we consider.

C.3.2. PARTITIONING CONTEXTS INTO SOURCES AND ABLATING CONTEXTS

In this section, we discuss how we partition contexts into sources and perform context ablations. For every dataset besides Hotpot QA, we use an off-the-shelf sentence tokenizer (Bird et al., 2009) to partition the context into sentences. To perform a context ablation, we concatenate all of the included sentences and provide the resulting string to the language as context. The Hotpot QA context consists of multiple documents, each of which includes annotations for individual sentences. Furthermore, the documents have titles, which we include in the prompt (see Appendix C.3). Here, we still treat sentences as sources and include the title of a document as part of the prompt if at least one of the sentences of this document is included.

C.4. Attributing selected statements from the response

In Section 2, we discussed attributing an *entire* generated response. In practice (and in our experiments), we might be interested in attributing a particular statement, e.g., a sentence or phrase. We define a *statement* to be any contiguous selection of tokens $r_i, \ldots, r_j$ from the response. To extend our setup to attributing specific statements, we let a context attribution method τ accept an additional argument (i, j) specifying the start and end indices of the statement to attribute. Instead of considering the probability of generating the *entire* original response, we consider the probability of generating the selected statement. Formally, in the definitions in Section 2, we replace $p_{\text{LM}}(R \mid C, Q)$ with

$$p_{\text{LM}}(\underbrace{r_i, \ \ldots \ , r_j}_{\text{statement to attribute}} \mid C, Q, \underbrace{r_1, \ \ldots \ , r_{i-1}}_{\text{response so far}}).$$

C.5. Learning a *sparse* linear surrogate model

In Figure 9, we illustrate that CONTEXTCITE can learn a faithful surrogate model with a small number of ablations by exploiting underlying sparsity. Specifically, we consider CNN DailyMail and Natural Questions. For $1,000$ randomly sampled validation examples for each dataset, we generate a response with `Llama-3-8B` using the prompt templates in Appendix C.3. Following the discussion in Appendix C.4, we split each response into sentences and consider each of these sentences to be a "statement." For the experiment in Figure 9a, for each statement, we ablate each of the sources individually and consider the source to be relevant if this ablation changes the probability of the statement by a factor of at least $\delta = 2$. For the experiment in Figure 9b, we report the average root mean squared error (RMSE) over these statements for surrogate models trained using different numbers of context ablations. See Appendices C.2 and C.3 for additional details on datasets and models.

C.6. Evaluating CONTEXTCITE

See Appendices C.1 to C.3 for details on implementation, datasets and models for our evaluations.

C.6.1. BASELINES FOR CONTEXT ATTRIBUTION

We provide a detailed list of baselines for context attribution in this section. In addition to the baselines described in Section 4, we consider additional attention-based and gradient-based baselines. We provide evaluation results including these baselines in Appendix D.4.

1. *Average attention*: We compute average attention weights across heads and layers of the model. We compute the sum of these average weights between every token of a source and every token of the generated statement to attribute as an attribution score. This is the attention-based baseline that we present in Figure 3.

2. *Attention rollout*: We consider the more sophisticated attention-based explanation method of Abnar & Zuidema (2020). Attention rollout seeks to capture the *propagated* influence of each token on each other token. Specifically, we first average the attention weights of the heads within each layer. Let $A_\ell \in \mathbb{R}^{n \times n}$ denote the average attention weights for the ℓ'th layer, where n is the length of the sequence. Then the propagated attention weights for the ℓ'th layer, which we denote $\tilde{A}_\ell \in \mathbb{R}^{n \times n}$, are defined recursively as $\tilde{A}_\ell = A_\ell \tilde{A}_{\ell-1}$ for $\ell > 1$ and $\tilde{A}_1 = A_1$. Attention rollout computes an "influence" of token j on token i by computing the product $(A_0 A_1 \cdots A_L)_{ij}$ where L is the total number of layers. When the model contains residual connections (as ours do), the average attention weights are replaced with $0.5 A_\ell + 0.5 I$ when propagating influences.

3. *Gradient norm*: Following Yin & Neubig (2022), in Section 4 we estimate the attribution score of each source by computing the ℓ_1-norm of the log-probability gradient of the response with respect to the embeddings of tokens in the source. In Appendix D.4, we also consider the ℓ_2-norm of these gradients, but find this to be slightly less effective.

4. *Gradient times input*: As an additional gradient-based baseline, we also consider taking the dot product of the gradients and the embeddings following Shrikumar et al. (2016) in Appendix D.4, but found this to be less effective than the gradient norm.

5. *Semantic similarity*: Finally, we consider attributions based on semantic similarity. We employ a pre-trained sentence embedding model (Reimers & Gurevych, 2019) to embed each source and the generated statement. We treat the cosine similarities between these as attribution scores.

C.7. Verifying generated statements

To verify generated statements, we first use CONTEXTCITE to identify a set of relevant sources. We then use the textual entailment classifier of Lewis et al. (2019), specifically, the large variant of BART fine-tuned on MultiNLI (Williams et al., 2017), to check whether these sources support the statement. We evaluate our verification pipeline on CNN DailyMail, but slightly modify the prompt from Appendix C.3. Specifically, we do not request a short summary and instead use the following prompt:

```
Context: {context}

Query: Please summarize the article.
```

Computing attribution sets. One simple approach for selecting a set of relevant sources using CONTEXTCITE would be to include the top-k scoring sources for some k. However, this approach may be sensitive to the choice of k and would perform poorly when a model relies on many sources to generate a response. Instead, we apply a threshold to the scores provided by CONTEXTCITE to obtain an attribution set.

In the case of CONTEXTCITE, the attribution scores are the weights of a surrogate model that estimates the logit-probability of the response as a function of the context ablation vector. Hence, if we pick a threshold t, we include sources that decrease the logit-probability of the response by at least t when they are excluded. When the probability of generating the original response is small (which is often the case, see, e.g., the right side of Figure 2), the logit transform behaves a lot like the

log transform. So, if we select a threshold of $t = \log(\lambda)$, this roughly corresponds to including sources that decrease the probability of the original response by a factor of at least λ when they are excluded. We use $\lambda = 100$ in this experiment.

Using the textual entailment classifier. A textual entailment classifier takes as input a premise and hypothesis and outputs probabilities for whether the premise entails, contradicts, or is neutral with respect to the hypothesis. As a post-processing step, we predict contradiction or neutral if the probabilities assigned to each of these choices exceeds 0.9 and entailment otherwise.

In our case, the hypothesis is simply the generated statement. The premise consists of the set of attributed sources. Specifically, we concatenate the sources in the attribution set in the order they appear in the context to form the premise. When sources are non-contiguous, we use ellipses to indicate that some sources have been omitted. For example, a premise might look like: $...s_1 s_2 ... s_5$.

Evaluating accuracy using GPT-4. As a proxy for human verification, we use GPT-4 (Achiam et al., 2023) to evaluate the accuracy of the generated statements. We use the `gpt-4o` model, setting the temperature to 0, the seed to 0, and providing it with the following prompt:

```
Context: {context}

Can we conclude that "{statement}"? Please respond with just yes or no.
```

C.8. Improving response quality by extracting query-relevant information from the context

Recall that in Appendix A.2, we use CONTEXTCITE to improve the question-answering capabilities of language models by extracting the most query-relevant sources from the context. We do so in three steps: (1) generate a response using the entire context, (2) use CONTEXTCITE to compute attribution scores for sources in the context, and (3) construct a query-specific context using only the top-k sources, which can be used to regenerate a response. In this section, we provide additional implementation-level details for this experiment.

1. *Datasets and models.* We evaluate this approach on two question-answering datasets: HotpotQA (Yang et al., 2018) and Natural Questions (Kwiatkowski et al., 2019). For each of these datasets, we evaluate the F_1 score of instruction-tuned `Llama-3-8B` (Figure 5) and `Llama-3-70B` (Figure 13) on $1,000$ randomly sampled examples from the validation set. In particular, we use the 4-bit quantized variant of `Llama-3-70B` due to compute constraints.

2. *Prompt.* We use the prompt template to elicit short answers from the model:

```
Context: {context}

Query: {query}

Please answer with a single word or phrase when possible.
If the question cannot be answered from the context, say so instead.
```

3. *Applying* CONTEXTCITE. We compute CONTEXTCITE attributions using 128 calls (i.e., sample size) and sample subsets of context sources uniformly at random.

4. *Constructing query-specific contexts.* Given a (context, query) pair and its CONTEXTCITE attributions, we construct a query-specific context by selecting the top-k sources according to the attribution scores and removing the rest. We then use this pruned context to regenerate a response. For Hotpot QA, wherein each context consists of multiple documents, we also include the sources that belong to the same document as the top-k sources to further improve the response quality.

D. Additional results

D.1. Linear surrogate model faithfulness on random examples from different benchmarks

On the right side of Figure 2, we show the actual logit-probabilities of different context ablations as well as the logit-probabilities predicted by a linear surrogate model. In that example, the linear surrogate model is quite faithful. In this section, we provide additional randomly sampled examples from CNN DailyMail (see Figure 6), Natural Questions (see Figure 7), and TyDi QA (see Figure 8). We use 256 context ablations to train the surrogate model, and observe that a linear surrogate model is broadly faithful across these benchmarks.

D.2. Exploiting sparsity to learn a surrogate model from a small number of ablations

One of the key design choices of CONTEXTCITE is to learn a *sparse* linear surrogate model. Specifically, we find that a generated statement can often be explained well by just a handful of sources. In particular, Figure 9a shows that the number of sources that are "relevant" for a particular generated statement is often small, even when the context comprises many sources. Motivated by this observation, we induce sparsity in the surrogate model via LASSO (Tibshirani, 1994). As we illustrate in Figure 9b, this enables learning a faithful linear surrogate model even with a small number of ablations.

D.3. Linear datamodeling score evaluation

In Section 4, we evaluate the quality of context attributions using the top-k log-probability drop. For a more fine-grained evaluation, we also consider whether attribution scores can accurately *rank* the effects of ablating different sets of sources on the log-probability of the response. Concretely, suppose that we sample a few different ablation vectors and compute the *sum* of the scores corresponding to the sources that are included by each. These summed scores may be viewed as the "predicted effects" of each ablation. We then measure the rank correlation between these predicted effects and the actual resulting probabilities. This metric, known as the *linear datamodeling score* (LDS), was first introduced by Park et al. (2023b) to evaluate methods for data attribution.

Definition D.1 (*Linear datamodeling score*)**.** Suppose that we are given a context attribution method τ. Let $v_1, \ldots, v_m$ be m randomly sampled ablation vectors and let $f(v_1), \ldots, f(v_m)$ be the corresponding probabilities of generating the original response. That is, $f(v_i) = p_{\text{LM}}(R \mid \textsc{Ablate}(C, v_i), Q)$. Let $\hat{f}_\tau(v) = \tau(s_1, \ldots, s_d)^\top v$ be the sum of the scores (according to τ) corresponding to sources that are included by v, i.e., the "predicted effect" of ablating according to v. Then the *linear datamodeling score* (LDS) of a context attribution τ is defined as

$$\text{LDS}(\tau) = \rho(\underbrace{\{f(v_1), \ldots, f(v_m)\}}_{\text{actual prob. of ablation}}, \underbrace{\{\hat{f}_\tau(v_1), \ldots, \hat{f}_\tau(v_m)\}}_{\text{"predicted effects" of ablation}}) \tag{3}$$

where ρ is the Spearman rank correlation coefficient (Spearman, 1904).

In Figure 10, we report the LDS of CONTEXTCITE and baselines with the same setup as in Section 4.

D.4. Additional evaluation

Using the same experiment setup as in Section 4, we evaluate CONTEXTCITE on additional models (`Phi-3-mini`) and additional benchmarks (TyDi QA and MS MARCO), and also compare it to additional baselines: ℓ_2-gradient norm, gradient-times-input, and attention rollout (Abnar & Zuidema, 2020). In Figure 11 and Figure 12, we show that CONTEXTCITE consistently outperforms the baselines across all models on the top-k log-probability drop metric and the linear datamodeling score, respectively.

D.5. Improving response quality for additional models

Recall that in Appendix A.2, we showed that constructing query-relevant contexts by removing irrelevant context sources (via CONTEXTCITE) can significantly improve the response quality of `Llama-3-8B`, and as a result, improve question-answering capabilities on Hotpot QA and Natural Questions. In Figure 13, we show that this approach also improves the response quality of `Llama-3-70B` on the same benchmarks.

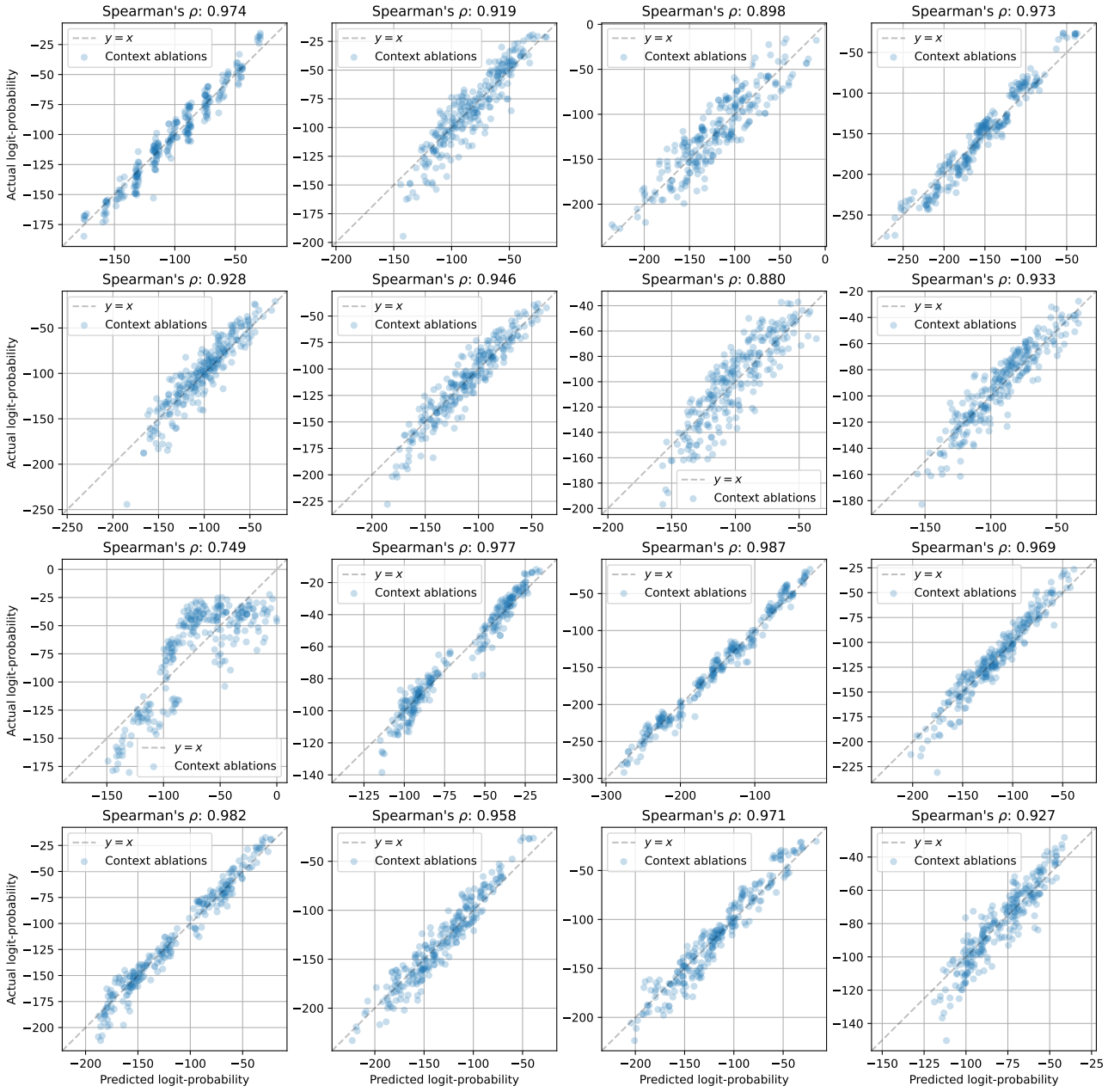


Figure 6. The predicted logit-probabilities of a surrogate model trained on 256 context ablations on randomly sampled examples from the CNN DailyMail, a summarization benchmark.

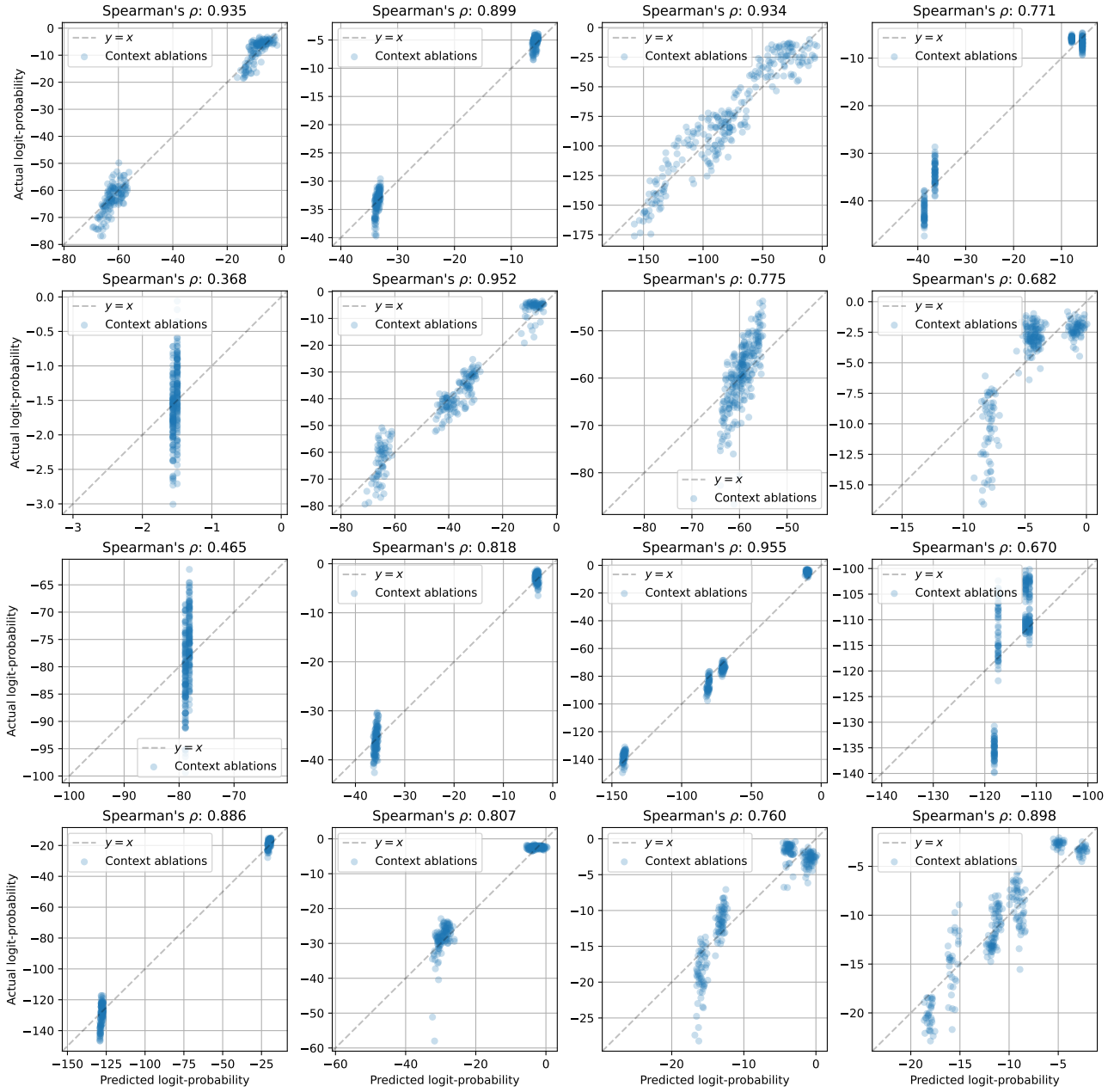


Figure 7. The predicted logit-probabilities of a surrogate model trained on 256 context ablations on randomly sampled (answerable) examples from the Natural Questions, a question answering benchmark.

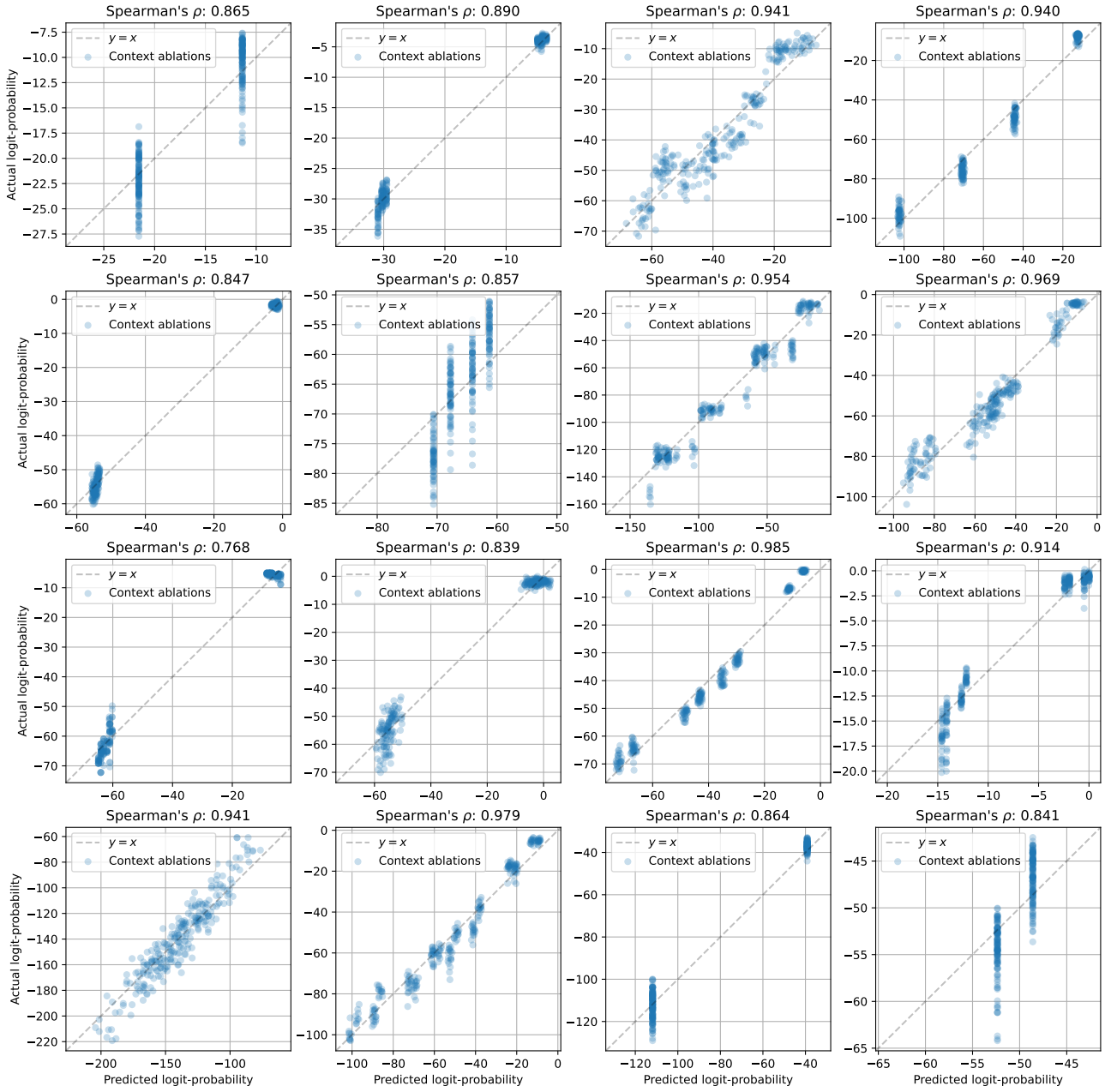
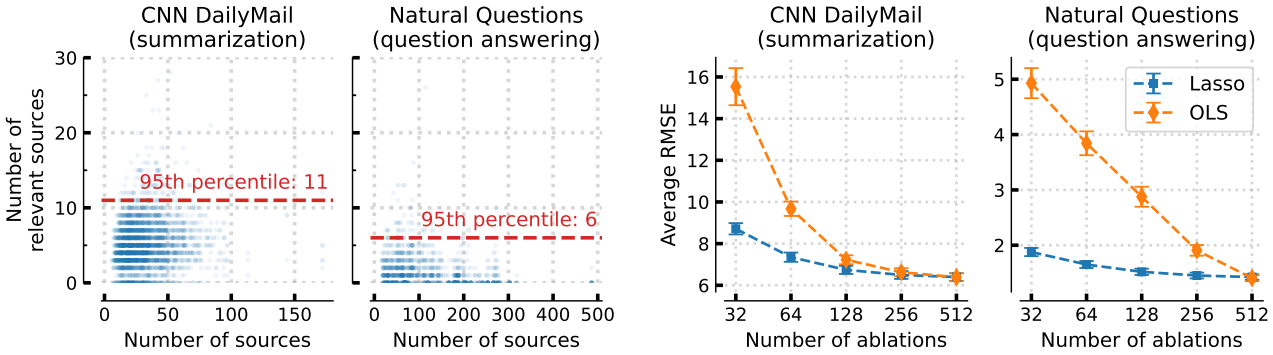


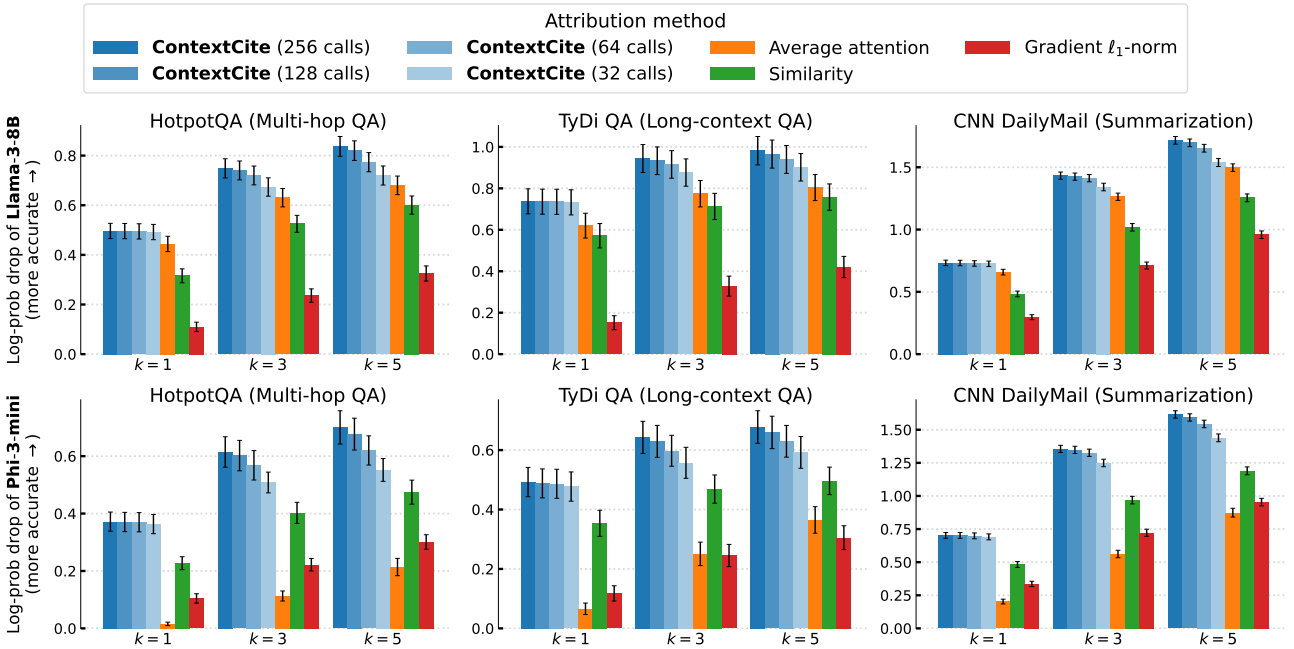
Figure 8. The predicted logit-probabilities of a surrogate model trained on 256 context ablations on randomly sampled (answerable) English examples from the TyDi QA, a question answering benchmark.



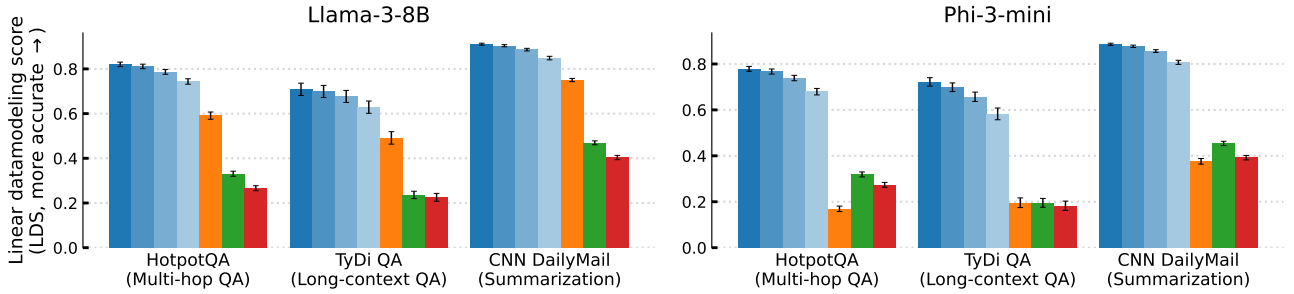
(a) The numbers of "relevant" and total sources for summarization (left) and question answering (right) tasks. A source is "relevant" if excluding it changes the probability of the response by a factor of at least $\delta = 2$.

(b) The root mean squared error (RMSE) of a surrogate model trained with LASSO and ordinary least squares (OLS) on held-out ablation vectors for two tasks: summarization (left) and question answering (right).

Figure 9. **Inducing *sparsity* improves the surrogate model's sample efficiency.** In CNN DailyMail (Nallapati et al., 2016), a summarization task, and Natural Questions (Kwiatkowski et al., 2019), a question answering task, we observe that the number of sources that are "relevant" sources for a particular statement generated by `Llama-3-8B` (AI, 2024) is often small, even when the context comprises many sources (Figure 9a). Therefore, by inducing sparsity via LASSO we can learn a faithful linear surrogate model even with a small number of ablations (Figure 9b). See Appendix C.5 for the exact setup.



(a) We report the top-k log-probability drop (1), which measures the effect of ablating top-scoring sources on the generated response. A higher drop indicates that the context attribution method identifies more relevant sources.



(b) We report the linear datamodeling score (LDS) (3), which measures the extent to which a context attribution can predict the effect of random context ablations.

Figure 10. **Evaluating context attributions.** We report the top-k log-probability drop (Figure 10a) and linear datamodeling score (Figure 10b) of CONTEXTCITE and baselines. We evaluate attributions of responses generated by `Llama-3-8B` and `Phi-3-mini` on up to $1,000$ randomly sampled validation examples from each of three benchmarks. We find that CONTEXTCITE using just 32 context ablations consistently outperforms the baselines—attention, gradient norm, and semantic similarity—across benchmarks and models. Increasing the number of context ablations to $\{64, 128, 256\}$ can further improve the quality of CONTEXTCITE attributions.

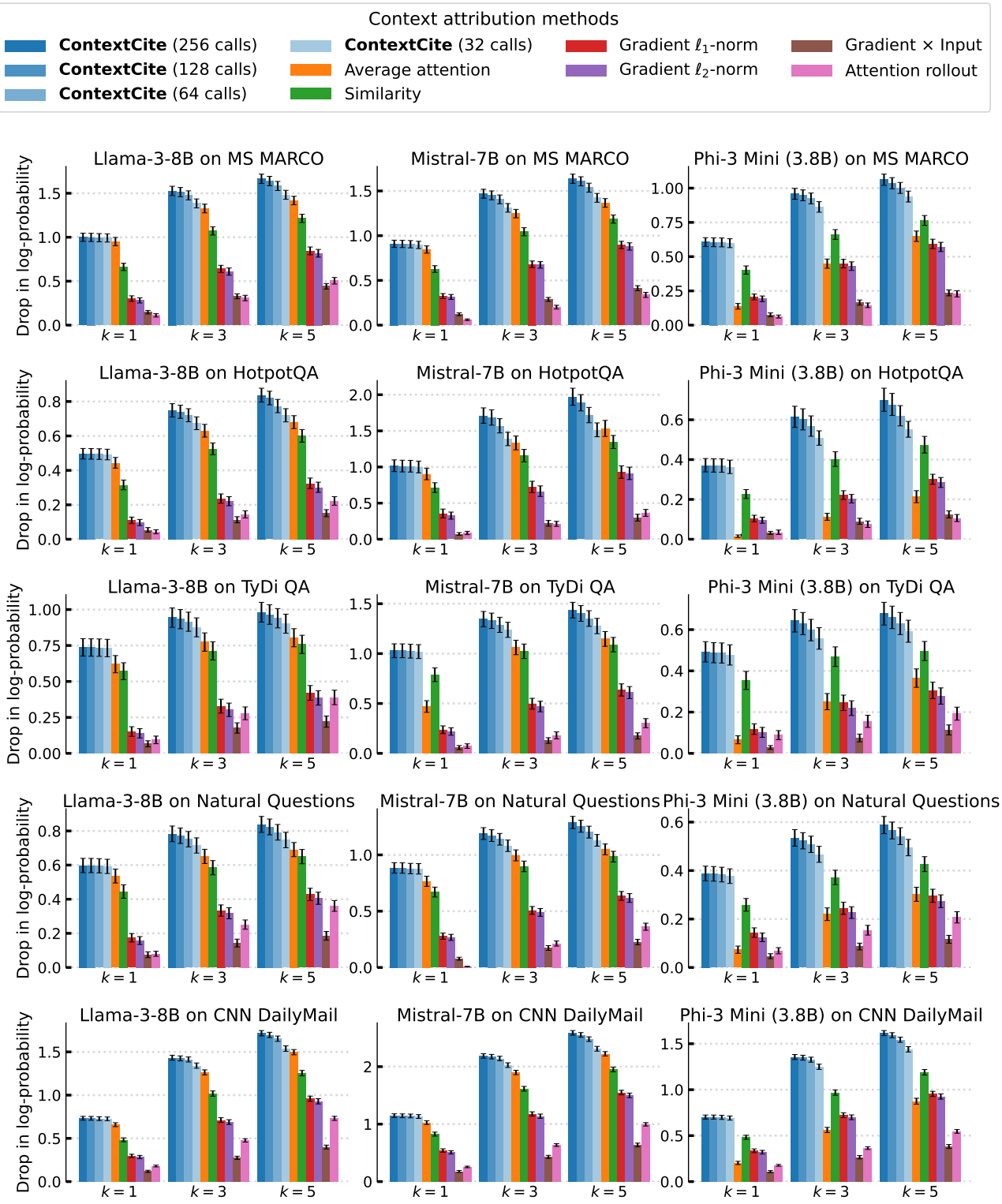


Figure 11. **Evaluating CONTEXTCITE on additional models and benchmarks using the top-k log-probability drop metric** (1). We compare CONTEXTCITE to additional baselines (ℓ_2-gradient norm, gradient-times-input, and attention rollout) on three models (`Llama-3-8B`, `Phi-3-mini`, `Mistral-7B`) and two additional benchmarks (`TyDi QA` and `MS-MARCO`). Each row corresponds to a different benchmark and each column corresponds to a different model. Across all benchmarks and models, CONTEXTCITE (with just 32 calls) consistently outperforms the baselines on the top-k log-probability drop metric, which measures the effect of ablating the top-k context sources with the highest attribution scores. Similar to our results in **??**, increasing the number of context ablations to $\{64, 128, 256\}$ can further improve the quality of CONTEXTCITE attributions in this setting as well.

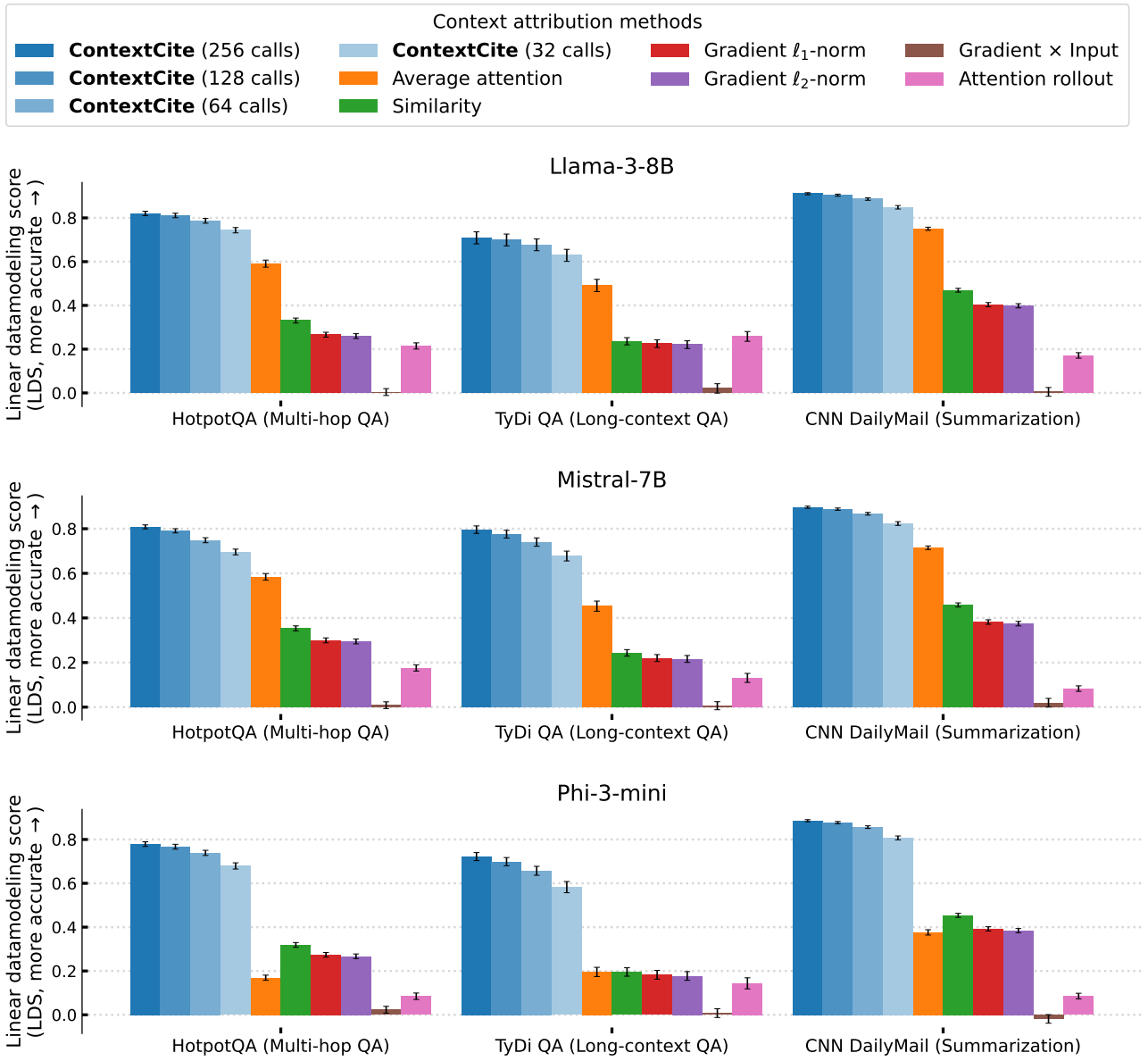


Figure 12. **Evaluating CONTEXTCITE on additional models and benchmarks using the linear datamodeling score (3).** Like in Figure 11, we compare CONTEXTCITE to additional baselines (ℓ_2-gradient norm, gradient-times-input, and attention rollout) on three models (`Llama-3-8B`, `Phi-3-mini`, `Mistral-7B`) and two additional benchmarks (`TyDi QA` and `MS-MARCO`). Each row corresponds to a different benchmark and each column corresponds to a different model. Across all benchmarks and models, CONTEXTCITE (with just 32 calls) consistently outperforms the baselines on the linear datamodeling score, which quantifies the extent to which context attributions predict the effect of ablating the context sources on the model response. Similar to our results in **??**, increasing the number of context ablations to $\{64, 128, 256\}$ further improves the quality of CONTEXTCITE attributions in this setting as well.

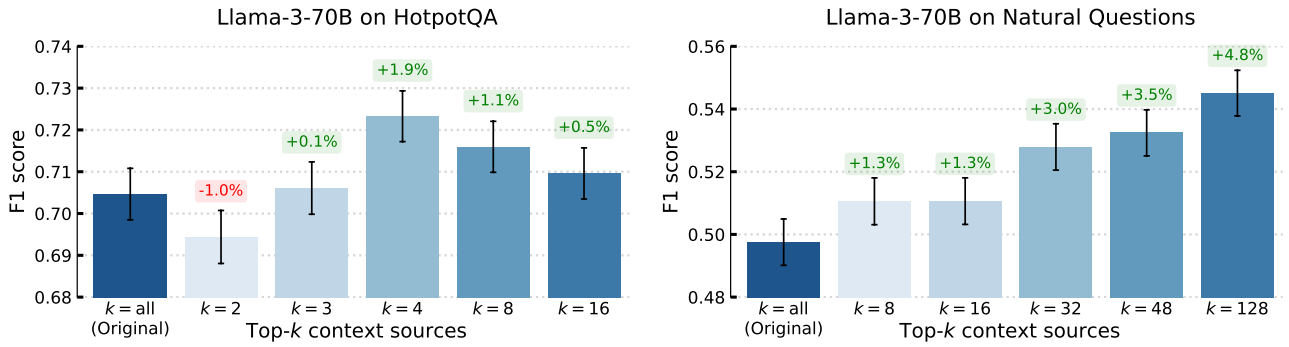


Figure 13. **Improving response quality of `Llama-3-70B` by constructing query-specific contexts via CONTEXTCITE.** On the left, we show that filtering contexts by selecting the top-$\{2, \ldots, 16\}$ query-relevant sources (via CONTEXTCITE) improves the average F_1-score of `Llama-3-70B` on $1,000$ randomly sampled examples from the Hotpot QA dataset. Similarly, on the right, replacing the entire context with the top-$\{8, \ldots, 128\}$ query-relevant sources boosts the average F_1-score of `Llama-3-70B` on $1,000$ randomly sampled examples from the Natural Questions dataset. Similar to our results on `Llama-3-8B` in Figure 5, CONTEXTCITE also improves response quality for `Llama-3-70B` by extracting the most query-relevant information from the context.

E. Additional discussion

E.1. Computational efficiency of CONTEXTCITE

Most of the computational cost of CONTEXTCITE comes from creating the surrogate model's training dataset. Hence, the efficiency of CONTEXTCITE depends on how many ablations it requires to learn a faithful surrogate model. We find that CONTEXTCITE requires just a small number of context ablations to learn a faithful surrogate model—in our experiments, 32 context ablations suffice. Thus, attributing responses using CONTEXTCITE is $32\times$ more expensive than generating the original response. We note that the inference passes for each of these context ablations can be fully parallelized. Furthermore, because CONTEXTCITE is a *post-hoc* method that can be applied to any existing response, a user could decide when they would like to pay the additional computational cost of CONTEXTCITE to obtain attributions. When we use CONTEXTCITE to attribute multiple statements in the response, we use the same context ablations and inference calls. In other words, there is a fixed cost to attribute (any part of) a generated response, after which it is very cheap to attribute specific statements.

E.1.1. WHY DO WE ONLY NEED A SMALL NUMBER OF ABLATIONS?

We provide a brief justification for why 32 context ablations suffice, even when the context comprises many sources. Since we are solving a linear regression problem, one might expect the number of ablations needed to scale *linearly* with the number of sources. However; in our sparse linear regression setting, we have full control over the covariates (i.e., the context ablations). In particular, we ablate sources in the context independently and each with probability $1/2$. This makes the resulting regression problem "well-behaved." Specifically, this lets us leverage a known result (see Theorems 7.16 and 7.20 of Wainwright (2019)) which tells us that we only need $O(k \log(d))$ context ablations, where d is the total number of sources and k is the number of sources with non-zero relevance to the response. In other words, the number of context ablations we need grows very slowly with the total number of sources. It only grows linearly with the number of sources that the model relies on when generating a particular statement. As we show empirically in Figure 9a, this number of sources is often small.

E.2. Limitations of CONTEXTCITE

In this section, we discuss a few limitations of CONTEXTCITE.

Potential failure modes. Although we find a *linear* surrogate model to often be faithful empirically (see Figure 2, Appendix D.1), this may not always be the case. In particular, we hypothesize that the linearity assumption may cease to hold when many sources contain the same information. In this case, a model's response would only be affected by excluding every one of these sources. In practice, to verify the faithfulness of the surrogate model, a user of CONTEXTCITE could hold out a few context ablations to evaluate the surrogate model (e.g., by measuring the LDS). They could then assess whether CONTEXTCITE attributions should be trusted.

Another potential failure mode of CONTEXTCITE is attributing generated statements that follow from previous statements. Consider the generated response: "He was born in 1990. He is 34 years old." with context mentioning a person born in 1990. If we attribute the statement "He was born in 1990." we would likely find the relevant part of the context. However, if we attribute the statement "He is 34 years old." we might not identify any attributed sources, despite this statement being grounded in the context. This is because this statement is conditioned on the previous statement. Thus, in this case there is an "indirect" attribution to the context through a preceeding statement that would not be identified by the current implementation of CONTEXTCITE.

Unintuitive behaviors. A potentially unintuitive behavior of CONTEXTCITE is that it can yield a low attribution score even for a source that supports a statement. This is because CONTEXTCITE provides contributive attributions. Hence, if a language model already knows a piece of information from pre-training and does not rely on the context, CONTEXTCITE would not identify sources. This may lead to unintuitive behaviors for users.

Validity of context ablations. In this work, we primarily consider sentences as sources for context attribution and perform context ablations by simply removing these sentences. One potential problem with this type of ablation is *dependencies* between sentences. For example, consider the sentences: "John lives in Boston. Charlie lives in New York. He sometimes visits San Francisco." In this case, "He" refers to Charlie. However, if we ablate just the sentece about Charlie, "He" will now refer to "John." There may be other ablation methods that more cleanly remove information without changing the meaning of sources because of dependencies.

Computational efficiency. As previously discussed, attributing responses using CONTEXTCITE is $32\times$ more expensive than generating the original response. This may be prohibitively expensive for some applications.