Provenance: A Light-weight Fact-checker for Retrieval Augmented LLM Generation Output

Hithesh Sankararaman[*], **Mohammed Nasheed Yasin**[†],
Tanner Sorensen, **Alessandro Di Bari**, and **Andreas Stolcke**

Uniphore Software Systems Pvt. Ltd, U.S.A

Abstract

We present a light-weight approach for detecting nonfactual outputs from retrieval-augmented generation (RAG). Given a context and putative output, we compute a factuality score that can be thresholded to yield a binary decision to check the results of LLM-based question-answering, summarization, or other systems. Unlike factuality checkers that themselves rely on LLMs, we use compact, open-source natural language inference (NLI) models that yield a freely accessible solution with low latency and low cost at run-time, and no need for LLM fine-tuning. The approach also enables downstream mitigation and correction of hallucinations, by tracing them back to specific context chunks. Our experiments show high area under the ROC curve (AUC) across a wide range of relevant open source datasets, indicating the effectiveness of our method for fact-checking RAG output.

1 Introduction

With natural language understanding applications increasingly relying on large language models (LLMs) to answer questions, summarize texts, and perform other tasks, detecting nonfactual claims in the generated text has become critical from an ethical and compliance standpoint. LLMs, while powerful, are prone to generate nonfactual or "hallucinated" information that can lead to misinformation and introduce errors in business processes. To address this problem, we present *Provenance*, a fact-checking method for output generated by LLMs, with respect to a given context that provides the factual basis for the output.

Provenance leverages compact cross-encoder models that offer substantial advantages over conventional LLM-based methods. These advantages include accessibility, low latency/high throughput, and interpretable judgments.

Provenance is evaluated on diverse open-source datasets, including the TRUE dataset (Honovich et al., 2022), MSMarco (Nguyen et al., 2016), TruthfulQA (Lin et al., 2022), HotpotQA (Yang et al., 2018), HaluEval (Li et al., 2023) and HaluBench (Ravi et al., 2024). These datasets encompass a variety of question-answering contexts, providing a robust testbed for our methods. We assess performance using standard detection metrics to demonstrate our method's efficacy as a factuality checker for LLM-generated content.

Our findings show that *Provenance* achieves competitive hallucination detection performance (as measured by AUC) across different datasets, thus contributing to improved trustworthiness and utility of LLMs in real-world applications.

2 Related Work

In prior work, three main approaches to factuality evaluation have been used: 1. LLM ablation, 2. LLM introspection, and 3. NLI methods.

LLM ablation refers to approaches such as SelfCheckGPT (Manakul et al., 2023) and Agrawal et al. (2024) that measure the consistency of multiple candidate generations for a given prompt. Methods such as Varshney et al. (2023) that gauge factuality based on the language model's output distributions also fall in this category.

LLM introspection refers to techniques that use the reasoning ability of modern language models to evaluate their own or another model's output. Work by Kadavath et al. (2022), Es et al. (2024) and Muller et al. (2023) are examples of this.

Natural language inference (NLI) methods exploit special-purpose cross-encoder models that indicate whether a claim is supported by a premise. This approach usually involves breaking down the context into a list of premises (*context items*), and

[*] hithesh.sankararaman@uniphore.com
[†] mohammed.yasin@uniphore.com

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 1305–1313
November 12-16, 2024 ©2024 Association for Computational Linguistics

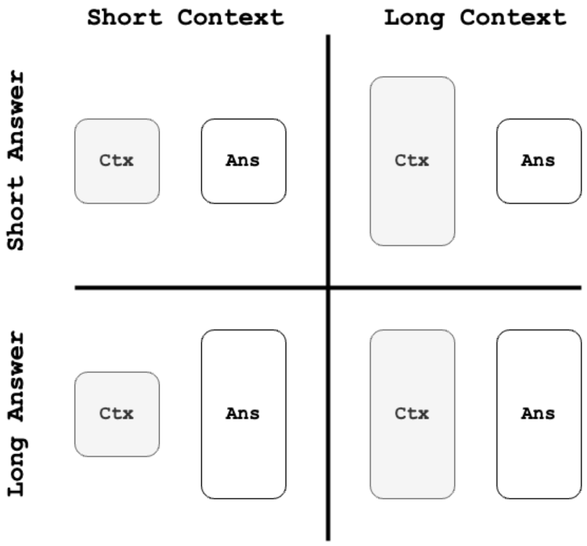

Figure 1: Typical Context vs. Answer length scenarios in which fact-checking is performed.

the generation into a list of claims. Laban et al. (2022) is a representative method that chunks the generation and context at the sentence level and computes pair-wise entailment judgments, which are then aggregated. However, this approach has some shortcomings: 1. the original prompt/query is ignored when evaluating entailment, and 2. context and generation chunking is overly simplistic. Our method falls into the NLI-based category, but addresses these shortcomings.

Broadly speaking, there are four scenarios (Fig. 1) in which a fact-checker may operate: 1. short context/short answer, 2. short context/short answer, 3. short context/long answer, and 4. long context/long answer. When the answer or context are long, we need a mechanism to break them into smaller units. We narrow our focus based on the following observations and practical considerations:

1. Reliable semantic chunking is an as yet evolving field in NLP (Yang et al., 2020; Zhai et al., 2017; Johnson and Zhang, 2005).

2. When it comes to chunking long contexts we can reuse the chunks that the RAG *retriever* returned. Retrievers need to chunk text due to input sequence length limitations in their embedder.

3. Lack of open-source datasets for long-answer benchmarking.

While we have a straightforward way to break down the contexts, it is still hard to chunk generated answers meaningfully. The chunking of information is an area for further research, since context and answers come in many forms, such as text, conversations, and tables. We limit the scope of this paper to text source for scenarios in the first row of Figure 1, namely, short context/short answer and long context/short answer. We also need to ensure that the chunk length chosen is viable for all the models in the system.

3 System Description

Contemporary fact-checking systems employ approaches based on LLMs as a judge (Zhu et al., 2023) to validate the generations of other LLMs. By virtue of being auto-regressive, the judge-LLMs themselves are prone to hallucinate. By contrast, Provenance (Figure 2) uses two cross-encoder based models that do not suffer from this tendency. As input, Provenance expects

1. a list of context items used by the generating LLM in the upstream RAG,

2. the user's original question or query, and

3. the generated text to fact-check.

The first cross-encoder model determines which of the context items are relevant to the given query and generates a score. This score is then used to select context items to build a smaller and more focused context, which we refer to as the *sources*. The selection process also produces a *weight* associated with each source. In parallel. we construct the *claim* by inserting the query and generation into a *claim prompt*. The *claim* and *sources* are then passed to the second cross-encoder model for validation, generating a *factuality score* for each *claim/source* pair. These scores are then aggregated using the source *weights* generated earlier to produce a single score for the LLM's output. This score can be thresholded to produce a binary factuality decision, with the threshold being tuned for a target dataset and task. Here we used threshold-invariant evaluation methods, such as receiver operator characteristics (ROC) and area under the curve (AUC).

3.1 Relevancy Scorer

To assess the relevance of context items to the query, we use a cross-encoder model to generate relevance scores for each context item. This process is similar to the re-ranking of search results

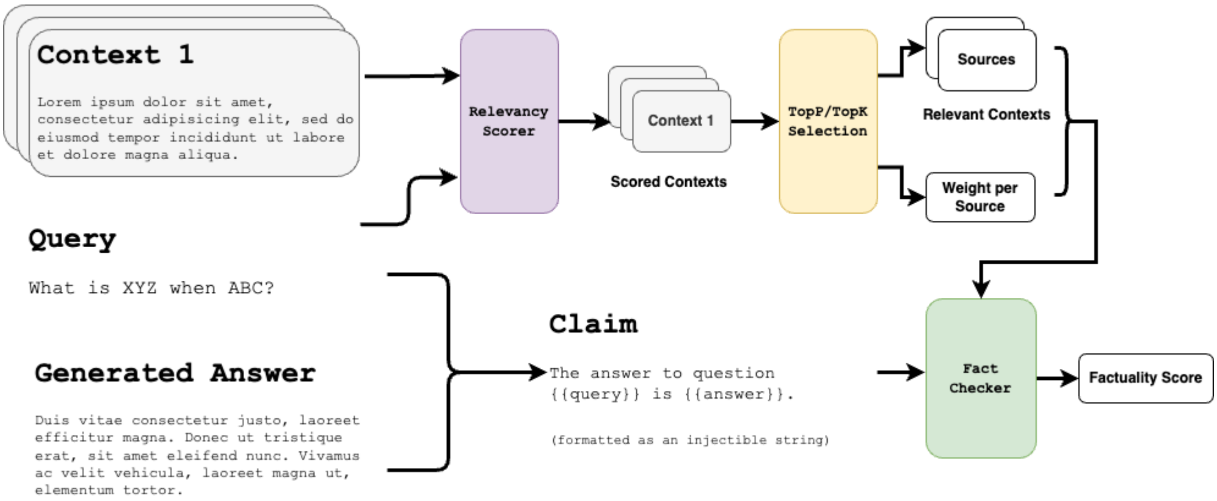
Context 1
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Query
What is XYZ when ABC?
Generated Answer
Duis vitae consectetur justo, laoreet efficitur magna. Donec ut tristique erat, sit amet eleifend nunc. Vivamus ac velit vehicula, laoreet magna ut, elementum tortor.
Relevancy Scorer
Context 1
Scored Contexts
TopP/TopK Selection
Sources
Relevant Contexts
Weight per Source
Claim
The answer to question {{query}} is {{answer}}.
(formatted as an injectible string)
Fact Checker
Factuality Score

Figure 2: Provenance system architecture.

w.r.t. queries in a RAG system, except that we do not perform the *top_k* sampling step. We leave this to a downstream component.
Given a query Q and a context item D, the relevance score S is calculated as

$$S = \text{Cross-Encoder}(Q, D) \tag{1}$$

Here, S is a real number in $(-\infty, \infty)$, but empirically scors range within $(-10, 10)$.

The cross-encoder used is a RoBERTa-based model[1] trained by Mixedbread.

3.2 Context Item Selection

To select the *sources* among the scored contexts, we employ one of two strategies. *TopK* is similar to the one used in the RAG retrieval and reranking steps. *TopP* is adapted from nucleus sampling (Holtzman et al., 2019), a commonly used method to sample from an LLM's output distribution. For both strategies, the relevance scores of all context items are normalized to be interpretable as probabilities, i.e., to have range $(0, 1)$ and sum to one.

The TopK selector simply retains the *top_k* contexts with highest relevance scores. The TopP selector retains a minimal set of contexts in order of decreasing relevance scores, such that their cumulative probability is at least *top_p*, where *top_k* and *top_p* respectively are hyperparameters.

Following the selection of the *sources*, we renormalize their relevance scores again, which then serve as the *weights* to be placed on each source later in fact-checking.

We have not carried out a systematic optimization of *top_k* and *top_p* values for this paper. For *top_p* we chose 0.9, which selected an average of 3 to 4 sources on our datasets. For *top_k* we chose 5, which is half the maximum of possible sources defined in the datasets used here (see Section 5).

Anecdotally, on real-world production datasets, we found that better results are achieved by choosing a single *top_p* value rather than setting *top_k*.

3.3 Fact Checker

Provenance uses cross-encoder NLI models to evaluate the factual consistency of the LLM's output, given a *source* and the user's query. The model we use is a specialized hallucination detection model[2] trained by Vectara.

The steps to compute factuality scores are

1. *Input preparation:* we insert the query and answer into a prompt that claims "The answer to question <QUERY> is <ANSWER>." This prepared claim prompt is then paired off with each *source*.

2. *Scoring:* The cross-encoder is used to compute a score indicating how well the answer is supported by the context in the light of the query. Here, the scoring function is $FScore = \text{nli-model}(S, C)$, where S is one of the sources and C is the prepared claim prompt.

3. *Aggregation* of the scores and weights for all the *sources* using one of the following func-

[1] Available on huggingface as mixedbread-ai/mxbai-rerank-base-v1

[2] Available on Huggingface as vectara/hallucination_evaluation_model

tions: (a) min, (b) max, or (c) weighted average.

The final factuality score can be normalized to indicate the probability of the claim being supported by the *sources*.

4 Data

We utilize several open-source datasets to evaluate the effectiveness of our approach in detecting non-factual texts generated by LLMs. These datasets provide a diverse range of question-answering contexts and candidate answers, ensuring a comprehensive assessment. Table 2 provides an overview of datasets showing the counts of Hallucination and Entailment (=Factual) labels. As shown, most data sources have a roughly balanced label distribution, though some (like the HaluEval GENERAL subset) are skewed toward one class.

4.1 TRUE

The TRUE dataset (Honovich et al., 2022) is comprised of eleven different subsets, each with questions, answers, and contexts. It is designed to test the factual accuracy of LLM outputs across various domains and question types.

4.2 MSMarco

MSMarco (Microsoft MAchine Reading COmprehension) (Nguyen et al., 2016) is a large-scale dataset created for machine reading comprehension tasks. The dataset is particularly useful for evaluating our method in the context of web-based information retrieval and answering user queries accurately.

4.3 Truthful QA

TruthfulQA (Lin et al., 2022) is a dataset specifically designed to test the truthfulness of LLM-generated responses. This dataset is crucial for assessing our approach's capability to handle tricky or potentially deceptive questions.

4.4 HotpotQA

HotpotQA (Yang et al., 2018) is a multi-hop question-answering dataset that requires the model to retrieve and reason over multiple pieces of evidence to generate a correct answer. The dataset includes questions, supporting facts, and distractor contexts, making it a complex and rigorous test for our method. The multi-hop nature of HotpotQA ensures that our approach can handle intricate reasoning and context synthesis tasks effectively.

4.5 HaluEval

Hallucination Evaluation Benchmark for Large Language Models (HaluEval) (Li et al., 2023) is a large collection of generated and human-annotated hallucinated samples for evaluating the performance of LLMs in recognizing hallucination.

4.6 HaluBench

HaluBench (Ravi et al., 2024) is a hallucination evaluation benchmark of 15k samples that consists of context-question-answer triplets annotated for whether the examples contain hallucinations. Compared to prior datasets, HaluBench is the first open-source benchmark containing hallucination tasks sourced from real-world domains that include finance and medicine.

5 Data Preparation

The MSMarco and HotpotQA datasets each contain 10 *sources* per question, with one relevant *source* per question in MSMarco and multiple relevant*sources* per question in HotpotQA. Other datasets have a single *source* paragraph given for each question. All *sources* were split into individual sentences, and all datasets were converted into triplets with the query and answer as strings, and the *sources* as a list of strings.Our framework processes these triplets and returns a score, which, combined with a set threshold, classifies the generated answer as hallucinated or factual. To calculate AUC, we ensured representation of the two classes by generating hallucinated answers for datasets lacking them.

For the MSMarco dataset (Nguyen et al., 2016), we randomly selected 252 out of 100,000 datapoints and generated hallucinated answers using the GPT-3.5-turbo model, which were verified manually.

For HotpotQA (Yang et al., 2018), we appended the QA data from HaluEval (Li et al., 2023), which includes 10K hallucinated samples based on HotpotQA.

6 Experiments

6.1 Preliminary Experiments

Before developing our final Provenance framework, we also experimented with a BERT-based Relevancy Scorer using TopP selection and a DeBERTa-based NLI model for computing factuality scores. These preliminary experiments showed the importance of (1) sorting of selected sources into

Data Type	Dataset	Sample Count	AUC (Provenance)	AUC (TRUE paper)	Model size (TRUE paper)
Paraphrase Detection	**PAWS**	8000	~~**94***~~	89.7^{Q^2}	11B
Dialogue Generation	**BEGIN**	836	80	**87.9**$^{BERT\_SCORE}$	750M
	DialFact	8689	**92***	86.1^{Q^2}	11B
	Q2	1088	**86***	80.9^{Q^2}	11B
Abstractive Summarization	**FRANK**	671	89	**89.4**^{ANLI}	11.5B
	MNBM	2500	**79***	77.9^{ANLI}	11.5B
	QAGS_CNNDM	235	76.3	**83.5**$^{Q^2}$	11B
	QAGS_XSUM	239	80.4	**83.8**^{ANLI}	11.5B
	Summ_Eval	1600	70.1	**81.7**$^{SC\_ZS}$	58.7M
Fact Verification	**VITAMIN C**	63054	~~**95.8***~~	88.3^{ANLI}	11.5B
	FEVER	18209	~~92~~	93.2^{ANLI}	11.5B

Table 1: Comparison of AUC scores and model sizes from the TRUE paper with our Provenance framework; we report AUC scores*100 for better readability, as in the TRUE paper (Honovich et al., 2022). Results from FEVER, PAWS, and VITAMIN C (reported above, but crossed-out) are not comparable to the TRUE results since our NLI model has seen samples from these datasets. The highest score for our method is in bold with an asterisk, while the highest score from the TRUE paper methods is in bold. The size of the Provenance model is $\approx$ 300M parameters.

Dataset Name	Sub Dataset Name	Label 0 (Hallucination)	Label 1 (Entailment)	Total Samples
TRUE	VITC	31570	31484	63054
	BEGIN	554	282	836
	DIALFACT	5348	3341	8689
	FEVER	11816	6393	18209
	FRANK	448	223	671
	MNBM	2245	255	2500
	PAWS	4461	3539	8000
	Q2	460	628	1088
	QAGS_CNNDM	122	113	235
	QAGS_XSUM	123	116	239
	SUMMEVAL	294	1306	1600
MS MARCO		252	252	504
HOTPOTQA		10000	100447	110447
HALUBENCH		7170	7730	14900
TRUTHFUL_QA		1716	1260	2976
HALUEVAL	DIALOGUE	10000	10000	20000
	QA	10000	10000	20000
	SUMMARIZATION	10000	10000	20000
	GENERAL	815	3692	4507
TOTAL		**107394**	**191061**	**298455**

Table 2: Overview of Datasets and Sub-Datasets Categorized by Hallucination and Entailment Labels, including Total Sample Counts. (Entailment corresponds to Factual for our purposes.)

their original temporal order and (2) cosine scoring (length normalization) of similarity scores; detailed results can be found in the Appendices A.2 and A.3.

6.2 Experiment 1: Provenance framework

The experimental setup follows the methodology described in Section 3. The pipeline consists of

Dataset Type	Dataset	AUC
Paraphrase Detection	**PAWS**	0.94
Dialogue Generation	**BEGIN**	0.80
	DIALFACT	0.92
	Q2	0.86
	HaluEval Dialogue	0.69
Abstractive Summarization	**FRANK**	0.89
	MNBM	0.79
	QAGS_CNNDM	0.76
	QAGS_XSUM	0.80
	Summ_Eval	0.70
	HaluEval Summarization	0.66
Fact Verification	**VITAMIN C**	0.96
	FEVER	0.92
	Truthful_QA	0.59
	MS_MARCO	0.84
	HaluBench	0.71
	HaluEval QA	0.74
Open Domain	**HaluEval General**	0.54

Table 3: Results for Experiment 1: Provenance

three main components: Relevancy Scorer, Context Item Selector, and Fact Checker. The Relevancy Scorer uses cross-encoder based models to rank context items based on their relevance to the given query. The Context Item Selector then selects top documents using either the TopK or TopP strategy. Finally, the Fact Checker evaluates the combined context to detect hallucinated content and returns a score. Results are presented in Table 3.

6.3 Experiment 2: Long context and multi-hop scenarios

The experimental setup aligns with that of Section 6.2. In scenarios involving longer contexts and multi-hop scenarios, where answers span multiple context claims, as seen in HotpotQA (Yang et al., 2018) and for some samples in HaluBench (Ravi et al., 2024), we aggregate the scores from the Fact Checker and weights from the Context Item Selector for each filtered *source*. Results are presented in Table 5.

7 Results

We report the ROC AUC of our system for all datasets mentioned in Section 4. The ROC curves in Figures 3 and 4 show the trade-off between false versus missed hallucination detections for the least

Models	HaluBench	Model Size
GPT-4o	87.9	1.7T
GPT-4-Turbo	86.0	1.7T
GPT-3.5-Turbo	62.2	175B
Claude-3-Sonnet	84.5	70B
Claude-3-Haiku	68.9	20B
RAGAS Faithfulness	70.6	100B
Mistral-Instruct-7B	78.3	7B
Llama-3-Instruct-8B	83.1	8B
Llama-3-Instruct-70B	87.0	70B
LYNX (8B)	85.7	8B
LYNX (70B)	**88.4**	70B
Provenance	65.6	300M

Table 4: Comparison of accuracies of different LLM-based methods in HaluBench (Ravi et al., 2024) with *Provenance*. The reported accuracy for *Provenance* corresponds to Experiment 2, utilizing top_k = 5 and the maximum aggregation logic.

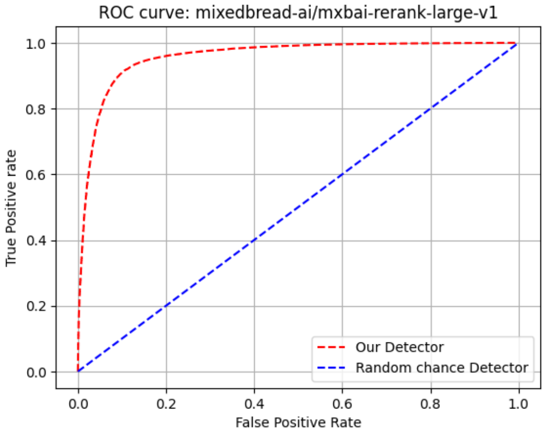

Figure 3: ROC curve for VITC task

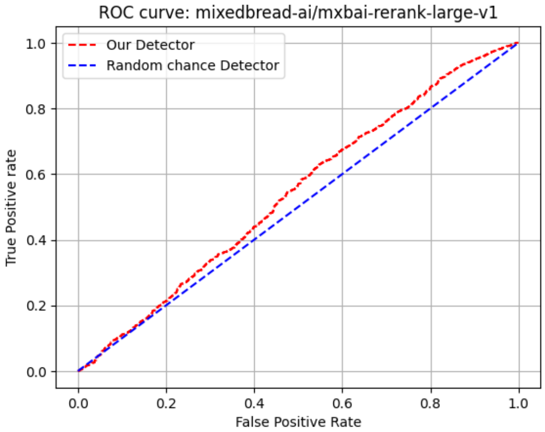

Figure 4: ROC curve for HALUEVAL-GENERAL task

Dataset	Selection Strategy	TopP 0.9	TopK 5
	Aggregation	**AUC**	**AUC**
HotpotQA	**min**	0.227	0.440
	max	0.809	0.688
	weighted average	0.252	0.372
HaluBench	**min**	0.645	0.644
	max	0.680	0.714
	weighted average	0.664	0.676

Table 5: Results from Experiment 2: Long context and multi-hop scenarios

Models	QA	Dialogue	Summarization	General	Model Size
ChatGPT	62.59	**72.40**	58.53	79.44	175B
Claude 2	**69.78**	64.73	57.75	75.00	135B
Claude	67.60	64.83	53.76	73.88	130B
Davinci002	60.05	60.81	47.77	**80.42**	6B
Davinci003	49.65	68.37	48.07	80.40	175B
GPT-3	49.21	50.02	51.23	72.72	13B
Llama 2	49.60	43.99	49.55	20.46	7B
ChatGLM	47.93	44.41	48.57	30.92	7B
Falcon	39.66	29.08	42.71	18.98	7B
Vicuna	60.34	46.35	45.62	19.48	7B
Alpaca	6.68	17.55	20.63	9.54	7B
Provenance	67.48	62.97	**62.27***	56.70	**300M**

Table 6: Comparison of *Provenance* accuracy to different models across various tasks presented in HaluEval (Li et al., 2023).

and the most difficult of the test sets, respectively. Note that we did not reproduce the evaluations of the LLM-based methods listed in Tables 4 and 6, and simply copied the results reported in the respective references.

7.1 AUC Analysis

Comparing our AUC scores with the TRUE dataset paper (Honovich et al., 2022) in Table 1, our framework achieves the best AUC for **3 out of 7** datasets (DialFact, MNBM, and Q2). Notably, the ANLI method (Honovich et al., 2022), which uses a 11B-parameter model, slightly outperforms ours on some datasets. Still, our model with $\approx$ **300M** parameters shows competitive results with minimal differences: 0.4% for FRANK and 3.4% for QAGS_XSUM, while performing **better by 2.9%** for MNBM.

7.2 Accuracy comparison

Comparing accuracy scores from the HaluEval benchmark (Li et al., 2023) in Table 6, *Provenance* achieves the best accuracy on the summarization task, **surpassing ChatGPT by 3.74%**, and is only 2.3% behind Claude2 on the QA task, despite Claude 2 having 135B parameters.

Comparing accuracy scores from the HaluBench benchmark (Ravi et al., 2024) in Table 4, *Provenance* is **surpassing GPT-3.5-Turbo by 3.38%**, and is only 3.32% behind Claude-3-Haiku, despite Claude-3-Haiku having two orders of magnitude more (20B) parameters.

8 Conclusion

We have presented Provenance, a practical approach to fact-checking of LLM output in RAG scenarios, based on light-weight cross-encoder models for relevance scoring and natural language inference. The factuality scoring takes the query into account when judging a generated answer against the retrieved information sources. Evaluation on a variety of open-source datasets shows our method to be effective for hallucination detection across a variety of tasks, at a model size that is a fraction of that of LLMs that are commonly used for this task. We expect our method to make the fact-checking of LLM output more accessible and scalable, contributing to the reliability and trustworthiness of LLM-based applications.

Acknowledgments

We thank Neha Gupta and Roberto Pieraccini for their support and advice in doing this research.

References

Ayush Agrawal, Mirac Suzgun, Lester Mackey, and Adam Kalai. 2024. Do language models know when they're hallucinating references? In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 912–928, St. Julian's, Malta. Association for Computational Linguistics.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. TRUE: Re-evaluating factual consistency evaluation. *arXiv preprint arXiv:2204.04991*.

Rie Johnson and Tong Zhang. 2005. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 1–9.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *Preprint*, arXiv:2207.05221.

Philippe Laban, Tobias Schnabel, Paul N. Bennett, and Marti A. Hearst. 2022. SummaC: Re-visiting NLI-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. *arXiv preprint arXiv:2305.11747*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proc. ACL (Volume 1: Long Papers)*, pages 3214–3252, Dublin.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Benjamin Muller, John Wieting, Jonathan Clark, Tom Kwiatkowski, Sebastian Ruder, Livio Soares, Roee Aharoni, Jonathan Herzig, and Xinyi Wang. 2023. Evaluating and modeling attribution for cross-lingual question answering. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 144–157, Singapore. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268.

Selvan Sunitha Ravi, Bartosz Mielczarek, Anand Kannappan, Douwe Kiela, and Rebecca Qian. 2024. Lynx: An open source hallucination evaluation model. *Preprint*, arXiv:2407.08488.

Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. 2023. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *Preprint*, arXiv:2307.03987.

Jinbiao Yang, Qing Cai, and Xing Tian. 2020. How do we segment text? Two-stage chunking operation in reading. *ENEURO*, 7(3).

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *Preprint*, arXiv:1809.09600.

Feifei Zhai, Saloni Potdar, Bing Xiang, and Bowen Zhou. 2017. Neural models for sequence chunking. In *Proc. AAAI Conference on Artificial Intelligence*, 1, pages 3365–3371.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2023. JudgeLM: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*.

A Appendix

Here we report preliminary experiments to test the ability of a vector similarity approach in determining context relevance to a query. The principal conclusion of these experiments was that we needed better recall, switching to a cross-encoder scoring approach enabled this.

Data Type	Dataset	Sample Count	EXP-0.1 AUC	EXP-0.2 AUC	EXP-0.3 AUC
Paraphrase Detection	**PAWS**	8000	0.678	0.777	0.805
Dialogue Generation	**BEGIN**	836	0.632	0.749	0.749
	DialFact	8689	0.653	0.853	0.859
	Q2	1088	0.637	0.735	0.737
Abstractive Summarization	**FRANK**	671	0.452	0.720	0.790
	MNBM	2500	0.594	0.747	0.752
	QAGS_CNNDM	235	0.375	0.507	0.576
	QAGS_XSUM	239	0.533	0.743	0.798
	Summ_Eval	1600	0.447	0.546	0.639
Fact Verification	**VITAMIN C**	63054	0.607	0.813	0.825
	FEVER	18209	0.678	0.817	0.928
	TRUTHFUL_QA	2976	0.557	0.607	0.595
	MS_MARCO	504	0.853	0.853	0.820

Table 7: Baseline results from preliminary experiments on dot-product relevance scoring (Appendix A.1), sources in temporal order (Appendix A.2), and cosine similarity (Appendix A.3).

A.1 Experiment 0.1: Dot-product scoring

Our framework involved three main components: a sentence-tokenizer, a context filter, and a detector. The *Spacy sentencizer*[3] tokenized the context paragraphs into sentences. These tokenized sentences, along with a formatted string combining the query and the answer ("The answer to the question {query} is {answer}."), are vectorized using a BERT-based model.[4] A dot product is computed between each context sentence and the formatted string, selecting the most relevant context sentences based on the TopP selection strategy. These filtered context sentences and the formatted string are then passed to the NLI model[5] to obtain the entailment scores. The ROC AUC score and ROC curve are derived from these entailment scores and ground-truth labels (0 for hallucination and 1 for correct answers). Results are presented in Table 7.

A.2 Experiment 0.2: Temporal ordering of sources

The experimental setup mirrors that of Appendix A.1, with a minor modification in the context filter. Previously, the TopP selection strategy returned a list of relevant indices, which were directly mapped to context claims. In this updated approach, the filtered indices are sorted before mapping to ensure temporal order, so the context claim at index n precedes the context claim at index $n+1$. The results are presented in Table 7.

A.3 Experiment 0.3: Scoring with cosine similarity

The experimental setup mirrors that of Appendix A.2, but with a minor modification in the context filter. The vectorized context sentences and the formatted string are normalized to recreate cosine similarity for the dot product calculation. The results are presented in Table 7.

Columns 4 and 5 in Table 7 show that maintaining the temporal order of filtered context claims enhances NLI model accuracy, especially for conversation-based use cases, yielding a **24.95%** overall improvement in AUC scores.

Columns 5 and 6 in Table 7 show that using cosine similarity results in a better threshold for the NLI model, with an overall **4.79%** improvement in AUC scores.

Column 6 in Table 7 and Column 3 in Table 4 demonstrate that the Relevancy Scorer with the Context Item Selector outperforms simple cosine similarity between context and query, leading to a **9.63%** overall improvement in AUC scores.

[3] https://spacy.io/api/sentencizer

[4] Available on huggingface as WhereIsAI/UAE-Large-V1

[5] Available on huggingface as microsoft/deberta-v2-xxlarge-mnli