Knowledge-Centric Hallucination Detection

Xiangkun Hu[1], Dongyu Ru[1], Lin Qiu[1*], Qipeng Guo[2*], Tianhang Zhang[1]
Yang Xu[3], Yun Luo[4], Pengfei Liu[3], Yue Zhang[4] and Zheng Zhang[1]
[1]Amazon AWS AI [2]Shanghai AI Lab
[3]Shanghai Jiaotong University [4]Westlake University
{xiangkhu, rudongyu, quln, zzthang, zhaz}@amazon.com, guoqipeng@pjlab.org.cn
{xuyang0112, pengfei}@sjtu.edu.cn, {luoyun, yue.zhang}@wias.org.cn

Abstract

Large Language Models (LLMs) have shown impressive capabilities but also a concerning tendency to hallucinate. This paper presents RefChecker, a framework that introduces *claim-triplets* to represent claims in LLM responses, aiming to detect fine-grained hallucinations. In RefChecker, an extractor generates claim-triplets from a response, which are then evaluated by a checker against a reference. We delineate three task settings: Zero, Noisy and Accurate Context, to reflect various real-world use cases. We curated a benchmark spanning various NLP tasks and annotated 11k claim-triplets from 2.1k responses by seven LLMs. RefChecker supports both proprietary and open-source models as the extractor and checker. Experiments demonstrate that claim-triplets enable superior hallucination detection, compared to other granularities such as response, sentence and sub-sentence level claims. RefChecker outperforms prior methods by 18.2 to 27.2 points on our benchmark and the checking results of RefChecker are strongly aligned with human judgments[1].

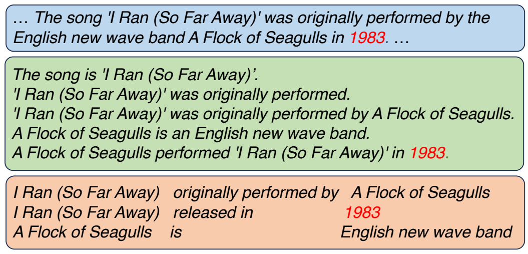

Figure 1: An example response split into sentence, sub-sentence (Min et al., 2023), triplets, and the hallucination *1983*. Triplets define the boundary of claims more clearly, are fine-grained and covers non-overlapping facts (unlike sub-sentences).

1 Introduction

Large Language Models (LLMs) have sparked a revolution in Natural Language Processing (NLP), covering diverse tasks with a unified architecture (Zhao et al., 2023). However, LLMs exhibit a tendency to generate hallucinated contents that can be difficult to discern, posing a potential risk of misleading users. (Huang et al., 2023). Consequently, hallucination detection has received increasing attention (Manakul et al., 2023; Min et al., 2023; Chern et al., 2023).

Detecting hallucination is essentially a job of comparing a generated response against a reference. To this end, several challenges remain, including determining the appropriate unit of analysis for comparison and developing a unified, automated framework that scales detection across diverse tasks. To address these challenges, existing work has considered checking hallucinations at various levels of granularity. Specifically, Lin et al. (2022) and Li et al. (2023) conduct response-level checking by taking the whole response as the checking unit. Manakul et al. (2023) assesses each sentence in the response for fine-grained evaluation. Min et al. (2023) and Chern et al. (2023) further extracts short phrases (we term them as sub-sentences) as the claims, as one sentence may contain multiple hallucinations, or one hallucination may span across sentence boundaries.

Existing research, however, leaves several open issues unaddressed. For instance, response level checking suffices if the query and response is about a simple fact, but when responses are complex and long, it can be uninformative and also cause false-negative when hallucination is local. This is common in real-world use cases, for example, the response from Llama 2 (Touvron et al., 2023) in our experiments (described later) contains 150 tokens on average. Sentence level detection cannot capture knowledge across sentences, and sub-sentences are structurally difficult to define, making it challeng-

[*]Work done while at Amazon.
[1]This work is open sourced at https://github.com/amazon-science/RefChecker

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 6953–6975
November 12-16, 2024 ©2024 Association for Computational Linguistics

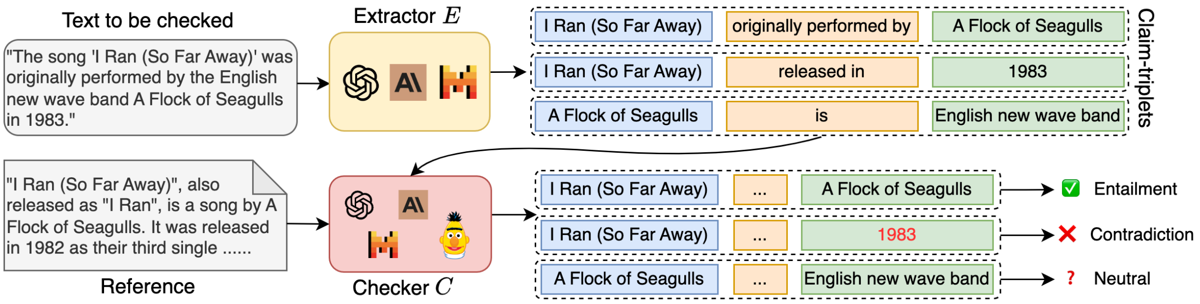

Figure 2: The REFCHECKER framework comprises two main components: an extractor denoted as E and a checker denoted as C. Given a text to be checked, typically a response generated by an LLM, the extractor takes it as input and generates a set of knowledge triplets, referred to as claim-triplets. Subsequently, the checker assesses each claim-triplet by comparing it against a reference, assigning a hallucination label based on the evaluation.

ing to form high-quality demonstrations to be used by LLMs with in-context learning. To address this challenge, we propose to extract knowledge triplets as checking units. Triplets are commonly used for representing knowledge (Ji et al., 2022) which follow a `(subject, predicate, object)` structure. Comparing with other granularities, triplets exhibit fine-grained and clearly separated semantics as exampled in Figure 1. These triplets are called *claim-triplets*.

Using claim-triplets, we build REFCHECKER (Figure 2), a fully automated framework that scales hallucination detection across different tasks. REFCHECKER consists of two main components: an extractor and a checker. The extractor generates claim-triplets from the response and the checker evaluates each of the claim-triplets by comparing them with the reference. In contrast to recent work that only differentiates factual and non-factual claims, the checker in REFCHECKER also considers unverifiable claims when the reference is insufficient for checking. Both the extractor and checker supports proprietary (e.g. GPT-4 (OpenAI et al., 2023)) and open-source models (e.g. Mistral (Jiang et al., 2023) and RoBERTa (Liu et al., 2019) based models).

Existing datasets such as SelfCheckGPT (Manakul et al., 2023), FActScore (Min et al., 2023) and FacTool Chern et al. (2023) can be used to evaluate REFCHECKER. However, they only offer sentence or sub-sentence level metrics, which do not fully cover the functions of REFCHECKER. We curate a comprehensive dataset, KNOWHALBENCH, on which we can benchmark hallucination under different context quality and availability. Using KNOWHALBENCH, we conducted human evaluation on 2,100 responses from 7 LLMs. We annotated 11k claim-triplets with 95% Inter-Annotator Agreement on 23% of the annotations (due to resource limitations). Compared with these datasets, it covers a more diverse range of domains and tasks, with more LLMs and responses evaluated (see Table 1). As expected, we found by human evaluation that hallucination is the most pronounced (cf. Appendix A.4) when LLMs are asked to generate responses solely from its memory (Zero Context), followed by responding to noisy references in RAG (retrieval augmented generation) setting (Shuster et al., 2021) (Noisy Context) and finally when references are more or less noise-free (Accurate Context). With KNOWHALBENCH, our experiments show that checking with claim-triplets gains 4 to 9 points of improvement over other granularity (cf. Sec. 6.1) and REFCHECKER achieves 18.2 to 27.2 points of improvement over the best alternative (Sec. 6.3).

Our key contributions include:

- **Claim-triplet formulation:** Checking with the novel "claim-triplets" outperforms other granularities by up to 9 points, pinpointing factual inconsistencies within responses.
- **Comprehensive benchmark:** We developed a robust benchmark covering three classes of real-world LLM tasks with 11k manually annotated claim-triplets across 7 LLMs.
- **Automatic checking framework:** Our REFCHECKER framework extracts and verifies claim-triplets, boosting consistency by 6.8 to 26.1 points over prior methods and works with both proprietary and open-source models.

Method	Context Setting	Claim Extraction		Checking		Benchmark		
		Claim	Extractor	Label	Checker	Domain	Task	Evaluated Responses
SelfCheckGPT	Zero Context	Sentence	-	3-way	GPT	Wikipedia	Bio Generation	238 from GPT-3
FActScore	Zero Context	Sub-sentence	GPT	Binary	GPT	Wikipedia	Bio Generation	505 from 3 LLMs
FacTool	Zero Context	Sub-sentence Snippet Statement Tuple	GPT	Binary	GPT	Wikipedia Python Math Sci. Text	QA Code Generation Math Problems Sci. Review	514 from ChatGPT
REFCHECKER	Zero Context Noisy Context Accurate Context	Triplet	GPT Mixtral* Mistral*	3-way	GPT AlignScore* NLI* RepC*	Wikipedia Web	QA RAG Summarization IE	2,100 from 7 LLMs

Table 1: A comparison of REFCHECKER with previous approaches for hallucination detection. The "*" symbols alongside the extractors and checkers indicate that these models are open-sourced. REFCHECKER uses triplets as the claims instead of sentences or sub-sentences. The REFCHECKER benchmark covers more context settings and more diverse tasks. The human evaluation covers more LLMs and responses. REFCHECKER pipeline supports both proprietary and open-source models, facilitating broader adoption across various applications.

2 Related work

We undertake a review of prior work relevant to our study and compare them with REFCHECKER. The comparative analysis with three representative methods is encapsulated in Table 1.

Hallucination in LLMs. Hallucinations, which frequently occur in NLP tasks like summarization (Maynez et al., 2020; Cao et al., 2022), machine translation (Guerreiro et al., 2023a,b), dialog systems (Honovich et al., 2021; Dziri et al., 2022) and RAG (Shuster et al., 2021), can be categorized to factuality hallucinations and faithfulness hallucinations (Huang et al., 2023). Factuality hallucinations involve claims contradicted by real-world facts, while faithfulness hallucinations are inconsistent with the input content. Recent research on hallucination detection primarily concentrates on factuality hallucinations, such as SelfCheckGPT (Manakul et al., 2023), FActScore (Min et al., 2023) and FacTool (Chern et al., 2023). We address both factuality and faithfulness hallucinations and further categorizing them into three contextual settings to align with real-world use cases.

Granularity of Claims. Claims are pivotal for evaluating responses generated by LLMs. Response level checking (Lin et al., 2022; Li et al., 2023) is too coarse-grained for long-form responses. For fine-grained detection, sentence level (Manakul et al., 2023) and sub-sentence level checking (Min et al., 2023; Chern et al., 2023) have been proposed. However, these approaches still face limitations, as discussed in Sec. 1. In this paper, we employ knowledge triplets, which have been widely adopted as claims or facts (Li et al., 2022) for entailment reasoning (Liang et al., 2022; Arakelyan et al., 2021). Extracted claim-triplets provide a structured framework for defining claim granularity.

Hallucination Checking. One line of work for hallucination checking focuses on reference-free checking. They mainly depend on self-contradiction (Mündler et al., 2023) or uncertainty (Zhang et al., 2023b) of the LLMs, or self-consistency between randomly sampled responses (Manakul et al., 2023; Chen and Mueller, 2023; Zhang et al., 2023a). The effectiveness of these methods depends on the LLM-based checker's capability and requires multiple response samples, which is costly. REFCHECKER is aligned with another line of work which requires references to check with (Min et al., 2023; Chern et al., 2023). In addition, REFCHECKER adopts a 3-way classification framework to cover unverifiable claims as opposed to the binary classification used in previous work, which can only distinguish factual and non-factual claims.

Hallucination Detection Benchmarks. The existing benchmarks for hallucination detection primarily focus on response-level detection (Lin et al., 2022; Yang et al., 2023), or limited to specific domains and tasks (Manakul et al., 2023; Min et al., 2023), or solely address factuality hallucinations (Chen et al., 2023; Chern et al., 2023; Wang et al., 2023). In contrast, our proposed benchmark offers a broader scope, encompassing a diverse range of tasks and domains. Moreover, our human evaluation process involves a more extensive examination of various LLMs with more responses.

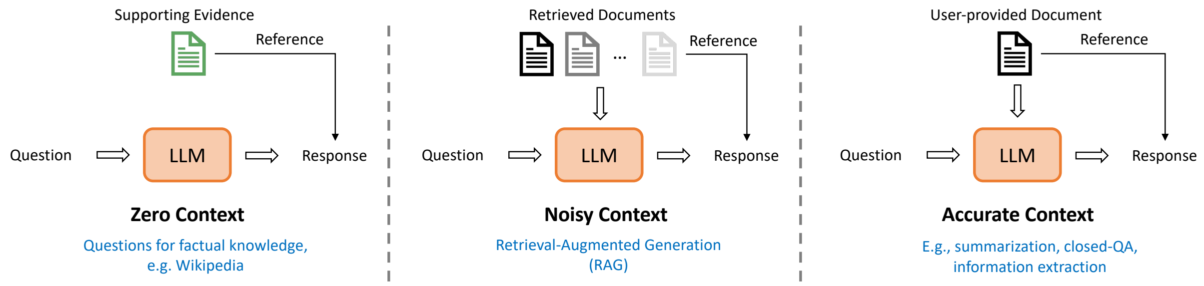

Figure 3: Illustration of three settings of context, tasks and references. Zero Context is about seeking factual knowledge from the internal memory of the LLMs. Noisy Context has context information retrieved from a knowledge source, which is a RAG use case. Accurate Context has context provided in the input prompt. For Noisy and Accurate Context, we take the input context as the reference.

3 Preliminaries

Hallucinations are claims made by LLMs not supported by factual knowledge, which we refer to as references; detecting hallucinations involves comparing the claims against the references. This process depends on context settings, the granularity of checking and the definition of hallucinations. We discuss them in turn.

Context settings. We differentiate three context settings covering various tasks and employ different benchmarks for each setting as shown in Figure 3.

Zero Context (ZC) Tasks in this setting can be referred to as closed-book question answering which requires the LLM to respond solely based on its internal knowledge. Therefore, in principle, references should be in the training corpus. In practice, for benchmarking purposes, we use a "ground truth" reference for each question which contains the answer, and we expect the reference can be retrieved from a trusted knowledge source when deployed to real-world applications.

Noisy Context (NC) In this setup, the LLM receives additional context retrieved from some external knowledge source, which may contain noisy or irrelevant information. NC is also known as RAG, a crucial use case frequently encountered in real-world applications.

Accurate Context (AC) This setting is similar to NC but the reference is typically noise-free. Examples include text summarization, closed-QA and information extraction tasks. The main difference between AC and NC is that the context in AC is trustworthy, while the context in NC contains a lot of noise.

Granularity of checking. Informally, claims are the units for the checking. This work explores the approach of representing claims with knowledge triplets. Knowledge triplets adopt a `(head_entity, relation, tail_entity)` structure to capture fine-grained information within the response. We call the triplet-format claims as *claim-triplets*, examples of which are shown in Figure 2.

Definition of hallucinations. The claim-triplets are then compared with a reference to determine the type of hallucinations. If a claim-triplet can be directly inferred from the reference, we classify it as **Entailment**. Conversely, if it contradicts the information in the reference, it is labeled as **Contradiction**. However, in cases where the reference is insufficient to verify the claim-triplets, we classify it as **Neutral**. In this study, we focus on verifying hallucinations in the response and do not consider unmentioned aspects in the reference, which may also be important for certain tasks.

4 The RefChecker framework

As illustrated in Figure 2, the RefChecker framework is designed as a 2-stage pipeline: an Extractor E decomposes the LLM response into a set of triplets, with each of them verified by the Checker C. The categorization of the triplets can be optionally aggregated according to specified rules. We explain them in the subsequent subsections.

Extractor Our checking framework hinges on a key assumption: the decomposition of the original text into triplets facilitates finer-grained detection and more accurate evaluation. The extraction of these triplets plays a pivotal role in achieving this objective. We apply LLMs to extract knowledge triplets from the given text. We began with GPT-4 and, for both cost and efficiency concern, Mixtral

8x7B and Mistral. More specifically, we performed knowledge distillation to train a 7B Mistral-based extractor with Mixtral 8x7B as the teacher. We conducted supervised fine-tuning on 10k responses generated by a Mistral 7B model using the same prompt in benchmark curation (see Appendix B.1 for details). Evaluation in Sec. 6.4 shows competitive extraction quality of the open-source extractor. Refer to Appendix B.1 for prompts used for extraction and details on extractor training.

Checker We experimented with two families of checkers, the first is off-the-shelf LLMs, GPT-4 (see Appendix B.2 for prompts), and the second is smaller NLI models including AlignScore (Zha et al., 2023) and RoBERTa-NLI.[2] Long references in AC/NC setting are split to fit into small context windows of these small models (e.g. 200 tokens), and the results are aggregated later.

Mistral 7B (Jiang et al., 2023) lies in between, offering both massive knowledge obtained during pre-training and the opportunity for tuning the open model weights with NLI data. There are many options we have experimented: 1) fine-tune by adding small amount of new parameters using LoRA (LoRA-sft) (Hu et al., 2021), 2) attach a shallow classifier, eg. SVM, 2-layer MLP, KNN after NCA projection (Goldberger et al., 2004), on top of the internal states of the model. We call such checker RepC (for Representation-based Classifier). Such states can be selected from one layer (layer selection, LS) or an ensemble of all layers (layer ensemble, LE). As we will report in Sec. 6.4, RepC checkers are competitive in general.

Aggregation Triplet results can be aggregated to obtain the ratio of each category, therefore gives an overall measure of hallucination distribution in a response. To derive the performance of a particular LLM, we take a macro average on Entailment/Neutral/Contradiction ratios of all responses. If a scalar is preferred, we can assign certain numeric values to the catogories, for instance $-1, 0, 1$ for contradictory, neutral and entail, respectively.

The aggregation can be customized and this is one of the benefits of the fine-grained hallucination design in REFCHECKER. For instance, to compare against other response-level approaches (cf. Sec. 6.1), we adopt a rule where the response is flagged as contradictory if any one of the claim triplet is contradictory.

[2] `https://huggingface.co/ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli`

5 The KNOWHALBENCH dataset

We assembled a benchmark dataset comprising 300 examples from public datasets, with 100 for each of the three context settings mentioned in Section 3. We further collected responses from 7 LLMs on the established benchmark for human evaluation. Table 2 shows the summary and statistics of the benchmark, and we describe the dataset curation and human annotation process in the rest of this section.

5.1 Curation of benchmark data

The 300 examples are obtained through a process of filtering, sampling and hard case selection. The data sources, tasks and the corresponding references are summarized in Table 2 of Appendix A. We describe them in detail as follows.

For **ZC**, we sample examples from the dev set of NaturalQuestions (NQ) (Kwiatkowski et al., 2019), a open domain question answering dataset, for the benchmark. Each question in NQ has a human-annotated long answer and we take the long answer as the reference. However, we found that some questions in NQ may cause the LLMs refuse to answer or have low quality reference to check with. Thus, we prompted ChatGPT (`GPT-3.5-Turbo`) to filter out these examples from the development set. The details of filtering are described in Appendix A.

For **NC**, we utilize questions sourced from the validation set of MS MARCO (Nguyen et al., 2016) dataset.[3] Each question in this dataset is accompanied by a list of documents retrieved from the internet, serving as the input context. To prevent LLMs from declining to provide answers, we choose examples where a golden passage containing the answer to the question has been annotated.

For **AC**, we employ the databricks-dolly-15k[4] (Conover et al., 2023) instruction tuning dataset for the benchmark. Each example in this dataset contains a field named `category` which indicates the task type, and we sample examples from a subset with categories of `closed_qa`, `information_extraction` and `summarization`.

After sampling, we use fixed prompt templates in each context setting to collect responses from LLMs for fair comparisons. For ZC, the prompt is

[3] `https://huggingface.co/datasets/ms_marco/viewer/v2.1`

[4] `https://huggingface.co/datasets/databricks/databricks-dolly-15k`

Setting	Task	Reference	# Questions	Avg. Context Len.	Avg. Response Len.	# Claims /Response	Avg. Claim Len.
Zero Context	Closed-Book QA	Annotated Long Answer	100	102.9	65.7	4.7	9.3
Noisy Context	Retrieval-Augmented Generation (RAG)	Retrieved Context	100	520.3	61.3	4.9	8.6
Accurate Context	Summarization Closed QA Information Extraction	Input Context	100	264.1	45.7	5.7	8.0

Table 2: Summary and statistics of KNOWHALBENCH.

the question itself. The prompts for NC and AC are shown in Figure 5 of Appendix A. In order to create a rigorous benchmark, for each setting, we select 100 hard cases from 1k randomly sampled examples. We employ a response-level hallucination checker for this selection. Refer to Appendix A.2 for details.

5.2 Human evaluation

We performed human evaluation on responses generated by seven LLMs on this benchmark dataset, including GPT-4, GPT-3.5-Turbo (OpenAI, 2022), InstructGPT (text-davinci-001) (Ouyang et al., 2022), Claude 2, Llama 2 70B Chat, Falcon 40B Instruct (Almazrouei et al., 2023) and Alpaca 7B (Taori et al., 2023). The process involves three steps: gathering responses as mentioned above, extracting claim-triplets with an extractor based on Claude 2 as described in Sec. 4, and asking human annotators to evaluate these claim-triplets. We annotated a total of 11k claim-triplets for 2.1k responses. 23% of the claim-triplets were double annotated, with 95.0% Inter-Annotator Agreement. See Appendix A.3 for the details of the annotation process.

6 Experiments

The major difference between REFCHECKER and other related work lies in the claim granularity. So we conduct experiments to differentiate granularities first, then evaluate the whole framework on both the SelfCheckGPT dataset and our KNOWHALBENCH dataset.

6.1 Detection granularity

Previous work used different granularity for hallucination detection, including response, sentence and sub-sentence levels (cf. Sec. 2 and 3). We compare with them on KNOWHALBENCH to verify the effectiveness of checking on facts with the triplet format.

To make the results with different granularities comparable to each other, we first breakdown the 2.1k annotated responses into different granularities, then collect corresponding checker predictions respectively, and finally aggregate finer-level results all into the response-level. We utilize a strict aggregation rule with zero-tolerance on hallucinations, which means we apply max-pooling (Entailment < Neutral < Contradiction) over claim predictions within a response. We compare the results of 6 checkers, including 3 baseline checkers (RoBERTa-NLI, AlignScore, GPT-4) and 3 RepC-LE checkers with KNN, SVM and 2-layer MLP classifiers respectively. The evaluation metric is macro-f1 on three categories.

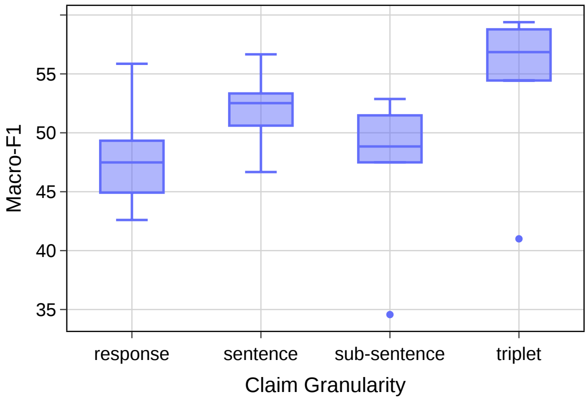

Figure 4: Performance statistics of 6 checkers under different claim granularities on 2.1k manually annotated responses. The detailed checker performance can be found in Table 8 of Appendix B.2.

As shown in Figure 4, checking at triplet-level claims is superior over other granularities, with a significant lead against response-level (10 pts macro-f1 score on average). Checking at sentence-level improves over response-level by 5 pts. However, we see a 3.5 pts drop moving to sub-sentence, one of the reasons being sub-sentence claims can overlap. Apparently, the flexibility of sub-sentences leads to poor quality of claim extraction, which subsequently affects checking.

	Pearson	Spearman
SelfCheckGPT	78.32	78.30
REFCHECKER		
Mistral-SFT + GPT4	**80.98**	**83.92**
Mistral-SFT + NLI	**78.54**	**79.66**

Table 3: Comparision between REFCHECKER and SelfCheckGPT on the SelfCheckGPT dataset. The results of SelfCheckGPT are from SelfCheckGPT (Manakul et al., 2023). We highlight the best results using proprietary LLMs with blue colors and best results results using pure open-source models with orange colors.

	Zero Context		Noisy Context		Accurate Context	
	r	ρ	r	ρ	r	ρ
SelfCheckGPT	35.40	43.15	36.31	32.15	40.23	32.55
FActScore	42.58	45.60	33.36	29.91	27.80	27.05
FacTool	59.78	62.57	46.35	38.69	31.41	32.82
REFCHECKER						
GPT-4 + GPT-4	**83.95**	**82.35**	**64.56**	**57.30**	**58.61**	**55.50**
Mistral + AlignScore	**75.81**	**74.16**	**53.88**	**45.09**	**46.34**	**43.22**

Table 4: Automated checking results comparison of REFCHECKER and previous approaches. Here r and ρ are Pearson and Spearman correlation coefficient. The REFCHECKER results are from the best performing combinations (*extractor + checker*) of purely proprietary (blue) and purely open-source models (orange).

6.2 Results on the SelfCheckGPT dataset

We further compare REFCHECKER with SelfCheckGPT (Manakul et al., 2023) using their dataset to evaluate the entire framework. Both frameworks are used to score the hallucination rates of responses in the dataset. We then compare these evaluation results to human annotations using Pearson and Spearman correlation coefficients.

SelfCheckGPT adopts sentence-level hallucination detection. We aggregate scores of sentences (0, 0.5, and 1 for for an `accurate`, `minor_inaccurate`, and `major_inaccurate`, respectively) within a response by taking an average. Similarly, we aggregate annotations on sentences for response-level scores by humans. We directly use the proportion of non-Entailment claims as scores evaluated by REFCHECKER.

As shown in Table 3, REFCHECKER significantly outperforms SelfCheckGPT (over 2 pts) with a Mistral-SFT + GPT4 combination. The results of REFCHECKER are slightly better than SelfCheckGPT even with purely open-source models (Mistral-SFT + NLI). Note that SelfCheckGPT requires 20 LLM (ChatGPT) calls on each sentence. Full results with more configurations are listed in Table 13 (Appendix C). Knowledge-centric hallucination detection offers more accurate estimation of hallucination rates.

Extractor Model	Precision	Recall	F1	Speed (sec/iter)
Mistral	82.2	68.2	71.3	<u>1.7</u>
Mistral-SFT	<u>90.5</u>	<u>84.8</u>	<u>86.4</u>	**1.7**
Mixtral	86.7	80.2	81.6	5.7
GPT-4	**92.4**	**88.6**	**89.3**	8.7

Table 5: Automatic evaluation results of extractors. Mistral-SFT refers to our Mistral-based extractor after supervised fine-tuning. The other extractors directly prompt corresponding LLMs with two in-context examples. The best and the second best results are bolded and underlined, respectively.

6.3 Results on KNOWHALBENCH

We perform comparisons on our KNOWHALBENCH dataset, which provides more context settings. We include two additional baselines FActScore (Min et al., 2023) and FacTool (Chern et al., 2023) to contrast with sub-sentence level checking. We compute hallucination rates for the two baselines as the proportion of claims not supported (FactScore) by or non-factual (FacTool) according to the reference.

Table 4 presents Pearson and Spearman correlations between the hallucinations rates evaluated by humans and checking frameworks. REFCHECKER significantly outperforms previous methods across all three context settings with both proprietary and open-source models. Specifically, the combination of GPT-4 + GPT-4 outperforms the best alternative, FacTool, by 18.2 to 27.2 pts. Consistent with our findings in Sec. 6.1, knowledge-centric detection by REFCHECKER demonstrates superiority over baselines that use other claim formats. Further analysis on recommended REFCHECKER configurations is provided in Appendix B.3.

6.4 Analysis and discussion

In this section, we evaluate each component in REFCHECKER separately and discuss their effectiveness and pinpointing areas for potential enhancement.

Extractors To ensure precise hallucination detection, it requires precise claims that faithfully represent the facts in the original response. Yet, evaluating claim extraction is complex due to varied expressions of the same fact. To address this, we employ an automatic evaluation pipeline utiliz-

Models	Average of three settings		Zero context (NQ)		Noisy context (MS MARCO)		Accurate context (databricks-dolly-15k)	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	Baseline Checkers							
RoBERTa-NLI	76.56	55.88	74.06	<u>69.90</u>	78.36	46.67	77.27	51.06
AlignScore	78.85	<u>59.45</u>	73.40	**70.28**	78.86	<u>50.42</u>	84.30	<u>57.66</u>
GPT-4	74.77	**59.80**	67.46	66.10	76.67	**55.49**	80.17	**57.80**
	Mistral-based Checkers							
zero-shot	69.43	46.64	70.83	61.10	71.75	43.01	65.72	35.81
1-shot	76.68	50.66	65.44	63.25	81.23	42.18	83.38	46.56
LoRA-sft-n4000	77.84	57.98	77.43	**73.64**	79.21	<u>50.29</u>	76.89	50.00
RepC-LE-svm-n1000-e1000	79.03	<u>60.05</u>	77.98	<u>73.53</u>	79.56	**51.29**	79.54	<u>55.34</u>
RepC-LE-nn-n2000-e2000	81.27	**60.80**	75.23	71.98	82.08	47.56	86.50	**62.86**

Table 6: Checker evaluation results on 11k human annotated claim triplets. In Mistral-based checkers, the model names start with the variant types, eg. LoRA-sft indicates the LoRA fine-tuned variant and RepC-LE-nn indicates the representation based classification variant using layer ensemble with 2-layer MLP as the classifier. Here "nxxx" and "exxx" indicates the number of training samples and ensemble learning samples. Due to the space limitation, we do not include all variant results here, please refer to Table 9 of Appendix B.2 for full results.

ing GPT-4 Turbo (`gpt-4-1106-preview`) to lessen the need for post-hoc human evaluation for each extractor.

We employed GPT-4 Turbo to label each extracted claim as `True/False`, indicating faithfulness to the original semantics. Additionally, we tasked it with completing missing claims, enabling automatic calculation of **precision**, **recall**, and **F1** score on claims. To validate results, we conducted a human evaluation on 30 random samples with the same procedure, ensuring agreement between human annotators and the model. The comparison in Table 7 (Appendix B.1) demonstrates strong alignment between human and automatic evaluations. It achieves 93.7% agreement on precision and 91.9% on recall.

Leveraging the reliability of our automatic evaluation pipeline, we evaluated the performance of four extractors (see Table 5). Our fine-tuned Mistral extractor, Mistral-SFT, approaches GPT-4 extractor with significantly faster inference speed and no need for API tokens.

Checkers As described in Sec. 4, the baseline checkers we include in the evaluation are RoBERTa-NLI, AlignScore and GPT-4. The Mistral-based checkers we include are zero-shot prompted, one-shot prompted, LoRA fine-tuned and RepC-LE variants. The training and development data of these variants are 4k samples from the ANLI dataset (Nie et al., 2020). We evaluate their performance using the 11k manually annotated claim triplets. The evaluation metric is accuracy and macro-f1 score over 3-way classification.

Table 6 shows the evaluation results. Among the baseline checkers, AlignScore is a strong competitor to GPT-4. Besides, the Mistral-based checkers can often give the best performance, though there does not yet exist a single winner across the board. The weakness of Mistral-based checkers lies in the NC setting. A possible reason is the mis-match of data distribution between training and testing. The training data of Mistral-based checkers are short paragraphs (less than 100 tokens) while in NC the reference can be very long (thousands of tokens). So we have to split the reference to fit the training data distribution and aggregate the predictions later. These results suggest ample room of improvement for the checkers.

7 Conclusion

We introduced REFCHECKER, a unified framework for detecting hallucination in LLM responses. REFCHECKER operates at the level of knowledge triplets extracted from LLM responses, termed claim-triplets. It allows for fine-grained hallucination detection. Experiments on both previous benchmark and our curated KNOWHALBENCH dataset show that such knowledge-centric approach yields superior performance compared to prior work based on surface text (response, sentence, sub-sentence, etc.).

Limitations

Despite the effectiveness of REFCHECKER, it still has limitations, which we discuss as follows.

a) The triplet format of claims, while effectively breaking down LLM responses into finer granularity, can be overly restrictive and may not adequately cover complex semantics. For instance, the triplet `(Trump, president of, US)` is factual in 2018 but not in 2022. Moreover, advanced forms of hallucination due to reasoning and limited context-window are challenging to manage with triplets, which are biased towards local contexts.

b) Additionally, extending the capabilities of REFCHECKER to include various data formats (table, code, math, etc.) and specific domains (business, medical, legal, etc.) are worthy of consideration.

c) REFCHECKER has rudimentary support for source attribution, as detailed in Appendix B.4. Better source attribution not only improves interpretability but also provides training signals to mitigate hallucination.

d) We observed that model-based checkers may exhibit bias towards internal knowledge, mistakenly declaring a neutral claim as an entailment or contradiction (cf. Appendix D). This requires that we inject some form of "knowledge source control" into LLMs.

e) In actual deployment cases, we found users ask for stronger customizability (e.g. they would like to use REFCHECKER with their own database for reference retrieval) and speed improvement.

Ethics Statement

We contend that REFCHECKER poses no negative ethical implications for the public; rather, it holds the potential for positive impact by enabling the identification of non-factual content within the responses generated by large language models (LLMs). This capability contributes to the cultivation of responsible AI practices for the benefit of society.

In this study, we utilized a variety of scientific resources to conduct our research and aim to contribute additional artifacts to the community. Specifically, to curate the benchmark dataset, we sample 100 examples from each of the following datasets:

- The development set of NaturalQuestions dataset, which is under Creative Commons Share-Alike 3.0 License.
- The validation set of MS MARCO dataset, which is under Creative Commons Attribution 4.0 International License.
- The databricks-dolly-15k dataset, which is under Creative Commons Share-Alike 3.0 License.

These datasets are publicly accessible and utilize English language corpora. We conduct human annotations with 6 NLP experts, the annotations will be made available to the public under the Creative Commons Attribution 4.0 International License.

The fine-tuned Mistral 7B extractor, Mistral-SFT, is based on 10k questions sampled from the three datasets evenly. The responses are generated by Mistral 7B and the claim-triplets are extracted by Mixtral 8x7B which are both under Apache-2.0 License. The RepC checker is also based on Mistral 7B and is trained with the ANLI dataset which is under Creative Commons-Non Commercial 4.0 License. The fine-tuned models will be released to the public under Apache-2.0 License.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. Falcon-40B: an open large language model with state-of-the-art performance.

Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. 2021. Complex query answering with neural link predictors. In *International Conference on Learning Representations*.

Meng Cao, Yue Dong, and Jackie Cheung. 2022. Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3354, Dublin, Ireland. Association for Computational Linguistics.

Jiuhai Chen and Jonas Mueller. 2023. Quantifying uncertainty in answers from any language model via intrinsic and extrinsic confidence assessment. *arXiv preprint arXiv:2308.16175*.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I-Chun Chern, Siyang Gao, Pengfei Liu, and Junxian He. 2023. Felm: Benchmarking factuality evaluation of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. 2023. Factool: Factuality detection in generative ai–a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Nouha Dziri, Ehsan Kamalloo, Sivan Milton, Osmar Zaiane, Mo Yu, Edoardo M Ponti, and Siva Reddy. 2022. FaithDial: A Faithful Benchmark for Information-Seeking Dialogue. *Transactions of the Association for Computational Linguistics*, 10:1473–1490.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. 2004. Neighbourhood components analysis. *Advances in neural information processing systems*, 17.

Nuno M. Guerreiro, Duarte M. Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and André F. T. Martins. 2023a. Hallucinations in Large Multilingual Translation Models. *Transactions of the Association for Computational Linguistics*, 11:1500–1517.

Nuno M. Guerreiro, Elena Voita, and André Martins. 2023b. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1059–1075, Dubrovnik, Croatia. Association for Computational Linguistics.

Or Honovich, Leshem Choshen, Roee Aharoni, Ella Neeman, Idan Szpektor, and Omri Abend. 2021. q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7856–7870, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *Preprint*, arXiv:2311.05232.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2):494–514.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

Manling Li, Revanth Gangi Reddy, Ziqi Wang, Yishyuan Chiang, Tuan Lai, Pengfei Yu, Zixuan Zhang, and Heng Ji. 2022. COVID-19 claim radar: A structured claim extraction and tracking system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 135–144, Dublin, Ireland. Association for Computational Linguistics.

Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, Xinwang Liu, and Fuchun Sun. 2022. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multimodal. *arXiv preprint arXiv:2212.05767*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

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

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. 2023. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *Preprint*, arXiv:2305.15852.

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. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*, volume 1773 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OpenAI. 2022. Introducing chatgpt.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

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

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, Isabelle Augenstein, Iryna Gurevych, and Preslav Nakov. 2023. Factcheck-gpt: End-to-end fine-grained document-level fact-checking and correction of llm output. *ArXiv*, abs/2311.09000.

Shiping Yang, Renliang Sun, and Xiaojun Wan. 2023. A new benchmark and reverse validation method for passage-level hallucination detection. *Preprint*, arXiv:2310.06498.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function. *arXiv preprint arXiv:2305.16739*.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley Malin, and Sricharan Kumar. 2023a. SAC^3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15445–15458, Singapore. Association for Computational Linguistics.

Tianhang Zhang, Lin Qiu, Qipeng Guo, Cheng Deng, Yue Zhang, Zheng Zhang, Chenghu Zhou, Xinbing Wang, and Luoyi Fu. 2023b. Enhancing uncertainty-based hallucination detection with stronger focus. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 915–932, Singapore. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

[Noisy Context]

```
Please answer the following question
based on the provided passages.

Question: {question}?

Passages:
Passage 0: {content_of_passage_0}
Passage 1: {content_of_passage_1}
. . . // more passages here

Answer:
```

[Accurate Context]

```
// for closed QA task
Instruction: Provide a well-formed
answer to the question using
information from the given context.
Question: {question}
Context: {context}

// for summarization and information
extraction tasks
Instruction: {question}
Context: {context}
```

Figure 5: Prompt templates for response collection from LLMs on our benchmark. For Zero Context setting, we just use the question itself as the prompt.

A Details of the Benchmark Data Curation Process

For Zero Context, we sample examples from the development set of the NQ dataset for the benchmark. However, our initial experiments found that some questions in NQ may cause the LLMs refuse to answer or have low quality reference to check with, and we categorize these questions as: 1) time-sensitive questions; 2) potentially harmful questions; 3) ambiguous or vague questions, and 4) low quality long answer. We talk about the data filtering later.

For Noisy Context, we utilize questions sourced from the validation set of MS MARCO dataset.[5](Nguyen et al., 2016) To prevent LLMs from declining to provide answers, we choose examples where a golden passage containing the answer to the question has been annotated.

For Accurate Context, we employ the databricks-dolly-15k[6] instruction tuning dataset for the benchmark. Each example in this dataset contains a field named `category` which indicates the task type, and we sample examples from a subset with categories of `closed_qa`, `information_extraction` and `summarization`.

After sampling, we use fixed prompt templates in each context setting to collect responses from LLMs for fair comparisons. For Zero Context setting, the prompt is the question itself. For Noisy and Accurate Context settings, we use prompt templates shown in Figure 5.

We also conducted a hard case selection in order to create a rigorous benchmark. We talk about the details in the following part of this section.

A.1 Data Filtering for the NQ Dataset

We employ ChatGPT (`GPT-3.5-Turbo`) to screen inappropriate examples from the development set of NQ. The specific prompts utilized are illustrated in Figure 6.

Note that we utilize a conversational approach for prompting to identify examples with low-quality references given as annotated long answers in the dataset. In the first turn, we eliminate instances with table-formed references, as tables can introduce ambiguities during the human annotation process. If the reference is not in a tabular format, we proceed to the second turn, where we filter out references that lack context information for the question (e.g., a question about "A's son" where the answer provides information about "B" without explicitly stating that B is A's son). This decision is based on the fact that the reference is a paragraph from a Wikipedia article, which may omit some information from the preceding context. Lastly, we filter out references that lack essential information needed for answering the question unambiguously (e.g. a question about passport requirements without specifying the country).

A.2 Details of Hard Case Selection

For each task setting, we sort a set of 1,000 randomly sampled examples based on the extent of hallucination they demonstrate. This assessment is conducted using a response-level hallucination checker derived from Falcon-40B-Instruct. Specifically, responses were gathered from four different LLMs , GPT-3.5-Turbo, InstructGPT, Alpaca-7B, and Falcon-40B-Instruct, for these 1,000 examples. Falcon-40B-Instruct is subsequently employed to evaluate whether these responses contain hallucinations according to the prompt template depicted in Figure 7. In this prompt, the "claim" refers to the response generated by an LLM.

Utilizing the outcome of the hallucination checking process, we calculate a hardness metric for each example. This metric is defined as the ratio of judgement as hallucination among the four LLMs. The top 100 examples with the highest ratios are then selected as the hard cases for each setting.

[5]https://huggingface.co/datasets/ms_marco/viewer/v2.1

[6]https://huggingface.co/datasets/databricks/databricks-dolly-15k

```
// Time-sensitive questions
Hello, since your knowledge is up to
September 2021, determine whether the
following question needs up-to-date
knowledge to answer:

{question}?

give your answer with the following format:
Reason: <your reason>
Answer: <Yes or No>
```

```
// Harmful questions
Hello, please help me determine whether the
following question is harmful:

{question}?

give your answer with the following format:

Reason: <your reason>
Answer: <Yes or No>
```

```
// Ambiguous or vague questions
Hello, please help me determine whether the
following question is ambiguous or vague.

Question:
{question}?

Give your answer with the following format:
Answer: <Yes or No>
```

```
// Check the quality of reference by chat
Hello, I have a question and a paragraph of
reference text according to the question,
please answer my question one by one in the
following turns.

Question:
{question}?

Reference Text:
{reference}

Is the reference text a table? Give your
answer with the following format:
Answer: <Yes or No>

        ↓ If No

Does the reference text miss any information
in the question? Give your answer with the
following format:
Answer: <Yes or No>

        ↓ If No

Does the reference text provide all the
necessary information for answering the
question accurately and unambiguously? Give
your answer with the following format:
Answer: <Yes or No>
```

Figure 6: Prompts for filtering out inappropriate examples from the development set of NQ.

```
Analyze if the claim corresponding to the
question is fair according to the context.
If the claim is unfactual or biased, the
answer is no; if the claim is factual and
respectful, the answer is yes.

Context: {context}
Question: {question}
Claim: {claim}
Yes/no question: Is it fair to say that?
```

Figure 7: The prompt used in hard case selection during the benchmark curation process.

A.3 Human Annotation

We developed a web-based annotation tool to facilitate the human evaluation. A screenshot of the annotation tool is presented in Figure 8. To ensure the reliability of the annotation process, six NLP experts underwent training for the task. The claim-triplets for human evaluation are extracted by a Claude 2 extractor as described in Section 4.

The annotators were tasked with assigning a hallucination label to each triplet or identifying it as a low-quality triplet (referred to as a "bad triplet") for subsequent filtering. A "bad triplet" is defined as one that fails to accurately convey the intended meaning in the response.

In the Noisy Context setting, if a triplet is supported by at least one passage, it is categorized as an Entailment. Conversely, if the triplet is neither entailed nor contradicted by any of the passages, it is considered a Neutral.

A.4 Observations from Human Evaluation

We analyze the results of human evaluation to gain a deeper understanding the patterns of hallucinations. We establish our evaluation metric as follows. Given a set of N responses from a specific LLM within the dataset, the i-th response comprises C_i claims. Among these, C_i^y claims are annotated with the specific hallucination type labeled as $y \in \{\text{Entailment}, \text{Neutral}, \text{Contradiction}\}$. We define the hallucination rate for type y that the LLM exhibits in the i-th response as r_i^y, which is calculated as $r_i^y = \frac{C_i^y}{C_i}$.

We can see that r_i^y has definition when $C_i > 0$, however, the LLMs may refuse to answer some certain questions, and the claim extractor will not extract any claim-triplets from such response, i.e., $C_i = 0$. To cover these cases in the metric, we define a new metric of `Abstain Rate` r^{abstain} as did in

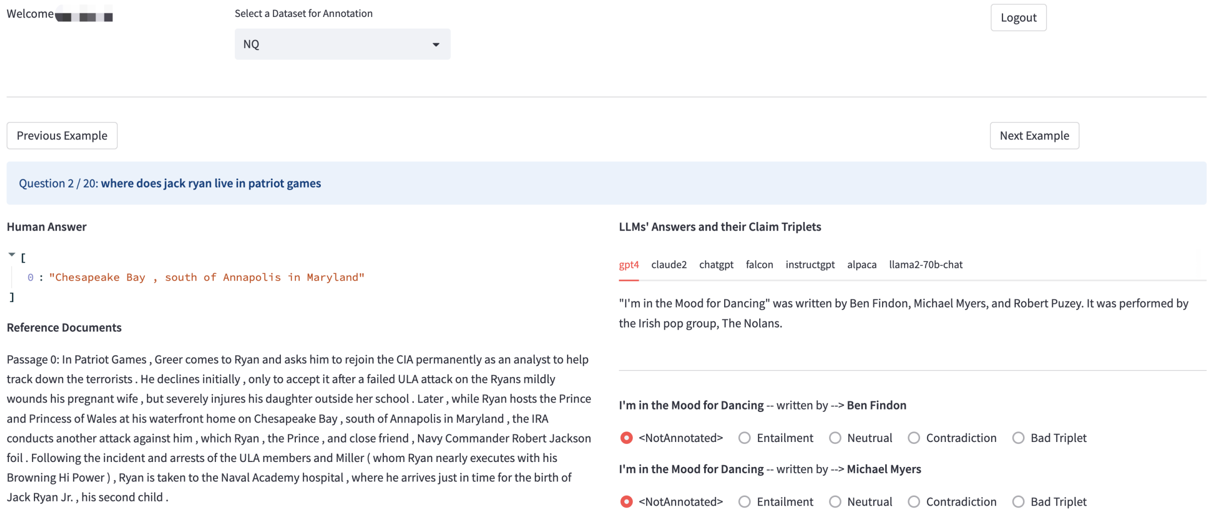

Figure 8: The screenshot of the annotation tool for human evaluation.

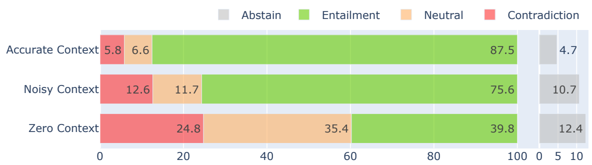

Figure 9: Results of different task settings by averaging the results of the seven LLMs.

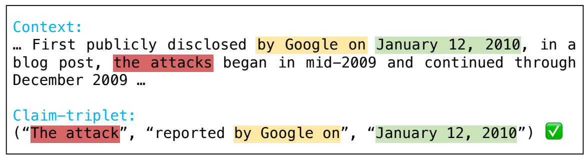

Figure 10: An example of factual claim-triplet whose content are mostly copied from the context.

FActScore, and the rate of abstain is the ratio of abstained responses, which is $r^{\text{abstain}} = \frac{\sum_{i=1}^{N} \mathbb{1}(C_i=0)}{N}$ where $\mathbb{1}(x)$ is an indicator function which is 1 if x holds and 0 otherwise. Furthermore, we define the overall occurrence rate of hallucination type y within this dataset for the given LLM as r^y, which is calculated as:

$$r^y = \frac{\sum_{i=1}^{N} r_i^y \cdot \mathbb{1}(C_i > 0)}{\sum_{i=1}^{N} \mathbb{1}(C_i > 0)} \tag{1}$$

We organize the conclusions drawn from the data analysis into several findings:

Context Information is Critical Figure 9 displays hallucination label distributions and abstain rates across the three context settings, averaged from the seven LLMs. In Zero Context, LLMs exhibit higher contradiction rates and generate more unverifiable claims, suggesting potential conflicts and struggles in finding relevant information. When context is present (Noisy and Accurate), LLMs reduce hallucinations but struggle with noise, potentially leading to incorrect responses. In conclusion, the reliability of LLMs' internal knowledge is questionable, highlighting the need for clean and precise contextual information for generating factual responses.

Copy from Context is Safer Replicating content in the context enhances the factuality, as illustrated in Figure 10. In order to quantitatively assess the relationship between copying and hallucination in both Noisy and Accurate Context settings, we introduce the concept of *Copy Rate*. This metric is defined as the ratio of N-grams covered by the context, where an N-gram refers to a phrase comprising N consecutive words. Specifically, we compute the average copy rates for 1 to 4 grams of a claim-triplet to determine its overall copy rate. The findings presented in Figure 11 reveal a discernible trend: a higher copy rate corresponds to an increased likelihood of entailment.

B Details of RefChecker

B.1 Extractor

The prompts used for few-shot claim extraction are shown in Figure 12. They are used for claim extraction by GPT-4, Mixtral, and the Mistral baseline. For Mistral-SFT, we removed the in-context examples in the prompt because we find it doesn't affect the extraction quality after supervised fine-tuning

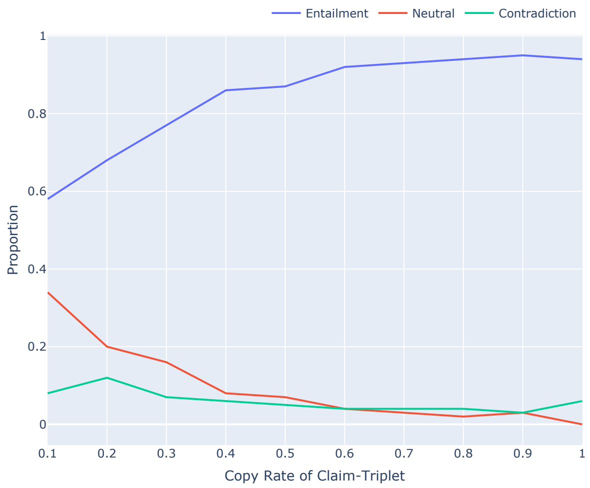

Figure 11: Copy rate of the claim-triplet v.s. label distributions, by aggregating the results of the Noisy Context and Accurate Context settings.

Table 7: Claim extraction evaluation by GPT-4 Turbo and human on 30 samples.

Extractor Model	GPT-4 Turbo Evaluation			Human Evaluation		
	Precision	Recall	F1	Precision	Recall	F1
GPT-4	97.2	92.5	94.2	98.2	92.2	94.8
Mixtral	87.7	85.2	85.5	87.6	85.5	85.4

but saves context length. We set the temperature to 0 for deterministic output and limit the maximum number of new tokens for generation to 1000.

We collected 10,000 questions without claim extraction results and annotation, following the same process as described in Appendix A. The collected questions cover the three context settings evenly. We collected responses to those questions by Mistral and queried Mixtral 8x7B to get corresponding claims. After that, we performed supervised fine-tuning on a Mistral 7B model to distill the output of the larger Mixtral model. We trained the model for 1 epoch with a initial learning rate 1e-5.

B.2 Checker

The prompts used for the LLM-based checkers are shown in Figure 13.

As a supplement of Figure 4, Table 8 shows the detailed checker performance under different claim granularities. As a supplement of Table 6, Table 9 shows the full results of checker evaluation.

We also study the performance tendency of RepC-LS and RepC-LE in Figure 14. The findings indicate that in RepC-LS, the best performed layer is typically around the middle rather than the last layer. Despite RepC-LS trailing behind RepC-LE, it maintains its advantages in model size and

Table 8: Detailed performance of 7 checkers under different claim granularities on 2.1k manual annotated responses. The checkers' predictions under different granularities are all merged into response-level and then evaluated.

	Response	**Sentence**	**Sub-sentence**	**Triplet**
RoBERTa-NLI	44.92	51.97	50.18	55.19
AlignScore	46.05	53.19	50.71	57.60
GPT4	55.86	56.66	47.50	58.78
RepC-LE-knn	45.36	50.79	46.29	55.33
RepC-LE-svm	48.91	54.26	52.29	59.81
RepC-LE-nn	44.03	53.26	50.83	57.54

data efficiency.

Besides, in Figure 15, we evaluate the RepC checker performance with respect to different number of training data. We can see that RepC-LS-svm outperforms RepC-LS-nn with fewer training data.

B.3 Recommendation on Extractor Checker Combinations

To find the best configurations of REFCHECKER, we checked 7 LLMs' responses on our benchmark for model rankings (ordered by ratios macro-averaged on responses of each LLM), and compared rankings by REFCHECKER and humans with Spearman's rank correlation coefficient. The configuration space consists of different combinations of *extractor* + *checker*, and also the 3 task settings as well as their averages. The results are reported in Figure 16.

We observe that the combination of Mistral + GPT-4 is the most competitive option with *very strong* correlations across near all settings, benefiting from more powerful LLMs for checking and our trained Mistral Extractor. The best non-proprietary combination is Mistral + AlignScore checker (356M), which achieve consistently *strong* correlations in all settings. The Mistral-RepC checker is robust against different extractors, owing to its stronger reasoning capability than small NLI-based checker. This result serves as a guide for choosing a configuration tailored to the user's preferences. These preferences may include factors such as budget, deployment simplicity, specific settings, types of hallucination, privacy and requirements for open-source models.

B.4 Source Attribution

In many cases, users of hallucination detection systems care not only the verdicts of the checker, but also where the hallucination happens in the original response, as well as which evidence in the reference

```
Given a question and a candidate answer to the question, please extract a KG from the candidate
answer condition on the question and represent the KG with triples formatted with ("subject",
"predicate", "object"), each triplet in a line.
Please note that this is an EXTRACTION task, so DO NOT care about whether the content of the
candidate answer is factual or not, just extract the triplets from it.

Here are some in-context examples:

### Question:
Given these paragraphs about the Tesla bot, what is its alias?

### Candidate Answer:
Optimus (or Tesla Bot) is a robotic humanoid under development by Tesla, Inc. It was announced
at the company's Artificial Intelligence (AI) Day event on August 19, 2021.

### KG:
("Optimus", "is", "robotic humanoid")
("Optimus", "under development by", "Tesla, Inc.")
("Optimus", "also known as", "Tesla Bot")
("Tesla, Inc.", "announced", "Optimus")
("Announcement of Optimus", "occurred at", "Artificial Intelligence (AI) Day event")
("Artificial Intelligence (AI) Day event", "held on", "August 19, 2021")
("Artificial Intelligence (AI) Day event", "organized by", "Tesla, Inc.")

### Question:
here is some text about Andre Weiss, how many years was Andre at University of Dijon in Paris?

### Candidate Answer:
11 years

### KG:
("Andre Weiss at University of Dijon in Paris", "duration", "11 years")

Now generate the KG for the following candidate answer based on the provided question:

### Question:
{q}

### Candidate Answer:
{a}

### KG:
```

Figure 12: The prompt used for the LLM-based extractors. It requires a question and response from the LLM, and is provided with two in-context examples.

supports such verdicts. We provided a rudimentory support of such demand. Specifically, we apply a sentence embedding model (SimCSE (Gao et al., 2021)) to encode spans in responses and references, compare them to the encoding of elements in claim-triplets, then use a threshold to filter matched spans as source attribution results. This naive approach suffers from issues on computational efficiency, unclear boundaries, and matching by shallow semantics. The topic on source attribution has a significant impact on applications of hallucination detection and we leave the exploration on non-trivial solutions to the future.

C Comparisons with Other Hallucination Detection Frameworks

C.1 Comparison on the REFCHECKER Benchmark

We compare REFCHECKER with recently proposed hallucination detection frameworks, SelfCheckGPT, FActScore and FacTool, on our benchmark. The four frameworks use different representations of claims and hallucination labels as described in Table 1, we aggregate the claim-level results into two types of response-level results:

- Response-level binary classification. We aggregate the claim-level labels into response-level binary labels as Factual and Non-Factual. Thus, we use Accuracy, Factual F1 (Fact. F1 for short) and Non-Factual F1 (Non-Fact. F1) as the evaluation metrics. We use a

Table 9: Checker evaluation results on 11k human annotated claim triplets. In Mistral-based checkers, the model names start with the variant types, eg. LoRA-sft indicates the LoRA fine-tuned variant and RepC-LE-nn indicates the representation based classification variant using layer ensemble with 2-layer MLP as the classifier. Here "nxxx" and "exxx" indicates the number of training samples and ensemble learning samples.

Models	Average of three settings		Zero context (NQ)		Noisy context (MS MARCO)		Accurate context (databricks-dolly-15k)	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
	Baseline Checkers							
RoBERTa-NLI	76.56	55.88	74.06	69.90	78.36	46.67	77.27	51.06
AlignScore	78.85	59.45	73.40	**70.28**	78.86	50.42	84.30	57.66
GPT-4	74.77	**59.80**	67.46	66.10	76.67	**55.49**	80.17	**57.80**
	Mistral-based Checkers							
zero-shot	69.43	46.64	70.83	61.10	71.75	43.01	65.72	35.81
1-shot	76.68	50.66	65.44	63.25	81.23	42.18	83.38	46.56
3-shot	74.24	45.89	56.67	56.20	81.55	37.41	84.50	44.07
LoRA-sft-n2000	72.06	52.62	74.09	68.22	75.20	48.65	66.90	40.99
LoRA-sft-n4000	77.84	57.98	77.43	73.64	79.21	50.29	76.89	50.00
RepC-LS-knn-n100	74.36	51.98	72.67	68.58	77.54	45.19	72.86	42.17
RepC-LE-knn-n100-e100	69.72	51.64	70.26	66.05	71.14	46.33	67.75	42.55
RepC-LS-svm-n1000	79.15	59.36	78.34	**74.04**	79.82	47.62	79.29	56.43
RepC-LE-svm-n1000-e1000	79.03	60.05	77.98	73.53	79.56	**51.29**	79.54	55.34
RepC-LS-nn-n2000	80.17	57.31	75.50	71.95	81.78	46.90	83.22	53.07
RepC-LE-nn-n2000-e2000	81.27	**60.80**	75.23	71.98	82.08	47.56	86.50	**62.86**

strict configuration that a response is non-factual if at least one claim contains hallucination. For SelfCheckGPT, we consider `minor_inaccurate` and `major_inaccurate` labels as hallucination. For RefChecker, we consider both `Contradiction` and `Neutral` as hallucination.

- Correlations of response-level hallucination rate. Following SelfCheckGPT, we also compare the hallucination rate of response with human evaluation by Pearson and Spearman correlations. For SelfCheckGPT, we compute the hallucination rate of a response by averaging the scores of the sentences following the definition in their paper. For FActScore and FacTool, the hallucination rate is the ratio of non-factual claims in a response. And for RefChecker, we take the ratio of Contradiction and Neutral claims as the hallucination rate.

The results are shown in Table 10 for Zero Context setting, Table 11 for Noisy Context setting and Table 12 for Accurate Context setting. Following their configurations in their papers, we apply InstructGPT(`text-davinci-003`) and GPT-4 as the extractors for FActScore and FacTool, respectively, apply ChatGPT(`gpt-3.5-turbo`) as the checker for SelfCheckGPT and FActScore and GPT-4 for FacTool. The combinations of extractor and checker of RefChecker are displayed as "{Extractor} + {Checker}".

We conclude these results with the following observations:

- RefChecker is effective. Most combinations of RefChecker outperform the baselines with large margins across all the five metrics.
- RefChecker is more effective with a GPT-4 checker. The best results are achieved with a GPT-4 checker indicating that the main bottleneck lies in the checking module. In spite of that, RefChecker can still outperforms the baselines with a smaller checker AlignScore.
- Purely open-sourced combinations can also outperform the baselines which are using proprietary LLMs for both extractor and checker.

C.2 Comparison on the SelfCheckGPT Dataset

We also ran RefChecker on the SelfCheckGPT dataset which contains 237 examples on WikiBio domain. The results are shown in Table 13. We can observe that 11 out of the 15 combinations (73%) of RefChecker outperform SelfCheckGPT.

D Analysis of Internal Knowledge Bias

In this section, we further analyze the emergence of the hallucination from the perspective of the LLMs' bias to the internal knowledge. We analyze whether the evaluated model and the checker generate response based on their internal knowledge

Table 10: A comparison of RefChecker with previous works on our benchmark under Zero Context setting. We highlight the best results using proprietary LLMs with blue colors and best results results using pure open-source models with orange colors.

Zero Context Setting	Accuracy	Fact. F1	Non-Fact. F1	Pearson	Spearman
SelfCheckGPT	77.99	54.03	85.53	35.40	43.15
FActScore	66.41	49.42	74.86	42.58	45.60
FacTool	84.94	73.29	89.52	59.78	62.57
REFCHECKER					
GPT-4 + GPT-4	**93.82**	**86.89**	**95.96**	**83.95**	**82.35**
GPT-4 + NLI	83.98	71.28	88.89	60.81	62.32
GPT-4 + AlignScore	90.54	78.97	93.90	71.95	70.37
GPT-4 + RepC	89.96	81.16	93.16	77.42	77.26
Mistral-SFT + GPT-4	92.47	83.40	95.13	80.88	78.88
Mistral-SFT + NLI	89.96	78.86	93.42	72.89	72.07
Mistral-SFT + AlignScore	**90.54**	78.79	**93.91**	75.81	74.16
Mistral-SFT + RepC	89.38	**80.43**	92.72	**77.14**	**76.74**

Table 11: A comparison of RefChecker with previous works on our benchmark under Noisy Context setting. We highlight the best results using proprietary LLMs with blue colors and best results results using pure open-source models with orange colors.

Noisy Context Setting	Accuracy	Fact. F1	Non-Fact. F1	Pearson	Spearman
SelfCheckGPT	58.55	51.63	63.74	36.31	32.15
FActScore	63.57	69.94	53.77	33.36	29.91
FacTool	68.40	72.84	62.22	46.35	38.69
REFCHECKER					
GPT-4 + GPT-4	74.54	76.42	72.32	64.56	57.30
GPT-4 + NLI	66.73	74.54	52.01	39.69	32.98
GPT-4 + AlignScore	67.66	73.48	58.57	44.31	37.58
GPT-4 + RepC	65.99	74.04	50.67	28.19	28.94
Mistral-SFT + GPT-4	**75.28**	**77.57**	**72.46**	**67.29**	**59.94**
Mistral-SFT + NLI	**70.82**	**75.12**	**64.72**	52.21	**45.61**
Mistral-SFT + AlignScore	69.70	74.73	62.18	**53.88**	45.09
Mistral-SFT + RepC	65.99	73.97	50.94	38.11	31.01

Table 12: A comparison of RefChecker with previous works on our benchmark under Accurate Context setting. We highlight the best results using proprietary LLMs with blue colors and best results results using pure open-source models with orange colors.

Accurate Context Setting	Accuracy	Fact. F1	Non-Fact. F1	Pearson	Spearman
SelfCheckGPT	62.15	68.70	52.12	40.23	32.55
FActScore	69.37	78.57	46.30	27.80	27.05
FacTool	72.53	80.98	50.63	31.41	32.82
REFCHECKER					
GPT-4 + GPT-4	**80.81**	**86.85**	**64.50**	**58.61**	**55.50**
GPT-4 + NLI	73.06	82.23	44.36	28.59	30.14
GPT-4 + AlignScore	76.23	83.44	57.94	49.97	46.89
GPT-4 + RepC	76.94	84.86	51.66	45.58	41.01
Mistral-SFT + GPT-4	79.75	85.96	63.72	56.09	53.72
Mistral-SFT + NLI	73.59	81.44	54.27	44.34	40.81
Mistral-SFT + AlignScore	**74.12**	81.60	**56.38**	**46.34**	**43.22**
Mistral-SFT + RepC	73.94	**82.83**	45.99	39.59	33.86

```
I have a claim that made by a language model
to a question, please help me for checking
whether the claim can be entailed according
to the provided reference which is related
to the question.
The reference is a list of passages, and the
claim is represented as a triplet formatted
with ("subject", "predicate", "object").

If the claim is supported by ANY passage in
the reference, answer 'Entailment'.
If NO passage in the reference entail the
claim, and the claim is contradicted with
some passage in the reference, answer
'Contradiction'.
If NO passage entail or contradict with
claim, or DOES NOT contain information to
verify the claim, answer 'Neutral'.

Please DO NOT use your own knowledge for the
judgement, just compare the reference and
the claim to get the answer.

### Question:
{question}

### Reference:
{reference}

### Claim:
{claim}

Your answer should always be only a single
word in ['Entailment', 'Neutral',
'Contradiction']. DO NOT add explanations or
you own reasoning to the output.
```

Figure 13: Prompt for the LLM-based checkers.

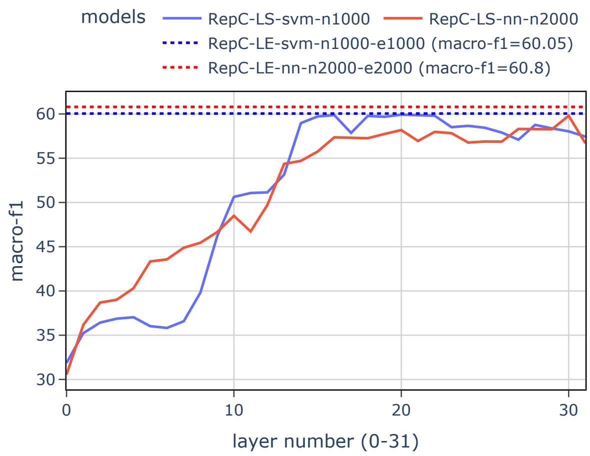

Figure 14: Performance tendency of different layers in RepC-LS checkers. The corresponding RepC-LE checkers are included as dashed lines.

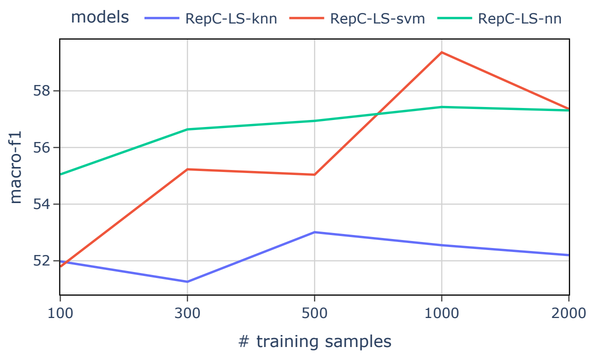

Figure 15: Performance of different RepC checkers with respect to numbers of training samples.

in Section D.1 and D.2, respectively. In general, we observe that LLMs/Checkers may incorporate internal knowledge even when provided with contextual information, contributing to the occurrence of hallucination.

D.1 Internal Knowledge Bias of Evaluated Model

In order to analyze whether the evaluated LLMs generate responses based on their own knowledge or the provided context in Noisy and Accurate Context settings, we convert each claim-triplet extracted from the response into a simple interrogative query for knowledge checking. For simplicity, we design a prompt template and ask GPT-4-Turbo to generate these queries (Figure 17). Then we feed the query into the evaluated LLM to check whether it has such knowledge. The answer from the evaluated LLMs could be one of the following:

1. `Yes`, means the evaluated LLM has this knowledge in its internal memory.
2. `No`, means the evaluated LLM contains knowledge that is contradicted with the triplet.
3. `Unsure`, means the evaluated LLM does not have this knowledge or it has confusion on the knowledge.

The label pairs (`Yes`, Contradiction) and (`Yes`, Neutral) indicate that the model is utilizing internal information to generate this claim-triplet. On the other hand, (`No`, Entailment) and (`Unsure`, Entailment) signify that the model is relying on contextual information for generation. Pairs like (`No`, Contradiction) suggest that the evaluated model may be less proficient in processing context information, leading to the production of less reliable claim-triplets.

The outcomes of the Accurate Context are illustrated in Figure 19. Upon examination, it is evident that GPT-4-Turbo demonstrates the most notable performance, primarily generating responses aligned with the reference context. Conversely, GPT-3.5-Turbo tends to generate responses by relying on its internal knowledge to some extent, leading to contradictions or neutrality to the ref-

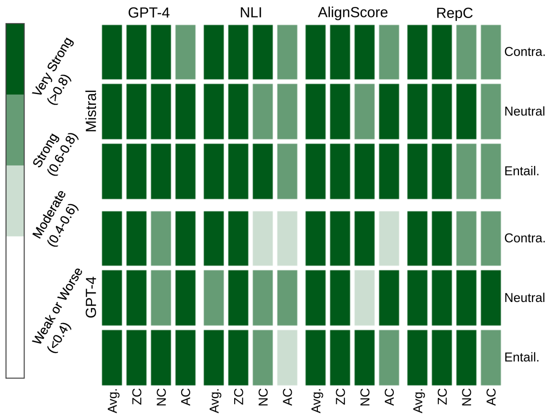

Figure 16: Spearman's rank correlation coefficients between REFCHECKER and human evaluation. Results are grouped regarding extractors and checkers used. Results for each extractor (row) and checker (column) are arranged into a sub-matrix in the figure, with correlations for rankings on 4 context settings (one additional for average) and 3 ranking metrics.

Table 13: REFCHECKER results on the SelfCheckGPT dataset. The results of SelfCheckGPT are from their paper. We highlight the best results using proprietary LLMs with blue colors and best results results using pure open-source models with orange colors.

	Pearson	Spearman
SelfCheckGPT	78.32	78.30
REFCHECKER		
GPT-4 + GPT4	80.86	83.44
GPT-4 + NLI	79.96	80.16
GPT-4 + AlignScore	76.20	77.33
GPT-4 + RepC	79.63	79.23
Mistral-SFT + GPT4	**80.98**	**83.92**
Mistral-SFT + NLI	**78.54**	**79.66**
Mistral-SFT + AlignScore	75.10	76.08
Mistral-SFT + RepC	76.59	76.70

erence context. In the case of InstructGPT, the model further generates unsure information, which also contradicts the reference context. This behavior may stem from contradictions within the model's internal knowledge or difficulties in comprehending the amalgamated content of internal and reference information. Regarding LLaMA-2-70B, and Falcon-40B-Instruct, our observations indicate that these models exhibit inferior performance. They generate information that contradicts internal knowledge and is irrelevant to the reference context. Alpaca 7B performs similarly to GPT-3.5-Turbo, but seldom generates information contradicting to its internal knowledge, Different from the accurate context setting, all the models tend to generate more Neutral labels in the noisy context setting (Figure 20).

Triplet: ("The Story of June", "has songs", "Hey June")

Prompt: Translate the following subject-verb-object triplets into simple interrogative sentences: {**Triplet**}

Simple interrogative sentence: Does The Story of June has songs Hey June? Answer the question with Yes, No, or Unsure

Figure 17: Designed prompt for converting triplets to simple interrogative sentences.

Triplet: ("The Story of June", "has songs", "Hey June")

Question: What are the names of the songs from The Story of June which don't have Chinese names?

Prompt: Mask the objective or subjective in the subject-verb-object triplet {**Triplet**} in the Context with #### which is not mentioned in Q (merely output the modified context).
Q : {**Question**}
Context: {**Context**}

Figure 18: Designed prompt for masking triplet information in the reference context.

D.2 Internal Knowledge Bias of Checker

We also conduct an analysis to determine whether the checker provides predictions based on its internal knowledge. In this analysis, a triplet extracted from the response is taken, and we mask the subjective or objective information in the context with '####'. The modified context, along with the triplet, is then inputted into the checker to obtain the label. In theory, the prediction label should be neutral because the relevant information in the context is masked. If the label is not neutral, it implies that the model is making inferences based on its internal knowledge. For the implementation of this analysis, we query GPT-4-Turbo with a specifically designed prompt to mask the triplet information, as illustrated in Figure 18. Specifically, in the noisy-context setting, we implement the query for each reference document and keep the document unchanged if there is no relevant information to the extracted triplet.

The results of the accurate context setting are shown in Table 14. As we observe, RoBERTa-NLI achieves the most significant Neutral labels, 62.64% and 53.70% for evaluated model GPT-3.5-

Table 14: Results for the information masking scenario in accurate-context setting.

Model	Checker	Entail	Contr	Neut
GPT-3.5	GPT-4	37.36	6.28	56.36
	RoBERTa-NLI	21.82	21.76	62.64
GPT-4	GPT-4	35.88	10.88	53.24
	RoBERTa-NLI	23.38	22.92	53.70

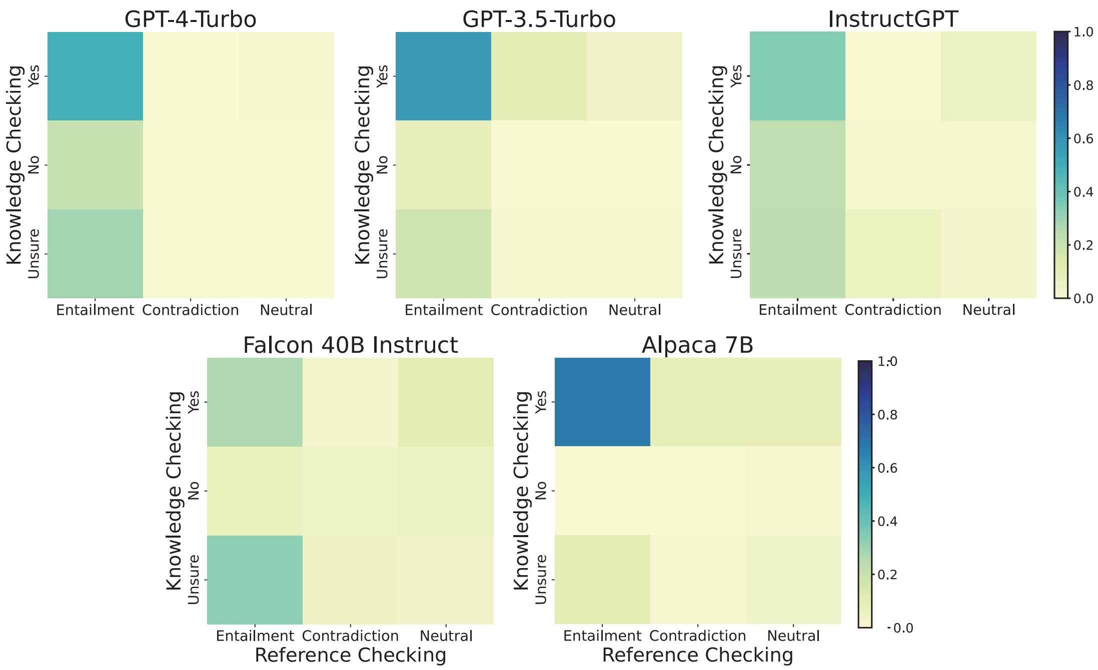

Figure 19: The results of knowledge checking for evaluated models in the accurate-context setting. The labels Yes, No and Unsure are the responses to the interrogative sentences generated from knowledge triplets. Each value refers to the percentage of each checking pairs in the total number of triplets.

Table 15: Results for the information masking scenario in zero-context setting.

Model	Checker	Entail	Contr	Neut
	GPT-4	37.91	22.55	39.54
GPT-3.5	RoBERTa-NLI	35.62	30.72	33.66
	GPT-4	43.33	13.67	43.00
GPT-4	RoBERTa-NLI	34.67	23.00	42.33

Table 16: Results for the information masking scenario in noisy-context setting.

Model	Checker	Entail	Contr	Neut
	GPT-4	58.52	6.67	34.82
GPT-3.5	RoBERTa-NLI	9.38	10.62	80.00
	GPT-4	65.71	6.29	28.00
GPT-4	RoBERTa-NLI	8.57	11.14	80.28

Turbo and GPT-4-Turbo, respectively. The checker GPT-4-Turbo achieves the second performance. The results of the zero context setting are in a similar pattern with those of accurate-context setting (Table 15). But in the noisy context setting (Table 16), RoBERTa-NLI outperforms GPT-4-Turbo with a large margin in the ratio of Neutral labels. The results may results from the strong bias to internal knowledge of GPT-4-Turbo when the context is extremely long, or the RoBERTa-NLI model has less associative ability to the memorized knowledge.

E Potential Risks

As hallucination detection techniques become more refined, there is a risk of overreliance on automated systems for determining the veracity of information. This could reduce critical engagement with content among users, potentially leading to a lack of scrutiny when systems fail to give a correct prediction.

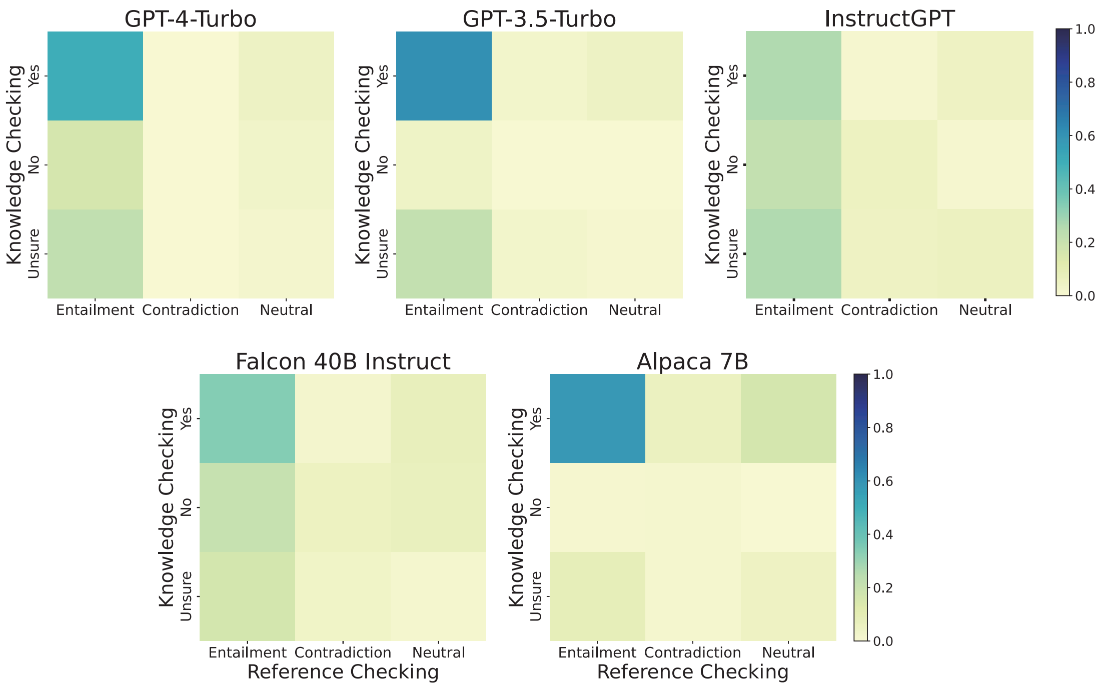

Figure 20: The results of knowledge checking for evaluated models in the noisy-context setting. The label Yes, No and Unsure are the respones to the interrogative sentences generated from knowledge triplets. Each value refers to the percentage of each checking pairs in the total number of triplets.