OpenFactCheck: A Unified Framework for Factuality Evaluation of LLMs

Anonymous ACL submission

Abstract

The increased use of large language models (LLMs) across a variety of real-world applications calls for mechanisms to verify the factual accuracy of their outputs. Difficulties lie in assessing the factuality of free-form responses in open domains. Also, different papers use disparate evaluation benchmarks and measurements, which renders them hard to compare and hampers future progress. To mitigate these issues, we propose **OpenFactCheck**, a unified factuality evaluation framework for LLMs. OpenFactCheck consists of three modules: (i) CUSTCHECKER allows users to easily customize an automatic fact-checker and verify the factual correctness of documents and claims, (ii) LLMEVAL, a unified evaluation framework assesses LLM's factuality ability from various perspectives fairly, and (iii) CHECKEREVAL is an extensible solution for gauging the reliability of automatic fact-checkers' verification results using human-annotated datasets. OpenFactCheck is publicly released at URL withheld.

1 Introduction

Large language models (LLMs) have demonstrated impressive capabilities in generating naturally-sounding answers over a broad range of human inquiries. However, GPT-4 (OpenAI, 2023) and other text generation models still produce content that deviates from real-world facts (Bang et al., 2023; Borji, 2023; Guiven, 2023). This degrades the system performance and undermines its reliability, representing a significant bottleneck in the deployment (Chuang et al., 2023; Geng et al., 2023).

Many studies have explored evaluating and improving the factuality of LLMs (Lee et al., 2022; Chuang et al., 2023; Shi et al., 2023; Chen et al., 2023). Two challenges have been identified for evaluation: (1) it is difficult to assess the factuality of open-domain free-form responses, and (2) different papers use different evaluation datasets and measurements, rendering them hard to compare and hampering future progress (Wang et al., 2024). To mitigate these issues, we introduce a fact-checking framework **OpenFactCheck**.

It includes three modules as shown in Figure 1. CUSTCHECKER allows users to customize an automatic fact-checker and to verify free-form documents to alleviate the first problem. A unified LLM factuality evaluation module LLMEVAL applies seven factuality-specific benchmarks to assess the LLM factuality ability from different aspects and then produces a report to illustrate the weakness and offer improvement advice, tackling the second challenge. We further incorporate CHECKEREVAL that assesses the verification accuracy of fact-checkers, equipped with a leaderboard in terms of accuracy, latency, and costs, aiming to encourage the development of advanced automatic fact-checking systems.

The three modules collaborate and help each other. The results of human verification derived from LLMEVAL can be used as the benchmark for evaluating the accuracy of automated fact-checkers. Simultaneously, the most effective checker identified in CHECKEREVAL can be deployed for automated fact-checking tasks. Each fact-checker in CHECKEREVAL can be an implementation in CUSTCHECKER. Complex user inquiries may be considered as potential candidates included the factuality assessment dataset utilized in LLMEVAL.

Users can tailor their checkers according to their specific needs, such as domain specialization, cost-effectiveness, or rapid processing, and identify factual errors for both human-written text (a claim or document) and the outputs of LLMs. LLM researchers and practitioners can directly submit their LLM responses to the LLMEVAL by downloading our question set. Subsequently, we conduct evaluations to assess the model's factual accuracy and to generate a report analyzing the model performance from multiple aspects. Similarly, developers

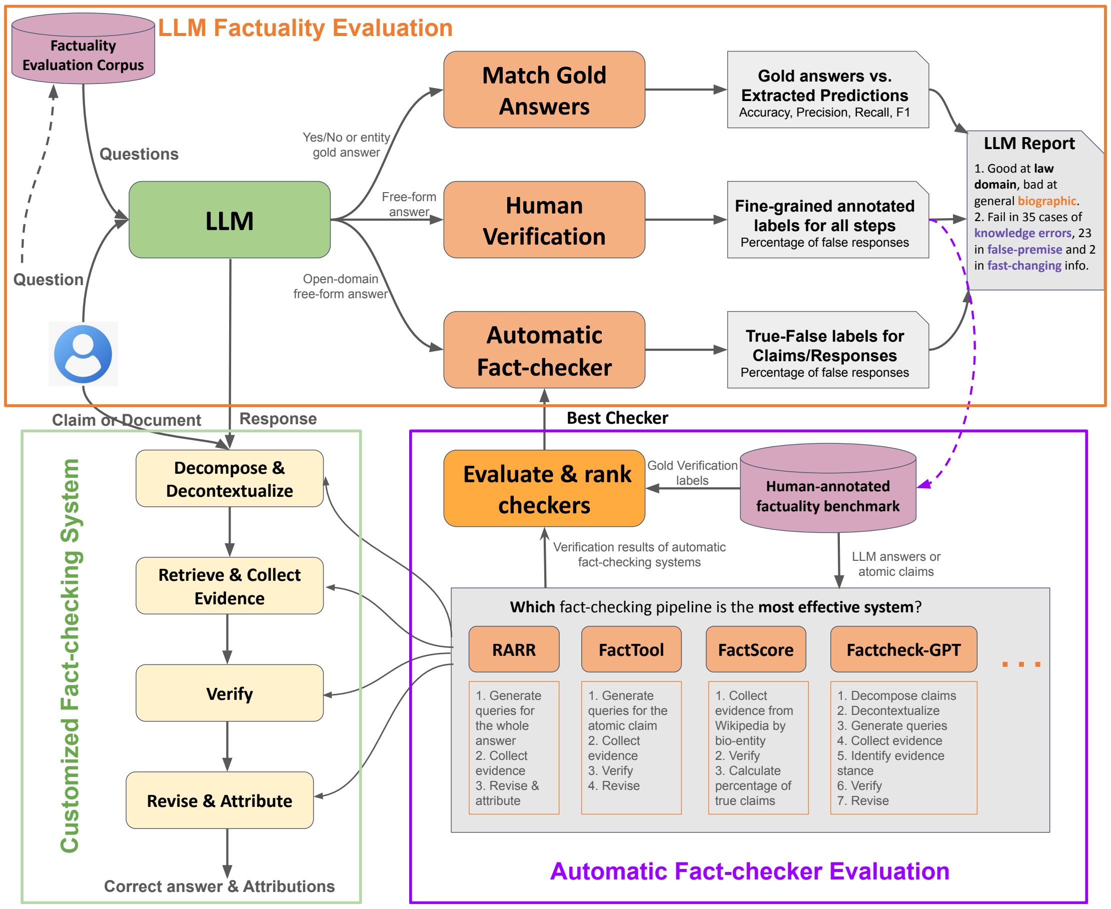

LLM Factuality Evaluation
Factuality Evaluation Corpus
Questions
Question
LLM
Yes/No or entity gold answer
Free-form answer
Open-domain free-form answer
Match Gold Answers
Human Verification
Automatic Fact-checker
Gold answers vs. Extracted Predictions
Accuracy, Precision, Recall, F1
Fine-grained annotated labels for all steps
Percentage of false responses
True-False labels for Claims/Responses
Percentage of false responses
LLM Report
1. Good at law domain, bad at general biographic.
2. Fail in 35 cases of knowledge errors, 23 in false-premise and 2 in fast-changing info.
Claim or Document
Response
Best Checker
Customized Fact-checking System
Decompose & Decontextualize
Retrieve & Collect Evidence
Verify
Revise & Attribute
Correct answer & Attributions
Evaluate & rank checkers
Gold Verification labels
Human-annotated factuality benchmark
Verification results of automatic fact-checking systems
LLM answers or atomic claims
Which fact-checking pipeline is the most effective system?
RARR
FactTool
FactScore
Factcheck-GPT
1. Generate queries for the whole answer
2. Collect evidence
3. Revise & attribute
1. Generate queries for the atomic claim
2. Collect evidence
3. Verify
4. Revise
1. Collect evidence from Wikipedia by bio-entity
2. Verify
3. Calculate percentage of true claims
1. Decompose claims
2. Decontextualize
3. Generate queries
4. Collect evidence
5. Identify evidence stance
6. Verify
7. Revise
Automatic Fact-checker Evaluation

Figure 1: Overview of the OpenFactCheck demo system for LLM factuality evaluation with three modules. Green CUSTCHECKER: a customized fact-checker to identify factual errors given the outputs of LLMs. Orange LLMEVAL: a unified LLM factuality evaluator to assess the LLM factual ability from different aspects and then to produce a report to illustrate the weakness and to offer improvement advice. Purple CHECKEREVAL: a fact-checker evaluator and leaderboard to encourage the development of advanced checkers in terms of performance, time and costs.

who seek to evaluate and to compare the efficacy of their fact-checking systems to other ones fairly can upload their checker's verification outcomes to CHECKEREVAL. Then, our system will show the ranking information in the leaderboard after evaluating under the same measurements.

To sum, this work investigates three questions:

- how to effectively identify factual errors in an LLM response;
- how to systematically evaluate the factuality ability of an LLM;
- which automatic fact-checker is the best, and which component dominates the final verification accuracy.

We initiate an open-source project and develop a preliminary version implementing the three modules, which is anticipated to serve as a stepping stone to facilitate future endeavors in this domain. We encourage extensive implementation of unique, effective, and robust claim processors, retrievers and verifiers within fact-checking pipelines, collections of challenging questions that LLMs tend to make factual errors, and human-annotated fine-grained verification examples. We believe that this will help to promote and to advance future research on LLM factuality.

2 Background

2.1 Fact-checking Systems

Fact-checking is the task of assessing whether claims made in writing are manipulated or true. Many recent papers have described automatic fact-checking systems used to evaluate the factuality of LLM responses, such as *RARR*, *FactScore*, *FacTool*, *CoVe*, and *Factcheck-GPT* (Gao et al., 2022; Min et al., 2023; Chern et al., 2023; Dhuliawala et al., 2023; Wang et al., 2023). Each checker has unique characteristics designed for specific domains or scenarios. *RARR* verifies a document as a whole and can generate an attribution report to explain factual errors. *FactScore* retrieves evidence from an offline Wikipedia dump mainly for the biography. *FacTool* is friendly to users with low latency, *CoVe* completely depends on the capability of LLMs, and *Factcheck-GPT* has a fine-grained pipeline to localize intermediate errors. Unlike the above work, our aim is to enable the easy customization of a fact-checker according to users' requirements and application scenarios, e.g., offline settings with a limited budget, by simply clicking dropdowns to choose offline retrievers and verifiers supported by small models without calling APIs.

Despite different designs and implementations of various checkers, they generally consist of three modules: (1) *claim processor*, which extracts context-independent atomic claims from a document, (2) *evidence retriever*, which searches related passages from the Internet or database, and then ranks them by relevance, and (3) *verifier*, which determines the claim/document factuality based on the collected evidence (Guo et al., 2022; Li et al., 2023b; Wang et al., 2024). To this end, we first unify different fact-checking systems into a unified pipeline with the three modules. Then, given a module, users can select a developed module from various implementations or develop one by themselves. In addition, the framework supports easy migration of existing fact-checking systems to our pipeline. However, the verification results of automatic fact-checkers are not necessarily accurate. How to evaluate and improve the accuracy of automated fact-checkers is critical, since the accuracy serves as a confidence and reliability signal for the verification results.

2.2 Evaluation of Fact-Checking Systems

How accurate are current fact-checking systems? Can they effectively serve as proxies for evaluating the factual accuracy of language models? Existing automatic fact-checking studies often first collect a set of human-annotated (LLM response, extracted claims, factuality of the claims), and then quantify the effectiveness of their systems by comparing the final verification results (i.e., whether a claim or a document is factually *true or false*) to human-annotated labels (Min et al., 2023; Chern et al., 2023; Dhuliawala et al., 2023). Recent work on long-form factuality in LLMs also demonstrates a statistically significant correlation between the outputs by automatic fact-checkers and labels by human annotators (Wei et al., 2024). Thus, we merge four human-annotated LLM factuality datasets including FacTool-QA, FELM-WK, Factcheck-Bench, and HaluEval, and then compare them to the results of automatic fact-checkers to assess the performance of fact-checking systems.

2.3 LLM Factuality Evaluation

There are subtle differences between evaluating LLM's general performance and factuality. Question answering (QA) datasets over various domains are always used for general performance evaluation (Hendrycks et al., 2021). The focus is on judging whether the question is answered correctly. If the model's response contains the correct answer, it counts; otherwise, it is void even though the response presents all facts. While research on factuality concentrates more on whether the response presents facts aligning with world knowledge even if some statements were irrelevant to the question.

Therefore, instead of using datasets for general performance assessment, we selected seven datasets that are specifically collected for factuality evaluation. They were selected to cover as diverse potential factual errors as possible, including vulnerabilities of snowballing hallucinations, awareness of self-uncertainty, robustness to false-premise questions, fresh questions with answers changing fast over time, and free-form responses spanning distinct domains, topics, and tasks.

Summary Observations above motivate us to integrate these three components into one framework to facilitate (i) users to flexibly configurate an automatic fact-checking system to verify the factuality of claims and documents, (ii) LLM developers to evaluate LLM's factuality under the same measurement scale, and (iii) researchers to assess the fact-checkers reliability under fine-grained annotated benchmarks.

3 Design and Implementation

OpenFactCheck is implemented by a Python server, a web user interface, and a database, deployed via AWS. The Python backend can also be used as a Python toolkit, allowing easy and flexible development. OpenFactCheck's design emphasizes two principles: (i) customizability and extensibility for both users and developers, and (ii) compatibility with existing methods and datasets. It consists of three modules: CUSTCHECKER, LLMEVAL, and

CHECKEREVAL. Below, we present the detailed design and implementation of each component.

3.1 CUSTCHECKER

CUSTCHECKER allows users to customize a fact-checking system by selecting a claim processor, a retriever, and a verifier in web pages. Current version supports the following fact-checking systems: *RARR*, *FacTool* and *Factcheck-GPT* (Gao et al., 2022; Chern et al., 2023; Wang et al., 2023). Users input either human-written text or outputs of LLMs into the box (see Figure 4), and then the fact-checker defined above will process and detect factual errors, showing the verification results including evidence, judgment, and explanations.

Configurable Pipeline We unify diverse fact-checking systems as a procedure with three steps, abstracted into three classes: a claim_processor, a retriever, and a verifier. The instances of the three classes are sequentially linked into a pipeline, solving the following tasks: (i) decomposing a document into atomic claims, (ii) collecting relevant evidence passages given a claim, and (iii) making a true/false judgment given both the claim and the evidence as input (see pseudo code in Figure 3). We refer to them as task solvers. The implementation of a task solver can be flexible, just ensuring that the input and the output are aligned with the abstract class definitions. For example, evidence can be retrieved by calling SerpAPI or by searching Wikipedia using BM25, but we must return a list of relevant passages given an input claim.

Moreover, task solvers in our pipeline are not hardcoded, but can be configured through a *yaml* configuration file. Thus, users can combine task-solver implementations from different systems (e.g., using *Factcheck-GPT*'s claim processor, *RARR*'s retriever, and *FacTool*'s verifier) and start the verification from any step. For example, users can start from the step of retrieval when the input does not need decomposition.

This functionality is achieved by a message-passing mechanism, where a *success flag* is used to indicate whether the current task solver successfully executes and returns the expected output. The success flag passes through the configured solver order of the pipeline, guaranteeing that the output of the preceding solver fits the input for the current solver, otherwise error warning will be issued. Practically, the input and the output parameter names for the task solvers are defined in the configuration file. To link different solvers into a pipeline, one only needs to ensure that the current solver output name matches the input name of the succeeding solver. A dictionary format *fact-checking-state* is kept throughout the pipeline to store all information in the verification.

Extendable Architecture Inspired by Fairseq, our framework is designed to be highly extendable by treating any third-party task solvers as plug-ins (Ott et al., 2019). As long as the developed task solvers adhere to our class interface definitions, they can be imported and used in our framework.

To sum, customizable and extendable nature of CUSTCHECKER allows general users to utilize it as an application with web-based user interfaces. Advanced developers have the flexibility to use it as a library, developing and integrating their solvers.

3.2 LLMEVAL

We observed that studies assessing language models' factuality or evaluating whether the methods are effective to mitigate model hallucinations use different datasets and metrics. This makes it difficult to compare, in the same conditions, the factuality of different models as well as to compare the effectiveness of different factuality enhancement approaches. Moreover, a lot of prior work applied datasets such as MMLU (Hendrycks et al., 2021), StrategyQA (Geva et al., 2021) and HotpotQA (Yang and et al., 2018) to evaluate model's factuality. These datasets tend to focus on assessing the general performance, rather than factuality. To this end, we first collect a dataset *FactQA* by gathering a large number of factual questions that probe diverse factual errors and span across a spectrum of domains, to fairly evaluate LLMs' factuality under the same criteria.

Factual Question Collection We collect a set of questions by gathering questions from seven existing corpora that is collected deliberately to assess LLM's factuality, including Snowball (Zhang et al., 2023a), SelfAware (Yin et al., 2023), FreshQA (Vu et al., 2023), *FacTool* (Chern et al., 2023), FELM-WK (Chen et al., 2023), *Factcheck-GPT* (Wang et al., 2023) and FactScore-Bio, a total of 6,480 examples shown in Table 1, referring to FactQA (see dataset details in Appendix B.1).

To concretely analyze models' vulnerability, we identify three labels for each question from the perspective of the knowledge domain, the topic, and

Dataset↓	**The Ability to Evaluate**	**Domain**	**Error**	**Size**
Snowball	Snowballing hallucination when model immediately output	Math, history, graph search	Type 2	1,500
SelfAware	Understand their own limitations on the unknowns	Biology, philosophy, psychology, history	Type 1,3	3,369
FreshQA	Answer questions changing fast over time or with false premises	Sports, entertainment, history, technology	Type 3	600
FacTool-QA	Respond knowledge-based questions	History, geography, biology, science	Type 1	50
FELM-WK	Answer world-knowledge questions	History, biology, geography, sports	Type 1	184
Factcheck-Bench	Answer open-domain, false-premise questions	Technology, history, science, sports	Type 1,2	94
FactScore-Bio	Generate detailed biographies	Biography	Type 1,3	683
Total	LLM factuality against world knowledge	482 domains, top20 accounts for 70%	Type 1,2,3	6,480

Table 1: **FactQA**: factual vulnerability, domain, potential error type and size across seven component datasets.

the potential error type if a LLM generates a factually incorrect response. So each example includes the following fields: *question*, *domain*, *topic*, *ability to test*, *task* and *source*. Domain and topic are identified using GPT-4 based on the (question, reference response).[1] Domains involve general, legal, biomedical, clinical, scientific and so on. Given a domain, we further fine-grained topics. Three common error types are presented.

Type1*: *Knowledge error is the most common error, occurring when the model produces hallucinated or inaccurate information due to lacking relevant knowledge or internalizing false knowledge in the pre-training stage or in the problematic alignment process. However, LLMs do not know what they do not know, sometimes overestimate their capacities and confidently output unknown information, leading to false responses. Mitigating such errors require: (a) learning and correcting parametric knowledge through the curation of corpora used in pre-training, supervised fine-tuning (SFT) and alignment, (b) augmenting by external knowledge in inference, (c) calibrating models to be aware of unknowns, and (d) configuring the decoding strategies (sample/beam-search, temperature), balancing diversity and accuracy (Zhang et al., 2023b).

Type2*: *Over-commitment error occurs when the model fails to recognize the falsehoods (or jokes) inherent in the prompt or previously-generated context, and provides an inaccurate or inappropriate response. The left-to-right generation strategy used by LLMs poses potential risks that LLMs sometimes over-commit to the false premise in the context, even when they recognize they are incorrect (Zhang et al., 2023b). To address this issue, engineering better prompts is helpful, such as explicitly instructing models to first detect false premises in the prompt (Vu et al., 2023) and asking the same question in a different way (*Is 10733 a prime number?* → *What are the factors of 10733? Let's think step-by-step.*)

[1]We used GPT-4 response as a reference response for a question as it is more likely to provide a relevant and correct answer, assisting the identification of domains and topics.

Dataset ↓	#True	#False	#Unknown	Total
FacTool-QA	177	56	0	233
FELM-WK	385	147	0	532
Factcheck-Bench	472	159	47	678
HaluEval	3,692	815	0	4,507

Table 2: The number of true, false claims and unknown (no-enough-evidence or opinions) for FacTool-QA, FELM-WK and Factcheck-Bench, the number of responses for HaluEval (no claim-level labels).

Type3*: *Disability error happens when the model is unable to search up-to-date information to correctly answer questions whose answers change over time, e.g., *What is today's gas price in New York* (fast-changing). Retrieving external knowledge and augmenting it in the context would help.

Note that we do not consider *reasoning errors* that arise when a claim employs flawed reasoning or faulty logic, and *irrelevant error* concerning that the content is unrelated to the question (Chen et al., 2023). The former highlights LLM's reasoning ability, which is more reflected in math and reasoning tasks, and the latter has more to do with response's helpfulness or human preference. They are important in LLM evaluation, and may implicitly influence factuality, but we will first focus on explicit causes, leaving the implicit for future work.

Evaluation Measurement For questions that can be answered by Yes/No or have a short gold answer, we perform exact matching between the model responses and the gold standard answer to judge whether the response is factually correct or not, and then to calculate accuracy, such as for Snowball and SelfAware. For FreshQA, we use the *FreshEval* proposed in Vu et al. (2023) to evaluate the correctness of model's responses, in which few-shot in-context learning based on GPT-4 is applied. We use the strict evaluation criterion which considers an answer to be correct only if all the claims in the response are factually true and also up-to-date.

For open-domain questions from the other four datasets with free-form and long responses, there are no gold standard answers. We use automatic fact-checking systems augmented with retrieved world-knowledge evidence to judge the correctness at the claim-level as well as the document level.

3.3 CHECKEREVAL

Automatic fact-checking systems aim to identify whether a claim or a document is factually correct or not with/without references, but the results are not necessarily correct. To assess the accuracy of automatic fact-checkers, we gather four LLM factuality benchmarks with human-annotated factual labels for three levels of granularity text: claims/segments/documents given (question, ChatGPT response) pairs, including FacTool-QA, FELM-WK, Factcheck-Bench and HaluEval as shown in Table 2. We refer to them as FactBench. We use precision, recall, and F1-score with respect to the *True* or *False* claim/document to evaluate the effectiveness of fact-checking systems.

This method regards the system as a whole, only assessing the final verification results, i.e., whether a claim or a document is true or false. The evaluation of intermediate results throughout the fact-checking pipelines will be incorporated in future updates, to localize which step eventually results in the erroneous factual judgment for claims.

Web Client We develop a web client based on Streamlit, consisting of four interfaces, with each corresponding to one of the three modules, along with a leaderboard, to enhance the user interaction, see more in Appendix C. The design principle is to invoke these functional modules in the form of third-party applications, avoiding excessive intervention in the system's architecture. This makes OpenFactCheck a three-in-one to users as a library, a command-line toolkit, and a web application.

4 Experiments

We first evaluate the factuality of three LLMs, and then we assess the accuracy of different automatic fact-checking systems in multiple settings.

4.1 LLaMA-2 and GPT-4 Evaluation

Based on questions/instructions in FactQA, we collected responses from LLaMA-2 (7B, 13B) and GPT-4. As shown in Table 7, the responses of GPT-4 tend to be shorter than that of LLaMA-2.

Results and Analysis On the Snowball dataset, we observe high error rates: >80% for LLaMA-2 and 65.5% for GPT-4, similar to the results on GPT-3.5-Turbo presented by Zhang et al. (2023a). However, when justifying previously generated content, GPT-4 can identify 87% of its own mistakes. Therefore, in these cases, errors are mostly attributed to the over-committing to the previously generated false context, rather than to large knowledge gaps in LLMs. An LLM over-commits to early mistakes, leading to more mistakes that it otherwise would not have made. Its prevalence in generative models leads to factual errors for simple facts.

SelfAware aims to evaluate LLMs' ability to understand their own limitations and unknowns, identifying unanswerable or unknowable questions. Higher precision than recall is achieved across three models with regard to unanswerable questions. This reveals that many truly unanswerable questions are incorrectly recognized as answerable, implying that models are always not aware of what they do not know. Poor performance on questions with rapidly changing answers (FreshQA) illustrates the inherent challenge of retrieving up-to-date information for LLMs.

We used *FacTool* equipped with Serper and GPT-3.5-Turbo to automatically evaluate the factuality of free-form responses over prompts in FacTool-QA, FELM-WK, and Factcheck-Bench. The results are shown in Figure 2, where we can make several interesting observations:

- The percentage of true claims is in the range of 89%-94%, revealing that the vast majority of claims are verified as true.
- The questions in FacTool-QA are relatively more challenging for the three LLMs to answer correctly than for the other two datasets, leading to a relatively lower percentage of true claims. The apparent lower number of false claims in FacTool-QA stems from its smaller dataset size, where 50 is less than 94 and 184.
- GPT-4 has the best factuality performance with a smaller number of false claims and higher percentage of true claims, followed by LLaMA-2 13B and then 7B;
- The cost for automatic evaluation mainly depends on the number of atomic claims and

Dataset →	Snowball				SelfAware				FreshQA	
Model ↓	Primality	Senator	GraphConnection	Full-set	Precision	Recall	Accuracy	F1-score	Accuracy	Perc_valid
LLaMA-2 7B	5.6%	20.4%	17.4%	14.5%	69.7%	30.3%	74.6%	42.0%	28.3%	93.2%
LLaMA-2 13B	0.0%	9.4%	32.4%	19.5%	64.9%	30.1%	73.6%	41.2%	29.7%	95.5%
GPT-4	0.2%	49.0%	71.0%	34.5%	71.7%	21.6%	73.4%	33.2%	39.5%	98.3%

Table 3: LLM factuality evaluation accuracy for Snowball: for each of its three topics as well as on average. Shown are SelfAware precision, recall, and F1-score when the positive label=*unanswerable*, and FreshQA accuracy, as well as percentage of valid assessments.

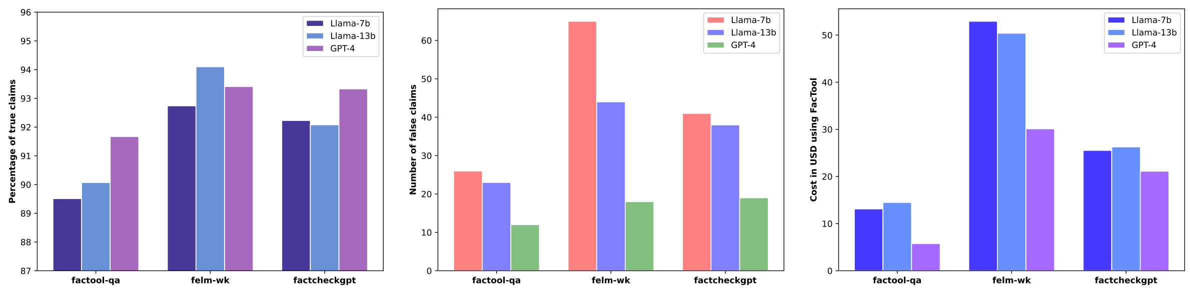


Figure 2: Automatic evaluation results for LLaMA-2 7B, 13B and GPT-4 responses on datasets of FacTool-QA, FELM-WK, and Factcheck-Bench using *FacTool*. *left:* the percentage of true claims, *center:* the number of false claims, and *right:* the cost of using *FacTool* in USD.

the price of the backend models used in *FacTool*. It spends \$0.02 for an atomic claim on average.

Summary snowballing hallucination, over-commitment to false premise, difficulty in identifying unknown knowledge and answering with up-to-date information are still challenging issues for LLMs. For general open-domain questions, on average less than 10% of the claims are factually incorrect in LLM responses. This somehow implies that models may poorly understand instructions and their knowledge scope, but they can correctly generate majority of content. This is aligned with the recent finding that what an LLM can generate, it may not understand (West et al., 2023). Additionally, it is costly to evaluate open-domain answers even if based on automatic fact-checkers, ∼ \$30 for 100 responses based on the cheapest GPT-3.5-Turbo.

4.2 Evaluating Fact-Checking Systems

We investigate automatic fact-checking systems in three aspects: accuracy, latency, and costs. Based on annotated factual labels for claims from three benchmarks of Factcheck-Bench, FacTool-QA, and FELM-WK, we evaluate the verification performance in multiple settings across different fact-checking frameworks, evidence sources, and verifiers.

Pipeline and core component modules of different fact-checking frameworks are basically similar, including obtaining atomic claims, collecting evidence and verifying correctness; all thus, while the implementations are different. For example, in terms of how to extract atomic claims, *RARR* does not include this step. *FactScore* first breaks down a document into paragraphs, and then applies NLTK (Bird and Loper, 2004) to split paragraphs into sentences, and then prompts LLMs to decompose to atomic claims (GPT3 was used in the original paper). However, this implementation neglects the decontextualization in the paragraph and in the sentence decomposition, making claims non-independent (e.g., *He is a university professor and the CEO of a tech startup company*). To mitigate, *FacTool* directly extracts claims based on the document, and *Factcheck-GPT* decontextualizes both sentences and claims based on the document.

Experimental Setup To ensure that all fact-checking systems verify the same sets of annotated claims, we skip the step of extracting atomic claims from the documents. All systems get a claim as an input, and they are expected to output whether or not the claim is true.

Recent fact-checking frameworks such as *FactScore*, *FacTool*, *Factcheck-GPT* and commercial retrieval-augmented generative models such as Perplexity.ai are evaluated, with evidence retrieved

Framework	Verifier	Source/ Retriever	Factcheck-Bench						FacTool-QA						FELM-WK					
			Label = True			Label = False			Label = True			Label = False			Label = True			Label = False		
			Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1	Prec	Recall	F1
Random	–	–	0.79	0.43	0.56	0.18	0.52	0.27	0.79	0.56	0.66	0.28	0.54	0.37	0.71	0.52	0.60	0.26	0.43	0.32
Always True	–	–	0.81	1.00	0.88	0.00	0.00	0.00	0.76	1.00	0.86	0.00	0.00	0.00	0.72	1.00	0.84	0.00	0.00	0.00
Always False	–	–	0.00	0.00	0.00	0.19	1.00	0.33	0.00	0.00	0.00	0.24	1.00	0.39	0.00	0.00	0.00	0.28	1.00	0.43
FactScore	LLaMA 3-Inst 8B	Wiki/BM25	0.87	**0.74**	0.80	0.34	0.56	0.42	0.82	0.68	0.74	0.34	0.52	0.41	0.76	0.66	0.71	0.34	0.46	0.39
FacTool	LLaMA 3-Inst 8B	Web/Serper	0.88	0.80	**0.84**	0.40	0.56	0.47	**0.93**	0.38	0.54	0.32	**0.91**	0.47	0.79	0.31	0.44	0.30	**0.79**	0.44
FactScore	GPT-3.5-Turbo	Wiki/BM25	0.87	0.67	0.76	0.31	0.60	0.41	0.82	0.58	0.68	0.31	0.59	0.40	0.77	0.71	0.74	0.36	0.43	0.39
FacTool	GPT-3.5-Turbo	Web/Serper	0.89	**0.74**	0.81	0.37	0.62	0.46	0.92	0.59	0.72	0.39	0.84	0.53	0.78	0.62	0.69	0.35	0.54	0.43
Factcheck-GPT	GPT-4	Web/SerpAPI	0.90	0.71	0.79	**0.52**	**0.80**	**0.63**	0.88	**0.88**	**0.88**	**0.63**	0.63	**0.63**	**0.81**	**0.87**	**0.84**	**0.55**	0.44	**0.49**
Perplexity.ai	Sonar-online	Web	**0.93**	0.73	0.83	0.40	0.76	0.53	0.82	**0.88**	0.85	0.50	0.38	0.43	0.76	0.82	0.79	0.40	0.31	0.35

Table 4: **Verification results** for human-annotated claims in Factcheck-Bench, FacTool-QA, and FELM-WK, judging whether or not a claim is factually true or false with external knowledge (Wikipedia or Web articles) as evidence. The implementation of *Factcheck-GPT* usedlangchain AutoGPT. GPT-4 refers to *gpt-4-turbo-2024-04-09*.

Fact-Checker ↓	Web search ($)	LLM ($)	Time (hrs)
FacTool	2.5 (Serper)	12.2 (GPT-3.5)	0.49
Factcheck-Bench	13.3 (SerpAPI)	26.6 (GPT-4)	7.67

Table 5: Time and USD cost for evaluating the 765 claims in FacTool-QA and FELM-WK.

from Wikipedia articles or web pages, as well as with various LLM-based verifiers that judge the factuality of a claim based on their internal knowledge and retrieved evidence as a reference.

Results and Analysis In Table 4, we observe that automatic fact-checking systems struggle to detect false claims. Across the three datasets we experiment with, it is consistently more arduous for systems to differentiate false claims compared to identifying true ones. This challenge may arise from the tendency of returning invalid evidence for false claims.

Retrieving evidence from the web using Serper (Google search engine results) is more effective than sourcing related passages from Wikipedia articles using BM25, given that a wider array of effective evidence is accessible on open web pages for open-domain questions. The verification accuracy of an LLM-based verifier primarily relies on the capabilities of the LLM and the effectiveness of the prompts used. For instance, the overall performance of GPT-4 surpasses that of both LLaMA-3-8B and GPT-3.5-Turbo, and thus the verification results of *Factcheck-GPT* outperform those of *FacTool*, *FactScore* and Perplexity.ai, despite all of them utilizing evidence sourced from the web. While *Factcheck-GPT* exhibits superior effectiveness, it is associated with considerable latency and substantial costs (see Table 5).

Latency and cost are largely contingent upon the implementation strategy. For instance, *FacTool* adopts asynchronous processing and leverages Serper ($0.001 per search) in conjunction with GPT-3.5-Turbo, rendering it faster and more economical compared to *Factcheck-GPT*. Notably, *Factcheck-GPT* uses SerpAPI ($0.015 per search) alongside GPT-4, where the cost of the most affordable GPT-4 model is 20 times that of GPT-3.5-Turbo (see Figure 8).

In summary, the efficacy of automated fact-checking systems is fundamentally dependent on implementation factors such as choice of search tool, prompts, and backend LLMs. This is primarily driven by engineering considerations.

5 Conclusion and Future Work

We proposed OpenFactCheck, a unified, easy-to-use and extensible framework. It supports the customization and evaluation of automatic fact-checking systems and LLM factuality evaluation. Specifically, OpenFactCheck allows general users to check whether a claim and a document are factual or not, and also facilitate LLM practitioners and developers to effectively and efficiently evaluate the factuality of their LLMs from various perspectives, and to assess the accuracy of automatic fact-checking systems.

Our extensive experiments indicate that more than 90% of the claims generated by LLMs in response to open-domain questions are factually correct. Nevertheless, models encounter challenges when addressing some straightforward questions such as *Is 7411 a prime number?* This difficulty can be attributed to the fact that LLMs demonstrate weaker comprehension abilities relative to their generation capabilities. Additionally, prevalent fact-checking systems struggle to identify false claims, with the retrieval of pertinent evidence posing a significant bottleneck. The latency and the cost associated with these systems primarily hinge on implementation strategies. In the future, we will continue to integrate new techniques, features, and evaluation benchmarks to OpenFactCheck to facilitate the research progress of LLM fact-checking.

Limitations

While OpenFactCheck presents a comprehensive framework for factuality evaluation of LLMs, several limitations must be acknowledged:

Evaluation Datasets The effectiveness of OpenFactCheck is dependent on the quality and diversity of the datasets used for evaluation. While we have integrated multiple datasets to cover a broad spectrum of domains and potential factual errors, the evaluation is still limited by the inherent biases and coverage gaps in these datasets. For instance, some specialized domains may not be adequately represented, potentially affecting the robustness of the evaluation for LLMs in those areas.

Latency and Costs The performance of automatic fact-checking systems integrated within OpenFactCheck can vary significantly in terms of latency and operational costs. High accuracy often comes at the expense of increased computational resources and processing time, which may not be feasible for all users, particularly those with limited budgets or time constraints.

Reliance on External Knowledge Sources The fact-checking modules depend heavily on external knowledge sources, such as Wikipedia and web search engines. The availability and reliability of these sources can affect the accuracy and completeness of the fact-checking process. Furthermore, the dynamic nature of web content means that the information retrieved may not always be up-to-date.

Ethical Statement

The development and deployment of OpenFactCheck are guided by a commitment to ethical principles, ensuring that the framework is used responsibly and for the benefit of society:

Transparency and Accountability We strive to maintain transparency in the design, implementation, and evaluation of OpenFactCheck. The source code and datasets are publicly available, enabling scrutiny and fostering trust within the research community. We encourage users to report any issues or biases they encounter, facilitating continuous improvement.

Bias Mitigation Recognizing that biases can exist in both datasets and LLMs, we are dedicated to minimizing such biases in OpenFactCheck. By integrating diverse evaluation benchmarks and encouraging the development of fair fact-checking approaches, we aim to reduce the impact of biases on factuality evaluation outcomes.

Social Impact By enhancing the factual accuracy of LLMs, OpenFactCheck aims to contribute positively to society. Accurate information is crucial for informed decision-making and public discourse. We believe that improving the reliability of LLM outputs can help combat misinformation and support the dissemination of truthful information.

References

Anonymous. 2023. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *Submitted to The Twelfth International Conference on Learning Representations*. Under review.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *CoRR*, abs/2302.04023.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Ali Borji. 2023. A categorical archive of chatgpt failures. *CoRR*, abs/2302.03494.

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. *arXiv preprint arXiv:2310.00741*.

I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, and Pengfei Liu. 2023. Factool: Factuality detection in generative AI - A tool augmented framework for multi-task and multi-domain scenarios. *CoRR*, abs/2307.13528.

Yung-Sung Chuang, Yujia Xie, and Hongyin Luo et al. 2023. Dola: Decoding by contrasting layers improves factuality in large language models. *CoRR*, abs/2309.03883.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models. *arXiv preprint arXiv:2309.11495*.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, Da-Cheng

Juan, et al. 2022. Attributed text generation via post-hoc research and revision. *arXiv preprint arXiv:2210.08726*.

Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2023. A survey of language model confidence estimation and calibration. *arXiv preprint arXiv:2311.08298*.

Mor Geva, Daniel Khashabi, and et al. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *TACL*, 9:346–361.

Guiven. 2023. Llm failure archive (chatgpt and beyond). `https://github.com/giuven95/chatgpt-failures`.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Generating text from structured data with application to the biography domain. *ArXiv e-prints, March*.

Nayeon Lee, Wei Ping, and Peng et al. Xu. 2022. Factuality enhanced language models for open-ended text generation. *NeuralPS*, 35:34586–34599.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023a. Halueval: A large-scale hallucination evaluation benchmark for large language models. *CoRR*, abs/2305.11747.

Miaoran Li, Baolin Peng, and Zhu Zhang. 2023b. Self-checker: Plug-and-play modules for fact-checking with large language models. *CoRR*, abs/2305.14623.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Alexander Cosgrove, Christopher D Manning, Christopher Re, Diana Acosta-Navas, Drew Arad Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue WANG, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Andrew Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*. Featured Certification, Expert Certification.

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.

Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *CoRR*, abs/2303.08896.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *CoRR*, abs/2305.14251.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2023. Generating benchmarks for factuality evaluation of language models. *CoRR*, abs/2307.06908.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Weijia Shi, Xiaochuang Han, and et al. 2023. Trusting your evidence: Hallucinate less with context-aware decoding. *arXiv preprint arXiv:2305.14739*.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. 2023. Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214*.

Yuxia Wang, Revanth Gangi Reddy, Zain Muhammad Mujahid, Arnav Arora, Aleksandr Rubashevskii, Jiahui Geng, Osama Mohammed Afzal, Liangming Pan, Nadav Borenstein, Aditya Pillai, et al. 2023. Factcheck-gpt: End-to-end fine-grained document-level fact-checking and correction of llm output. *arXiv preprint arXiv:2311.09000*.

Yuxia Wang, Minghan Wang, Muhammad Arslan Manzoor, Georgi Georgiev, Rocktim Jyoti Das, and Preslav Nakov. 2024. Factuality of large language models in the year 2024. *CoRR*, abs/2402.02420.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V. Le. 2024. Long-form factuality in large language models. *CoRR*, abs/2403.18802.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, Benjamin Newman, Pang Wei Koh, Allyson Ettinger, and Yejin Choi. 2023. The generative AI paradox: "what it can create, it may not understand". *CoRR*, abs/2311.00059.

Zhilin Yang and Peng Qi et al. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP 2018*, pages 2369–2380.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. 2023. Do large language models know what they don't know? In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8653–8665, Toronto, Canada. Association for Computational Linguistics.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. 2023a. How language model hallucinations can snowball. *CoRR*, abs/2305.13534.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023b. Siren's song in the AI ocean: A survey on hallucination in large language models. *CoRR*, abs/2309.01219.

Appendix

A CustChecker Pseudo Code

```python
def claim_processor(document: str) -> List[str]:
    # FactScore
    paragraphs = documents.split("\n")
    sentences = [NLTK(para) for para in paragraphs]
    claims = [call_LLM(sentence, prompt="decompose into atomic claims") for sentence in sentences]

    # FacTool
    claims = call_LLM(document, promot="extract context-independent atomic claims based on the
    ↪   document")

    return claims

def retriever(claim: str, database: DB, retrieval_strategy: obj, search_api_key: str) -> List[str]:
    # offline DB dump
    evidence = retrieval_strategy(claim, database)

    # online web pages by calling API
    evidence = serper_or_serpapi(claim, search_api_key)

    return evidence

def verifier(claim: str, evidence: List[str]) -> bool:
    # call LLMs
    factual_label = call_LLM(claim, evidence, prompt="based on the evidence and your own knowledge,
    ↪   determine whether the claim is true or false.")

    # use NLI models
    stance2factual = {
        "entailment": true,
        "contradiction": false,
        "neutral": "not enough evidence"
    }
    stances = [nli(evid, claim) for evid in evidence]
    majority_stance = majority_vote(factual_labels)
    factual_label = stance2factual[majority_stance]

    return factual_label
```

Figure 3: Pseudo code for the three modules in CustChecker.

B Factuality Datasets

B.1 FactQA Component Datasets

Snowball dataset (Zhang et al., 2023a) comprises three question–answering subsets: primality testing, senator search, and graph connectivity, each with 500 yes/no questions. They aim to investigate snowballing hallucination when a model immediately outputs an incorrect answer (yes or no) as false generated context. Specifically, they prompt the language model to first output a yes/no answer and then to provide explanations. When the immediate answer is wrong, the model tends to continue to snowball the false statements instead of correcting them.

SelfAware (Yin et al., 2023) aims to evaluate LLMs' ability to understand their own limitations and unknowns. This is achieved by assessing models' ability to identify unanswerable or unknowable questions. They compiled a collection of 1,032 unanswerable questions from online platforms like Quora and HowStuffWorks. In addition, they gathered 2,337 answerable questions from sources such as SQuAD, HotpotQA, and TriviaQA, resulting in a total of 3,369 questions.

FreshQA (Vu et al., 2023) is composed of 600 natural, open-ended questions, segmented into four primary categories based on the answer's stability: *never-changing*, for answers that rarely alter, *slow-changing*, for those that evolve over several years, *fast-changing*, for answers that shift within a year or less, and *false-premise*, encompassing questions with factually incorrect premises that need to be countered.

FacTool (Chern et al., 2023) detected factual errors in LLM generations across four different tasks: knowledge-based QA, code generation, mathematical reasoning, and scientific literature review. During model evaluation, they reported both response-level and claim-level accuracy when the responses consist of several claims. We used 50 knowledge-based QA: FacTool-QA in FactQA.

FELM (Chen et al., 2023) introduced a benchmark for factuality evaluation of LLMs. This benchmark collects responses generated from LLMs and annotated factuality labels in a fine-grained manner. The dataset consists of 5 categories, with examples per category as follows: 194 math, 208 reasoning, 125 science, 184 world knowledge (wk), and 136 writing recordings. We used 184 world-knowledge questions, referring to FELM-WK.

Factcheck-Bench (Wang et al., 2023) Factcheck-GPT gathered a total of 94 highly challenging questions from sources including Twitter posts, internal brainstorming, and Dolly-15k, encompassing 678 claims.

FactScore-Bio (Min et al., 2023) selected 183 entities, and collected responses from three LLMs including Davinci-text-003, ChatGPT, and PerplexityAI, and then annotated factual labels (supported, not-supported and irrelevant) for each atomic claim by humans. Specifically, if the atomic claim was clearly not related to the prompt, and thus should be removed from the bio without a validation step, they assigned *Irrelevant*. If the claim was relevant, they validated it based on the English Wikipedia, and labeled it either as *Supported* or *Not-supported*. Additionally, the annotators also edited the text to make it factually correct. The annotators were also asked to correct factual errors and to remove the sentence if the information it contains is entirely off. Their data can be downloaded from `https://drive.google.com/drive/folders/1bLHGu_imkZVtX6O0mpZ-G0-4ofTLM1ZA`. They proposed automatic checking based on retriever + LLM + masked LLM calculating the perplexity to determine the factual labels of atomic claims and to calculate the error rate or a FactScore for an LLM. They further collected responses from 12 LLMs based on another 500 entities and evaluated their factuality using automatic estimators. Overall, they labeled 183 biographies * 3 models = 549 examples, and further used 500 * 12 = 6,000 unlabeled examples.

Domain and Topic There are 482 unique domains and 4,740 unique topics (unique by lexicons without semantic clustering). The top-20 domains are shown in Table 6, accounting for 70% or 4,523 examples. Except for the Snowball dataset: 500 examples for each of primality testing, US senator search and graph connectivity-flight search, there are fewer than 11 examples per topic, generally 1–3 examples.

Domain	**Size**	**Domain**	**Size**
History	771	Science	143
Biography	683	Physics	136
Mathematics	612	Social Sciences	111
Transportation	519	Literature	100
Biology	259	Geography	87
Philosophy	229	Astronomy	82
Technology	208	Economics	69
Entertainment	191	Music	66
Psychology	169	Religion	63
Sports	157	General Knowledge	53
Total			**4,523 (69.8%)**

Table 6: FactQA's top-20 domains and the number of examples from each domain.

B.2 Other Datasets

TruthfulQA (Lin et al., 2022) is a benchmark designed to measure whether a language model is truthful when generating answers to questions and is robustness with respect to false beliefs and misconceptions. It comprises 817 questions spanning 38 categories, including health, law, finance, and politics. The dataset includes both text generation and multiple-choice components, with the multiple-choice questions having a variable number of options. The questions are designed to be "adversarial" to test for weaknesses in the truthfulness of language models rather than testing models on a useful task.

CoVe used four datasets (Dhuliawala et al., 2023). One is selected from Wikidata — listings of people with specific professions born in a certain city (56 questions), and the other one is listing works from specific categories based on Wiki-category (55 questions), e.g., *Name some Mexican animated horror films* or *Name some Endemic orchids of Vietnam*. The third dataset consists of 418 questions sampled from the MultiSpanQA test set with shorter answers per span (up to three tokens per item). They also used the dataset of generated biographies from FactScore.

SelfCheckGPT (Manakul et al., 2023) generated synthetic Wikipedia articles about individuals/concepts from the WikiBio dataset (Lebret et al., 2016) using GPT-3, followed by manual annotation to assess the factuality of each passage at the sentence level.

HaluEval (Li et al., 2023a) comprises 5,000 general user queries accompanied by ChatGPT responses and 30,000 specialized examples from three distinct tasks: question answering, knowledge-grounded dialogue, and text summarization.

Self-Contradictory (Anonymous, 2023) constructed a dataset by sampling language model responses about various topics, consisting of 360 text descriptions covering 30 diverse topics.

FACTOR (Muhlgay et al., 2023) introduced a framework that automatically transforms a factual corpus of interest into a benchmark for evaluating an LM's factuality, and created two benchmarks using this framework: Wiki-FACTOR and News-FACTOR. Wiki-FACTOR is based on the Wikipedia section of The Pile's validation split and consists of 2,994 examples, while News-FACTOR is based on Reuters articles extracted from the RefinedWeb Dataset and consists of 1,036 examples.

HELM (Liang et al., 2023) is a living benchmark designed to enhance the transparency of language models. The dataset implements a multi-metric approach, encompassing various user-facing tasks, domains, and languages. It includes a core set of scenarios and metrics, covering tasks such as question answering, information retrieval, summarization, and toxicity detection across different domains and languages. The dataset is intended to be continuously updated with new scenarios, metrics, and models, and all raw model prompts and completions are released publicly for further analysis.

C Web Client

We develop a web client based on Streamlit, consisting of four interfaces, with each corresponding to one of the three modules, along with a leaderboard, to enhance the user interaction.

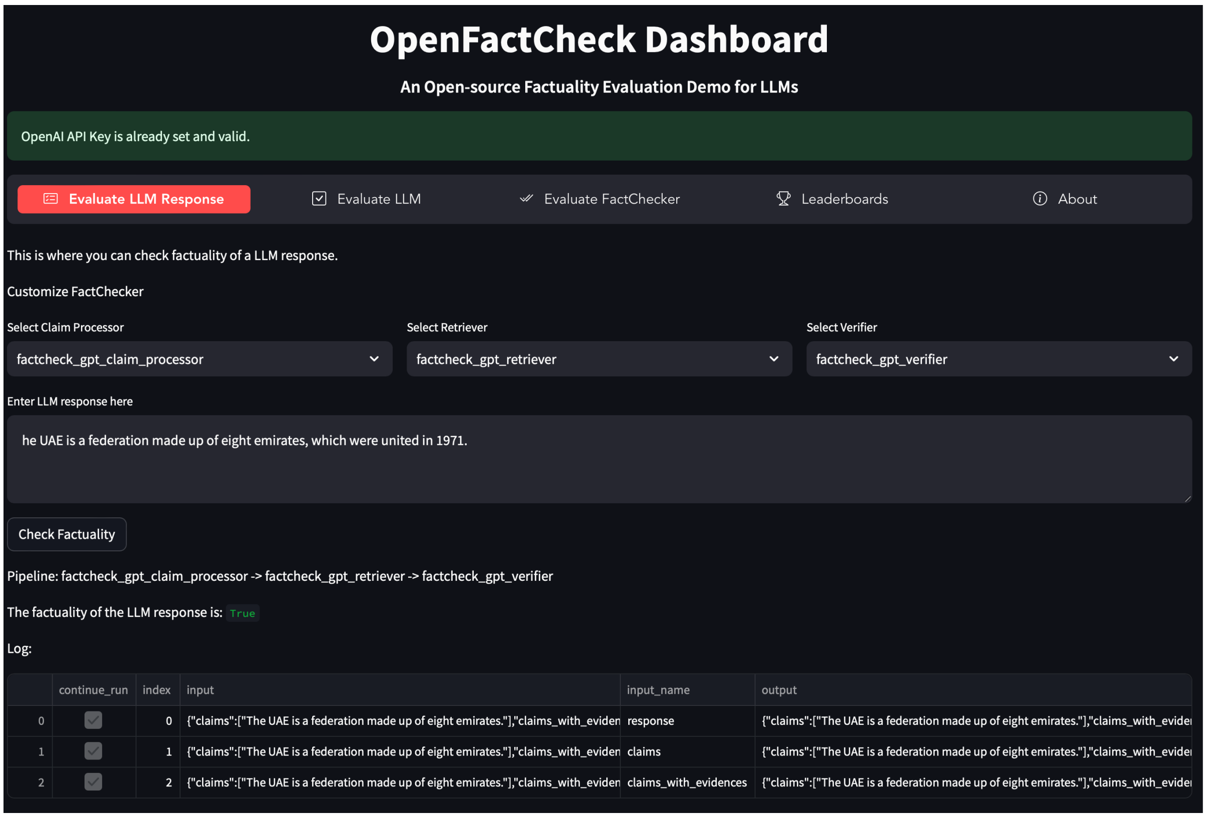
As CUSTCHECKER interface shown in Figure 4, users can freely select different combinations of claim processors, retrievers, and verifiers. When given an input document or claim, the CUSTCHECKER backend (§3.1) executes the fact-checking pipeline. The final verification results and the intermediate processing outcomes are presented on the page for reference.

Figure 5 corresponds to the module of LLMEVAL in §3.2. Users first download our predefined question set and then upload their model responses, the system forwards them to background tasks, using LLMEVAL for evaluation. Afterwards, a comprehensive report is generated and emailed to the user, notifying them of the availability of the report's PDF for download. Moreover, if users consent to publish the evaluation results, we display them on the corresponding leaderboard page.

CHECKEREVAL page in Figure 6 evaluates the performance of fact-checking systems. Users can download claims or documents to be checked from this page, and then use their fact-checking system to predict factuality. The results including True/False, time, and USD costs are subsequently uploaded. We evaluate the submitted fact-checker results based on the ground truth labels of the human-annotated datasets, we rank and display them on the leaderboard.

Leaderboard page in Figure 7 is maintained for both the LLM factuality evaluation and the automatic fact-checking system evaluation. This leaderboard is updated in real time, allowing users to track their performance and to compare it to others. The leaderboard is accessible from the main page, providing a comprehensive overview of the system's performance.

The design principle of our web client is to invoke these functional modules in the form of third-party independent applications, without excessively intervening in the system's architecture. Consequently, our system is made available to users in the form of a library, a command-line toolkit, and a web application.



OpenFactCheck Dashboard

An Open-source Factuality Evaluation Demo for LLMs

OpenAI API Key is already set and valid.

This is where you can check factuality of a LLM response.

Customize FactChecker

Select Claim Processor: factcheck_gpt_claim_processor

Select Retriever: factcheck_gpt_retriever

Select Verifier: factcheck_gpt_verifier

Enter LLM response here

he UAE is a federation made up of eight emirates, which were united in 1971.

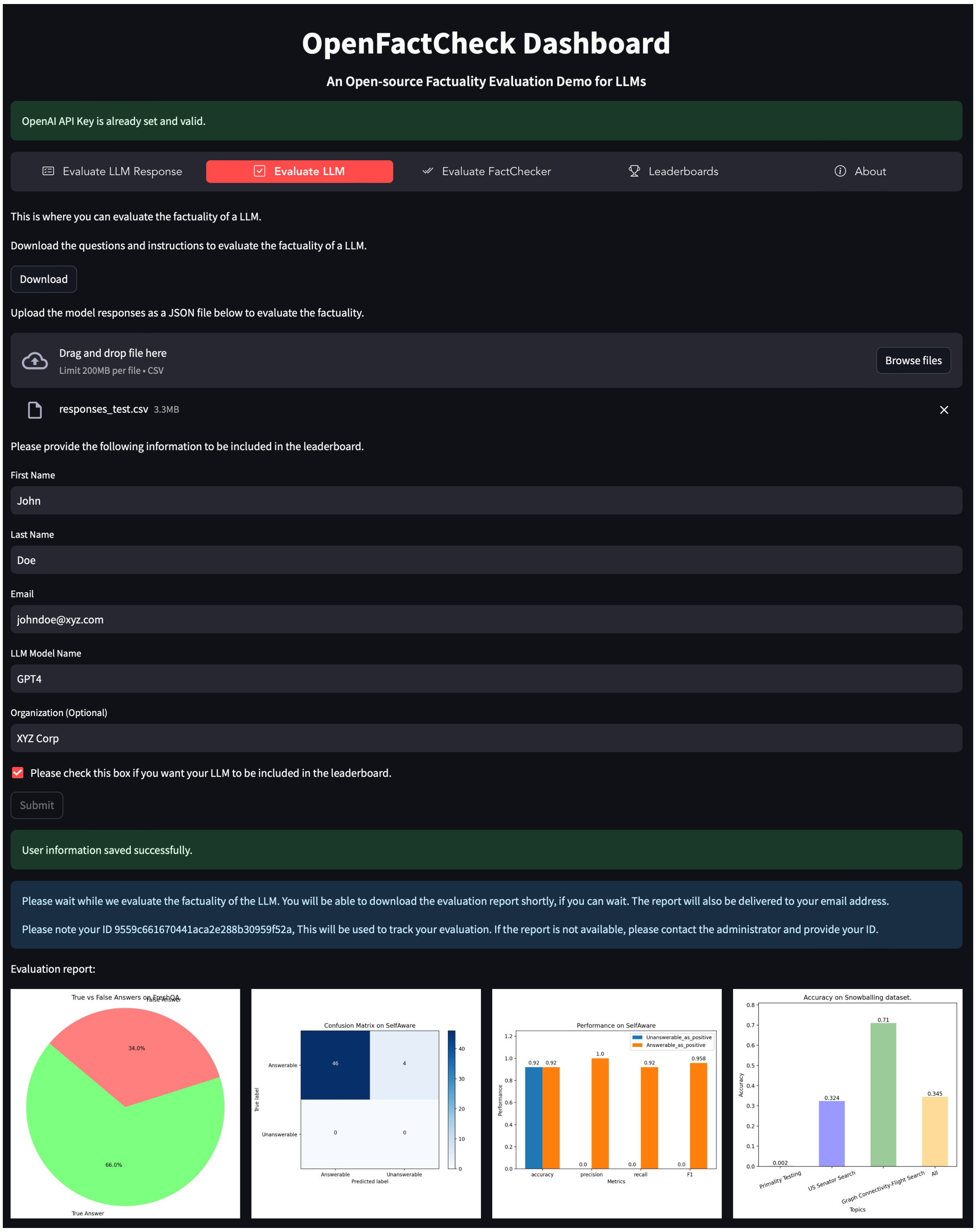
Pipeline: factcheck_gpt_claim_processor -> factcheck_gpt_retriever -> factcheck_gpt_verifier

The factuality of the LLM response is: True

Log:

	continue_run	index	input	input_name	output
0	✓	0	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_eviden	response	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_evide
1	✓	1	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_eviden	claims	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_evide
2	✓	2	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_eviden	claims_with_evidences	{"claims":["The UAE is a federation made up of eight emirates."],"claims_with_evide

Figure 4: The interface of the Customized Fact-checking System page. The response "*The UAE is a federation made up of eight emirates, which were united in 1971*" is a random example for demonstration purposes. We can see the final *False* judgment and the intermediate results.



OpenFactCheck Dashboard

An Open-source Factuality Evaluation Demo for LLMs

OpenAI API Key is already set and valid.

This is where you can evaluate the factuality of a LLM.

Download the questions and instructions to evaluate the factuality of a LLM.

Upload the model responses as a JSON file below to evaluate the factuality.

Drag and drop file here
Limit 200MB per file • CSV

responses_test.csv 3.3MB

Please provide the following information to be included in the leaderboard.

First Name
John

Last Name
Doe

Email
johndoe@xyz.com

LLM Model Name
GPT4

Organization (Optional)
XYZ Corp

[✓] Please check this box if you want your LLM to be included in the leaderboard.

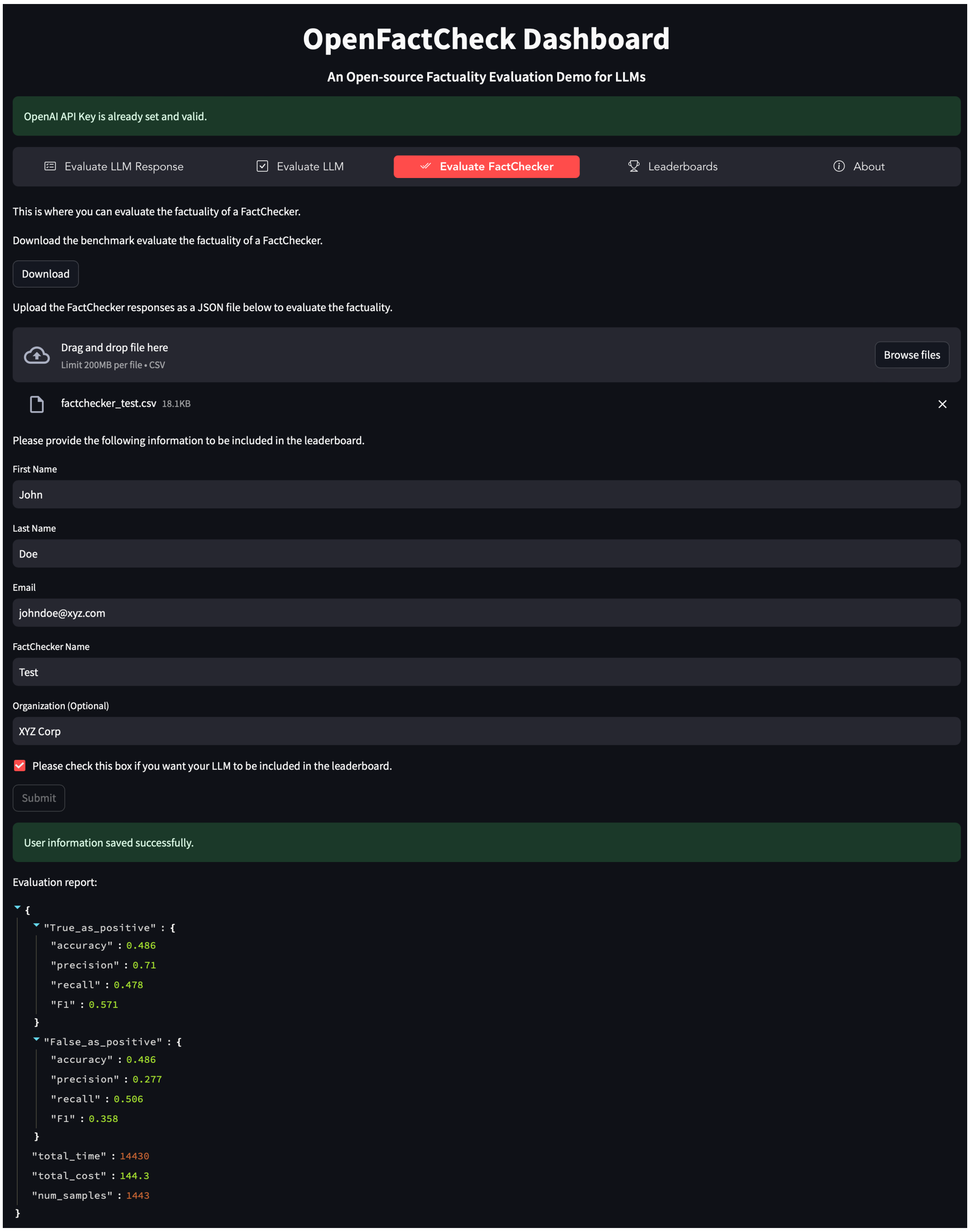
User information saved successfully.

Please wait while we evaluate the factuality of the LLM. You will be able to download the evaluation report shortly, if you can wait. The report will also be delivered to your email address.

Please note your ID 9559c661670441aca2e288b30959f52a, This will be used to track your evaluation. If the report is not available, please contact the administrator and provide your ID.

Evaluation report:

Figure 5: The interface of the LLM Factuality Evaluation page. A random evaluation result is shown for demonstration purposes.



OpenFactCheck Dashboard

An Open-source Factuality Evaluation Demo for LLMs

OpenAI API Key is already set and valid.

This is where you can evaluate the factuality of a FactChecker.

Download the benchmark evaluate the factuality of a FactChecker.

Upload the FactChecker responses as a JSON file below to evaluate the factuality.

factchecker_test.csv 18.1KB

Please provide the following information to be included in the leaderboard.

First Name: John

Last Name: Doe

Email: johndoe@xyz.com

FactChecker Name: Test

Organization (Optional): XYZ Corp

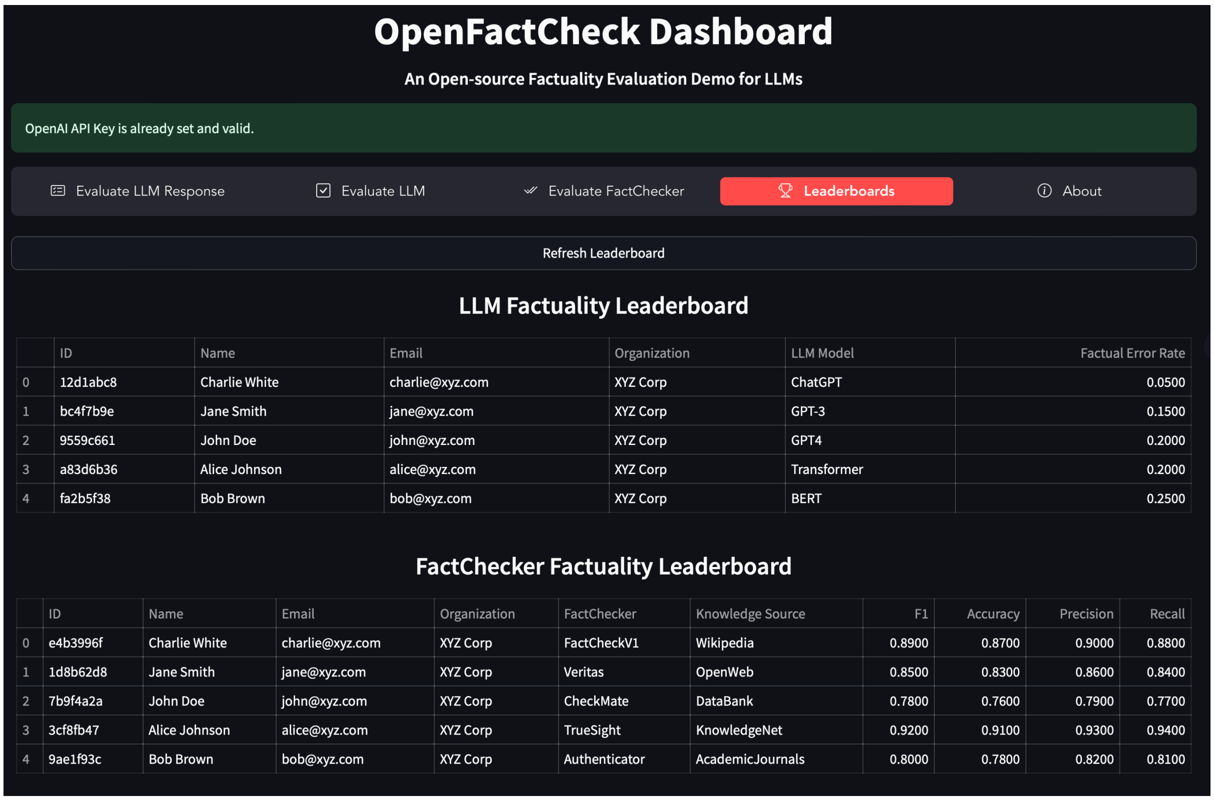
[✓] Please check this box if you want your LLM to be included in the leaderboard.

User information saved successfully.

Evaluation report:

```
{
  "True_as_positive" : {
    "accuracy" : 0.486
    "precision" : 0.71
    "recall" : 0.478
    "F1" : 0.571
  }
  "False_as_positive" : {
    "accuracy" : 0.486
    "precision" : 0.277
    "recall" : 0.506
    "F1" : 0.358
  }
  "total_time" : 14430
  "total_cost" : 144.3
  "num_samples" : 1443
}
```

Figure 6: The interface of the Automatic Fact-checker Evaluation page. A random factchecker evaluation result is shown for demonstration purposes.



OpenFactCheck Dashboard

An Open-source Factuality Evaluation Demo for LLMs

OpenAI API Key is already set and valid.

LLM Factuality Leaderboard

	ID	Name	Email	Organization	LLM Model	Factual Error Rate
0	12d1abc8	Charlie White	charlie@xyz.com	XYZ Corp	ChatGPT	0.0500
1	bc4f7b9e	Jane Smith	jane@xyz.com	XYZ Corp	GPT-3	0.1500
2	9559c661	John Doe	john@xyz.com	XYZ Corp	GPT4	0.2000
3	a83d6b36	Alice Johnson	alice@xyz.com	XYZ Corp	Transformer	0.2000
4	fa2b5f38	Bob Brown	bob@xyz.com	XYZ Corp	BERT	0.2500

FactChecker Factuality Leaderboard

	ID	Name	Email	Organization	FactChecker	Knowledge Source	F1	Accuracy	Precision	Recall
0	e4b3996f	Charlie White	charlie@xyz.com	XYZ Corp	FactCheckV1	Wikipedia	0.8900	0.8700	0.9000	0.8800
1	1d8b62d8	Jane Smith	jane@xyz.com	XYZ Corp	Veritas	OpenWeb	0.8500	0.8300	0.8600	0.8400
2	7b9f4a2a	John Doe	john@xyz.com	XYZ Corp	CheckMate	DataBank	0.7800	0.7600	0.7900	0.7700
3	3cf8fb47	Alice Johnson	alice@xyz.com	XYZ Corp	TrueSight	KnowledgeNet	0.9200	0.9100	0.9300	0.9400
4	9ae1f93c	Bob Brown	bob@xyz.com	XYZ Corp	Authenticator	AcademicJournals	0.8000	0.7800	0.8200	0.8100

Figure 7: The interface of the Leaderboard page. Random data is shown for demonstration purposes.

D LLM Responses

Word-level Length Table 7 shows word length of three LLMs' responses, including LLaMA-2 7B, 13B and GPT-4.

Dataset ↓	LLaMA-2 7B	LLaMA-2 13B	GPT-4
FacTool-QA	127.3	129.5	39.5
FELM-WK	131.0	125.5	62.8
Factcheck-Bench	152.7	143.9	117.3
Seven datasets	132.1	121.9	82.2

Table 7: Word length for responses of three LLMs over the datasets of FacTool-QA, FELM-WK, Factcheck-Bench and over seven evaluation datasets.

Claim Statistics Figure 8 shows the number of atomic claims extracted from LLaMA-2 7B, 13B and GPT-4 responses elicted from the prompts of FacTool-QA, FELM-WK and Factcheck-Bench. There are 50, 184 and 94 prompts in the three datasets respectively, decomposing approximately into 400, 1,600 and 800 atomic claims for LLaMA-2 responses. The answers of GPT-4 are generally shorter, resulting in 200, 800, and 600 claims. Note that the three original datasets were annotated with factual labels for GPT-3.5-Turbo responses, and there are 233, 532 and 678 claims for FacTool-QA, FELM-WK and Factcheck-Bench.

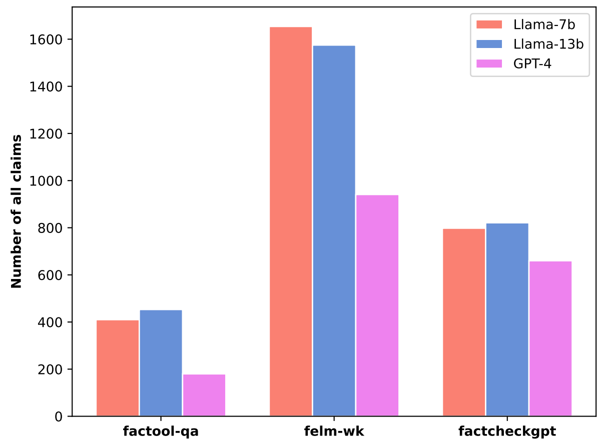


Figure 8: The number of extracted atomic claims using *FacTool* across responses of LLaMA-2 7B, 13B, and GPT-4.

E OpenAI GPT Models

Table 8 shows the price, the input context window and the maximum output tokens for two GPT models: *gpt-3.5-turbo-0125* and *gpt-4-turbo-2024-04-09*. The price of the latter is 20 times the price of the former.

Model	Input	Output	Input context window	Maximum output tokens
gpt-3.5-turbo-0125	\$0.5 / 1M tokens	\$1.5 / 1M tokens	128k	4096
gpt-4-turbo-2024-04-09	\$10 / 1M tokens	\$30 / 1M tokens	16k	4096

Table 8: Price, input context window, and maximum output tokens for two GPT models.