InFi-Check: Interpretable and Fine-Grained Fact-Checking of LLMs on Grounding Documents

Anonymous ACL submission

Abstract

Large language models (LLMs) often hallucinate, yet most existing fact-checking methods treat factuality evaluation as a binary classification problem, offering limited interpretability and failing to capture fine-grained error types. In this paper, we introduce InFi-Check, a framework for interpretable and fine-grained fact-checking of LLMs on grounding documents. Specifically, we first propose a controlled data synthesis pipeline that generates high-quality data with explicit evidence, fine-grained error type labels, justifications, and corrections. Based on this, we construct large-scale training data and a manually verified benchmark, called InFi-Check-FG, for fine-grained fact-checking of LLM outputs on grounding documents. Building on these high-quality training data, we further propose InFi-Checker, which can jointly provide supporting evidence, classify fine-grained error types, and produce justifications along with corrections. Experiments show that InFi-Checker achieves state-of-the-art performance on InFi-Check-FG benchmark and strong generalization across various downstream tasks, significantly improving the utility and trustworthiness of factuality evaluation.

1 Introduction

Recent advances in large language models (LLMs) have fundamentally transformed the paradigm of human-computer interaction (Achiam et al., 2023; DeepSeek-AI et al., 2025a). However, LLMs are still prone to producing hallucinations (Ji et al., 2023; Huang et al., 2025), posing significant risks and severely compromising their trustworthiness. To address this, research has focused on factuality evaluation and alignment frameworks (Zhao et al., 2023; Lin et al., 2024). Central to these efforts is *fact-checking on grounding documents* (Tang et al., 2024a; Lei et al., 2025; Seo et al., 2025), which assesses whether the generated claims can be strictly supported by a reliable knowledge source.

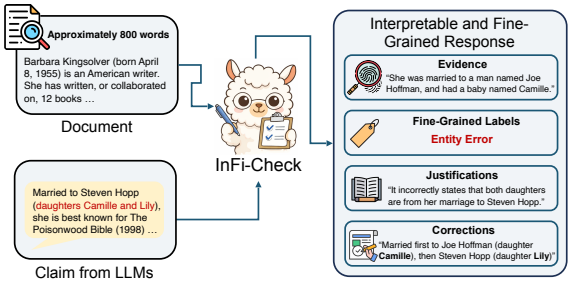

Figure 1: The illustration of our InFi-Check. InFi-Check can simultaneously provide the corresponding evidence, fine-grained labels, justifications, and corrections.

Recent studies focus on how to effectively train a fact-checking model used to evaluate the factuality of LLM-generated responses on grounding documents, including entailment-based (Kryściński et al., 2019; Goyal and Durrett, 2021), question-answering-based (Wang et al., 2020; Durmus et al., 2020; Fabbri et al., 2022), atomic-fact-based (Min et al., 2023), and synthetic-data-based (Tang et al., 2024a; Lei et al., 2025; Seo et al., 2025) methods. However, these methods simply treat the fact-checking task as a binary prediction task, classifying the entire response from LLMs as hallucinated or not. Consequently, these methods present several notable limitations: (1) *Lack of Interpretability:* Existing fact-checking models typically output a single predicted label for the entire LLM-generated response (Zha et al., 2023; Tang et al., 2024a), without providing the explicit justification or supporting evidence. This lack of interpretability makes it difficult to analyze model hallucination patterns and limits its practical utility for localizing hallucinations and performing targeted corrections in real-world applications. (2) *Lack of Fine-Grained Fact Checking:* Also, these fact-checking models typically formulate hallucination detection as a binary classification problem, merely predicting whether a response is factual or hallucinated. However, hallucinations are not all the same and can appear in

different types (Pagnoni et al., 2021a; Mitra et al., 2025; Zhang et al., 2025), such as incorrect entities, fabricated facts, or unsupported relations. By collapsing these distinct error types into a single label, i.e., hallucinated, this oversimplified formulation fails to provide fine-grained distinctions among hallucinations, thereby limiting detailed error analysis and specified revision, hindering the development of targeted strategies for hallucination mitigation.

To bridge these gaps, we propose a novel framework called **InFi-Check** to enable interpretable and fine-grained fact-checking for comprehensive factuality evaluation of LLM outputs on grounding documents. To achieve this end, we first introduce an effective data synthesis pipeline to automatically generate fact-checking data that contains four key output elements. These elements are typically provided by professional fact-checkers in real-world scenarios, including explicit evidence support, error type identification, justifications, and corrections. In this way, models trained on such data can provide fine-grained predictions and detailed justifications for real-world users. To ensure the data quality and avoid labeling errors, our pipeline incorporates the controlled generation strategies to guide LLMs in generating corresponding claims under given fine-grained error types. Specifically, for the given document-claim pairs, we first query the advanced LLM (e.g., GPT-4o (Achiam et al., 2023)) to find grounding sentences as explicit evidence support. Subsequently, we provide specific fine-grained error types along with the document-claim pair as input, requiring the LLMs to first follow our designed structured reasoning process and generate corresponding justifications and claims exhibiting the specified error types. By doing so, we can control the inclusion of specified fine-grained hallucinations in newly generated claims from advanced LLMs and ensure that there are no labeling errors. Also, by comparing the original claim with the newly synthesized claim, we can obtain corresponding corrections, enabling transparent justification of fact-checking decisions. To ensure the data diversity, our designed pipeline involves 9 hallucination construction strategies that cover the diverse yet nuanced hallucination patterns commonly observed in advanced LLMs such as GPT-4o. Based on our well-designed pipeline above, we construct a training set as well as a manually corrected benchmark InFi-Check-FG for evaluating the fine-grained classification capabilities of fact-checking models. By using the training data from our well-designed data synthesis pipeline, we introduce InFi-Checker, an advanced fact-checking model capable of fine-grained hallucination detection and interpretable analysis. The InFi-Checker can perform comprehensive tasks including identifying relevant evidence, detecting fine-grained error types, and providing justification and direct corrections.

Our extensive experiments show that even the most advanced LLMs, such as GPT-5 (OpenAI, 2025a), still struggle to capture fine-grained error types. In contrast, our InFi-Checker achieves state-of-the-art performance on InFi-Check-FG benchmark, and shows strong generalization across diverse scenarios, covering question answering (QA), summarization, and retrieval-augmented generation (RAG). Different from previous fact-checking models, InFi-Checker can also offer fine-grained analysis and the corresponding justifications to improve the interpretability and trustworthiness.

2 Related Work

LLMs are prone to hallucinations across various settings, generating information that cannot be supported by any source (Ji et al., 2023; Huang et al., 2025). Therefore, verifying whether LLM-generated responses contain hallucinated claims is crucial (Goyal and Durrett, 2021). Recent works focus on using reliable knowledge bases or documents to identify hallucination in LLM outputs, namely fact-checking on grounding documents (Tang et al., 2024a). For example, Kryściński et al. (2019) proposed training specific models for fact-checking, which outperform overlap-based metrics such as ROUGE. Early alternatives employed entailment or question-answering models to check document-claim consistency (Laban et al., 2022; Fabbri et al., 2022), while subsequent advancements improved performance via model ensembling (Zha et al., 2023) and semantic-graph representations (Ribeiro et al., 2022). Building on these foundations, researchers have precisely annotated hallucinations in model-generated claims to assemble datasets for quantitative fact-checking and factuality evaluation (Cao and Wang, 2021; Pagnoni et al., 2021a; Zhang et al., 2024).

Recent studies have turned to utilize the power of LLMs to train more capable fact-checking models (Ádám Kovács and Recski, 2025; Sun et al., 2025; Yeh et al., 2026). For example, MiniCheck (Tang et al., 2024a) uses advanced LLMs to synthesize training data and surpasses prior fact-checking

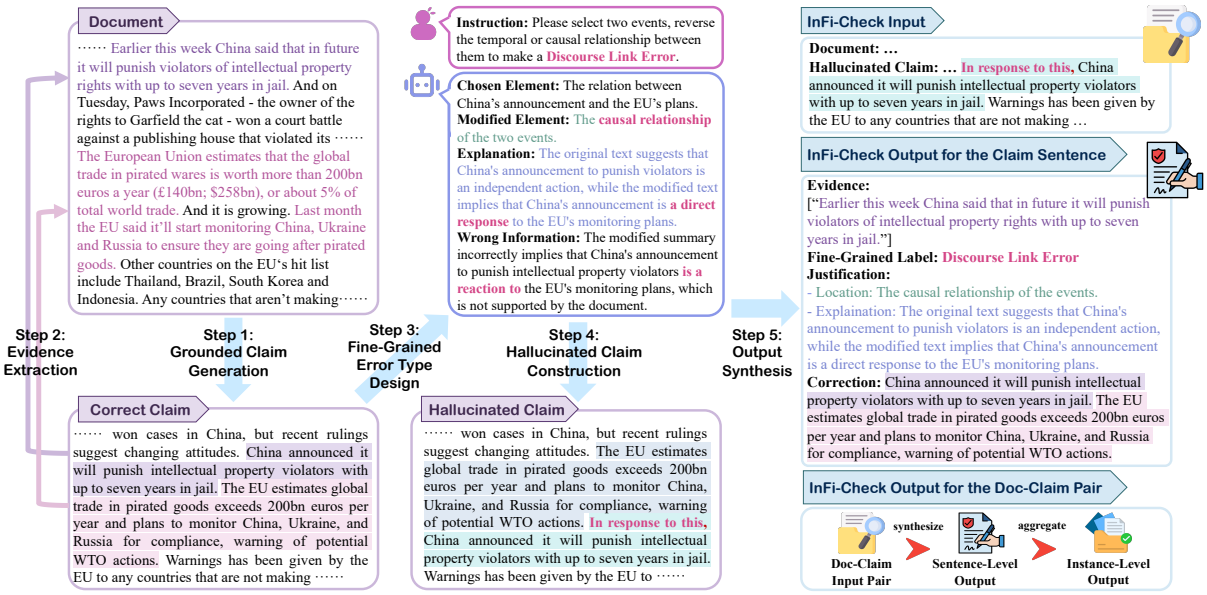

Figure 2: Overview of the InFi-Check pipeline. Some of the text is simplified for better demonstration.

methods. FactCG (Lei et al., 2025) further enhances synthetic data with knowledge graphs to improve the performance of fact-checking models. ClearCheck (Seo et al., 2025) uses synthetic data with multi-task training, allowing the model to perform reasoning before answering. However, despite these advances in fact-checking performance, current models continue to generate only binary predictions, lacking interpretable justifications and fine-grained error labels that can support users, limiting the utility and trustworthiness. Different from these works, our model can jointly provide supporting evidence, classify fine-grained error types, and produce justifications along with corrections, significantly improving the utility and trustworthiness of realistic and user-friendly fact-checking.

3 Methodology

In this section, we introduce InFi-Check, a controlled data synthesis pipeline designed to generate high-quality, interpretable fact-checking data. The core of our InFi-Check lies in systematically constructing grounded claims with fine-grained, realistic hallucinations, alongside their corresponding diagnostic analysis. As illustrated in Figure 2, the pipeline operates in five sequential stages: (1) *Generate Grounded Claim*, (2) *Extract Supporting Evidence*, (3) *Design Hallucination with Chain-of-Thought*, (4) *Construct Hallucinated Claim*, and (5) *Synthesize Interpretable Output*. A key advantage of this pipeline is its scalability and independence from the original document corpus; by varying hallucination injection points and generating new claims, it can be arbitrarily expanded. The prompts used across all stages are provided in Appendix F.

3.1 Controlled Data Synthesis Pipeline

Stage 1 & 2: Grounded Claim Generation and Evidence Extraction. We begin with a collection of source documents. Existing datasets often fall short in terms of claim complexity and factual coverage, as the provided claims are not always fully supported by the corresponding documents. To address this limitation, we generate claims directly conditioned on each source document. For every document, a claim is produced, and each sentence in the claim is automatically annotated with its corresponding *grounding sentences* as evidence from the source document. To ensure claim quality, we apply an iterative refinement process based on the extracted grounding sentences, including majority voting and rewriting. We further conduct human evaluation to validate the reliability of this pipeline. Details are provided in the Appendix A.

Stage 3: Fine-Grained Error Type Design. This stage is the cornerstone of our method, where we strategically implant controlled hallucinations into the grounded claims. We adopt and extend the fine-grained error typology from Pagnoni et al. (2021b), which is applicable to summarization and other grounded generation tasks. This taxonomy is linguistically motivated (Fillmore, 1976; Brown and Yule, 1983) and has been empirically validated to cover about 99% of hallucinations. We exclude

Error Category	Abbr.	Construction Strategy
Predicate Error	PredE	Swap Relation
		Modify Predictions
Entity Error	EntE	Swap Entities
		Compress Phrases
Circumstance Errors	CircE	Swap Circumstances
Co-Reference Error	CorefE	Swap Pronouns
		Merge Sentences
Discourse Link Error	LinkE	Reverse Logic
Extrinsic Error	OutE	Add Extrinsic Information

Table 1: Fine-grained error types and corresponding construction strategies in InFi-Check. Detailed descriptions and examples can be found in Table 29.

grammatical errors as they pertain to fluency rather than factuality. Our construction system encompasses six major categories, as summarized in our Table 1, each implemented via specific strategies.

To ensure high-quality and diverse errors while enabling interpretable analysis, we leverage LLMs following a structured chain-of-thought (CoT) process. For a given grounded claim and its evidence, we specify a target error type and prompt the model to: (a) analyze the original content, (b) execute the corresponding construction strategy (e.g., swap an entity), and (c) produce a detailed justification for the change. This step-by-step reasoning not only improves the controllability of the constructed errors but also provides a transparent record of how each hallucination was introduced, forming the basis for interpretable outputs in Stage 5.

Stage 4 & 5: Hallucinated Claim Construction and Output Synthesis. Using the CoT from Stage 3, we generate the final hallucinated claim sentence containing the specified hallucination. Ultimately, we can synthesize the *interpretable output*: a structured analysis containing the erroneous sentence, its grounding evidence (or lack thereof), the hallucination category, the justification from the CoT, and the corrected sentence. This final output is organized in a sentence-by-sentence manner (see Appendix E for examples), providing a clear, traceable reasoning path from error construction to diagnosis. The structured format offers two key benefits: (1) ***Interpretability through traceable reasoning***, making model decisions transparent and easing manual verification by presenting grounding contexts and corrections side-by-side; and (2) ***Higher quality from controlled generation***, as the CoT process and explicit output schema enforce rigorous, context-aware error construction and analysis. This output serves as the gold standard for training and evaluating fact-checking models.

3.2 Fine-Grained Error Type Construction

Building upon the error typology outlined in Stage 3, this subsection details the specific strategies employed to construct each category of hallucination. Our methodology adapts and extends prior work on error construction (Kryściński et al., 2019; Cao and Wang, 2021; Pagnoni et al., 2021b), introducing novel strategies to capture subtle and challenging error types often overlooked in existing datasets.

Semantic-Level Hallucinations. These errors involve inaccuracies within a single proposition, including Predicate Errors (PredE), Entity Errors (EntE), and Circumstance Errors (CircE).

- *Element Swapping*: Following established swapping techniques (Kryściński et al., 2019; Cao and Wang, 2021), we perturb a target element (e.g., an **entity**, **predicate**, or **circumstantial phrase**) within a claim sentence into a semantically congruent but factually incorrect alternative from the source document.
- *Modifying Predictions*: This strategy targets the confusion between speculative language (e.g., "predicted," "might") and factual assertions. We guide the LLM to identify sentences containing modal verbs or speculative phrases and alter them to state the speculated outcome as a fact (e.g., changing "The company *forecast* growth" to "The company *grew*").
- *Compressing Phrases*: This strategy generates errors through oversimplification, where specific technical terms or nuanced descriptions are replaced with overly broad terms, distorting meaning (e.g., compressing "net revenue attributable to the parent company" to "net revenue"). A two-stage LLM verification process ensures the compression introduces a factual distortion rather than mere paraphrasing.

Discourse-Level Hallucinations. These errors span multiple sentences, disrupting coherence and reference. They include Co-reference Errors (CorefE) and Discourse Link Errors (LinkE).

- *Swapping Pronouns*: We extend Kryściński et al. (2019) by swapping pronouns of all types, creating inconsistent references while

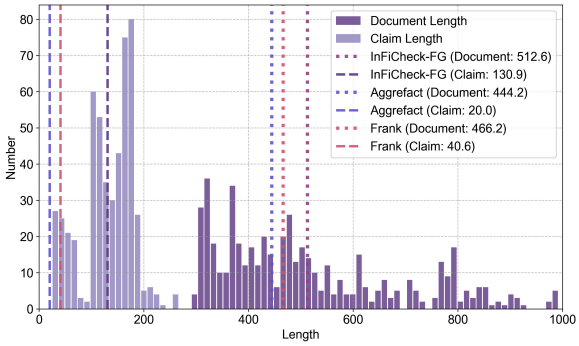

Figure 3: Document and claim length (words) distribution of InFi-Check-FG with average length comparison.

preserving grammatical correctness. To increase difficulty, entities are first anonymized into pronouns before substitution.

- *Merging Sentences*: To simulate conflation errors, given two semantically related claim sentences, we construct a plausible merged sentence by assigning actions or attributes from both sentences to a single subject.
- *Reverse Logical Relationship*: Given two events connected by a discourse relation r (e.g., temporal or causal), we construct a modified relation $\hat{r}$ by reversing the original direction. The claim is then rewritten to express $\hat{r}$ while keeping event contents unchanged.

Extrinsic Hallucinations. Extrinsic Errors (OutE) introduce external, unsubstantiated details to a valid claim. To construct these, we perturb a grounded sentence by injecting plausible but entirely unsupported entities, events, or numerical values, thereby ensuring the augmented information remains strictly extrinsic to the source document.

4 InFi-Check Dataset and Model

Building upon the controlled synthesis pipeline detailed in the Section 3, we construct the interpretable and fine-grained fact-checking training data, namely InFi-Check-TR without human effort, and further propose a manually verified benchmark InFi-Check-FG. Subsequently, we train our model InFi-Checker based on InFi-Check-TR. This section shows the details of these core resources.

4.1 Dataset and Benchmark

InFi-Check serves a dual purpose: it automatically creates a large-scale training corpus InFi-Check-TR, and also a high-quality and challenging benchmark InFi-Check-FG for evaluating fine-grained classification capabilities of fact-checking models.

Construction Process. To ensure factual grounding and diversity, our dataset construction begins with source documents rather than existing summaries. We randomly sample news and encyclopedic documents of varying lengths from Gupta et al. (2022) and Xu and Lapata (2019) datasets. For each document, we generate a grounded claim using multiple LLMs via iterative rewriting to mitigate bias and ensure quality, with each claim sentence linked to its evidential grounding sentences in the source (see Appendix A for details). Subsequently, for each document-claim pair, we apply the error construction strategies from InFi-Check to generate one hallucinated claim for each strategy, while retaining the original claim as a "No Error" sample, resulting in 10 total samples per pair.

InFi-Check-TR Details. Our InFi-Check-TR comprises 15,660 samples for training, with more statistical details in Appendix B. To verify the quality of InFi-Check-TR, we conducted a human evaluation where expert annotators evaluated 100 randomly sampled instances across four dimensions: claims, evidence sentences, justifications, and hallucination validity. Results in Table 27 show a 95% agreement rate on hallucination validity, demonstrating the capability of the InFi-Check pipeline.

InFi-Check-FG Details. To further ensure evaluation reliability, we construct InFi-Check-FG based on manual check and from a distinct document split, which is a high-quality benchmark that consists of 519 manually verified samples. As illustrated in Figure 3, claims in InFi-Check-FG exhibit longer length compared to existing benchmarks, increasing complexity by preserving richer source-document details, thereby introducing more subtle and challenging hallucinations for evaluation. More details are provided in Appendix A.

4.2 Fact-Checking Model: InFi-Checker

Finally, we train our model InFi-Checker on the synthetic training data using supervised fine-tuning (SFT). The model is trained to perform a comprehensive, structured output task: given a document and a claim, it identifies grounding evidence, classifies the fine-grained error type, and provides a natural language justification alongside a direct correction. This end-to-end training regime, powered by the rich annotations, enables InFi-Checker to deliver interpretable, fine-grained fact-checking for LLM outputs, moving beyond binary classification.

Model	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE	BAcc
The Open-Source Models								
Llama-3.1-8B-Instruct	18.82	9.52	31.03	3.13	0.00	0.00	61.11	17.66
Llama-3.1-8B-Instruct※	32.94	4.76	13.79	26.56	0.00	0.00	88.49	23.79
Qwen3-8B	76.47	52.38	44.83	50.00	0.00	50.00	61.90	47.94
Qwen3-8B※	76.47	52.38	24.14	70.31	0.00	50.00	80.16	50.49
The State-of-the-Art LLMs								
Claude-3.7-Sonnet	70.59	66.67	34.48	37.50	26.67	81.58	53.97	53.07
Claude-3.7-Sonnet※	68.24	71.43	34.48	21.88	23.33	78.95	46.03	49.19
DeepSeek-V3.2-NoThink	78.82	76.19	58.62	9.38	26.67	86.84	80.95	59.64
DeepSeek-V3.2-NoThink※	75.29	71.43	55.17	15.62	13.33	89.47	**94.05**	59.19
GPT-4o	80.00	76.19	44.83	50.00	3.33	84.21	84.92	60.50
GPT-4o※	84.71	61.90	44.83	75.00	0.00	94.74	91.67	64.69
GPT-4.1	85.88	76.19	37.93	42.19	0.00	63.16	75.79	54.45
GPT-4.1※	85.88	76.19	41.38	56.25	3.33	68.42	80.95	58.91
GPT-5	63.53	57.14	62.07	42.19	16.67	60.53	41.27	49.06
GPT-5※	63.53	61.9	51.72	45.31	13.33	47.37	41.27	46.35
o3	80.00	61.90	58.62	43.75	0.00	81.58	80.16	58.00
o3※	81.18	57.14	58.62	56.25	3.33	81.58	87.30	60.77
InFi-Checker (Ours)								
InFi-Checker-Llama	**95.29**	90.48	79.31	95.31	86.67	**100.00**	89.29	90.91
InFi-Checker-Qwen	93.51	**91.67**	**87.50**	**100.00**	**88.89**	96.77	88.01	**92.34**

Table 2: Results (%) on InFi-Check-FG. We display the accuracy of each error type, as well as the balanced accuracy (BAcc), which calculates the average accuracy of all error types. Best performances are marked in bold. For baselines, models marked with ※ are tested in a one-shot setting, while the others are tested in a zero-shot setting.

Model	0-shot	1-shot
The Open-Source Models		
Llama-3.1-8B-Instruct	66.13	71.49
Qwen3-8B	65.31	70.16
The State-of-the-Art LLMs		
Claude-3.7-Sonnet	65.89	63.42
DeepSeek-V3.2-NoThink	66.99	77.14
GPT-4o	74.37	74.75
GPT-4.1	75.19	75.19
GPT-5	76.53	75.70
o3	75.54	74.57
InFi-Checker (Ours)		
InFi-Checker-Llama	77.17	
InFi-Checker-Qwen	**77.20**	

Table 3: Fine-grained results on FRANK, with balanced accuracy calculated in a binary mapped setting due to error types overlapping (See Appendix C for details).

5 Experiment

5.1 Experimental Setup

Implementation of InFi-Checker. We implement Llama-3.1-8B-Instruct and Qwen3-8B as backbones for InFi-Checker, using InFi-Check-TR as training dataset and conducted supervised fine-tuning (details in Appendix C.3).

Other Benchmarks. To evaluate InFi-Checker's out-of-distribution generalization, we conduct experiments across existing multiple factuality evaluation benchmarks, including *(1) fine-grained hallucination labeling benchmark*, where we adapt the **FRANK** benchmark, which is a human-annotated benchmark that uses the same fine-grained hallucination label taxonomy as InFi-Check-FG. *(2) commonly used binary hallucination labeling benchmark*, where we select benchmarks of diverse text sources and generation tasks: **ClaimVerify** (Liu et al., 2023) and **Factcheck-Bench** (Wang et al., 2024) for search queries and responses, **ExpertQA** (Malaviya et al., 2024) and **REVEAL** (Jacovi et al., 2024) for QA, as well as **MediaSum** and **MeetingBank** for dialogue summarization (Tang et al., 2024b). We test in fine-grained settings for FRANK, and binary settings for other benchmarks, as they do not have fine-grained labeling. Appendix C provides more implementation details.

Baselines. We validate InFi-Checker through an extensive comparison with two separate groups of competitive baselines. On InFi-Check-FG and FRANK, which require fine-grained error type labeling, we employ **State-of-the-Art LLMs** including: (1) the backbones of InFi-Checker, **open-source models**: Llama-3.1-8B-Instruct (Kassianik et al., 2025) and Qwen3-8B (Yang et al., 2025); and (2) **closed-source models**: GPT-4o (Jaech et al., 2024), GPT-4.1 (OpenAI, 2025b), GPT-5 (OpenAI, 2025a), o3 (OpenAI, 2025c), Claude-3.7-Sonnet (Anthropic, 2025) and DeepSeek-V3.2 (DeepSeek-AI et al., 2025b). Baselines are tested following the InFi-Check reasoning format, and intentionally limit the number of demonstration examples to zero/one-shot to mitigate degradation from excessive prompt length (see Appendix F for three/five-shot result). For binary fact-checking benchmarks, we include **specialized models** specif-

Model	Claim Verify	Expert QA	Factcheck-Bench	REVEAL	MediaSum	MeetingBank	Average
The Open-Source Models							
Llama-3.1-8B-Instruct	63.6	49.8	69.8	78.2	50.8	62.3	62.4
Qwen3-8B	**66.1**	**55.6**	**78.4**	**83.2**	**77.7**	**74.2**	**72.5**
The State-of-the-Art LLMs							
Claude-3.7-Sonnet	**83.7**	74.4	86.9	88.0	**85.4**	84.0	83.7
DeepSeek-V3.2-NoThink	75.4	74.4	87.9	91.0	65.5	82.9	79.5
GPT-4o	78.3	68.3	86.0	86.9	71.5	76.9	78.0
GPT-4.1	81.6	**80.3**	**91.3**	93.2	75.9	**86.3**	84.8
GPT-5	87.7	75.9	90.4	**93.7**	80.2	87.6	**85.9**
o3	83.3	79.6	86.9	92.2	82.9	83.8	84.8
Specialized Fact-Checking Models							
ClearCheck (COT)	85.4	72.7	87.9	87.0	67.8	75.8	79.4
AlignScore-large	79.8	75.0	83.7	92.2	75.8	76.5	80.5
FactCG	76.2	75.3	**89.0**	90.0	79.1	71.9	80.3
MiniCheck	85.6	72.9	86.8	**91.0**	74.3	77.8	81.4
InFi-Checker-Llama	75.9	**78.3**	83.7	87.7	73.5	65.8	77.5
InFi-Checker-Qwen	**89.6**	75.7	88.0	90.0	**80.4**	**78.5**	**83.7**

Table 4: Macro-F1 (%) on six binary fact-checking benchmarks. Of note, **Bold** and underline highlight the best Macro-F1 within each group of baselines and the best overall Macro-F1, respectively.

ically optimized for binary factuality evaluation: **ClearCheck** (Seo et al., 2025), that leverages multi-task training for robustness, **AlignScore-large** (Zha et al., 2023), a holistic metric using a unified alignment function, **FactCG** (Lei et al., 2025), which enhances training data via knowledge graphs, and **MiniCheck** (Tang et al., 2024a), the state-of-the-art binary evaluator that utilizes a novel document-claim pair synthesis method for training data.

5.2 InFi-Check-FG Results

The performance of various models on InFi-Check-FG is detailed in Table 2; from these results, we draw the following critical observations:

InFi-Checker consistently outperforms all baselines on InFi-Check-FG, especially on complex data. InFi-Checker achieves a substantial 27.65% improvement in balanced accuracy over the second-best performer, one-shot GPT-4o. This advantage is most pronounced in discourse-level errors (*CorefE* and *LinkE*), where even leading closed-source models often fail to identify any instances correctly. This performance gap underscores two key strengths of InFi-Checker: (1) Cross-frame analysis capability: Our novel hallucination construction strategies of discourse-level errors enhance the model's ability to handle hallucinations spanning across semantic frames; and (2) Structured reasoning: Our synthetic justification design effectively addresses the limitations of standard LLMs in complex hallucination analysis.

InFi-Check provides a significant performance boost to the backbones. By comparing InFi-Checker (InFiChecker-Llama and InFiChecker-Qwen) with their prompted backbone counterparts (Llama-3.1-8B-Instruct※ and Qwen3-8B※), we observe that the gains are not solely due to instruction-guided interpretable reasoning. Instead, the improvement primarily stems from the model's ability to internalize the nature of all types of hallucinations from our curated dataset, enabling robust and accurate fact-checking across different backbones.

5.3 OOD Generalization Results

We further evaluate the transferability of InFi-Checker across various benchmarks (Table 3 and Table 4), leading to the following conclusions:

The fine-grained fact-checking capabilities of InFi-Checker is generalizable. On the FRANK benchmark (Table 3), which shares the same error-type taxonomy but is derived from a different corpus, InFi-Checker consistently outperforms closed-source models and shows substantial improvements over its backbones. These results suggest that the fine-grained hallucination detection capabilities of InFi-Checker transcend simple pattern memorization. Instead, InFi-Checker captures transferable and fundamental diagnostic features, therefore generalizing effectively across out-of-distribution data.

Fine-grained training enhances binary fact-checking. Although InFi-Checker is optimized for fine-grained tasks, it achieves competitive results on six binary benchmarks (Table 4). These datasets cover a broad spectrum of document sources and various downstream tasks, including summarization, QA, and search-based generation, which comprehensively validates the model's robustness across varied contexts in binary mode. InFi-Checker surpasses binary classification specialized models and approaches or even exceeds the performance of closed-source models, suggesting that explicitly training on diverse error types heightens the model's overall sensitivity to hallucinations, thereby benefiting even binary classification.

Model	BAcc(Nor.)	BAcc(Str.)	SAR(Avg.)
The Open-Source Models			
Llama-3.1-8B-Instruct†	27.69	21.33	72.57
Llama-3.1-8B-Instruct※†	23.79	18.12	79.41
Qwen3-8B†	47.94	41.08	86.50
Qwen3-8B※†	50.49	44.78	87.22
The State-of-the-Art LLMs			
Claude-3.7-Sonnet※	49.19	45.12	94.28
DeepSeek-V3.2-NoThink※	59.19	52.31	87.33
GPT-4o※	64.69	60.27	80.19
GPT-4.1※	58.91	55.13	94.87
GPT-5※	46.35	42.28	92.58
o3※†	60.77	56.76	94.60
InFi-Checker (Ours)			
InFi-Checker-Llama	90.91	85.49	94.02
InFi-Checker-Qwen	92.34	87.74	94.92

Table 5: Normal(Nor.) and strict(Str.) balanced accuracy (%) and averaged SAR on InFi-Check-FG. The reported SAR(Avg.) is the average of error-type specific SAR across all evaluable error types. Models marked with † contain types with zero accuracy (excluded from the average), which may lead to an optimistic estimation of their SAR. ※ means one-shot settings. Full result with zero-shot settings are displayed in Appendix D.

Index	**J**ust.+Corr.	**E**vid.	**S**ent.	BAcc
JES	✓	✓	✓	90.91
ES	-	✓	✓	75.02
JS	✓	-	✓	69.85
JE	✓	✓	-	61.17
J	✓	-	-	36.95
E	-	✓	-	53.76
S	-	-	✓	30.02
raw	-	-	-	20.59

Table 6: Result (%) for ablation study on InFi-Checker-Llama on InFi-Check-FG. "Just.+Corr." means outputting error justification and correction, "Evid." means outputting evidence sentences, and "Sent." means analyzing sentence-by-sentence. See Appendix D for separate ablation of justification and correction.

Generalization across synthesis pipelines. To evaluate robustness against synthesis bias, we construct an additional evaluation set, **InFi-Check-DS**, using a generator model independent from InFi-Check-TR and InFi-Check-FG pipeline while preserving the same annotation protocol. Table 22 shows that InFi-Checker consistently outperforms frontier LLM baselines, suggesting that its gains extend beyond generator-specific patterns.

5.4 In-depth Sentence Level Analysis

To evaluate the models' interpretability, we assess performance at the sentence level by requiring models to not only categorize the fine-grained error type but also precisely localize it within the text. Specifically, we define **Strict Accuracy**, which demands: (1) correct error-type assignment, (2) precise labeling on both the hallucinated and hallucination-free claim sentences. We then introduce the **Sentence Alignment Ratio (SAR)**, defined as the ratio of strict accuracy to normal accuracy of each error type. While normal metrics only verify labels, SAR quantifies the consistency between a model's final judgment and its underlying reasoning. A higher SAR indicates authentic comprehension rather than coincidental guessing. As shown in Table 5, InFi-Checker ranks among the highest SAR scores, which is increasingly difficult as the base normal accuracy rises, therefore confirming InFi-Checker's performance gain is driven by robust and interpretable reasoning.

5.5 Further Analysis

Ablation Study We conduct ablation experiments to demonstrate the importance of the key components of InFi-Check output, including interpretable justifications and corrections (**J**), evidence sentence (**E**), as well as the sentence-by-sentence claim analysis (**S**). As shown in Table 6, removing any component leads to a notable performance drop, highlighting the importance of each component.

Additional Analysis More detailed analyses and discussions are provided in Appendix D, including **reliability verification** (confusion matrices, confidence intervals, output parse rate, and LLM-as-Judge evaluation of the quality of each component), **robustness** (result under different label granularities or binary-collapsed settings of fine-grained dataset), **qualitative review**, and **cost efficiency**.

6 Conclusion

In this paper, we propose a new framework termed InFi-Check, which integrates error typologies, synthetic data generation pipelines, and fine-grained annotations for comprehensive fact-checking. To be specific, we first develop diverse methods to synthesize six hallucination error types in grounded generation, including both intrinsic and extrinsic, semantic and discourse level errors. In this way, we construct InFi-Check-TR and InFi-Check-FG, two novel datasets characterized by fine-grained error type design, interpretable justifications and corrections, as well as comprehensive claims. Building upon InFi-Check-TR, we develop InFi-Checker, an advanced factuality evaluation model capable of fine-grained hallucination analysis. Also, we conduct extensive experiments to verify the superiority of InFi-Check-FG and InFi-Checker.

Limitations

Our pipeline's effectiveness is constrained by inherent limitations in LLM capabilities. While we employ sentence-level verification, the models still generate document-unsupported hallucinations, or fail in extracting full grounding evidence sentences. Additionally, they struggle to differentiate between factual incompleteness and legitimate information simplification, particularly affecting error construction quality for more complex cases. We adapt manual check and filtering for InFi-Check-FG to mitigate this bias in evaluation.

The validity of InFi-Checkers is also effected by the coverage of the hallucination typology. The typology we adopt is grounded in linguistic theory (notably Frame Semantics (Fillmore, 1976) and Linguistic Discourse Analysis (Brown and Yule, 1983), and is designed to systematically cover hallucinations across multiple linguistic levels, from semantic to discourse-level. Importantly, it also explicitly includes Out-of-Article (extrinsic) errors, which capture hallucinations that cannot be aligned to the grounding document. This typology has been empirically validated through manual analysis of $2,250$ model-generated summaries, successfully categorizing about 99% of hallucinations with high inter-annotator agreement, and our human evaluation further confirms its coverage on our data.

Nevertheless, rare or previously unseen hallucination patterns may still fall outside the taxonomy, potentially limiting the ability of InFi-Checkers to detect them, although such cases are expected to be infrequent. Moreover, as InFi-Check framework is designed to be modular and extensible, if some truly novel hallucination patterns emerge in the future, whether from reasoning-heavy models or new data distributions, our architecture allows for the seamless integration of new error types by simply updating the error type design in Stage 3 and the construction strategy in Stage 4. This ensures the framework remains a dynamic and future-proof tool for fine-grained factuality evaluation.

In addition, hallucinations may occasionally involve multiple error types within a single sentence. As such instances are typically dominated by one primary error, our framework addresses them through iterative correction, where the most salient error is fixed first and subsequent errors are identified in later passes. Future work could explore more natural formulations that explicitly model multiple error types simultaneously.

Ethics Statement

Our work focuses on improving the fact-checking abilities of document-grounded generation systems through interpretable and fine-grained methods. While synthetic errors and human annotations are central to our approach, we recognize potential risks, such as misuse for generating misinformation or unintended biases in the dataset. To mitigate these concerns, we ensure transparency in our methodologies and emphasize their use for research purposes only. Additionally, our dataset and model are designed to generalize across diverse scenarios, avoiding overfitting to specific benchmarks. All our contributions are released under research-focused licenses, fostering ethical use and encouraging the broader community to leverage our work to advance the reliability of AI systems.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2025. Claude 3.7 sonnet system card. Technical report, OpenAI.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Gillian Brown and George Yule. 1983. *Discourse Analysis*. Cambridge Textbooks in Linguistics. Cambridge University Press.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao,

Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2025a. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Erhang Li, Fangqi Zhou, Fangyun Lin, Fucong Dai, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Li, Haofen Liang, Haoran Wei, Haowei Zhang, Haowen Luo, Haozhe Ji, Honghui Ding, Hongxuan Tang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui Zeng, Jialiang Huang, Jiashi Li, Jiaxin Xu, Jiewen Hu, Jingchang Chen, Jingting Xiang, Jingyang Yuan, Jingyuan Cheng, Jinhua Zhu, Jun Ran, Junguang Jiang, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Kexin Huang, Kexing Zhou, Kezhao Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Wang, Liang Zhao, Liangsheng Yin, Lihua Guo, Lingxiao Luo, Linwang Ma, Litong Wang, Liyue Zhang, M. S. Di, M. Y Xu, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Panpan Huang, Peixin Cong, Peiyi Wang, Qiancheng Wang, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Ruiling Xu, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runqiu Yin, Runxin Xu, Ruomeng Shen, Ruoyu Zhang, S. H. Liu, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoyuan Chen, Shengding Hu, Shengyu Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, Songyang Zhou, Tao Ni, Tao Yun, Tian Pei, Tian Ye, Tianyuan Yue, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjun Gao, Wentao Zhang, Xi Gao, Xiangwen Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xingyou Li, Xinyu Yang, Xinyuan Li, Xu Chen, Xuecheng Su, Xuehai Pan, Xuheng Lin, Xuwei Fu, Y. Q. Wang, Yang Zhang, Yanhong Xu, Yanru Ma, Yao Li, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Qian, Yi Yu, Yichao Zhang, Yifan Ding, Yifan Shi, Yiliang Xiong, Ying He, Ying Zhou, Yinmin Zhong, Yishi Piao, Yisong Wang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyuan Liu, Yonglun Yang, Yongqiang Guo, Yongtong Wu, Yu Wu, Yuan Cheng, Yuan Ou, Yuanfan Xu, Yuduan Wang, Yue Gong, Yuhan Wu, Yuheng Zou, Yukun Li, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehua Zhao, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhixian Huang, Zhiyu Wu, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihui Gu, Zijia Zhu, Zilin Li, Zipeng Zhang, Ziwei Xie, Ziyi Gao, Zizheng Pan, Zongqing Yao, Bei Feng, Hui Li, J. L. Cai, Jiaqi Ni, Lei Xu, Meng Li, Ning Tian, R. J. Chen, R. L. Jin, S. S. Li, Shuang Zhou, Tianyu Sun, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xinnan Song, Xinyi Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, Dongjie Ji, Jian Liang, Jianzhong Guo, Jin Chen, Leyi Xia, Miaojun Wang, Mingming Li, Peng Zhang, Ruyi Chen, Shangmian Sun, Shaoqing Wu, Shengfeng Ye, T. Wang, W. L. Xiao, Wei An, Xianzu Wang, Xiaowen Sun, Xiaoxiang Wang, Ying Tang, Yukun Zha, Zekai Zhang, Zhe Ju, Zhen Zhang, and Zihua Qu. 2025b. Deepseek-v3.2: Pushing the frontier of open large language models. *Preprint*, arXiv:2512.02556.

Esin Durmus, He He, and Mona Diab. 2020. FEQA: A question answering evaluation framework for faithfulness assessment in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5055–5070, Online. Association for Computational Linguistics.

Alexander Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2022. QAFactEval: Improved QA-based factual consistency evaluation for summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2587–2601, Seattle, United States. Association for Computational Linguistics.

Charles J. Fillmore. 1976. Frame semantics and the na-

ture of language *. *Annals of the New York Academy of Sciences*, 280.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1449–1462, Online. Association for Computational Linguistics.

Anushka Gupta, Diksha Chugh, Anjum, and Rahul Katarya. 2022. Automated news summarization using transformers. In *Sustainable advanced computing: select proceedings of ICSAC 2021*, pages 249–259. Springer.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Alon Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor Geva. 2024. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains. *Preprint*, arXiv:2402.00559.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.

Paul Kassianik, Baturay Saglam, Alexander Chen, Blaine Nelson, Anu Vellore, Massimo Aufiero, Fraser Burch, Dhruv Kedia, Avi Zohary, Sajana Weerawardhena, Aman Priyanshu, Adam Swanda, Amy Chang, Hyrum Anderson, Kojin Oshiba, Omar Santos, Yaron Singer, and Amin Karbasi. 2025. Llama-3.1-foundationai-securityllm-base-8b technical report. *Preprint*, arXiv:2504.21039.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *Preprint*, arXiv:1910.12840.

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

Deren Lei, Yaxi Li, Siyao Li, Mengya Hu, Rui Xu, Ken Archer, Mingyu Wang, Emily Ching, and Alex Deng. 2025. FactCG: Enhancing fact checkers with graph-based multi-hop data. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5002–5020, Albuquerque, New Mexico. Association for Computational Linguistics.

Sheng-Chieh Lin, Luyu Gao, Barlas Oguz, Wenhan Xiong, Jimmy Lin, Wen tau Yih, and Xilun Chen. 2024. FLAME : Factuality-aware alignment for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Nelson Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7001–7025, Singapore. Association for Computational Linguistics.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2024. ExpertQA: Expert-curated questions and attributed answers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3025–3045, Mexico City, Mexico. 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.

Kushan Mitra, Dan Zhang, Sajjadur Rahman, and Estevam Hruschka. 2025. Factlens: Benchmarking fine-grained fact verification. *Preprint*, arXiv:2411.05980.

OpenAI. 2025a. Gpt-5 system card. Technical report, OpenAI. Accessed: 2026-01-05.

OpenAI. 2025b. Introducing gpt-4.1 in the api. `https://openai.com/index/gpt-4-1/`.

OpenAI. 2025c. Openai o3 and o4-mini system card. Technical report, OpenAI.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021a. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021b. Understanding factuality in abstractive summarization with frank: A benchmark for factuality metrics. *Preprint*, arXiv:2104.13346.

Leonardo F. R. Ribeiro, Mengwen Liu, Iryna Gurevych, Markus Dreyer, and Mohit Bansal. 2022. FactGraph: Evaluating factuality in summarization with semantic graph representations. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3238–3253, Seattle, United States. Association for Computational Linguistics.

Wooseok Seo, Seungju Han, Jaehun Jung, Benjamin Newman, Seungwon Lim, Seungbeen Lee, Ximing Lu, Yejin Choi, and Youngjae Yu. 2025. Verifying the verifiers: Unveiling pitfalls and potentials in fact verifiers. *arXiv preprint arXiv:2506.13342*.

ZhongXiang Sun, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. 2025. RedeEP: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability. In *The Thirteenth International Conference on Learning Representations*.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024a. Minicheck: Efficient fact-checking of llms on grounding documents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8818–8847.

Liyan Tang, Igor Shalyminov, Amy Wong, Jon Burnsky, Jake Vincent, Yu'an Yang, Siffi Singh, Song Feng, Hwanjun Song, Hang Su, Lijia Sun, Yi Zhang, Saab Mansour, and Kathleen McKeown. 2024b. TofuEval: Evaluating hallucinations of LLMs on topic-focused dialogue summarization. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4455–4480, Mexico City, Mexico. Association for Computational Linguistics.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5020, Online. Association for Computational Linguistics.

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. 2024. Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers. *Preprint*, arXiv:2311.09000.

Yumo Xu and Mirella Lapata. 2019. Weakly supervised domain detection. *Transactions of the Association for Computational Linguistics*, 7:581–596.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Samuel Yeh, Sharon Li, and Tanwi Mallick. 2026. LUMINA: Detecting hallucinations in RAG system with context–knowledge signals. In *The Fourteenth International Conference on Learning Representations*.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada. Association for Computational Linguistics.

Huajian Zhang, Yumo Xu, and Laura Perez-Beltrachini. 2024. Fine-grained natural language inference based faithfulness evaluation for diverse summarisation tasks. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1701–1722, St. Julian's, Malta. Association for Computational Linguistics.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Chen Xu, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2025. Siren's song in the ai ocean: A survey on hallucination in large language models. *Preprint*, arXiv:2309.01219.

Yiran Zhao, Jinghan Zhang, I Chern, Siyang Gao, Pengfei Liu, Junxian He, et al. 2023. Felm: Benchmarking factuality evaluation of large language models. *Advances in Neural Information Processing Systems*, 36:44502–44523.

Ádám Kovács and Gábor Recski. 2025. Lettucedetect: A hallucination detection framework for rag applications. *Preprint*, arXiv:2502.17125.

Appendix

A InFi-Check-FG Construction Details

A.1 Detailed Hallucination Categorization and Error Construction Examples

We provide a full hallucination categorization along with illustrative examples for each error construction methodology in Table 29.

A.2 Grounded Claim Dataset Construction

Our pipeline starts with an arbitrary document dataset, which can include pre-existing supported claims or simply the documents themselves. Due to the limitations of existing document-claim datasets in terms of length (news datasets being relatively short, and academic paper dataset being excessively long), combined with challenges that some automatically extracted claims lack complete support from the document (such as the TL;DR dataset (Völske et al., 2017)), we opted to construct our dataset beginning with the document itself, which also streamlines the extraction of evidence and proves the broad usability of our method.

Specifically, for each document in the dataset, we prompt LLMs to generate a series of document-grounded claims that capture the core factual content. In our implementation, we leverage a summarization-style objective for this generation process. This approach is chosen because such condensed representations naturally require every generated claim to be strictly grounded in the source context, providing an ideal foundation for fact-checking data. Section 5.2 demonstrates that our pipeline and model possess strong generalizability to other document-grounded generation tasks.

Subsequently, we apply an extract-and-rewrite process to ensure the factuality of each claim sentence based on the grounding document: 1) LLMs are prompted to locate the grounding sentences as evidence from the source document for each sentence in these claims. 2) A voting mechanism involving three LLMs is applied to determine whether a claim sentence is sufficiently supported by its grounding sentences. If a sentence lacks adequate support, it undergoes a rewriting process, and the voting process repeats until complete support from the reference is achieved. The aforementioned process also serves as the preparation for the synthetic model output. Ultimately, we achieve a set of claims, each with sentence-level grounding evidence. Importantly, due to the flexibility of the base dataset, our pipeline can be applied to any document dataset and any data distribution, supporting the scalability of InFi-Check.

A.3 Document Dataset Selection

We adopt the BBC News Summary Dataset (Gupta et al., 2022) and the DetNet Wikipedia Dataset (Xu and Lapata, 2019) as the initial document datasets. These datasets offer a diverse range of documents covering news and encyclopedic content, classified by domain, with a varied length distribution. We ensured diversity by choosing documents from different domains and lengths.

The BBC News Summary Dataset (Gupta et al., 2022) consists of extractive summaries only, so we did not use it as the original claim. This data set categorizes news articles into five distinct categories: business, entertainment, politics, sports, and technology. The DetNet Wikipedia Dataset (Xu and Lapata, 2019) is designed for domain detection, with Wikipedia data labeled for seven domains: "Business and Commerce" (BUS), "Government and Politics" (GOV), "Physical and Mental Health" (HEA), "Law and Order" (LAW), "Lifestyle" (LIF), "Military" (MIL), and "General Purpose" (GEN). For the BBC News Summary Dataset, we randomly selected 150 documents from each category. For the DetNet Wikipedia Dataset, we select 100 documents from each domain. The document length in both datasets varies greatly. To ensure the models have a certain degree of robustness, but also efficiency while training, we filtered documents to have lengths within the range of 300 to 1000 words, and prompted the language models (LLMs) to control the claim length within the range of $[100, \min(doc_len/3 + 10, 200)]$ words.

A.4 LLM Selection in InFi-Check Pipeline

To address potential biases where a single model might favor its own generated text, and to avoid issues where training exclusively with one model's outputs might cause out-of-domain problems for texts generated by other models, we utilized two different sets of LLMs at each step of our pipeline. In addition, the models used for generation and evaluation were different. For rewriting, we use the strongest model in the support determination model set. Usage of LLMs is outlined in Table 7.

A.5 Curation for InFi-Check-FG

Based on the results of human evaluation, we design a set of strict filtering prompts to further curate

Task	LLM Set 1	LLM Set 2
Summarization	GPT-4o	DeepSeek-R1
Reference extraction	Claude-3.7-Sonnet	GPT-4o
Support determination	(Claude-3.7-Sonnet, Qwen-2.5, Gemini-1.5)	(GPT-4o, Qwen-2.5, Gemini-2.0)
Rewriting	Claude-3.7-Sonnet	GPT-4o
Error data construction	GPT-4o	Claude-3.7-Sonnet

Table 7: Usage of LLMs in dataset construction

	#doc	#doc-pair	#PredE	#EntE	#CircE	#CorefE	#LinkE	#OutE	#NoE
InFi-Check-TR	1263	15660	2946	1998	1386	2889	1495	1457	3489
InFi-Check-FG	190	519	85	21	29	64	30	38	252

Table 8: Statistics of the dataset constructed from InFi-Check

the validation and test sets. For each instance, we employ GPT-4.1 to independently check the presence of any potential generation errors categorized in human evaluation, including but not limited to incorrect extraction of grounding sentences and misclassification of hallucination or error types. Each possible error is examined in isolation through dedicated prompts, and an instance is retained only if it is verified to be fully correct across all checks. See Appendix F for filtering prompts. We intentionally design this filtering procedure to be conservative, prioritizing precision over recall. While it may discard some instances that are in fact correct, it substantially reduces the risk of retaining flawed samples. Empirically, this process is able to identify all of the errors observed in human evaluation (flawed summary, false negative error, wrong error type, wrong error reasoning). We apply this filtering pipeline to both the validation and test sets, resulting in a final curated set of 519 instances that constitute InFi-Check-FG.

In this step, the role of GPT-4.1 is a quality control filter that imitates human judgment, and its role is to exclude the noisy data that does not comply with human judgment in the synthesized data, rather than just keeping some easy-to-classify data. To further validate this point, we have conducted an additional human evaluation directly on the InFi-Check-FG benchmark. Specifically, we sampled 140 instances (20 per error type plus 20 "No Error" instances) and instructed three human annotators to independently verify the correctness of the labels and claims (using the same rubric GPT-4.1 adopted). The results show that only 2 out of 140 instances are labeled problematic, both from the "No Error" (NoE) category, where subtle issues (one Entity Error and one Circumstance Error) were overlooked. Importantly, all hallucinated instances (i.e., all error types) were confirmed to be correct by human annotators. These findings support two key conclusions: (1) The GPT-4.1 filtering step effectively preserves instances that align with human judgment, particularly for hallucination cases, which are the primary focus of InFi-Check-FG. (2) While GPT-4.1 may not perfectly guarantee strict correctness in NoE cases, this does not introduce a systematic bias favoring our model. In fact, as shown in Table 2, the performance gap of InFi-Checker on the NoE category is relatively small compared to other error types, suggesting that its overall advantage does not stem from exploiting such artifacts, but from genuinely improved fine-grained hallucination analysis.

B Statistics of InFi-Check Datasets

The overall statistics of the dataset constructed using InFi-Check pipeline is shown in Table 8, in which InFi-Check-TR is the train set of InFi-Checker, and InFi-Check-FG is the test set based on manual filtering. The two datasets are derived from different document sets to avoid data leakage.

C Implementation Details

C.1 Baseline Details

ClearCheck (Seo et al., 2025) is fine-tuned from Llama-3.1-8B-Instruct, it improves verification robustness through multi-task training tailored for hallucination detection across various grounded generation scenarios. **AlignScore** (Zha et al., 2023) is a holistic metric that evaluates factual consistency through a unified alignment function trained across diverse NLP tasks such as NLI and QA.

Model	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE	BAcc
The Open-Source Models								
Llama-3.1-8B	43.75	31.49	10.34	4.06	0.00	7.21	77.34	24.88
Qwen-3-8B	58.75	37.45	12.64	12.90	0.00	9.03	65.30	28.01
The State-of-the-Art LLMs								
DeepSeek-V3.2	52.69	43.34	22.02	6.82	7.41	18.02	78.88	32.74
GPT-4o	59.14	49.85	24.77	5.68	0.00	14.24	69.11	31.83
o3	53.76	47.37	12.84	3.41	3.70	13.66	73.65	29.77
InFi-Checker (Ours)								
InFi-Checker-Llama	28.12	62.13	34.48	20.97	14.50	9.47	78.95	35.52
InFi-Checker.Qwen	42.19	48.09	49.43	24.19	16.67	10.50	80.12	38.74

Table 9: Per-category fine-grained accuracy on FRANK under any-match protocol.

We adopt the largest and best-performing version (AlignScore-large) as our baseline. **FactCG** (Lei et al., 2025) enhances model performance by generating complex training data via multi-hop reasoning on context graphs extracted from documents. We use the best-performing version (FactCG-DBT) as our baseline. **MiniCheck** (Tang et al., 2024a) achieves state-of-the-art binary performance by utilizing a novel document-claim pair synthesis method to train lightweight fact-checkers. We use the largest and best-performing version (Bespoke-MiniCheck-7B) as our baseline.

C.2 Benchmark and Metric Details

For InFi-Check-FG, we calculate the accuracy of each error type (as well as "No Error"), and report their average as the **balanced accuracy**, where each error type contributes equally regardless of the number of samples. The "accuracy" is defined as whether the model correctly points out the specific error type in samples containing this type of error. If a model reports a sample to be hallucinated, but with the wrong error type, it is judged as incorrect.

The **FRANK** benchmark focuses on factual consistency in abstractive summarization using the CNN/DM and XSum news datasets. It provides a manual annotated fine-grained typology of errors found in the outputs of multiple state-of-the-art summarization models, offering a rigorous testbed for news-domain grounding. To facilitate a standardized comparison across diverse benchmarks, we also report balanced accuracy on the FRANK dataset instead of correlation-based metrics, which often suffer from limited comparability across different model scales. Given the complexity of FRANK—where each instance is annotated by multiple experts and often contains overlapping error types—we adopt a multi-annotator consensus criterion for evaluation. Specifically, a model's prediction is considered correct if the predicted error type aligns with any of the labels assigned by the human annotators for that specific instance. This approach accounts for the inherent subjectivity and legitimate label diversity in fine-grained hallucination detection. Furthermore, while the FRANK dataset contains fine-grained error categories, many instances exhibit multiple co-occurring error types, which can introduce noise into categorical classification. To ensure a consistent and robust evaluation, we reformulate the task into a binary consistency check for metric calculation. We map all fine-grained error types to a "hallucinated" category, and treat "No Error" as "factual". To mitigate the impact of class imbalance within the dataset, we report balanced accuracy, providing a more reliable reflection of the model's discriminative performance across both factual and non-factual claims.

We also provide a per-category fine-grained accuracy on FRANK just for relative comparison in Table 9. Specifically, we retain all labels obtained via annotator majority voting. For instances where no single label achieves a majority vote, but the majority of annotators agree that the instance is hallucinated, we exclude them for clarity. We then treat each (instance, label) pair as an independent evaluation unit and check whether the model predicts that label. We note that this evaluation has *inherent limitations*: due to the presence of multi-label instances, while models are prone to output a single dominant error type, they may correctly identify one valid error but not necessarily all co-occurring ones. As a result, this metric *underestimates* model performance in multi-error scenarios. Nevertheless, from Table 9, we can observe that InFi-Checker still achieves the best overall performance (BAcc) under the per-category evaluation, outperforming all baselines. Notably, the gains are most pronounced on more complex error types such as Co-reference Error and Discourse Link Error, which require stronger cross-sentence reasoning.

Model	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
The State-of-the-Art LLMs							
GPT-4o ❋	84.71 [75.57, 90.84]	61.90 [40.88, 79.25]	44.83 [28.41, 62.45]	75.00 [63.18, 83.99]	0.00 [0.00, 11.35]	94.74 [82.71, 98.54]	91.67 [87.60, 94.49]
DeepSeek-V3.2-NoThink❋	75.29 [65.17, 83.24]	71.43 [50.04, 86.19]	55.17 [37.55, 71.59]	15.62 [8.71, 26.43]	13.33 [5.31, 29.68]	89.47 [75.87, 95.83]	94.05 [90.41, 96.36]
o3❋	81.18 [71.59, 88.07]	57.14 [36.55, 75.53]	58.62 [40.74, 74.49]	56.25 [44.09, 67.71]	3.33 [0.59, 16.67]	81.58 [66.58, 90.78]	87.30 [82.62, 90.86]
InFi-Checker (Ours)							
InFi-Checker-Llama	96.43 [90.02, 98.78]	90.48 [71.09, 97.35]	79.31 [61.61, 90.15]	95.31 [87.10, 98.39]	86.67 [70.32, 94.69]	100.00 [90.82, 100.00]	90.00 [85.65, 93.13]
InFi-Checker-Qwen	92.94 [85.44, 96.72]	90.48 [71.09, 97.35]	86.21 [69.44, 94.50]	100.00 [94.34, 100.00]	90.00 [74.38, 96.54]	97.37 [86.51, 99.53]	88.10 [83.51, 91.53]

Table 10: The 95% Confidence Intervals (CIs) for all per-category accuracy metrics using the Wilson Score Interval.

gold\pred	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
PredE	81	0	0	0	2	0	1
EntE	0	19	1	0	0	0	1
CircE	1	4	23	0	1	0	0
CorefE	0	2	0	61	1	0	0
LinkE	3	0	0	0	26	0	1
OutE	0	0	0	0	0	38	0
NoE	4	4	1	13	3	0	225

Table 11: Confusion Matrix of InFi-Checker-Llama

gold\pred	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
PredE	79	1	2	1	2	0	0
EntE	0	19	1	0	0	0	1
CircE	1	3	25	0	0	0	0
CorefE	0	0	0	64	0	0	0
LinkE	2	0	0	0	27	0	1
OutE	0	1	0	0	0	37	0
NoE	4	13	2	9	2	0	222

Table 12: Confusion Matrix of InFi-Checker-Qwen

gold\pred	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
PredE	72	0	6	0	0	4	3
EntE	3	13	0	0	0	1	3
CircE	12	2	13	0	0	1	0
CorefE	3	5	2	48	0	1	4
LinkE	4	0	6	0	0	2	18
OutE	0	1	1	0	0	36	0
NoE	8	2	5	1	1	3	231

Table 13: Confusion Matrix of GPT-4o (1-shot)

gold\pred	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
PredE	64	0	8	0	0	4	5
EntE	0	15	0	0	1	1	3
CircE	1	11	16	0	0	0	1
CorefE	1	38	0	10	0	1	11
LinkE	3	1	6	0	4	0	15
OutE	1	2	0	0	0	34	0
NoE	2	4	4	0	0	2	237

Table 14: Confusion Matrix of DeepSeek-V3.2 (1-shot)

For our evaluation on binary fact-checking benchmarks, we acknowledge the data quality issues identified by Seo et al. (2025), which notes that several widely used datasets contain significant annotation noise, linguistic ambiguities, and skewed label distributions. To ensure a robust and reliable assessment, we adopt the filtered and reprocessed versions of these benchmarks from Seo et al. (2025). To maintain strict parity with established baselines and facilitate a direct comparison with state-of-the-art methods, we follow the evaluation protocol of Seo et al. (2025) and report the Macro F1 score. The six binary fact-checking benchmarks are: **ClaimVerify** audits the generative search engine task, using responses from commercial systems (e.g., Bing Chat) across diverse queries from Google history and Reddit. It emphasizes the accuracy of in- line citations and the verifiability of claims against real-world web sources. **ExpertQA** targets high-stakes, long-form question answering across 32 specialized fields (e.g., medicine, law). It features expert-curated questions and claims generated by representative LLMs, requiring models to verify professional, domain-specific knowledge with high precision. **Factcheck-Bench** is a comprehensive benchmark for open-domain document-level factuality, evaluating LLM-generated responses at the claim, sentence, and document levels. It encompasses a wide array of general-purpose topics, aiming to test the end-to-end verification capabilities of automatic systems. **REVEAL** focuses on complex Chain-of-Thought (CoT) reasoning in open-domain settings. It provides step-level labels for attribution and logical correctness, testing whether a model can verify the intermediate reasoning claims of a language model's answer. **MediaSum** and **MeetingBank** are subset from the summarization dataset ToFuEval. MediaSum centers on topic-focused dialogue summarization within the media interview domain. It challenges models to maintain factual integrity when distilling multi-party conversations into concise, grounded summaries. MeetingBank focuses on professional meeting transcripts and evaluates the consistency of summaries generated from lengthy, informal, and multi-speaker interactions. It serves as a robust test for grounding claims in complex, non-structured dialogue data.

C.3 Parameter Details of the experiment

Training Hyperparameter of InFi-Checker We fine-tune both Llama-3.1-8B-Instruct and

gold\pred	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE
PredE	69	0	8	0	0	3	1
EntE	2	12	0	1	0	0	5
CircE	9	2	17	0	0	0	1
CorefE	1	17	2	36	0	1	1
LinkE	4	0	11	0	1	1	11
OutE	0	1	1	0	0	31	0
NoE	12	5	8	0	0	5	220

Table 15: Confusion Matrix of o3 (1-shot)

Model	BAcc(Nor.)	BAcc(Str.)	SAR(Avg.)
The Open-Source Models			
Llama-3.1-8B-Instruct†	27.69	21.33	72.57
Llama-3.1-8B-Instruct※†	23.79	18.12	79.41
Qwen3-8B†	47.94	41.08	86.50
Qwen3-8B※†	50.49	44.78	87.22
The State-of-the-Art LLMs			
Claude-3.7-Sonnet	53.07	48.48	92.21
Claude-3.7-Sonnet※	49.19	45.12	94.28
DeepSeek-V3.2-NoThink	59.64	53.94	89.43
DeepSeek-V3.2-NoThink※	59.19	52.31	87.33
GPT-4o	60.50	56.57	94.97
GPT-4o※	64.69	60.27	80.19
GPT-4.1	54.45	49.44	78.17
GPT-4.1※	58.91	55.13	94.87
GPT-5	49.06	44.41	92.05
GPT-5※	46.35	42.28	92.58
o3†	58.00	55.07	95.48
o3※†	60.77	56.76	94.60
InFi-Checker (Ours)			
InFi-Checker-Llama	90.91	85.49	94.02
InFi-Checker-Qwen	92.34	87.74	94.92

Table 16: Normal(Nor.) and strict(Str.) balanced accuracy (%) and averaged SAR on InFi-Check-FG. The reported SAR (Avg.) is the average of error-type-specific SAR across all evaluable error types. Models marked with † contain types with zero accuracy (excluded from the average), which may lead to an optimistic estimation of their SAR. ※ means one-shot settings.

Qwen3-8B for 3 epochs on InFi-Check-TR, using a batch size of 32 and the Adam optimizer. The learning rate follows a cosine-decay from $1e-5$ to $1e-6$, and we set the warm-up fraction to 0.1. We trained InFi-Checker-Qwen with standard generation ("no-think" setting) to follow the uniform structure of InFi-Check, which includes reasoning in the structured output rather than the <think> area.

Inference Hyperparameter To enforce deterministic generation and ensure reproducibility across runs, we adopt deterministic decoding settings whenever possible. For open-source LLMs and InFi-Checkers, we set num_beams $= 1$ and disabled sampling (do_sample=False), which corresponds to temperature $= 0$. For closed-source LLMs, we set temperature $= 0$ if possible, otherwise following the required settings of the API.

Setting	BAcc
Justification + Correction	90.91
Justification only	86.30
Correction only	83.17
No interpretable reasoning	75.02

Table 17: Ablation study of InFi-Checker-Llama on InFi-Check-FG for assessing justification and correction separately. Other component, such as evidence and sentence-by-sentence reasoning patterns, is preserved throughout the experiment.

Model	BAcc
Llama-3.1-8B-Instruct	71.5
o3	74.6
InFi-Checker-Llama-Ablation(J)	73.1
InFi-Checker-Llama	77.2
InFi-Checker-Qwen	77.2

Table 18: Comparison of Ablation(J) with other models on FRANK. Ablation(J) achieves only a marginal gain compared to the vanilla Llama model.

D Further Results and Analysis

D.1 Reliability Verification

Confidence Intervals Since the number of instances varies across fine-grained error categories, raw per-category accuracy may not fully reflect the reliability of model performance, particularly for categories with relatively few evaluation samples. To account for this uncertainty, we report 95% Confidence Intervals (CIs) for all per-category accuracy metrics using the Wilson Score Interval, which provides more stable interval estimates than normal approximation under limited sample sizes.

As shown in Table 10, InFi-Checker consistently achieves strong performance across all error categories while maintaining relatively narrow confidence intervals, indicating that the observed gains are not driven by random fluctuations or category imbalance. Importantly, the intervals remain substantially above those of the strongest baselines for most error types, suggesting that the improvements are statistically robust rather than resulting from a small number of favorable examples. We also observe that categories involving discourse-level reasoning exhibit wider intervals across all models due to their intrinsic difficulty and smaller sample sizes. Nevertheless, InFi-Checker maintains clear advantages under these more uncertain settings. Overall, these results support that the improvements of InFi-Checker remain reliable even after accounting for estimation uncertainty.

Model	Claim Verify	Expert QA	Factcheck-Bench	REVEAL	MediaSum	MeetingBank	Average
The Open-Source Models							
Llama-3.1-8B-Instruct	63.6	49.8	69.8	78.2	50.8	62.3	62.4
The State-of-the-Art LLMs							
GPT-4.1	81.6	**80.3**	**91.3**	93.2	75.9	**86.3**	84.8
GPT-5	87.7	75.9	90.4	**93.7**	80.2	87.6	**85.9**
o3	83.3	79.6	86.9	92.3	82.9	83.8	84.8
Specialized Fact-Checking Models							
AlignScore-large	79.8	75.0	83.7	92.2	75.8	76.5	80.5
MiniCheck	85.6	72.9	86.8	91.0	74.3	77.8	81.4
InFi-Checker-Llama-Ablation(J)	68.5	55.5	71.7	75.8	56.9	58.3	64.5
InFi-Checker-Llama	75.9	78.3	83.7	87.7	73.5	65.8	77.5
InFi-Checker-Qwen	89.6	75.7	88.0	90.0	80.4	78.5	83.7

Table 19: Comparison of Ablation(J) with other models on binary datasets.

Model	InFi-Check-FG	FRANK	Claim Verify	Expert QA	Factcheck-Bench	REVEAL	MediaSum	MeetingBank
Qwen3-8B (prompted binary)	51.2	72.3	67.4	60.1	74.8	86.2	76.9	76.1
Qwen3-8B (1-shot of InFi-Check reasoning)	46.8	71.0	66.1	55.6	78.4	83.2	77.7	74.2
InFi-Check-Qwen-Binary	82.7	72.0	70.9	64.2	69.6	76.0	74.9	63.5
InFi-Check-Qwen	92.3	80.8	89.6	75.7	88.0	90.0	80.4	78.5

Table 20: Comparison between different settings of prompting Qwen3-8B and training InFi-Check-Qwen. "Prompted binary" means the model is prompted to do binary classification. The performance on InFi-Check-FG and FRANK is calculated on the binary-collapsed setting following Appendix D.6.

Confusion Matrices To further address the reliability of InFi-Checkers against possible randomness, we also provide the confusion matrices for InFi-Checkers and the baselines from Table 11 to Table 15. From the matrices, we can draw the following conclusion: (1) InFi-Checker maintains extremely high diagonal scores across all error categories, indicating precise classification when an error is present. (2) When baseline models fail to identify an error, their predictions do not follow a consistent distribution. They do not systematically misclassify one specific error as another; rather, their errors are scattered (e.g., when the correct answer is Error A, some baseline predict Error B, while others predict Error C, or, at a rather high possibility, No Error). This suggests that the higher NoE performance of baselines may not stem from superior understanding, but rather from a lack of sensitivity. Their failure to recognize complex hallucinations often results in a default NoE prediction, whereas InFi-Checker's failures are more often nuanced misclassifications of the error type itself.

D.2 Additional Results for SAR and Ablation

SAR Analysis Table 16 shows the full strict accuracy and SAR result on the FRANK benchmark, including both zero-shot and one-shot results.

Ablation for Reasoning Component As shown in Table 17, we further decompose the "interpretable justification and correction" reasoning part into justification and correction, respectively. While retaining either component still yields competitive results, the integration of both processes is essential for achieving optimal performance.

To further demonstrate that structured reasoning is a fundamental capability, we evaluated the justification-only model (denoted as Ablation(J)) on the OOD datasets. As shown in Table 18 and Table 19, despite being exposed to the reasoning distribution during training, the InFi-Checker-Llama-Ablation(J) model showed only marginal improvements over the vanilla Llama backbone and even performance degradation on certain datasets. This reinforces our finding that structured sentence-by-sentence analysis and explicit evidence handling (which showed a dominant role in our ablation study) are the primary drivers of InFi-Checker's superior performance and generalization.

Ablation for Fine-Grained Taxonomy To explicitly disentangle the contribution of fine-grained supervision from that of training on synthesized data, we conduct an additional ablation by training a binary variant of our model, InFi-Check-Qwen-Binary, using the same InFi-Check-TR corpus but collapsing all labels into "No Error" vs. "Halluci-

Model	Parse Rate (↑)	Evidence Error Rate (↓)	Correction Error Rate (↓)
The Open-Source Models			
Llama3.1-8B-Instruct※	24.09	23.53	17.65
Qwen3-8B※	87.57	13.13	1.01
The State-of-the-Art LLMs			
DeepSeek-V3.2-NoThink※	93.21	5.26	0.75
GPT-4o※	86.96	5.09	2.36
o3※	97.54	7.50	0.62
InFi-Checker (Ours)			
InFi-Checker-Llama	92.91	4.87	1.77
InFi-Checker-Qwen	94.61	4.73	0.93

Table 21: Structured generation robustness and component-level consistency evaluation. We report parse success rate and LLM-as-Judge error rates for evidence extraction and hallucination correction on strictly correct predictions.

Model	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE	BAcc
The Open-Source Models								
Llama-3.1-8B-Instruct※	29.41	0.00	0.00	20.31	0.00	0.00	92.06	20.25
Qwen3-8B※	92.94	14.29	0.00	45.31	0.00	28.95	97.62	39.87
The State-of-the-Art LLMs								
GPT-4o※	57.65	33.33	6.90	26.56	0.00	89.47	95.63	44.22
o3※	64.71	19.05	68.97	20.31	3.33	55.26	85.32	45.28
InFi-Checker (Ours)								
InFi-Checker-Llama	91.76	38.10	86.21	89.06	80.00	89.47	90.48	80.73
InFi-Checker-Qwen	90.59	71.43	62.07	90.63	76.67	86.84	92.46	81.53

Table 22: Results (%) on InFi-Check-DS. BAcc serves for the balanced accuracy.

nated" following Appendix D.6. For a fair comparison, we further include a strong non-trained baseline, Qwen3-8B (prompted binary), where the backbone model is directly prompted to perform binary classification without any SFT. From the results shown in Table 20, we can draw the following conclusion about the difference between fine-grained and binary settings:

(1) Fine-grained supervision provides substantial gains beyond binary training. Even when evaluated under binary settings, the full InFi-Check-Qwen consistently outperforms InFi-Check-Qwen-Binary by a large margin on both InFi-Check-FG and all OOD benchmarks. This demonstrates that the improvements of our framework cannot be attributed solely to training on synthesized data. Instead, the fine-grained error taxonomy, together with structured outputs (evidence, justification, and correction) and sentence-level reasoning, contributes critically to performance. In other words, collapsing supervision to binary labels leads to a clear degradation, confirming that the "fine-grained" aspect of our pipeline is essential.

(2) Binary SFT on synthesized data alone does not yield robust, transferable factuality judgments. While InFi-Check-Qwen-Binary shows a substantial improvement over Qwen3-8B (prompted binary) on the in-domain InFi-Check-FG benchmark, its gains on OOD datasets are limited and inconsistent. Specifically, it achieves slight improvements on Claim Verify and Expert QA, but degrades on the majority of other benchmarks. This indicates that training on synthesized data with only binary supervision is insufficient for learning a stable and generalizable notion of hallucination. In contrast, InFi-Check-Qwen, which is trained with fine-grained labels and structured reasoning, exhibits strong and consistent generalization, suggesting that it learns transferable diagnostic features of hallucinations, which can then be effectively mapped to binary decisions.

(3) Prompting on InFi-Check-structured reasoning alone is insufficient to elicit transferable hallucination analysis. We also observe that Qwen3-8B (1-shot of InFi-Check reasoning) does not consistently outperform Qwen3-8B (prompted binary), and in some cases performs worse. This suggests that simply providing a structured reasoning format via prompting is not enough for the model to internalize fine-grained hallucination analysis. The ability to perform long-form, structured, and sentence-level reasoning over hallucinations does not emerge reliably from few-shot demonstrations alone; rather, it requires supervised training to be

Model	Coarse-grained Four-Category Classification					Binary Classification		
	Semantic	Discourse	Extrinsic	No error	BAcc	Hallucinated	No Error	BAcc
The Open-Source Models								
Llama3.1-8B-Instruct※	14.81	8.51	0.00	88.49	15.95	18.35	88.49	29.41
Qwen3-8B※	42.22	13.83	50.00	80.16	32.06	39.70	80.16	41.48
The State-of-the-Art LLMs								
DeepSeek-V3.2-NoThink※	85.19	14.89	89.47	94.05	70.90	83.90	94.05	88.97
GPT-4o※	89.63	51.06	94.74	91.67	81.77	88.39	91.67	90.03
o3※	88.15	39.36	81.58	87.30	74.10	87.27	87.30	87.28
InFi-Checker (Ours)								
InFi-Checker-Llama	97.78	91.49	100.00	89.29	93.32	98.13	89.29	93.71
InFi-Checker-Qwen	97.04	93.62	96.77	88.01	**94.98**	98.88	88.01	**94.08**

Table 23: Results on InFi-Check-FG with coarser label granularity, with highest balanced accuracy marked **bold**.

Model	InFi-Check-FG	FRANK
The State-of-the-Art LLMs		
DeepSeek-V3.2※	59.19	78.16
GPT-4o※	64.69	76.69
o3※	60.77	75.95
Specialized Binary Models		
MiniCheck-7B	68.39	70.79
FactCG	87.60	66.65
ClearCheck	81.42	70.22
InFi-Checker (Ours)		
InFi-Checker-Llama	90.91	78.88
InFi-Checker-Qwen	92.34	80.83

Table 24: Result of binary-specialized fact-checking models on InFi-Check-FG and FRANK with binary-collapsed setting. Performances are reported by balanced accuracy of "No Error" and "Hallucinated" (%).

Model	Cost($)
The State-of-the-Art LLMs	
GPT-5	23.9
Claude-3.7-Sonnet	12.7
o3	12.4
InFi-Checker (Ours)	
InFi-Checker-Llama	3.9
InFi-Checker-Qwen	4.4

Table 25: Comparison of costs of InFi-Checker and LLMs on InFi-Check-FG. InFi-Checker cost is calculated with $0.8 per GPU hour.

fully activated. This further supports our design choice of training with structured, fine-grained annotations instead of relying solely on prompting.

D.3 Validity of Complicated Structured Generation

Due to the relatively complex structured response format of InFi-Check, we conduct additional analyses to verify that InFi-Checkers can reliably follow the required structure while producing correct outputs. We evaluate the validity of generation through two parts, with the results shown in Table 21:

(1) Parse success and truncation. We first measure the structured-output parsing success rate across models. Overall, InFi-Checkers achieve parse rates comparable to strong closed-source models, suggesting that an 8B model is sufficient to learn stable structured generation. For the unparsed outputs of InFi-Checker-Qwen and InFi-Checker-Llama, 46.43% and 40.47% of failures are caused by output truncation, which corresponds to only 3.28% and 2.50% of the full evaluation set, respectively. This issue stems mainly from our limited training resources, where both training and inference were conducted with a 4096-token context limit. In practical deployments, this limitation can be easily alleviated by extending context windows.

(2) Quality of components We assess the quality of **evidence** and **correction** in InFi-Check reasoning using an LLM-as-Judge setting. Specifically, we employ Gemini-3-pro-preview (not included in our baselines, and showing compatible performance on InFi-Check-FG) to assess the strictly correct cases of each model, where the model correctly predicts both the correct error type and the position of the erroneous sentence. As shown in Table 21, both InFi-Checkers and closed-source models generally identify relevant evidence and produce reasonable corrections. However, InFi-Checkers exhibit consistently lower error rates, while also being evaluated on *a much larger number* of strictly correct instances. This suggests that InFi-Checkers are not only capable of generate structured outputs, but also demonstrate a stronger understanding and sensitivity of hallucinations, making them well-suited to serve as reliable critics in hallucination detection and correction based on a grounding document.

Model	PredE	EntE	CircE	CorefE	LinkE	OutE	NoE	BAcc
LongFormer-base-4096	2.35	61.90	0.00	81.25	10.00	23.68	0.40	25.65

Table 26: Result of LongFormer-base-4096 on InFi-Check-TR.

Label	Proportion(%)
No Problem	78
Flaws in Claim	11
Incomplete Grounding	6
False Negative Error	3
Wrong Error Type	1
Wrong Error Justification	1

Table 27: Human evaluation results on a sample of 100 instances from our dataset.

D.4 Robustness to Synthesis Pipeline

As InFi-Check-FGand InFi-Check-TRare constructed by the same set of LLMs (although in completely different data splits), to fully validate InFi-Checkers' generalization across completely different error distributions, we constructed a new evaluation benchmark, **InFi-Check-DS**. To ensure a distinct distributional shift from the frontier closed-source models used in our original pipeline, we employ an open-source-model for this evaluation set. As smaller open-source models (e.g., Ministral-3-8B and Qwen3-8B) struggled to maintain the high-quality structured generation required for fine-grained error injection in our preliminary tests, we employ DeepSeek-V3.2 (a leading open-source Mixture-of-Experts model with 671B total parameters and 37B activated parameters) as the error generator. DeepSeek-V3.2 provides a high-fidelity yet fundamentally different generation distribution compared to the GPT or Claude families. While maintaining the same nine construction strategies, the "error data construction" step (the most sensitive step identified by the reviewer) was performed exclusively by DeepSeek-V3.2.

D.5 Robustness to Label Granularity

Performance comparisons may potentially be affected by taxonomy ambiguity or guideline mismatch, where multiple labels could be considered plausible for a given instance. In such cases, accuracy may partially reflect adherence to the annotation taxonomy rather than the underlying ability to verify hallucinations. To examine the robustness of our findings to label granularity, we perform additional evaluations on InFi-Check-FG under coarser label groupings. Specifically, we evaluate under two alternative evaluation schemes: **(1) Four-category classification**, where both gold labels and model predictions are mapped to Semantic (PredE, EntE, and CircE), Discourse (CorefE and LinkE), Extrinsic, and No Error categories. **(2) Binary classification**, where labels are collapsed into Hallucinated vs. No Error (which is further discussed in Appendix D.6).

The results are reported in Table 23. From the results, we can draw the conclusion that InFi-Checkers maintain a clear performance advantage across both evaluation schemes. Notably, even under the more permissive four-category setting, closed-source models still exhibit low accuracy, particularly for discourse-level errors. From a linguistic perspective, discourse-level and semantic-level hallucinations are generally not considered inherently ambiguous categories. These results therefore, suggest that the observed performance gap cannot be fully attributed to fuzzy label boundaries, but instead reflects a substantial difficulty in detecting discourse-level hallucinations.

D.6 Robustness to Binary-Collapsed Setting

To address the absence of specialized trained fact-checking baselines in fine-grained datasets, we test MiniCheck, FactCG, and ClearCheck on both InFi-Check-FG [1] and FRANK. Since these models are designed for binary factuality prediction, for fairness, we perform a **binary-collapsed** comparison: (1) we map all fine-grained error types to a single "Hallucinated" label for InFi-Checker and the frontier closed-source model baseline, and (2) we evaluate all models using balanced accuracy under this binary setting (note that the FRANK results here differ from Table 3, which actually reports fine-grained evaluation).

The results are shown in Table 24, from which we observe that even under this binary-collapsed setting, InFi-Checker consistently outperforms all specialized fact-checking models across both benchmarks. This suggests that the gains of InFi-Checker are not solely due to scaling supervised fine-tuning, but stem from the proposed fine-

[1] We note that the primary challenge of InFi-Check-FG lies in correctly identifying the type of hallucination, rather than simply detecting whether an error exists. Consequently, the binary setting substantially reduces the task difficulty.

Model	Predicate	Entity	Circumstance	Co-reference	Discourse Link	Extrinsic	No Error	bacc
GPT-4o (0-shot)	80.00	76.19	44.83	50.00	3.33	84.21	84.92	60.50
GPT-4o (1-shot)	84.71	61.90	44.83	75.00	0.00	94.74	91.67	64.69
GPT-4o (3-shot)	80.00	57.14	34.48	50.00	3.33	81.58	88.49	56.43
GPT-4o (5-shot)	80.00	57.14	31.03	45.31	10.00	84.21	87.70	56.49
DeepSeek-V3.2-NoThink (0-shot)	78.82	76.19	58.62	9.38	26.67	86.84	80.95	59.64
DeepSeek-V3.2-NoThink (1-shot)	75.29	71.43	55.17	15.62	13.33	89.47	94.05	59.19
DeepSeek-V3.2-NoThink (3-shot)	82.35	71.43	34.48	18.75	23.33	78.95	75.00	54.90
DeepSeek-V3.2-NoThink (5-shot)	80.00	57.14	68.97	31.25	23.33	81.58	84.92	61.03
o3-mini (0-shot)	81.18	61.90	51.72	14.06	10.00	47.37	75.79	48.86
o3-mini (1-shot)	82.35	61.90	41.38	28.13	10.00	44.74	80.16	49.81
o3-mini (3-shot)	85.88	71.43	24.14	26.56	20.00	34.21	84.13	49.48
o3-mini (5-shot)	78.82	52.38	58.62	25.00	30.00	44.74	82.14	53.10
InFi-Checker-Llama	95.29	90.48	79.31	95.31	86.67	100.00	89.29	90.91
InFi-Checker-Qwen	93.51	91.67	87.50	100.00	88.89	96.77	88.01	92.34

Table 28: Performance comparison of the state-of-the-art LLMs with different shot num.

grained data synthesis pipeline and structured reasoning outputs, therefore learning richer and more transferable representations of hallucination.

D.7 Cost and Time Efficiency Evaluation

Table 25 presents the results of the cost efficiency analysis. We compare the computational cost of InFi-Checker with the best-performing baseline LLMs. The cost of InFi-Checker is estimated based on GPU inference time, while the cost of LLMs is calculated according to token usage in API calls. As shown in the table, InFi-Checker is substantially more cost-efficient than the closed-source baselines, with the total cost being only about one-third to one-sixth of those methods.

In addition, in terms of latency, in the long-document-long-claim setting of InFi-Check-FG, our internal measurements show that InFi-Checker-Llama and InFi-Checker-Qwen require approximately $33.8s$ and $38.2s$ per instance, respectively, showing time efficiency compared to around $50s$ for frontier LLM APIs (depending on the specific service). In real applications, the claims are often shorter. These results suggest that, despite incorporating sentence-level reasoning, InFi-Checkers does not incur prohibitive latency or cost, and remains applicable in realistic settings.

D.8 Analysis of Syntactic Fingerprints

To prove that InFi-Checker is unbiased by simple syntactic features from the training dataset, we investigate whether a shallow classifier can exploit syntactic fingerprints to explain InFi-Checker's gains. We trained a LongFormer-base-4096 (Beltagy et al., 2020) model on InFi-Check-TR as a strong discriminative baseline. The model was trained with standard hyper-parameters ($LR = 2 \times 10^{-5}$, $Epochs = 3$, $Batch\ Size = 8$), receiving document-claim pair as input, and are trained to perform 7-way classification. We transformed the validation set of InFi-Check into the LongFormer validation set to select the best model during training. The best results are shown in Table 26.

As shown in the table, the best LongFormer achieves a Balanced Accuracy (BAcc) of only 25.65%, which is significantly lower than InFi-Checker. Notably, on the LinkE category (where the reviewer observed the largest performance gap), the baseline accuracy is a mere 10.00%. These results show that a shallow classifier based on ordinary surface features is far from sufficient to explain the performance of InFi-Checker.

When varying the random seed across multiple runs, the model always tend to predict only a few of the labels (such as EntE and CorefE in the tabel above), but they did not collapse to a single fixed label bias. For example, in another random seed, the model tend to predict OutE, NoE, and EntE. This further suggests that LongFormer cannot find any valid pattern. Therefore, the success of InFi-Checker does not come from merely learning superficial lexical or syntactic cues.

D.9 Analysis of the False Negative Samples of InFi-Checker

While InFi-Checker-Llama and InFi-Checker-Qwen show slightly lower NoE accuracy than certain one-shot baselines, they achieve a significantly higher Recall on Hallucinated Data (91.37% and 92.83%, while the highest one-shot recall is 60.20%). In practical fact-checking scenarios, the cost of a False Negative (failing to detect a hallucination) is typically much higher than a False Positive (over-reporting a potential error), there-

fore, this trade-off is an acceptable design.

We also conduct a qualitative review of the False Positives of InFi-Checkers, and the result reveals a specific pattern related to compression ambiguity. Specifically, InFi-Checker is highly sensitive to how information is condensed from the grounding document. While some compression is factually "acceptable" (e.g., summarizing a list of exhibits as a single representative example), it can sometimes cross into factual distortion (e.g., incorrectly narrowing the entire scope of a museum to a single category). Distinguishing between these subtle patterns requires deep semantic reasoning. In a small number of cases, InFi-Checker's rigorous analysis flags acceptable compression as an Entity or Co-Reference Error. Conversely, baseline models often lack the sensitivity to detect these structural nuances at all. Because they do not "see" the compression, they default to a correct "No Error" label for the wrong underlying reason. In contrast, InFi-Checker's occasional over-prediction is a byproduct of its more granular and active error verification process, which is acceptable in real application scenario given the high recall of hallucinations.

E Data Structure Example

We provide an example of data from InFi-Check-FG in Figure 4, which demonstrates the data structure of InFi-Check output. "Related sentence(s) from the document" refers to the extracted evidence, while justification and correction are combined in "reason", as well as a detailed location of the hallucination for reading convenience.

F Prompts

Figure 5, Figure 6, Figure 7 and Figure 8 show the prompts in the InFi-Check pipeline. Figure 9 shows the prompt used for the LLM baselines in our experiment. Note that as shown in Table 28, increasing the number of shots yields very limited performance gains and, in some cases, causes fluctuations or degradation due to the extreme context length within InFi-Check-FG. Consequently, we limit our experimental setup to zero-shot and one-shot configurations for better model performance. Figure 10 is the prompt used for filtering samples in InFi-Check-FG.

G Human Evaluation

We did human evaluation on a sample of 100 for InFi-Check-TR. Annotators (all holding PhD or Master's degrees) are instructed to check whether the total claim and output fall into any of the given mistakes. The instructions for detecting defined mistakes are:

- Flaws in Claim: A summary error exists if: - A summary sentence is annotated as "supported or not: YES", - BUT the summary contains factual errors that contradict or mismatch with the original document (including entity, time, location, numerical, or core semantic errors). Important: - If ANY single summary sentence has this issue, the ENTIRE data sample must be considered problematic.
- Incomplete Grounding: A grounding error exists if ANY of the following is true: - The listed "related sentence(s) from the document" are NOT sufficient to support the summary sentence, and there exist other ESSENTIAL sentences in the document that must be cited. - Any listed grounding sentence does NOT actually appear in the original document (not minor truncation or paraphrase, but clearly non-existent). - The same grounding sentence is explicitly duplicated (appears two or more times). Note: - Missing an essential supporting sentence is a definite error.
- False Negative Error: A false negative error exists if: - A summary sentence is annotated as "supported or not: NO", - BUT the summary is factually correct and fully supported by the document, with NO factual discrepancies in its core meaning. This applies only when the summary is clearly correct.
- Wrong Error Type: An error-type mismatch exists if: - The summary is indeed incorrect, - BUT the annotated "error type" is not the most appropriate one according to the SIS-Fact definitions below, - AND a clearly better error type applies. Only flag this error when the mismatch is obvious. (We also provide the error taxonomy for the annotators to refer to.)
- Wrong Error Justification: A reasoning error exists if: - The annotated error type is CORRECT, - BUT the "reason" field contains incorrect analysis, such as: - Misidentifying the nature or location of the error; - Proposing a correction that is STILL factually wrong; - Explaining the error in a way that contradicts the original document. Note that the problem must be substantive and factual.

```
[
    {
        "summary sentence": "The column in Piazza Santa Maria Maggiore in Rome, crowned with a statue of the Virgin in 1614,
        set a precedent for many European columns.",
        "related sentence(s) from the document": [
            "The column in Piazza Santa Maria Maggiore in Rome was one of the first.",
            "Within decades it served as a model for many columns in Italy and other European countries."
        ],
        "supported or not": "YES",
        "reason": "The summary sentence faithfully reflects the related sentences.",
        "error type": "No Error"
        },
        {
        "summary sentence": "The first Marian column north of the Alps was Munich's Mariensäule in 1714, inspiring similar
        structures in Prague and Vienna.",
        "related sentence(s) from the document": [
            "The first column of this type north of the Alps was the Mariensäule built in Munich in 1638 to celebrate the sparing
            of the city from both the invading Swedish army and the plague.",
            "It inspired for example Marian columns in Prague and Vienna, but many others also followed very quickly."
        ],
        "supported or not": "NO",
        "reason": "This sentence is not supported by the related sentence(s).\n- Location: '1714'.\n- Explanation: The year of the
        construction of Munich's Mariensäule was changed from 1638 to 1714, falsely altering the historical timeline.\n-
        Correction: The first Marian column north of the Alps was Munich's Mariensäule in 1638, inspiring similar structures in
        Prague and Vienna.",
        "error type": "Circumstance Error"
    },
    {
        "summary sentence": "The Prague column, built post-Thirty Years' War, was destroyed in 1918 due to its association with
        Habsburg rule.",
        "related sentence(s) from the document": [
            "The Prague column was built in Old Town Square (Staroměstské náměstí) shortly after the Thirty Years' War in
            thanksgiving to the Virgin Mary Immaculate for helping in the fight with the Swedes.",
            "Unfortunately, many Czechs later connected its placement and erection with the hegemony of the Habsburgs in their
            country, and after declaring the independence of Czechoslovakia in 1918 a crowd of people pulled this old monument
            down and destroyed it in an excess of revolutionary fervor."
        ],
        "supported or not": "YES",
        "reason": "All content in the summary sentence is accurately derived from the related sentences.",
        "error type": "No Error"
    }
]
```


Figure 4: An example of the data in InFi-Check-FG, the data is truncated due to space limitations.

As shown in Table 27, most errors originated in the initial claim generation or evidence extraction phases, underscoring the inherent difficulty of grounded generation and highlighting the robustness of our controlled hallucination pipeline.

Categorization			Method	Example
Intrinsic Errors	Semantic Frame Errors	Predicate Error (PredE)	Swapping Relation Masking	Its impact on theaters is now emerging. Its absence from theaters is now receding.
			Modifying Predictions	Despite it all, 2023 should still reach the $9 billion in domestic gross hoped for this year. Despite it all, 2023 reached the $9 billion in domestic gross hoped for this year.
		Entity Error (EntE)	Swapping Entities	Her last stage role was in My Fair Lady. Her last stage role was in Bless This House.
			Compressing Words	In France and Italy, he wrote his last work. In France, he wrote his last work.
		Circumstance Error (CircE)	Swapping Circumstances	The shooting left 10 students and 2 teachers dead. The shooting left 2 students and 10 teachers dead.
	Discourse Errors	Co-reference Error (CorefE)	Swapping Pronouns	Gonzales was also indicted. She was also indicted.
			Merging Sentences	The charges were first reported by the San Antonio Express-News. District Attorney Christina Mitchell did not return requests for comment. The charges were first reported by the San Antonio Express-News, who did not return requests for comment.
		Discourse Link Error (LinkE)	Reverse Logical Relationship	The six-month Hollywood labor disruption has finally ended. Immediately following the settlement, Disney announced delays in its upcoming release schedule. After the announcement of the delays in Disney's upcoming release schedule, the six-month Hollywood labor disruption has finally ended.
Extrinsic Errors			Introducing Extrinsic Information	Robert escaped to Visegrád disguised as a civilian, aided by Nicholas, son of Radoslav, who defended him against five attackers. Robert escaped to Visegrád disguised as a civilian, aided by Nicholas, son of Radoslav, a renowned swordsman known for his exceptional skill in battle, who defended him against five attackers.

Table 29: Categories of the error data, with comparison between the supported and hallucinated claims. The blue part is the selected text for modification, and the red part is the modified text.

I'll provide you with a document. Your task is to write a short summary for this document according to the following requirements:
1. The length of the summary should be within <WORD_CONSTRAIN> words.
2. Every sentence in the summary should be directly supported by the content of the document.
3. For each event, make sure every important entity such as person, location and time is kept in the summary, especially entities that occurs in parallel.
4. When doing simplification, make sure each complex event or idea remains true to the original meaning. Avoid over-simplification that leads to in-consistency with the origin document.

Document:<Document>

Directly output the summary without any extra words.

Figure 5: Prompt for writing summaries.

Here is a document with a corresponding summary. Your task is to analyze the summary sentence by sentence. For each sentence in the summary, provide the exact sentences from the document that supports the summary sentence. If the summary comes from multiple sentences, report all sentences.
You should ensure that sentences are considered as individual units based on punctuation. Specifically:
- Treat each sentence as ending at a period (.), question mark (?), or exclamation mark (!), even if multiple sentences are enclosed within quotation marks.
- Do not truncate any sentence. If a portion of a sentence is extracted (e.g., ending at a comma, semicolon, or any punctuation other than ., ?, !), you must include the rest of the sentence so that the entire sentence is fully reproduced.

You should only respond in format as described below. Do not return anything else. START YOUR RESPONSE WITH '['
Return the result as a Python list of dictionaries, where each dictionary has the following keys:
"summary sentence": The sentence from the summary.
"sentences from the document": A Python list of the exact supporting sentences from the document. Ensure that the sentences are in the same order as they appear in the original document. Each sentence should be reported fully, without any omission.

Figure 6: Prompt for locating the grounding sentences.

Here are some pieces from the SOURCE_DESCRIPTION, and a summary sentence of it. Your task is to:
1. Compare the summary with the document and determine whether if the summary is fully supported by the content in the document. Specifically, verify if all key points made in the summary are traceable back to the sentence in the document. State whether the summary is fully supported with 'YES' or 'NO'.
2. If the answer is 'NO', revise the summary to align it fully with the document. Make sure the fluency and grammar correctness of the revised sentence, and ensure it accurately reflects the information.

You should only respond in format as described below. Do not return anything else. START YOUR RESPONSE WITH '['
Return the result as a Python list of dictionaries, where each dictionary has the following keys:
"summary sentence": The sentence from the summary.
"sentences from the document": A Python list of the exact supporting sentences from the document. Ensure that the sentences are in the same order as they appear in the original document. Each sentence should be reported fully, without any omission.

Figure 7: Prompt for determining whether a summary sentence is sufficiently supported by its grounding sentences. The purple part is only used in the LLM for rewriting and is omitted for support determination.

Here is a document with a summary. Please create a fake summary based on the origin summary by the following steps:
Instruction:
1. Analyze the document and identify sentences that contain future predictions (e.g., those using modal verbs like 'will,' 'might,' or phrases like 'predict,' 'suppose').
2. Select a sentence where the prediction is the main clause, not just a subordinate clause, and modifying the prediction into a factual statement will result in the most significant change in meaning. You can modify this sentence so that the prediction is completely transformed into a factual statement about an event that has already occurred. Ensure the modified sentence is grammatically correct and fully removes any speculative language.
3. Based on the changed sentence, modify some part of the summary to include the fake information in the changed sentence, so the summary cannot be fully supported by the origin document.
Make sure the new summary should not be fully supported by the document, and not change any other part in the summary besides those associated with the modification.

You should only respond in format as described below. Do not return anything else. START YOUR RESPONSE WITH '{{'.
Return the result as a Python dictionary with the following keys:
Format:
- "original text in summary": The original sentence containing the prediction from the document.
- "chosen element": The chosen prediction or future-oriented statement in the original text.
- "modification explanation": Description of the modification.
- "modified element": The new factual statement replacing the prediction.
- "modified text": The sentence after the modification, now a factual statement.
- "explanation": A clear explanation of how the meaning of the original text has been altered.
- "full text of modified summary": The full text of the modified summary.
- "wrong information": Point out the specific wrong information introduced in the summary after the modification.
Replace any line breaks in the values with '\n' so that the dictionary can be parsed using eval().

Figure 8: Prompt for generating InFi-Check-FG. The colored part is construction-method specific.

Task:
Evaluate the given summary by comparing it with the original document and identify any errors. These errors may include incorrect information, over-simplifications, misrepresentations, or other discrepancies. The possible types of errors to consider are as follows: Predicate Error, Entity Error, Circumstance Error, Co-reference Error, Discourse Link Error, Extrinsic Error.

Instructions:
You are provided with the full text of the original document and a summary that may contain errors. You should analyze the summary sentence by sentence and returning the results in the following Python list format:
Each item in the list should correspond to a summary sentence and be represented as a Python dictionary with the following keys:
- "summary sentence": The summary sentence.
- "related sentence(s) from the document": A list of sentences from the original document that support the summary sentence, if the summary sentence is fully supported. If the summary sentence contains an error, list the sentences needed to point out the error.
- "supported or not": "YES" if the summary sentence is fully supported by the related sentences, otherwise "NO".
- "reason": A brief analysis explaining whether the summary sentence is supported or not. If the summary sentence is not supported, specify where the error occurs, explain the incorrect information conveyed in the summary, and provide a corrected version of the sentence.
- "error type": The type of error found (choose from the types listed above), or "No Error" if the sentence is fully supported.

Document: <Document>
Summary: <Summary>

Figure 9: Prompt for evaluating LLM baseline.

You are an expert auditor for synthetic data quality. Your task is to strictly examine whether the given synthetic annotation data contains ANY clear construction errors, according to the error categories defined below. If a clear problem exists, you MUST identify it and mark the data as problematic. If no clear problems exist, mark the data as valid.

<POSSIBLE ERROR TYPES>
<INPUT DATA>

Final Judgment Rules
- If ANY of the five error types is found in ANY summary sentence, the entire data sample is problematic.
- If no errors are found, the data sample is valid.
- When evidence is unclear or debatable, flag an error.

Requirements
1. For the entire data sample, judge whether it contains ANY of the 5 types of construction errors;
2. Output ONLY JSON format with the following fields:
- is_problematic: Boolean (True = has any construction error, False = no errors);
- error_details: element is a dict with "sentence_idx", "error_type"
- empty if no errors;
- reason: Str (brief summary of all errors, or "No construction errors detected").

Figure 10: Prompt for the filtering process of InFi-Check-FG. Possible error types are the same as human evaluation instructions, and the input data are the document-claim pairs.