TU WIEN Informatics

Offene Informationsextraktion zur Faktenprüfung großer Sprachmodelle

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Ilir Osmanaj, B.Sc
Matrikelnummer 11770999

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Ass.PhD. Gábor Recski
Mitwirkung: Adam Kovacs

Wien, 22. April 2025

[redacted]

Ilir Osmanaj

Signed by:	Gabor Andras Recski
Date:	23.04.2025 05:36:16

This document is digitally signed!

This document bearing a qualified electronic signature has the same legal standing as a handwritten document in accordance with Article 25 (2) of Regulation (EU) No. 910/2014 of 23 July 2014 ("eIDAS-VO").

Validation: Information for checking the electronic signature can be found at: www.a-trust.at/pdf

Gábor Recski

TU WIEN Informatics

Open Information Extraction for Fact-Checking Large Language

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Ilir Osmanaj, B.Sc
Registration Number 11770999

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Ass.PhD. Gábor Recski
Assistance: Adam Kovacs

Vienna, 22nd April, 2025

Ilir Osmanaj

Gábor Recski

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Ilir Osmanaj, B.Sc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 22. April 2025

Ilir Osmanaj

Danksagung

Ich möchte meinem Hauptbetreuer, Univ.Ass. PhD Gábor Recski, und meinem Zweitbetreuer, Adam Kovacs, meinen aufrichtigen Dank aussprechen – für ihre Unterstützung, ihre Anleitung sowie die wertvollen Diskussionen und Einblicke während der gesamten Forschungsarbeit. Ihre Betreuung war entscheidend für die Ausrichtung und Qualität dieser Arbeit.

Ein herzlicher Dank gilt meiner Familie für ihre lebenslange Unterstützung – ich bin meinen Eltern unendlich dankbar für ihre Opfer und ihren Glauben an mich. Meiner Frau Donika und meiner Tochter Siera – ihr seid meine Inspiration und meine Motivation, stets weiterzumachen. Ohne euch hätte ich das nicht geschafft! Ich liebe euch von ganzem Herzen.

Acknowledgements

I would like to express my sincere gratitude to my main supervisor, Univ.Ass.PhD Gábor Recski and my co-supervisor, Adam Kovacs, for their support, guidance, valuable discussions and insights throughout the course of this research. Their mentorship has been instrumental in shaping this thesis.

A heartfelt thank you to my family for their life-long support — I'm forever grateful for my parents' sacrifices and belief in me. To my wife, Donika, and my daughter, Siera — you are my inspiration and my motivation to keep moving forward. I could not have done this without you! I love you all deeply.

Kurzfassung

Große Sprachmodelle (Large Language Models, LLMs) haben bemerkenswerte Fähigkeiten bei der Erzeugung kohärenter und kontextuell relevanter Antworten gezeigt. Dennoch stellt ihre Neigung zur Generierung von Halluzinationen – also plausibler, aber falscher oder nicht treuer Informationen – eine erhebliche Herausforderung für praktische Anwendungen dar. Daher wurde die Erkennung solcher Halluzinationen in von LLMs erzeugten Texten intensiv erforscht. Frühere Studien haben sich jedoch überwiegend auf grobkörnige Ansätze oder textbasierte Spanerkennung gestützt, was die präzise Erkennung von Halluzinationen auf der Ebene einzelner Wissenseinheiten erschwert.

In dieser Arbeit präsentieren wir ein Framework zur Halluzinationserkennung auf Wissensebene, das Halluzinationen auf der Ebene strukturierter Wissenseinheiten – konkret Triplets (arg1, relation, arg2) – identifiziert. Unser Ansatz extrahiert und überprüft Triplets sowohl aus den von LLMs generierten Ausgaben als auch aus Referenztexten und vergleicht diese beiden Mengen zur Erkennung von Halluzinationen. Durch den Einsatz von LLMs sowohl für die Triplet-Extraktion als auch für die Validierung vermeiden wir komplexe mehrstufige Pipelines und erreichen gleichzeitig eine hohe Erkennungsgenauigkeit.

Zur Evaluation unseres Ansatzes führten wir Experimente zur Halluzinationserkennung durch, wobei wir uns auf besonders herausfordernde Fälle konzentrierten – darunter absolute Erfindungen, kontextuelle Erfindungen und detaillierte Informationsverfälschungen. Unser Framework wurde mit halluzinierten Daten getestet, die mithilfe des BioASQ-Datensatzes und unseres Hallucinated Data Generators erzeugt wurden.

Die experimentellen Ergebnisse zeigen, dass unser Ansatz ein starkes Gleichgewicht zwischen Sensitivität (0,88) und Spezifität (0,81) erreicht und die bisherige Methode zur Halluzinationserkennung auf Wissensebene deutlich übertrifft. Im Vergleich zu bestehenden Triplet-basierten Verifizierungsmodellen verbessert unser Framework nicht nur die Erkennungsgenauigkeit, sondern reduziert auch die Anzahl notwendiger Verifizierungsanfragen und steigert so die Effizienz.

Darüber hinaus heben unsere Experimente die entscheidende Rolle von Prompting-Techniken bei der Verbesserung der Halluzinationserkennung hervor. Strukturierte und detaillierte Anweisungen steigern die Faktentreue erheblich, während Few-Shot-Beispiele und Chain-of-Thought-Reasoning die Spezifität verbessern. Unser leistungsstärkster

Prompt, der detaillierte Anweisungen mit Chain-of-Thought-Reasoning kombinierte, erzielte eine Sensitivität von 0,88 und eine Spezifität von 0,81. Diese Ergebnisse unterstreichen den Einfluss von Prompt Engineering auf die Optimierung der Halluzinationserkennung bei LLMs.

Unsere Studie stellt ein effizientes und fein abgestuftes Framework zur Erkennung von Halluzinationen vor und bietet eine skalierbare und präzise Lösung zur wissensbasierten Faktenerkennung in großen Sprachmodellen.

Abstract

Large language models (LLMs) have demonstrated remarkable capabilities in generating coherent and contextually relevant responses. However, their tendency to produce hallucinations—plausible but incorrect or unfaithful information—poses a significant challenge for practical applications. As a result, hallucination detection in LLM-generated outputs has been extensively studied. However, previous studies have primarily relied on coarse-grained approaches or text-based span detection, making it challenging to detect hallucinations at precise knowledge units.

In this thesis, we present a knowledge-level hallucination detection framework that identifies hallucinations at the level of structured knowledge units, specifically triplets (arg1, relation, arg2). Our approach extracts and verifies triplets from both LLM-generated outputs and reference texts, comparing these two sets for hallucination detection. By leveraging LLMs for both triplet extraction and validation, our method circumvents the need for complex multi-step pipelines while maintaining high detection accuracy.

To evaluate our approach, we conducted hallucination detection experiments focusing on challenging hallucination cases, including Absolute Fabrication, Contextual Fabrication, and Detailed Information Modification. We tested our framework using hallucinated data generated with the BioASQ dataset and our Hallucinated Data Generator.

Experimental results demonstrate that our method achieves a strong balance between sensitivity (0.88) and specificity (0.81), significantly outperforming the previous state-of-the-art knowledge-level detection approach. Compared to existing triplet-based verification models, our framework not only enhances hallucination detection accuracy but also reduces the number of verification requests, improving efficiency.

Furthermore, our experiments highlight the critical role of prompting techniques in enhancing hallucination detection. Structured and detailed instructions significantly improve factual accuracy, while few-shot examples and chain-of-thought reasoning contribute to better specificity. Our best-performing prompt, which combined detailed instructions with chain-of-thought reasoning, achieved 0.88 sensitivity and 0.81 specificity. These findings underscore the impact of prompt engineering in optimizing LLM-based hallucination detection.

Our study introduces an efficient and fine-grained hallucination detection framework, providing a scalable and accurate solution for knowledge-level fact verification in LLMs.

Contents

CHAPTER 1

Introduction

Recent advancements in neural architectures, massive pre-training, and large-scale datasets have enabled language models to produce coherent, contextually relevant, and highly sophisticated text responses [ZYW+23]. However, alongside these impressive developments, there is growing concern about LLMs' propensity to produce hallucinations [BCL+23], which lead to content that appears credible but lacks factual accuracy. This issue is further exacerbated by LLMs' ability to generate highly persuasive, human-like responses, making it particularly difficult to identify these hallucinations. As a result, deploying language models in practical real-world information retrieval (IR) systems—such as chatbots, search engines, and recommender systems becomes increasingly challenging [HYM+23]. These solutions have integrated into our daily lives, where factual integrity is crucial. The pressing question is how to reliably detect and isolate hallucinations in generated text, especially when only certain portions of a response are incorrect or misleading [MKL+23, HRQ+24].

A predominant body of work in hallucination detection has focused on sentence-level [MLG23] or whole-response verification [LHE21, MZV+24]. While these techniques can identify large-scale errors, they may overlook or misclassify partial inaccuracies—for instance, where only a single factual detail (e.g., a date, name, or numeric value) is wrong in an otherwise correct sentence [HRQ+24].

To address these limitations, more fine-grained approaches have been explored, primarily falling into two categories: span prediction [MH22] and sub-sentence-level [MKL+23, CCC+23, EJAS24] approaches. Span prediction methods attempt to highlight hallucinated portions within a response by directly marking unsupported spans. While this technique provides detailed localization, it is highly sensitive to surface-level variations in phrasing, depends on costly fine-grained annotations, and struggles with paraphrased or multi-hop reasoning contexts [MH22, TZJ+24].

In contrast, sub-sentence-level approaches focus on verifying the factual accuracy of smaller textual units, such as entity-relationship pairs or individual claims. While these methods improve robustness in detecting factual inconsistencies, they often rely on multi-step pipelines—such as entity recognition, fact-checking modules, and knowledge graph alignment—making implementation complex [RZA+24, ZXG+24]. This challenge is especially significant when working with closed-source language models, where internal parameters are inaccessible.

Motivated by these challenges, detecting hallucinations in a reliable, fine-grained manner has emerged as an important task in natural language processing (NLP). In this work, we present a novel method for knowledge-level hallucination detection, which focuses on extracting and comparing (arg1, relation, arg2) triplets from both LLM-generated outputs and reference documents. The term knowledge-level hallucination detection refers to detecting hallucinations at the level of individual knowledge units—triplets. This term carries the same meaning as sub-sentence-level or unit-fact-level detection in previous hallucination detection research. However, we use this term because it is commonly used in the information retrieval (IR) domain, and we aim to bridge these two streams of research. By operating at a finer knowledge-unit level, we aim to isolate specific erroneous facts without dismissing entire sentences.

For a more detailed introduction to our research, the following sections provide in-depth explanations of key concepts. Section 1.1 defines and categorizes hallucinations—drawing on a widely referenced taxonomy and define what kind of hallucination we focus on. Section 1.2 provides an overview of existing hallucination detection approaches at varying levels of granularity—response, sentence, and information—then examines triplet-based and LLM-based detection methods. Finally, section 1.3 introduces our proposed approach, highlighting its advantages and outlining the key steps involved in using closed-source LLMs for knowledge-level hallucination detection.

1.1 Hallucination

Hallucination in large language models (LLMs) refers to instances where generated content appears coherent but is factually incorrect or inconsistent [HYM+23, TZJ+24]. To clearly identify hallucinations, it is essential to first define a suitable reference standard against which generated outputs are evaluated. The choice of reference depends heavily on the specific application context. For example, in retrieval-augmented generation (RAG) scenarios, consistency with provided input contexts is paramount. Conversely, general-purpose models or systems requiring precise knowledge representations must prioritize consistency with established real-world facts. This distinction constitutes the first dimension of hallucination addressed in this study—**hallucination by reference used** [HFFQ21, HYM+23].

Once a reference is established, hallucination can manifest in various ways, ranging from entirely fabricated facts to minor numerical inaccuracies. The severity and complexity of detecting hallucination vary considerably based on these deviations, leading us to

define our second dimension—**hallucination by how it differs from reference** [GVM22, LFA+18]. In the following sections, we categorize hallucinations along these two dimensions and clarify the specific aspects our study addresses.

1.1.1 Hallucination by reference used

The widely accepted categorization of hallucination is based on the reference used to identify hallucination— [HYM+23] defines two primary types: Factuality Hallucination and Faithfulness Hallucination, depending on whether the deviation occurs from real-world facts or the given input. Building on this taxonomy, hallucinations can be broadly classified from a reference-based perspective into these two categories.

A **factuality hallucination** arises when an LLM generates content that contradicts established real-world knowledge or introduces unverifiable claims. This type of hallucination misleads users by presenting false or unsupported information as fact.

In contrast, a **faithfulness hallucination** occurs when the generated output diverges from the provided reference document or user instructions. Even if the response contains factually correct information in a general sense, it may still be unfaithful to the specific source material it was supposed to align with.

Consider the following example:

- **Example Answer:**

  > "Thyroid hormone receptor beta1 (TR-beta1) upregulates ChREBP expression by interacting with LXRE3, thereby influencing T3-induced hepatic lipogenesis. This regulatory role extends to other metabolic genes, including P450."

- **Reference Document (Excerpt):**

  > "The carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c, regulated by liver X receptors (LXRs), play central roles in hepatic lipogenesis. Because LXRs and thyroid hormone receptors (TRs) influence each other's transcriptional activity, researchers investigated whether TRs control ChREBP expression. They found that thyroid hormone (T3) and TR-beta1 upregulate ChREBP by binding direct repeat-4 elements (LXRE1/2)."

From this example, we observe two distinct types of hallucinations:

- **Factuality Hallucination:** If the model generates an output stating that "TR-beta1 downregulates ChREBP expression," this contradicts established biological knowledge. Since the actual role of TR-beta1 is to **upregulate** ChREBP, this represents a factuality hallucination.
- **Faithfulness Hallucination:** If the model states that "TR-beta1 upregulates ChREBP by interacting with LXRE3," the overall claim remains scientifically valid; however, it diverges from the given reference, which explicitly states that TR-beta1 interacts with LXRE1/2. This discrepancy makes the output unfaithful to the source document, even though it does not contradict general scientific knowledge.

While both factuality and faithfulness hallucinations pose challenges, faithfulness hallucinations require an evaluation of the alignment between generated content and the reference source, making them more tangible yet complex to assess. This type of hallucination has become increasingly important as many services are now integrated with RAG [SPC+21] and AI agents [SRR23], where compliance with the input context is crucial. In such contexts, hallucinations that contradict the input context indicate a failure of the respective system.

Given this challenge, our research primarily focuses on **faithfulness hallucinations**. By examining the alignment between generated outputs and their source documents, we aim to develop a robust detection mechanism that enhances the reliability of LLM-generated content in applications where maintaining contextual fidelity is critical.

Hallucination by How It Differs from Reference

While many hallucinations in language model outputs are easily identifiable through explicit contradictions or verifiably false statements, more subtle forms of hallucination remain challenging to detect [ZXG+24]. In particular, hallucinations can deviate from reference materials in nuanced ways, ranging from outright fabrications to minor but critical factual distortions. Understanding these deviations is crucial for developing detection mechanisms that extend beyond surface-level inconsistencies.

Existing research has identified certain types of hallucinations that are relatively easy to detect due to their overt nature [GVM22]. However, other subtle contextual misalignments, and plausible-sounding but incorrect details—pose significant challenges [GVM22]. These subtle hallucinations often evade conventional fact-checking methods, especially in high-stakes domains where precision is paramount (e.g., medicine, finance, and academia). For example, even minor numerical alterations can lead to substantial misinterpretations, yet studies specifically addressing numerical hallucinations remain limited. Furthermore, ongoing research debates how language models encode and interpret numerical values [LG24, HTYZ24], raising concerns about their ability to detect factual inconsistencies in cases of minor numerical variation.

Given these challenges, our study focuses on hallucinations that are particularly difficult to detect. We categorize them based on how they diverge from reference content, emphasizing cases where standard verification techniques may fail. While some hallucinations, such as explicit factual contradictions, can be easily identified, more subtle deviations—such as plausible but incorrect details or minor numerical alterations—pose significant detection challenges. These issues are particularly problematic in high-stakes domains like medicine, finance, and research, where even slight factual inconsistencies can lead to serious consequences.

To systematically address these challenges, we examine three challenging types of hallucinations identified in previous studies: **Absolute Fabrication**, where entirely new and unsupported information is introduced [GVM22, QHQP23][1]; **Contextual Fabrication** , which subtly misrepresents details while maintaining a surface-level resemblance to the reference [GVM22]; and **Detailed Information Modification** , involving small but impactful changes, such as numerical distortions [DMG23, QHQP23]. A detailed explanation of these hallucination types is provided in **Section 4.5: Generation of Challenging Hallucination Types**. By analyzing these fine-grained hallucination patterns, we aim to evaluate our approach on previously unexplored aspects of hallucination detection.

1.2 Hallucination Detection Methods and Their Granularity

As large language models (LLMs) have become widely adopted across various domains, detecting hallucinations in their outputs has emerged as a critical challenge. This issue has garnered significant attention from both academia and industry, leading to the development of diverse detection strategies. Recently, the SemEval-2024 SHROOM shared task on hallucination detection [MZV+24] attracted participation from over 40 research groups, each employing various approaches such as LLM fine-tuning [DS24], LLM prompting [DS24, MHO+24, Sii24], LM fine-tuning [LSZH24], and ensemble methods

[1] Although absolute fabrications are generally considered easier to detect, they can become significantly more challenging if the fabricated information closely resembles the original context, appearing coherent and plausible [GVM22]

[MHO+24]. While these efforts demonstrate significant progress, existing research and shared tasks have predominantly focused on response-level hallucination detection, leaving finer-grained analysis largely unexplored.

Prior research on hallucination detection has explored various levels of granularity, primarily focusing on three approaches: response-level [LRW+22, LHE21], sentence-level [MLG23], and knowledge-level [MKL+23, CCC+23, HRQ+24], each presenting distinct trade-offs in precision and scope. Response-level methods assess an entire generated response, flagging it as hallucinated if any part contains inaccurate or fabricated information. While effective for simple fact-based queries, this approach becomes less informative for longer responses where only specific parts may be incorrect. Sentence-level methods refine this by identifying erroneous statements within a response, but they may still overlook localized factual inaccuracies—such as incorrect names or dates—embedded in otherwise valid sentences.

To address these limitations, knowledge-level methods focus on finer-grained units, such as entities and relationships, allowing for the detection of small-scale discrepancies without discarding entire sentences [HRQ+24, EJAS24]. While text-based methods, such as RAGAs [EJAS24], segment and compare knowledge represented purely as text, methods based on knowledge triplets have been studied over a longer period and are closely aligned with traditional research directions in information retrieval [BQH+23, YV23, RZA+24]. **triplet-based hallucination detection** decomposes text into structured tuples (arg1, relation, arg2) to capture factual statements. This approach is advantageous as it facilitates the integration of external knowledge sources, such as knowledge graphs, and aligns with existing NLP pipelines that rely on triplets. However, many current triplet-based methods involve complex, multi-step processes—including named-entity recognition, relation extraction, and knowledge graph verification—which require different models for each steps and difficult to implement compared to closed-source LLM APIs [RZA+24].

Another emerging approach leverages **LLMs for fact verification** in hallucination detection [MLG23, ZQG+23, HRQ+24]. These methods often employ LLMs for self-consistency checks or retrieval-based verification. Compared to triplet-based approaches, LLM-based methods offer a more direct implementation, as they do not require explicitly structured knowledge bases or complex extraction pipelines. However, their effectiveness at fine-grained, knowledge-level detection remains less well understood [HRQ+24]. Additionally, LLM-based fact verification heavily relies on prompt design, yet there is limited research on how to systematically construct prompts that maximize factual accuracy and reliability. Furthermore, the suitability of LLMs across diverse verification tasks remains an open question, raising concerns about their robustness and consistency.

1.3 Our Approach

With these in mind, this research proposes an knowledge-level hallucination detection framework that is both sufficiently granular and relatively simple to deploy. We first

prompt an LLM to extract triplets(arg1, relation, arg2) from both the model's generated answer and the reference documents used to generate that answer. Unlike approaches that rely on specialized pipelines with entity detectors [BQH+23, RZA+24] or external knowledge bases, our method leverages LLM capabilities to parse text into triplets. After generating two sets of triplets—one from the LLM output and one from the reference text—we then compare them to identify which triplets are unsubstantiated or conflict with the source text. By operating at the level of discrete information units, we can pinpoint localized inaccuracies rather than labeling entire sentences as hallucinated.

By testing our framework, we explore the feasibility of using LLMs for fact-checking through triplet set comparison. If successful, this approach could replace the complex multi-step pipelines of traditional triplet-based methods while enabling seamless integration with information retrieval-based approaches. This would allow for a more scalable and adaptable fact verification model, bridging structured knowledge-based verification with retrieval-augmented methods.

Furthermore, to address the limitations of LLM-based fact verification, our approach investigates techniques to mitigate hallucinations that may arise within the verification process itself. Specifically, we explore methods to enhance the reliability and performance of LLM-based fact verification by refining prompt design strategies and incorporating self-consistency mechanisms to maximize accuracy. Since LLM performance is highly dependent on prompt formulation, we also analyze different prompt structures to determine optimal configurations for improving factual accuracy.

Key Advantages

- Adequate Granularity: By focusing on information units, we can isolate small-scale factual discrepancies while preserving correct information.
- Simplicity: Our two-step process eliminates the need for multi-layered pipelines, reducing implementation complexity and potential points of failure.
- Compatibility: The method naturally integrates with prior triplet-based pipelines and can be easily adapted to LLM-based frameworks.
- Efficiency: Unlike previous triplet-based approaches [YV23, HRQ+24], which verify answer triplets one-by-one, necessitating separate model runs for each answer triplet. For instance, verifying 100 answer triplets requires 100 separate comparisons. In contrast, our method simultaneously evaluates multiple answer triplets against all reference triplets in a single model run, significantly enhancing computational efficiency.
- Optimized Prompt Design: By refining prompt structures, our approach enhances the reliability of LLM-based fact verification, mitigating hallucination risks in verification outputs and improving overall model performance.

The remainder of this thesis is organized as follows. Section 1 provides a comprehensive background on hallucination detection, defining key concepts and discussing different categorization methods. It further explores existing detection approaches, focusing on triplet-based and LLM-based methods, before introducing our proposed approach and its key advantages. Section 2 reviews related works, covering previous triplet-based and LLM-based hallucination detection methods, their limitations, and our contributions to the field. Section 3 details our proposed method, including how we integrate triplets and LLMs for hallucination detection, the triplet generation and verification processes, and our optimized prompting strategies. It also outlines the key research questions guiding our work. Section 4 describes our experimental setup, including the hallucination detection task, evaluation metrics, dataset characteristics, generation of hallucinated triplets, and experimental conditions. Section 5 presents and discusses our results, analyzing detection performance across different hallucination types, the impact of prompting methods, ablation studies, and comparisons with reference-text-based approaches. Finally, Section 6 concludes the study by summarizing our key findings, acknowledging limitations, and suggesting future directions for advancing LLM-based hallucination detection research.

CHAPTER 2

Related Works

2.1 Related Works

Building upon the challenges introduced in Introduction, we now survey relevant methods and frameworks in hallucination detection. While Introduction (Section 1) provides a broad overview of hallucination detection across multiple approaches and levels of granularity, this section focuses specifically on research most relevant to our work.

Much of the early focus has been on either labeling an entire *response* as hallucinated or verifying content at the *sentence* level. While these approaches can quickly flag large-scale factual inconsistencies, they often overlook localized inaccuracies that involve only a fraction of a sentence—for example, an incorrect year or misattributed name [MLG23, MKL+23]. To address this limitation, some researchers have pursued more granular techniques, such as extracting *sub-sentence* content units or knowledge *triplets*, to precisely pinpoint erroneous facts without dismissing otherwise correct material.

Given our focus on knowledge-level hallucination detection, we primarily review studies that employ fine-grained, knowledge-based, and related approaches.

2.2 Triplet-Based Hallucination Detection

Among knowledge-level research approaches, triplet-based verification has emerged as a structured method for identifying factual inconsistencies by decomposing text into discrete knowledge units—triplets. This approach represents information as *(arg1, relation, arg2)* statements, enabling more precise fact-checking. Several studies have explored the use of triplets as a structured backbone for detecting hallucinations. *RHO* [JLL+22], for instance, mitigates extrinsic and intrinsic hallucinations by grounding each token in an underlying knowledge graph, followed by a re-ranking module that filters out passages misaligned with the extracted entities and relations. *FLEEK* [BQH+23] refines this

concept by segmenting content into discrete factual triples, retrieving external evidence from the web or knowledge bases, and rewriting segments that fail verification. Similarly, *FactAlign* [RZA+24] constructs a knowledge graph from generated text and aligns it with a reference graph, detecting factual discrepancies through node and edge mismatches. Meanwhile, *Query-Based Entity Comparison* [PKCGH19] employs SPARQL queries to validate structured constraints (e.g., "both entities have more than 30,000 employees"), though its primary focus remains on entity-to-entity comparisons rather than broader open-ended text generation.

While these methods highlight the advantages of triplet-based verification, many rely on complex multi-step pipelines [RZA+24, ZXG+24] involving named-entity recognition, knowledge graph construction, and specialized reasoning modules. Such dependencies make them difficult to scale, particularly in scenarios involving Language Models(LM) or closed-source LLM APIs. Furthermore, these structured approaches presuppose well-defined entities and relations, limiting their adaptability to domains where knowledge is unstructured or contextually implied.

An alternative approach proposed by [YV23] introduces a zero-shot fact-checking framework that leverages triplet-based verification in conjunction with natural language inference (NLI) models. Unlike traditional knowledge graph alignment techniques, this method employs NLI models to directly assess the relationships between claim triplets and evidence triplets. The two-stage verification process—triple-level verification followed by claim-level aggregation—demonstrates superior generalization to adversarial datasets such as FEVER-Symmetric, FEVER 2.0, and Climate-FEVER. However, the approach performs a one-to-one comparison between all answer triplets and source triplets, leading to computational complexity that scales proportionally with the number of triplets in both sets. While effective in zero-shot settings, this design imposes scalability constraints, particularly when handling large-scale fact verification in real-world applications.

2.3 LLM-Based Hallucination Detection

Triplet-based approaches center on structured knowledge representations, but another line of research tackles hallucinations by treating the language model itself as both the source of answers and the judge of correctness. *SelfCheckGPT* [MLG23] exploits the idea that factual statements should consistently reappear across multiple sampled outputs from the same model; conflicting or contradictory restatements suggest hallucinations. In a similar vein, *FActScore* [MKL+23] decomposes long responses into "atomic" facts—short expressions of single factual claims—and uses LLMs to verify each fact individually. These methods rely on the LLM's own knowledge and consistency to flag spurious segments and can perform well for QA or short text. However, they may struggle with long-form or domain-specific tasks, where partial misrepresentations of numeric or technical information can be overshadowed by the model's otherwise consistent style.

Recent frameworks adopt large language models directly to extract or verify triplets [FHT+24, CCC+23, HRQ+24], aiming to simplify multi-step processes or replacing

trained in-house model with LLM. *FacTool* uses a five-stage pipeline—(LLM based) claim extraction, query generation, tool querying, evidence aggregation, and final (LLM based) agreement—to detect factual errors across varied tasks (QA, code generation, math solving, scientific literature) via LLM prompting.

In contrast, *REFCHECKER* [HRQ+24] systematically extracts "claim-triplets" from the LLM's output and evaluates each against a reference text. Compared to sentence-level checks, their triplet-based approach isolates smaller, more precise factual units. Their results show that evaluating knowledge at the triplet level yields significantly fewer missed hallucinations, outperforming sub-sentence or sentence-based detectors by large margins. However, these approaches primarily focus on triplet extraction rather than comparing triplet sets in a structured verification process. Furthermore, since REFCHECKER verifies each claim-triplet individually, the computational complexity of the fact-checking process scales linearly with the number of extracted triplets, making it more resource-intensive for longer responses with numerous factual claims.

2.4 Limitations and our Contribution

Overall, both LLM-based approaches and knowledge-centric pipelines show promise in mitigating hallucinations; however, each comes with trade-offs. LLM-based methods rely on the model's own internal knowledge and consistency to verify factuality, but they are highly dependent on prompt design and the performance of the LLM on comparing two triplet set is untested. There is still limited research on how to systematically construct prompts that maximize factual accuracy and reliability. The lack of standardized evaluation frameworks for LLM-driven verification further complicates efforts to optimize prompt engineering techniques for different verification tasks.

On the other hand, triplet-based hallucination detection provides a more structured approach by breaking down text into (arg1, relation, arg2) units, enabling more granular verification. However, existing triplet-based methods typically require multi-step pipelines involving named-entity recognition, relation extraction, and knowledge graph construction. These pipelines often necessitate different models for each step and require dedicated computational resources for execution. Moreover, while previous research has demonstrated the benefits of triplet extraction for factuality assessment, most existing approaches primarily focus on extracting triplets while relying on pairwise natural language inference or traditional knowledge graph comparison methods for triplet evaluation. The former requires a one-to-one comparison for each triplet in the source and answer sets, leading to an exponential increase in model computations as the number of triplets grows [FHT+24, SRMS24]. The latter assumes clearly defined entities and relations for both source and answer triplets, making it less adaptable when entities are paraphrased, relations are implied, or the underlying knowledge structure is not explicitly defined [RZA+24]. This gap highlights the need for a more efficient approach that leverages LLMs for structured triplet verification while reducing dependence on predefined knowledge graphs.

To bridge these gaps, our work integrates knowledge-level granularity via triplets with LLM-based verification methods. Specifically, we evaluate the feasibility of using LLMs to perform structured triplet-set comparisons, rather than relying on traditional triplet-based pipelines, which typically involve multiple processing stages with specialized models at each step. Furthermore, our approach significantly reduces the number of required model runs—at least by a factor of $\frac{\text{average length of input triplet}}{\text{(input sequence length - C)}}$, where C denotes the length of additional prompt components such as instructions and reference triplets—thereby improving both computational speed and efficiency.

To elaborate, traditional approaches necessitate separate model runs for each answer triplet; for example, verifying 100 answer triplets requires 100 separate comparisons, each involving all reference triplets against one answer triplet. In contrast, our method enables simultaneous verification of multiple answer triplets against all reference triplets within a single model run. Because the maximum number of tokens per model run is limited by the model's input sequence length, each run accommodates as many answer triplets as possible after accounting for other prompt elements. Concretely, our method can validate approximately $\frac{\text{(input sequence length - C)}}{\text{average length of input triplet}}$ answer triplets per run.

If successful, this approach could replace complex, multi-step knowledge graph alignment processes and facilitate seamless integration with LLM-based methods. Additionally, we explore techniques to enhance the reliability of LLMs in fact verification by systematically refining prompt structures and incorporating self-consistency mechanisms, addressing a critical challenge in contemporary LLM-based fact-checking research.

In the next sections, we describe our approach to triplet extraction and knowledge verification, then detail how we address these issues of scalability, partial references, and localized claim errors.

CHAPTER 3

Proposed Method

In this section, we introduce a knowledge-level hallucination detection framework that capitalizes on the strengths of Large Language Models (LLMs) by decomposing and verifying text at the granularity of *(arg1, relation, arg2)* triplets. Our approach comprises two principal steps: (i) *Triplet Generation*, which extracts factual statements from both the LLM's generated outputs and the corresponding reference materials; and (ii) *Triplet Validation*, where these sets of triplets are systematically compared to identify any unsubstantiated claims. By encapsulating knowledge in structured triplets, we not only enhance the precision of hallucination detection but also preserve the valid segments of text that do not require correction.

3.1 Triplet-Based LLM Hallucination Detection

Our proposed framework uses a two-stage pipeline aimed at isolating hallucinations at a knowledge-level approaches. Specifically, we focus on:

1. **Triplet Generation:** We prompt a LLM to extract *(arg1, relation, arg2)* triplets from its own generated answers and from the reference documents used during answer generation.
2. **Triplet Validation:** We compare the triplets derived from the LLM's output with those extracted from the reference text. Any triplet that is not directly supported by, or is in conflict with, the reference triplets is flagged as a hallucination.

By implementing LLM-based comparison of triplets, we can integrate multiple steps of conventional pipeline approaches into a single prompt-driven mechanism, thus reducing complexity while leveraging the reasoning and language-understanding capabilities of large models. Moreover, this *knowledge-level* checking can serve as a bridge between

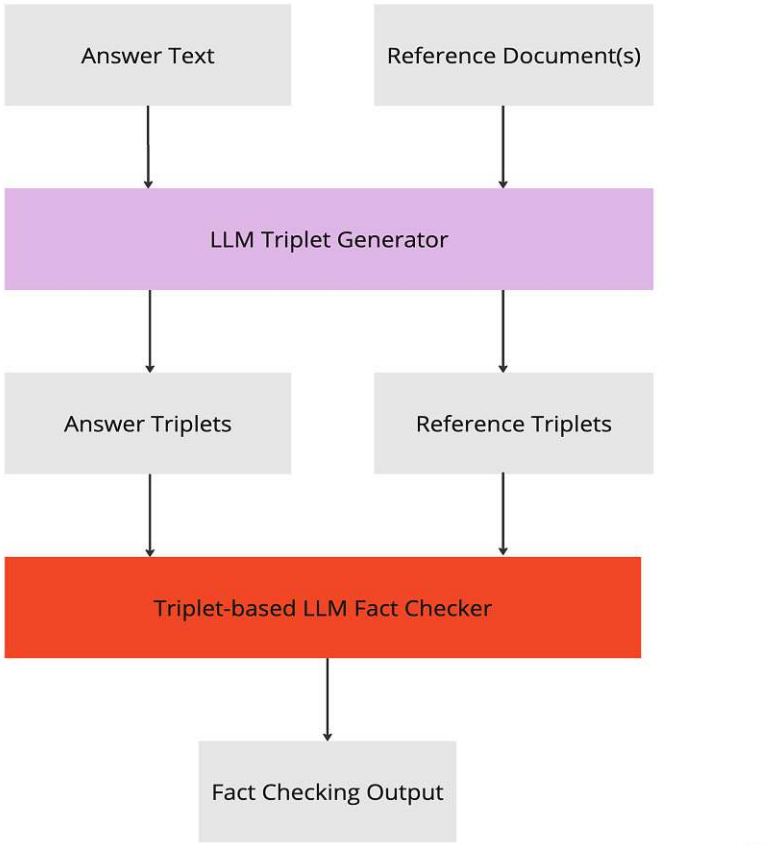

Figure 3.1: LLM Fact Checker

purely LLM-based self-consistency checks and more traditional triplet-based systems, thereby retaining the advantages of both. The whole pipeline of our approach is illustrated in Figure 1.

We introduce a real example from the BioASQ dataset throughout the rest of the report, as it helps illustrate our method, as well as the structure of its inputs and outputs:

1. **Question:**

   "Which genes does thyroid hormone receptor beta1 regulate in the liver?"

2. **Ground Truth Answer(Included in Dataset):**

   "LDL receptor, ChREBP, malic enzyme, cytochrome P450 oxidoreductase"

3. **Example Answer:**

   "Thyroid hormone receptor beta1 (TR-beta1) upregulates ChREBP expression by interacting with LXRE2, thereby influencing T3-induced hepatic lipogenesis. This regulatory role extends to other metabolic genes, including P450."

4. **Reference Document (Excerpt):**

   "The carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c, regulated by liver X receptors (LXRs), play central roles in hepatic lipogenesis. Because LXRs and thyroid hormone receptors (TRs) influence each other's transcriptional activity, researchers investigated whether TRs control ChREBP expression. They found that thyroid hormone (T3) and TR-beta1 upregulate ChREBP by binding direct repeat-4 elements (LXRE1/2)."

5. **(Generated) Answer Triplets:**

   - `["Thyroid hormone receptor beta1 (TR-beta1)", "upregulates", "ChREBP expression by interacting with LXRE2"]`
   - `["TR-beta1", "influences", "T3-induced hepatic lipogenesis"]`
   - `["TR-beta1's regulatory role", "extends to", "other metabolic genes, including P450R"]`

6. **(Generated) Reference Triplets:**

   - `["carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c", "are regulated by", "liver X receptors (LXRs)"]`
   - `["carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c", "play central roles in", "hepatic lipogenesis"]`
   - `["liver X receptors (LXRs) and thyroid hormone receptors (TRs)", "influence", "each other's transcriptional activity"]`
   - `["researchers", "investigated whether", "thyroid hormone receptors (TRs) control ChREBP expression"]`
   - `["thyroid hormone (T3) and TR-beta1", "upregulate", "ChREBP"]`
   - `["thyroid hormone (T3) and TR-beta1", "upregulate ChREBP by binding", "direct repeat-4 elements (LXRE1/2)"]`

7. **Identified Hallucinated Triplet:**

   `["TR-beta1's regulatory role", "extends to", "other metabolic genes, including P450R"]`

3.2 LLM Triplet Generator

For triplet generation, we prompt the LLM to decompose a given text—whether from its own generated response or a reference document—into structured *(arg1, relation, arg2)* triplets. To ensure the accuracy, interpretability, and completeness of the extracted triplets, the following principles are enforced:

1. **Full Contextualization:** Each argument (*arg1* and *arg2*) should be explicitly defined to eliminate ambiguity. Pronouns, vague descriptors, or insufficiently specified terms must be avoided to ensure that each triplet can be interpreted independently of its original context[1]. For example, consider the following triplet extracted from a model-generated response:

   Incorrect: `["This regulatory role", "extends to", "other metabolic genes, including P450R"]`

   To improve clarity, the subject should be explicitly specified:

   Correct: `["TR-beta1's regulatory role", "extends to", "other metabolic genes, including P450R"]`

[1] Additionally, when the same word appears multiple times with different meanings the extracted triplets must retain sufficient contextual information to distinguish between distinct usages.

By ensuring explicit references, the extracted triplets become more interpretable and self-contained, thereby reducing potential ambiguity during validation.

2. **Accuracy and Completeness:** The extracted triplets must fully capture the factual assertions present in the source text. Essential relationships should not be omitted, nor should information be excessively generalized, as such omissions may lead to misinterpretation.

   Example: Suppose the generated response includes the following statement:

   > TR-beta1 interacts with SREBP-1c, a key regulator of lipid metabolism.

   A triplet that fails to retain critical contextual information would be:

   Incomplete triplet: `["TR-beta1", "interacts with", "SREBP-1c"]`

   Whereas a well-structured triplet that preserves the full meaning of the original statement would be:

   Complete triplet: `["TR-beta1", "interacts with", "SREBP-1c, a key regulator of lipid metabolism"]`

   By incorporating essential contextual details, the complete triplet ensures alignment with the intended meaning of the source text.

3. **Grammatical and Semantic Consistency:** The extracted triplets must be grammatically well-formed and semantically coherent. The *relation* component must correctly and unambiguously link *arg1* and *arg2*, preserving the factual integrity of the original text. Logical inference errors and grammatical inconsistencies should be minimized, as they may lead to misinterpretations.

   Example: Given the following source statement:

   > "T3 was shown to accelerate intracellular calcium transients and reduce diastolic calcium levels, suggesting a sensitization of the contractile apparatus to calcium."

   A triplet extracted without proper contextual awareness may introduce errors:

   Incorrect: `["T3", "suggests", "sensitization of the contractile apparatus to calcium"]`

   In contrast, a correctly structured triplet that preserves the logical flow of the original statement is:

   Correct: `["Findings that T3 was shown to accelerate intracellular calcium transients and reduce diastolic calcium levels", "suggest", "sensitization of the contractile apparatus to calcium"]`

   This approach ensures that the extracted triplet accurately reflects the causal relationship expressed in the original text.

3.3 Triplet Fact Checker

Following triplet generation, the *Triplet Fact Checker* evaluates the factual alignment between the triplets extracted from the model-generated response and those obtained from the reference documents. Notably, an exact lexical match is not a prerequisite for validation; rather, the verification process relies on semantic entailment. The fact checker must assess whether a generated triplet is logically supported by the reference triplets, even if paraphrased, restructured, or inferred from multiple sources. This is critical, as factual verification in real-world contexts frequently involves implicit reasoning rather than direct textual correspondence.

For example, consider the triplets extracted from the model-generated response and the reference document:

- **Answer Triplet:** `["Thyroid hormone receptor beta1 (TR-beta1)", "upregulates", "ChREBP expression by interacting with LXRE2"]`
- **Reference Triplet:** `["thyroid hormone (T3) and TR-beta1", "upregulate ChREBP by binding", "direct repeat-4 elements (LXRE1/2)"]`

Although these triplets are not lexically identical, they convey equivalent factual information. The fact checker must recognize that the answer triplet is semantically supported by the reference triplet rather than erroneously flagging it as a hallucination.

In instances where no verbatim match exists between the answer triplet and any reference triplet, the fact checker must employ inferential reasoning to determine factual consistency. Consider the following example:

- **Answer Triplet:** `["T3 signaling", "induces", "fatty acid metabolism in hepatocytes"]`
- **Reference Triplets:**
  - `["thyroid hormone (T3)", "activates", "TR-beta1"]`
  - `["TR-beta1", "regulates", "ChREBP"]`
  - `["ChREBP", "induces", "fatty acid metabolism in hepatocytes"]`

Here, the claim that `["T3 signaling", "induces", "fatty acid metabolism in hepatocytes"]` is not explicitly stated in any single reference triplet. However, the reference triplets collectively establish a logical progression: *T3 activation* → *TR-beta1 regulation* → *ChREBP activation* → *fatty acid metabolism*. Since this causal chain substantiates the claim made in the answer triplet, the fact checker should infer that the information is supported rather than hallucinated.

Conversely, consider the following triplet:

- **Answer Triplet:** `["TR-beta1's regulatory role", "extends to", "other metabolic genes, including P450R"]`

This triplet lacks supporting evidence in the reference text. While the reference triplets confirm that TR-beta1 regulates ChREBP, they do not establish any connection between TR-beta1 and "other metabolic genes, including P450R." Consequently, this triplet is flagged as a hallucination.

The objective of our fact checker is to refine hallucination detection by distinguishing between supported and contradicted triplets, thereby improving the reliability of factual consistency assessments. Our approach aims to highlight contradicted triplets explicitly and classify them as potential hallucinations for further review.

3.4 Prompting Method

The efficacy of our system heavily depends on well-crafted prompts that guide the LLM in both triplet generation and triplet validation. We employ the following techniques:

(1) Few-shot Generation We provide the LLM with short demonstration examples to illustrate the expected structure and depth of reasoning [TLI+23]. This approach:

- Improves *formatting consistency*, ensuring systematic presentation of extracted and validated triplets.
- Boosts *accuracy* by furnishing concrete patterns for the model to emulate.
- Enhances *instruction compliance* by reinforcing the importance of the user's guidelines through explicit examples.

(2) Chain-of-Thought (CoT) We encourage the model to articulate its intermediate reasoning steps before delivering the final output [WWS+22]. This includes:

- **Transparent Reasoning**: Users can examine the logical basis for each *True/False* verdict.
- **Error Reduction**: Articulating intermediate thoughts helps the model identify and correct contradictions early.
- **Clear Delineation of Result and Rationale**: Separating the reasoning details from the final label (e.g., *True* vs. *False*) makes the model's verdict easier to verify.

Because our task differs from typical generation tasks, where a single response is generated, we instead produce multiple outputs from multiple triplets. To ensure consistency and accuracy, we prompt the model to generate a concise CoT inference for each triplet, allowing for more structured and interpretable verification.

(3) Detailed Instructions We provide comprehensive and explicit instructions to ensure that the model fully understands the task requirements, leading to more accurate and relevant outputs [OWJ+22]. This includes:

- **Output Formatting**: Clearly defining the expected structure of the model's response to maintain consistency.
- **Validation Criteria**: Specifying the exact rules and conditions for determining whether a triplet is supported by the source data.
- **Guided Responses**: Providing step-by-step instructions to help the model align its outputs with the user's intended objectives.

By integrating these prompting techniques, we optimize the LLM's ability to generate and validate triplets with greater accuracy, consistency, and interpretability. Prior research has shown that few-shot prompting [TLI+23] improves instruction adherence and structured output generation. Similarly, Chain-of-Thought reasoning [WWS+22] has been demonstrated to enhance logical coherence in complex decision-making tasks, and explicit instructions [OWJ+22] significantly reduce hallucination rates by providing clearer operational constraints. Our approach synthesizes these best practices, ensuring that the model performs reliable fact verification while maintaining efficiency and scalability across diverse input conditions. The detailed prompts used in our methodology are provided in the Appendix.

3.5 Research Questions

Building upon the proposed methodology and prior work, our study aims to investigate the feasibility and effectiveness of LLM-based triplet comparison for hallucination detection. Specifically, we address the following research questions:

1. **How can we optimize prompt design for triplet comparison to maximize accurate hallucination detection?**
   - Given that LLM-based fact verification is highly sensitive to prompt formulation [OWJ+22], we investigate the impact of different prompt structures—including persona-based, few-shot, and chain-of-thought prompting—on model accuracy and consistency.
2. **Can an LLM effectively compare knowledge-level triplets extracted from different sources without requiring multi-step pipelines?**
   - Traditional triplet-based methods rely on multi-stage processing, including named-entity recognition and knowledge graph construction [RZA+24]. We examine whether an LLM can perform this comparison in a single inference

step while maintaining reliability.To the best of our knowledge, this is the first study to investigate this approach.

3. **What limitations arise when using an LLM for triplet validation?**

   - Since LLMs have not been extensively tested for directly validating triplets, this approach may have inherent limitations. We analyze whether this validation introduces systematic biases, reliability issues, or inconsistencies in factual verification.

4. **How does performance vary across different hallucination types, and what refinements are necessary to handle LLM-based fact verification challenges?**

   - Hallucinations can manifest in different forms—fabricated entities, incorrect numerical values, and misattributed relationships [MKL+23]. We evaluate whether certain types of hallucinations are more difficult to detect and propose refinements for improving robustness.

By addressing these questions, we aim to develop a streamlined yet rigorous approach for hallucination detection that integrates structured knowledge validation with LLM-based reasoning. The subsequent sections detail our empirical setup, evaluation metrics, and findings, demonstrating the effectiveness of our method in minimizing hallucination risks while maintaining interpretability and efficiency.

CHAPTER 4

Experiments

In this section, we describe the experimental setup used to evaluate our proposed hallucination detection framework. Our primary goal is to assess the effectiveness of LLM-based triplet comparison in identifying hallucinated content while minimizing false positives. We conduct experiments on a well-defined hallucination detection task, establish appropriate evaluation metrics, and systematically compare different prompting strategies. Additionally, we introduce a hallucinated data generator to create controlled test cases and explore various aspects that influence our model's performance.

4.1 Hallucination Detection Task

We evaluate our approach by conducting a hallucination detection task, where the objective is to assess the model's ability to distinguish hallucinated triplets in LLM-generated responses. This evaluation follows the framework introduced by FactScore [MKL+23].

Task Definition: The hallucination detection task involves determining whether the information contained in a generated triplet aligns with a given set of reference triplets. Specifically, the model must assess whether a triplet is semantically supported by the reference triplets or if it introduces incorrect or unsubstantiated information.

If a triplet from the generated answer is logically entailed by or explicitly supported by the reference triplets, it is classified as **supported**. Conversely, if the generated triplet introduces incorrect, contradictory, or unverified information, it is labeled as **hallucinated**. By performing this evaluation at the triplet level rather than at the sentence or response level, our approach ensures a more precise and fine-grained assessment of factual consistency.

The classification for each triplet t is formally defined as follows:

$$\mathcal{F}_{\text{FactCheck}}(t, T_{\text{ref}}) = \begin{cases} \text{Supported}, & \text{if the LLM predicts that } t \text{ is entailed by } T_{\text{ref}}, \\ \text{Hallucinated}, & \text{otherwise.} \end{cases}$$

Example:

- **Extracted Answer Triplet:** `["TR-beta1", "directly regulates", "SREBP-1c expression"]`
- **Reference Triplets:**
  - `["carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c", "are regulated by", "liver X receptors (LXRs)"]`
  - `["liver X receptors (LXRs) and thyroid hormone receptors (TRs)", "influence", "each other's transcriptional activity"]`
- **Classification:** Hallucinated

Since the extracted triplet contradicts the verified reference triplets, it is labeled as hallucinated. This evaluation is conducted independently for each extracted triplet, enabling a detailed analysis of the model's hallucination detection capabilities.

4.2 Evaluation Metrics

To quantify the effectiveness of our hallucination detection framework, we introduce two primary performance metrics: **Hallucination Detection Performance**-sensitivity and **Fact Preservation Performance**-specificity. These metrics provide complementary insights into the model's ability to accurately identify hallucinated triplets while preserving valid factual statements.

Rather than traditional precision and recall, we adopted sensitivity and specificity under the assumption of an application context where undetected hallucinations (false negatives) pose significant risks. Given this scenario, sensitivity directly aligns with our core objective—minimizing the occurrence of undetected hallucinations. Conversely, specificity evaluates the extent to which supported (non-hallucinated) triplets are correctly preserved, reflecting a secondary yet valuable objective of retaining factual accuracy. This choice of metrics thus reflects a deliberate, scenario-driven design decision established at the project's outset and guided subsequent model optimization efforts.

(1) Hallucination Detection Performance This metric measures the model's effectiveness in identifying hallucinated triplets and minimizing missed hallucination. It is evaluated using sensitivity, which quantifies the proportion of actual hallucinated triplets that are correctly flagged by the model.

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

where:

- TP (True Positive): The number of hallucinated triplets correctly identified as hallucinated.
- FN (False Negative): The number of hallucinated triplets classified as supported.

A high sensitivity score indicates that the model effectively detects hallucinated triplets, reducing the risk of unverified or misleading information remaining undetected.

(2) Fact Preservation Performance This metric evaluates the model's ability to correctly classify factual triplets as supported, thereby preventing valid information from being mistakenly flagged as hallucinated. It is measured using specificity, which determines the proportion of triplets predicted as supported that are truly not hallucinated.

$$\text{Specificity} = \frac{TN}{TN + FP}$$

where:

- TN (True Negative): The number of supported triplets correctly identified as supported.
- FP (False Positive): The number of supported triplets incorrectly classified as hallucinated.

A high specificity score ensures that the model maintains factual integrity by preserving legitimate triplets.

By balancing hallucination detection performance (sensitivity) and fact preservation performance (specificity), our evaluation framework provides a comprehensive assessment of the model's ability to detect hallucinations while minimizing unnecessary filtering of valid information. This dual-metric approach ensures that the model is both sensitive to factual inconsistencies and robust in maintaining accurate knowledge representation.

4.3 Dataset

Evaluating model performance on hallucination detection requires a carefully selected dataset that provides reliable reference contexts and well-defined ground truths. Considering this, we selected the BioASQ dataset, a reputable biomedical question-answering benchmark containing structured questions, verified answers, and corresponding reference documents from expert-curated biomedical literature. For our experiments, the BioASQ dataset served as the original source from which we systematically generated hallucination data using a specialized pipeline, described in detail in Section 4.4. Here, we first introduce the BioASQ dataset, emphasizing its structure, reliability, and suitability as a foundational resource for realistic and controlled hallucination evaluation experiments.

BioASQ Dataset

BioASQ [TBM+15] is a large-scale biomedical question-answering benchmark designed to advance research in information retrieval and natural language processing. It is part of the BioASQ challenge series [TBM+15], which provides a standardized dataset for evaluating biomedical question-answering models. The dataset is constructed using high-quality biomedical literature sources such as PubMed and MEDLINE, ensuring its reliability for scientific applications.

The BioASQ dataset is curated by biomedical experts to include a diverse range of question types, covering topics such as genetics, molecular biology, diseases, and treatments. Each question is accompanied by expert-verified answers, categorized into factoid, list, and summary (ideal) answers. Additionally, relevant reference passages from biomedical articles are linked to each question to provide supporting evidence.

For our experiments, we used a structured subset of the BioASQ dataset available on HuggingFace (`rag-datasets/rag-mini-bioasq`[1]) as the original source for generating hallucinated data. This subset consists of two main components: a *question-answer-passages* dataset and a *text-corpus* dataset. The *question-answer-passages* dataset contains columns for `question`, `answer`, `relevant_passage_id`, and `id`, while the *text-corpus* dataset consists of `passage` and `id` columns. By linking relevant passages from the text corpus using their IDs, this structured subset provides clear question-answer pairs along with corresponding reference documents.

In addition to providing biomedical question-answer pairs, the dataset enables evaluation across different types of relationships between reference documents and their corresponding answers. These relationships include:

1. **Detailed Information Extraction**: Extracting precise details from the source material to ensure the accuracy and reliability of the provided answer.

[1] https://huggingface.co/datasets/rag-datasets/rag-mini-bioasq

- **Example Question:** "Which genes does thyroid hormone receptor beta1 regulate in the liver?"
- **Example Answer:** "Thyroid hormone receptor beta1 (TR-beta1) upregulates ChREBP expression by interacting with LXRE2, thereby influencing T3-induced hepatic lipogenesis."
- **Reference Text:** "Thyroid hormone (T3) and TR-beta1 upregulate ChREBP by binding direct repeat-4 elements (LXRE1/2)."
- **Related Answer Triplet:** ["Thyroid hormone (T3) and TR-beta1", "upregulate", "ChREBP"]

2. **Summarization**: Assessing whether the answer effectively conveys the main ideas from the reference text while avoiding the inclusion of unsupported information.

   - **Example Question:** "How does thyroid hormone receptor beta1 influence hepatic lipogenesis?"
   - **Example Answer:** "Thyroid hormone receptor beta1 regulates hepatic lipogenesis by influencing ChREBP and SREBP-1c."
   - **Reference Text:** "The carbohydrate response element-binding protein (ChREBP) and sterol response element-binding protein (SREBP)-1c, regulated by liver X receptors (LXRs), play central roles in hepatic lipogenesis."
   - **Related Answer Triplet:** ["TR-beta1", "regulates", "hepatic lipogenesis through ChREBP and SREBP-1c"]

3. **Inference**: Evaluating the model's ability to derive logical conclusions from the reference text and accurately infer indirect relationships.

   - **Example Question:** "What indirect effects does TR-beta1 have on metabolic gene regulation?"
   - **Example Answer:** "TR-beta1 indirectly regulates metabolic genes, including P450R, through its role in hepatic lipogenesis."
   - **Reference Text:** "TR-beta1 influences T3-induced hepatic lipogenesis, which plays a role in metabolic regulation."
   - **Related Answer Triplet:** ["TR-beta1", "indirectly regulates", "other metabolic genes via hepatic lipogenesis"]
   - **Hallucinated Triplet:** ["TR-beta1"s regulatory role", "extends to", "other metabolic genes, including P450R"]

These different reference-answer relationships provide a foundation for evaluating the model's ability to handle varying levels of complexity in biomedical question answering.

4.4 Hallucinated Triplet Generator

To systematically evaluate our model's capability to detect hallucinations, we developed a triplet-based hallucination generator that creates structured hallucination samples derived from the original reference texts and associated questions. Using this generator, we constructed a dedicated hallucinated triplet dataset specifically for our experiments[2]. The carefully curated hallucinated triplets allowed us to perform a precise evaluation of model performance by explicitly distinguishing factual content from hallucinated content at the triplet level.

The hallucinated triplet generation process consists of three main steps. First, we generate a structured hallucinated sample consisting of three distinct components using a large language model (LLM) based on a given question and its corresponding reference triplets. Next, we convert the hallucinated answer into structured triplets using our triplet generator. Finally, we compare these extracted triplets with the non-hallucinated reference triplets to identify and index hallucinated triplets.

Step 1: Structured Hallucinated Sample Generation Given a question and its corresponding reference triplets, we generate a structured hallucinated sample consisting of the following three components:

- **Hallucinated Answer**: A response that introduces fabricated, incorrect, or unsupported information.
- **Non-Hallucinated Answer**: A response that strictly adheres to the reference triplets without any additional or incorrect information.
- **Hallucination Description**: A structured explanation detailing the specific hallucinated elements present in the hallucinated answer.

For example, consider the following structured hallucinated sample:

1. **Hallucinated Answer:**

   "Thyroid hormone receptor beta1 (TR-beta1) not only upregulates ChREBP but also directly regulates SREBP-1c and PGC-1alpha, leading to widespread effects on hepatic metabolism."

2. **Non-Hallucinated Answer:**

   "Thyroid hormone receptor beta1 (TR-beta1) upregulates ChREBP expression by interacting with LXRE2, thereby influencing T3-induced hepatic lipogenesis."

[2]The created dataset is available at https://github.com/KRLabsOrg/RAGFactChecker.

3. **Hallucination Description:**

   "Thyroid hormone receptor beta1 (TR-beta1) not only upregulates ChREBP but also directly regulates SREBP-1c and PGC-1alpha,"

Step 2: Triplet Extraction Using Triplet Generator Once the hallucinated sample is generated, we extract triplets from the hallucinated answer using our triplet generator. This process converts the hallucinated answer into a structured (arg1, relation, arg2) format, ensuring that factual inconsistencies can be systematically analyzed.

For the above example, the extracted triplets from the hallucinated answer are:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'upregulates', 'ChREBP']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'SREBP-1c']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'PGC-1alpha']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'leads to', 'widespread effects on hepatic metabolism']

Similarly, the extracted triplets from the non-hallucinated answer are:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'upregulates', 'ChREBP expression']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'interacts with', 'LXRE2']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'influences', 'T3-induced hepatic lipogenesis']

Step 3: Hallucinated Triplet Identification and Indexing After extracting triplets from both the hallucinated and non-hallucinated answers, we compare them to identify hallucinated triplets. Any triplet that appears exclusively in the hallucinated answer but is absent from the non-hallucinated answer is classified as a hallucinated triplet.

In our example, the hallucinated triplet is:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'PGC-1alpha']

This process ensures that hallucinated triplets are systematically indexed for evaluation, allowing for a more granular analysis of hallucination detection performance.

Manual Verification To further enhance dataset quality, we incorporate an optional manual verification process. While this step can be omitted in fully automated settings, we apply it specifically for rigorous experimentation to ensure results closely aligned with factual accuracy. Human annotators review the generated triplets to ensure correctness, independent of the model's operation. This manual annotation serves as an additional validation layer to prevent systematic errors. Since hallucination detection is a challenging task, verifying the dataset before model evaluation ensures that detected hallucinations are indeed hallucinations and not misclassifications caused by improper dataset construction.

The final hallucinated dataset, constructed through this procedure, is then employed to test the model's hallucination detection performance under controlled conditions.

4.5 Generation of Challenging Hallucination Types

In addition to the systematic extraction and indexing of hallucinated triplets, we further refine our dataset by deliberately incorporating hallucination types that are known to be difficult to detect. Prior work [GVM22] primarily focused on hallucinations that are easier to identify; however, our objective is to challenge the model with more subtle deviations. To this end, we introduce three distinct hallucination types that maintain a close resemblance to the reference content while embedding critical factual deviations: *Absolute Fabrication*, *Contextual Fabrication*, and *Detailed Information Modification*. Each type is designed to probe a specific aspect of hallucination detection, particularly in domains where even minor factual discrepancies can have serious consequences (e.g., medicine, research, finance).

Absolute Fabrication
Definition: Absolute Fabrication occurs when the hallucinated answer introduces entirely new and unsupported information that has no grounding in the reference material. Although the generated content adheres to the overall structural format (i.e., a triplet), the fabricated element is completely alien to the established facts. While this type of hallucination might be comparatively easier to detect than others, we carefully constructed the fabricated elements to closely align with the context of the reference material, thereby increasing the difficulty of detection.

Example: Consider a reference consisting of the following triplets:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'upregulates', 'ChREBP']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'SREBP-1c']

An absolute fabrication might add:

- ["TR-beta1", "was found to improve", "cardiac output by approximately 15% in patients with heart failure in a 2022 study"]

Here, the introduction of insulin is a clear deviation from biological fact, as insulin is known to be produced by the pancreas rather than the thyroid gland.

Contextual Fabrication
Definition: Contextual Fabrication involves the insertion of hallucinated details that, while contextually related to the reference, subtly misrepresent the factual content. This type of hallucination blends elements of the original context with newly introduced, yet misleading, information.

Example: Given the reference:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'upregulates', 'ChREBP']
- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'SREBP-1c']

A contextual fabrication might generate:

- ['Thyroid hormone receptor beta1 (TR-beta1)', 'directly regulates', 'PGC-1alpha']

Although TSH is a hormone involved in the endocrine system, its production is factually associated with the pituitary gland. This example challenges the model to recognize the nuanced difference between contextually plausible but factually incorrect associations.

Detailed Information Modification
Definition: Detailed Information Modification is characterized by minor yet critical alterations to specific details—such as numerical values, dates, or object attributes—that can significantly alter the factual interpretation. This hallucination type closely mirrors the reference content, with only slight modifications that may lead to substantial discrepancies. in some domain(medical, financial, academia), checking numerical values when fact checking is crucial, however there are less study which tested this type of hallucination alone. and There are questions about how language model interpretes numerical values [LG24, HTYZ24]. it is questionable that that the model could check hallucination even in a little numerical change. so we test this

Example: If a reference triplet states:

- ["Thyroid hormone receptor beta1 (TR-beta1)", "increase", "ChREBP expression by 30%"]

a hallucinated version via detailed information modification might present:

- ["Thyroid hormone receptor beta1 (TR-beta1)", "increase", "ChREBP expression by 32%"]

Even though the difference is numerically minor, such discrepancies can be critical in domains where precision is paramount.

Unified Objective Each hallucination introduced is intentionally designed to produce content that is closely related to the reference material. This similarity ensures that the hallucinations are not trivially obvious but rather require detailed, triplet-level analysis to be detected. By targeting these three challenging hallucination types—Absolute Fabrication, Contextual Fabrication, and Detailed Information Modification—we create a robust testbed for evaluating the sensitivity and accuracy of our hallucination detection model, particularly in high-stakes environments where even subtle factual deviations can lead to significant consequences.

4.6 Experiment Conditions

To thoroughly evaluate the effectiveness of our hallucination detection approach, we conduct experiments under various conditions. A primary focus is assessing the impact of structured prompt engineering on hallucination detection performance. By comparing models with and without optimized prompts, we analyze whether different prompting techniques contribute to improved factual consistency and hallucination detection accuracy.

Our prompts are designed to include multiple structured components, each playing a critical role in guiding the model's reasoning and validation process. The key components of our prompt structure include:

- **Few-shot Examples:** Demonstrative examples illustrate the expected reasoning process and output format, helping the model learn from structured cases and align its responses with predefined patterns. This improves both format consistency and factual accuracy [TLI+23]. In this study, we use two few-shot examples for our experiments. While increasing the number of examples typically enhances performance, we limit our selection to two due to the substantial length of each example. [TLI+23]. In this research we used 2 Few-shot samples for experiment. More few shot samples usually improve performance [BMR+20], however, because our few-shot sample is extremely long, we only use two samples.

- **Chain-of-Thought (CoT) Reasoning:** The model is explicitly instructed to articulate intermediate reasoning steps before delivering its final verdict on whether a triplet is supported. This enhances interpretability, reduces logical inconsistencies, and allows users to verify the rationale behind each classification [WWS+22].

- **Detailed Instructions:** Clear and explicit guidelines are included in the prompt to standardize model responses. These instructions specify output formatting, validation criteria, and logical constraints, ensuring the model follows a structured and objective assessment process [QWJ+22].

To measure the effectiveness of these structured prompting techniques, we compare different prompting methods such as few-shot generation, chain-of-thought, and detailed instructions. While incorporating all components within a single prompt may seem ideal, excessively long prompts may introduce context overload, degrade model performance, and increase computational cost. Thus, we conduct an ablation study by selectively removing certain components (e.g., excluding CoT reasoning or few-shot examples) to evaluate their individual contributions.

In addition to prompt variations, we investigate how different types of hallucinations, Absolute Fabrication, Contextual Fabrication, and Detailed information modification affect detection performance. The detailed explanation of hallucination types are stated in 4.4 Hallucinated Triplet Generator

By systematically evaluating the model's performance across different prompt structures and hallucination types, we aim to identify optimal configurations for hallucination detection. These experiments provide deeper insights into how structured prompts influence factual validation and how hallucination types affect detection reliability.

4.7 Implementation Details

We follow the experimental setup of the most relevant studies for evaluation. In line with previous research [YV23, HRQ+24], all evaluations in this work are conducted at the triplet level for hallucination detection. However, unlike previous methods, we compare answer triplets and reference triplets collectively rather than on a one-to-one basis.

To ensure robust evaluation, we implement several technical optimizations. One key aspect is prompt splitting, as large reference triplet sets can degrade LLM performance due to context length limitations. To mitigate this issue, we split extensive reference sets into smaller context windows, allowing for sequential processing by the model. This approach prevents performance degradation and ensures more effective fact verification.

We use GPT-4o as the primary model for hallucination detection experiments. Additionally, we evaluate a lighter variant, GPT-4o-mini, which demonstrates lower performance in fact verification tasks. To ensure reproducibility, we set the temperature parameter to zero, thereby enforcing deterministic outputs. Other hyperparameters, including max tokens and top-p, are also fixed to maintain consistency across experimental runs.

Since hallucination detection relies on extracting structured outputs from LLM-generated responses, we enforce a fixed output format to prevent parsing errors. However, certain cases result in unexpected or malformed outputs, leading to erroneous triplet extractions.

To address this, we employ an error detection and mitigation strategy. First, we identify and filter common error patterns within LLM-generated responses. If an output contains errors, we re-request a corrected response up to a predefined threshold. If errors persist beyond this threshold, we log them for further analysis to identify configurations that contribute to systematic failures.

By implementing these measures, we ensure that our model evaluations remain accurate, reproducible, and free from system-induced artifacts. This structured approach enables us to rigorously assess the effectiveness of hallucination detection across different prompting strategies, hallucination types, and dataset variations.

Lastly, we follow the experimental setup of the most relevant studies for the evaluation. In line with previous research [YV23, HRQ+24], all evaluations in this work are conducted at the triplet level for hallucination detection.

CHAPTER 5

Results and Discussion

5.1 Hallucinated Data

To evaluate the nature of hallucinated outputs, we first analyze the hallucinated data generated by our methodology[1]. This includes the types of hallucinations identified, descriptive statistics, and a breakdown of how different types manifest in the dataset.

From a total of 60 samples derived from questions containing the keyword 'Thyroid,' we generated 731 triplets. Among these, 121 triplets were designated as hallucinated, comprising approximately 16.6% of the dataset. This hallucination rate aligns with that reported in QA tasks using LLMs in the biomedical domain [LCR+24], indicating that our results are representative of real-world conditions.

- **Detailed Information Modification**: 46
- **Absolute Fabrication**: 42
- **Contextual Fabrication**: 32

Just as intended, the model produced only the three specified types of hallucinations. Representative examples for each type are provided below:

Detailed Information Modification: Hallucinated Triplet:

```
[ "loss of heterozygosity (LOH) at the PTEN locus", "occurs
in exactly", "30% of follicular thyroid tumors" ]
```

Reference Triplet (if exists):

[1] The dataset is available at https://github.com/KRLabsOrg/RAGFactChecker.

```
[ "loss of heterozygosity (LOH) at the PTEN locus", "occurs
in approximately", "25% of follicular thyroid tumors" ]
```

In this instance, the hallucinated output alters a critical numerical detail while maintaining structural similarity to the reference, exemplifying a subtle yet potentially significant modification.

Absolute Fabrication: Original Triplets:

```
[ "Recent studies", "have identified", "patients with inactivating
mutations in TRβ1" ]
[ "These cases", "are distinct and do not represent", "the
typical RTH phenotype characterized by TRβ2 mutations" ]
```

Added (Hallucinated) Triplet:

```
[ "A small cohort of patients in 2022", "exhibited", "symptoms
resembling RTH" ]
```

Here, the hallucinated triplet introduces entirely new content that is not supported by the reference, thereby challenging the detection system with a clear-cut instance of fabricated information.

Contextual Fabrication: Original Triplet:

```
[ "PTEN's expression", "is often silenced through", "various
epigenetic mechanisms" ]
```

Added (Hallucinated) Triplet:

```
[ "complete loss of PTEN mRNA expression", "is evident in",
"up to 8% of analyzed tumors" ]
```

This example demonstrates a hallucination where additional context is provided that is closely related to the reference, yet introduces subtle factual inaccuracies.

Example Analysis and Discussion The examples above clearly illustrate how our methodology generates hallucinated triplets in accordance with the predefined categories. In the case of Detailed Information Modification, the hallucinated triplet deviates from the reference by altering a numerical value, a modification that is particularly challenging for detection systems given its structural similarity to the original content. The Absolute Fabrication example introduces entirely new content that is unsupported by any reference information, while the Contextual Fabrication example subtly adjusts the context to include an inaccurate statistic.

Notably, the hallucination data is generated in a well-balanced manner. The nearly uniform distribution across the three types—46 instances of Detailed Information Modification, 42 instances of Absolute Fabrication, and 29 instances of Contextual Fabrication—demonstrates that our methodology reliably produces diverse and nuanced hallucination examples. This balanced generation is critical, as it ensures that subsequent evaluations of hallucination detection are robust and comprehensive, effectively challenging the model to detect both overt and subtle deviations from the reference information.

These results confirm that our approach not only generates hallucinated content systematically but also maintains a controlled and varied distribution of hallucination types. This lays a solid foundation for evaluating and improving hallucination detection mechanisms in high-stakes domains such as medicine, research, and finance.

5.2 Hallucination Detection Performance

To assess the effectiveness of our hallucination detection framework, we conducted extensive evaluations using multiple configurations and keyword-specific test sets. Our experiments reveal that our approach is capable of accurately identifying hallucinated triplets across diverse scenarios, demonstrating both robustness and generalizability.

Overall Performance: Our evaluation on the full dataset demonstrates that our framework achieves strong performance when incorporating detailed triplet analysis, chain-of-thought (CoT) reasoning, and persona-based contextualization. Notably, the `detail + CoT` configuration yielded a sensitivity score of 0.8833 and a specificity score of 0.8179.

For comparison, we evaluated the most relevant baseline, REFCHECKER [HRQ+24], on our dataset. To precisely replicate REFCHECKER's evaluation procedure, we followed the original prompt and verification method described in their paper, in which the hallucination detection process involves inserting each individual answer triplet and its corresponding reference texts(not reference triplets) into their pre-defined prompt (see Appendix C for detailed prompt example). Here, each request evaluates only one answer triplet at a time. The results showed perfect sensitivity of 1.00 but a significantly lower specificity of 0.2379. While REFCHECKER effectively avoids false positives, the notably low specificity indicates it fails to identify more than three-fourths of correct knowledge

claims, resulting in a strong bias toward classifying triplets as hallucinations. Importantly, this performance gap persisted even when testing REFCHECKER strictly under its original evaluation condition, where only one answer triplet is processed per request.

These results—summarized in Table 5.1—highlight the trade-offs between sensitivity and specificity in hallucination detection, demonstrating the advantages of our approach in achieving a well-balanced performance while maintaining computational efficiency. Notably, our method excels in overall balanced accuracy (BA) score[2]. We hypothesize that the hallucinations identified in our evaluation are more subtle and require a careful, nuanced approach for accurate differentiation.

Performance by Keyword: To further validate the robustness of our framework, we tested its performance on keyword-specific subsets that serve as proxies for different contexts:

- **Hormone:** For samples involving the keyword "Hormone," the model achieved performance scores of 0.8657 and 0.8702. This high level of performance demonstrates the framework's capacity to handle biological and medical terminology reliably.
- **RNA:** For samples centered around "RNA," the scores were 0.8889 and 0.8166. Despite the shift in domain focus, the performance remains consistently high, underscoring the generalizability of our approach across various subject areas.

Configuration/Keyword	TP	FP	FN	TN	Sensitivity	Specificity
Thyroid	106	108	14	485	0.8833	0.8179
Hormone	58	58	9	389	0.8657	0.8702
RNA	208	265	26	1180	0.8889	0.8166

Table 5.1: Performance Metrics for Hallucination Detection

The results indicate that our hallucination detection framework not only performs well on the overall dataset but also exhibits balanced and robust detection capabilities across different data. The high performance across keywords suggests that the hallucinated data, generated using our controlled methodology, is well-distributed and representative of real-world challenges. This balanced distribution is crucial for ensuring that our evaluation framework accurately reflects both overt and subtle forms of hallucinated content.

Table 5.1 summarizes these results and further illustrates the consistency of our detection performance across various experimental settings.

[2]Balanced accuracy (BA) is defined as the arithmetic mean of sensitivity and specificity: (sensitivity + specificity) / 2.

Performance by Prompting Methods: We evaluated the impact of different prompting strategies on hallucination detection performance by systematically varying prompt components. Here, the prompt feature *Baseline* refers to prompts that contain standard elements, including a brief task description and predefined classification categories, following the structure of REFCHECKER [HRQ+24]. The results in Table 5.2 reveal several key trends:

Prompt Feature	TP	FP	FN	TN	Sensitivity	Specificity	BA
REFCHECKER [HRQ+24]	120	466	0	145	1.0000	0.2373	0.6187
Baseline	55	31	66	579	0.4545	0.9492	0.7019
CoT	29	6	65	477	0.3085	0.9876	0.6481
Few-shot	47	47	74	563	0.3884	0.9230	0.6557
Detail	110	224	10	386	0.9091	0.6328	0.7710
Detail + CoT	106	108	14	485	0.8833	0.8179	0.8506
Detail + Few-shot	98	125	23	485	0.8099	0.7951	0.8025
Detail + CoT + Few-shot	99	108	21	503	0.8250	0.8232	0.8241

Table 5.2: Performance of different prompting methods in hallucination detection.

First, detailed instruction alone (`Detail`) was the most effective individual prompt feature, achieving a high specificity of 0.9091 and an BA score of 0.7710. However, its specificity was relatively low (0.6328), indicating that while it helped reduce false negatives, it missed a significant portion of true knowledge claims. This suggests that detailed, structured instructions improve factual accuracy but may require additional strategies to improve specificity.

Second, few-shot prompting alone (`Few-shot`), which provides in-context examples, did not significantly improve performance on its own, yielding an BA score of 0.6557. However, when combined with detailed instructions (`Detail + Few-shot`), BA increased to 0.8025, and specificity also improved. This implies that few-shot examples enhance fact preservation when paired with structured guidance, allowing the model to recognize factual consistency more effectively.

Third, Chain-of-Thought (CoT) reasoning alone (`CoT`) resulted in high specificity (0.9876) but very low sensitivity (0.3085), leading to an BA score of 0.6481. This suggests that while CoT reasoning expands the model's ability to retrieve more factual knowledge, it also increases false negatives. However, when CoT was combined with detailed instructions (`Detail + CoT`), sensitivity improved dramatically (0.8833), specificity remained high (0.8179), and BA reached 0.8506, making it one of the most balanced configurations. This implies that CoT enhances knowledge retrieval but requires structured guidance to maintain sensitivity.

Finally, the results from applying all prompt features—`Detail + CoT + Few-shot`—showed an BA score of 0.8241, with balanced sensitivity (0.8250) and specificity (0.8232). We hypothesize that the increased prompt length resulting from the inclusion of few-shot

examples may have adversely affected model performance. Although incorporating additional prompt features has the potential to enhance performance, the corresponding increase in prompt length could lead to diminishing marginal benefits or even a decline in overall effectiveness.

These findings emphasize that structured prompts alone can significantly improve sensitivity but require additional methods to enhance specificity. CoT and few-shot examples independently provide little benefit, but when combined with structured prompts, they create a well-balanced framework for hallucination detection. This suggests that hallucination detection systems should integrate multiple prompting techniques rather than relying on a single approach for optimal performance.

5.3 Performance by Hallucination Type

To further investigate the strengths and limitations of our hallucination detection framework, we analyzed its performance separately on each hallucination type. Table 5.3 summarizes the detection results across the three types of hallucinations generated: Detailed Information Modification, Absolute Fabrication, and Contextual Fabrication.

Hallucination Type	**Detection Rate**
Detailed Information Modification	0.8478
Absolute Fabrication	0.9047
Contextual Fabrication	0.9063

Table 5.3: Detection Performance by Hallucination Type

As shown in Table 5.3, our model achieves a detection rate of 0.8478 for Detailed Information Modification, which is notably lower than the performance for Absolute Fabrication (0.9047) and Contextual Fabrication (0.9063). This indicates that while our framework is highly effective at detecting hallucinations that involve entirely fabricated or contextually misleading information, it faces more challenges when the hallucination involves subtle modifications of detailed information (e.g., minor numerical changes or slight descriptive alterations).

Unlike previous research [GVM22]—which primarily focused on broader categories of hallucinations—the results presented here show that the detection rates for Absolute Fabrication and Contextual Fabrication are quite similar. The introduction of Detailed Information Modification, a more nuanced form of hallucination, reveals a lower detection performance, suggesting that minor yet critical deviations require further refinement in our detection approach.

These findings underscore that our detection framework achieves a well-balanced performance across different types of hallucinations. The results demonstrate that our method is particularly effective at identifying both overt fabrications and contextually misleading claims, while still performing well on more subtle modifications, which are traditionally

harder to detect. Furthermore, the strong performance across various hallucination types suggests that our approach generalizes well beyond specific cases.

CHAPTER 6

Conclusion

6.1 Key Findings and Contributions

In this study, we proposed a novel framework for hallucination detection in large language models (LLMs), focusing on knowledge-level validation through structured triplet extraction and comparison. Our approach decomposes text into discrete factual units—(arg1, relation, arg2) triplets—allowing for precise detection of hallucinated information at a finer granularity than conventional sentence- or sub-sentence-level methods. By leveraging LLMs for triplet fact-checking in a single request, our method eliminates the need for complex multi-step pipelines or multiple queries in the fact verification process, thereby significantly enhancing efficiency. The results of this approach are packaged in a python (pip) installable package [1].

To evaluate the effectiveness of our approach, we conducted hallucination detection experiments, specifically targeting challenging hallucination cases: Absolute Fabrication, Contextual Fabrication, and Detailed Information Modification. Using the BioASQ Dataset and our Hallucinated Data Generator, we systematically generated hallucinated triplets corresponding to these hallucination types.

Results demonstrate that our method achieves a strong balance between sensitivity (0.8833) and specificity (0.8179), significantly outperforming the previous state-of-the-art knowledge-level detection approach [HRQ+24]. Compared to existing triplet-based verification models, our framework not only enhances hallucination detection accuracy but also drastically reduces the number of verification requests. Specifically, our method requires $\frac{\text{average length of input triplet}}{\text{(input sequence length - C)}}$ times fewer requests, leading to a significant improvement in efficiency.

Additionally, our experiments highlight the critical role of prompting techniques in enhancing hallucination detection. Structured and detailed instructions significantly improve

[1] The package source code is available at https://github.com/KRLabsOrg/RAGFactChecker.

factual accuracy, while few-shot examples and chain-of-thought reasoning contribute to better specificity. Our best-performing prompt combined detailed instructions with chain-of-thought reasoning, achieving 0.8833 sensitivity, 0.8179 specificity, and an BA score of 0.8444. Interestingly, adding few-shot examples slightly decreased performance, yielding 0.8250 sensitivity, 0.8232 specificity, and an BA score of 0.8241. We hypothesize that the additional length introduced by few-shot examples may have negatively impacted model performance. Furthermore, while additional prompt features could enhance performance, they also increase total prompt length, which may introduce diminishing returns or even performance degradation.

6.2 Limitations

Despite its strong performance, our approach has several limitations that warrant further investigation. First, the triplet extraction process is inherently dependent on the capabilities of the LLM used. Variations in LLM outputs, particularly in paraphrased or contextually implied relationships, can impact detection accuracy. Second, while our method effectively detects most hallucination types, it exhibits slightly lower performance in identifying Detailed Information Modifications—hallucinations involving minor yet critical numerical or descriptive changes. This suggests that additional refinements are needed to improve sensitivity to subtle factual inconsistencies.

Another limitation is the reliance on reference triplets derived from the same source as the generated text. While this approach ensures alignment between reference and generated content, it restricts the scope of verification to the information present in the dataset. The effectiveness of our method in detecting hallucinations across broader knowledge sources (e.g., external knowledge graphs) remains an open question.

6.3 Future Work

Future research directions include several key improvements and expansions to our hallucination detection framework. One promising avenue is the incorporation of external knowledge sources—such as structured knowledge graphs or domain-specific databases—to enhance reference triplet generation. By integrating these external sources, we aim to improve hallucination detection accuracy, particularly in cases where the LLM-generated content lacks clear reference alignment.

Additionally, further optimization of prompt engineering strategies could enhance the reliability of our approach. Exploring alternative CoT formulations, retrieval-augmented generation (RAG) methods, or multi-step verification processes may help mitigate false positives and improve handling of difficult-to-detect hallucination types.

Lastly, future studies should examine the applicability of our framework across different LLM architectures, including smaller, more efficient models designed for real-time

applications. Evaluating how our approach performs across multiple LLM families will provide insights into its scalability and adaptability to various computational constraints.

By addressing these areas, we aim to further refine our hallucination detection methodology, making it more effective, adaptable, and practical for real-world deployment in AI-driven information retrieval and decision-making systems.

APPENDIX A

Appendix

A.1 Fact Checker Prompts

This appendix provides the system and user prompts used for fact-checking triplets in various experimental settings.

A.1.1 Baseline Prompt

System Prompt:

```
You are an assistant responsible for verifying whether each
input triplet is supported by the source triplets.  For each
triplet in the input triplets, determine whether there is a
similar triplet in the source triplets or whether the input
triplet can be logically inferred from the source triplets.

Consider paraphrased information, contextual clues (e.g.,
pronouns or synonyms), and combinations of source triplet
information to make this determination.

Provide the results in the following format:
triplet_idx:result

Where result can be one of the following:
- True:  The input triplet is either highly similar to a triplet
in the source or can be logically inferred from the source
triplets.
-   False:  The input triplet cannot be matched or inferred from
any triplet in the source.

Be concise and only output the results as specified.
```

User Prompt:

```
Task Description:
Compare the input triplets against the source triplets to
determine if each input triplet is either highly similar to a
source triplet or can be logically inferred from the source
triplets.
Consider paraphrasing, contextual changes, and indirect
references such as pronouns or synonyms.
Output True if the triplet matches or is inferable; otherwise,
output False.

Input Triplets:
{answer_triplets}

Source Triplets:
{reference_triplets}
```

A.1.2 Few-shot Prompt

System Prompt:

```
You are an assistant responsible for verifying whether each
input triplet is supported by the source triplets.
For each triplet in the input triplets, determine whether
there is a similar triplet in the source triplets or whether
the input triplet can be logically inferred from the source
triplets.
Consider paraphrased information, contextual clues (e.g.,
pronouns or synonyms), and combinations of source triplet
information to make this determination.

If few-shot demonstrations are provided, carefully follow their
approach.  If they demonstrate caution and only assign True
when the evidence is indisputable, follow that pattern.

Provide the results in the following format:
triplet_idx:result
Where result can be one of the following:

- True:  The input triplet is either highly similar to a triplet
in the source or can be logically inferred from the source
triplets.
- False:  The input triplet cannot be matched or inferred from
any triplet in the source.

Be concise and only output the results as specified.
```

User Prompt:

```
Task Description:
Compare the input triplets against the source triplets to
determine if each input triplet is either highly similar to a
source triplet or can be logically inferred from the source
triplets.
Consider paraphrasing, contextual changes, and indirect
references such as pronouns or synonyms.
Output True if the triplet matches or is inferable; otherwise,
output False.

(Optional)
If few-shot examples are provided here, they will look like
this:

[BEGIN FEW-SHOT-EXAMPLES]
<Example 1 Input/Output Pair>
<Example 2 Input/Output Pair>
...
[END FEW-SHOT-EXAMPLES]
If these examples are present, incorporate their style and
approach into your solution.
{examples}

Input Triplets:
{answer_triplets}

Source Triplets:
{reference_triplets}
```

A.1.3 Chain-of-Thought (CoT) Prompt

System Prompt:

```
You are an assistant responsible for verifying whether each
input triplet is supported by the source triplets.
For each triplet in the input triplets, determine whether
there is a similar triplet in the source triplets or whether
the input triplet can be logically inferred from the source
triplets.
Consider paraphrased information, contextual clues (e.g.,
pronouns or synonyms), and combinations of source triplet
information to make this determination.

Your final output must contain exactly two sections in this
order:

[REFERRED TRIPLETS]
- For each input triplet, list the source triplets (by ID or
index) that support or contradict the input.
- Provide a short explanation (a brief chain-of-thought)
describing how they led you to choose True or False.
- Example:

triplet_idx_1:  (source triplets #1, #3) → [very brief
reasoning]
triplet_idx_2:  None
...

[FINAL ANSWER]
- For each input triplet, output one line in the format
triplet_idx:True or triplet_idx:False.
- No additional explanation or text beyond these lines.
```

User Prompt:

```
Task Description:
Compare the input triplets against the source triplets to
determine if each input triplet is either highly similar to a
source triplet or can be logically inferred from the source
triplets.  Consider paraphrasing, contextual changes, and
indirect references such as pronouns or synonyms.

Output Requirements:
1.  Provide only the two sections [REFERRED TRIPLETS] and
[FINAL ANSWER].
2.  Under [REFERRED TRIPLETS], for each input triplet, specify
which source triplets (if any) were used, along with a brief
explanation of how they support or contradict the input.
3.  Under [FINAL ANSWER], output exactly one line per input
triplet in the format:

triplet_idx:True
triplet_idx:False

4.  No further explanations or additional text should be
included outside these two sections.
5.  The number of lines in [FINAL ANSWER] must match the number
of input triplets.

Follow the revised system prompt carefully to decide True or
False for each input triplet.

Input Triplets:
{answer_triplets}
Source Triplets:
{reference_triplets}
```

A.1.4 Detailed Instruction Prompt

System Prompt:

```
You are an assistant responsible for verifying whether each
input triplet is supported by the source triplets.  For each
input triplet, you must decide whether it is True or False,
strictly based on the source triplets.  Follow these detailed
rules:

1.  True Condition
- Exact or Strictly Equivalent Match:  If the input triplet
directly quotes or very closely paraphrases the source with the
same meaning (including specific data, facts, or relationships),
choose True.
- Numeric Data, Names, Key Facts:  All specific numbers,
measurements, or details must match or be strictly equivalent
to the source.
- Locations, Timeframes, or Qualifiers:  Must be identical or
demonstrably the same.
- Straightforward Inference:  If it is logically clear from the
source triplets that the specific details in the input triplet
can be inferred without speculation or guesswork, mark True.
- Example:  If the source says, 'A hormone X is specifically
found in both the hippocampus and the cortex,' then 'hormone X
is found in the hippocampus' is a valid inference.
- But if the source is significantly more general or omits
critical details (e.g., only says 'several hormones' without
naming them), do not fill in any specifics on your own.
- Allowable Inferences:  If the input triplet's statements can
be derived by combining or interpreting information already in
the source, without speculation, guesswork, or introduction of
new details, choose True.
```

```
2. False Condition
- Unsupported or New Details: If the input triplet introduces
any detail (numeric value, location, name, or condition) that
the source triplets do not clearly confirm, choose False--even
if it is a real-world fact.
- Contradiction or Mismatch: If any part of the input triplet
conflicts with the source triplets, choose False (e.g.,
different numbers, different subject-object relationships, or
the source is more general while the input is overly specific).
- Speculation or Guessing: If you cannot directly verify the
triplet or logically deduce it from the source without making
an assumption or inference that is not clearly supported,
choose False.

3. Additional Rule for Exactness of Numeric or Specific
Details
- If the input triplet specifies a particular quantity, time
period, location, or other condition, confirm that the source
triplets match it exactly.
- Even a slight difference in numeric value or specific wording
means False if there is no explicit mention of a range or
approximation in the source.

4. Output Format
- For each input triplet, output one line in the format
triplet_idx:True or triplet_idx:False.
- No additional explanation or text beyond these lines.
- The number of lines here must match the number of input
triplets exactly.
```

```
5.  Instructions to Follow Carefully
1.  Compare each input triplet with the source triplets in
detail.
2.  Decide True or False using the above rules and be very
strict about numeric data, specific locations, times, or
qualifiers.
3.  List only triplet_idx:True or triplet_idx:False for each
input triplet.
```

User Prompt:

```
Task Description:
Compare each input triplet to the provided source triplets.
Following the revised system prompt instructions, determine
whether each input triplet is supported (True) or not supported
(False).

Key Reminders from the System Prompt:
- If the input triplet introduces details (numeric values,
specific conditions, or qualifiers) not explicitly supported by
the source triplets, you must mark it False, even if it might
be true in reality.
- If the input triplet has any mismatch in numbers, times,
measurements, or specificity beyond what the source triplets
state, mark it False.
- For detailed facts with numbers, partial or approximate
matches are insufficient; all details must exactly or
straightforwardly match.

Input Triplets:
{answer_triplets}

Source Triplets:
{reference_triplets}

Output Requirements:
1.  Output exactly one line per input triplet in the format:

triplet_idx:True
triplet_idx:False
```

```
2.  No further explanations or additional text should be
included outside the triplet_idx:True.
3.  The number of lines must match the number of input
triplets.

Follow the revised system prompt carefully to decide True or
False for each input triplet.
```

A.2 Prompts for Hallucinated Data Generation

A.2.1 Hallucinated Data Generator Prompt

System Prompt:

```
You are HallucinationDataGenerator, an assistant specialized
in creating subtle, plausible hallucinations within your
responses.
Your task is to generate answers that are primarily grounded
in the provided reference documents and directions, but also
incorporate carefully crafted, believable fictional elements.
These hallucinations should not be outlandish; instead, focus
on small details that could easily be overlooked--such as
specific years, dosage values, or timeframes.
For instance, you might slightly alter a reported year,
introduce a modest yet unverified numerical detail, or specify
a plausible interval for symptom onset that isn't explicitly
stated.
Ensure that the hallucinated details blend seamlessly with
the given context and do not contradict major facts in the
reference documents.
Maintain coherence, relevance, and credibility throughout your
response.

If few-shot demonstration examples are provided, use them
as a guide to understand the style, approach, and complexity
expected in the hallucinated output.
You may adopt a similar manner of integrating subtle fictional
details as demonstrated in the examples.
```

User Prompt:

```
Reference Document:
{reference_documents}

(Optional) Few-Shot Demonstrations:
If few-shot examples are provided here, they will look like
this:

[BEGIN FEW-SHOT-EXAMPLES]
<Example 1 Input/Output Pair>
<Example 2 Input/Output Pair>
...
[END FEW-SHOT-EXAMPLES]

If these examples are present, incorporate their style and
approach into your solution.
{examples}

Question:
{question}
Task:
1.  Non-Hallucinated Answer:
- Produce a comprehensive, evidence-based answer to the question
using the provided references.
- Include reasoning, background context, and supporting evidence
from the references, making sure the answer is not overly
brief.
```

```
2.  Hallucinated Answer:
- Start with the exact same text as the Non-Hallucinated Answer.
- Introduce subtle hallucinations that are small, credible,
and closely related to the context found in the references.
These hallucinations should be challenging to detect without
carefully checking the provided references.  For instance,
slightly alter a date, a name, a relationship between entities,
or introduce a minor detail that sounds plausible but does not
appear in the references.
- Highlight each hallucinated detail in the text (e.g., italics
or a parenthetical note).
- Apart from the hallucinated elements, the rest of
the Hallucinated Answer should remain identical to the
Non-Hallucinated Answer.

3.  Hallucinated Details Section:
- After the Hallucinated Answer, list each hallucinated fact as
a separate bullet point under a 'Hallucinated Details' heading,
clearly identifying the fabricated elements.
Format Example:

Non-Hallucinated Answer:
[Comprehensive, evidence-based answer here, with no
hallucinations]

Hallucinated Answer:
[Identical to Non-Hallucinated Answer except where subtle,
contextually plausible hallucinated details are introduced and
highlighted]

Hallucinated Details:
- [List each hallucinated fact here as a bullet point]
```

A.2.2 Hallucinated Triplet Extraction Prompt

System Prompt:

```
You are an advanced language model trained to analyze textual
information for hallucination detection.  This time, you
will not reference or compare non_hallucinated_triplets.
Instead, you will work directly from the provided answer
and hallucinated_answer to determine which triplets in
answer_triplets are hallucinated.

Your primary goal is to produce output lists that match
the exact number of triplets in answer_triplets.  Under no
circumstances should you produce more or fewer boolean values
than the number of provided triplets.

Specifically, you will:
1.  Consider the hallucinated_answer and the original answer.
2.  Identify hallucinated triplets among answer_triplets by
determining which details appear in the hallucinated_answer but
are not supported by the information in the original answer.
3.  Output two boolean lists--both having the exact same length
as the answer_triplets list--where:
- One boolean list includes comments explaining the reasoning
for each corresponding triplet.
- The other boolean list includes no comments.
In both lists:
- false indicates the triplet is supported by the original
answer (not hallucinated).
- true indicates the triplet is hallucinated (introduces
unsupported or fabricated details).
```

```
4.  There must be a one-to-one correspondence between the
triplets and the boolean values in both lists.  For example, if
there are 4 triplets, you must produce exactly 4 boolean values
in the commented list and exactly 4 boolean values in the plain
list.

5.  Provide only the boolean lists (and optional comments for
one version) unless the user requests additional details.

Failure to match the exact number of boolean values to the
number of answer_triplets means you have not followed the
instructions correctly.  Make sure to count the answer_triplets
and produce the exact same number of boolean values.

Follow the format and instructions given in the user prompt.
```

User Prompt:

```
Here are the inputs for hallucination detection:
1. Original Answer (Used to generate supported triplets):
{answer}
2. Hallucinated Answer (Source of potential hallucinations):
{hallucinated_answer}
3. Answer Triplets (Extracted from the hallucinated answer):
{answer_triplets}

Important: The number of boolean values you provide must match
the number of triplets in answer_triplets. Do not produce any
extra or fewer boolean values.

Task:
1. Identify which of the answer_triplets are hallucinated by
checking if the information in each triplet can be supported by
the original answer.
2. Generate two boolean lists of the exact same length as
answer_triplets:
- A boolean list with comments explaining why each triplet is or
is not hallucinated.
- A plain boolean list without comments.

Ensure both lists have the same number of boolean values as
there are triplets.
```

```
Example Output:
1. Boolean List with Comments:

[
false, // 'Triplet 1 explanation...'
true, // 'Triplet 2 explanation...'
] 2. Plain Boolean List:

[false, true]

In this example, if there were exactly 2 triplets, we have
provided exactly 2 boolean values for each list.

Now analyze the provided inputs and generate the requested
outputs, making sure the number of boolean values matches the
number of answer_triplets exactly.
```

A.3 Prompts from Previous Research

A.3.1 REFCHECKER Prompt

System Prompt:

```
I have a claim that was made by a language model
in response to a question.  Please help me check
whether the claim can be entailed according
to the provided reference, which is related
to the question.

The reference is a list of passages, and the
claim is represented as a triplet formatted
as ('subject', 'predicate', 'object').

If the claim is supported by ANY passage in
the reference, answer True.

If NO passage in the reference entails the
claim, and the claim is contradicted by
some passage in the reference, answer
False.

If NO passage entails or contradicts the
claim, or DOES NOT contain sufficient
information to verify the claim, answer False.

Please DO NOT use your own knowledge for the
judgment.  Just compare the reference and
the claim to determine the answer.
```

User Prompt:

```
You are HallucinationDataGenerator, an assistant specialized
in creating subtle, plausible hallucinations within your
responses.
Your task is to generate answers that are primarily grounded
in the provided reference documents and directions, but also
incorporate carefully crafted, believable fictional elements.
These hallucinations should not be outlandish; instead, focus
on small details that could easily be overlooked--such as
specific years, dosage values, or timeframes.
For instance, you might slightly alter a reported year,
introduce a modest yet unverified numerical detail, or specify
a plausible interval for symptom onset that isn't explicitly
stated.
Ensure that the hallucinated details blend seamlessly with
the given context and do not contradict major facts in the
reference documents.
Maintain coherence, relevance, and credibility throughout your
response.

If few-shot demonstration examples are provided, use them
as a guide to understand the style, approach, and complexity
expected in the hallucinated output.
You may adopt a similar manner of integrating subtle fictional
details as demonstrated in the examples.
```

List of Figures

List of Tables

Bibliography

[BCL+23] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*, 2023.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[BQH+23] Farima Fatahi Bayat, Kun Qian, Benjamin Han, Yisi Sang, Anton Belyi, Samira Khorshidi, Fei Wu, Ihab F Ilyas, and Yunyao Li. Fleek: Factual error detection and correction with evidence retrieved from external knowledge. *arXiv preprint arXiv:2310.17119*, 2023.

[CCC+23] I Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, Pengfei Liu, et al. Factool: Factuality detection in generative ai–a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*, 2023.

[DMG23] Aniket Deroy, Subhankar Maity, and Saptarshi Ghosh. Prompted zero-shot multi-label classification of factual incorrectness in machine-generated summaries. *arXiv preprint arXiv:2312.01087*, 2023.

[DS24] Souvik Das and Rohini K Srihari. Compos mentis at semeval2024 task6: A multi-faceted role-based large language model ensemble to detect hallucination. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1449–1454, 2024.

[EJAS24] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 150–158, 2024.

[FHT+24] Xinyue Fang, Zhen Huang, Zhiliang Tian, Minghui Fang, Ziyi Pan, Quntian Fang, Zhihua Wen, Hengyue Pan, and Dongsheng Li. Zero-resource hallucination detection for text generation via graph-based contextual knowledge triples modeling. *arXiv preprint arXiv:2409.11283*, 2024.

[GVM22] Nuno M Guerreiro, Elena Voita, and André FT Martins. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. *arXiv preprint arXiv:2208.05309*, 2022.

[HFFQ21] Yichong Huang, Xiachong Feng, Xiaocheng Feng, and Bing Qin. The factual inconsistency problem in abstractive text summarization: A survey. *arXiv preprint arXiv:2104.14839*, 2021.

[HRQ+24] Xiangkun Hu, Dongyu Ru, Lin Qiu, Qipeng Guo, Tianhang Zhang, Yang Xu, Yun Luo, Pengfei Liu, Yue Zhang, and Zheng Zhang. Knowledge-centric hallucination detection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6953–6975, 2024.

[HTYZ24] Yi Hu, Xiaojuan Tang, Haotong Yang, and Muhan Zhang. Case-based or rule-based: How do transformers do the math? *arXiv preprint arXiv:2402.17709*, 2024.

[HYM+23] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2023.

[JLL+22] Ziwei Ji, Zihan Liu, Nayeon Lee, Tiezheng Yu, Bryan Wilie, Min Zeng, and Pascale Fung. Rho (ρ): Reducing hallucination in open-domain dialogues with knowledge grounding. *arXiv preprint arXiv:2212.01588*, 2022.

[LCR+24] Junyi Li, Jie Chen, Ruiyang Ren, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. The dawn after the dark: An empirical study on factuality hallucination in large language models. *arXiv preprint arXiv:2401.03205*, 2024.

[LFA+18] Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. Hallucinations in neural machine translation. 2018.

[LG24] Amit Arnold Levy and Mor Geva. Language models encode numbers using digit representations in base 10. *arXiv preprint arXiv:2410.11781*, 2024.

[LHE21] Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021.

[LRW+22] Manling Li, Revanth Gangi Reddy, Ziqi Wang, Yi-Shyuan Chiang, Tuan Lai, Pengfei Yu, Zixuan Zhang, and Heng Ji. Covid-19 claim radar: A structured claim extraction and tracking system. In *Proceedings of the 60th annual meeting of the association for computational linguistics: system demonstrations*, pages 135–144, 2022.

[LSZH24] Wei Liu, Wanyao Shi, Zijian Zhang, and Hui Huang. Hit-mi&t lab at semeval-2024 task 6: Deberta-based entailment model is a reliable hallucination detector. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1788–1797, 2024.

[MH22] Andreas Marfurt and James Henderson. Unsupervised token-level hallucination detection from summary generation by-products. In *Proceedings of the 2nd Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 248–261, 2022.

[MHO+24] Rahul Mehta, Andrew Hoblitzell, Jack O'keefe, Hyeju Jang, and Vasudeva Varma. Halu-nlp at semeval-2024 task 6: Metacheckgpt-a multi-task hallucination detection using llm uncertainty and meta-models. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 342–348, 2024.

[MKL+23] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*, 2023.

[MLG23] Potsawee Manakul, Adian Liusie, and Mark JF Gales. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.

[MZV+24] Timothee Mickus, Elaine Zosa, Raúl Vázquez, Teemu Vahtola, Jörg Tiedemann, Vincent Segonne, Alessandro Raganato, and Marianna Apidianaki. Semeval-2024 shared task 6: Shroom, a shared-task on hallucinations and related observable overgeneration mistakes. *arXiv preprint arXiv:2403.07726*, 2024.

[OWJ+22] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[PKCGH19] Alina Petrova, Egor V Kostylev, Bernardo Cuenca Grau, and Ian Horrocks. Query-based entity comparison in knowledge graphs revisited. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference,*

Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I 18, pages 558–575. Springer, 2019.

[QHQP23] Haoyi Qiu, Kung-Hsiang Huang, Jingnong Qu, and Nanyun Peng. Amrfact: Enhancing summarization factuality evaluation with amr-driven negative samples generation. *arXiv preprint arXiv:2311.09521*, 2023.

[RZA+24] Mohamed Rashad, Ahmed Zahran, Abanoub Amin, Amr Abdelaal, and Mohamed AlTantawy. Factalign: Fact-level hallucination detection and classification through knowledge graph alignment. In *Proceedings of the 4th Workshop on Trustworthy Natural Language Processing (TrustNLP 2024)*, pages 79–84, 2024.

[Sii24] Marco Siino. Brainllama at semeval-2024 task 6: Prompting llama to detect hallucinations and related observable overgeneration mistakes. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 82–87, 2024.

[SPC+21] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

[SRMS24] Hannah Sansford, Nicholas Richardson, Hermina Petric Maretic, and Juba Nait Saada. Grapheval: A knowledge-graph based llm hallucination evaluation framework. *arXiv preprint arXiv:2407.10793*, 2024.

[SRR23] Anne-Dominique Salamin, David Russo, and Danièle Rueger. Chatgpt, an excellent liar: how conversational agent hallucinations impact learning and teaching. In *Proceedings of the 7th International Conference on Teaching, Learning and Education*, 2023.

[TBM+15] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, 16:1–28, 2015.

[TLI+23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[TZJ+24] SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*, 2024.

[WWS+22] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[YV23] Zhangdie Yuan and Andreas Vlachos. Zero-shot fact-checking with semantic triples and knowledge graphs. *arXiv preprint arXiv:2312.11785*, 2023.

[ZQG+23] Tianhang Zhang, Lin Qiu, Qipeng Guo, Cheng Deng, Yue Zhang, Zheng Zhang, Chenghu Zhou, Xinbing Wang, and Luoyi Fu. Enhancing uncertainty-based hallucination detection with stronger focus. *arXiv preprint arXiv:2311.13230*, 2023.

[ZXG+24] Jiawei Zhang, Chejian Xu, Yu Gai, Freddy Lecue, Dawn Song, and Bo Li. Knowhalu: Hallucination detection via multi-form knowledge based factual checking. *arXiv preprint arXiv:2404.02935*, 2024.

[ZYW+23] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*, 2023.