mathematics

MDPI

Review

Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review

Wan Zhang and Jing Zhang *

School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China; zhangwan@seu.edu.cn
* Correspondence: jingz@seu.edu.cn

Abstract: Retrieval-augmented generation (RAG) leverages the strengths of information retrieval and generative models to enhance the handling of real-time and domain-specific knowledge. Despite its advantages, limitations within RAG components may cause hallucinations, or more precisely termed confabulations in generated outputs, driving extensive research to address these limitations and mitigate hallucinations. This review focuses on hallucination in retrieval-augmented large language models (LLMs). We first examine the causes of hallucinations from different sub-tasks in the retrieval and generation phases. Then, we provide a comprehensive overview of corresponding hallucination mitigation techniques, offering a targeted and complete framework for addressing hallucinations in retrieval-augmented LLMs. We also investigate methods to reduce the impact of hallucination through detection and correction. Finally, we discuss promising future research directions for mitigating hallucinations in retrieval-augmented LLMs.

Keywords: large language models; hallucination; retrieval-augmented generation; hallucination mitigation

MSC: 68T07

Academic Editor: Victor Mitrana

Received: 4 February 2025
Revised: 1 March 2025
Accepted: 3 March 2025
Published: 4 March 2025

Citation: Zhang, W.; Zhang, J. Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review. *Mathematics* **2025**, *13*, 856. https://doi.org/10.3390/math13050856

1. Introduction

In recent years, large language models (LLMs) such as GPT-4 [1], LLaMA [2], and Gemini [3] have rapidly advanced, achieving significant progress in the field of natural language processing (NLP). These models learn statistical representations of linguistic patterns and word distributions through complex deep learning architectures and pre-training on vast datasets and adapt to domain-specific tasks through supervised fine-tuning. They are widely used for various tasks, including semantic analysis [4], text generation [5], and reasoning [6]. However, due to their reliance on fixed parameters, LLMs often produce task-irrelevant outputs or generate factually inconsistent responses when faced with tasks beyond the scope of their training data. This limitation can be understood as the knowledge boundary of LLMs, which refers to the extent of their knowledge derived solely from the patterns and associations in the training data. This phenomenon, where the model generates responses that are inconsistent with facts or appear meaningless, is commonly referred to as hallucination or confabulation. Hallucinations undermine the reliability and trustworthiness of LLMs, and in scenarios requiring precise information, they can lead to severe consequences. To address the limitations of LLMs' knowledge boundaries and the high training costs, retrieval-augmented generation (RAG) [7] aims to alleviate knowledge deficiencies by retrieving external information and providing it to the LLM, improving the quality of domain-specific responses and reducing hallucinations. While RAG technology has shown great promise in

mitigating hallucinations, the RAG paradigm itself also has limitations, with the insufficient capabilities of its components contributing to the generation of hallucinations.

Investigating the root causes of hallucinations in the RAG paradigm helps to understand the mechanisms of its components at different stages and how their limitations affect LLM-generated responses. It also provides guidance for developing hallucination mitigation strategies for retrieval-augmented LLMs. Based on the retrieval and generation phases of RAG, we categorize the sub-tasks in each phase and analyze the fundamental issues causing hallucinations in each sub-task. This classification spans the entire RAG process, allowing a comprehensive analysis of the root causes and mechanisms of hallucinations at each stage. Specifically, in the retrieval phase, we focus on data source, query, retriever, and retrieval strategy problems. In the generation phase, we address context noise, context conflict, and the middle curse in long contexts, alignment problems, and model capability boundary problems.

In addition, based on the causes of hallucinations, we surveyed and categorized corresponding hallucination mitigation methods. We found that prompt engineering plays a significant role in mitigating hallucinations, addressing tasks such as designing prompts to directly reduce hallucinations, using prompt techniques to automate the construction of resources like demonstrations and labeled datasets for hallucinations mitigation, and evaluating model performance on specific tasks to further guide the tuning of generative models. Given its simplicity, efficiency, universality, and interpretability compared to model optimization and supervised fine-tuning, most of the hallucination mitigation methods we reviewed extensively utilize prompt techniques. This review aims to encourage researchers to further explore the potential of prompt engineering in hallucination mitigation tasks while consciously adhering to the norms of using prompt engineering for various research tasks [8]. Considering that hallucinations are still unavoidable even after optimizing each subtask of RAG, it is also worth investigating methods to reduce the impact of hallucinations through detection and correction.

Finally, we also discuss the challenges faced by existing methods, provide some promising directions to mitigate hallucinations in retrieval-augmented LLMs, and offer suggestions for future research.

1.1. Comparison with Existing Reviews and Surveys

The hallucination or confabulation issue in generative artificial intelligence (GAI) has had a significant impact on the reliability and trustworthiness of many existing applications based on generative AI, making it a major challenge in the field. Numerous reviews and surveys have been conducted on this topic [9–14]. These surveys explore the hallucination problem from various perspectives and offer valuable insights. However, our review specifically focuses on hallucinations within the RAG paradigm. Liu et al. [9] provided an in-depth investigation of hallucinations in large visual language models (LVLMs), introducing innovative evaluation methods and benchmarks, analyzing the root causes of hallucinations, and discussing corresponding mitigation techniques. Sahoo et al. [10] established a precise definition and structured classification of hallucinations, offering a comprehensive overview of hallucination detection and mitigation techniques for multimodal foundation models. Alghamdi et al. [11] discussed knowledge graph-based enhancement techniques from three perspectives, knowledge-aware inference, training, and knowledge-aware validation, assessing their effectiveness in reducing hallucinations. Tonmoy et al. [12] classified hallucination mitigation techniques into prompt engineering and model design, providing a thorough review. Huang et al. [13] offered a hierarchical classification method for hallucinations and presented a detailed, coherent introduction to their causes and solutions across data, training, and inference stages. Zhang et al. [14] analyzed the causes of hallucinations in LLMs and provided a comprehensive review of

mitigation methods across four stages: pre-training, supervised fine-tuning, reinforcement learning from human feedback (RLHF), and inference. However, our review specifically focuses on classifying and analyzing the root causes and mechanisms of hallucinations in the sub-tasks of the RAG paradigm and provides corresponding mitigation methods. These methods emphasize the potential of prompt engineering in reducing hallucinations.

1.2. Organization of the Review

This paper reviews recent research on hallucinations in retrieval-augmented LLMs. Figure 1 shows the internal organizational principles of this review, which include the causes of hallucinations and corresponding mitigation methods in retrieval-augmented LLMs. Specifically, the causes of hallucination mitigation in different subtasks of the generation deficiency and retrieval failure stages in Figure 1 are analyzed in detail in Section 3. Sections 4 and 5 review hallucination mitigation methods corresponding to the causes in the subtasks of the retrieval and generation stages, respectively. Additionally, Section 2 introduces the classical RAG pipeline, delineates the conceptual distinctions between hallucination and confabulation, summarizes existing taxonomies of hallucination phenomena, and overviews prompt techniques commonly employed in hallucination mitigation strategies. Section 6 covers techniques for mitigating hallucinations after detection. Finally, Section 7 discusses the challenges faced by current mitigation methods and provides suggestions for future research directions.

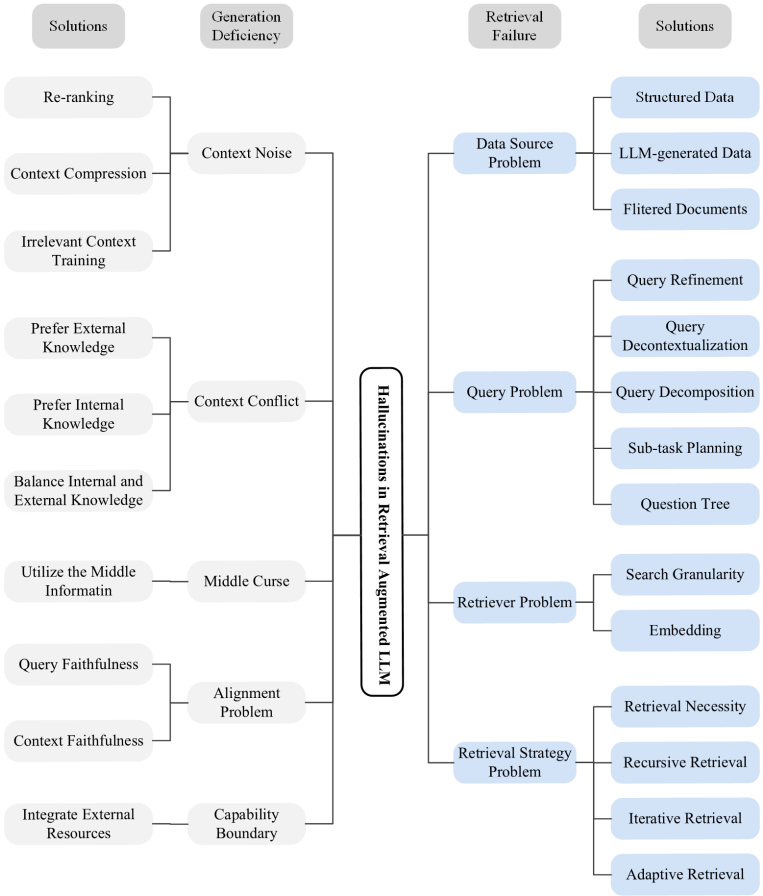

Figure 1. Causes of hallucinations and corresponding mitigation methods in retrieval-augmented LLMs. Hallucinations in RAG models arise from two primary stages: retrieval failure and generation deficiency. Retrieval failure includes four sub-problems: data sources, queries, retrievers, and retrieval strategy problems. Generation deficiency encompasses five sub-problems: context noise, context conflict, middle curse, alignment problems, and capability boundary. For each sub-problem, corresponding solutions for mitigating hallucinations are provided.

2. Concepts and Definitions

This section provides some important concepts and definitions related to retrieval-augmented LLMs.

2.1. Retrieval-Augmented Generation

A typical retrieval-augmented generation (RAG) process is shown in Figure 2. The user inputs a question about recent news, but due to the real-time nature of news, the LLM's internal parameters do not contain the necessary knowledge to answer the question. External knowledge needs to be retrieved and integrated to fill this information gap. In the retrieval phase, the input question is pre-processed and passed as a query to the retriever. RAG uses external retrieval engines such as BM25, TF-IDF, or more complex neural retrieval models to perform dense or sparse retrieval from web pages, pre-constructed document corpora, knowledge bases, or databases to find relevant documents or paragraphs. Dense retrieval involves encoding documents and queries using a pre-trained embedding model to generate dense vector representations. The retriever typically returns the top-k most relevant text segments based on the query-document relevance. In the generation phase, the retrieved documents or paragraphs are combined with the user's original question and passed to the generation model to generate a response for the user.

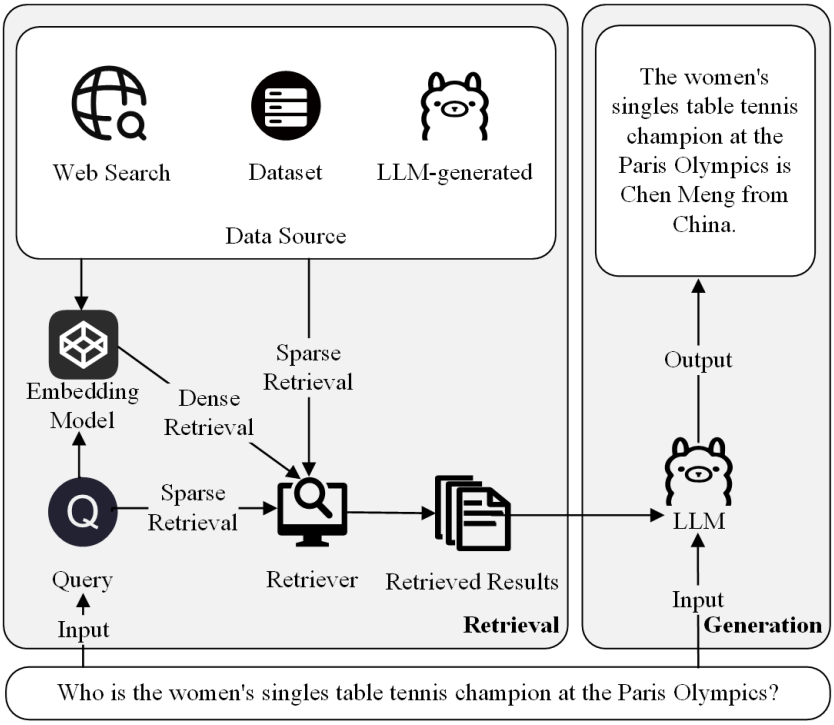

Figure 2. The basic workflow of retrieval-augmented generation.

2.2. Hallucinations or Confabulation in Large Language Models

The concept of hallucination originated in psychology, referring to perceptual experiences that occur without external stimuli or perceptions that deviate from reality [15]. In natural language processing, hallucinations typically refer to information generated by LLMs that is inconsistent with facts or nonsensical. However, Smith et al. [16] contend that the term hallucination inappropriately suggests that LLMs possess sensory perception, thereby constituting an imprecise metaphor. They propose the concept of confabulation, derived from psychiatry, as a more suitable alternative. Confabulation denotes the unconscious fabrication of false memories by individuals with impaired memory or cognitive deficits to fill gaps in their recollection [17]. Unlike perceptual experiences, confabulations

represent erroneous reconstructions of information that are influenced by internal and external knowledge and the environment. This aligns more closely with the behavior of LLMs, which generate outputs that are erroneous yet plausible based on their training data and context. Although the term confabulation has gained traction in public discourse on artificial intelligence [18] and has garnered consensus among researchers, it has not yet been widely adopted in the academic literature. Consequently, we continue to employ the term hallucination to review and analyze the relevant methods.

Some researchers have categorized hallucinations based on different criteria. And we list examples of different types of hallucinations in Table 1. Ji et al. [19] categorized hallucinations in NLP into intrinsic and extrinsic based on whether they can be verified against the input source. Intrinsic hallucination occurs when the generated content contradicts the input, while extrinsic hallucination refers to content that cannot be verified from the input but may be correct in external knowledge. Huang et al. [13] refined this classification by introducing factuality hallucination and faithfulness hallucination based on the consistency with real-world knowledge and user-centered contexts. Factuality hallucination involves inconsistencies with real-world facts, such as factual contradiction (responses contradicting known facts) or factual fabrication (responses fabricated as facts). Faithfulness hallucination refers to inconsistencies with user instructions, context, or internal consistency, including instruction inconsistency (responses deviating from the original instructions), context inconsistency (responses inconsistent with provided context), and logical inconsistency (internal contradictions in responses). Rawte et al. [20] proposed several classification methods, including hallucinations caused by correct or incorrect prompts, the severity of hallucinations, and specific error types. They termed the responses that deviate from facts when LLMs process factually correct prompts as the factual mirage and the seemingly plausible but incorrect responses generated from factually incorrect prompts as the silver lining. They further categorized hallucinations into six specific types: acronym ambiguity, numeric nuisance, generated golem, virtual voice, geographic erratum, and time warp.

Table 1. Examples of different types of hallucinations.

Type	Question	Response	Explanation
Intrinsic Hallucination	Company A's financial report shows a net cash flow from operating activities of CNY 520 million. Analyze A's cash flow situation.	The net cash flow from investing activities is CNY 380 million.	The actual financial report shows a figure of CNY 520 million, which contradicts the facts.
Extrinsic Hallucination		The improvement in cash flow is attributed to the expansion in the Southeast Asian market.	The financial report does not disclose regional details, but it may be consistent with external knowledge.
Factuality Hallucination	Please introduce the history of the Great Wall of China.	The Great Wall of China was ordered to be built by Emperor Qin Shi Huang after he unified the six warring states.	After Emperor Qin Shi Huang unified the six warring states, he undertook large scale repairs and expansions of the Great Wall, but this was not the time when the Great Wall was originally built.

Table 1. *Cont.*

Type	Question	Response	Explanation
Faithfulness Hallucination	Translate: Who was the director of Avatar?	Avatar was directed by James Cameron.	The LLM's response is inconsistent with the instruction and failed to translate the sentence as required.
Factual Mirage	Conan Doyle is the author of the Sherlock Holmes series. Please introduce Conan Doyle.	Conan Doyle was awarded the title of "Royal Deective" for his exceptional reasoning skills.	Arthur Conan Doyle was never awarded the title of "Royal Detective". When LLM processes a factually correct prompt, the generated output may deviate from the truth.
Silver Lining	Why did Lin Daiyu uproot the weeping willow?	Lin Daiyu uprooted the weeping willow because she was sorrowful over her tragic fate, using it as a way to express her grief over the impermanence of life and the unfairness of destiny.	Lin Daiyu did not uproot the weeping willow. The LLM generated false content when handling prompts with factual errors.

In retrieval-augmented LLMs, hallucination caused by the inherent knowledge boundary of LLMs is mitigated by retrieving external information, but hallucinations may still occur [21]. When the retrieved information contains false, outdated content, or conflicts with the internal parameters of the model, factuality hallucination may occur. When the query does not accurately reflect the user's intention, the retrieved content does not accurately reflect the user's query, which will cause faithfulness hallucination. When LLMs cannot make good use of the context obtained from the query, both types of hallucinations may occur.

2.3. Common Prompt Techniques

Prompt engineering directs the model to generate higher-quality, more user-friendly responses through the careful design and construction of input prompts. Compared to other methods, such as model architecture modification or fine-tuning, prompt engineering offers significant advantages in terms of simplicity, efficiency, generality, and interpretability. It enables rapid adaptation to new tasks and efficient optimization by simply designing appropriate input prompts without the need to modify the model structure or retrain the model. Additionally, prompt techniques have minimal computational and data requirements, making them easy to implement and deploy. Prompt engineering also supports timely adjustment and optimization, maintaining its universality across different models. As a result, prompt engineering plays a crucial role in the study of hallucination mitigation in retrieval-augmented LLMs. On the one hand, well-constructed prompts can guide the model to generate outputs with higher semantic alignment to input queries by optimizing the conditional probability distribution over token sequences, leading to more accurate and consistent responses. On the other hand, prompt engineering can also automate the creation of labeled datasets, saving time and cost, and evaluate the model's performance, guiding further tuning. The following section introduces common prompt techniques used in the hallucination mitigation methods discussed later to help further understand the role of prompt engineering in these tasks. Table 2 shows examples of different prompt techniques.

- Clear and Precise Prompts

Constructing clear and specific prompts can effectively guide the model to generate the desired output. LLMs, pre-trained on vast datasets containing extensive knowledge, may struggle to interpret vague or broad prompts, often resulting in generalized responses. In contrast, clear and precise prompts can reduce ambiguity and polysemy in language, clearly point out the core content that users are concerned about, make it easier for the model to identify and match patterns related to these keywords, and provide more accurate and satisfactory responses.

- Role-Playing Prompts

Role-playing prompts [22] assign a specific role or identity to the LLMs, guiding them to generate responses that align with a particular style or area of expertise. This approach helps the model better emulate human conversational patterns, enhancing the accuracy and relevance of its output. For instance, in customer service applications, the model can assume the role of a customer service representative, thereby providing more professional and user-friendly interactions.

- Few-Shot Prompts

Few-shot prompts [23] involve providing a small number of carefully designed examples within the prompt to help the model quickly grasp the task requirements and the expected output format. This approach enhances the accuracy and consistency of the model's responses. The few-shot prompt is particularly beneficial in scenarios with limited data, as it reduces the reliance on large-scale annotated datasets while significantly improving the model's performance across various tasks.

- Chain-of-Thought (CoT) Prompts

CoT prompts [24] simulate the step-by-step reasoning process of humans when solving complex problems, guiding the model to generate intermediate reasoning steps and gradually work through each step to reach the final answer. This approach significantly enhances the model's performance in multi-step reasoning tasks, such as solving mathematical problems, common-sense reasoning, and symbolic reasoning. Zhang et al. [25] introduced Auto-CoT, which automatically prompts LLMs to generate reasoning chains by incorporating the phrase "Let's think step by step" into the prompt. These chain-of-thought promptings force the model to transform the implicit probability distribution calculation process into an explicit logical chain of text output. When the model answers complex problems step by step, it faces subtasks with lower entropy, making it easier to generate accurate responses and reducing the risk of error accumulation. As a result, it can better answer complex questions.

- Program-of-Thought (PoT) Prompts

PoT prompts [26] direct the language models to express their reasoning process using a programming language, delegating computational tasks to an external program interpreter through a clear task description, sample code, and specific questions. PoT decomposes complex reasoning processes into distinct coding steps, leveraging the iteration and loop structures of programming languages to efficiently manage tasks requiring multiple iterations. By utilizing a program interpreter for complex computations, PoT helps mitigate errors in the language model's calculations, thereby enhancing both the accuracy and efficiency of the model in numerical reasoning tasks.

- Opinion-Based Prompts

The opinion-based prompts [27] reformulate the context into a narrator's statement and transform the question into an inquiry about the narrator's opinion on that statement.

This approach alters the nature of the question, compelling the model to focus more on the specific statements in the context rather than its internally stored factual knowledge.

Table 2. Examples of different prompt techniques.

Type	Example Prompt	Explanation
Clear and Precise	I want to learn about healthy diets, especially low-sugar, high-fiber meal combinations suitable for diabetic patients to eat for breakfast.	The prompt "I want to learn about healthy diets" is too ambiguous, leading to generic suggestions. A clear prompt specifying the target audience, time, and needs helps the model provide more tailored advice.
Role-Playing	You are a professional customer service representative, responsible for providing customer support for an electronics company.	Assigning a role (such as a customer service representative) to the LLM in the prompt enables it to generate role-specific responses.
Few-Shot	Question: Classify the sentiment of the following sentences into "positive" or "negative". Examples: "I absolutely love this phone!" positive "The product is terrible. I hate it." negative Input: "The movie was amazing!" Output:	By showing it how to classify sentences as "positive" or "negative", the model can then apply this pattern to new inputs, like "The movie was amazing!"
CoT	Question: If the train leaves at 2:30 p.m. and the journey takes 3 h and 45 min, what time will it arrive? Step-by-step reasoning: 1. Start time: 2:30 p.m. 2. Add 3 h to 2:30 p.m. to get 5:35 p.m. 3. Add 5 min to 5:30 p.m. to get 5:30 p.m. Output:	Through step-by-step reasoning, the CoT prompt helps the model perform addition operations in chronological order to arrive at the correct arrival time.
Auto-CoT	What should be included when writing a proposal for regional environmental protection measures? Let's think step by step.	Using the Auto-CoT prompt with "Let's think step by step" gives model more time to process and reason.

PoT	Question: What is the sum of 1 to 100?
	# Python 3.8 code
	total_sum = 0
	for i in range(1, 101):
	  total_sum+=i
	print(total_sum)
		This PoT prompt uses a clear task description and example code to guide the model in reasoning through code. The program interpreter executes the task and computes the result efficiently.

Table 2. *Cont.*

Type	Example Prompt	Explanation
Opinion-Based	Li Hua said: "The new president of the United States is Donald Trump". According to Li Hua's opinion, who is the new president of the United States?	Opinion-based prompt makes the model focus on Li Hua's statement and reflect that individual's belief, not its internal knowledge.

3. Hallucination Causes in Retrieval-Augmented Generation Process

LLMs often suffer from hallucinations when generating text, which affects the reliability and accuracy of the model output. In order to mitigate hallucinations in LLMs and improve the factuality of their output, the RAG framework was proposed [28–31]. RAG combines the retrieval module and the generation module and uses the external knowledge base to provide the model with richer and more accurate contextual information, thereby improving the accuracy and credibility of the generated content. Although RAG has significant advantages in mitigating hallucinations, it still has the possibility of introducing hallucinations at different stages. Specifically, hallucinations in RAG models arise from two primary stages: retrieval failure and generation deficiency. In the retrieval stage, the retrieval module may not be able to provide accurate contextual information due to the unreliability of the data source, the ambiguity of the user query, or the limitations of the retriever and retrieval strategy. In the generation phase, the generation module may generate content inconsistent with the retrieved information or input requirements due to context noise, context conflict, middle curse or alignment problems, or inaccurate generation due to limited model capabilities. We will start from these two stages to reveal the specific mechanism of hallucination in the process of retrieval-enhanced generation and provide a theoretical basis for subsequent research.

3.1. Retrieval Failure

In the retrieval augmentation system, the retrieval process plays a core role and is responsible for obtaining contextual information related to user queries from external knowledge bases to enhance the generation ability of language models. However, the retrieval process may cause hallucinations due to various reasons. This section will analyze the causes of hallucinations from four perspectives: data source problems, query problems, retriever problems, and retrieval strategy problems.

- Data Source Problem

Retrieval-augmented LLMs are prone to generate hallucinations due to three key data source issues: low quality, obsolescence, and singleness of retrieval sources. Low-quality data sources may rely on unverified and unreliable materials, such as inaccurate or biased information, which can easily cause the model to generate unrealistic responses based on wrong data. In particular, with the development of generative artificial intelligence, more and more LLM-generated content is published on the Internet. Modern retrieval models increasingly favor LLM-generated content over human-created sources. This preference stems from two factors: (1) the high semantic similarity between generated contexts and user queries and (2) the inherent incompleteness of retrieved contexts. The tendency of LLMs to produce factual hallucinations will aggravate the reliability problem of retrieval sources. In addition, various data poisoning attacks against knowledge bases will also affect the correctness of data sources. Outdated data sources often contain data that have not been updated on time, causing LLMs to make inferences based on outdated research or data, thereby producing hallucinations that are inconsistent with current knowledge. The singleness problem is that the model relies on only a limited or single type of data source,

ignoring relevant information in other fields, which will lead to incomplete or wrong inferences. In short, these problems affect the reliability and completeness of information, prompting LLMs to generate inaccurate or unrealistic content.

- Query Problem

User queries play a fundamental role in RAG, which directly determines the direction and accuracy of the information obtained by the retrieval system. However, ambiguous queries and complex queries often cause RAG systems to generate hallucinations, affecting the quality of generated content. Ambiguous queries refer to user queries that contain omissions, coreferences, and ambiguities. Due to the lack of clarity or the existence of multiple possible interpretations, such queries make it difficult for the retrieval system to accurately identify the user's true intentions, thereby increasing the difficulty of obtaining accurate and relevant information. Table 3 lists several examples of ambiguous queries. Complex queries usually require intensive reasoning capabilities or involve information retrieval from multiple aspects. Such queries may exceed the capabilities of current retrieval methods based on keywords or semantic matching, resulting in one-sided or wrong retrieval results. Complex queries require advanced processing and decomposition capabilities to accurately retrieve and integrate information from multiple sources to generate accurate responses. These two types of query problems not only affect the quality of retrieval results but also reduce the credibility of the responses generated by the model, ultimately affecting the user experience.

- Retriever Problem

The role of the retriever in RAG is crucial. It is responsible for finding information related to user input from the knowledge base or document library to provide contextual support for the subsequent generation process. However, the retriever may also become the source of large model hallucination, mainly in terms of retrieval granularity and embedding.

Table 3. Examples of ambiguous queries that can cause hallucinations.

Type	Example Dialog	Explanation
Ambiguity	Query: Apple is very popular, why?	The word "Apple" in this query is ambiguous, referring to either the fruit or the phone brand. The system erroneously retrieved and explained information about the phone brand.
	Retrieved Paragraph: Apple's popularity is driven by its innovative products, which blend style with functionality…	
	Answer: Apple's popularity can be attributed to its innovative approach in combining style and functionality…making it a preferred choice for many consumers.	
Omission	Query: Who is the 2024 Olympic 100 m champion?	The question omitted a specific 100 m race, so the system only searched for the men's 100 m race results and gave the champion of the men's 100 m race.
	Retrieved Paragraph: USA sprinter Noah Lyles won the men's 100 m title at the Paris 2024 Olympic Games by the smallest of margins in a dramatic photo finish.	
	Answer: The 2024 Olympic 100 m champion is Noah Lyles from the United States. He won the gold medal with a time of 9.784 s, narrowly defeating Jamaica's Kishane Thompson by just five-thousandths of a second.	

Table 3. *Cont.*

Type	Example Dialog	Explanation
Coreference	Query: Li Hua recommended a history book to Xiaoming. He thought it was very interesting and bought it. Who bought a history book?	The query gives two people, but "He" does not clearly indicate who they are, which causes the system to perform inappropriate retrieval and incorrect model generation.
	Retrieved Paragraph: Li Hua is a common protagonist in English exam essays, often set as a character who recommends books or shares experiences with others.	
	Answer: Li Hua is the one who recommended the book and found it interesting; it is likely that Li Hua is the one who bought the history book.	

First, the retrieval granularity determines the size and detail level of the retrieved information blocks. The retrieval structure matching different retrieval granularities will also affect the effectiveness of the retriever. If the retrieval granularity is too large, the retrieved information block may be too broad and contain a lot of irrelevant content, which will increase the burden of the language model to process information and may cause the generated answer to be inaccurate or even deviate from the core of the user's question. On the contrary, if the retrieval granularity is too small, although the information block is more targeted, it may not provide enough contextual information, making it difficult for the language model to integrate this fragmented knowledge, thus affecting the coherence and completeness of the answer. Secondly, many RAG frameworks rely on flat data representation, which cannot capture multi-hop relationship paths between encoded entities and limits the system's ability to capture complex relationships between entities. For example, when a user asks "How does artificial intelligence technology change the diagnosis and treatment methods in the medical industry?", the RAG system may retrieve separate documents on artificial intelligence, medical diagnosis, and treatment methods. However, due to the reliance on flat data representation, these systems may not be able to effectively integrate the complex relationships between these documents, resulting in fragmented generated answers. Finally, the quality of the embedding model directly affects the accuracy of retrieval. The embedding model converts text into a vector representation for calculating similarity scores. If the embedding model is suboptimal, it may reduce the matching degree between the retrieval results and the user query, which in turn affects the output of the language model and leads to hallucinations. Therefore, optimizing the retrieval granularity and embedding model of the retriever is crucial to improving the overall performance of RAG and reducing the generation of hallucinations.

- Retrieval Strategy Problem

Retrieval strategies involve the necessity, frequency, and timing of retrieval. Regarding the necessity of retrieval, not all queries require retrieval. When the internal knowledge of the model is sufficient to answer the question, blind retrieval may lead to conflicts between the retrieved information and the internal knowledge on the one hand and introduce noise into the context on the other hand, both of which will lead to misleading responses. Regarding the frequency and timing of retrieval, the earliest strategy of performing a single retrieval on the original input [32,33] achieved good performance in handling single-hop problems. However, a single retrieval often has difficulty handling complex queries, especially those that require multi-step reasoning or involve multiple sub-questions. In addition, a single retrieval requires extremely high precision and recall of the retrieval

system. Otherwise, it is easy to affect the generation quality due to incomplete or inaccurate information retrieved. To better cope with complex information retrieval, it is necessary to propose a better strategy to overcome the defects of a single retrieval on the premise of determining the necessity of retrieval.

3.2. Generation Deficiency

After the retrieval process, the generation phase, as the final output core of RAG, is responsible for integrating the retrieved relevant information with the user input and generating accurate and context-related answers through LLMs. However, the generation process may lead to hallucinations due to various reasons. This section will specifically analyze the causes of hallucinations in relation to five aspects: context noise, context conflict, middle information utilization, alignment problems, and ability boundaries.

- Context Noise

Contextual noise is often caused by the failure of the retrieval process, which can result in the retrieved context containing information that is irrelevant or redundant to the user's question, fragmented information that lacks coherence and completeness, and wrong or outdated information. Cuconasu et al. [34] pointed out that a large number of interfering documents retrieved by the retriever may contain semantic information related to the query, but this information may not support the correct answer and may cause the LLM to generate incorrect answers based on misleading information. At the same time, they also found that adding irrelevant random documents to the context reduces the focus of attention, avoids entropy collapse, and can significantly improve the performance of LLMs. Zhao et al. [35] pointed out that since LLMs are trained on a large amount of historical text, there is an inherent risk that outdated information matches the model's internal knowledge base. This matching may prompt LLMs to tend to and perpetuate outdated information. In short, large language models are extremely sensitive to this noise information. After the noise information is passed to the generative model, the answers generated by the model may be logically incoherent or generate completely fictitious content, thus leading to hallucinations.

- Context Conflict

Context conflict mainly refers to the conflict between knowledge in the context and model internal parameter knowledge. RAG brings richer context to the model input, but it may also bring information that conflicts with the model's internal parameter knowledge. Xu et al. [36] believe that the conflict between context and model parameter knowledge is mainly caused by temporal misalignment and misinformation pollution. Temporal misalignment means that the temporal information based on pre-training data may be outdated, while the external contextual knowledge may contain the latest information. This temporal difference will cause the model to fail to accurately reflect the current reality when dealing with problems. Misinformation pollution refers to the possibility that the external context contains erroneous or misleading information, which may be mistakenly adopted by LLMs, resulting in the generation of wrong or hallucination content. Longpre et al. [37] replaced the entity mentioned in the standard document with another entity to create a conflicting context, revealing the model's tendency to over-rely on parameter knowledge. Chen et al. [38] used multiple paragraphs and found that when the model performed best, it mainly relied on contextual knowledge. They believe that the entity replacement method reduces the semantic coherence of the interfering paragraphs, and Longpre et al. [37]'s study is based on a single paragraph, which leads to the model's preference for parameter knowledge. Xie et al. [39] found that as long as the external knowledge is coherent and convincing, LLMs tend to obey external knowledge, even if this knowledge conflicts with internal parameter knowledge.

Many recent studies have also emphasized that the model tends to be consistent with the user's views—that is, the model exhibits "sycophancy" [40–43].

- Middle Curse

Despite retrieving factually correct and query-relevant information, LLMs may still exhibit degraded generation quality due to insufficient utilization of long contexts, especially the middle curse issue [44]. The self-attention mechanism in Transformer [45] models exhibits attention decay and insufficient resolution in positional encoding when processing long texts, compounded by limitations in memory and computational resources. These factors collectively lead to a significant degradation in the model's ability to capture information from the middle portion of the input text. When critical information is located in the middle of the context, LLMs often struggle to effectively localize and utilize such information, resulting in a notable decline in accuracy. This problem is particularly prominent in Multi-doc QA tasks. Recent studies have further demonstrated that the middle curse also exists in tasks such as abstraction [43], long-form QA [46], and paragraph ranking [47].

- Alignment Problem

The alignment problem is mainly reflected in two aspects: context-generation alignment and query-generation alignment in the retrieval-augmented large language model. Context-generation alignment emphasizes the consistency and relevance of the generated text with the input context in terms of content and logic. If the generative model fails to effectively integrate the retrieved contextual information, or the context is too complex, the generated text may deviate from the factual consistency of the input content, resulting in faithfulness hallucination. Query-generation alignment requires that the generated content maintains high semantic correspondence with the content and structural patterns of the user's query. In the RAG framework, there is a semantic, contextual, or structural mismatch between the query and the retrieved knowledge base. For example, there may be inconsistencies between the query raised by the user and the retrieved literature or data, resulting in the generated answer not aligning with the user's real needs. The root cause of this alignment problem is usually the complexity of the query, the diversity of external knowledge, and the limitations of the retrieval mechanism. If the retrieved knowledge does not fully meet the intent of the query, the generated answer may have information omissions or logical errors, thus causing hallucinations.

- Capability Boundary

The capability boundaries of LLMs are mainly reflected in the deficiencies in precise calculation, data retrieval, and complex logic processing. These deficiencies are mainly due to their design goals based on statistical learning and language patterns, rather than being built for mathematical reasoning or symbolic operations. LLMs lack built-in mathematical logic, and their training data mainly comes from unstructured natural language texts. The model does not specifically learn mathematical rules, resulting in reliance on language patterns rather than real mathematical operations when dealing with mathematical problems. In addition, LLMs have limited digital representation capabilities, and complex mathematical expressions will be split into multiple tokens, affecting the accuracy of calculations. The presentation of mathematical problems is also relatively limited, lacking strict mathematical formulas and reasoning, which makes the model lack a sufficient data foundation for precise calculations. In addition, the reasoning process of LLMs is based on language generation rather than logical reasoning, lacking an understanding of logical rules and causal relationships and having difficulty handling complex multi-step reasoning tasks. These capabilities make LLMs prone to hallucinations. Prompt methods such as chain-of-thought encourage LLMs to decompose the problem, but when solving

the problem, even if the problem is correctly decomposed, LLMs often make mistakes in logic and arithmetic [48].

4. Solutions to Retrieval Failure

This section reviews several key solutions to the problem of retrieval failure. First, the accuracy and credibility of information can be significantly enhanced by ensuring the reliability of data sources. Second, effective query construction methods have been proposed to deal with ambiguous and complex queries. Third, the effectiveness of the retriever is enhanced from two aspects, retrieval granularity and the embedding method, which improves its ability to handle complex queries. Finally, well-designed retrieval strategies ensure the efficient execution of the retrieval process and the accuracy of the results. These solutions target different subproblems of retrieval failure, aiming to comprehensively improve the overall performance of the retrieval system and effectively reduce hallucinations in retrieval-augmented LLMs.

4.1. Data Source Reliability

Structured data, such as knowledge graphs (KGs), are usually built from verified data sources and are constantly updated. This verification process ensures their ability to deliver accurate, reliable knowledge. The modular retrieval-augmented LLM KnowledGPT [49] combines LLMs with knowledge bases (KBs). It generates Python 3.8 code through the program of thoughts (PoT) [26] to perform knowledge base operations, retrieves information from different knowledge bases, and enables multi-hop retrieval for complex questions. Additionally, KnowledGPT [49] introduces a personalized knowledge base (PKB). This allows users to store knowledge related to professional fields or personal interests according to their needs. It also supports multiple knowledge representation forms, including entity descriptions, relationship triples, and entity-aspect information, enabling a broader range of knowledge storage.

The context generated by LLMs is aligned with the pre-training target and, therefore, tends to contain more relevant information. GENREAD [50] replaces the retriever with an LLM generator, which first prompts the LLM to generate context documents based on a given question and then generates a final response based on the generated documents. Cheng et al. [51] found that the output of LLMs has a memory that is more similar to the data distribution during inference than the training data and thus proposed the selfmem framework to improve the generation quality by using the output of the model. Slefmem creates an unbounded memory pool by the retrieval-augmented generator and uses a memory selector to select output as the memory for subsequent generation rounds; it then iterates through the generation process to enhance the model generation performance.

In addition, filtering and evaluating the credibility of retrieval sources can also prevent retrieval failures. Inspired by common practices in pre-training data processing [52], Asai et al. [53] considered a quality filter to ensure the high quality of retrieval source data storage. Xie et al. [39] found that relying solely on internal knowledge to evaluate the credibility of information is unreliable. Pan et al. [54] proposed a credibility-aware (CAG) framework to assign credibility levels to document-level and sentence-level retrieval information based on the relevance, timeliness, and reliability of the source of the information. CoT [24] was used on an existing QA dataset to prompt the large language model to generate explanations based on questions, retrieved documents with credibility annotations, and golden answers. The LLM was fine-tuned through instructions to generate responses based on credibility, which mitigated the impact of introducing defective information in the retrieval process.

4.2. User Query Construction

4.2.1. Ambiguous Queries

Query refinement can supplement and improve background information and solve the problem of necessary background information that is short, ambiguous, and lacking in search queries. Query refinement can be achieved through query expansion or query rewriting.

Query expansion is used to achieve disambiguation and satisfy the user's rich intents. Wang et al. [55] proposed a query2doc query expansion method, which first generates pseudo-related documents that help query disambiguation and guide the retriever through a small number of samples to prompt LLMs, and then connects them with the original query to form a new query. KnowledGPT [49] uses the functions of the knowledge base to obtain candidate entities and related information and uses few-shot prompting to prompt LLMs to align the entities mentioned in the text with the entities in the knowledge base to achieve disambiguation, avoiding the introduction of irrelevant and erroneous knowledge due to ambiguous retrieval. Jagerman et al. [56] generate new query terms through LLMs and combine them with the original query terms repeated five times to form a new expanded query. Studies have shown that applying CoT prompts [24] to this query expansion effectively improves the search engine recall rate. Wang et al. [57] found that ambiguous, multi-faceted queries, such as "What is jazz?", often contain multiple sub-intents, and users want to obtain comprehensive answers covering multiple aspects. Based on this, Wang et al. [57] proposed the RichRAG framework, which explicitly models various sub-aspects of the query and retrieves external knowledge to build a diverse document candidate pool, thereby providing comprehensive long-form answers and satisfying the user's rich intent. Kim et al. [58] proposed the Tree of Clarification (ToC) framework to construct more effective queries by identifying all clarification questions for fuzzy queries. TOC recursively builds the clarification tree of AQ by leveraging external knowledge for few-shot hints and pruning when necessary. TOC solves the challenge of clarification from multiple dimensions through a recursive tree and satisfies the external knowledge required to identify clarification questions and their answers through the retriever.

Query rewriting is used to align queries with retrieval requirements. Ma et al. [59] proposed a re-writer trained by reinforcement learning. The re-writer uses the LLM performance as a reward, learning to adapt the retrieval query to improve the reader on downstream tasks. Mao et al. [60] used the feedback signal of a publicly available re-ranker to train the rewriting model, eliminating the dependence on labeled data.

Decontextualization is used to effectively solve dialog dependency problems presented by questions in conversational retrievals, such as omission, ambiguity, and coreference [61–63]. Choi et al. [64] first proposed the decontextualization task. This task aims to extract sentences from their rich contexts and rewrite them into independent sentences that can be understood without context while retaining their original meaning. Choi et al. [64] designed an annotation pipeline to provide a high-quality annotated dataset and evaluation scripts and trained an automatic decontextualization model. Yoon et al. [65] guided an advanced LLM to generate various potential rewrites through multiple prompting methods, collected the retrieval feedback on each rewrite, and constructed a large-scale dataset called RF_COLLECTION. The dataset contains retriever feedback on more than 410,000 query rewrites from 12,000 conversations optimized for retrieval. Then, Yoon et al. [65] optimized the rewriter based on this dataset to generate retriever-preferred query rewrites, achieving the goal of contextualization.

4.2.2. Complex Queries

Query decomposition is a common approach when faced with complex queries involving multiple aspects, as it breaks down complex queries into subqueries to retrieve

information more accurately. Shao et al. [66] proposed a method named STROM, focusing on applying LLMs to write long texts in an organized and well-founded manner. This method decomposes topics into various perspectives by retrieving Wikipedia articles on similar topics and gives LLMs specific perspectives, simulating multiple rounds of dialogue between authors and experts to collect knowledge. Finally, a multi-level section heading outline is generated by combining the internal knowledge of LLMs and the collected information, and a long text is generated based on this.

Sub-task planning is often used to decompose and solve complex queries that require intensive reasoning step by step. Through sub-task planning, complex and intricate queries can be decomposed into manageable, sequential steps that can be addressed individually. When combined, these steps provide context for the original problem and answer it in full. To overcome the challenge of easy-to-hard generalization in the chain of thought, Zhou et al. [67] propose a novel prompting strategy, least-to-most prompting. The foundational premise of this approach is to systematically decompose a complex problem into a series of easily solvable sub-problems and then solve these sub-problems sequentially in order, with each sub-problem solved with the help of the answer of the previously solved sub-problem. Khot et al. [68] also proposed decomposed prompting (DECOMP) based on the idea of subproblem planning. Compared with least-to-most, DECOMP has the following advantages: First, DECOMP has a modular structure, and each sub-task is processed by a dedicated processor. The sub-task processor can be independently optimized, debugged, and upgraded. If necessary, the sub-tasks that are more difficult for LLMs can be further decomposed. The task can be recursively decomposed into the same task with smaller inputs. The symbolic information retrieval module can be easily integrated within the decomposition framework to improve the performance on long-text multi-hop question answering and open-domain multi-hop question answering tasks. Second, DECOMP can improve previous work by simply adding an error correction sub-task processor as a post-processing step. Finally, the sub-task processor can be shared between different tasks to improve the efficiency of task-solving.

Question tree structure enables complex reasoning in knowledge-intensive, multi-hop question answering by decomposing complex questions into hierarchical nodes, facilitating global reasoning, and avoiding negative retrieval through retrieving and aggregating answers from different knowledge sources at each node. Cao et al. [69] proposed probabilistic tree-of-thought reasoning (ProbTree). First, LLMs transform a complex question into a query tree, where each non-root node represents a sub-question of its parent node, and leaf nodes are atomic questions that cannot be further decomposed. Then, ProbTree performs probabilistic reasoning on the tree, from leaf nodes to root nodes, and solves the problem by post-order traversal while considering the confidence of question decomposition and answer. For leaf nodes, ProbTree selects more confident answers from closed-book question answering and open-book question answering, avoiding the problem of negative retrieval. For non-leaf nodes, the tree hierarchy is used to obtain information from child nodes for global reasoning, thereby recovering from local errors and alleviating the problem of error propagation. Chu et al. [70] proposed a reasoning framework called beam aggregation reasoning (BeamAggR). Compared with ProbTree, it first performs complementary multi-source reasoning on leaf nodes and then performs fine-grained answer aggregation, which can better perform knowledge collaboration. For non-leaf nodes, BeamAggR enumerates the combination of sub-questions that the node depends on. Then, it selects the most promising prediction through beam aggregation, optimizes the reasoning trajectory, and reduces cascading errors.

4.3. Retriever Validity

4.3.1. Search Granularity

Appropriate search granularity can better meet complex retrieval requirements. Chunk-structured data sources face the problem of balancing the length of the context and the completeness of the semantics. Gao et al. [71] proposed to narrow the semantic gap between questions and answers by adding metadata attachments to the blocks. This includes metadata filtering for retrieving the original document (e.g., page numbers, file names, authors, categories, and timestamps) as well as artificially constructed metadata, such as paragraph summaries and hypothetical questions.

Partitioning the complete database allows the retriever to focus more on key memories and reduce retrieval noise. Wang et al. [72] proposed to divide the external database into multiple partitions. Each partition serves as the basic unit for RAG execution, thereby achieving fine-grained retrieval and focusing on key memories. Through the multi-agent reinforcement learning framework, the two agents, Agent-S and Agent-R, are responsible for selecting appropriate partitions and refining the retrieved memories, respectively. This process further improves text generation performance.

Tree and graph structures are used to organize long document information. Their hierarchical nature allows them to represent complex relationships. This improves information retrieval efficiency and enhances the ability to process complex queries. It also addresses the limitations of fixed-size chunking, which struggles to capture the structure and dependencies of long documents.

In RAG systems, most existing methods only retrieve short, continuous text fragments from the retrieval corpus, which limits the effective utilization of document-level contextual information. This limitation is reflected in the model's inability to capture long-range dependencies and integrate cross-fragment semantics. Sarthi et al. [73] proposed recursive abstractive processing for tree-organized retrieval (RAPTOR), which recursively embeds, clusters, and summarizes text blocks to build a tree structure with different summary levels from the bottom up. RAPTOR retrieves information from this tree during reasoning and integrates information at different levels of abstraction in long documents.

Graph structure can represent complex dependencies between entities well [74]. It helps organize information from multiple sources into coherent and contextual information, significantly reducing the possibility of hallucinations. LightRAG [75] extracts entity nodes and relationship edges from text through LLMs to construct text key–value pairs. After deduplication, it obtains a knowledge graph and dynamically integrates new data through incremental updates. It not only enhances the ability to handle complex queries but also improves the efficiency of data updates and retrieval performance through optimized key–value data structures. LightRAG [75] adopts a two-level retrieval paradigm, where low-level retrieval focuses on retrieving specific entities and their detailed information, while high-level retrieval focuses on aggregating multiple entities and relations to provide insights into broad topics. In the retrieval process, the graph structure and vector representation are combined to extract local and global keywords in the query and match relevant entities and relations by using the few-shot hint LLMs of role assignment, thus achieving comprehensive and efficient retrieval. Experiments on four datasets from the UltraDomain benchmark [76] show that this method outperforms block-based retrieval methods [71,77,78] in handling large-scale token queries and complex queries. To capture the logical relationship between document content and structure, Knowledge Graph Prompting (KGP) [79] implements indexing between multiple documents by constructing a knowledge graph. It treats paragraphs (text content) and structures (e.g., pages and tables) as nodes. The semantic, lexical, or structural relationships between paragraphs are repre-

sented as edges. This approach effectively addresses knowledge retrieval across multiple documents. It answers questions by extracting subgraphs to retrieve relevant information.

In addition, Chen et al. [80] introduced propositions as a new retrieval granularity to strike a balance between reducing noise and providing complete content. A proposition is defined as an atomic expression in text that encapsulates a distinct fact, presented in concise and self-sufficient natural language.

4.3.2. Embedding

By fine-tuning the embedding model to adapt to the domain knowledge of different downstream tasks. Retrieve and plug (REPLUG) [31] prepends the retrieved documents to the input text, feeds it into the frozen black-box language model, and optimizes the retrieval model by using the output of the black-box language model as a supervisory signal. Muennighoff et al. [81] proposed a generative representational instruction tuning (GRIT) method, which distinguishes between generation and embedding tasks through instructions and trains large language models to handle both tasks simultaneously. GRIT reduces the number of forward passes during reasoning through caching operations, thereby improving efficiency. Zhang et al. [82] enhance the inaccessible black-box embedding model by introducing a trainable embedding model. Zhang et al. [82] concatenate the output of the black-box embedding model and the trainable white-box model to form a new embedding function. Different fine-tuning criteria are adopted based on the availability of relevance labels between queries and candidate passages.

4.4. Retrieval Strategies

To prevent unnecessary retrieval from introducing knowledge that conflicts with the internal parameters of the model, it is necessary to determine the necessity of retrieval. Asai et al. [83] evaluate the necessity of retrieval by introducing special retrieve tags. The model is trained to generate these tags by designing specific prompts and demonstrations. Wang et al. [84] proposed a self-knowledge-guided retrieval enhancement (SKR) method. It allows LLMs to refer to the problems they have encountered before and adaptively call external resources when dealing with new problems. SKR collects self-knowledge of training problems by comparing the performance with and without retrieval enhancement. It then proposes strategies such as direct prompting, in-context learning, training a classifier, and nearest neighbor search to detect the self-knowledge corresponding to a problem by referring to the collected training problems. Jiang et al. [85] decide whether external information needs to be retrieved based on the uncertainty in the generation process. If there are low-confidence words, the model will use the sentence as a retrieval query to retrieve relevant documents from the external knowledge base, and regenerate the sentence based on the retrieved information.

Retrieval frequency refers to the frequency of searching the external knowledge base during the generation process, which controls the degree of reliance on the search results and is usually more worthy of consideration than the necessity of the search. There are a variety of multiple search strategies, such as iterative retrieval [86,87], recursive retrieval [88,89], conditional retrieval, adaptive retrieval [90,91], etc.

In iterative retrieval, information is retrieved in multiple steps. Each step builds on the previous ones, making this approach suitable for tasks that require detailed exploration and improvement. Trivedi et al. [86] proposed an iterative retrieval method called IRCoT, in which retrieval and reasoning steps are performed alternately, allowing reasoning to guide retrieval and using retrieval results to improve reasoning. Shao et al. [87] proposed a method called ITER-RETGEN. This method also adopts an iterative strategy, using the generated output of the model as the context of retrieval, to more accurately obtain relevant

knowledge and improve subsequent generation results. The iterative retrieval method solves the limitations of traditional single-shot retrieval methods in multi-step reasoning tasks. Combining dynamic retrieval and reasoning significantly improves the accuracy of retrieval and the performance of downstream question-answering tasks while reducing factual errors in the reasoning chain generated by the model.

Recursive retrieval creates a tree or graph structure of retrieval, which contains retrieval that can call itself and performs well in managing hierarchical information. Chen et al. [89] proposed the MemWalker method, which constructs a tree structure by splitting long text into small paragraphs and recursively generating summary nodes. After receiving the query, the model starts navigating from the root node of the tree, checks the summary content of the child nodes level by level, and decides which child node to enter to continue searching for relevant information through reasoning. When the current node information is insufficient to answer the question, it backtracks until it finds a node containing enough information and generates an answer. The hierarchical summarization and dynamic navigation of the tree structure break through the limitations of the context window of the language model, significantly improve the efficiency and accuracy of long text processing, enhance reasoning ability and interpretability, and support error recovery, making it perform well in long text question-answering tasks. Kang et al. [88] proposed the SURGE framework, which uses a recursive strategy in the knowledge graph to better capture dependencies between entities in context by extracting subgraphs related to the conversation context. The structured organization of the knowledge graph enables the systematic integration of multi-level information, improving the accuracy and contextual relevance of retrieved knowledge. This structured approach facilitates the generation of more coherent and contextually consistent conversational outputs.

Conditional retrieval ensures strict adherence to predefined criteria and is suitable for scenarios with clear rules and fixed requirements. Adaptive retrieval dynamically adjusts retrieval behavior according to current task requirements, contextual information, or model status, providing flexibility in dynamic environments. Jeong et al. [90] proposed Adaptive-RAG, which aims to dynamically adjust the retrieval strategy according to the complexity of the query. Adaptive-RAG uses a small LM as a classifier to predict the complexity label of the query and selects the most suitable method from a simple no-retrieval strategy to a complex multi-step retrieval strategy based on the prediction results. Islam et al. [91] proposed a hybrid adaptive retrieval method that dynamically decides whether to retrieve based on model confidence, balancing retrieval frequency and reasoning speed. Adaptive retrieval optimizes reasoning efficiency by dynamically adjusting the retrieval frequency, especially avoiding unnecessary retrieval overhead when processing simple queries and fully using retrieval information when processing complex queries.

5. Solutions to Generation Deficiency

This section reviews relevant solutions to several key issues in generation deficiency. First, we review improving the relevance of retrieved documents by reducing contextual noise. Second, we investigate methods to balance knowledge conflicts, aiming to coordinate the relationship between external knowledge and internal parameters of the model, to ensure the consistency and factual accuracy of generated content. Third, for the problem of middle curse, we outline ways to compress long texts and train models to extract key information, making full use of context. Finally, this section investigates how to ensure the alignment of generation with query and context, and how to break through the limitations of model capabilities by integrating external resources. These solutions help to comprehensively improve the performance of generation models and reduce deviations and errors in the generation process.

5.1. Reduce Context Noise

We begin with the strategy of reducing the impact of noise by re-ranking to retain more relevant context. RichRAG [57] builds a diverse pool of document candidates based on modeled latent sub-aspects to retrieve external knowledge, which often contains noise. To reduce the noise of retrieved documents, RichRAG re-ranks candidate documents and selects the top k documents as context. The multifaceted queries targeted by RichRAG require rich context, and the users of the information retrieval (IR) model are LLMs, which require the ranking of the ranker to be aligned with the preferences of the LLMs. Based on this, the optimization of the ranker is divided into two stages. The first stage is supervised fine-tuning (SFT), where a coverage utility function is used to create silver targets for training, enhancing the ranker's ability to cover query aspects. The second stage is reinforcement learning, which improves ranking quality and aligns it with LLM preferences using the direct preference optimization algorithm (DPO) [92] and a unilateral significance sampling strategy (US3) to build valuable training samples for stable optimization. Vu et al. [93] proposed a few-shot prompt method, FreshPrompt, which generates a list of evidence by extracting sources, dates, titles, highlighted words, etc., from the results retrieved from the search engine and sorts them from old to new in time, encouraging the model to focus on the latest evidence to reduce noise. Vu et al. [93] also proposed a new dynamic question-answering benchmark, FreshQA, which includes 600 natural questions, covering questions that require rapidly changing knowledge and require refutation of false premises, to measure the performance of LLMs in dealing with questions that require the latest world knowledge.

In addition, context compression is used to filter out irrelevant information. Jiang et al. [94] presented LLMLingua, a coarse-to-fine prompt compression method. It involves a budget controller to maintain semantic integrity under high compression ratios, a token-level iterative compression algorithm to better model the interdependence between compressed contents, and an instruction tuning-based method for distribution alignment between language models. However, the tokenizer differences between the small model and the large model may cause the prompt length to be underestimated. Li [95] proposed a method called Selective Context. It leverages self-information to evaluate the informativeness of vocabulary units (such as sentences, phrases, or tokens) in the context and selectively retains the highly informative content. This approach provides a more compact and efficient context representation without sacrificing the performance of the model on a variety of tasks. Wang et al. [96] proposed to identify useful contexts based on lexical and information-theoretic methods and train a context-filtering model that can filter the retrieved contexts to compress the contexts. Xu et al. [97] presented two compressors—the extractive compressor selects useful sentences from the retrieved documents, and the abstractive compressor generates summaries by synthesizing information from multiple documents. These compressors are trained to improve the performance of LLMs while keeping the prompts concise. Liu et al. [98] proposed summary compression and semantic compression. Summary compression achieves compression through summarization, while semantic compression deletes tags that have little impact on semantics.

However, filtering irrelevant paragraphs and information may also cause some relevant information to be discarded. A more effective solution is to train LLMs to ignore irrelevant context by adding irrelevant context to the training data. Chain-of-note (CoN) [99] uses GPT-4 to collect data from the Natural Questions (NQ) corpus [100] and trains the model in a standard supervised manner to generate continuous reading notes for each retrieved document. This process helps evaluate their relevance to a given question and identify the most critical and reliable information. If all documents are irrelevant, the model

either relies on intrinsic knowledge to answer or answers "Unknown" when the answer cannot be accurately determined.

5.2. Balance Knowledge Conflicts

Context conflict mainly refers to the conflict between the model's internal parameters and the retrieved external knowledge, namely, context–memory conflict [36]. When external knowledge is relevant to the task and has high credibility, it should be aligned with the contextual knowledge [27]. When external knowledge is insufficient or unclear, the model should be encouraged to be skeptical of the context and support parameter knowledge [101,102]. A broader strategy is to treat context and knowledge separately, provide disentangled answers [103], or use context and parameter knowledge to improve the factuality of the answer [104].

Specifically, Zhou et al. [27] reconstructed the question by asking the narrator's point of view through opinion-based prompts and used counterfactual demonstrations to show examples of conflicts with memorized knowledge, jointly improving the model's attention to context. Pan et al. [101] built a misleading information dataset by prompting LLMs, trained a misleading information detection model, and guided LLMs to be more cautious by adding additional warnings or guidance in the prompts. Xu et al. [102] constructed a "Fact to Misinform" (Farm) dataset containing a series of factual questions and their corresponding misleading information, and based on this, tracked the changes in LLMs' beliefs when facing misleading information through multiple rounds of dialogue, revealing the vulnerability of LLMs when facing misleading information. Xu et al. [102] used system prompts to remind LLMs to be vigilant about potential misinformation and list all the supporting evidence before responding. Both of the above methods improve the ability to adhere to factual and parameter information of the model when facing potential misinformation through vigilant prompts. Wang et al. [103] aims to consider LLM's ability to detect conflicts, locate conflicts, and generate different answers based on conflicting information. Wang et al. [103] prompted LLMs to generate entity lists in different subject areas and create conflicts through in-domain named entity substitution and in-domain entity shuffling. The chain-of-thought prompts that generate knowledge decompose context, classify conflicts sentence by sentence, and finally improve the ability to detect conflicts of LLMs. By clearly distinguishing the two knowledge sources and repeating the questions, the model's ability to generate different answers based on the knowledge source is improved. Zhang et al. [104] considered pairing the retrieved paragraphs with LLM-generated paragraphs obtained via specially designed prompts to form compatible pairs to generate the final answer. Zhang et al. [104] first evaluated whether the retrieved paragraphs contained correct evidence and then evaluated whether LLM-generated paragraphs were consistent with the retrieved paragraphs. By screening and matching factual and consistent paragraph pairs, the negative impact of knowledge conflicts on the model is reduced.

5.3. Utilize Middle Information

To address the middle curse issue in LLMs, recent studies have focused on effectively utilizing the middle information within long contexts. These approaches aim to enhance the model's ability to capture and leverage critical information located in the middle portions of the input text, thereby improving overall generation quality. Wang et al. [105] proposed a method called SoftPromptComp, which compresses long text into a concise but semantically rich representation through natural language summarization technology and then integrates it into the input of an LLM through soft prompts. This approach not only reduces the computational overhead by decreasing the volume of text processed but also enhances the model's performance in processing and generating compressed text through the dynamic adaptability of soft prompts. He et al. [44] proposed a method called Position-

Agnostic Multi-step QA (PAM QA), which trains the model to accurately identify and extract key information from long texts through position-independent multi-step reasoning. Specifically, PAM QA reinforces the model's attention to the question through question repetition, assists in filtering relevant information via index prediction, and produces the final output through answer summarization, thereby improving the model's retrieval and summarization performance on long texts. Ravaut et al. [106] analyzed the contextual utilization of LLMs in long text summarization, revealed its bias in processing long inputs, and explored three alternative reasoning methods: hierarchical summarization, incremental summarization, and focus prompt. Hierarchical summarization aggregates information through block summarization and secondary summarization, incremental summarization updates the summary content block by block, and focus prompt guides LLMs to focus on the middle part of the input through prompts. Ravaut et al. [106] also sampled from a long-input summarization dataset to ensure that the key information is in the middle part of the context window, constructing a new evaluation benchmark MiddleSum.

5.4. Alignment Resolution

The alignment problem here mainly focuses on the alignment between context and generation, namely, context faithfulness, and the alignment between query and generation, namely, query faithfulness.

5.4.1. Context Faithfulness

Contextual faithfulness refers to the consistency and relevance of the generated text to the input context in terms of content and logic. Better contextual faithfulness is usually achieved by attributing the generated response to the context. Gao et al. [107] proposed the ALCE framework, which aims to enable LLMs to generate text with citations, thereby improving the accuracy and verifiability of their output. The ALCE framework first retrieves text fragments related to the question from the corpus and then guides LLMs to generate answers and cite supporting fragments through different prompting strategies (such as VANILLA, SUMM, SNIPPET, INLINESEARCH, etc.). Fierro et al. [108] proposed a plan-based text generation method that aims to improve the credibility and attribution quality of generated text by generating citations. The method decomposes the text generation task into a series of questions, which serve as the blueprint for the generated content. The model then generates a summary based on the blueprint and generates citations simultaneously during the generation process to ensure that the generated content is supported by reliable external evidence. These methods address the shortcomings of existing generative models in attribution, such as credibility issues caused by inaccurate or lacking citations. However, prompt-based methods can only provide sentence-level attribution and rely on the internal knowledge of LLMs, which have high computational overhead. Incomplete compliance with instructions by LLMs can also lead to problems such as incorrect citation format, citing non-existent sources, or failing to accurately reflect contextual usage.

Given the inherent defects of prompt-based methods, Ye et al. [109] proposed the AGREE method, which aims to improve the accuracy and credibility of answers by fine-tuning LLMs to cite retrieved document fragments as support when generating answers automatically. AGREE first uses unlabeled queries and natural language inference (NLI) models to automatically generate training data with citations. Then, LLMs are fine-tuned through supervised learning to enable them to generate accurate answers based on retrieved fragments and clearly annotate the citation sources. Finally, in the test phase, AGREE uses an iterative reasoning strategy in the retest phase to allow LLMs to actively retrieve additional information based on self-evaluation results to further optimize the answer. Qi et al. [110] proposed the MIRAGE method, which aims to enhance the credibility of

answer attribution in RAG systems by leveraging model internals. MIRAGE first identifies the words affected by the context in the generated sentences by comparing the changes in the model's predicted distribution with and without context. Then, it uses feature attribution techniques such as gradients to attribute the generation of these sensitive words to specific context words and converts the results into a standard reference format. MIRAGE directly uses the calculation process inside the model to reflect the importance of the input in the generation process, thereby avoiding the inaccuracy and computational overhead that may be caused by external validators.

5.4.2. Query Faithfulness

Query faithfulness refers to how well the generated content aligns with the user's query and its semantic intent. Zhang et al. [111] proposed the MixAlign framework, which aims to address the knowledge alignment problem that occurs when LLMs interact with external knowledge bases—i.e., there is a semantic, contextual, structural, or logical mismatch between the user's question and the retrieved knowledge. MixAlign automatically detects these mismatches through interaction with users and knowledge bases. It generates clarifying questions to obtain further guidance from users, enhancing the model's knowledge utilization performance. Peng et al. [112] proposed a plug-and-play LLM-Augmenter framework for revising prompts through feedback on the responses of LLMs. The framework first integrates the retrieved external knowledge, constructs an evidence chain by linking related entities and pruning irrelevant information, and generates a response based on the evidence chain. It then evaluates the alignment between the response and specific business requirements using rule-based or model-based utility functions, generates feedback information to revise the prompt, and queries the LLM again. This process repeats until the generated response is verified.

5.5. Integrate External Resources

By combining the reasoning capabilities of LLMs with external tools, external resources (computing power, data retrieval capabilities, complex logic processing, etc.) can be effectively utilized to allow the model to adapt to new tasks with only a small number of examples or feedback, and to solve various challenges in complex tasks. Yao et al. [113] proposed ReAct, which aims to combine the reasoning and acting capabilities of LLMs to solve complex language and decision-making tasks. ReAct enables the model to dynamically adjust action plans during the reasoning process by alternately generating reasoning traces and task-related actions and obtaining necessary information to support reasoning through interaction with the external environment. Paranjape et al. [114] proposed the ART method, which solves the reasoning limitations and insufficient tool use of LLMs in dealing with complex tasks through automated multi-step reasoning and tool calls. ART retrieves relevant task examples from the task library as prompts, uses LLMs to generate programs containing intermediate reasoning steps and tool calls, pauses the generation to call the tool and integrate the output, and finally, continues to generate the final answer. In addition, ART also supports human feedback, and users can optimize performance by editing programs or adding tools. To compensate for the deficiencies of LLMs in computation and reasoning, Gao et al. [48] propose program-aided language models (PALs). These employ LLMs to decompose natural language problems and generate programs as intermediate reasoning steps through least-to-most prompting. PALs offload the solution step to a runtime such as a Python interpreter, separating problem decomposition from solution execution.

6. Mitigation After Hallucination Detection

Although the components of RAG have been optimized based on the causes of hallucination to mitigate its occurrence, hallucinations remain inevitable. However, their impact can be reduced through detection and correction.

Active hallucination detection and mitigation techniques detect and correct hallucinations in real-time during the generation process to prevent the accumulation and propagation of hallucinations. Varshney et al. [115] proposed an active hallucination detection and mitigation method. In the detection stage, it identifies the key concepts in the generated sentences and creates verification questions for these concepts. It retrieves relevant knowledge and answers verification questions through web searches and model parameter queries to verify the response knowledge in the generated sentences to detect hallucinations. In the mitigation stage, the retrieved knowledge is used as evidence to repair sentences that may have hallucinations, and the corrected sentences are appended to the generation context. The real-time verification and rectification (EVER) framework proposed by Kang et al. [116] also realizes hallucination detection and mitigation through three stages: generation, verification, and correction. In the verification stage, EVER divides hallucinations into internal hallucinations and external hallucinations, corrects internal hallucinations, and rewrites external hallucinations. The corrected sentences will be verified again. If the external illusion still exists, the system will choose to mark the user to warn or give up the answer according to the task requirements, thereby enhancing the credibility of the output content. EVER also uses the data it generates to build preference data pairs, and further improves the factuality of the model through preference adjustment.

Detecting and editing the output of LLMs is universal in hallucination detection and mitigation for different tasks. Gao et al. [117] proposed the retrofit attribution using the research and revision (RARR) method. RARR is divided into research and revision stages. In the research stage, query generation and evidence retrieval are performed for the responses of LLMs to achieve attribution. In the revision stage, the consistency model detects disagreements, and the editing model corrects the text based on the disagreements and retains the structural and stylistic properties of the original text. RARR also proposed a new metric for the attribution editing task, emphasizing the balance between attribution and preservation. Rawte et al. [20] considered replacing high-entropy words in the generation to mitigate hallucinations. Due to the inaccessibility of many LLMs, Rawte et al. [20] used open-source LLMs to identify high-entropy words and defined the hallucination vulnerability index (HVI) of LLMs. High-entropy words were replaced with LLMs based on lower HVI to mitigate hallucinations in generation. They also reported effective models for detecting and replacing high-entropy words, respectively.

7. Conclusions

In this review, we conducted a comprehensive investigation into the hallucination phenomenon caused by the RAG paradigm, including the causes of hallucinations in each subtask at different stages of the RAG framework and the corresponding hallucination mitigation methods. Additionally, we explored techniques for reducing hallucinations through detection and subsequent mitigation. We also provided a summary of the metrics, datasets, and challenges of representative hallucination mitigation methods in subtasks at different stages in Table 4.

Among the methods we reviewed, even though prompt techniques are widely used, they still have the following limitations that we should be concerned about: (1) Prompt techniques have weak generalization ability, and multiple iterations are required to fine-tune the prompts in practical applications, which may be very time-consuming. (2) Some mitigation methods have restrictions on the length of the prompt [94]. (3) The good

performance of prompt engineering also depends on the powerful capabilities of the embedding model [82] and the generative model [99].

Despite extensive research and significant achievements in hallucination mitigation, hallucinations in retrieval-augmented LLMs remain a highly concerning issue and continue to face multifaceted challenges. For example, current multimodal retrieval-augmented LLMs extend the knowledge sources beyond text to include data from multiple modalities, such as images, audio, video, and code. This expansion enriches the context and supplements textual information, enabling the model to generate more satisfactory responses [118,119]. However, the diversity of data sources also introduces additional factors that can lead to hallucinations. Large language models may struggle to accurately interpret the relationships between entities in images, which further leads to the generation of hallucinations. Therefore, further research into hallucination mitigation strategies for multimodal retrieval-augmented LLMs is necessary. Moreover, malicious external attacks, such as carefully crafted jailbreaking prompts [120] and adversarial prompts [121], can induce the model to generate specific hallucinations. Research into attack and defense strategies for inducing hallucinations represents a promising direction for future work. Finally, we hope that this review provides valuable research materials for the development of more reliable and trustworthy LLMs.

Table 4. Summary of metrics, datasets, and challenges of representative hallucination mitigation methods in different stages of subtasks.

Category	Method	Metric	Dataset	Challenges
Data Source	CAG [54]	EM; Noise Ratio; Document Credibility	HotpotQA [122]; 2WikiMQA [123]; MuSiQue [124]; ASQA [125]; RGB [126]; EvolvTempQA [54]; NewsPolluQA [54]	Reliance on additional training data and annotations; limited scope for external resource integration
Query	RETPO [65]	MRR; NDCG; Recall@k; Clarity; Conciseness; Informativeness	QReCC [62]; TopiOCQA [63]	Limited testing scope; limited prompting methods
	Beam AggR [70]	F1	Bamboogle [127]; MuSiQue [124]; HotpotQA [122]; 2WikiMQA [123]	Multi-source reasoning and probabilistic aggregation raise computational costs; external knowledge is unstructured

Table 4. *Cont.*

Category	Method	Metric	Dataset	Challenges
Retriever	KGP [79]	Acc; EM; F1; PDFTriage; Struct-EM	MuSiQue [124]; HotpotQA [122]; 2WikiMQA [123]; IIRC [128]; PDFTriage [129]	Details not provided
	GRIT [81]	Acc; MAP; AP; V-Measure; nDCG; SC; EM; pass@1; Win Ratio	E5 [130];Tülu2 [131]; MMLU [4]; TyDi QA [132] GSM8K [133]; MTEB [134]; HumanEvalSyn [135]; AlpacaEval [136];	Lack of autonomous retrieval capability; insufficient pre-training data; lengthy GRITLM format
Retrieval Strategy	Adaptive RAG [90]	F1; EM; Acc	SQuAD [137]; NQ [100]; TabularQA [138]; MuSiQue [124]; HotpotQA [122]; 2WikiMQA [123]	Lack of datasets for training query complexity classifier; not good performance of classifier
Context Noise	CoN [99]	EM; F1; Reject Ratio; Acc	NQ [100]; TriviaQA [139]; WebQuestions [140]; RealTimeQA [141]	The sequential generation of reading notes extends response times
Knowledge Conflicts	COMBO [104]	EM; F1	NQ [100]; TriviaQA [139]; HotpotQA [122]; WebQ [140]	Lack of adaptability to multi-hop question answering tasks
Utilize Context	SPC [105]	Processing Time; Acc; Cost; Compression Rate	CNN/Daily Mail [142]; SST-2 [143]; AG News [144]; SQuAD v2.0 [137]	Need to balance efficiency and thoroughness to avoid losing important information through excessive compression
Alignment	Plan-based Text Generation [108]	Correctness; Attribution; ROUGE-L; Answer-ability; AutoAIS; ANLI	AQuAMuSe [145]; NQ [100];	Explore decoder-only models, passage indexing; use questions as additional training signals for retrieval
	Mix-Align [111]	Gold Answer Coverage; Accepted; Hallucination	FuzzyQA [111]	Clarification steps add computational load and time cost; no causal path links question, evidence, and answer

Table 4. *Cont.*

Category	Method	Metric	Dataset	Challenges
External Resource	PAL [48]	Acc	GSM8K [133]; SVAMP [146]; ASDIV [147]; MAWPS [148]; BIG-BenchHard [149];	Details not provided
Mitigation after Detection	Active Detection [115]	Precision; Hallucination Ratio; Recall; Mitigation Success Ratio	HotpotQA [122]; Manual	Details not provided
	Entropy Word [20]	HVI; Hallucination Ratio; Mitigation Success Rate	HILT [20];	Details not provided

Author Contributions: Conceptualization, W.Z. and J.Z.; methodology, W.Z.; investigation, W.Z.; resources, J.Z.; writing—original draft preparation, W.Z.; writing—review and editing, W.Z. and J.Z.; visualization, W.Z.; supervision, J.Z.; funding acquisition, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded Fundamental Research Funds for the Central Universities, No. RF1028623059.

Data Availability Statement: The original contributions presented in this study are included in the article; further inquiries can be directed to the corresponding author.

Acknowledgments: The authors thank the MDPI *Mathematics* editorial team for inviting us to publish this review in this Special Issue.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F.L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. Gpt-4 technical report. *arXiv* **2023**, arXiv:2303.08774.
2. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. Llama: Open and efficient foundation language models. *arXiv* **2023**, arXiv:2302.13971.
3. Team, G.; Anil, R.; Borgeaud, S.; Alayrac, J.B.; Yu, J.; Soricut, R.; Schalkwyk, J.; Dai, A.M.; Hauth, A.; Millican, K.; et al. Gemini: A family of highly capable multimodal models. *arXiv* **2023**, arXiv:2312.11805.
4. Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; Steinhardt, J. Measuring massive multitask language understanding. *arXiv* **2020**, arXiv:2009.03300.
5. Zhang, T.; Ladhak, F.; Durmus, E.; Liang, P.; McKeown, K.; Hashimoto, T.B. Benchmarking large language models for news summarization. *Trans. Assoc. Comput. Linguist.* **2024**, *12*, 39–57. [CrossRef]
6. Yu, F.; Zhang, H.; Tiwari, P.; Wang, B. Natural language reasoning, a survey. *ACM Comput. Surv.* **2024**, *56*, 1–39. [CrossRef]
7. Kim, H.J.; Cho, H.; Kim, J.; Kim, T.; Yoo, K.M.; Lee, S.g. Self-generated in-context learning: Leveraging auto-regressive language models as a demonstration generator. *arXiv* **2022**, arXiv:2206.08082.
8. Morris, M.R. Prompting Considered Harmful. *Commun. ACM* **2024**, *67*, 28–30. [CrossRef]
9. Liu, H.; Xue, W.; Chen, Y.; Chen, D.; Zhao, X.; Wang, K.; Hou, L.; Li, R.; Peng, W. A survey on hallucination in large vision-language models. *arXiv* **2024**, arXiv:2402.00253.
10. Sahoo, P.; Meharia, P.; Ghosh, A.; Saha, S.; Jain, V.; Chadha, A. A comprehensive survey of hallucination in large language, image, video and audio foundation models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*; Association for Computational Linguistics: Miami, FL, USA, 2024; pp. 11709–11724.
11. Alghamdi, G.A.T.K.Z.; Liu, H. Can Knowledge Graphs Reduce Hallucinations in LLMs?: A Survey. *arXiv* **2023**, arXiv:2311.07914.

12. Tonmoy, S.; Zaman, S.; Jain, V.; Rani, A.; Rawte, V.; Chadha, A.; Das, A. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv* **2024**, arXiv:2401.01313.
13. Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Trans. Inf. Syst.* **2023**, *43*, 1–55. [CrossRef]
14. Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; et al. Siren's song in the AI ocean: A survey on hallucination in large language models. *arXiv* **2023**, arXiv:2309.01219.
15. Macpherson, F.; Platchias, D. *Hallucination: Philosophy and Psychology*; EBL-Schweitzer, MIT Press: Cambridge, MA, USA, 2013.
16. Smith, A.L.; Greaves, F.; Panch, T. Hallucination or confabulation? Neuroanatomy as metaphor in large language models. *PLoS Digit. Health* **2023**, *2*, e0000388. [CrossRef]
17. Weinstein, E.A. Linguistic aspects of amnesia and confabulation. In *Principles, Practices, and Positions in Neuropsychiatric Research*; Elsevier: Amsterdam, The Netherlands, 1972; pp. 439–444.
18. Sui, P.; Duede, E.; Wu, S.; So, R.J. Confabulation: The Surprising Value of Large Language Model Hallucinations. *arXiv* **2024**, arXiv:2406.04175.
19. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Madotto, A.; Fung, P. Survey of hallucination in natural language generation. *ACM Comput. Surv.* **2023**, *55*, 1–38. [CrossRef]
20. Rawte, V.; Chakraborty, S.; Pathak, A.; Sarkar, A.; Tonmoy, S.; Chadha, A.; Sheth, A.P.; Das, A. The troubling emergence of hallucination in large language models–an extensive definition, quantification, and prescriptive remediations. *arXiv* **2023**, arXiv:2310.04988.
21. Barnett, S.; Kurniawan, S.; Thudumu, S.; Brannelly, Z.; Abdelrazek, M. Seven failure points when engineering a retrieval augmented generation system. In Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI, Lisbon, Portugal, 14–15 April 2024; pp. 194–199.
22. Shanahan, M.; McDonell, K.; Reynolds, L. Role play with large language models. *Nature* **2023**, *623*, 493–498. [CrossRef]
23. Logan IV, R.L.; Balažević, I.; Wallace, E.; Petroni, F.; Singh, S.; Riedel, S. Cutting down on prompts and parameters: Simple few-shot learning with language models. *arXiv* **2021**, arXiv:2106.13353.
24. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24824–24837.
25. Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic chain of thought prompting in large language models. *arXiv* **2022**, arXiv:2210.03493.
26. Chen, W.; Ma, X.; Wang, X.; Cohen, W.W. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv* **2022**, arXiv:2211.12588.
27. Zhou, W.; Zhang, S.; Poon, H.; Chen, M. Context-faithful prompting for large language models. *arXiv* **2023**, arXiv:2303.11315.
28. Izacard, G.; Lewis, P.; Lomeli, M.; Hosseini, L.; Petroni, F.; Schick, T.; Dwivedi-Yu, J.; Joulin, A.; Riedel, S.; Grave, E. Atlas: Few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.* **2023**, *24*, 1–43.
29. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.t.; Rocktäschel, T.; et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9459–9474.
30. Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; Shoham, Y. In-context retrieval-augmented language models. *Trans. Assoc. Comput. Linguist.* **2023**, *11*, 1316–1331. [CrossRef]
31. Shi, W.; Min, S.; Yasunaga, M.; Seo, M.; James, R.; Lewis, M.; Zettlemoyer, L.; Yih, W.T. Replug: Retrieval-augmented black-box language models. *arXiv* **2023**, arXiv:2301.12652.
32. Lazaridou, A.; Gribovskaya, E.; Stokowiec, W.; Grigorev, N. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv* **2022**, arXiv:2203.05115.
33. Borgeaud, S.; Mensch, A.; Hoffmann, J.; Cai, T.; Rutherford, E.; Millican, K.; Van Den Driessche, G.B.; Lespiau, J.B.; Damoc, B.; Clark, A.; et al. Improving language models by retrieving from trillions of tokens. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 2206–2240.
34. Cuconasu, F.; Trappolini, G.; Siciliano, F.; Filice, S.; Campagnano, C.; Maarek, Y.; Tonellotto, N.; Silvestri, F. The power of noise: Redefining retrieval for rag systems. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, Washington, DC, USA, 14–18 July 2024; pp. 719–729.
35. Zhao, B.; Brumbaugh, Z.; Wang, Y.; Hajishirzi, H.; Smith, N.A. Set the clock: Temporal alignment of pretrained language models. *arXiv* **2024**, arXiv:2402.16797.
36. Xu, R.; Qi, Z.; Guo, Z.; Wang, C.; Wang, H.; Zhang, Y.; Xu, W. Knowledge conflicts for llms: A survey. *arXiv* **2024**, arXiv:2403.08319.
37. Longpre, S.; Perisetla, K.; Chen, A.; Ramesh, N.; DuBois, C.; Singh, S. Entity-based knowledge conflicts in question answering. *arXiv* **2021**, arXiv:2109.05052.
38. Chen, H.T.; Zhang, M.J.; Choi, E. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. *arXiv* **2022**, arXiv:2210.13701.

39. Xie, J.; Zhang, K.; Chen, J.; Lou, R.; Su, Y. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. *arXiv* **2023**, arXiv:2305.13300.
40. Perez, E.; Ringer, S.; Lukošiūtė, K.; Nguyen, K.; Chen, E.; Heiner, S.; Pettit, C.; Olsson, C.; Kundu, S.; Kadavath, S.; et al. Discovering language model behaviors with model-written evaluations. *arXiv* **2022**, arXiv:2212.09251.
41. Turpin, M.; Michael, J.; Perez, E.; Bowman, S. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 74952–74965.
42. Wei, J.; Huang, D.; Lu, Y.; Zhou, D.; Le, Q.V. Simple synthetic data reduces sycophancy in large language models. *arXiv* **2023**, arXiv:2308.03958.
43. Sharma, M.; Tong, M.; Korbak, T.; Duvenaud, D.; Askell, A.; Bowman, S.R.; Cheng, N.; Durmus, E.; Hatfield-Dodds, Z.; Johnston, S.R.; et al. Towards understanding sycophancy in language models. *arXiv* **2023**, arXiv:2310.13548.
44. He, J.; Pan, K.; Dong, X.; Song, Z.; Liu, Y.; Liang, Y.; Wang, H.; Sun, Q.; Zhang, S.; Xie, Z. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv* **2023**, arXiv:2311.09198.
45. Waswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the NIPS 2017, Beach, CA, USA, 7 December 2017.
46. Chen, H.T.; Xu, F.; Arora, S.; Choi, E. Understanding retrieval augmentation for long-form question answering. *arXiv* **2023**, arXiv:2310.12150.
47. Tang, R.; Zhang, X.; Ma, X.; Lin, J.; Ture, F. Found in the middle: Permutation self-consistency improves listwise ranking in large language models. *arXiv* **2023**, arXiv:2310.07712.
48. Gao, L.; Madaan, A.; Zhou, S.; Alon, U.; Liu, P.; Yang, Y.; Callan, J.; Neubig, G. Pal: Program-aided language models. In Proceedings of the International Conference on Machine Learning, PMLR 2023, Honolulu, HI, USA, 23–29 July 2023; pp. 10764–10799.
49. Wang, X.; Yang, Q.; Qiu, Y.; Liang, J.; He, Q.; Gu, Z.; Xiao, Y.; Wang, W. Knowledgpt: Enhancing large language models with retrieval and storage access on knowledge bases. *arXiv* **2023**, arXiv:2308.11761.
50. Yu, W.; Iter, D.; Wang, S.; Xu, Y.; Ju, M.; Sanyal, S.; Zhu, C.; Zeng, M.; Jiang, M. Generate rather than retrieve: Large language models are strong context generators. *arXiv* **2022**, arXiv:2209.10063.
51. Cheng, X.; Luo, D.; Chen, X.; Liu, L.; Zhao, D.; Yan, R. Lift yourself up: Retrieval-augmented text generation with self-memory. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 43780–43799.
52. Black, S.; Biderman, S.; Hallahan, E.; Anthony, Q.; Gao, L.; Golding, L.; He, H.; Leahy, C.; McDonell, K.; Phang, J.; et al. Gpt-neox-20b: An open-source autoregressive language model. *arXiv* **2022**, arXiv:2204.06745.
53. Asai, A.; Zhong, Z.; Chen, D.; Koh, P.W.; Zettlemoyer, L.; Hajishirzi, H.; Yih, W.T. Reliable, adaptable, and attributable language models with retrieval. *arXiv* **2024**, arXiv:2403.03187.
54. Pan, R.; Cao, B.; Lin, H.; Han, X.; Zheng, J.; Wang, S.; Cai, X.; Sun, L. Not All Contexts Are Equal: Teaching LLMs Credibility-aware Generation. *arXiv* **2024**, arXiv:2404.06809.
55. Wang, L.; Yang, N.; Wei, F. Query2doc: Query expansion with large language models. *arXiv* **2023**, arXiv:2303.07678.
56. Jagerman, R.; Zhuang, H.; Qin, Z.; Wang, X.; Bendersky, M. Query expansion by prompting large language models. *arXiv* **2023**, arXiv:2305.03653.
57. Wang, S.; Yu, X.; Wang, M.; Chen, W.; Zhu, Y.; Dou, Z. Richrag: Crafting rich responses for multi-faceted queries in retrieval-augmented generation. *arXiv* **2024**, arXiv:2406.12566.
58. Kim, G.; Kim, S.; Jeon, B.; Park, J.; Kang, J. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. *arXiv* **2023**, arXiv:2310.14696.
59. Ma, X.; Gong, Y.; He, P.; Zhao, H.; Duan, N. Query rewriting for retrieval-augmented large language models. *arXiv* **2023**, arXiv:2305.14283.
60. Mao, S.; Jiang, Y.; Chen, B.; Li, X.; Wang, P.; Wang, X.; Xie, P.; Huang, F.; Chen, H.; Zhang, N. RaFe: Ranking Feedback Improves Query Rewriting for RAG. *arXiv* **2024**, arXiv:2405.14431.
61. Qu, C.; Yang, L.; Chen, C.; Qiu, M.; Croft, W.B.; Iyyer, M. Open-retrieval conversational question answering. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Xi'an, China, 25–30 July 2020; pp. 539–548.
62. Anantha, R.; Vakulenko, S.; Tu, Z.; Longpre, S.; Pulman, S.; Chappidi, S. Open-domain question answering goes conversational via question rewriting. *arXiv* **2020**, arXiv:2010.04898.
63. Adlakha, V.; Dhuliawala, S.; Suleman, K.; de Vries, H.; Reddy, S. Topiocqa: Open-domain conversational question answering with topic switching. *Trans. Assoc. Comput. Linguist.* **2022**, *10*, 468–483. [CrossRef]
64. Choi, E.; Palomaki, J.; Lamm, M.; Kwiatkowski, T.; Das, D.; Collins, M. Decontextualization: Making sentences stand-alone. *Trans. Assoc. Comput. Linguist.* **2021**, *9*, 447–461. [CrossRef]
65. Yoon, C.; Kim, G.; Jeon, B.; Kim, S.; Jo, Y.; Kang, J. Ask Optimal Questions: Aligning Large Language Models with Retriever's Preference in Conversational Search. *arXiv* **2024**, arXiv:2402.11827.

66. Shao, Y.; Jiang, Y.; Kanell, T.A.; Xu, P.; Khattab, O.; Lam, M.S. Assisting in writing wikipedia-like articles from scratch with large language models. *arXiv* **2024**, arXiv:2402.14207.
67. Zhou, D.; Schärli, N.; Hou, L.; Wei, J.; Scales, N.; Wang, X.; Schuurmans, D.; Cui, C.; Bousquet, O.; Le, Q.; et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv* **2022**, arXiv:2205.10625.
68. Khot, T.; Trivedi, H.; Finlayson, M.; Fu, Y.; Richardson, K.; Clark, P.; Sabharwal, A. Decomposed prompting: A modular approach for solving complex tasks. *arXiv* **2022**, arXiv:2210.02406.
69. Cao, S.; Zhang, J.; Shi, J.; Lv, X.; Yao, Z.; Tian, Q.; Li, J.; Hou, L. Probabilistic Tree-of-thought Reasoning for Answering Knowledge-intensive Complex Questions. *arXiv* **2023**, arXiv:2311.13982.
70. Chu, Z.; Chen, J.; Chen, Q.; Wang, H.; Zhu, K.; Du, X.; Yu, W.; Liu, M.; Qin, B. BeamAggR: Beam Aggregation Reasoning over Multi-source Knowledge for Multi-hop Question Answering. *arXiv* **2024**, arXiv:2406.19820.
71. Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H. Retrieval-augmented generation for large language models: A survey. *arXiv* **2023**, arXiv:2312.10997.
72. Wang, Z.; Teo, S.X.; Ouyang, J.; Xu, Y.; Shi, W. M-RAG: Reinforcing Large Language Model Performance through Retrieval-Augmented Generation with Multiple Partitions. *arXiv* **2024**, arXiv:2405.16420.
73. Sarthi, P.; Abdullah, S.; Tuli, A.; Khanna, S.; Goldie, A.; Manning, C.D. Raptor: Recursive abstractive processing for tree-organized retrieval. *arXiv* **2024**, arXiv:2401.18059.
74. Rampášek, L.; Galkin, M.; Dwivedi, V.P.; Luu, A.T.; Wolf, G.; Beaini, D. Recipe for a general, powerful, scalable graph transformer. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 14501–14515.
75. Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; Huang, C. Lightrag: Simple and fast retrieval-augmented generation. *arXiv* **2024**, arXiv:2410.05779.
76. Qian, H.; Zhang, P.; Liu, Z.; Mao, K.; Dou, Z. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv* **2024**, arXiv:2409.05591.
77. Gao, L.; Ma, X.; Lin, J.; Callan, J. Precise zero-shot dense retrieval without relevance labels. *arXiv* **2022**, arXiv:2212.10496.
78. Chan, C.M.; Xu, C.; Yuan, R.; Luo, H.; Xue, W.; Guo, Y.; Fu, J. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv* **2024**, arXiv:2404.00610.
79. Wang, Y.; Lipka, N.; Rossi, R.A.; Siu, A.; Zhang, R.; Derr, T. Knowledge graph prompting for multi-document question answering. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 26 February 2024; Volume 38, pp. 19206–19214.
80. Chen, T.; Wang, H.; Chen, S.; Yu, W.; Ma, K.; Zhao, X.; Zhang, H.; Yu, D. Dense x retrieval: What retrieval granularity should we use? *arXiv* **2023**, arXiv:2312.06648.
81. Muennighoff, N.; Su, H.; Wang, L.; Yang, N.; Wei, F.; Yu, T.; Singh, A.; Kiela, D. Generative representational instruction tuning. *arXiv* **2024**, arXiv:2402.09906.
82. Zhang, M.; Lan, S.; Hayes, P.; Barber, D. Mafin: Enhancing Black-Box Embeddings with Model Augmented Fine-tuning. *arXiv* **2024**, arXiv:2402.12177.
83. Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; Hajishirzi, H. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv* **2023**, arXiv:2310.11511.
84. Wang, Y.; Li, P.; Sun, M.; Liu, Y. Self-knowledge guided retrieval augmentation for large language models. *arXiv* **2023**, arXiv:2310.05002.
85. Jiang, Z.; Xu, F.F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; Neubig, G. Active retrieval augmented generation. *arXiv* **2023**, arXiv:2305.06983.
86. Trivedi, H.; Balasubramanian, N.; Khot, T.; Sabharwal, A. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv* **2022**, arXiv:2212.10509.
87. Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; Chen, W. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv* **2023**, arXiv:2305.15294.
88. Kang, M.; Kwak, J.M.; Baek, J.; Hwang, S.J. Knowledge graph-augmented language models for knowledge-grounded dialogue generation. *arXiv* **2023**, arXiv:2305.18846.
89. Chen, H.; Pasunuru, R.; Weston, J.; Celikyilmaz, A. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv* **2023**, arXiv:2310.05029.
90. Jeong, S.; Baek, J.; Cho, S.; Hwang, S.J.; Park, J.C. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv* **2024**, arXiv:2403.14403.
91. Islam, S.B.; Rahman, M.A.; Hossain, K.; Hoque, E.; Joty, S.; Parvez, M.R. OPEN-RAG: Enhanced Retrieval-Augmented Reasoning with Open-Source Large Language Models. *arXiv* **2024**, arXiv:2410.01782.

92. Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C.D.; Ermon, S.; Finn, C. Direct preference optimization: Your language model is secretly a reward model. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 53728–53741.
93. Vu, T.; Iyyer, M.; Wang, X.; Constant, N.; Wei, J.; Wei, J.; Tar, C.; Sung, Y.H.; Zhou, D.; Le, Q.; et al. Freshllms: Refreshing large language models with search engine augmentation. *arXiv* **2023**, arXiv:2310.03214.
94. Jiang, H.; Wu, Q.; Lin, C.Y.; Yang, Y.; Qiu, L. Llmlingua: Compressing prompts for accelerated inference of large language models. *arXiv* **2023**, arXiv:2310.05736.
95. Li, Y. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering. *arXiv* **2023**, arXiv:2304.12102.
96. Wang, Z.; Araki, J.; Jiang, Z.; Parvez, M.R.; Neubig, G. Learning to filter context for retrieval-augmented generation. *arXiv* **2023**, arXiv:2311.08377.
97. Xu, F.; Shi, W.; Choi, E. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv* **2023**, arXiv:2310.04408.
98. Liu, J.; Li, L.; Xiang, T.; Wang, B.; Qian, Y. Tcra-llm: Token compression retrieval augmented large language model for inference cost reduction. *arXiv* **2023**, arXiv:2310.15556.
99. Yu, W.; Zhang, H.; Pan, X.; Ma, K.; Wang, H.; Yu, D. Chain-of-note: Enhancing robustness in retrieval-augmented language models. *arXiv* **2023**, arXiv:2311.09210.
100. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. Natural questions: A benchmark for question answering research. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 453–466. [CrossRef]
101. Pan, Y.; Pan, L.; Chen, W.; Nakov, P.; Kan, M.Y.; Wang, W.Y. On the risk of misinformation pollution with large language models. *arXiv* **2023**, arXiv:2305.13661.
102. Xu, R.; Lin, B.S.; Yang, S.; Zhang, T.; Shi, W.; Zhang, T.; Fang, Z.; Xu, W.; Qiu, H. The Earth is Flat because...: Investigating LLMs' Belief towards Misinformation via Persuasive Conversation. *arXiv* **2023**, arXiv:2312.09085.
103. Wang, Y.; Feng, S.; Wang, H.; Shi, W.; Balachandran, V.; He, T.; Tsvetkov, Y. Resolving knowledge conflicts in large language models. *arXiv* **2023**, arXiv:2310.00935.
104. Zhang, Y.; Khalifa, M.; Logeswaran, L.; Lee, M.; Lee, H.; Wang, L. Merging generated and retrieved knowledge for open-domain QA. *arXiv* **2023**, arXiv:2310.14393.
105. Wang, C.; Yang, Y.; Li, R.; Sun, D.; Cai, R.; Zhang, Y.; Fu, C. Adapting llms for efficient context processing through soft prompt compression. In Proceedings of the International Conference on Modeling, Natural Language Processing and Machine Learning, Xi'an, China, 17–19 May 2024; pp. 91–97.
106. Ravaut, M.; Joty, S.; Sun, A.; Chen, N.F. On position bias in summarization with large language models. *arXiv* **2023**, arXiv:2310.10570.
107. Gao, T.; Yen, H.; Yu, J.; Chen, D. Enabling large language models to generate text with citations. *arXiv* **2023**, arXiv:2305.14627.
108. Fierro, C.; Amplayo, R.K.; Huot, F.; De Cao, N.; Maynez, J.; Narayan, S.; Lapata, M. Learning to Plan and Generate Text with Citations. *arXiv* **2024**, arXiv:2404.03381.
109. Ye, X.; Sun, R.; Arik, S.Ö.; Pfister, T. Effective large language model adaptation for improved grounding. *arXiv* **2023**, arXiv:2311.09533.
110. Qi, J.; Sarti, G.; Fernández, R.; Bisazza, A. Model Internals-based Answer Attribution for Trustworthy Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2406.13663.
111. Zhang, S.; Pan, L.; Zhao, J.; Wang, W.Y. The knowledge alignment problem: Bridging human and external knowledge for large language models. *arXiv* **2023**, arXiv:2305.13669.
112. Peng, B.; Galley, M.; He, P.; Cheng, H.; Xie, Y.; Hu, Y.; Huang, Q.; Liden, L.; Yu, Z.; Chen, W.; et al. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv* **2023**, arXiv:2302.12813.
113. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. React: Synergizing reasoning and acting in language models. *arXiv* **2022**, arXiv:2210.03629.
114. Paranjape, B.; Lundberg, S.; Singh, S.; Hajishirzi, H.; Zettlemoyer, L.; Ribeiro, M.T. Art: Automatic multi-step reasoning and tool-use for large language models. *arXiv* **2023**, arXiv:2303.09014.
115. Varshney, N.; Yao, W.; Zhang, H.; Chen, J.; Yu, D. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv* **2023**, arXiv:2307.03987.
116. Kang, H.; Ni, J.; Yao, H. Ever: Mitigating hallucination in large language models through real-time verification and rectification. *arXiv* **2023**, arXiv:2311.09114.
117. Gao, L.; Dai, Z.; Pasupat, P.; Chen, A.; Chaganty, A.T.; Fan, Y.; Zhao, V.Y.; Lao, N.; Lee, H.; Juan, D.C.; et al. Rarr: Researching and revising what language models say, using language models. *arXiv* **2022**, arXiv:2210.08726.
118. Hu, Z.; Iscen, A.; Sun, C.; Wang, Z.; Chang, K.W.; Sun, Y.; Schmid, C.; Ross, D.A.; Fathi, A. Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 23369–23379.

119. Zhu, Y.; Ren, C.; Xie, S.; Liu, S.; Ji, H.; Wang, Z.; Sun, T.; He, L.; Li, Z.; Zhu, X.; et al. REALM: RAG-Driven Enhancement of Multimodal Electronic Health Records Analysis via Large Language Models. *arXiv* **2024**, arXiv:2402.07016.
120. Wei, A.; Haghtalab, N.; Steinhardt, J. Jailbroken: How does llm safety training fail? *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 80079–80110.
121. Paulus, A.; Zharmagambetov, A.; Guo, C.; Amos, B.; Tian, Y. Advprompter: Fast adaptive adversarial prompting for llms. *arXiv* **2024**, arXiv:2404.16873.
122. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv* **2018**, arXiv:1809.09600.
123. Ho, X.; Nguyen, A.K.D.; Sugawara, S.; Aizawa, A. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv* **2020**, arXiv:2011.01060.
124. Trivedi, H.; Balasubramanian, N.; Khot, T.; Sabharwal, A. MuSiQue: Multihop Questions via Single-hop Question Composition. *Trans. Assoc. Comput. Linguist.* **2022**, *10*, 539–554. [CrossRef]
125. Stelmakh, I.; Luan, Y.; Dhingra, B.; Chang, M.W. ASQA: Factoid questions meet long-form answers. *arXiv* **2022**, arXiv:2204.06092.
126. Chen, J.; Lin, H.; Han, X.; Sun, L. Benchmarking large language models in retrieval-augmented generation. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 26–27 February 2024; Volume 38, pp. 17754–17762.
127. Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N.A.; Lewis, M. Measuring and narrowing the compositionality gap in language models. *arXiv* **2022**, arXiv:2210.03350.
128. Ferguson, J.; Gardner, M.; Hajishirzi, H.; Khot, T.; Dasigi, P. IIRC: A dataset of incomplete information reading comprehension questions. *arXiv* **2020**, arXiv:2011.07127.
129. Saad-Falcon, J.; Barrow, J.; Siu, A.; Nenkova, A.; Yoon, D.S.; Rossi, R.A.; Dernoncourt, F. Pdftriage: Question answering over long, structured documents. *arXiv* **2023**, arXiv:2309.08872.
130. Wang, L.; Yang, N.; Huang, X.; Yang, L.; Majumder, R.; Wei, F. Improving text embeddings with large language models. *arXiv* **2023**, arXiv:2401.00368.
131. Ivison, H.; Wang, Y.; Pyatkin, V.; Lambert, N.; Peters, M.; Dasigi, P.; Jang, J.; Wadden, D.; Smith, N.A.; Beltagy, I.; et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv* **2023**, arXiv:2311.10702.
132. Clark, J.H.; Choi, E.; Collins, M.; Garrette, D.; Kwiatkowski, T.; Nikolaev, V.; Palomaki, J. Tydi qa: A benchmark for information-seeking question answering in ty pologically di verse languages. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 454–470. [CrossRef]
133. Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. Training verifiers to solve math word problems. *arXiv* **2021**, arXiv:2110.14168.
134. Muennighoff, N.; Tazi, N.; Magne, L.; Reimers, N. MTEB: Massive text embedding benchmark. *arXiv* **2022**, arXiv:2210.07316.
135. Muennighoff, N.; Liu, Q.; Zebaze, A.; Zheng, Q.; Hui, B.; Zhuo, T.Y.; Singh, S.; Tang, X.; Von Werra, L.; Longpre, S. Octopack: Instruction tuning code large language models. In Proceedings of the NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following, Montréal, QC, Canada, 8–10 August 2023.
136. Li, X.; Zhang, T.; Dubois, Y.; Taori, R.; Gulrajani, I.; Guestrin, C.; Liang, P.; Hashimoto, T.B. *Alpacaeval: An Automatic Evaluator of Instruction-Following Models*; GitHub: San Francisco, CA, USA, 2023.
137. Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. Squad: 100,000+ questions for machine comprehension of text. *arXiv* **2016**, arXiv:1606.05250.
138. Gupta, V.; Mehta, M.; Nokhiz, P.; Srikumar, V. INFOTABS: Inference on tables as semi-structured data. *arXiv* **2020**, arXiv:2005.06117.
139. Joshi, M.; Choi, E.; Weld, D.S.; Zettlemoyer, L. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv* **2017**, arXiv:1705.03551.
140. Berant, J.; Chou, A.; Frostig, R.; Liang, P. Semantic parsing on freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WD, USA, 18–21 October 2013; pp. 1533–1544.
141. Kasai, J.; Sakaguchi, K.; Le Bras, R.; Asai, A.; Yu, X.; Radev, D.; Smith, N.A.; Choi, Y.; Inui, K. Realtime qa: What's the answer right now? *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 49025–49043.
142. Nallapati, R.; Zhou, B.; Gulcehre, C.; Xiang, B. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv* **2016**, arXiv:1602.06023.
143. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.Y.; Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
144. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. *Adv. Neural Inf. Process. Syst.* **2015**, *28*.
145. Kulkarni, S.; Chammas, S.; Zhu, W.; Sha, F.; Ie, E. Aquamuse: Automatically generating datasets for query-based multi-document summarization. *arXiv* **2020**, arXiv:2010.12694.
146. Patel, A.; Bhattamishra, S.; Goyal, N. Are NLP models really able to solve simple math word problems? *arXiv* **2021**, arXiv:2103.07191.
147. Miao, S.Y.; Liang, C.C.; Su, K.Y. A diverse corpus for evaluating and developing English math word problem solvers. *arXiv* **2021**, arXiv:2106.15772.

148. Koncel-Kedziorski, R.; Roy, S.; Amini, A.; Kushman, N.; Hajishirzi, H. MAWPS: A math word problem repository. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 1152–1157.
149. Suzgun, M.; Scales, N.; Schärli, N.; Gehrmann, S.; Tay, Y.; Chung, H.W.; Chowdhery, A.; Le, Q.V.; Chi, E.H.; Zhou, D.; et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv* **2022**, arXiv:2210.09261.