

Data Engineering



Letters

Letter from the Editor-in-Chief.....	<i>Haixun Wang</i>	1
Letter from the Special Issue Editors.....	<i>Xin Luna Dong¹, Meng Jiang², Kai Sun¹, Xiao Yang¹</i>	2

Opinions

Task-Focused Information Retrieval in the Generative AI Era.....	<i>Chirag Shah, Ryen W. White</i>	3
------------------------------------------------------------------	-----------------------------------	---

Special Issue on Retrieval-Augmented Generation

Knowledge Graph and Large Language Model Co-learning via Structure-oriented Retrieval Augmented Generation.....	<i>Carl Yang, Ran Xu, Linhao Luo, Shirui Pan</i>	9
Retrieval Augmented Generation in the Wild: A <u>System 2</u> Perspective.....	<i>Sajjadur Rahman, Dan Zhang, Nikita Bhutani, Estevam Hruschka, Eser Kandogan</i>	47
RAG-based Question Answering over Heterogeneous Data and Text..	<i>Philipp Christmann, Gerhard Weikum</i>	71
Evaluating the Factuality of Large Language Models using Large-Scale Knowledge Graphs.....	<i>Xiaoze Liu, Feijie Wu, Tianyang Xu, Zhuo Chen, Yichi Zhang, Xiaoqian Wang, Jing Gao</i>	87
Increasing Accuracy of LLM-powered Question Answering on SQL databases: Knowledge Graphs to the Rescue.....	<i>Juan Sequeda, Dean Allemang, Bryon Jacob</i>	109
Symphony: Towards Trustworthy Question Answering and Verification using RAG over Multimodal Data Lakes.....	<i>Nan Tang, Chenyu Yang, Zhengxuan Zhang, Yuyu Luo, Ju Fan, Lei Cao, Sam Madden, Alon Halevy</i>	135
REFIT: Reranker Relevance Feedback during Inference.....	<i>Revanth Gangi Reddy, Pradeep Dasigi, Md Arafat Sultan, Arman Cohan, Avirup Sil, Heng Ji, Hannaneh Hajishirzi</i>	147
KDD Cup CRAG competition: Systems, Findings and Learnings.....	<i>Xiao Yang, Yifan Ethan Xu, Kai Sun, Jiaqi Wang, Lingkun Kong, Wen-tau Yih, Xin Luna Dong</i>	163

Conference and Journal Notices

TCDE Membership Form.....		183
---------------------------	--	-----

Editorial Board

Editor-in-Chief

Haixun Wang
EvenUp
haixun.wang@evenup.ai

Associate Editors

Xiaokui Xiao
National University of Singapore
Singapore

Steven Euijong Whang
KAIST
Daejeon, Korea

Themis Palpanas
University of Paris
Paris, France

Xin Luna Dong
Meta (Facebook)
Menlo Park, California, USA

Production Editor

Narendra Choudhary
Amazon

Distribution

Brookes Little
IEEE Computer Society
10662 Los Vaqueros Circle
Los Alamitos, CA 90720
eblittle@computer.org

The TC on Data Engineering

Membership in the TC on Data Engineering is open to all current members of the IEEE Computer Society who are interested in database systems. The TCDE web page is <http://tab.computer.org/tcde/index.html>.

The Data Engineering Bulletin

The Bulletin of the Technical Community on Data Engineering is published quarterly and is distributed to all TC members. Its scope includes the design, implementation, modeling, theory and application of database systems and technology.

Letters, conference information, and news should be sent to the Editor-in-Chief. Papers for each issue are solicited by and should be sent to Associate Editor responsible for the issue.

Opinions expressed in contributions are those of the authors and do not necessarily reflect the positions of TC on Data Engineering, IEEE Computer Society, or authors' organizations.

The Data Engineering Bulletin web site is at http://tab.computer.org/tcde/bull_about.html.

TCDE Executive Committee

Chair

Murat Kantarcioglu
University of Texas at Dallas

Executive Vice-Chair

Karl Aberer
EPFL

Executive Vice-Chair

Thomas Risse
Goethe University Frankfurt

Vice Chair

Erich J. Neuhold
University of Vienna, Austria

Vice Chair

Malu Castellanos
Teradata Aster

Vice Chair

Xiaofang Zhou
The University of Queensland

Editor-in-Chief of Data Engineering Bulletin

Haixun Wang
Instacart

Diversity & Inclusion and Awards Program

Coordinator

Amr El Abbadi
University of California, Santa Barbara

Chair Awards Committee

S Sudarshan
IIT Bombay, India

Membership Promotion

Guoliang Li
Tsinghua University

TCDE Archives

Wookey Lee
INHA University

Advisor

Masaru Kitsuregawa
The University of Tokyo

SIGMOD Liaison

Fatma Ozcan
Google, USA

Letter from the Editor-in-Chief

In this issue, we begin with a timely opinion piece by Chirag Shah and Ryen White summarizing a recent Microsoft workshop on task-focused information retrieval in the era of generative AI. The workshop brought together diverse participants to explore how generative AI is transforming information access and task completion, highlighting the pressing need for advances in this rapidly evolving field.

As information systems evolve to meet these emerging challenges, Retrieval-Augmented Generation (RAG) has emerged as a promising paradigm shift in information retrieval, offering a novel approach to accessing and synthesizing knowledge from heterogeneous data sources and formats. By combining the precision of retrieval systems with the generative capabilities of large language models (LLMs), RAG is uniquely positioned to address the challenges of modern data management and information retrieval. Its potential to seamlessly integrate text, structured data, images, and other modalities sets the stage for breakthroughs in answering complex queries, contextualizing information, and providing deeper insights across diverse domains. As the data landscape continues to evolve, RAG's ability to adapt and process such multifaceted information will drive significant innovation in both information retrieval (IR) and database management systems (DBMS).

We extend our gratitude to Dr. Luna Dong for curating this special issue, which features high-quality contributions from leading researchers in the field. The papers selected for this issue collectively showcase the transformative power of RAG in addressing real-world data challenges. From enabling the construction and enhancement of knowledge graphs to refining trustworthiness in question answering, these contributions highlight the field's ongoing commitment to pushing the boundaries of what RAG can achieve. Dr. Dong's thoughtful curation has ensured a comprehensive exploration of RAG's applications, innovations, and challenges, providing a valuable resource for researchers and practitioners alike.

The papers in this issue address key questions in the development of RAG systems, with contributions spanning several domains. For instance, the paper on RAG-based question answering over heterogeneous data highlights techniques for unifying structured and unstructured information to handle complex queries. Symphony demonstrates a robust framework for trust and verification, particularly in multimodal data lakes, while ReFIT introduces mechanisms to incorporate user feedback during inference to improve relevance. Across these contributions, there is a shared focus on enhancing RAG's ability to address multi-step queries, which require reasoning, reflection, and "System 2" approaches. These developments also tackle critical issues like reducing hallucination and improving the interpretability of LLM-powered systems, making strides toward reliable and trustworthy AI applications.

Haixun Wang
EvenUp

Letter from the Special Issue Editors

How convenient would it be to have an AI assistant to share the latest updates for your favorite sports team, or to present you the recent sales trend for your business with a single natural language query. Retrieval Augmented Generation (RAG), bringing latest and targeted information into Large Language Models (LLMs), makes this desire come true.

How to build a RAG system for trustworthy Question Answering (QA)? Some of the challenges that we need to solve are the following. First, we shall evaluate and debug RAG systems to draw insights and iterate quickly. How to measure the quality of RAG systems? How to conduct evaluation with minimum manual efforts and affordable costs? How to do error attribution easily so that we can understand the system and iterate development quickly? Second, we need to conduct retrieval effectively and efficiently. For a given query, shall we trigger retrieval or leave it to the LLM to answer? How do we retrieve information effectively from different sources: unstructured text, structured text, multi-modal data? How to handle ambiguity and context understanding? And how to achieve all the above within a given latency and computation budget? Last but not the least, we need to be able to use the retrieved information effectively to power answer generation. An ideal RAG system should be able to provide correct answers when useful information is retrieved without adding any hallucination. Practically, this requires selecting and/or ranking relevant information from large volumes of data, oftentimes with the presence of irrelevant, noisy, or even conflicting information, to fit in the context window.

This issue collects a set of papers around RAG shedding lights on how to address the aforementioned challenges. We start with two papers for addressing the first challenge – evaluation: **Yang et al.** provided an overview and reflection for the *KDD Cup 2024 CRAG Challenge*. It discussed the rationale behind the benchmark and challenge design, highlighted the winning solutions, and shared learnings from hosting a large-scale challenge for RAG. **Liu et al.** introduced *GraphEval* for large-scale factuality evaluation. It attempted to address the scalability and domain-agnostic challenges in existing evaluation methods, by integrating KGs for question generation and a lightweight judge model for evaluation. We then present 2 papers focusing on the second challenge – retrieval: **Reddy et al.** introducing *ReFit*, a method to leverage re-ranker to improve retriever’s recall in a Retrieval - Reranking pipeline. **Christmann et al.** presents the *Quasar* system for QA over unstructured text, structured tables, and knowledge graphs, with unified treatment of all sources and innovative design for question understanding and evidence re-ranking. Finally, we offer 4 papers for building RAG systems from different angles and also discussing the third challenge. **Tang et al.** introduced *Symphony*, a system designed for trustworthy QA over multimodal data lakes, aiming to use RAG to improve QA system’s accuracy via reasoning and verification. The next two papers focused on understanding Knowledge Graphs (KGs) with RAG: **Sequeda et al.** tried to understand to what extent KGs can increase the accuracy of LLM-powered QA systems, on SQL databases in particular. **Yang et al.** discuss the *co-learning* of KGs and LLMs, through LLM-aided KG construction, KG-guided LLM enhancement, and knowledge-aware multi-agent federation, emphasizing the RAG paradigm, towards fully utilizing the value of complex data. The last paper is by **Rahman et al.**, which proposed to shift from a “fast, intuitive thinking” system to a “slow, deliberate, analytical thinking” to improve RAG in complex enterprise applications.

Overall, the above papers represent an interesting sample of the ongoing work on the recent research on RAG. We hope that this special issue will further help and inspire the research community in its quest to solve this challenging problem. We would like to thank all the authors for their valuable contributions, as well as Haixun Wang for giving us the opportunity to put together this special issue, and Nurendra Choudhary for his help in its publication.

Xin Luna Dong¹, Meng Jiang², Kai Sun¹, Xiao Yang¹
¹ Meta, ² University of Notre Dame

Task-Focused Information Retrieval in the Generative AI Era

Chirag Shah¹ and Ryen W. White²

¹University of Washington, Seattle, WA, USA

²Microsoft Research, Redmond, WA, USA

chirags@uw.edu, ryenw@microsoft.com

Abstract

Generative Artificial Intelligence (GenAI) is revolutionizing how people access information and how they tackle and complete complex information tasks. This report is a summary of a recent workshop at Microsoft on this important and pressing topic. The event brought together a diverse mix of attendees from different professions and at different career stages for an engaging day of presentations and discussions. The emergent themes are described in detail in this summary.

1 Introduction

The second workshop on “Task-Focused Information Retrieval in the Generative AI Era” was held on September 27, 2024 on Microsoft campus in Redmond, Washington. Around 60 participants from various organizations – academic and industry – and various positions – students, faculty, professionals – from across the United States came together for this one day in discussing issues related to information retrieval and access systems in the context of GenAI, specifically GenAI tools such as Large Language Models (LLMs). More information on the workshop, including the agenda, is available at <https://ir-ai.github.io>.

At the beginning of the workshop, the participants were asked to come up with a set of specific questions or topics pertaining to the larger area of task-focused Information Retrieval (IR) systems in the context of GenAI. Dozens of questions, ideas, and topics were posted on a large whiteboard using sticky notes. Participants then arranged these notes into four broad categories: (1) theory, (2) benchmarks and evaluation, (3) users and user experience, and (4) applications and integration.

For the remainder of the day, we organized breakout sessions where the participants used the notes for the corresponding topics to stem their discussions and expand on their ideas. The groups took notes in a shared document. The following sections summarize the key points from their notes and the discussions.

2 Theory

While there were many threads of discussions on various theoretical constructs in GenAI, such as context, language, and interactions, the groups spent a significant amount of time talking about relearning (updating the model knowledge or capabilities based on new data or feedback), unlearning (removing knowledge learned during training, e.g., for privacy, copyright, etc.), and readjustments for LLMs when it comes to information access. This is particularly needed to address issues of privacy, bias, and toxicity while also providing a more flexible architecture for further learning and refinements. For example, the following approaches were discussed for unlearning in LLMs:

1. **Training the Foundational Model:** This approach was found to be not feasible in most situations due to the need to retrain the model for every data removal request.
2. **Decoding Strategies:** This will involve preventing generation of certain tokens. However, models might find alternative ways to express similar intents.
3. **Guardrails/Censorship:** This idea requires implementing a layer to discourage certain topics and training the LLM to provide more diplomatic answers instead of deleting information.
4. **Reinforcement Learning from Human Feedback (RLHF):** This popular technique to align LLMs with human preferences [1] can be used after initial training to discourage specific concepts/tokens.

Workshop participants also discussed alternative ways to train a foundational model for better, more flexible and nuanced training, e.g.,

1. **Speculative Decoding:** This approach is based on student-teacher concept with a small and large model where they decode tokens sequentially and a large model that verifies tokens as they go. This approach improves efficiency and has been found that it does not affect accuracy [2].
2. **Segmented Corpora:** This approach involves training different segments of a large corpus based on expertise for specific motives.
3. **Multi-agent Auditing:** Use experts to prevent other LLMs from generating unlearned content.
4. **Distributed Models vs. Single/Centralized Model:** Mimic the human neuron system for more efficient inference and storage.
5. **Graph/Network of Models:** Each node is responsible for a specific concept, requiring sufficient common ground for communication.

3 Benchmarks and Evaluation

Two breakout groups focused on issues related to evaluation, datasets, and benchmarks for using GenAI for information access applications. The participants emphasized the importance of reliability and validity in evaluating and benchmarking LLMs. They noted that before establishing benchmarks, it is crucial to ensure both the benchmarks and the LLMs themselves are reliable and valid. This foundation is necessary to address issues of fairness, bias, and equity.

The groups highlighted the need for shared definitions of key terms and discussed how metrics should evolve to be more meaningful within specific tasks. Benchmarks should be context-specific to provide accurate evaluations.

When it comes to business use cases and **personas**, the discussion focused on evaluating personalization effectiveness in relation to human preferences, laws, and values. The participants explored how to structure use cases, noting that product design often uses “personas” to capture diverse user needs. However, it is challenging to cover all user differences with benchmarks, leading to questions about grouping users and assessing personalization without creating echo chambers or experiencing distribution collapse.

The groups also addressed the need for data to perform reliable evaluations. They discussed the scarcity of comprehensive **open-source data** and suggested two solutions: using community data collection and encouraging organizations to release data collaboratively. Maintaining the quality of

human evaluation was another key point. The participants emphasized the importance of context-specific questions to get accurate feedback.

On the topic of **alignments and ethics**, they discussed aligning safety and ethical principles, evaluating alignment success, and maintaining privacy.

Finally, the groups touched on the concept of **knowing**—specifically, how to get models to acknowledge when they do not know something. They suggested including confidence intervals in outputs and having models confirm or paraphrase inputs to improve transparency and reliability.

The discussion also covered the potential to teach LLMs appropriateness through system-level **content moderation**, including parental controls, flags, and guardrails. They considered the importance of reading levels and the classification and generation of documents, noting that higher volumes of content could bypass filters.

Evaluating **appropriateness** was another key topic. The group suggested using personalization algorithms to measure what is appropriate, understanding negative feedback, and utilizing both explicit and implicit user feedback to improve satisfaction. They also mentioned the importance of historical behavior logs and cultural evaluation and alignment, noting that standards change over time.

Context was highlighted as crucial, with understanding intent being particularly challenging due to fuzzy boundaries and user subjectivity. The ability to solve complex queries and provide feedback interfaces for improvement was also discussed, along with fine-tuning for pluralistic alignment.

Finally, the group discussed setting contextual measures and measuring controllability, emphasizing the need for dynamic and temporal evaluation and the ability of LLMs to evaluate higher-level constructs.

4 Users and User Experience

In the breakout groups for discussing users and user experience, participants delved into the intricacies of enhancing user interaction and trust in LLMs. They began by emphasizing the importance of referring to “people” instead of “users” to better capture the human aspect of these interactions.

The conversation then shifted to the **typology of tasks** that these people do, highlighting the need for systems that can effectively respond to various goals and intentions. For instance, assisting someone in learning how to apply for a green card requires a nuanced understanding of their needs, circumstances, and queries.

The usefulness of LLMs was discussed, with a focus on how it depends on both the individual and the system. Understanding the user involves considering the language used in queries, persona/user modeling, and cultural sensitivity. The group debated whether to curate pretraining data for users or to employ post-processing training methods.

Developing robust **user simulators** emerged as a critical point, as current interaction patterns with LLMs are not well-defined. The challenge lies in creating a “good enough” user simulator that accurately reflects real-world interactions.

Participants noted that while users may prefer simpler answers, which can increase the acceptance of LLMs, this preference can also lead to **misinformation**. Balancing user engagement with well-being is crucial.

Extending the issue of misinformation, the discussion steered towards **ethical considerations**. The group explored who controls the data, ownership, and access, questioning whether LLMs should always provide certifiable truths and discussing the broader social responsibilities of these models.

Building **trust** was identified as fundamental. It is essential for LLMs to acknowledge when they do not know something. Using prompts to eliminate out-of-bound questions and effectively conveying uncertainty were highlighted as vital strategies for building trust.

The group also debated the necessity of pseudo-relevance feedback versus using LLMs to formulate queries. They explored whether a new form of **relevance feedback**, more suited to the LLM era, is needed.

Designing systems that provide balanced perspectives and defining **diversity** through user actions were key topics. The group discussed democratizing information access and involving user preferences before deploying models.

Understanding user behaviors and creating accurate **user profiles** were emphasized. The discussion included improving existing user graphs (used to model and analyze connections between users, their activities, and the resources they interact with) and addressing privacy issues related to personalization.

Educating users on how to interact effectively with LLMs was deemed crucial. The group debated the benefits of long description queries and how to capture diverse user preferences to ensure the system aligns with a broad user base.

Personalization was recognized as having inherent risks, such as creating echo chambers. The group discussed whether the default mode should cater to general popular preferences or if users should be nudged with information from diverse contexts.

Addressing the **cold start problem** and the influence of search systems/LLMs on query writing were also key points. The extent to which ideal queries should be dictated by the system was debated.

Throughout the discussion, references to foundational works, such as Robert S. Taylor’s study on question negotiation [3] and information seeking in libraries, and Nicholas J. Belkin’s concept of Anomalous States of Knowledge (ASK) [4], provided a theoretical backdrop.

Establishing trust and creating mechanisms to escape the pitfalls of personalization were emphasized as critical components for the future development of LLMs. The group concluded that ethical practices, user education, and robust evaluation methods are essential for enhancing the effectiveness and reliability of LLMs.

5 Applications and Integration

Finally, we had a breakout group for discussing multifaceted applications and integration of LLMs. They began by comparing the merits of general-purpose LLMs with those fine-tuned for specific tasks, weighing the benefits of versatility against the precision of specialization.

The conversation naturally flowed into the realm of **multi-modal systems**, where information is conveyed through various formats such as text, images, and dynamic presentations. Participants debated the criteria for selecting these modalities, using examples like exploratory search, which might benefit from summaries, reference documents, and diverse outputs. They pondered whether LLMs should generate both text and images or focus solely on summarization, drawing parallels to Wikipedia’s multi-modal approach.

The potential for LLMs to guide users along **learning paths** was another key topic. Designing interactions that support dynamic search was emphasized, contrasting with traditional recommendation systems for movies and music, which can sometimes lead users into uninteresting rabbit holes. Unlike these systems, LLMs require carefully designed feedback mechanisms to ensure relevance and engagement. Learning, they noted, is not just about acquiring information on a specific topic; broader context and serendipity play crucial roles. Multi-modality was seen as particularly beneficial in applications such as claim verification, where processing images alongside text can provide a more comprehensive understanding.

The group also discussed the limitations of chatbots as the primary interface for LLMs, suggesting that generating websites or other content might be more effective in certain contexts. They explored the concept of **mixed-initiative systems**, where the system takes some initiative by being proactive, and

highlighted the challenges of controllability and the unpredictability of outputs when the system acts on behalf of the user.

Operational control and the availability of datasets for training these models were also discussed. Public datasets such as Microsoft’s Common Objects in Context (COCO) [5] were mentioned, but the difficulty of experimenting and obtaining feedback was acknowledged. Risk assessment was highlighted as a critical first step in any LLM application, with accountability extending to all involved in the development process.

Ethical considerations, once again, were a significant part of the discussion. Examples such as the United States Transportation Security Administration’s use of facial recognition and OpenAI’s decision not to roll out emotion detection features in the European Union due to risk illustrated the ethical dilemmas and potential stifling of development. The group debated the use of foundation models for tasks that currently require extensive experimentation and iteration.

The participants then discussed how **effective feedback mechanisms** are essential for refining LLMs. Ideally, models should immediately incorporate feedback, but current practices often involve RLHF or fine-tuning phases. The challenge of maintaining memory across chat sessions and deciding whether feedback should apply to the current session or persist indefinitely was also discussed.

The potential for extreme **personalization** in a privacy-preserving manner was seen as a significant benefit of LLM applications. Participants considered what context should be local versus cloud-based and noted the inconsistency in LLM behavior, which can make it difficult to restrict certain types of responses.

The group noted that there has been a shift in consumer expectations, with some tolerance for LLM errors. However, **reliability** remains an issue, as illustrated by the need for specific output formats in tasks such as the National Institute of Standards and Technology’s Text REtrieval Conference (TREC) and the reluctance to answer certain types of questions.

The group debated whether the creators of LLMs should decide what is appropriate, referencing comprehensive experiments by organizations such as Anthropic. The concern was that a small number of people making content moderation decisions could impact everyone.

Overall, the discussion highlighted the complexities and challenges of integrating LLMs into various applications. Ethical considerations, robust feedback mechanisms, and careful design are essential to ensure effective and reliable user interactions, paving the way for the future development of LLMs.

6 Futures

There is clearly a wealth of opportunity for research in the area of task-focused information retrieval, and information access and use in general, in the era of GenAI. In our forthcoming edited book [6], derived from discussions in the first event in this workshop series (held at Microsoft in 2023) we dive into some of these issues in more depth. However, there are also other issues that are gaining more traction that are covered in this report (e.g., applications and integration), signifying the rapid pace of change, the growing opportunities in this area, and in the case of applications and integration, the realities of deploying GenAI technologies in applications at scale. Information access is essential for an informed citizenry. GenAI can make this access more effective. We hope that this summary is useful and that it inspires researchers and practitioners to engage on some of the topics highlighted, and help to realize the full potential of GenAI to assist with people’s complex information challenges.

Acknowledgment

The workshop was supported by NSF award IIS-2023924, the ACM Special Interest Group on Information Retrieval (SIGIR), and Microsoft Research.

References

- [1] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. arXiv preprint arXiv:1909.08593, 2019.
- [2] Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In International Conference on Machine Learning, pages 19274–19286. PMLR, 2023.
- [3] Robert S Taylor. Question-negotiation and information seeking in libraries. College & research libraries, 29(3):178–194, 1968.
- [4] Nicholas J Belkin. Anomalous states of knowledge as a basis for information retrieval. Canadian journal of information science, 5(1):133–143, 1980.
- [5] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13, pages 740–755. Springer, 2014.
- [6] Ryen W. White and Chirag Shah, editors. Information Access in the Era of Generative AI. Springer Nature, Switzerland, 2025.

Knowledge Graph and Large Language Model Co-learning via Structure-oriented Retrieval Augmented Generation

Carl Yang*, Ran Xu, Linhao Luo, Shirui Pan

Abstract

Recent years have witnessed major technical breakthroughs in AI– facilitated by tremendous data and high-performance computers, large language models (LLMs) have brought disruptive progress to information technology from accessing data to performing analysis. While demonstrating unprecedented capabilities, LLMs have been found unreliable in tasks requiring factual knowledge and rigorous reasoning. Despite recent works discussing the hallucination problem of LLMs, systematic studies on empowering LLMs with the ability to plan, reason, and ground with explicit knowledge are still lacking. On the other hand, real-world data are enormous and complex, coming from different sources and bearing various modalities. Data professionals have spent tremendous efforts collecting and curating countless datasets with different schemas and standards. Transforming the separate datasets into unified knowledge graphs (KGs) can facilitate their integrative analysis and utilization, but these processes would often require strong domain expertise and significant human labor. In this paper, we discuss recent progress and promise in the co-learning of KGs and LLMs, through LLM-aided KG construction, KG-guided LLM enhancement, and knowledge-aware multi-agent federation, particularly emphasizing a structure-oriented retrieval augmented generation (SRAG) paradigm, towards fully utilizing the value of complex data, unleashing the power of generative models, and expediting next-generation trustworthy AI.

1 Introduction

Large language models (LLMs) have reshaped AI research and implementations, with unprecedented capabilities widely shown in various language-related tasks [1–7], bringing humans ever close to general AI [8–10]. Recent research on multi-agent systems has further magnified LLMs’ advantages of *broad knowledge*, *language comprehension*, and *generalizability* through conversational collaborations, showing strong promise for further human-model collaboration for critical applications [11–15]. Recent studies have also revealed the limitations of LLMs regarding their *lack of planning* [16–20], *fuzzy inference* [21–25], and *hallucination* [26–33]. Specifically, in many real-world application scenarios, the lack of accurate planning can be caused by the lack of access to high-quality domain knowledge, especially the rapidly evolving new knowledge; the fuzzy inference nature can lead to difficulties in conducting reliable comprehension and stable predictions for complex questions; and hallucination creating factual errors and misinformation can cause fatal and life-threatening problems in critical applications [34–39].

Knowledge graphs (KGs) have been widely studied across academia and industry, due to their advantages in *storing accurate knowledge*, *facilitating explicit inference*, and *allowing easy editing of the knowledge* [40–43]. However, the creation of KGs relies much on the *standardization of data*, which requires significant schematic designs and human efforts. In many application domains, researchers and professionals have spent decades collecting, processing, and curating various types of data towards the

*Corresponding author (j.carlyang@emory.edu); Department of Computer Science, Emory University, Atlanta, GA 30322, USA

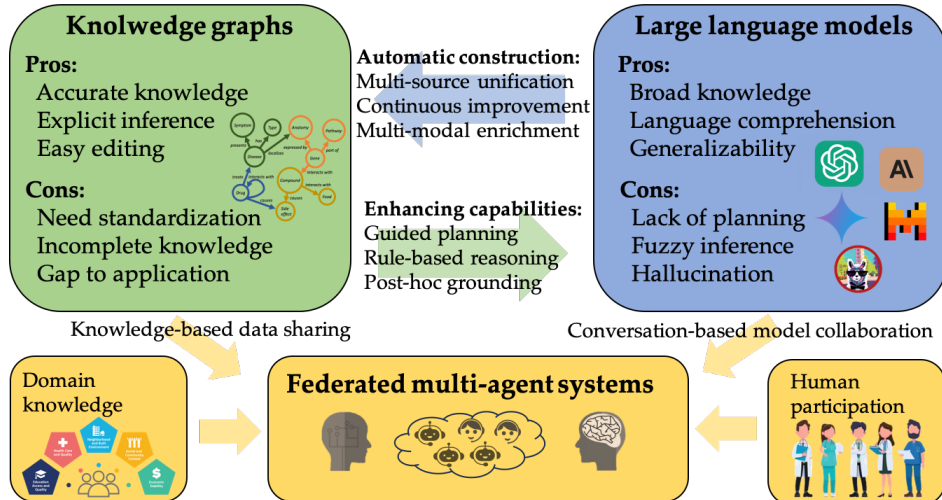


Figure 1: Overview of the proposed knowledge graph and large language model co-learning framework.

construction of KGs [44–53], which are widely used to support downstream application such as search and ranking [54–56], user modeling and recommendation [57–59], basic science research [60–65], healthcare [66–68] and education [69, 70]. Nevertheless, KGs still suffer from *incomplete knowledge coverage*, due to the stringent requirements on data standardization and limited data volumes compared to unstructured data in the wild, while specific algorithms are often needed to fill *their gaps to applications*. Moreover, real-world data are noisy and complex, where entities and relations come from various sources such as institutions using different data schemas and naming conventions [71–73], and the data can also include multiple modalities such as tables, texts, images, and time series [74–85]. While such multi-source and multi-modality data hold great promise in integrative and comprehensive analysis, unifying and extracting high-quality knowledge from them is non-trivial.

Recently, significant research attention has been drawn to the synergies between KGs and LLMs [86–92], due to their naturally complementary advantages (Figure 1). The construction and modeling of KGs have often relied on advances in natural language processing (NLP), with recent efforts intensively exploring language models towards the embedding [93–98], completion [99–111] and construction [112–140] of KGs. Studies in the recent years have also bloomed to explore the utilization of KGs for enhancing LLMs through providing new sources of knowledge during pre-training [141–179] or inference [180–207], and enabling knowledge-based interpretation and evaluation [21, 22, 208–217]. Finally, pioneering studies have also been conducted to explore LLM-based multi-agent systems, mostly through prompt-based role-plays to simulate human collaborations [218–226].

In this paper, we re-emphasize the promise of KG and LLM co-learning, especially through a structure-oriented retrieval augmented generation (SRAG) paradigm, where LLMs extract structured knowledge from unstructured data, which can be further retrieved to enhance the capabilities and reliabilities of LLMs during applications. Specifically, we give examples and discuss several natural and promising use cases where LLMs can be utilized to automate the construction of high-quality KGs. Furthermore, we summarize and highlight several limitations of current LLMs that can be potentially mitigated through the utilization of KGs. Finally, we envision a federated multi-agent system where models and data are disentangled while humans and knowledge are deeply engaged. Future directions are further discussed in the end.

2 LLM-aided KG Construction

In this section, we study and establish the advantages of utilizing LLMs for the construction of KGs, by demonstrating their effectiveness in improving the *accuracy*, *consistency*, *coverage*, and *freshness* of knowledge. Popular KGs such as Freebase [227], Yago [228] and Wikidata [229] contain hundreds of millions of real-world entities like people, places, and things, along with their multi-typed relations. However, since the KGs are collected and curated by different platforms and institutions, they do not use a unified coding system or thesaurus. The varying terminologies due to different conventions or abbreviations can lead to high degrees of duplication and inconsistency when multiple KGs are directly put together. Moreover, the sheer amounts of data in existing KGs are enormous, but the knowledge is still never comprehensive enough to serve various needs of real-world applications, especially those requiring rapidly updated knowledge. Recently, pioneering studies including ours have demonstrated strong promise of utilizing LLMs to automate the construction, integration, and enrichment of KGs [112–122]. In the following, we give several examples of promising attempts of these kinds and discuss more natural use cases of LLMs and multi-modal foundation models (MMFMs) toward constructing high-quality KGs as promising future directions.

2.1 Integrating existing KGs

KG integration, also known as knowledge fusion or knowledge alignment, represents a fundamental challenge in the broader landscape of knowledge engineering, which involves integrating multiple KGs that originate from varied sources and formats [230, 231]. While individual KGs often excel in specific domains or use cases, their true potential can be unlocked through effective integration, enabling more comprehensive and robust knowledge representation [47, 232]. As the number and diversity of KGs continue to grow, the need for effective integration methods becomes increasingly critical.

However, the integration of existing KGs faces several key challenges: (1) *semantic heterogeneity across sources*: Different KGs often use varying terminologies, definitions, and contextual frameworks to represent similar concepts [233]; (2) *varying granularity levels in knowledge representation*: KGs may differ in the detail and depth with which they describe entities and relationships, impacting the consistency and usability of integrated data. Although several neural approaches have been proposed for entity alignment on KGs [234–236], these methods generally depend heavily on labeled data for training. However, obtaining sufficient labeled data often involves substantial manual effort and can be rather costly.

LLMs have emerged as a promising solution to these challenges with unique advantages: First, their strong natural language understanding capabilities enable them to capture semantic relationships among concepts that may be missed by traditional string-matching or embedding-based approaches [109]. Second, LLMs can draw on their extensive knowledge acquired during pre-training to aid in disambiguating entities and mapping relationships across different KGs [237]. Third, LLMs possess robust few-shot learning abilities, making them particularly valuable for specialized domain applications where labeled data are limited [8, 238].

Recent work has demonstrated the effectiveness of LLM-based approaches in KG integration. For example, Lu et al. [239] developed HiPrompt (framework shown in Figure 2), which aligns entities between biomedical KGs and standardized hierarchical entity taxonomies. This task poses significant challenges due to the scarcity of available pairs and the inconsistent naming conventions between KGs and entity taxonomies. Their two-stage approach combines traditional information retrieval techniques (BM25) with LLM-based re-ranking using hierarchy-oriented prompts, achieving superior performance in few-shot biomedical knowledge-graph integration. Building on this direction, Xie et al. [240] developed PromptLink, a framework that leverages both domain-specific language models and GPT-4 for cross-

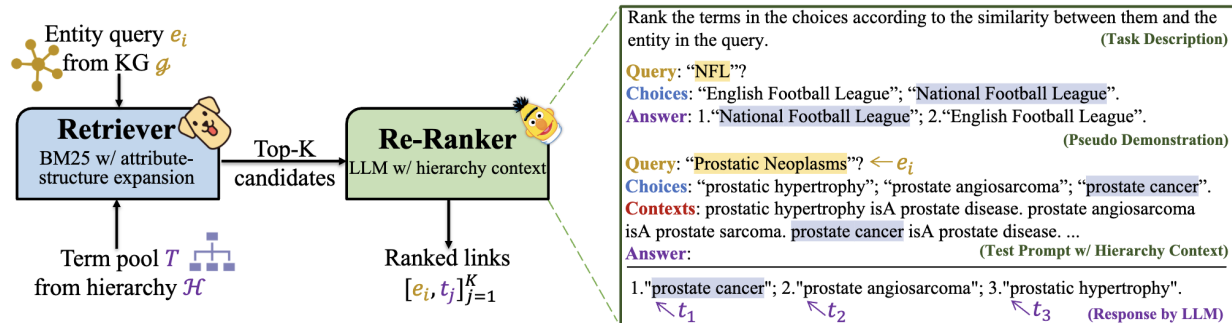


Figure 2: The overall framework of Hiprompt.

source biomedical concept linking. The framework employs a two-stage prompting mechanism by first eliciting the biomedical prior knowledge from the LLM for the concept linking task and then enforcing the LLM to reflect on its own predictions to further enhance their reliability. PromptLink’s success in zero-shot scenarios illustrates the potential of LLMs to generalize across diverse data sources without extensive training data.

In contrast to the two LLM approaches that primarily use LLMs for ranking candidate sets, AutoAlign [241] employs off-the-shelf LLMs to construct a predicate-proximity graph that captures relationships between entity types rather than individual entities. It then aligns the entity embeddings of two KGs into a common vector space by calculating similarity based on entity attributes.

These advances suggest several promising future directions for LLM-aided KG integration. For example, LLMs could potentially facilitate the continuous integration of new knowledge into existing KGs dynamically and automatically by identifying and resolving conflicts between new and existing information, while maintaining consistency across the integrated knowledge base. Such real-time data integration is especially beneficial for dynamic applications like live event monitoring and real-world decision-making. Furthermore, integrating KGs with LLMs remains challenging due to the risk of generating false or untrustworthy information [242]. To mitigate this issue, there is a growing need for improved human-in-the-loop systems. Specifically, enhanced interfaces [243, 244] can enable experts to more effectively verify and interpret LLM-generated recommendations, ensuring greater reliability and transparency.

2.2 Constructing and Completing KGs

KGs have high-standard requirements on the quality of knowledge, regarding accuracy, consistency, coverage and freshness. No matter constructed through manual curation, NLP tools, or their combinations, KGs can unavoidably include erroneous knowledge. Moreover, when multiple KGs are integrated, conflicting knowledge can emerge. Finally, new knowledge is constantly generated from new experiments and research, making existing knowledge inaccurate and incomplete. LLMs have emerged as a promising solution, leveraging the vast and adaptable knowledge acquired during pre-training to overcome these limitations [208, 245–247]. The key advantage of LLMs for KG construction and completion is their ability to generate novel, semantically coherent information with minimal reliance on additional labeled data.

Recent studies have highlighted the effectiveness of LLM-based approaches for KG construction and completion. Zhu et al. [112] utilize in-context learning capabilities of LLMs to complete tuples with missing entities or relations to generate new knowledge triplets for augmenting existing KGs. Wei et al. [248] and Wang et al. [249] tackle KG completion as a candidate identification and ranking task, proposing a “retrieve-rank” pipeline where LLMs are used to rerank top-retrieved entities, thus creating

additional knowledge triplets.

An alternative approach to prompting LLMs involves using code-based prompts, rather than natural language, to incorporate new entities into existing KGs [250, 251]. Code LLMs, extensively trained on structured data such as programming code, are inherently well-suited to the structured nature of KGs. Using a code-based interface enables more effective handling of graph-like structures, logical relationships, and precise reasoning [252–254]. Specifically, Bi et al. [250] first encoded the schema of KGs by modeling code definitions, to capture the structural information inherent in the data. They then employed chain-of-thought prompting to produce accurate knowledge triples. This methodology demonstrates improved performance over traditional natural language-based prompts. Similarly, Zeng et al. [251] proposed CodeTaxo, which represents entities within a base 'Entity' class, mirroring hierarchical relationships in programming constructs. This approach enables LLMs to efficiently create taxonomic structures by leveraging syntactic capabilities commonly used in code tasks, thus enhancing the organization and completeness of KGs.

The above methods primarily focus on prompting LLMs for KG construction and completion. While these methods show promise, they fall short in fully adapting LLMs to target tasks and can suffer from hallucination issues. To address these limitations, several studies aim to improve the quality of LLM-generated content for KG tasks. Zhang et al. [116] presented a three-phase framework for constructing KGs with LLMs to enhance contextual understanding and schema alignment. It starts with open information extraction to identify relation triplets from unlabeled textual corpora, followed by schema definition where LLMs generate contextually relevant descriptions for schema components. Finally, the extracted triplets are aligned with the schema. To further enhance the quality of the extracted triplets and minimize the risk of misinformation, a schema retriever is used to generate a list of candidate entities and relations to guide the triplet extraction steps. This approach works for both predefined schemas and situations where the schema needs to be inferred from the context. Additionally, various studies explored fine-tuning LLMs specifically for KG completion. Zhang et al. [255] proposed KOPA that first applies structural pre-training to create embeddings of entities and relations in KGs, and project embeddings into the textual space as virtual knowledge tokens. These tokens act as prefixes in LLM input prompts, enabling structure-aware reasoning that leverages both the generative power of LLMs and the retrieval of structured KG information to improve the accuracy and completeness of KG completion tasks. Jiang et al. [256] introduced KG-FIT for using open-world knowledge from LLMs to enhance KG embeddings. It initially constructs a semantically coherent, hierarchical structure of entity clusters, guided by LLM-powered entity representations. It then fine-tunes these embeddings by integrating the hierarchical structure with textual embeddings. This hybrid approach allows KG-FIT to capture both the semantic depth of LLMs and the structural information intrinsic to KGs, resulting in more comprehensive KG representations.

2.3 Enriching KGs with multi-modality data

Traditionally, specialized models and algorithms have been developed to process and analyze various modalities of data such as tables, texts, images, and time series. These methods can hardly perform integrative analysis across data modalities and generalize across different data platforms. Recently, LLM-based multi-modality foundation models (MMFMs) have shown strong promise in analyzing multi-modality data through the unified interface of languages [257–261]. Integrating multi-modal data into KGs creates a more comprehensive representation of entities and their relationships, enhancing performance in open-world applications like image classification and visual question answering [262]. Many studies focus on using MMFMs for specific tasks, such as entity extraction [263–265], relation extraction [266–268], or event extraction [269, 270]. However, these works often isolate extraction tasks without unifying entities and relations into a structured KG. A pioneering approach in this direction is

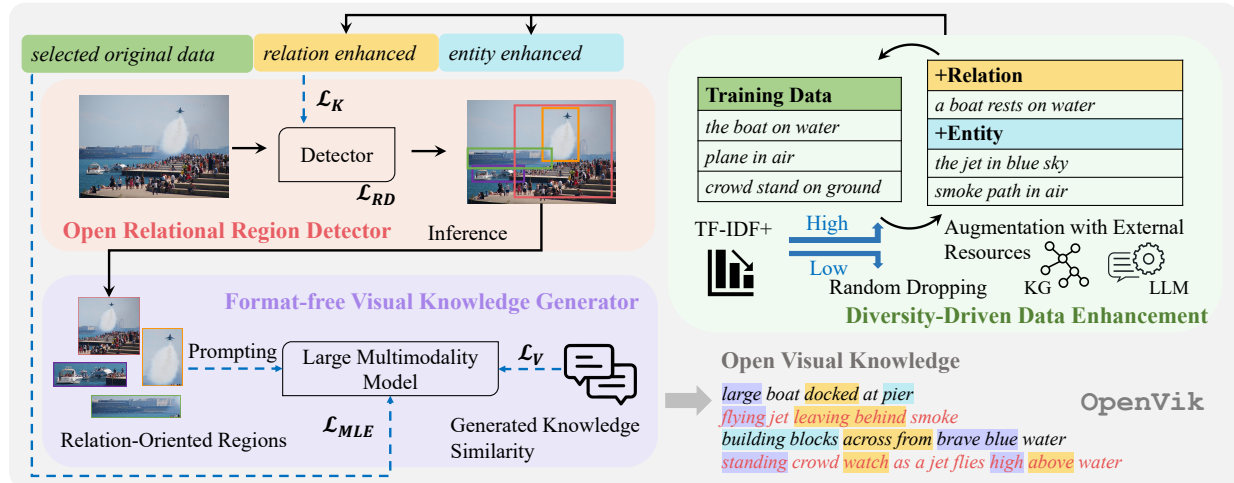


Figure 3: The overall framework of OpenVik consists of two main components: (1) an open relational region detector, highlighted in the orange and purple panels, which includes a region regression loss (\mathcal{L}_{RD}) and a regional description loss (\mathcal{L}_K), and (2) a format-free visual knowledge generator, incorporating knowledge generation loss (\mathcal{L}_{MLE}) and diversity regularization (\mathcal{L}_V). These modules work collaboratively to extract open visual knowledge, incorporating novel entities and diverse relations in a format-free manner.

OpenVik [271] (framework shown in Figure 3). It first trains an open relational region detector to locate image regions containing relational information. It then employs a visual knowledge generator to create format-free knowledge descriptions by prompting a large visual language model. Through the use of MMFM, OpenVik advances KG completion by integrating rich visual context, expanding knowledge coverage, and enhancing the accuracy of representation within the resulting KG. Application-wise, Yang et al. [272] proposed an automated approach to constructing product KGs from raw images in e-commerce. This method first employs vision-language models (VLMs) to extract detailed image information and then uses an LLM to reason and infer additional KG properties not visually present, hierarchically expanding, and linking nodes to develop comprehensive, scalable KGs without human input.

3 KG-guided LLM Enhancement

LLMs have shown impressive communication and question-answering capabilities, demonstrating strong promise in various applications [238, 261, 273–288]. However, to reliably model domain-specific data and generate factual and accurate answers, LLMs still face the challenges of lacking domain knowledge, fuzzy inferences, and hallucination [16–33]. Retrieval augmented generation (RAG) [289], which aims at retrieving query-relevant evidence and generating evidence-based answers, has strong promise in evidence-critical domains. However, effective and efficient RAG for complex queries is still challenging which requires LLMs to be able to (1) generate logical plans for retrieving multiple pieces of relevant evidence from complex data, (2) conduct valid reasoning and inference to compose the pieces of evidence towards generating coherent answers, and (3) reliably guarantee the detection and removal of errors. In the following, we discuss how these challenges can be addressed with well-designed planning, reasoning, and reflection frameworks with the help of KGs.

3.1 Planning with domain knowledge

While LLMs excel in many NLP tasks [8, 290], they still face challenges in acquiring domain knowledge. To address this issue, many attempts seek assistance from KGs, which are often constructed to represent knowledge in specific domains, such as medicine [50], law [291], and finance [292]. The integration of KGs and LLMs has shown promising results in various applications, such as question answering [181], recommendation [186], and dialogue systems [293]. Despite the success, there are still challenges in effectively obtaining useful information from KGs and incorporating them into LLMs.

Existing methods typically depend on a retriever to obtain relevant triples. For example, Baek et al. [294] proposed a direct retrieval method to retrieve relevant triples from KGs. However, the retriever may not always retrieve the most relevant triples, leading to suboptimal performance. Additionally, KGs contain a wealth of domain-specific knowledge, making it challenging for LLMs with limited domain expertise to comprehend and utilize this information. To further unleash LLMs’ capabilities of leveraging domain knowledge, the *plan-and-solve* paradigm [295] has been proposed, in which LLMs are prompted to first generate a plan. Based on the plan, LLMs can retrieve the relevant domain knowledge and conduct reasoning to generate answers [296]. However, existing methods are incapable of handling the complex structured knowledge in KGs to enable effective planning and reasoning. To address this issue, we propose a *planning-retrieval-reasoning* framework named **RoG** that enables LLMs to plan and reason on KGs [182]. The overall framework is illustrated in Figure 4.

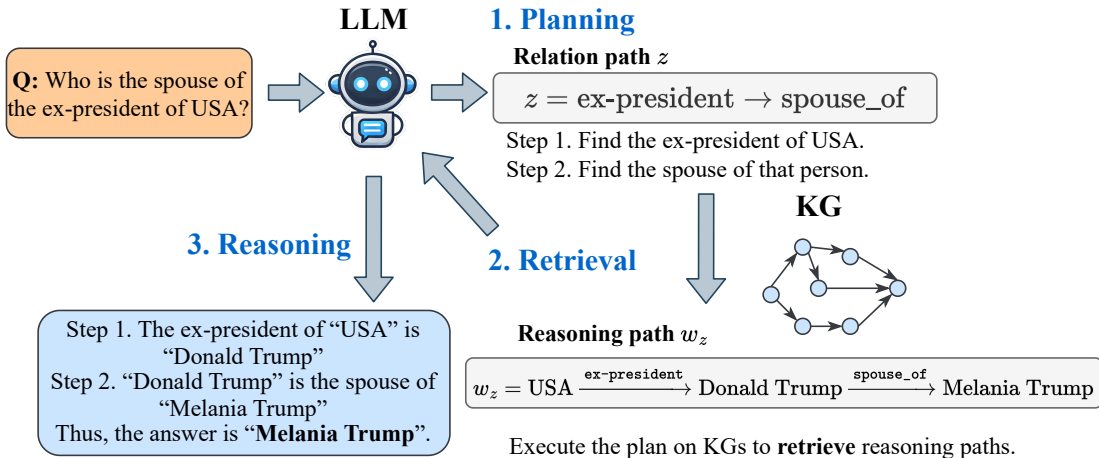


Figure 4: The overall framework of planning and reasoning on KGs (RoG).

RoG first generates several relation paths that are grounded by KGs as plans. Relation paths, which capture semantic relations between entities, have been utilized in many reasoning tasks on KGs [297, 298]. Based on relation paths, we can always retrieve the latest knowledge from KGs with a simple constrained breadth-first search. Therefore, relation paths can serve as faithful plans to guide the retrieval and reasoning on domain-specific KGs. Additionally, by treating relation paths as plans, we can make sure the plans are grounded by KGs, which enables LLMs to retrieve relevant knowledge and conduct faithful reasoning. To this end, we formulate our RoG as an optimization problem that aims to maximize the probability of reasoning the answer from a KG \mathcal{G} w.r.t the question q by generating relation paths z as the plan:

$$P_{\theta}(a|q, \mathcal{G}) = \sum_{z \in \mathcal{Z}} P_{\theta}(a|q, z, \mathcal{G}) P_{\theta}(z|q), \quad (1)$$

where θ denotes the parameters of LLMs and a denotes the final answer. To enable accurate planning

with domain knowledge, we design two instruction tuning tasks: 1) *planning optimization*, which distills the knowledge from KGs into LLMs to generate faithful relation paths as plans; 2) *retrieval-reasoning optimization*, which enables LLMs to reason based on the retrieved reasoning paths. The final objective function of **RoG** is the combination of the planning optimization and retrieval-reasoning optimization, which can be formulated as

$$\mathcal{L} = \underbrace{\log P_{\theta}(a|q, \mathcal{Z}_K^*, \mathcal{G})}_{\text{Retrieval-reasoning}} + \underbrace{\frac{1}{|\mathcal{Z}^*|} \sum_{z \in \mathcal{Z}^*} \log P_{\theta}(z|q)}_{\text{Planning}}, \quad (2)$$

where we use the shortest paths $\mathcal{Z}^* \subseteq \mathcal{Z}$ between q and a in KGs as supervision signals. We maximize the probability of LLMs generating faithful relation paths through distilling the knowledge from KGs. In this way, with the proposed **RoG**, LLMs can effectively retrieve domain knowledge from KGs with planning, which significantly enhances the reasoning capability of LLMs.

3.2 Reasoning with structured knowledge

KGs capture abundant factual knowledge in a structured format, which provides a faithful knowledge source for improving the reasoning abilities of LLMs [87]. Nevertheless, because of the unstructured nature of LLMs, directly applying them to reason on structured KGs is challenging. Early works focus on fine-tuning LLMs together with structured knowledge from KGs to enrich the knowledge of LLMs for better reasoning [152, 159]. For example, KEPLER [89] directly employs both KG embedding training objective and Masked token pre-training objective into a shared transformer-based encoder. Through fine-tuning, LLMs can better understand the structured knowledge in KGs for reasoning. However, the fine-tuning process is computationally expensive and incapable of efficiently adapting to the evolving real-world knowledge.

Recently, researchers have combined the strengths of retrieval-based methods with the prompting technique to enable LLMs to reason on KGs [181, 289]. CoK [25] and KD-CoT [299] retrieve facts from an external KG to guide the CoT performed by LLMs. To capture graph structure, GNN-RAG [148] adopts a lightweight graph neural network to effectively retrieve knowledge from KGs, which are formatted as a sentence path to elicit the reasoning process of LLMs. Mindmap [196] builds a prompt-based method that endows LLMs with the capability of comprehending KG and reasoning with it. Despite the success of these methods, they still face challenges in designing principled prompts to represent KGs and conduct reasoning. Moreover, LLMs still have limited capabilities in understanding the graph structure and reasoning with the text-based graph prompts [300].

Different from existing efforts that require a computationally expensive fine-tuning phase or design ad-hot prompts for LLMs, we recently introduced a KG-constrained reasoning (**GCR**) paradigm [301]. **GCR** connects unstructured reasoning in LLMs with structured knowledge in KGs, seeking to achieve efficient and effective reasoning on structured knowledge. The overall framework is illustrated in Figure 5.

Graph-constrained reasoning, inspired by the concept that LLMs reason through decoding [302], incorporates the KG structure into the LLM decoding process. This enables LLMs to directly reason on graphs by generating reliable reasoning paths grounded in KGs that lead to correct answers. Specifically, given a question, we first adopt a retrieval module to find a relevant KG that is helpful for reasoning. Then, we convert the KG into a structured index, KG-Trie, to facilitate efficient reasoning on KG using LLMs. Trie is also known as the prefix tree [303] that compresses a set of strings, which can be used to restrict LLM output tokens to those starting with valid prefixes. KG-Trie encodes the reasoning paths in KGs as formatted strings to constrain the decoding process of LLMs. Then, we propose graph-constrained decoding that employs a lightweight KG-specialized LLM to generate multiple KG-grounded reasoning paths and answers. With the constraints from KG-Trie, we ensure faithful reasoning while leveraging the strong reasoning capabilities of LLMs to efficiently explore paths on

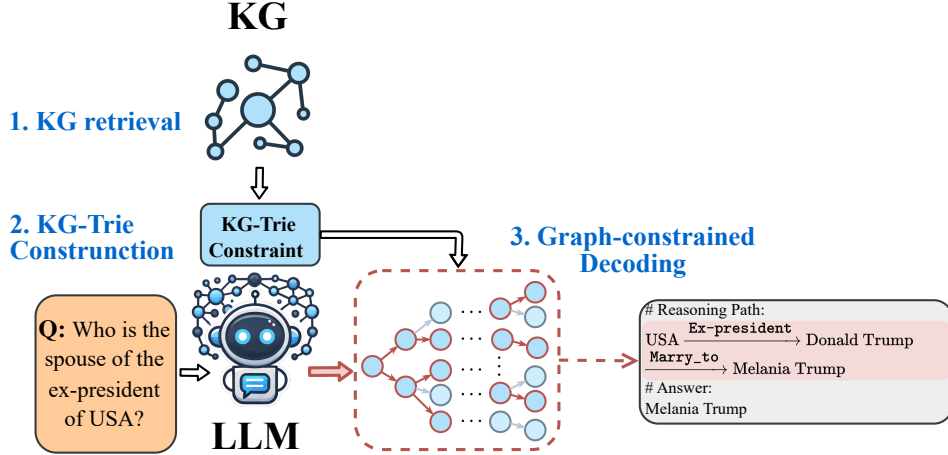


Figure 5: The overall framework of KG-constrained reasoning (GCR).

KGs in constant time. In this way, GCR bridges the gap between structured knowledge in KGs and unstructured reasoning in LLMs, allowing for efficient reasoning on KGs via LLM decoding.

3.3 Reflecting with atomic knowledge

LLMs have shown impressive capabilities in encapsulating massive knowledge and conducting reasoning. However, they still face challenges in generating factually correct and faithful responses, especially in the presence of hallucinations [304]. KGs store atomic knowledge in a structured format, which can be used to verify the correctness of generated responses and detect hallucinations [305]. To incorporate the factual knowledge from KGs into LLM hallucination detection, Guan et al. [200] proposed a retrieval-based method called KG-based retrofitting (KGR). KGR retrieves relevant facts from KGs during the LLM reasoning process, which are used to mitigate factual hallucination by retrofitting the initial responses. KGR enables an autonomous knowledge verifying and refining procedure with the factual knowledge retrieved from KGs, which significantly improves the reliability of LLMs.

The hallucination of LLMs is usually attributed to the lack of factual knowledge of LLMs. To systematically evaluate the factual knowledge inside LLMs, as shown in Figure 6a, we propose a novel framework to automatically assess the factual knowledge in LLMs by using KGs [214]. Unlike conventional methods that rely on human-annotated question-answering datasets, we systematically generate valid and diverse questions from KGs with different difficulties while also ensuring knowledge coverage. Specifically, we retire the atomic knowledge from KGs as sets of triples. Then, we utilize different question generation methods, e.g., template-based and LLM-based methods, to convert the triples into question-answer pairs. The generated pairs are used to evaluate the factual knowledge of LLMs by comparing the generated answers with the ground-truth answers. The evaluation results can be used to reflect the factual knowledge of LLMs. In this way, we can systematically evaluate the factual knowledge of LLMs and provide insights into the hallucination behavior of LLMs, which can be used to improve the reliability of LLMs in various applications.

Apart from the factual knowledge, the structure of KGs can be also utilized to justify the reasoning process of LLMs. Minh-Vuong et al [306] designed a framework that delves deeper into the CoT reasoning capabilities of LLMs in multi-hop question answering by utilizing KGs, as shown in Figure 6b. The framework contains two evaluation modules: discriminative evaluation and generative evaluation. The discriminative evaluation aims to analyze whether the LLMs possess enough knowledge to conduct

faithful reasoning. It feeds both valid and invalid reasoning paths retrieved from KGs into LLMs and asks them to predict the validity of these paths. The generative evaluation, on the other hand, aims to evaluate the faithfulness of the reasoning process of LLMs by grounding it on KGs. Given a reasoning process generated by LLMs, the generative evaluation module retrieves the facts from KGs, which are compared with the ground-truth reasoning paths. The evaluation results can be used to reflect the reasoning capabilities of LLMs and provide insights into the faithfulness of LLM reasoning. Based on the findings, although LLMs have shown impressive reasoning capabilities, they still face challenges in conducting faithful reasoning, especially in multi-hop question answering.

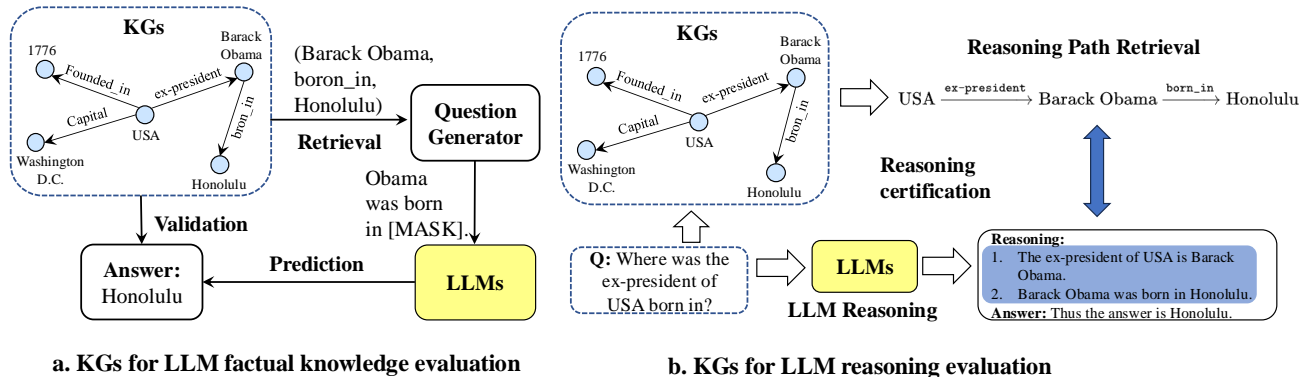


Figure 6: The illustration of LLM reflection with KGs. (a) The evaluation of the factual knowledge inside LLMs. (b) The evaluation of the reasoning process of LLMs with KGs.

4 Knowledge-aware Multi-agent Federation

While KGs and LLMs can mutually enhance each other, data are properties and many real-world datasets are privately collected and owned by different institutions, which cannot be simply put together to train more powerful models. Moreover, many real-world applications require domain-specific knowledge that may not have been captured by general-purpose KGs and LLMs yet, and such knowledge can also be private properties. Finally, while the development of KGs and LLMs is highly automated and data-driven, the values and needs of different human stakeholders may not have been properly reflected in the data and models. Federated learning (FL) provides a robust and principled framework for privacy-protected multi-site collaboration, but proper implementation of FL in the new era of generative AI remains unclear; the further incorporation of domain knowledge and human participation is also highly under-explored. In the following, we will envision an innovative Federated Multi-Agent System (FedMAS) for multi-site privacy-protected, knowledge-infused, and human-engaged KG-LLM co-learning scenarios.

Nowadays, while common practices in AI applications still largely resort to in-house development of models based on public and local data, the successes of generative AI, where complex models are trained with large-scale data, have demonstrated a strong need to collaboratively utilize local data towards obtaining powerful models that can generalize across institutions, finding and utilizing deep data patterns underlying common and rare use cases. Towards protecting local data privacy during collaborative model training, FL provides a promising solution [307, 307, 308, 308–310]. However, existing FL frameworks, by merely preventing the direct sharing of training data, are not effective in the scenarios of KG-LLM co-learning, because (1) as the construction of comprehensive KGs necessitates the

incorporation of knowledge discovered from local data, private information may get reversely inferred from the collaboratively constructed KG; (2) as powerful generative models like LLMs can easily memorize training data, collaboratively trained LLMs may expose private information facing deliberately composed jailbreaking prompts [311–313].

In our pioneering studies on FL for graphs [314–321], we developed several novel algorithms for different graph separation scenarios. In FedDEP [317], we developed a prototype-based embedding sharing algorithm with local graph differential privacy (DP) guarantees, and demonstrated its utility in FL for global graph embedding models across private local subgraphs; in FedR [319], we showed that sharing relation embeddings across local KGs can help FL for global KG embeddings with less privacy leakage. These studies have laid the foundations for our envisioned framework here, which will build on the private embedding sharing algorithms to construct *multi-view KGs* that can facilitate multi-site knowledge sharing with minimum risks of exposing local private data.

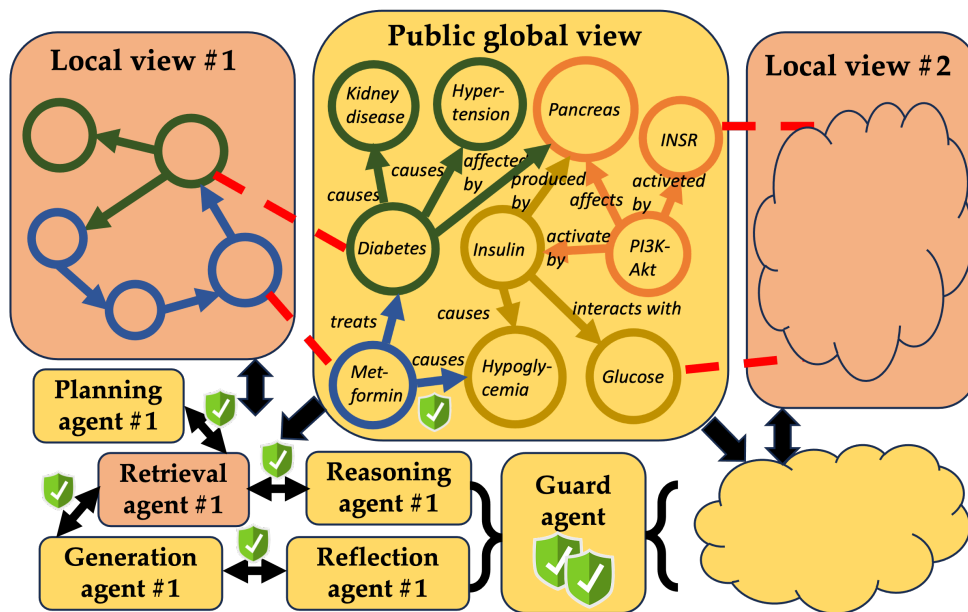


Figure 7: The envisioned framework of a federated multi-agent system (FedMAS).

As illustrated in Figure 7, our envisioned FedMAS will include a multi-view KG and various LLM agents. The federation on KG will be implemented by collectively constructing a multi-view KG by all participating sites, where knowledge from public resources is integrated into the global view, and knowledge from private resources is kept in each site’s local view only visible to itself. For each site, entities in its local view are linked with the global view, and only the embeddings related to these linked entities are shared with the server and other sites, via privacy-guaranteed embedding sharing algorithms such as those developed in FedDEP [317] and FedR [319]. In this way, each local site can compute embeddings on its local view and the public view as if they can see all other sites’ local views, allowing them to effectively adjust their local knowledge and further enhance their local LLMs, all without actually seeing the other sites’ local views (knowledge). The server will periodically adjust knowledge in the global view also based on the shared privacy-guaranteed embeddings, and apply additional privacy checks to make sure no sensitive knowledge gets propagated into the global view.

To rigorously protect the more sensitive patient data during the federation on LLM, it is possible to train multiple LLM agents in each site with different functions and let them collaborate through conversations instead of traditional model sharing, so the system can strictly control the level of private

data each agent can access and monitor or moderate their collaborations. Specifically, since the retrieval agents need to access all local patient data, one plan is to implement programmable guardrails [322] on its conversations with all other agents within the same site and forbid them from directly communicating with agents from other sites. Since all other agents can only access patient data indirectly through the retrieval agents, it is possible to adapt techniques such as our recently developed GuardAgent [323] based on the clear goals and typical outputs of each agent to monitor all cross-site conversations, detecting and removing any suspicious private information.

When applying FedMAS to specific application domains such as finance, law, education, and healthcare, it is promising to leverage the knowledge-based data sharing mechanism to incorporate existing domain knowledge towards further alleviating knowledge gaps and mitigating potential biases. Built on recent promising results from LLM-based data annotations as discussed in Section 2, FedMAS can utilize specialized LLM agents to perform comprehensive extraction of structured knowledge from existing guidelines and tutorials and automatically integrate them with existing general and domain-specific KGs. For example, in healthcare, one type of important domain-specific entity is social determinants of health (SDOH) [324–326]. The system can start with a set of known SDOH such as defined by WHO [327, 328], and further extend the set and discover their impacts and relationships with various risk factors by investigating relevant healthcare literature. The KGs enhanced through these steps are supposed to facilitate the alleviation of various health disparities when utilized by subsequent LLM agents in the FedMAS.

While FedMAS utilizes AI advances to automate multi-site data integration and modeling, comprehensive and trustworthy AI systems need to also incorporate the values and needs of various stakeholders, who can have different and even contradictory perspectives. LLMs, especially in our multi-agent conversational environment, provide unique convenience for effective and efficient human participation, where different stakeholders can verify, influence, and complement the decision processes and outputs of different LLM agents, all based on natural languages as the interface. Specifically, we envision a novel multi-stage intervention mechanism to efficiently enable the participation of different stakeholders in the LLM-based multi-agent conversational environment. The potential stages could include (1) LLM uncertainty quantification, where LLMs highlight their own uncertain outputs; (2) Rubrics-based rating, where humans create rubrics to automatically rate the LLM outputs; (3) Focused human interactions, where humans directly interact with LLMs, focusing on the problematic scenarios identified in the previous stages. The overall multi-stage mechanism is supposed to allow FedMAS to adapt to human values through iteratively integrating the language-based feedback via interactions with various stakeholders.

5 Conclusions and Future Directions

In this paper, we discuss the trending efforts of co-learning KGs and LLMs. Through the lens of SRAG, we showcase promising attempts to utilize LLMs to automate the construction, integration, and enrichment of KGs, and discuss how KGs can help with planning paths, guide reasoning with structure, and ground knowledge with reflection, enhancing the reliability of LLMs for downstream tasks. We also envision a novel system of multiple agents collaborating in a conversational federated learning environment based on the knowledge-infused, human-engaged LLMs. While the co-learning of KGs and LLMs holds great potential, we envision several promising directions especially from the SRAG perspective.

Effective evaluation of LLM-generated knowledge. To achieve effective knowledge enrichment for KGs with LLMs, it is critical to evaluate and guarantee the quality of added and/or modified knowledge. However, new knowledge is hard to evaluate in nature due to the lack of ground truth. Exhaustive

human evaluation is costly, but LLMs can be utilized to lubricate the collaboration between humans and machines toward efficient new knowledge evaluation. For example, humans can create guidelines and rubrics for LLMs to screen and rate the new knowledge from different perspectives. LLMs can also evaluate the quality of knowledge with confidence or uncertainty quantifications. Humans can then focus on the LLM-flagged suspicious or uncertain new knowledge to conduct close manual evaluation.

Unified versus specialized KGs. Due to the diversity and breadth of knowledge, it might be difficult to integrate all knowledge into a single unified KG, which might potentially harm the knowledge integrity. As a potential alternative, it may become practical to construct specialized KGs depending on the knowledge needs of different applications. It then remains an open problem regarding how to measure the relevance of knowledge with respect to specific applications and decide what to include/exclude from the specialized KGs.

More powerful KGs. Current KGs mostly include general, binary, and pair-wise relations. However, when KGs are used in certain applications, the knowledge may not equally hold for every context. For example, one drug may treat a disease for only certain groups of patients. In such scenarios, specific mechanisms are needed to model the various contexts for knowledge. Moreover, relations are not always binary and pair-wise (between pairs of entities). They can be true with a probability and involve more than two entities. Such scenarios are ubiquitous in reality, so probabilistic KGs and n-ary KGs should receive wider adoption and study.

Trade-off between effectiveness and efficiency of retrieval. Most existing retrieval-based methods focus on developing an effective retrieval mechanism to accurately retrieve relevant knowledge from KGs [180, 329]. However, they often overlook the efficiency of the retrieval process. In practice, the retrieval process can be computationally expensive, especially when the KG is large. Meanwhile, real-world application often requires prompt responses, which further exacerbates the efficiency issue. Therefore, it is essential to strike a balance between the effectiveness and efficiency of the retrieval process [330].

Resolving knowledge conflicts (internal LLM knowledge versus external knowledge). LLMs contain a vast amount of knowledge obtained via pre-training. However, the knowledge might be inaccurate or outdated, which could conflict with the knowledge retrieved from KGs [331]. To resolve the conflict, SPARE [332] utilizes the internal activations of LLMs to identify the conflict. AstuteRAG [333] uses a novel RAG approach to adaptively elicit LLM internal knowledge and iteratively consolidate internal and external knowledge. Despite the attempts, how to effectively identify and resolve the conflict between the internal knowledge of LLMs and the external knowledge retrieved from KGs remains an open problem.

Retrieval from multi-modal data. KGs store knowledge in diverse modalities such as text, image, and video [334]. Existing KG retrieval methods mainly focus on retrieving textual knowledge. However, the retrieval from multi-modal data is still under-explored. Knowledge from different modalities can complement each other, which could potentially enhance the retrieval performance. Therefore, it is essential to develop retrieval methods that can effectively retrieve knowledge from multi-modal data [335].

Robustness/safety of SRAG for LLMs. The safety and robustness of LLMs are receiving increasing attention due to their critical role in developing trustworthy AI systems. Previous research has primarily focused on attacking the LLMs themselves [336]. However, integrating LLMs with KG retrieval systems

expands the attack surface. Attackers could manipulate KGs and the retrieval systems to mislead LLMs, potentially leading to severe consequences [337]. Therefore, enhancing the robustness and safety of the combined KG and LLM systems is an important research direction.

Acknowledgements

This research was partially supported by the National Science Foundation under Award Number 2319449 and Award Number 2312502, as well as the National Institute Of Diabetes And Digestive And Kidney Diseases of the National Institutes of Health under Award Number K25DK135913. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views, either expressed or implied, of the National Science Foundation, National Institutes of Health, or the U.S. government. The authors wish to thank the editors and reviewers for their valuable efforts and suggestions.

References

- [1] OpenAI, “Gpt-4 technical report,” [arXiv preprint arXiv:2303.08774](#), 2023.
- [2] C. Leiter, R. Zhang, Y. Chen, J. Belouadi, D. Larionov, V. Fresen, and S. Eger, “Chatgpt: A meta-analysis after 2.5 months,” [Machine Learning with Applications](#), vol. 16, p. 100541, 2024.
- [3] G. T. Google, “Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context,” [arXiv preprint arXiv:2403.05530](#), 2024.
- [4] NVIDIA, “Nemotron-4 340b technical report,” [arXiv preprint arXiv:2406.11704](#), 2024.
- [5] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. Le Scao, T. Lavril, T. Wang, T. Lacroix, and W. El Sayed, “Mistral 7b,” [arXiv preprint arXiv:2310.06825](#), 2023.
- [6] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, “Llama: Open and efficient foundation language models,” [arXiv preprint arXiv:2302.13971](#), 2023.
- [7] G. T. Google, “Gemma: Open models based on gemini research and technology,” [arXiv preprint arXiv:2403.08295](#), 2024.
- [8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” in [Proceedings of the Conference on Neural Information Processing Systems \(NeurIPS\)](#), 2020.
- [9] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, “Sparks of artificial general intelligence: Early experiments with gpt-4,” [arXiv preprint arXiv:2303.12712](#), 2023.
- [10] Y. Ge, W. Hua, K. Mei, J. Tan, S. Xu, Z. Li, and Y. Zhang, “Openagi: When llm meets domain experts,” in [Proceedings of the Conference on Neural Information Processing Systems \(NeurIPS\)](#), 2024.

- [11] S. C. Bankes, “Agent-based modeling: A revolution?” Proceedings of the National Academy of Sciences, vol. 99, no. suppl_3, pp. 7199–7200, 2002.
- [12] Y. Bai, S. Kadavath, S. Kundu, A. Askill, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, C. Chen, C. Olsson, C. Olah, D. Hernandez, D. Drain, D. Ganguli, D. Li, E. Tran-Johnson, E. Perez, J. Kerr, J. Mueller, J. Ladish, J. Landau, K. Ndousse, K. Lukosuite, L. Lovitt, M. Sellitto, N. Elhage, N. Schiefer, N. Mercado, N. DasSarma, R. Lasenby, R. Larson, S. Ringer, S. Johnston, S. Kravec, S. E. Showk, S. Fort, T. Lanham, T. Telleen-Lawton, T. Conerly, T. Henighan, T. Hume, S. R. Bowman, Z. Hatfield-Dodds, B. Mann, D. Amodei, N. Joseph, S. McCandlish, T. Brown, and J. Kaplan, “Constitutional ai: Harmlessness from ai feedback,” arXiv preprint arXiv:2212.08073, 2022.
- [13] E. Bonabeau, “Agent-based modeling: Methods and techniques for simulating human systems,” Proceedings of the National Academy of Sciences, vol. 99, no. suppl_3, pp. 7280–7287, 2002.
- [14] G. Li, H. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem, “Camel: Communicative agents for “mind” exploration of large language model society,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2023.
- [15] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, A. H. Awadallah, R. W. White, D. Burger, and C. Wang, “Autogen: Enabling next-gen llm applications via multi-agent conversation,” in First Conference on Language Modeling, 2024.
- [16] X. Hu, J. Chen, X. Li, Y. Guo, L. Wen, P. S. Yu, and Z. Guo, “Do large language models know about facts?” arXiv preprint arXiv:2310.05177, 2023.
- [17] S. M. Mousavi, S. Alghisi, and G. Riccardi, “Is your llm outdated? benchmarking llms & alignment algorithms for time-sensitive knowledge,” arXiv preprint arXiv:2404.08700, 2024.
- [18] Y. A. Yadkori, I. Kuzborskij, A. György, and C. Szepesvári, “To believe or not to believe your llm,” arXiv preprint arXiv:2406.02543, 2024.
- [19] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, “Self-RAG: Learning to retrieve, generate, and critique through self-reflection,” in The Twelfth International Conference on Learning Representations (ICLR), 2024.
- [20] Y. Yu, W. Ping, Z. Liu, B. Wang, J. You, C. Zhang, M. Shoenybi, and B. Catanzaro, “RankRAG: Unifying context ranking with retrieval-augmented generation in LLMs,” in Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS), 2024.
- [21] H. Liu, R. Ning, Z. Teng, J. Liu, Q. Zhou, and Y. Zhang, “Evaluating the logical reasoning ability of chatgpt and gpt-4,” arXiv preprint arXiv:2304.03439, 2023.
- [22] K. Zhu, J. Chen, J. Wang, N. Z. Gong, D. Yang, and X. Xie, “Dyval: Dynamic evaluation of large language models for reasoning tasks,” in Proceedings of the International Conference on Learning Representations (ICLR), 2023.
- [23] H. H. Zhuo, X. Chen, and R. Pan, “On the roles of llms in planning: Embedding llms into planning graphs,” arXiv preprint arXiv:2403.00783, 2024.
- [24] C. Yuan, Q. Xie, J. Huang, and S. Ananiadou, “Back to the future: Towards explainable temporal reasoning with large language models,” in Proceedings of the International World Wide Web Conference (WWW), 2024.

- [25] J. Wang, Q. Sun, X. Li, and M. Gao, “Boosting language models reasoning with chain-of-knowledge prompting,” arXiv preprint arXiv:2306.06427, 2023.
- [26] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, “Survey of hallucination in natural language generation,” ACM Computing Surveys, vol. 55, no. 12, pp. 1–38, 2023.
- [27] Z. Bai, P. Wang, T. Xiao, T. He, Z. Han, Z. Zhang, and M. Z. Shou, “Hallucination of multimodal large language models: A survey,” arXiv preprint arXiv:2404.18930, 2024.
- [28] S. Tonmoy, S. Zaman, V. Jain, A. Rani, V. Rawte, A. Chadha, and A. Das, “A comprehensive survey of hallucination mitigation techniques in large language models,” arXiv preprint arXiv:2401.01313, 2024.
- [29] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, “On faithfulness and factuality in abstractive summarization,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2020.
- [30] Y. Xiao and W. Y. Wang, “On hallucination and predictive uncertainty in conditional language generation,” in Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL), 2021.
- [31] S. Farquhar, J. Kossen, L. Kuhn, and Y. Gal, “Detecting hallucinations in large language models using semantic entropy,” Nature, vol. 630, no. 8017, pp. 625–630, 2024.
- [32] Z. Ji, T. Yu, Y. Xu, N. Lee, E. Ishii, and P. Fung, “Towards mitigating llm hallucination via self reflection,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2023.
- [33] C. Chen, K. Liu, Z. Chen, Y. Gu, Y. Wu, M. Tao, Z. Fu, and J. Ye, “Inside: Llms’ internal states retain the power of hallucination detection,” in Proceedings of the International Conference on Learning Representations (ICLR), 2024.
- [34] M. Wornow, Y. Xu, R. Thapa, B. Patel, E. Steinberg, S. Fleming, M. A. Pfeffer, J. Fries, and N. H. Shah, “The shaky foundations of large language models and foundation models for electronic health records,” NPJ Digital Medicine, vol. 6, no. 1, p. 135, 2023.
- [35] Y. Shen, L. Heacock, J. Elias, K. D. Hentel, B. Reig, G. Shih, and L. Moy, “Chatgpt and other large language models are double-edged swords,” Radiology, vol. 307, no. 2, p. e230163, 2023.
- [36] A. Pal, L. K. Umapathi, and M. Sankarasubbu, “Med-halt: Medical domain hallucination test for large language models,” in Proceedings of the Conference on Computational Natural Language Learning (CoNLL), 2023.
- [37] R. Xu, H. Liu, S. Nag, Z. Dai, Y. Xie, X. Tang, C. Luo, Y. Li, J. C. Ho, C. Yang, and Q. He, “Simrag: Self-improving retrieval-augmented generation for adapting large language models to specialized domains,” arXiv preprint arXiv:2410.17952, 2024.
- [38] D. P. Panagoulas, M. Virvou, and G. A. Tsihrintzis, “Evaluating llm-generated multimodal diagnosis from medical images and symptom analysis,” arXiv preprint arXiv:2402.01730, 2024.
- [39] Z. Gu, C. Yin, F. Liu, and P. Zhang, “Medvh: Towards systematic evaluation of hallucination for large vision language models in the medical context,” arXiv preprint arXiv:2407.02730, 2024.

- [40] A. Hogan, E. Blomqvist, M. Cochez, C. D’amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, and S. Neumaier, “Knowledge graphs,” ACM Computing Surveys, vol. 54, no. 4, pp. 1–37, 2021.
- [41] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, “A survey on knowledge graphs: Representation, acquisition, and applications,” IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 2, pp. 494–514, 2021.
- [42] Q. Wang, Z. Mao, B. Wang, and L. Guo, “Knowledge graph embedding: A survey of approaches and applications,” IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 29, no. 12, pp. 2724–2743, 2017.
- [43] X. Zou, “A survey on application of knowledge graph,” in Journal of Physics: Conference Series, vol. 1487, no. 1. IOP Publishing, 2020, p. 012016.
- [44] H. Cui, J. Lu, S. Wang, R. Xu, W. Ma, S. Yu, Y. Yu, X. Kan, T. Fu, C. Ling, J. Ho, F. Wang, and C. Yang, “A survey on knowledge graphs for healthcare: Resources, application progress, and promise,” in ICML 3rd Workshop on Interpretable Machine Learning in Healthcare (IMLH), 2023.
- [45] S. Chang, Y. Hou, S. Rajendran, J. R. M. A. Maasch, Z. Abedi, H. Zhang, Z. Bai, A. Cuturrufo, W. Guo, F. F. Chaudhry, G. Ghahramani, J. Tang, F. Cheng, Y. Li, R. Zhang, J. Bian, and F. Wang, “Biomedical discovery through the integrative biomedical knowledge hub (ibkh),” iScience, vol. 26, no. 4, p. 106460, 2023.
- [46] R. Cornet and N. de Keizer, “Forty years of snomed: a literature review,” BMC Medical Informatics and Decision Making, vol. 8, pp. 1–6, 2008.
- [47] A. Santos, A. R. Colaço, A. B. Nielsen, L. Niu, P. E. Geyer, F. Coscia, N. J. W. Albrechtsen, F. Mundt, L. J. Jensen, and M. Mann, “Clinical knowledge graph integrates proteomics data into clinical decision-making,” bioRxiv, pp. 2020–05, 2020.
- [48] J. E. Harrison, S. Weber, R. Jakob, and C. G. Chute, “Icd-11: an international classification of diseases for the twenty-first century,” BMC Medical Informatics and Decision Making, vol. 21, pp. 1–10, 2021.
- [49] C. E. Lipscomb, “Medical subject headings (mesh),” Bulletin of the Medical Library Association, vol. 88, no. 3, p. 265, 2000.
- [50] O. Bodenreider, “The unified medical language system (umls): integrating biomedical terminology,” Nucleic Acids Research, vol. 32, no. suppl_1, pp. D267–D270, 2004.
- [51] J. Xu, S. Kim, M. Song, M. Jeong, D. Kim, J. Kang, J. F. Rousseau, X. Li, W. Xu, V. I. Torvik, Y. Bu, C. Chen, I. A. Ebeid, D. Li, and Y. Ding, “Building a pubmed knowledge graph,” Scientific Data, vol. 7, no. 1, p. 205, 2020.
- [52] L. Li, P. Wang, J. Yan, Y. Wang, S. Li, J. Jiang, Z. Sun, B. Tang, T.-H. Chang, S. Wang, and Y. Liu, “Real-world data medical knowledge graph: construction and applications,” Artificial Intelligence in Medicine, vol. 103, p. 101817, 2020.
- [53] Q. Lv, G. Chen, H. He, Z. Yang, L. Zhao, K. Zhang, and C. Y.-C. Chen, “Tcm bank-the largest tcm database provides deep learning-based chinese-western medicine exclusion prediction,” Signal Transduction and Targeted Therapy, vol. 8, no. 1, p. 127, 2023.

- [54] C. Xiong, R. Power, and J. Callan, “Explicit semantic ranking for academic search via knowledge graph embedding,” in Proceedings of the International World Wide Web Conference (WWW), 2017.
- [55] Z. Liu, C. Xiong, M. Sun, and Z. Liu, “Entity-duet neural ranking: Understanding the role of knowledge graph semantics in neural information retrieval,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2018.
- [56] T. Vu, T. D. Nguyen, D. Q. Nguyen, D. Phung et al., “A capsule network-based embedding model for knowledge graph completion and search personalization,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2019.
- [57] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, “Kgat: Knowledge graph attention network for recommendation,” in Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2019.
- [58] X. Wang, T. Huang, D. Wang, Y. Yuan, Z. Liu, X. He, and T.-S. Chua, “Learning intents behind interactions with knowledge graph for recommendation,” in Proceedings of the International World Wide Web Conference (WWW), 2021.
- [59] Y. Yang, C. Huang, L. Xia, and C. Li, “Knowledge graph contrastive learning for recommendation,” in Proceedings of the ACM International Conference on Research and Development in Information Retrieval (SIGIR), 2022.
- [60] F. Feng, F. Tang, Y. Gao, D. Zhu, T. Li, S. Yang, Y. Yao, Y. Huang, and J. Liu, “Genomickb: a knowledge graph for the human genome,” Nucleic Acids Research, vol. 51, no. D1, pp. D950–D956, 2023.
- [61] X. Shao, C. Li, H. Yang, X. Lu, J. Liao, J. Qian, K. Wang, J. Cheng, P. Yang, H. Chen, X. Xu, and X. Fan, “Knowledge-graph-based cell-cell communication inference for spatially resolved transcriptomic data with spatalk,” Nature Communications, vol. 13, no. 1, p. 4429, 2022.
- [62] X. Quan, W. Cai, C. Xi, C. Wang, and L. Yan, “Aimedgraph: a comprehensive multi-relational knowledge graph for precision medicine,” Database, vol. 2023, p. baad006, 2023.
- [63] S. Zhang, X. Lin, and X. Zhang, “Discovering dti and ddi by knowledge graph with mhrw and improved neural network,” in Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2021.
- [64] B. Li and H. Chen, “Prediction of compound synthesis accessibility based on reaction knowledge graph,” Molecules, vol. 27, no. 3, p. 1039, 2022.
- [65] J. Jeong, N. Lee, Y. Shin, and D. Shin, “Intelligent generation of optimal synthetic pathways based on knowledge graph inference and retrosynthetic predictions using reaction big data,” Journal of the Taiwan Institute of Chemical Engineers, vol. 130, p. 103982, 2022.
- [66] Y. Xu, X. Chu, K. Yang, Z. Wang, P. Zou, H. Ding, J. Zhao, Y. Wang, and B. Xie, “Seqcare: Sequential training with external medical knowledge graph for diagnosis prediction in healthcare data,” in Proceedings of the ACM Web Conference 2023, 2023, pp. 2819–2830.

- [67] P. Jiang, C. Xiao, A. R. Cross, and J. Sun, “Graphcare: Enhancing healthcare predictions with personalized knowledge graphs,” in The Twelfth International Conference on Learning Representations, 2024.
- [68] R. Xu, W. Shi, Y. Yu, Y. Zhuang, B. Jin, M. D. Wang, J. C. Ho, and C. Yang, “Ram-ehr: Retrieval augmentation meets clinical predictions on electronic health records,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2024.
- [69] P. Chen, Y. Lu, V. W. Zheng, X. Chen, and B. Yang, “Knowedu: A system to construct knowledge graph for education,” IEEE Access, vol. 6, pp. 31 553–31 563, 2018.
- [70] M. Rizun, “Knowledge graph application in education: a literature review,” Acta Universitatis Lodzianis. Folia Oeconomica, vol. 3, no. 342, pp. 7–19, 2019.
- [71] H. Tang, “Intelligent processing and classification of multisource health big data from the perspective of physical and medical integration,” Scientific Programming, vol. 2022, no. 1, p. 5799354, 2022.
- [72] L. Yuan, Y. Wang, P. M. Thompson, V. A. Narayan, and J. Ye, “Multi-source feature learning for joint analysis of incomplete multiple heterogeneous neuroimaging data,” NeuroImage, vol. 61, no. 3, pp. 622–632, 2012.
- [73] J. Zhang, “Multi-source remote sensing data fusion: status and trends,” International Journal of Image and Data Fusion, vol. 1, no. 1, pp. 5–24, 2010.
- [74] Y. Huang, C. Du, Z. Xue, X. Chen, H. Zhao, and L. Huang, “What makes multi-modal learning better than single (provably),” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2021.
- [75] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, “Multimodal machine learning: A survey and taxonomy,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 2, pp. 423–443, 2018.
- [76] J. N. Acosta, G. J. Falcone, P. Rajpurkar, and E. J. Topol, “Multimodal biomedical ai,” Nature Medicine, vol. 28, no. 9, pp. 1773–1784, 2022.
- [77] Q. Cai, H. Wang, Z. Li, and X. Liu, “A survey on multimodal data-driven smart healthcare systems: approaches and applications,” IEEE Access, vol. 7, pp. 133 583–133 599, 2019.
- [78] C. Zhang, X. Chu, L. Ma, Y. Zhu, Y. Wang, J. Wang, and J. Zhao, “M3care: Learning with missing modalities in multimodal healthcare data,” in Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2022.
- [79] L. R. Soenksen, Y. Ma, C. Zeng, L. Boussieux, K. Villalobos Carballo, L. Na, H. M. Wiberg, M. L. Li, I. Fuentes, and D. Bertsimas, “Integrated multimodal artificial intelligence framework for healthcare applications,” NPJ Digital Medicine, vol. 5, no. 1, p. 149, 2022.
- [80] T. Shaik, X. Tao, L. Li, H. Xie, and J. D. Velásquez, “A survey of multimodal information fusion for smart healthcare: Mapping the journey from data to wisdom,” Information Fusion, p. 102040, 2023.
- [81] F. Kronen, U. Marikkar, G. Parsons, A. Szmul, and A. Mahdi, “Review of multimodal machine learning approaches in healthcare,” arXiv preprint arXiv:2402.02460, 2024.

- [82] F. Cremonesi, V. Planat, V. Kalokyri, H. Kondylakis, T. Sanavia, V. M. M. Resinas, B. Singh, and S. Uribe, “The need for multimodal health data modeling: A practical approach for a federated-learning healthcare platform,” Journal of Biomedical Informatics, vol. 141, p. 104338, 2023.
- [83] D. Iakovidis and C. Smailis, “A semantic model for multimodal data mining in healthcare information systems,” in Quality of Life through Quality of Information. IOS Press, 2012, pp. 574–578.
- [84] A. Kline, H. Wang, Y. Li, S. Dennis, M. Hutch, Z. Xu, F. Wang, F. Cheng, and Y. Luo, “Multimodal machine learning in precision health: A scoping review,” NPJ Digital Medicine, vol. 5, no. 1, p. 171, 2022.
- [85] S. R. Stahlschmidt, B. Ulfenborg, and J. Synnergren, “Multimodal deep learning for biomedical data fusion: a review,” Briefings in Bioinformatics, vol. 23, no. 2, p. bbab569, 2022.
- [86] J. Pan, S. Razniewski, J.-C. Kalo, S. Singhanian, J. Chen, S. Dietze, H. Jabeen, J. Omeliyanenko, W. Zhang, M. Lissandrini, R. Biswas, G. de Melo, A. Bonifati, E. Vakaj, M. Dragoni, and D. Graux, “Large language models and knowledge graphs: Opportunities and challenges,” Transactions on Graph Data and Knowledge, 2023.
- [87] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, “Unifying large language models and knowledge graphs: A roadmap,” IEEE Transactions on Knowledge and Data Engineering, vol. 1, no. 1, pp. 2:1–2:38, 2024.
- [88] S. Pan, Y. Zheng, and Y. Liu, “Integrating graphs with large language models: Methods and prospects,” IEEE Intelligent Systems, vol. 39, no. 1, pp. 64–68, 2024.
- [89] X. Wang, T. Gao, Z. Zhu, Z. Zhang, Z. Liu, J. Li, and J. Tang, “Kepler: A unified model for knowledge embedding and pre-trained language representation,” Transactions of the Association for Computational Linguistics, vol. 9, pp. 176–194, 2021.
- [90] D. Yu, C. Zhu, Y. Yang, and M. Zeng, “Jakot: Joint pre-training of knowledge graph and language understanding,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2022.
- [91] M. Yasunaga, A. Bosselut, H. Ren, X. Zhang, C. D. Manning, P. S. Liang, and J. Leskovec, “Deep bidirectional language-knowledge graph pretraining,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2022.
- [92] Z. Jiang, L. Zhong, M. Sun, J. Xu, R. Sun, H. Cai, S. Luo, and Z. Zhang, “Efficient knowledge infusion via kg-llm alignment,” arXiv preprint arXiv:2406.03746, 2024.
- [93] X. Wang, Q. He, J. Liang, and Y. Xiao, “Language models as knowledge embeddings,” in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2022.
- [94] Y. Yu, C. Xiong, S. Sun, C. Zhang, and A. Overwijk, “Coco-dr: Combating distribution shifts in zero-shot dense retrieval with contrastive and distributionally robust learning,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022.
- [95] X. Xie, Z. Li, X. Wang, Z. Xi, and N. Zhang, “Lambdakg: A library for pre-trained language model-based knowledge graph embeddings,” in Proceedings of the International Joint Conference on Natural Language Processing, 2023.

- [96] N. Choudhary and C. K. Reddy, “Complex logical reasoning over knowledge graphs using large language models,” arXiv preprint arXiv:2305.01157, 2023.
- [97] Z. Zhang, X. Liu, Y. Zhang, Q. Su, X. Sun, and B. He, “Pretrain-kge: learning knowledge representation from pretrained language models,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2020.
- [98] P. Ke, H. Ji, Y. Ran, X. Cui, L. Wang, L. Song, X. Zhu, and M. Huang, “Jointgt: Graph-text joint representation learning for text generation from knowledge graphs,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2021.
- [99] L. Yao, C. Mao, and Y. Luo, “Kg-bert: Bert for knowledge graph completion,” arXiv preprint arXiv:1909.03193, 2019.
- [100] B. Kim, T. Hong, Y. Ko, and J. Seo, “Multi-task learning for knowledge graph completion with pre-trained language models,” in Proceedings of the International Conference on Computational Linguistics (COLING), 2020.
- [101] X. Lv, Y. Lin, Y. Cao, L. Hou, J. Li, Z. Liu, P. Li, and J. Zhou, “Do pre-trained models benefit knowledge graph completion? a reliable evaluation and a reasonable approach,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2022.
- [102] J. Shen, C. Wang, L. Gong, and D. Song, “Joint language semantic and structure embedding for knowledge graph completion,” in Proceedings of the International Conference on Computational Linguistics (COLING), 2022.
- [103] B. Choi and Y. Ko, “Knowledge graph extension with a pre-trained language model via unified learning method,” Knowledge-Based Systems, vol. 262, p. 110245, 2023.
- [104] B. Wang, T. Shen, G. Long, T. Zhou, Y. Wang, and Y. Chang, “Structure-augmented text representation learning for efficient knowledge graph completion,” in Proceedings of the International World Wide Web Conference (WWW), 2021.
- [105] L. Wang, W. Zhao, Z. Wei, and J. Liu, “Simkge: Simple contrastive knowledge graph completion with pre-trained language models,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2022.
- [106] D. Li, B. Zhu, S. Yang, K. Xu, M. Yi, Y. He, and H. Wang, “Multi-task pre-training language model for semantic network completion,” ACM Transactions on Asian and Low-Resource Language Information Processing, vol. 22, no. 11, pp. 1–20, 2023.
- [107] A. Saxena, A. Kochsiek, and R. Gemulla, “Sequence-to-sequence knowledge graph completion and question answering,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2022.
- [108] C. Chen, Y. Wang, B. Li, and K.-Y. Lam, “Knowledge is flat: A seq2seq generative framework for various knowledge graph completion,” in Proceedings of the International Conference on Computational Linguistics (COLING), 2022.
- [109] C. Chen, Y. Wang, A. Sun, B. Li, and K.-Y. Lam, “Dipping plms sauce: Bridging structure and text for effective knowledge graph completion via conditional soft prompting,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2023.

- [110] J. Lovelace and C. Rosé, “A framework for adapting pre-trained language models to knowledge graph completion,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022.
- [111] X. Xie, N. Zhang, Z. Li, S. Deng, H. Chen, F. Xiong, M. Chen, and H. Chen, “From discrimination to generation: Knowledge graph completion with generative transformer,” in Companion Proceedings of the International World Wide Web Conference (WWW), 2022.
- [112] Y. Zhu, X. Wang, J. Chen, S. Qiao, Y. Ou, Y. Yao, S. Deng, H. Chen, and N. Zhang, “Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities,” arXiv preprint arXiv:2305.13168, 2023.
- [113] H. Ye, N. Zhang, H. Chen, and H. Chen, “Generative knowledge graph construction: A review,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022, pp. 1–17.
- [114] C. Qin, A. Zhang, Z. Zhang, J. Chen, M. Yasunaga, and D. Yang, “Is chatgpt a general-purpose natural language processing task solver?” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2023.
- [115] V. K. Kommineni, B. König-Ries, and S. Samuel, “From human experts to machines: An llm supported approach to ontology and knowledge graph construction,” arXiv preprint arXiv:2403.08345, 2024.
- [116] B. Zhang and H. Soh, “Extract, define, canonicalize: An LLM-based framework for knowledge graph construction,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2024, pp. 9820–9836.
- [117] J. Vizcarra, S. Haruta, and M. Kurokawa, “Representing the interaction between users and products via llm-assisted knowledge graph construction,” in Proceedings of the IEEE International Conference on Semantic Computing (ICSC), 2024.
- [118] S. Yu, T. Huang, M. Liu, and Z. Wang, “Bear: Revolutionizing service domain knowledge graph construction with llm,” in Proceedings of the International Conference on Service-Oriented Computing, 2023.
- [119] Y. Hu, F. Zou, J. Han, X. Sun, and Y. Wang, “Llm-tikg: Threat intelligence knowledge graph construction utilizing large language model,” Computers & Security, p. 103999, 2024.
- [120] L.-P. Meyer, C. Stadler, J. Frey, N. Radtke, K. Junghanns, R. Meissner, G. Dziwis, K. Bulert, and M. Martin, “Llm-assisted knowledge graph engineering: Experiments with chatgpt,” in Proceedings of the Working Conference on Artificial Intelligence Development for a Resilient and Sustainable Tomorrow, 2023.
- [121] M. Hofer, J. Frey, and E. Rahm, “Towards self-configuring knowledge graph construction pipelines using llms—a case study with rml,” in ESWC Workshop on Knowledge Graph Construction, 2024.
- [122] R. Yang, B. Yang, A. Feng, S. Ouyang, M. Blum, T. She, Y. Jiang, F. Lecue, J. Lu, and I. Li, “Graphusion: A rag framework for knowledge graph construction with a global perspective,” arXiv preprint arXiv:2410.17600, 2024.

- [123] I. Melnyk, P. Dognin, and P. Das, “Grapher: Multi-stage knowledge graph construction using pretrained language models,” in NeurIPS Workshop on Deep Generative Models and Downstream Applications, 2021.
- [124] A. Kumar, A. Pandey, R. Gadia, and M. Mishra, “Building knowledge graph using pre-trained language model for learning entity-aware relationships,” in Proceedings of the IEEE International Conference on Computing, Power and Communication Technologies, 2020.
- [125] X. Chen, N. Zhang, X. Xie, S. Deng, Y. Yao, C. Tan, F. Huang, L. Si, and H. Chen, “Know-prompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction,” in Proceedings of the International World Wide Web Conference (WWW), 2022.
- [126] Z. Wan, F. Cheng, Z. Mao, Q. Liu, H. Song, J. Li, and S. Kurohashi, “Gpt-re: In-context learning for relation extraction using large language models,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2023.
- [127] Y. Wang, H. Xin, and L. Chen, “Kglink: A column type annotation method that combines knowledge graph and pre-trained language model,” arXiv preprint arXiv:2406.00318, 2024.
- [128] H. Li, G. Appleby, and A. Suh, “A preliminary roadmap for llms as assistants in exploring, analyzing, and visualizing knowledge graphs,” arXiv preprint arXiv:2404.01425, 2024.
- [129] H. Yan, T. Gui, J. Dai, Q. Guo, Z. Zhang, and X. Qiu, “A unified generative framework for various ner subtasks,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2021.
- [130] N. Ding, Y. Chen, X. Han, G. Xu, X. Wang, P. Xie, H. Zheng, Z. Liu, J. Li, and H.-G. Kim, “Prompt-learning for fine-grained entity typing,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2022.
- [131] S. Efeoglu and A. Paschke, “Retrieval-augmented generation-based relation extraction,” arXiv preprint arXiv:2404.13397, 2024.
- [132] B. Li, W. Yin, and M. Chen, “Ultra-fine entity typing with indirect supervision from natural language inference,” Transactions of the Association for Computational Linguistics, vol. 10, pp. 607–622, 2022.
- [133] T. Ayoola, S. Tyagi, J. Fisher, C. Christodoulopoulos, and A. Pierleoni, “Refined: An efficient zero-shot-capable approach to end-to-end entity linking,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2022.
- [134] A. Cattan, A. Eirew, G. Stanovsky, M. Joshi, and I. Dagan, “Cross-document coreference resolution over predicted mentions,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2021.
- [135] R. Thirukovalluru, N. Monath, K. Shridhar, M. Zaheer, M. Sachan, and A. McCallum, “Scaling within document coreference to long texts,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2021.
- [136] C. Alt, M. Hübner, and L. Hennig, “Improving relation extraction by pre-trained language representations,” in Proceedings of the Conference on Automated Knowledge Base Construction (AKBC), 2019.

- [137] Q. Guo, Y. Sun, G. Liu, Z. Wang, Z. Ji, Y. Shen, and X. Wang, “Constructing chinese historical literature knowledge graph based on bert,” in Proceedings of the International Conference on Web Information Systems and Applications (WISA), 2021.
- [138] A. Bosselut, H. Rashkin, M. Sap, C. Malaviya, A. Celikyilmaz, and Y. Choi, “Comet: Commonsense transformers for knowledge graph construction,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2019.
- [139] S. Hao, B. Tan, K. Tang, H. Zhang, E. P. Xing, and Z. Hu, “Bertnet: Harvesting knowledge graphs from pretrained language models,” arXiv preprint arXiv:2206.14268, 2022.
- [140] P. West, C. Bhagavatula, J. Hessel, J. Hwang, L. Jiang, R. Le Bras, X. Lu, S. Welleck, and Y. Choi, “Symbolic knowledge distillation: from general language models to commonsense models,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2022.
- [141] L. Hu, Z. Liu, Z. Zhao, L. Hou, L. Nie, and J. Li, “A survey of knowledge enhanced pre-trained language models,” IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 36, pp. 1413–1430, 2023.
- [142] X. Wei, S. Wang, D. Zhang, P. Bhatia, and A. Arnold, “Knowledge enhanced pretrained language models: A comprehensive survey,” arXiv preprint arXiv:2110.08455, 2021.
- [143] D. Yin, L. Dong, H. Cheng, X. Liu, K.-W. Chang, F. Wei, and J. Gao, “A survey of knowledge-intensive nlp with pre-trained language models,” arXiv preprint arXiv:2202.08772, 2022.
- [144] S. Wang, Z. Wei, J. Xu, T. Li, and Z. Fan, “Unifying structure reasoning and language model pre-training for complex reasoning,” arXiv preprint arXiv:2301.08913, 2023.
- [145] Y. Sun, Q. Shi, L. Qi, and Y. Zhang, “Jointlk: Joint reasoning with language models and knowledge graphs for commonsense question answering,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2022.
- [146] Y. Liu, Y. Wan, L. He, H. Peng, and S. Y. Philip, “Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2021.
- [147] P. Qi, H. Lee, O. T. Sido, and C. D. Manning, “Answering open-domain questions of varying reasoning steps from text,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2021.
- [148] C. Mavromatis and G. Karypis, “Gnn-rag: Graph neural retrieval for large language model reasoning,” arXiv preprint arXiv:2405.20139, 2024.
- [149] X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. D. Manning, and J. Leskovec, “Greaselm: Graph reasoning enhanced language models,” in Proceedings of the International Conference on Learning Representations (ICLR), 2022.
- [150] Y. Feng, X. Chen, B. Y. Lin, P. Wang, J. Yan, and X. Ren, “Scalable multi-hop relational reasoning for knowledge-aware question answering,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.

- [151] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, “Qa-gnn: Reasoning with language models and knowledge graphs for question answering,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2021.
- [152] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, “Ernie: Enhanced language representation with informative entities,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2019.
- [153] Y. Sun, S. Wang, S. Feng, S. Ding, C. Pang, J. Shang, J. Liu, X. Chen, Y. Zhao, Y. Lu, W. Liu, Z. Wu, W. Gong, J. Liang, Z. Shang, P. Sun, W. Liu, X. Ouyang, D. Yu, H. Tian, H. Wu, and H. Wang, “Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation,” arXiv preprint arXiv:2107.02137, 2021.
- [154] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, “K-bert: Enabling language representation with knowledge graph,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2020.
- [155] D. Dai, L. Dong, Y. Hao, Z. Sui, B. Chang, and F. Wei, “Knowledge neurons in pretrained transformers,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2022.
- [156] T. Shen, Y. Mao, P. He, G. Long, A. Trischler, and W. Chen, “Exploiting structured knowledge in text via graph-guided representation learning,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.
- [157] D. Zhang, Z. Yuan, Y. Liu, F. Zhuang, H. Chen, and H. Xiong, “E-bert: A phrase and product knowledge enhanced language model for e-commerce,” arXiv preprint arXiv:2009.02835, 2020.
- [158] H. Tian, C. Gao, X. Xiao, H. Liu, B. He, H. Wu, H. Wang, and F. Wu, “Skep: Sentiment knowledge enhanced pre-training for sentiment analysis,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2020.
- [159] C. Rosset, C. Xiong, M. Phan, X. Song, P. Bennett, and S. Tiwary, “Knowledge-aware language model pretraining,” arXiv preprint arXiv:2007.00655, 2020.
- [160] S. Li, X. Li, L. Shang, C.-J. Sun, B. Liu, Z. Ji, X. Jiang, and Q. Liu, “Pre-training language models with deterministic factual knowledge,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022.
- [161] W. Xiong, J. Du, W. Y. Wang, and V. Stoyanov, “Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model,” in Proceedings of the International Conference on Learning Representations (ICLR), 2020.
- [162] B. He, D. Zhou, J. Xiao, X. Jiang, Q. Liu, N. J. Yuan, and T. Xu, “Bert-mk: Integrating graph contextualized knowledge into pre-trained language models,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2020.
- [163] Y. Su, X. Han, Z. Zhang, Y. Lin, P. Li, Z. Liu, J. Zhou, and M. Sun, “Cokebert: Contextual knowledge selection and embedding towards enhanced pre-trained language models,” AI Open, vol. 2, pp. 127–134, 2021.

- [164] H. Zhu, H. Peng, Z. Lyu, L. Hou, J. Li, and J. Xiao, “Pre-training language model incorporating domain-specific heterogeneous knowledge into a unified representation,” Expert Systems with Applications, vol. 215, p. 119369, 2023.
- [165] C. Feng, X. Zhang, and Z. Fei, “Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs,” arXiv preprint arXiv:2309.03118, 2023.
- [166] N. Huang, Y. R. Deshpande, Y. Liu, H. Alberts, K. Cho, C. Vania, and I. Calixto, “Endowing language models with multimodal knowledge graph representations,” arXiv preprint arXiv:2206.13163, 2022.
- [167] O. Agarwal, H. Ge, S. Shakeri, and R. Al-Rfou, “Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2021.
- [168] Y. Xu, M. Namazifar, D. Hazarika, A. Padmakumar, Y. Liu, and D. Hakkani-Tur, “Kilm: Knowledge injection into encoder-decoder language models,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2023.
- [169] B. Oguz, X. Chen, V. Karpukhin, S. Peshterliev, D. Okhonko, M. Schlichtkrull, S. Gupta, Y. Mehdad, and S. Yih, “Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering,” in Findings of the Association for Computational Linguistics: NAACL (NAACL-Findings), 2022.
- [170] Y. Tan, Z. Zhou, H. Lv, W. Liu, and C. Yang, “Walklm: A uniform language model fine-tuning framework for attributed graph embedding,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2024.
- [171] R. Xu, W. Shi, Y. Yu, Y. Zhuang, Y. Zhu, M. D. Wang, J. C. Ho, C. Zhang, and C. Yang, “Bmretriever: Tuning large language models as better biomedical text retrievers,” in Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2024.
- [172] T. Sun, Y. Shao, X. Qiu, Q. Guo, Y. Hu, X.-J. Huang, and Z. Zhang, “Colake: Contextualized language and knowledge embedding,” in Proceedings of the International Conference on Computational Linguistics (COLING), 2020.
- [173] T. Zhang, C. Wang, N. Hu, M. Qiu, C. Tang, X. He, and J. Huang, “Dkplm: decomposable knowledge-enhanced pre-trained language model for natural language understanding,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2022.
- [174] H. Ye, N. Zhang, S. Deng, X. Chen, H. Chen, F. Xiong, X. Chen, and H. Chen, “Ontology-enhanced prompt-tuning for few-shot learning,” in Proceedings of the International World Wide Web Conference (WWW), 2022.
- [175] H. Luo, Z. Tang, S. Peng, Y. Guo, W. Zhang, C. Ma, G. Dong, M. Song, W. Lin, Y. Zhu, and L. A. Tuan, “Chatkbqa: A generate-then-retrieve framework for knowledge base question answering with fine-tuned large language models,” arXiv preprint arXiv:2310.08975, 2023.
- [176] A. Martino, M. Iannelli, and C. Truong, “Knowledge injection to counter large language model (llm) hallucination,” in Proceedings of the European Semantic Web Conference (ESWC), 2023.

- [177] W. Chen, Y. Su, X. Yan, and W. Y. Wang, “Kgpt: Knowledge-grounded pre-training for data-to-text generation,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.
- [178] C. N. d. Santos, Z. Dong, D. Cer, J. Nham, S. Shakeri, J. Ni, and Y.-H. Sung, “Knowledge prompts: Injecting world knowledge into language models through soft prompts,” arXiv preprint arXiv:2210.04726, 2022.
- [179] F. Moiseev, Z. Dong, E. Alfonseca, and M. Jaggi, “Skill: Structured knowledge infusion for large language models,” in Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2022.
- [180] S. Li, Y. Gao, H. Jiang, Q. Yin, Z. Li, X. Yan, C. Zhang, and B. Yin, “Graph reasoning for question answering with triplet retrieval,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2023.
- [181] J. Jiang, K. Zhou, X. Zhao, and J.-R. Wen, “Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph,” in Proceedings of the International Conference on Learning Representations (ICLR), 2023.
- [182] L. Luo, Y.-F. Li, R. Haf, and S. Pan, “Reasoning on graphs: Faithful and interpretable large language model reasoning,” in Proceedings of the International Conference on Learning Representations (ICLR), 2024.
- [183] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, L. Ni, H.-Y. Shum, and J. Guo, “Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph,” in Proceedings of the International Conference on Learning Representations (ICLR), 2024.
- [184] Y. Wang, B. Jiang, Y. Luo, D. He, P. Cheng, and L. Gao, “Reasoning on efficient knowledge paths: Knowledge graph guides large language model for domain question answering,” arXiv preprint arXiv:2404.10384, 2024.
- [185] L. Luo, J. Ju, B. Xiong, Y.-F. Li, G. Haffari, and S. Pan, “Chatrule: Mining logical rules with large language models for knowledge graph reasoning,” arXiv preprint arXiv:2309.01538, 2023.
- [186] Y. Wang, Z. Chu, X. Ouyang, S. Wang, H. Hao, Y. Shen, J. Gu, S. Xue, J. Y. Zhang, Q. Cui, L. Li, J. Zhou, and S. Li, “Enhancing recommender systems with large language model reasoning graphs,” arXiv preprint arXiv:2308.10835, 2023.
- [187] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, P. Nyczyk, and T. Hoefler, “Graph of thoughts: Solving elaborate problems with large language models,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2024.
- [188] X. Wang, P. Kapanipathi, R. Musa, M. Yu, K. Talamadupula, I. Abdelaziz, M. Chang, A. Fokoue, B. Makni, N. Mattei, and M. Witbrock, “Improving natural language inference using external knowledge in the science questions domain,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2019.
- [189] Y. Wang, N. Lipka, R. A. Rossi, A. Siu, R. Zhang, and T. Derr, “Knowledge graph prompting for multi-document question answering,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2024.

- [190] D. Allemang and J. Sequeda, “Increasing the llm accuracy for question answering: Ontologies to the rescue!” arXiv preprint arXiv:2405.11706, 2024.
- [191] Z. Ji, Z. Liu, N. Lee, T. Yu, B. Wilie, M. Zeng, and P. Fung, “Rho: Reducing hallucination in open-domain dialogues with knowledge grounding,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2023.
- [192] S. Feng, V. Balachandran, Y. Bai, and Y. Tsvetkov, “Factkb: Generalizable factuality evaluation using language models enhanced with factual knowledge,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2023.
- [193] D. McDonald, R. Papadopoulos, and L. Benningfield, “Reducing llm hallucination using knowledge distillation: A case study with mistral large and mmlu benchmark,” Preprint available at techrxiv.171665607, 2024.
- [194] Y. Lee, J. J. Y. Chung, T. S. Kim, J. Y. Song, and J. Kim, “Promptiverse: Scalable generation of scaffolding prompts through human-ai hybrid knowledge graph annotation,” in Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI), 2022.
- [195] R. Brate, M.-H. Dang, F. Hoppe, Y. He, A. Meroño-Peñuela, and V. Sadashivaiah, “Improving language model predictions via prompts enriched with knowledge graphs?” in ISWC Wrokshop on Deep Learning for Knowledge Graphs, 2022.
- [196] Y. Wen, Z. Wang, and J. Sun, “Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models,” arXiv preprint arXiv:2308.09729, 2023.
- [197] D. Wilmot and F. Keller, “Memory and knowledge augmented language models for inferring salience in long-form stories,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2021.
- [198] R. Logan, N. F. Liu, M. E. Peters, M. Gardner, and S. Singh, “Barack’s wife hillary: Using knowledge graphs for fact-aware language modeling,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2019.
- [199] Y. Wu, Y. Zhao, B. Hu, P. Minervini, P. Stenetorp, and S. Riedel, “An efficient memory-augmented transformer for knowledge-intensive nlp tasks,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022.
- [200] X. Guan, Y. Liu, H. Lin, Y. Lu, B. He, X. Han, and L. Sun, “Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2024.
- [201] J. Dong, B. Fatemi, B. Perozzi, L. F. Yang, and A. Tsitsulin, “Don’t forget to connect! improving rag with graph-based reranking,” arXiv preprint arXiv:2405.18414, 2024.
- [202] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, “Raptor: Recursive abstractive processing for tree-organized retrieval,” in Proceedings of the International Conference on Learning Representations (ICLR), 2024.
- [203] X. He, Y. Tian, Y. Sun, N. V. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, “G-retriever: Retrieval-augmented generation for textual graph understanding and question answering,” arXiv preprint arXiv:2402.07630, 2024.

- [204] J. Han, N. Collier, W. Buntine, and E. Shareghi, “Pive: Prompting with iterative verification improving graph-based generative capability of llms,” arXiv preprint arXiv:2305.12392, 2023.
- [205] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, “From local to global: A graph rag approach to query-focused summarization,” arXiv preprint arXiv:2404.16130, 2024.
- [206] B. J. Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, “Hipporag: Neurobiologically inspired long-term memory for large language models,” arXiv preprint arXiv:2405.14831, 2024.
- [207] X. Liang, S. Niu, S. Zhang, S. Song, H. Wang, J. Yang, F. Xiong, B. Tang, and C. Xi, “Empowering large language models to set up a knowledge retrieval indexer via self-learning,” arXiv preprint arXiv:2405.16933, 2024.
- [208] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu, and A. Miller, “Language models as knowledge bases?” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019.
- [209] Z. Jiang, F. F. Xu, J. Araki, and G. Neubig, “How can we know what language models know?” Transactions of the Association for Computational Linguistics, vol. 8, pp. 423–438, 2020.
- [210] L. Adolphs, S. Dhuliawala, and T. Hofmann, “How to query language models?” arXiv preprint arXiv:2108.01928, 2021.
- [211] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh, “Autoprompt: Eliciting knowledge from language models with automatically generated prompts,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.
- [212] A. Mallen, A. Asai, V. Zhong, R. Das, H. Hajishirzi, and D. Khashabi, “When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories,” arXiv preprint arXiv:2212.10511, 2022.
- [213] W. W. Cohen, W. Chen, M. De Jong, N. Gupta, A. Presta, P. Verga, and J. Wieting, “Qa is the new kr: Question-answer pairs as knowledge bases,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2023.
- [214] L. Luo, T. Vu, D. Phung, and R. Haf, “Systematic assessment of factual knowledge in large language models,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2023.
- [215] A. Orogat, I. Liu, and A. El-Roby, “Cbench: towards better evaluation of question answering over knowledge graphs,” in Proceedings of the International Conference on Very Large Data Bases (VLDB), 2021.
- [216] Y. Bai, S. Feng, V. Balachandran, Z. Tan, S. Lou, T. He, and Y. Tsvetkov, “Kgquiz: Evaluating the generalization of encoded knowledge in large language models,” in Proceedings of the International World Wide Web Conference (WWW), 2024.
- [217] X. Ho, A.-K. D. Nguyen, S. Sugawara, and A. Aizawa, “Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps,” in Proceedings of the International Conference on Computational Linguistics (COLING), 2020.

- [218] M. Tracy, M. Cerdá, and K. M. Keyes, “Agent-based modeling in public health: current applications and future directions,” Annual Review of Public Health, vol. 39, no. 1, pp. 77–94, 2018.
- [219] X. Tang, A. Zou, Z. Zhang, Z. Li, Y. Zhao, X. Zhang, A. Cohan, and M. Gerstein, “Medagents: Large language models as collaborators for zero-shot medical reasoning,” in ICLR Workshop on LLM Agents, 2024.
- [220] D. Kaur, S. Uslu, M. Durresi, and A. Durresi, “Llm-based agents utilized in a trustworthy artificial conscience model for controlling ai in medical applications,” in International Conference on Advanced Information Networking and Applications, 2024.
- [221] J. Li, S. Wang, M. Zhang, W. Li, Y. Lai, X. Kang, W. Ma, and Y. Liu, “Agent hospital: A simulacrum of hospital with evolvable medical agents,” arXiv preprint arXiv:2405.02957, 2024.
- [222] Y. Kim, C. Park, H. Jeong, Y. S. Chan, X. Xu, D. McDuff, C. Breazeal, and H. W. Park, “Adaptive collaboration strategy for llms in medical decision making,” arXiv preprint arXiv:2404.15155, 2024.
- [223] S. A. Gebreab, K. Salah, R. Jayaraman, M. H. ur Rehman, and S. Ellaham, “Llm-based framework for administrative task automation in healthcare,” in Proceedings of the International Symposium on Digital Forensics and Security, 2024.
- [224] L. Yue and T. Fu, “Ct-agent: Clinical trial multi-agent with large language model-based reasoning,” arXiv preprint arXiv:2404.14777, 2024.
- [225] B. Pan, J. Lu, K. Wang, L. Zheng, Z. Wen, Y. Feng, M. Zhu, and W. Chen, “Agentcoord: Visually exploring coordination strategy for llm-based multi-agent collaboration,” arXiv preprint arXiv:2404.11943, 2024.
- [226] Y. Xiao, J. Liu, Y. Zheng, X. Xie, J. Hao, M. Li, R. Wang, F. Ni, Y. Li, J. Luo, S. Jiao, and J. Peng, “Cellagent: An llm-driven multi-agent framework for automated single-cell data analysis,” bioRxiv, pp. 2024–05, 2024.
- [227] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, “Freebase: a collaboratively created graph database for structuring human knowledge,” in Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2008.
- [228] F. M. Suchanek, G. Kasneci, and G. Weikum, “Yago: A large ontology from wikipedia and wordnet,” Journal of Web Semantics, vol. 6, no. 3, pp. 203–217, 2008.
- [229] D. Vrandečić and M. Krötzsch, “Wikidata: a free collaborative knowledgebase,” Communications of the ACM, vol. 57, no. 10, pp. 78–85, 2014.
- [230] Y. Yan, Y. Hou, Y. Xiao, R. Zhang, and Q. Wang, “Knownet: Guided health information seeking from llms via knowledge graph integration,” IEEE Transactions on Visualization and Computer Graphics, 2024.
- [231] J. Lu and C. Yang, “Open-world taxonomy and knowledge graph co-learning,” in Proceedings of the Conference on Automated Knowledge Base Construction (AKBC), 2022.
- [232] D. S. Himmelstein, A. Lizee, C. Hessler, L. Brueggeman, S. L. Chen, D. Hadley, A. Green, P. Khankhanian, and S. E. Baranzini, “Systematic integration of biomedical knowledge prioritizes drugs for repurposing,” Elife, vol. 6, p. e26726, 2017.

- [233] X. Liu, H. Hong, X. Wang, Z. Chen, E. Kharlamov, Y. Dong, and J. Tang, “Selfkg: Self-supervised entity alignment in knowledge graphs,” in Proceedings of the ACM Web Conference 2022, 2022, pp. 860–870.
- [234] Z. Wang, Q. Lv, X. Lan, and Y. Zhang, “Cross-lingual knowledge graph alignment via graph convolutional networks,” in Proceedings of the 2018 conference on empirical methods in natural language processing, 2018, pp. 349–357.
- [235] Q. Zhu, H. Wei, B. Sisman, D. Zheng, C. Faloutsos, X. L. Dong, and J. Han, “Collective multi-type entity alignment between knowledge graphs,” in Proceedings of The Web Conference 2020, 2020, pp. 2241–2252.
- [236] Y. Yan, L. Liu, Y. Ban, B. Jing, and H. Tong, “Dynamic knowledge graph alignment,” in Proceedings of the AAAI conference on artificial intelligence, vol. 35, no. 5, 2021, pp. 4564–4572.
- [237] P. Sancheti, K. Karlapalem, and K. Vemuri, “Llm driven web profile extraction for identical names,” in Companion Proceedings of the ACM on Web Conference 2024, 2024, pp. 1616–1625.
- [238] M. Agrawal, S. Hegselmann, H. Lang, Y. Kim, and D. Sontag, “Large language models are few-shot clinical information extractors,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022.
- [239] J. Lu, J. Shen, B. Xiong, W. Ma, S. Staab, and C. Yang, “Hiprompt: Few-shot biomedical knowledge fusion via hierarchy-oriented prompting,” in Proceedings of the ACM International Conference on Research and Development in Information Retrieval (SIGIR), 2023.
- [240] Y. Xie, J. Lu, J. Ho, F. Nahab, X. Hu, and C. Yang, “Promptlink: Leveraging large language models for cross-source biomedical concept linking,” in Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2589–2593.
- [241] R. Zhang, Y. Su, B. D. Trisedya, X. Zhao, M. Yang, H. Cheng, and J. Qi, “Autoalign: fully automatic and effective knowledge graph alignment enabled by large language models,” IEEE Transactions on Knowledge and Data Engineering, 2023.
- [242] L. Yang, H. Chen, Z. Li, X. Ding, and X. Wu, “Give us the facts: Enhancing large language models with knowledge graphs for fact-aware language modeling,” IEEE Transactions on Knowledge and Data Engineering, 2024.
- [243] N. Hassan, G. Zhang, F. Arslan, J. Caraballo, D. Jimenez, S. Gawsane, S. Hasan, M. Joseph, A. Kulkarni, A. K. Nayak et al., “Claimbuster: The first-ever end-to-end fact-checking system,” Proceedings of the VLDB Endowment, vol. 10, no. 12, pp. 1945–1948, 2017.
- [244] P. Nakov, D. Corney, M. Hasanain, F. Alam, T. Elsayed, A. Barrón-Cedeño, P. Papotti, S. Shaar, G. Da San Martino et al., “Automated fact-checking for assisting human fact-checkers,” in Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, 2021, pp. 4551–4558.
- [245] J. Yu, X. Wang, S. Tu, S. Cao, D. Zhang-Li, X. Lv, H. Peng, Z. Yao, X. Zhang, H. Li, C. Li, Z. Zhang, Y. Bai, Y. Liu, A. Xin, K. Yun, L. GONG, N. Lin, J. Chen, Z. Wu, Y. Qi, W. Li, Y. Guan, K. Zeng, J. Qi, H. Jin, J. Liu, Y. Gu, Y. Yao, N. Ding, L. Hou, Z. Liu, X. Bin, J. Tang, and J. Li, “KoLA: Carefully benchmarking world knowledge of large language models,” in The Twelfth International Conference on Learning Representations, 2024.

- [246] R. Xu, H. Cui, Y. Yu, X. Kan, W. Shi, Y. Zhuang, W. Jin, J. Ho, and C. Yang, “Knowledge-infused prompting: Assessing and advancing clinical text data generation with large language models,” in Findings of the Association for Computational Linguistics: ACL 2024, 2024.
- [247] B. AlKhamissi, M. Li, A. Celikyilmaz, M. Diab, and M. Ghazvininejad, “A review on language models as knowledge bases,” arXiv preprint arXiv:2204.06031, 2022.
- [248] Y. Wei, Q. Huang, Y. Zhang, and J. Kwok, “KICGPT: Large language model with knowledge in context for knowledge graph completion,” in Findings of the Association for Computational Linguistics: EMNLP 2023. Singapore: Association for Computational Linguistics, 2023, pp. 8667–8683.
- [249] Y. Wang, M. Hu, Z. Huang, D. Li, D. Yang, and X. Lu, “KC-GenRe: A knowledge-constrained generative re-ranking method based on large language models for knowledge graph completion,” in Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, 2024, pp. 9668–9680.
- [250] Z. Bi, J. Chen, Y. Jiang, F. Xiong, W. Guo, H. Chen, and N. Zhang, “Codekgc: Code language model for generative knowledge graph construction,” ACM Transactions on Asian and Low-Resource Language Information Processing, vol. 23, no. 3, pp. 1–16, 2024.
- [251] Q. Zeng, Y. Bai, Z. Tan, Z. Wu, S. Feng, and M. Jiang, “Codetaxo: Enhancing taxonomy expansion with limited examples via code language prompts,” arXiv preprint arXiv:2408.09070, 2024.
- [252] A. Madaan, S. Zhou, U. Alon, Y. Yang, and G. Neubig, “Language models of code are few-shot commonsense learners,” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, 2022, pp. 1384–1403.
- [253] W. Chen, X. Ma, X. Wang, and W. W. Cohen, “Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks,” Transactions on Machine Learning Research, 2023.
- [254] W. Shi, R. Xu, Y. Zhuang, Y. Yu, J. Zhang, H. Wu, Y. Zhu, J. C. Ho, C. Yang, and M. D. Wang, “Ehrgent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records,” in ICLR 2024 Workshop on Large Language Model (LLM) Agents, 2024.
- [255] Y. Zhang, Z. Chen, L. Guo, Y. Xu, W. Zhang, and H. Chen, “Making large language models perform better in knowledge graph completion,” in Proceedings of the 32nd ACM International Conference on Multimedia. Association for Computing Machinery, 2024, p. 233–242.
- [256] P. Jiang, L. Cao, C. Xiao, P. Bhatia, J. Sun, and J. Han, “Kg-fit: Knowledge graph fine-tuning upon open-world knowledge,” arXiv preprint arXiv:2405.16412, 2024.
- [257] Z. Yang, L. Li, K. Lin, J. Wang, C.-C. Lin, Z. Liu, and L. Wang, “The dawn of lmms: Preliminary explorations with gpt-4v (ision),” arXiv preprint arXiv:2309.17421, 2023.
- [258] N. Fei, Z. Lu, Y. Gao, G. Yang, Y. Huo, J. Wen, H. Lu, R. Song, X. Gao, T. Xiang, H. Sun, and J.-R. Wen, “Towards artificial general intelligence via a multimodal foundation model,” Nature Communications, vol. 13, no. 1, p. 3094, 2022.

- [259] C. Li, Z. Gan, Z. Yang, J. Yang, L. Li, L. Wang, and J. Gao, “Multimodal foundation models: From specialists to general-purpose assistants,” Foundations and Trends in Computer Graphics and Vision, vol. 16, no. 1-2, pp. 1–214, 2024.
- [260] C. Li, C. Wong, S. Zhang, N. Usuyama, H. Liu, J. Yang, T. Naumann, H. Poon, and J. Gao, “Llava-med: Training a large language-and-vision assistant for biomedicine in one day,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2024.
- [261] K. Zhang, J. Yu, E. Adhikarla, R. Zhou, Z. Yan, Y. Liu, Z. Liu, L. He, B. Davison, X. Li, H. Ren, S. Fu, J. Zou, W. Liu, J. Huang, C. Chen, Y. Zhou, T. Liu, X. Chen, Y. Chen, Q. Li, H. Liu, and L. Sun, “Biomedgpt: a unified and generalist biomedical generative pre-trained transformer for vision, language, and multimodal tasks,” arXiv preprint arXiv:2305.17100, 2023.
- [262] Z. Chen, Y. Zhang, Y. Fang, Y. Geng, L. Guo, X. Chen, Q. Li, W. Zhang, J. Chen, Y. Zhu *et al.*, “Knowledge graphs meet multi-modal learning: A comprehensive survey,” arXiv preprint arXiv:2402.05391, 2024.
- [263] L. Sun, K. Zhang, Q. Li, and R. Lou, “Umie: Unified multimodal information extraction with instruction tuning,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 17, 2024, pp. 19 062–19 070.
- [264] J. Li, H. Li, Z. Pan, D. Sun, J. Wang, W. Zhang, and G. Pan, “Prompting chatgpt in mner: Enhanced multimodal named entity recognition with auxiliary refined knowledge,” in Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 2787–2802.
- [265] X. Hu, J. Chen, A. Liu, S. Meng, L. Wen, and P. S. Yu, “Prompt me up: Unleashing the power of alignments for multimodal entity and relation extraction,” in Proceedings of the 31st ACM International Conference on Multimedia, 2023, pp. 5185–5194.
- [266] Q. Yu, J. Li, Y. Wu, S. Tang, W. Ji, and Y. Zhuang, “Visually-prompted language model for fine-grained scene graph generation in an open world,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 21 560–21 571.
- [267] L. Li, J. Xiao, G. Chen, J. Shao, Y. Zhuang, and L. Chen, “Zero-shot visual relation detection via composite visual cues from large language models,” Advances in Neural Information Processing Systems, vol. 36, 2024.
- [268] R. Li, S. Zhang, D. Lin, K. Chen, and X. He, “From pixels to graphs: Open-vocabulary scene graph generation with vision-language models,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 28 076–28 086.
- [269] S. Chen, J. Tracey, A. Bies, and S. Strassel, “Schema learning corpus: Data and annotation focused on complex events,” in Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), 2024, pp. 14 393–14 399.
- [270] H. Rasheed, M. Maaz, S. Shaji, A. Shaker, S. Khan, H. Cholakkal, R. M. Anwer, E. Xing, M.-H. Yang, and F. S. Khan, “Glamm: Pixel grounding large multimodal model,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 13 009–13 018.
- [271] H. Cui, X. Fang, Z. Zhang, R. Xu, X. Kan, X. Liu, Y. Yu, M. Li, Y. Song, and C. Yang, “Open visual knowledge extraction via relation-oriented multimodality model prompting,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2024.

- [272] Z. Yang, H. Zhang, F. Chen, A. Bolimera, and M. Savvides, “Hierarchical knowledge graph construction from images for scalable e-commerce,” [arXiv preprint arXiv:2410.21237](#), 2024.
- [273] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin et al., “A survey on large language model based autonomous agents,” [Frontiers of Computer Science](#), vol. 18, no. 6, p. 186345, 2024.
- [274] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, “Bloomberggpt: A large language model for finance,” [arXiv preprint arXiv:2303.17564](#), 2023.
- [275] J. Cui, Z. Li, Y. Yan, B. Chen, and L. Yuan, “Chatlaw: Open-source legal large language model with integrated external knowledge bases,” [arXiv preprint arXiv:2306.16092](#), 2023.
- [276] Y. Chen and J. Zou, “Genept: A simple but effective foundation model for genes and cells built from chatgpt,” [bioRxiv](#), 2024.
- [277] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, P. Payne, M. Seneviratne, P. Gamble, C. Kelly, N. Scharli, A. Chowdhery, P. Mansfield, B. Aguera y Arcas, D. Webster, G. S. Corrado, Y. Matias, K. Chou, J. Gottweis, N. Tomasev, Y. Liu, A. Rajkomar, J. Barral, C. Sementurs, A. Karthikesalingam, and V. Natarajan, “Large language models encode clinical knowledge,” [Nature](#), vol. 620, no. 7972, pp. 172–180, 2023.
- [278] K. Singhal, T. Tu, J. Gottweis, R. Sayres, E. Wulczyn, L. Hou, K. Clark, S. Pfohl, H. Cole-Lewis, D. Neal, M. Schaeckermann, A. Wang, M. Amin, S. Lachgar, P. Mansfield, S. Prakash, B. Green, E. Dominowska, B. Aguera y Arcas, N. Tomasev, Y. Liu, R. Wong, C. Sementurs, S. S. Mahdavi, J. Barral, D. Webster, G. S. Corrado, Y. Matias, S. Azizi, A. Karthikesalingam, and V. Natarajan, “Towards expert-level medical question answering with large language models,” [arXiv preprint arXiv:2305.09617](#), 2023.
- [279] C. E. Haupt and M. Marks, “Ai-generated medical advice? gpt and beyond,” [JAMA](#), vol. 329, no. 16, pp. 1349–1350, 2023.
- [280] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, “Capabilities of gpt-4 on medical challenge problems,” [arXiv preprint arXiv:2303.13375](#), 2023.
- [281] P. Lee, S. Bubeck, and J. Petro, “Benefits, limits, and risks of gpt-4 as an ai chatbot for medicine,” [New England Journal of Medicine](#), vol. 388, no. 13, pp. 1233–1239, 2023.
- [282] S. L. Fleming, K. Morse, A. Kumar, C.-C. Chiang, B. Patel, E. Brunskill, and N. Shah, “Assessing the potential of usmle-like exam questions generated by gpt-4,” [medRxiv](#), pp. 2023–04, 2023.
- [283] Z. Chen, A. H. Cano, A. Romanou, A. Bonnet, K. Matoba, F. Salvi, M. Pagliardini, S. Fan, A. Köpf, A. Mohtashami, A. Sallinen, A. Sakhaeirad, V. Swamy, I. Krawczuk, D. Bayazit, A. Marmet, S. Montariol, M.-A. Hartley, M. Jaggi, and A. Bosselut, “Meditron-70b: Scaling medical pretraining for large language models,” [arXiv preprint arXiv:2311.16079](#), 2023.
- [284] X. Yang, A. Chen, N. PourNejatian, H. C. Shin, K. E. Smith, C. Parisien, C. Compas, C. Martin, A. B. Costa, M. G. Flores, Y. Zhang, T. Magoc, C. A. Harle, G. Lipori, D. A. Mitchell, W. R. Hogan, E. A. Shenkman, J. Bian, and Y. Wu, “A large language model for electronic health records,” [NPJ Digital Medicine](#), vol. 5, no. 1, p. 194, 2022.

- [285] R. Luo, L. Sun, Y. Xia, T. Qin, S. Zhang, H. Poon, and T.-Y. Liu, “Biogpt: generative pre-trained transformer for biomedical text generation and mining,” Briefings in Bioinformatics, vol. 23, no. 6, p. bbac409, 2022.
- [286] N. Mehandru, B. Y. Miao, E. R. Almaraz, M. Sushil, A. J. Butte, and A. Alaa, “Evaluating large language models as agents in the clinic,” NPJ Digital Medicine, vol. 7, no. 1, p. 84, 2024.
- [287] S. S. Biswas, “Role of chat gpt in public health,” Annals of Biomedical Engineering, vol. 51, no. 5, pp. 868–869, 2023.
- [288] D. Dash, R. Thapa, J. M. Banda, A. Swaminathan, M. Cheatham, M. Kashyap, N. Kotecha, J. H. Chen, S. Gombar, L. Downing, R. Pedreira, E. Goh, A. Arnaout, G. K. Morris, H. Magon, M. P. Lungren, E. Horvitz, and N. H. Shah, “Evaluation of gpt-3.5 and gpt-4 for supporting real-world information needs in healthcare delivery,” arXiv preprint arXiv:2304.13714, 2023.
- [289] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2020.
- [290] Y. Bang, S. Cahyawijaya, N. Lee, W. Dai, D. Su, B. Wilie, H. Lovenia, Z. Ji, T. Yu, W. Chung et al., “A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity,” in Proceedings of the International Joint Conference on Natural Language Processing and Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, 2023.
- [291] X. Kang, L. Qu, L.-K. Soon, Z. Li, and A. Trakic, “Bridging law and data: Augmenting reasoning via a semi-structured dataset with irac methodology,” arXiv preprint arXiv:2406.13217, 2024.
- [292] Y. Liu, Q. Zeng, J. Ordieres Meré, and H. Yang, “Anticipating stock market of the renowned companies: a knowledge graph approach,” Complexity, vol. 2019, 2019.
- [293] Y.-L. Tuan, S. Beygi, M. Fazel-Zarandi, Q. Gao, A. Cervone, and W. Y. Wang, “Towards large-scale interpretable knowledge graph reasoning for dialogue systems,” in Findings of the Association for Computational Linguistics (ACL-Findings), 2022.
- [294] J. Baek, A. F. Aji, J. Lehmann, and S. J. Hwang, “Direct fact retrieval from knowledge graphs without entity linking,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2023.
- [295] L. Wang, W. Xu, Y. Lan, Z. Hu, Y. Lan, R. K.-W. Lee, and E.-P. Lim, “Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models,” in Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), 2023.
- [296] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, and Y. Cao, “React: Synergizing reasoning and acting in language models,” in Proceedings of the International Conference on Learning Representations (ICLR), 2022.
- [297] H. Wang, H. Ren, and J. Leskovec, “Relational message passing for knowledge graph completion,” in Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2021.

- [298] X. Xu, P. Zhang, Y. He, C. Chao, and C. Yan, “Subgraph neighboring relations infomax for inductive link prediction on knowledge graphs,” in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2022.
- [299] K. Wang, F. Duan, S. Wang, P. Li, Y. Xian, C. Yin, W. Rong, and Z. Xiong, “Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering,” arXiv preprint arXiv:2308.13259, 2023.
- [300] J. Huang, X. Zhang, Q. Mei, and J. Ma, “Can llms effectively leverage graph structural information through prompts, and why?” Transactions on Machine Learning Research, 2024.
- [301] L. Luo, Z. Zhao, C. Gong, G. Haffari, and S. Pan, “Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models,” arXiv preprint arXiv:2410.13080, 2024.
- [302] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, “Chain-of-thought prompting elicits reasoning in large language models,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2022.
- [303] E. Fredkin, “Trie memory,” Communications of the ACM, vol. 3, no. 9, pp. 490–499, 1960.
- [304] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin et al., “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions,” arXiv preprint arXiv:2311.05232, 2023.
- [305] G. Agrawal, T. Kumarage, Z. Alghamdi, and H. Liu, “Can knowledge graphs reduce hallucinations in llms?: A survey,” arXiv preprint arXiv:2311.07914, 2023.
- [306] T. Nguyen, L. Luo, F. Shiri, D. Phung, Y.-F. Li, T.-T. Vu, and G. Haffari, “Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2024.
- [307] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, “Federated learning for smart healthcare: A survey,” ACM Computing Surveys, vol. 55, no. 3, pp. 1–37, 2022.
- [308] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, “Federated learning for healthcare: Systematic review and architecture proposal,” ACM Transactions on Intelligent Systems and Technology (TIST), vol. 13, no. 4, pp. 1–23, 2022.
- [309] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, “Federated learning for healthcare informatics,” Journal of Healthcare Informatics Research, vol. 5, pp. 1–19, 2021.
- [310] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, S. Ourselin, M. Sheller, R. M. Summers, A. Trsk, D. Xu, M. Baust, and M. J. Cardoso, “The future of digital health with federated learning,” NPJ Digital Medicine, vol. 3, no. 1, pp. 1–7, 2020.
- [311] A. Wei, N. Haghtalab, and J. Steinhardt, “Jailbroken: How does llm safety training fail?” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2024.
- [312] B. Wang, W. Chen, H. Pei, C. Xie, M. Kang, C. Zhang, C. Xu, Z. Xiong, R. Dutta, R. Schaeffer, S. T. Truong, S. Arora, M. Mazeika, D. Hendrycks, Z. Lin, Y. Cheng, S. Koyejo, D. Song, and B. Li, “Decodingtrust: A comprehensive assessment of trustworthiness in gpt models,” in Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), 2023.

- [313] Z. Xu, Y. Liu, G. Deng, Y. Li, and S. Picek, “Llm jailbreak attack versus defense techniques—a comprehensive study,” arXiv preprint arXiv:2402.13457, 2024.
- [314] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, “Fedgraphnn: A federated learning system and benchmark for graph neural networks,” in ICLR Workshop on Distributed and Private Machine Learning, 2021.
- [315] H. Xie, J. Ma, L. Xiong, and C. Yang, “Federated graph classification over non-iid graphs,” in Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), 2021.
- [316] K. Zhang, C. Yang, X. Li, L. Sun, and S. M. Yiu, “Subgraph federated learning with missing neighbor generation,” in Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS), 2021.
- [317] K. Zhang, L. Sun, B. Ding, S. M. Yiu, and C. Yang, “Deep efficient private neighbor generation for subgraph federated learning,” in Proceedings of the 2024 SIAM International Conference on Data Mining (SDM), 2024.
- [318] H. Xie, L. Xiong, and C. Yang, “Federated node classification over graphs with latent link-type heterogeneity,” in Proceedings of the International World Wide Web Conference (WWW), 2023.
- [319] K. Zhang, Y. Wang, H. Wang, L. Huang, C. Yang, X. Chen, and L. Sun, “Efficient federated learning on knowledge graphs via privacy-preserving relation embedding aggregation,” in Findings of the Association for Computational Linguistics: EMNLP (EMNLP-Findings), 2022.
- [320] Z. Gu, K. Zhang, G. Bai, L. Chen, L. Zhao, and C. Yang, “Dynamic activation of clients and parameters for federated learning over heterogeneous graphs,” in Proceedings of the IEEE International Conference on Data Engineering (ICDE), 2023.
- [321] H. Xie, L. Xiong, and C. Yang, “Federated node classification over distributed ego-networks with secure contrastive embedding sharing,” in Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM), 2024.
- [322] T. Rebedea, R. Dinu, M. N. Sreedhar, C. Parisien, and J. Cohen, “Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails,” in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2023.
- [323] Z. Xiang, L. Zheng, Y. Li, J. Hong, Q. Li, H. Xie, J. Zhang, Z. Xiong, C. Xie, C. Yang, D. Song, and B. Li, “Guardagent: Safeguard llm agents by a guard agent via knowledge-enabled reasoning,” arXiv preprint arXiv:2406.09187, 2024.
- [324] S. Artiga, K. Orgera, and O. Pham, “Disparities in health and health care: Five key questions and answers,” Kaiser Family Foundation, 2020.
- [325] S. Artiga and E. Hinton, “Beyond health care: the role of social determinants in promoting health and health equity,” Kaiser Family Foundation, 2018.
- [326] W. C. on Social Determinants of Health and W. H. Organization, Closing the gap in a generation: health equity through action on the social determinants of health: Commission on Social Determinants of Health final report. World Health Organization, 2008.

- [327] M. Marmot and R. Wilkinson, Social determinants of health. Oup Oxford, 2005.
- [328] J. C. Phelan, B. G. Link, and P. Tehranifar, “Social conditions as fundamental causes of health inequalities: theory, evidence, and policy implications,” Journal of Health and Social Behavior, vol. 51, no. 1_suppl, pp. S28–S40, 2010.
- [329] R. Yang, H. Liu, Q. Zeng, Y. H. Ke, W. Li, L. Cheng, Q. Chen, J. Caverlee, Y. Matsuo, and I. Li, “Kg-rank: Enhancing large language models for medical qa with knowledge graphs and ranking techniques,” arXiv preprint arXiv:2403.05881, 2024.
- [330] M. Dehghan, M. Alomrani, S. Bagga, D. Alfonso-Hermelo, K. Bibi, A. Ghaddar, Y. Zhang, X. Li, J. Hao, Q. Liu, J. Lin, B. Chen, P. Parthasarathi, M. Biparva, and M. Rezagholizadeh, “EWEK-QA : Enhanced web and efficient knowledge graph retrieval for citation-based question answering systems,” in Findings of the Association for Computational Linguistics: ACL (ACL-Findings), 2024.
- [331] R. Xu, Z. Qi, C. Wang, H. Wang, Y. Zhang, and W. Xu, “Knowledge conflicts for llms: A survey,” arXiv preprint arXiv:2403.08319, 2024.
- [332] Y. Zhao, A. Devoto, G. Hong, X. Du, A. P. Gema, H. Wang, K.-F. Wong, and P. Minervini, “Steering knowledge selection behaviours in llms via sae-based representation engineering,” arXiv preprint arXiv:2410.15999, 2024.
- [333] F. Wang, X. Wan, R. Sun, J. Chen, and S. Ö. Arik, “Astute rag: Overcoming imperfect retrieval augmentation and knowledge conflicts for large language models,” arXiv preprint arXiv:2410.07176, 2024.
- [334] X. Zhu, Z. Li, X. Wang, X. Jiang, P. Sun, X. Wang, Y. Xiao, and N. J. Yuan, “Multi-modal knowledge graph construction and application: A survey,” IEEE Transactions on Knowledge and Data Engineering, vol. 36, no. 2, pp. 715–735, 2022.
- [335] X. Long, J. Zeng, F. Meng, Z. Ma, K. Zhang, B. Zhou, and J. Zhou, “Generative multi-modal knowledge retrieval with large language models,” in Proceedings of the AAAI International Conference on Artificial Intelligence (AAAI), 2024.
- [336] A. Kumar, C. Agarwal, S. Srinivas, A. J. Li, S. Feizi, and H. Lakkaraju, “Certifying llm safety against adversarial prompting,” arXiv preprint arXiv:2309.02705, 2023.
- [337] P. Cheng, Y. Ding, T. Ju, Z. Wu, W. Du, P. Yi, Z. Zhang, and G. Liu, “Trojanrag: Retrieval-augmented generation can be backdoor driver in large language models,” arXiv preprint arXiv:2405.13401, 2024.

Retrieval Augmented Generation in the Wild: A System 2 Perspective

Sajjadur Rahman* Dan Zhang* Nikita Bhutani Estevam Hruschka Eser Kandogan
Megagon Labs
{sajjadur, dan_z, nikita, estevam, eser}@megagon.ai

Abstract

Large language models (LLMs), despite their impressive capabilities in natural language understanding tasks in open-domain, often lack effectiveness with similar tasks in enterprise applications due to potential hallucinations, weak multi-hop reasoning ability, and limitations in adapting to heterogeneous data types, among others. Such issues primarily arise due to the absence of private, on-premises enterprises from an LLM’s training corpus. Knowledge-intensive tasks in enterprise often require multi-step reasoning, deep contextual understanding, and integration of information stored and accessed in heterogeneous formats (*e.g.*, tables, graphs, documents, and JSON), which LLMs aren’t inherently equipped to handle without significant adaptation. To this end, retrieval augmented generation (RAG) offers promise in instrumenting such adaptations on demand. While RAG-based approaches focus on controlling the generation and mitigating hallucinations, existing solutions are not sufficient for the requirements of the enterprise settings.

In this paper, we outline our approaches toward understanding and implementing a more effective RAG workflow in the wild. To achieve the goal, we draw on the cognitive science concepts of System 1 (fast, intuitive thinking) and System 2 (slow, deliberate, analytical thinking.) In particular, we discuss how existing RAG approaches are more aligned to System 1 and propose to shift from traditional single-model architectures to compound AI systems within a System 2 framework to improve RAG, especially in complex enterprise applications. Such compound AI systems adopt a more systematic approach by assigning specialized tasks to different intelligent agents, optimizing retrieval and generation performance with a retrieval-augmented generation workflow.

1 Introduction

Large Language Models (LLMs), despite their impressive performance across various natural language understanding tasks, exhibit significant limitations when applied to enterprise applications in the wild. Primarily, these models may hallucinate—generating plausible-sounding but factually incorrect content—when their parametric knowledge does not align with specific enterprise data [1–3]. An LLM’s parametric knowledge depends on its pre-training corpus and can also be influenced by the chosen training strategy and model architecture. This gap is especially problematic since enterprise applications frequently use private, on-premises data, which may differ substantially from the domains in LLMs’ pre-training corpora. Such domain misalignment can lead to severe inaccuracies, where the LLMs produce unreliable or misleading information. In addition, enterprises often require consistency and reliability in their outputs. However, LLMs can be sensitive to prompt wording [4], producing inconsistent results even with minor phrasing changes, which undermines their reliability in high-stakes enterprise tasks.

*The first two authors contributed equally.

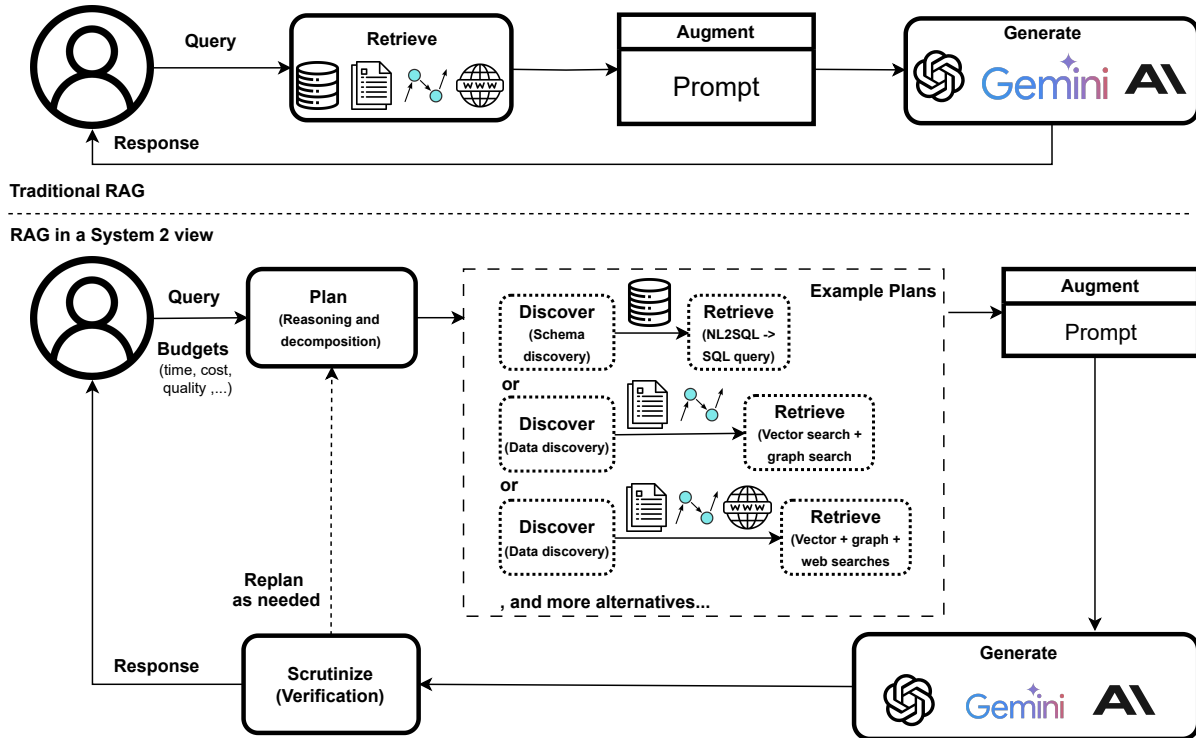


Figure 8: Compared to the traditional RAG setup (above), where a fixed Retrieval-Augmentation-Generation workflow is executed, a System 2 approach (below) involves more deliberate reasoning and action based on critical analysis. For instance, given the same user query, the system must first plan by analyzing the task and may decide to decompose it into smaller components, such as data discovery, natural language-to-query translation, and actual query execution. During the planning phase, constraints or budgets related to factors like time, cost, or quality, along with the nature of the multimodal data sources, may influence the direction of the workflow. After the generation step, a verification process is typically required to evaluate the outcome, which may lead to revisions in subsequent iterations.

Another key challenge for LLMs in enterprise settings is their limited capacity for complex reasoning. Many tasks in this domain require multi-step reasoning, deep contextual understanding, and coherent integration across different data sources, models, and pipelines. LLMs are not inherently designed to manage such complexities without extensive, task-specific adaptation and external grounding. These limitations hinder adopting LLMs in enterprise fields where accuracy, consistency, and reasoning depth are critical, such as healthcare, legal, HR, and data-driven decision-making applications.

Several approaches can help mitigate these limitations, including controlled generation, fact-checking, and post-processing. Controlled generation seeks to constrain the outputs of LLMs by guiding the model toward more reliable responses through techniques such as prompt engineering [5–7], fine-tuning [8, 9], and reinforcement learning from human feedback [10]. Fact-checking [11, 12] involves verifying generated content against highly credible sources to ensure accuracy, with any identified inaccuracies filtered out or corrected during post-processing. Although these techniques offer improvements, they may still struggle with domain-specific challenges and rapidly evolving knowledge. Among the various approaches, augmenting LLMs with external information sources has emerged as one of the most widely adopted

solutions for enhancing accuracy and robustness. This strategy enables LLMs to access up-to-date, domain-specific knowledge, making them more adaptable to new fields or emerging topics. Techniques like retrieval-augmented generation (RAG) [13] integrate external databases or knowledge graphs to ground the model’s outputs in verifiable information. By incorporating external data, such as multi-modal documents or enterprise-specific datasets, LLMs can produce more accurate and contextually relevant responses, thereby reducing the likelihood of hallucinations and making them more suitable for critical applications. However, simply using RAG approaches is not necessarily yet the definitive solution. Challenges remain, such as ensuring the quality and reliability of the retrieved information, handling ambiguous or conflicting data, and seamlessly integrating retrieval with generation to maintain coherent and contextually appropriate responses. Consequently, there is still considerable room for improvement in creating more robust and reliable AI systems.

As already shown in the literature, the idea of System 1 and System 2 [14], can be helpful to contextualize the current capabilities and limitations of LLMs [15]. In cognitive sciences, System 1 refers to fast, intuitive, and automatic thinking. This type of system can be characterized by fast thinking or quick judgments and decisions that rely on heuristics and subconscious processing. System 1 is highly efficient for everyday tasks that require intuitive, fast, unconscious, and immediate responses. System 2, however, is associated with slow, deliberate, and analytical thinking. It is more helpful and used for complex problem-solving, critical analysis, and tasks that require conscious, sequential, algorithmic planning and reasoning. System 2 is more resource-intensive and slower but more reliable for tasks requiring careful consideration.

In this System 1/System 2 context, we argue that even though RAG approaches have strong potential to contribute to reducing limitations of current LLMs (by playing a role more closely related to System 2), most current RAG approaches only weakly resemble System 2 thinking. The retrieval and generation steps are often designed to be fast and instantaneous, aligning with System 1 thinking, rather than slow and logical as in System 2, which presents challenges on both the data and model sides. For example, research [16, 17] has shown that augmenting LLMs with retrieval without rigorously assessing necessity may adversely impact overall performance.

Therefore, as illustrated in Figure 8, we advocate for a shift from traditional single-model architectures to compound AI systems within a System 2 framework to enhance RAG, particularly in complex enterprise applications. Compound systems enable a collaborative approach to problem-solving by distributing specialized tasks across distinct agents, each optimized for specific functions, improving both retrieval and generation performance in challenging real-world settings.

On the retrieval side, enterprise applications often involve complex, multi-step, and sometimes ambiguous tasks that require deeper reasoning and structured workflows. A compound system can enhance this by assigning specialized agents to handle diverse aspects of data, such as heterogeneous formats (e.g., text, tables, graphs, parametric information) and noisy or incomplete data sources. This allows for agents skilled in reconciliation and semantic querying to refine the retrieval process through iterative, logic-driven interaction, improving both precision and relevance of context.

On the generation side, challenges like hallucination, fact verification, and adherence to context remain key obstacles. In a compound system setup, individual agents can be tasked with verifying facts, maintaining context alignment, and evaluating outputs for accuracy before finalizing responses. This division of labor can be exploited towards reducing hallucinations and enhancing reliability by enabling dynamic inter-agent evaluation, where each agent iteratively cross-checks and validates the others’ outputs [18, 19]. For example, in domain-specific conversational AI, particularly in regulated industries, compound AI systems offer a pathway to safer, more reliable, and robust deployments by integrating domain expertise, context sensitivity, and rigorous validation at each step.

The paper is organized as follows. First, in section 1, we provide a brief overview of traditional RAG models and highlight their limitations, especially in real-world, domain-specific applications. We then

motivate more concretely the need for a System 2 RAG approach, in section 3. In section 4, several approaches to enhance RAG adaptability to System 2 thinking are discussed. Finally, we present our vision for future research in section 5, exploring the potential of compound AI systems as System 2 solutions to RAG.

2 Background

2.1 Traditional Approaches Towards RAG

Retrieval augmented generation (RAG) aims to address hallucinations and factual inaccuracies in LLM-generated content. RAG infuses external knowledge [20, 21], such as knowledge bases and web documents, while prompting LLMs to help generate responses grounded on relevant information. The integration of RAG-based workflows in prompting LLMs has enjoyed widespread adoption, enhancing the suitability of LLMs for real-world applications. Development of a RAG system often begins with an indexing step, which involves cleaning and segmenting the documents — segmentation is required to prepare chunks of information that carry meaningful and strong signals, and in addition, to fit into an LLM’s context window (when using LLMs with limited context windows). Each chunk is then encoded into a vector representation using an embedding model and stored in a vector database. This step is essential for enabling efficient similarity searches in the subsequent retrieval phase. As shown in Figure 8, the following are the standard phases of a traditional RAG workflow:

Retrieval. Given a user query, a RAG system employs an encoding model used during indexing to transform the query into a vector representation and calculates the similarity scores between two vectors: query and candidate text chunks within the indexed corpus. Based on these scores, the system retrieves the top- K chunks with the highest similarity to the query, which are then used as the expanded context for the next stage.

Augmentation. The selected chunks are incorporated into a prompt as expanded context to provide additional relevant information. The goal of such enhancement is to reduce hallucination and improve accuracy of the model’s response. By providing targeted context, the RAG system ensures the model can ground its answer in the most pertinent retrieved data.

Generation. The user query, along with the augmented context of selected documents, is synthesized into a coherent prompt, which is then provided to an LLM that will perform the final generation task. Depending on the task requirements, the model may either draw upon its internal knowledge or limit its response to information in the provided documents.

2.2 Limitations of RAGs

Traditional approaches to RAG do not directly apply to real-world scenarios due to the heterogeneity of data, the complexity of workflows, and the strict constraints on expected task performances.

2.2.1 Lack of Robust Deliberation

Recent studies show how RAGs are not universally effective [22, 23]. Adding noisy or irrelevant passages can override correct LM knowledge, leading to errors (see Table 1). An effective RAG should balance accurate recall with selective retrieval. Identifying when to recall versus retrieve raises key questions: (a) What factors impact an LM’s recall accuracy? (b) What influences RAG performance? (c) What error patterns are common between LM and retriever responses?

Previous research on memorization in LMs and retriever performance has some limitations: (a) it focuses only on entities, while real-world information includes both entities and relations [24, 25]. (b) It

Triple: (Chicago, country, United States of America)	Entity Popularity: 95.0%ile
Question: What country is Chicago located in?	Entity-Relation Popularity: 97.4%ile
LM Answer: United States [Correct]	
Context: The Chicago Municipal Tuberculosis Sanitarium was located in Chicago, Illinois, USA... [Correct Retrieval]	
RALM Answer: USA [Correct]	
Triple: (George H.W. Bush, educated at, Yale University)	Entity Popularity: 89.5%ile
Question: What educational institution did George H.W. Bush attend?	Entity-Relation Popularity: 41.8%ile
LM Answer: Yale University [Correct]	
Context: The George H.W. Bush Presidential Library is located on a site on the west campus of Texas A&M University in College Station, Texas... [Wrong Retrieval]	
RALM Answer: Texas A&M University [Wrong]	
Triple: (Ellen Litman, educated at, University of Pittsburgh)	Entity Popularity: 10.3%ile
Question: What educational institution was Ellen Litman educated at?	Entity-Relation Popularity: 17.9%ile
LM Answer: Stanford University [Wrong]	
Context: Ellen Litman Ellen Litman (born 1973) is an American novelist. She received the Rona Jaffe Foundation Writers' Award in 2006. Born in Moscow, Russia, she emigrated with her parents in 1992 to Pittsburgh, Pennsylvania. She was educated at the University of Pittsburgh and earned a B.S. in Information Science. ... [Correct Retrieval]	
RALM Answer: University of Pittsburgh [Correct]	

Table 1: QA examples from WitQA with predictions of varying popularity of question entity and entity-relation pair. The predictions from LM (GPT-3.5) with no augmentation and RALM (GPT-3.5+BM25) are shown. In the top row, both LM and RALM provide correct answers for the popular question. In the middle row, LM generates correct answer but RALM provides incorrect answer due to retrieval errors. In the bottom row, LM provides incorrect answer for an infrequent entity-relation pair.

examines either retrievers or LM recall independently, overlooking their interplay [26–28]. To address these limitations, in previous work, [16] focused on the QA task and analyzed the performance of 10 LMs across 5 retrieval settings. They introduced WitQA [16], a new dataset of QA pairs generated from Wikipedia triples, selected based on entity and relation popularity, each paired with supporting passages and popularity scores. The investigation of RAGs zero-shot performance on WitQA yields the following key findings (see Figure 9):

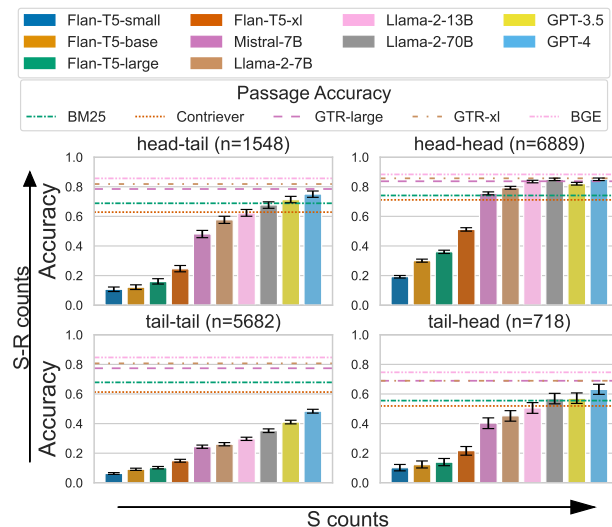


Figure 9: Analysis on Vanilla LMs with BM25, Contriever, GTR, and BGE passage accuracy over S-R counts and S counts (n = the number of questions in the group). In the top row, S-R counts are higher than the median. In the bottom row, they are less than or equal to the median. In the left column, S counts are less than or equal to the median, and in the right column, they are higher than the median.

- LMs can often recall frequently encountered entity-relation pairs from pre-training without retrieval, but this depends on model size; larger models capture more long-tail relations for popular entities, though accuracy drops for less common facts.
- For long-tail entity-relation pairs, retrieval performs better than LM recall, suggesting retrieval augmentation benefits these cases but may introduce override issues with well-known pairs.
- LMs outperform retrievers on well-known entity-relation pairs involving long-tail entities, contrasting prior studies where large LMs struggled with these pairs.

Using these insights, a selective memory integration module was designed, that applies retrieval augmentation or LM recall based on entity-relation popularity. The main idea is closely related to a System 2 approach, in which before trying to answer a question, the system first tries to figure out (reasoning task) whether it should use the retrieval mechanism or not. It was found it could improve QA performance by up to 10.1% [16].

Another limitation of most current RAG models is characterized by their reliance on localized context retrieval, making them less effective for tasks that require *holistic reasoning*—the ability to synthesize, aggregate, and analyze information across multiple documents. For instance, when asked, “Which company employed the most people?” traditional retrieval models may return individual statistics for each company without a comprehensive comparison. This core limitation reveals that while RAG systems are adept at fact retrieval, they falter with broader, cross-document reasoning. Bridging this gap requires models capable of multi-document synthesis, comparative analysis, and extensive dataset integration.

One alternative to address these limitations of RAG models in holistic reasoning is to bypass retrieval altogether and use long-context language models (LCLMs). These models are designed to handle and process significantly larger chunks of information, enabling them to reason effectively over extensive contexts or large sets of documents without the need for iterative retrieval steps. By eliminating the dependency on retrieval mechanisms, LCLMs reduce the risk of retrieving irrelevant or incomplete information, which can compromise reasoning quality. Furthermore, their ability to maintain coherence across lengthy inputs makes them particularly well-suited for tasks requiring nuanced understanding, cross-referencing of details, and synthesis of insights from diverse sources within a single reasoning framework.

To investigate into this, [29] conducted a comparative study of LCLMs and RAG models using HoloBench, a benchmark specifically designed to evaluate holistic reasoning capabilities. It compared two large LCLMs, Llama-3.1-405b and GPT-4o, alongside a smaller LCLM, Llama-3.1-8b. For document retrieval, the work employed `BAAI/bge-large-en-v1.5`, an effective embedding-based model that retrieves the 2k tokens most similar to the query.

The findings in [29] reveal that as context length exceeds 4k tokens, larger vanilla LCLMs consistently outperform RAG-based models, indicating their superior ability to manage longer contexts where RAGs struggle to retrieve relevant information (see Figure 50). Interestingly, with smaller models like Llama-3.1-8b, RAG performs better when context length surpasses 16k tokens. This aligns with previous findings[16] that retrieval models can enhance the performance of weaker models by compensating for their reasoning limitations, even in the presence of retrieval errors. A promising future direction would be to adopt a System 2-based approach and integrate a dynamic mechanism for determining the optimal amount of information to retrieve based on the query and context length, particularly when working with weaker models for holistic reasoning.

2.2.2 Impact of Prompt Sensitivity

A notable limitation of LLMs is their sensitivity to the arrangement of components within prompts, which directly influences their performance in understanding and reasoning on specific tasks. Prior

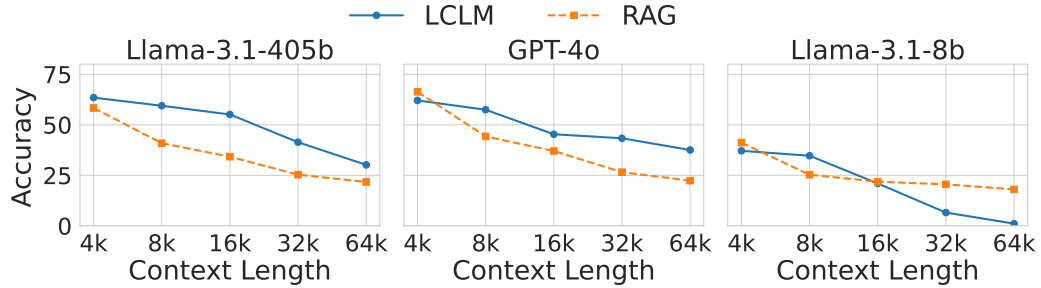


Figure 10: Performance comparison of LCLM and RAG. For long contexts, large models outperform RAG but a retriever helps a small model due to its limited ability to handle long contexts.

research has shown that LLMs are affected by the ordering of few-shot demonstrations [30]. These findings raise an important question: are LLMs similarly affected by the order of elements in prompts across diverse tasks? For instance, in multiple-choice question (MCQ) answering tasks, does the order of answer options impact LLM performance? Figure 11 (extracted from [31]) shows the sensitivity of GPT-4 to options order using a sample from the common sense QA benchmark. Within this context, [31] aims to address the following research questions: (1) To what extent do LLMs exhibit sensitivity to the order of options in multiple-choice questions? (2) What factors contribute to LLMs’ sensitivity to the order of options?

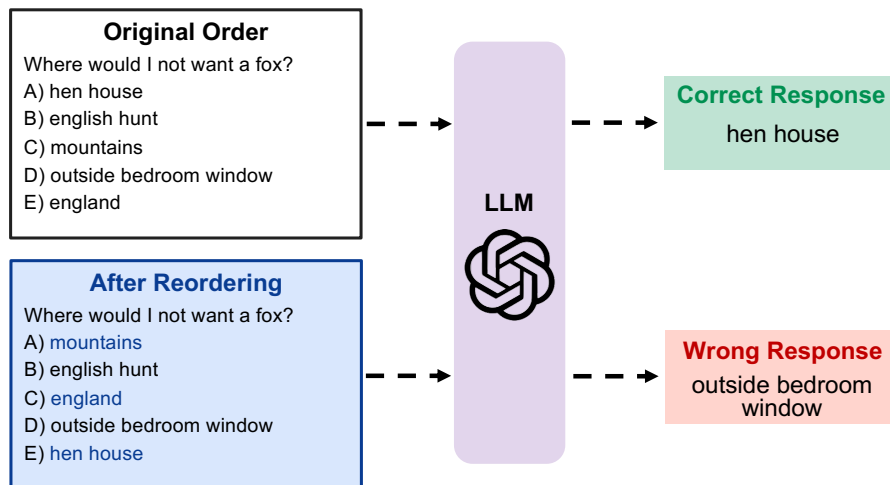


Figure 11: GPT-4 sensitivity to reordering options: Upon changing the order of choices, GPT-4 changes its prediction from “hen house” to “outside of bedroom window” (the example is from the CSQA dataset).

To address the first question, [31] conducted experiments using GPT-4, InstructGPT (text-davinci-003), and Llama-2-13b (chat version) across five multiple-choice question benchmarks. A surprisingly high sensitivity gap of up to 85% in the zero-shot setting (see Table 2) was found. Furthermore, in the few-shot setting, introducing demonstrations to the prompt led only to marginal gains in robustness, if any improvement was observed.

Regarding the second question, it is hypothesized that this sensitivity arises from positional bias, where LLMs display a preference for certain answer placements when uncertain. To investigate, [31] analyzed instances where the models’ predictions shifted upon reordering answer options. Additionally, it was also found that increasing the number of options, while keeping the top possible answers, only

Tasks	GPT-4			InstructGPT			Llama-2-13b		
	Vanila	Min	Max	Vanila	Min	Max	Vanila	Min	Max
CSQA	84.3	-12.6	+10.3	72.3	-24.0	+19.1	62.2	-28.9	+25.5
Logical Deduction	92.3	-8.1	+5.0	64.0	-39.4	+34.7	53.0	-30.7	+34.7
Abstract Algebra	57.0	-30.0	+23.0	33.0	-31.0	+39.0	32.0	-32.0	+53.0
High School Chemistry	71.9	-23.6	+18.2	44.8	-28.5	+38.0	40.6	-32.7	+45.6
Professional Law	66.1	-12.7	+12.1	48.6	-24.9	+25.7	43.8	-32.8	+32.9

Table 2: **Zero-shot order sensitivity**; all three LLMs display a notable level of sensitivity to the order of options across various benchmarks.

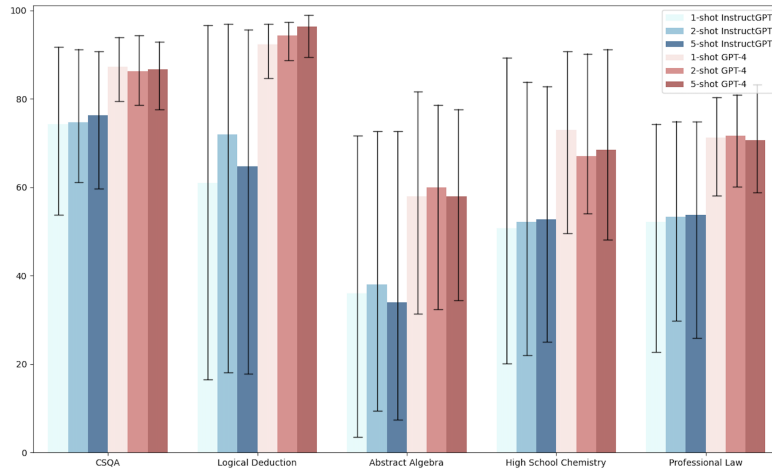


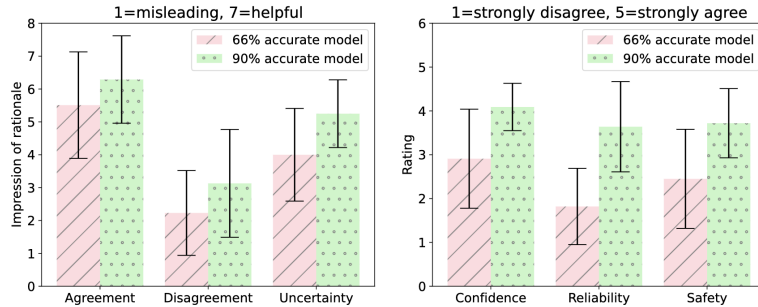
Figure 12: **Order sensitivity in the few-shot setting**: The error bars represent the range of minimum and maximum accuracy achievable in each task through oracle reordering. Our observations are as follows: (1) The sensitivity gap consistently remains substantial in the few-shot setting. (2) As performances improve, the sensitivity gap shrinks. (3) Adding more demonstrations does not necessarily results in a reduction of the gap.

gradually affected performance, suggesting that positional bias rather than option count plays a larger role in LLM sensitivity (see Figure 12).

Another interesting finding from [31] is that, instead of using the original order of the multiple-choice options, one can adopt a "system 2" approach and reason before output the first answer generated by the LLM. In this sense, before deciding on the final answer for the question, the same question can be posed to the LLM multiple times (five times in the referenced study), each time with a randomly shuffled set of choices. Afterward, the answers are aggregated through a reasoning mechanism (in the paper, a very simple majority voting reasoning was employed). This approach has been shown to improve the LLM's overall performance.

2.2.3 Transparency and Accountability in Downstream Applications

To study the implications of trust and accountability of RAG pipelines in downstream applications, [32] considers the task of generating natural language explanations of knowledge-intensive task (KIT) decisions such as multiple-choice question answering. Given the setting for generating corroborating and refutation complete rationales for KIT model decisions, the suitability of retrieval-augmented rationale



(a) Impact on user perception (b) XAI Trust Scale feedback

Figure 13: (a) Irrespective of agreement or disagreement with the KIT model prediction, participants indicated a more negative impression about the rationalization of the lower confidence model prediction. (b) Participant feedback on the trust scale indicates lower confidence for lower accuracy model rationalization.

generation using LLMs is explored. The prompt to LLMs is enriched with relevant knowledge from external sources to condition the rationale generation on facts. Three human subject studies were conducted to evaluate the effectiveness of such rationales in communicating KIT model decisions.

More specifically, two studies were conducted, via crowdsourcing, to evaluate the preferability and acceptability of such rationales to crowd-workers. In another study involving experts — motivated by existing literature on trust in explainable AI [33, 34] — the implications of faithfully rationalizing KIT model decisions irrespective of their correctness was explored. The crowd-sourced studies demonstrate that, more often than not, crowd workers prefer LLM-generated rationales to crowdsourced rationales in existing datasets, citing their factuality, sufficiency, and convincing refutation. Follow-up fine-grained analysis reveals that LLM-generated rationales still have significant room for improvement along dimensions such as insightfulness (*i.e.*, providing new information), redundancy (*i.e.*, avoiding repetitive text), and generalizability (*i.e.*, domain invariance.) The expert-sourced study confirms that faithful rationalization of incorrect model predictions degrades humans’ trust in the generated rationales. The work further explores the utility of instrumenting mechanisms to intervene in the incorrect predictions via a review-then-rationalize pipeline instead of faithfully rationalizing and find that even simple strategies may help intervene up to 71% of the incorrect predictions.

Figure 13a [32] summarizes the participants’ impression of a rationale immediately after viewing the model prediction. When the participants disagreed with the model prediction, they exhibited a stronger negative impression about the rationales for the 66% accuracy condition compared to the 90% accuracy condition. Even when participants agreed with the model prediction, their impression of the rationales remained more negative. The intuition is that the higher disagreement with the model coupled with observing the faithful rationalization of the incorrect prediction negatively impacted participants’ perception of the reliability of the rationales. These observations are confirmed by analyzing the results of the follow-up survey (see Figure 13b.) Unsurprisingly, participants for the 66% accuracy condition rated their confidence in the generated rationales and the reliability of the rationalizer significantly lower compared to the 90% accuracy condition.

3 Towards System 2 RAG in the wild

Towards productizing generative AI, there has been a shift from monolithic models to compound AI systems [35] that incorporate various components other than LLMs for data retrieval, coordination,

and utilizing proprietary models and services. Examples of such systems include Hiring Assistant by LinkedIn [36], AI-BI Genie by Databricks [37], Agentforce by Salesforce [38], and Magentic-One by Microsoft [39], among others. These systems are designed to ensure performance for complex tasks, adaptability to heterogeneous data and use cases, and a higher degree of trust in the production setting.

Motivating example. Consider LinkedIn and Indeed, two global job-matching and hiring platforms in the HR domain. These companies are employing RAG-based workflows for a multitude of tasks in HR, such as matching, recruitment, and career guidance, among others [40, 41]. Given the task of matching job seekers with job postings, a popular use-case of generative AI is communicating meaningful explanations to job seekers about their relevance to the job. To support such a use-case, enterprises use RAG to infuse domain-specific relevant information, which guides the generation of explanations using LLMs [40]. Identifying the relevant information to be provided to the LLM in such a domain-specific context can be challenging due to the heterogeneity and scale of the data. Moreover, within such a production-oriented setting, enterprises have to remain cognizant of real-world constraints such as cost, latency, accuracy, and trustworthiness.

3.1 Data Heterogeneity in the Wild

Within an enterprise setting, different teams in an organization manage project-specific sources of data, collected and often transformed through various workflows over time. Such an observation is derived from the authors’ experience working with enterprise data in the human resources (HR) domain at Megagon Labs. For example, going back to the example of explaining job seeker and job posting match, a team tasked with such work will be interested in unstructured resumes and job descriptions, structured data extracted from the text and their representations, and HR domain-specific knowledge, *i.e.*, relationships among different concepts such as jobs and benefits. Another team working on assisting job-seekers with search (*e.g.*, role and company) will be more interested in rather job market trends, company-specific information and search logs, among others.

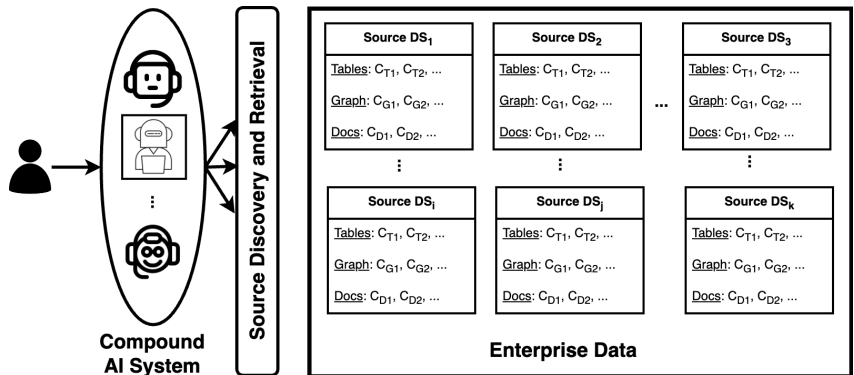


Figure 14: Conceptual data model for RAG in the wild.

We introduce the data model for such a setting in Figure 14: compound AI systems operating over such enterprise data [3]. The data model is multi-granular and multi-modal. Each team-specific data source DS_i represents a collection of sources ($S_m \in DS_i$) corresponding to different modalities (m) — such as multiple tables and documents — created by specific teams. While enterprises may contain data corresponding to other modalities such as audio, video, and image, we limit our scope to documents ($m = D$), tables ($m = T$), and graphs ($m = G$) in this case. Each data source is organized in a hierarchical manner — at the coarse-grained level data is organized in data sources DS_i . Within each data source, data is organized in various collections C_m depending on the modality. Within each

collection, data can be stored and managed by different systems (C_{mi}) depending on the downstream application such as data warehouse, data lake, and lakehouse systems [42, 43]. Therefore, the assumption of traditional RAG where these data sources or collections or databases are known beforehand, breaks down in such a setting. Rather a multi-step approach to data discovery (closer to System 2) is required to identify the relevant coarse- and fine-grained data given a user query.

3.2 Rethinking RAG

To this end, a major consideration in moving a RAG pipeline from System 1 to System 2 thinking involves shifting toward deliberate analytics and explicit reasoning. This transition means that decisions—such as when and where to retrieve information, how much data to retrieve, and how to integrate retrieved information into the generated response—should be made through rigorous reasoning. This reasoning should carefully assess the unique characteristics of the task, data, tools, and available models to ensure outcomes that are both optimal and context-aware. Note that recently released LLM such as o1¹ instruments slow thinking by executing some type of reasoning mechanism before generating the final answer. Even though there has been discussions within certain research circles, which mentions the possibility of having chain of thought and self-reflection empowering o1 models before they generate an answer, it is important to state that no official documentation confirms the use of such techniques in the o1 models) — One consequence of this new reasoning capabilities of these models is the additional time spent thinking, that makes it more effective for complex reasoning tasks, particularly in science and mathematics. However, the additional step leads to higher latency during inference, which may vary depending on the task. In addition to that, o1 is similar to the earlier monolithic LLMs where the parametric knowledge remains abstract, thereby lacking transparency and controllability as no affordances are provided to the users to interact with the decision-making process.

Given a knowledge-intensive task that requires complex reasoning and planning, a systematic approach is necessitated to ensure transparency of the workflow, integrate user guidance at various steps, and enable optimization under real-world constraints such as cost, latency, and accuracy. As illustrated in Figure 8, at the core of the System 2 RAG pipeline lies a planner, which functions as a reasoning module. This planner is grounded in specific tasks and budgets (*e.g.*, time, cost, quality) and operates by foraging and analyzing the properties of data and agents within the registries. Moreover, retrieval is expanded to not only extracting information from documents but also other formats of raw data and data management systems as outlined in Figure 14. Adaptation of retrieval to large-scale heterogenous data necessitates reconciliation and reranking of retrieved information. With the presence of heterogeneous information, the augmented generation requires further scrutinization through fact-checking and verification,

4 Design components

To achieve effective System 2 RAG pipelines, it is essential to address challenges in both data and model aspects. System 2 decision-making relies on an organized, contextually rich data pool for informed outcomes, making efficient identification and organization of relevant information critical. Data enhancements, such as data discovery agents, play a key role by locating, structuring, and tagging pertinent data to create a streamlined and accessible knowledge base. Similarly, selecting and configuring models for each task is crucial. Choosing the right model, tuning it to align with user expectations, and ensuring seamless integration with the data pipeline all contribute to achieving optimal performance. In this section, we provide a high-level overview of our work in tackling these data and model challenges in developing a System 2 RAG pipeline.

¹<https://openai.com/o1/>

4.1 Data-level enhancements

4.1.1 Data Discovery

As an initial effort to understand the potential impact in the data discovery task, we adapted existing datasets and benchmarks in open-domain — from question answering and complex reasoning tasks to natural language querying over structured data — to evaluate coarse- and fine-grained data discovery and task execution performance. Our experiments reveal the impact of data retriever design on downstream task performance — 46% drop in task accuracy on average — across various modalities, data sources, and task difficulty. The results indicate the need to develop optimization strategies to identify appropriate LLM agents and retrievers for efficient execution of CASs over enterprise data. This need is well-aligned with the System 2 type of thinking, in which before performing the retrieval, there is a need for reasoning on what data is available and how it should be retrieved. After such a reasoning step, the retrieval (and the RAG results in general) have the potential to improve.

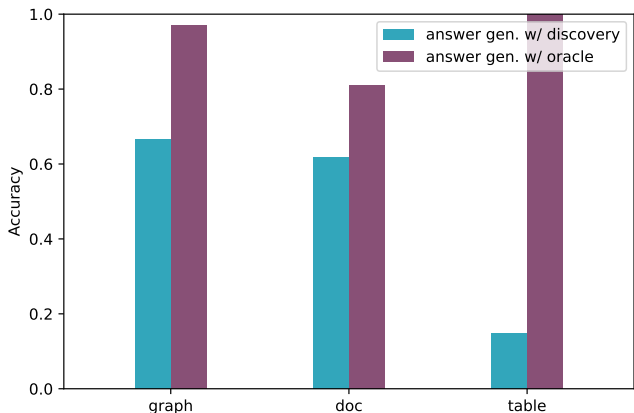


Figure 15: Task execution performance degrades due to poor data discovery despite the purported proficiency of LLMs.

When we explored the impact of data discovery performance on the accuracy of downstream task execution, Figure 15 captures the task execution accuracy for two scenarios: oracle and discovery. Oracle is the case where the ground truth discoverable element is provided to the LLM (*e.g.*, GPT-3.5 turbo) to provide the final answer to a question. Discovery captures the scenario where elements retrieved by the best-performing document, sub-graph, and table discovery models are provided to the LLM. We observe a significant drop in performance from oracle to the discovery scenario, showcasing a 46% decrease in accuracy.

4.1.2 Less is More for Evaluation

Evaluating text generation is crucial for creating high-quality systems. However, aligning automatic evaluation metrics with human judgment remains challenging [44, 45]. While LLMs demonstrate promising correlations with human evaluations, they encounter issues like high costs and the Lost-in-the-Middle problem [46], where key information in the middle of lengthy documents is frequently overlooked in summary evaluations.

To tackle these challenges, [47] introduced a straightforward yet effective approach known as *Extract-then-Evaluate*. At run-time, this method begins by extracting significant sentences from a lengthy source document and concatenating them until the extracted text reaches a predefined length. Subsequently, it assesses the quality of the summary based on this extracted content using LLMs. Figure 16 shows

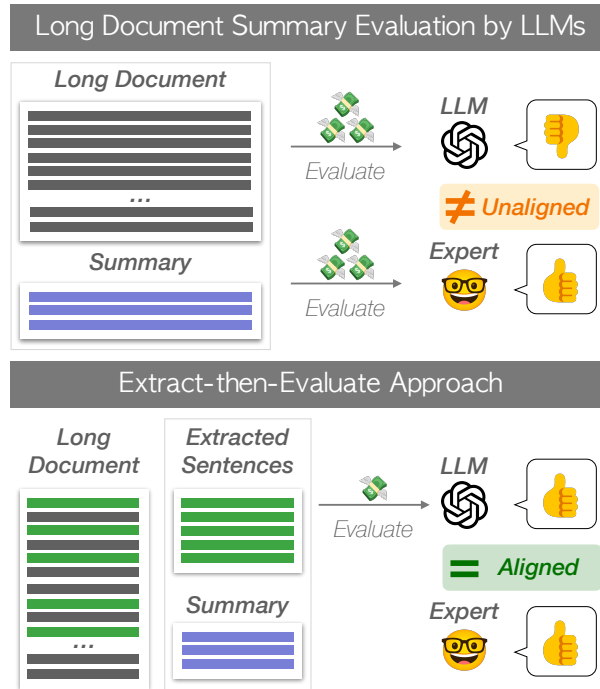


Figure 16: Overview of the long document summary evaluation task by LLMs. Evaluating long document summaries by LLMs is expensive and shows limited alignment with human evaluations. This study demonstrates that extracting important sentences for evaluation in advance not only reduces evaluation costs but also exhibits better alignment with human evaluations.

the overview of our approach. Notice that Extract-then-Evaluate brings a System 2 type of thinking. The evaluation requires the analysis of the original document and the identification of the key elements. Only after reasoning and identifying such key elements is the evaluation of the quality of the summary performed.

The experiments explore various sentence extraction techniques, encompassing both matching-based and model-based methods, such as LEAD, ROUGE, BERTScore, and NLI. Their performance is evaluated across multiple datasets, including arXiv, GovReport, PubMed, and SQuALITY [48, 49]. The main results are shown in Table 3. In the experiments, LLMs demonstrated a notable enhancement in correlation with human judgment when compared to non-LLM baselines. However, this improvement came with increased evaluation costs due to the full document prompt length. Extracting key information before evaluation not only reduced costs but also improved performance, attributed to the Lost-in-the-middle problem, where LLMs struggle with critical information in lengthy documents. This trend showed that LLMs perform better with shorter, more informative documents. Lastly, even within a limited budget, the approach delivered comparable performance to top configurations, achieving similar results to the best extraction method while cutting evaluation costs in half.

Based on these observations, it is possible to conclude that effective data pre-processing can reduce costs while allowing the model to concentrate on key information, ultimately enhancing performance.

4.2 Model-level enhancements

Despite ongoing advancements in LLM and RAG models and their continual scaling, there appears to be no clear limit to their growth potential. As a result, integrated systems and workflows—often

Methods	Consistency						Relevance						Faithfulness					
	arXiv			GovReport			arXiv			GovReport			PubMed			SQuALITY		
	r	ρ	\mathcal{L}	r	ρ	\mathcal{L}	r	ρ	\mathcal{L}	r	ρ	\mathcal{L}	r	ρ	\mathcal{L}	r	ρ	\mathcal{L}
<i>Reference-based metrics</i>																		
ROUGE-1	-0.08	-0.13	-	-0.12	-0.11	-	0.29	0.25	-	0.53	0.52	-	0.32	0.30	-	-0.33	-0.13	-
BERTScore	-0.09	-0.10	-	0.00	-0.04	-	0.22	0.18	-	0.38	0.38	-	0.49	0.49	-	-0.12	0.02	-
BARTScore	0.32	0.36	-	0.51	0.48	-	0.00	0.03	-	0.18	0.24	-	0.49	0.47	-	-0.06	-0.17	-
<i>Reference-free metrics</i>																		
FactCC	0.22	0.19	-	0.28	0.27	-	0.13	0.13	-	0.05	0.04	-	-0.09	-0.14	-	0.13	0.14	-
SummaC	0.32	0.32	-	0.39	0.38	-	0.09	0.08	-	0.05	0.04	-	0.51	0.55	-	0.18	0.24	-
<i>Reference-free metrics with LLM (ours)</i>																		
Full document	0.61	0.46	\$0.15	0.33	0.34	\$0.10	0.58	0.52	\$0.15	0.12	0.11	\$0.10	0.64	0.70	\$0.11	0.51	0.38	\$0.14
Best extraction	0.71	0.50	\$0.05	0.62	0.60	\$0.09	0.63	0.58	\$0.07	0.36	0.40	\$0.07	0.76	0.80	\$0.07	0.85	0.81	\$0.04
Pareto efficient	0.71	0.50	\$0.05	0.60	0.61	\$0.05	0.55	0.48	\$0.04	0.37	0.37	\$0.05	0.75	0.75	\$0.05	0.85	0.81	\$0.04

Table 3: Results for Pearson correlation (r), Spearman correlation (ρ), and the average evaluation cost per instance (\mathcal{L}) indicate that extracting important sentences before evaluation (Best extraction) can yield a higher correlation. Even under a limited budget (Pareto efficient), these results show comparable or even higher correlations compared to the full document setting, with lower costs.

called compound systems or agentic workflows—are emerging. These systems combine LLMs with multiple components, including repeated model calls, retrievers, and external tools, through commercial frameworks like LangChain, LlamaIndex, Auto-GPT, and AgentGPT. Such frameworks empower developers to create agents with unique decision-making capabilities, specialized expertise, and integration with proprietary systems or datasets, as well as build diverse applications ranging from customer service chatbots to advanced decision-support systems. We now highlight several research efforts that showcase compound systems designed to address complex NLP tasks and emphasize that the presence of a System 2 type enhances the performance in such complex tasks.

4.2.1 Multi-conditional ranking

Ranking items based on multiple conditions has wide-ranging applications across various fields. In recommendation systems, for example, once top candidates are shortlisted, re-ranking them according to specific conditions—like genre or category—can greatly enhance the user experience. Similarly, in competitive job markets, this approach is essential for matching resumes to job postings, allowing for prioritization by skills, experience, and other relevant factors. While there has been considerable advancement in ranking extensive document collections given a query [50–52], the nuanced task of ranking a smaller set of items based on multiple conditions has not been addressed in prior research.

To address this gap, [53] defines and investigates the task of multi-conditional ranking (MCR) through the introduction of MCRank, a comprehensive benchmark that encompasses various item types and ranking conditions for evaluating MCR performance. MCRank includes a diverse array of conditions, including positional, locational, temporal, trait-based, and reasoning conditions. Specifically, MCRank was developed by creating a dataset with 18 scenarios varying in item categories, number of conditions (1, 2, or 3), and item set sizes (3, 5, or 7). Each scenario included 200 samples, generated by compiling data and labels for different condition types, featuring randomly ordered item sets with correct rankings. Positional conditions were sourced from Big-Bench’s auto-categorization task and Amazon reviews. For scenarios requiring multiple conditions, additional criteria like character counts or positional conditions were added to simulate realistic complexity. This process ensured a robust dataset for evaluating holistic reasoning in scenarios that simulate situations close to real-cases in which users want to rank items based on different conditions defined by their own needs.

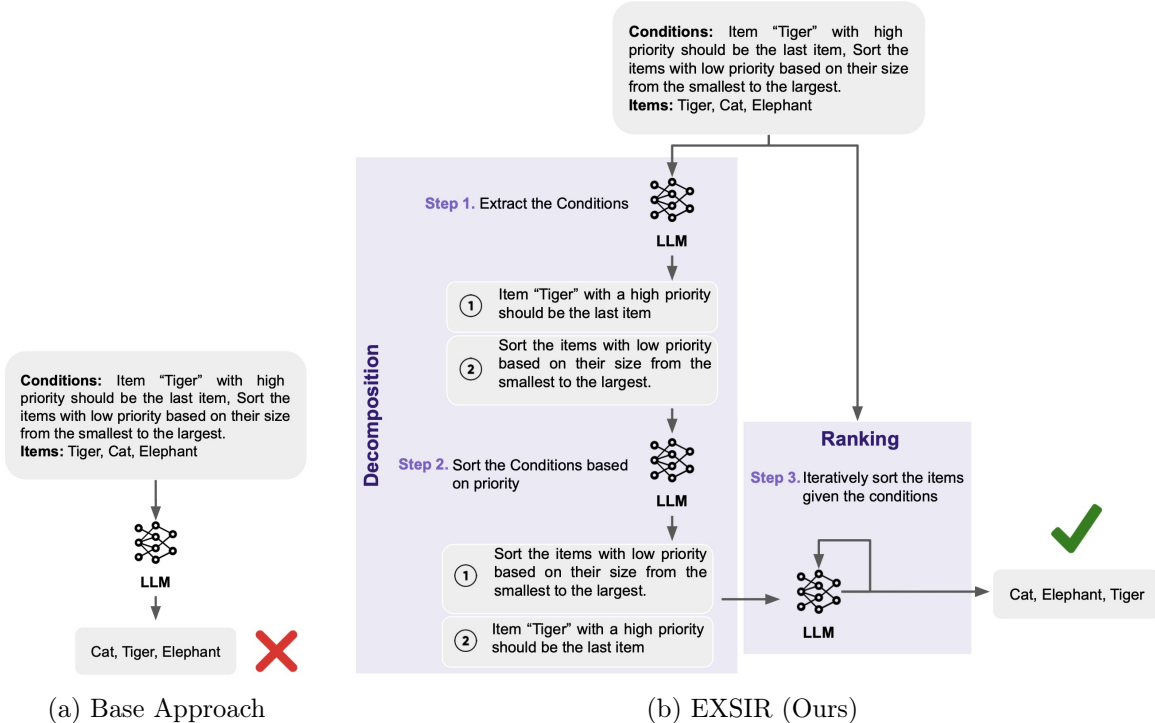


Figure 17: Overview of multi-conditional ranking. Instead of directly prompting LLMs to rank items based on the given conditions, we first extract and sort the conditions based on their priority. Then, we iteratively apply these sorted conditions to the item list.

Furthermore, we develop EXSIR, a novel decomposed reasoning method that iteratively refines rankings. The process begins with extracting individual conditions from a given string and organizing them into a coherent list. A sorting mechanism then arranges these conditions based on their assigned priorities. Finally, the sorted conditions are applied iteratively to the item list, refining the rankings in each cycle based on the current condition. Figure 17 illustrates the workflow of EXSIR along with an example of MCRank.

Initial investigations into existing LLM performance on MCRank show a clear decline in accuracy as both the number of items and conditions increase. Specifically, we observe a sharp drop in ranking accuracy for LLMs like OpenAI o1-mini, GPT-4, ChatGPT (both turbo versions), Llama 3.1-70B, and Mistral (7B) when tasked with three conditions and seven items, with accuracy nearing 0%. EXSIR improves ranking accuracy on MCRank by up to 14.4%, outperforming strong baselines such as Chain-of-Thought (CoT) (see Figure 18). These results demonstrate how the initial steps towards the System 2 type of thinking (present in the EXSIR approach) allowed to improve the performance of even very strong baselines.

4.2.2 Trust but Verify

Motivated by the observations from the study reported in Section 2.2.3, we create a two-stage review-then-rationalize (see Figure 19) pipeline to evaluate the impact of intervening incorrect model predictions before rationalization. The pipeline instruments a `reviewer` module that employs another LLM (GPT-3.5 `text-davinci-003` (`temperature = 0`)) to evaluate the correctness of the knowledge-intensive task (KIT) model and refrain from rationalizing potentially incorrect decisions.

Depending on the task and data domain, the suitability of the reviewer model may vary. Given the

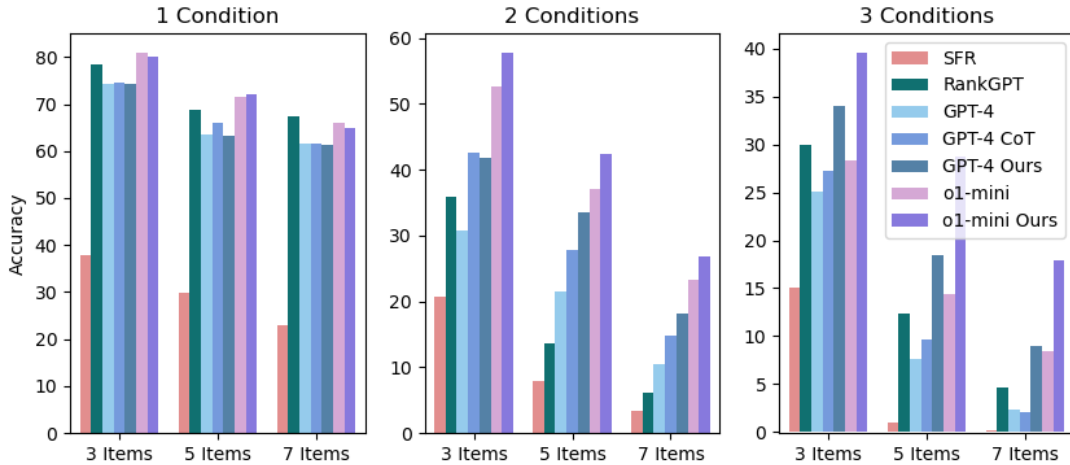


Figure 18: Evaluating the impact of EXSIR against zero-shot CoT prompting for token-level items. We additionally report SFR and RankGPT performances as representatives of existing rankers.

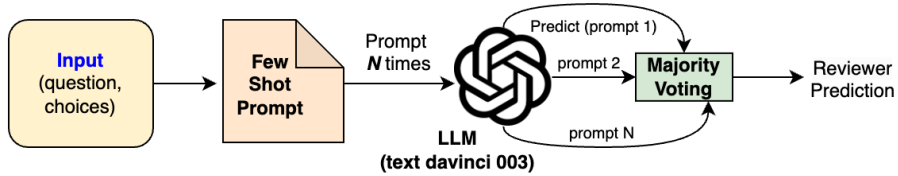


Figure 19: Self-consistency-based Reviewer—intervene for any disagreement with the KIT model prediction.

complexity of knowledge-intensive tasks, we employ a self-consistency-based decoding strategy [54] where the reviewer is asked the same question N ($=5$) times, and the final response is selected via majority voting. The reviewer then compares the model’s prediction with its prediction, and The rationalizer is utilized only when the KIT model and the reviewer agree.

Dataset	Prediction Errors (Test Set)	Errors Intervened	
		Greedy Decoding	Self-consistency
CSQA	321	166 (51.71%)	187 (58.26%)
OBQA	155	102 (65.81%)	110 (70.97%)

Table 4: The review-then-rationalize pipeline helps intervene in incorrect predictions of a knowledge-intensive task (KIT) model. The self-consistency-based reviewer outperforms the greedy decoding-based reviewer.

As shown in Table 4, for knowledge-intensive tasks such as Commonsense QA and Openbook QA, the proposed pipeline helps intervene up to 58% and 71% of the incorrect predictions. Unsurprisingly, the self-consistency-based reviewer outperforms the greedy decoding-based reviewer. Overall, the results draw attention to the importance of responsibly communicating LLM-generated rationales to humans and, consequently, instrumenting guardrails as an effective intervention strategy.

4.3 Orchestration Under Real-World Constraints

In real-world applications, RAG systems must operate under various constraints such as processing time, resource limitations, and compliance requirements. Efficient orchestration of RAG pipelines, involving the coordination of multiple processes (retrieval, generation, and post-processing), can be challenging but offers unique advantages. We propose a blueprint architecture [35] where the key orchestration concept is “streams” to coordinate the flow of data and instructions among components of varying compute requirements.

Key components in the blueprint architecture include (Figure 52): (1) agents, agent and data registries as key touch points and interfaces to seamlessly integrate with existing deployed models, APIs, databases, and services, (2) streams to orchestrate data and instructions across components, and (3) task and data planners to optimize for cost and quality constraints in task execution and data retrieval. It is designed for seamless integration into existing infrastructure, enabling extensibility, customizability, and reusability through well-defined touchpoints and interfaces. It supports externalized orchestration and flexible task coordination via declarative plans, ensuring observability, controllability, and optimized performance while meeting quality-of-service constraints.

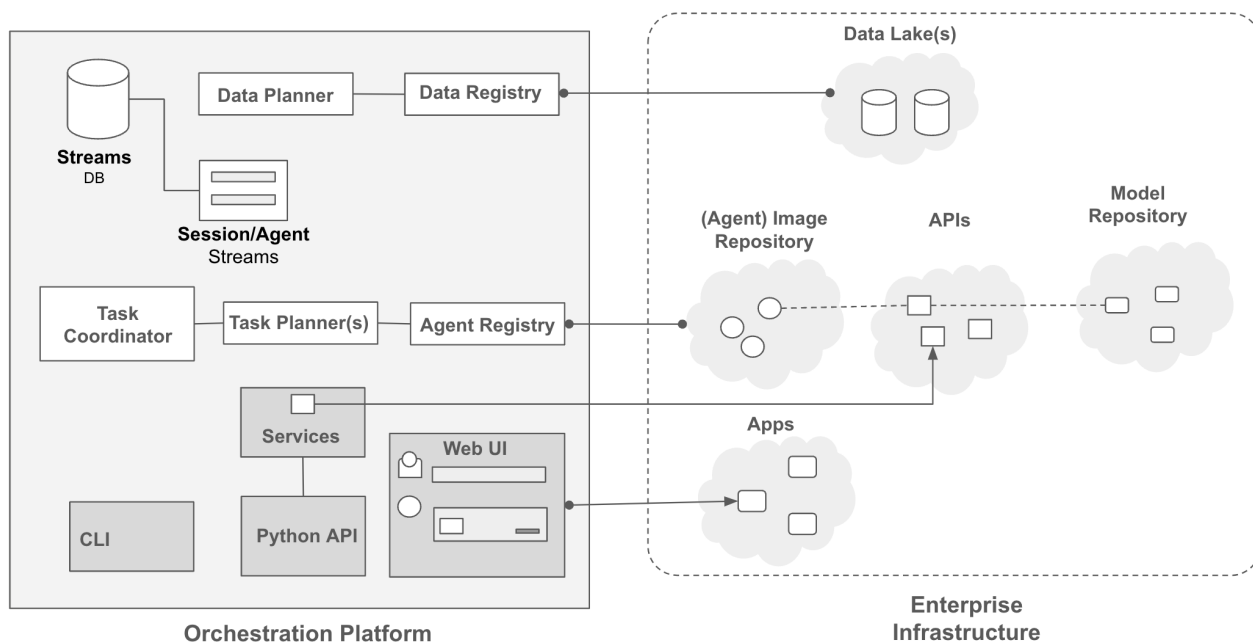


Figure 20: Blueprint Architecture: Data and Agent Registries are touch points that define existing data, models, APIs, and services in the enterprise for utilization by agents.

5 Challenges and Opportunities

The work discussed covers limited aspects of building an effective System 2 RAG solution. Numerous intriguing research questions remain unanswered in the fields of NLP, AI, databases, and HCI, presenting ample opportunities for interdisciplinary collaboration. Here, we will discuss some of these questions.

5.1 Planning and reasoning

Despite emerging attempts to explore LLMs’ reasoning capabilities and use them as planners [55, 56] or ‘routers’ of existing tools and APIs [57–59], LLMs alone still cannot solve the planning problem [60, 61]. Key questions remain, including: How to exploit LLMs for planning, yet add verification and constraints? How to perform planning over multi-modal (relational, graph, documents, parametric) data sources? How to interact with the user in regards to planning, present and refine plans collaboratively? How to learn feedback and attribute back to agents and operators?

In addition, Optimization is critical for planning for production both as a driver of QoS and business-wise, as cost and performance affect the bottom line. Optimization is a well-studied subject, but new questions emerge: How to perform cost estimation for (new) agents, given the dependence on data (size and beyond)? How to handle uncertainty in sources such as LLMs? How to estimate the overall plan cost? How to incorporate an accrued budget into planners?

Additional future research opportunities in reasoning and decision-making systems exist in addressing the overhead introduced by deliberate, system 2 thinking processes, such as planning, which can be slow and cause notable differences in enterprise setups. A promising direction involves alleviating this burden by moving parts of the system 2 thinking process offline, continuously distilling and materializing knowledge into diverse representations. Metaphorically, this is akin to learning to drive: while initial skill acquisition requires deliberate system 2 reasoning, skilled drivers rely on system 1 instincts, reacting fluidly without explicitly thinking about each action, as their expertise becomes ingrained like muscle memory. Similarly, RAG systems can benefit significantly from insights derived from both online and offline learning processes. These insights can extend beyond traditional model weights to include artifacts like graphs, tables, and natural language documents. This calls for research in areas such as knowledge distillation, insight extraction, and planning, with a focus on understanding when and where to trust instinctual, system 1 insights versus when to engage in more rigorous, system 2 reasoning.

5.2 Multi-modal data

Enterprise environments often contain highly varied data sources, including databases, document collections, graphs, and structured tables with heterogeneous schemas. RAG systems designed for these environments must be capable of handling data from diverse formats while preserving contextual coherence across data types. They face challenges ranging from architectural and representational choices to managing ambiguity and uncertainty across modalities. How can RAG models balance capturing detailed, structured knowledge from tables or graphs with synthesizing general information from unstructured text? How should confidence levels and uncertainty be managed when retrieving from different data types? And how should the relevance of retrieved information be measured when dealing with multiple data types, given that existing metrics are often optimized for text-based retrieval?

5.3 From Data to Insights

A core opportunity in RAG systems lies in their ability to transform raw data into actionable insights. This insight-driven retrieval allows systems to dynamically generate responses tailored to user-specific needs or industry contexts. For example, RAG systems in human resources might leverage real-time job market statistics to enhance job recommendations, improving matches based on current industry trends. By deriving insights, systems can synthesize contextual knowledge from data, supporting more accurate and adaptive output generation.

However, research challenges remain. For instance, how can RAG systems accurately capture and prioritize real-time, evolving information from different data sources to ensure that insights remain relevant and current? In dynamic fields such as job searching, the timeliness and accuracy of insights

can be critical. Moreover, what methods can be developed to quantify and communicate the reliability or confidence level of synthesized insights to end users? Trustworthiness becomes especially important when RAG systems support high-stakes decision-making.

6 Conclusion

In this work, we explored the limitations of current Retrieval-Augmented Generation (RAG) models and proposed that a System 2 perspective should be adopted to address the challenges faced by LLMs in complex, domain-specific enterprise applications. Despite the advancements in integrating external information for grounding LLM outputs, we highlighted the shortcomings of existing RAG approaches, which often lack rigorous reasoning and deliberative analytics characteristic of System 2 thinking. Our analysis is based on the literature review and results obtained in previous work on different aspects of LLMs limitations and current RAG approaches, and it reinforces the necessity of transitioning from monolithic LLM architectures to compound AI systems, which employ specialized agents to enhance retrieval, ensure factual correctness, and mitigate issues like hallucination.

Based on the results already obtained by the previously described approaches that incorporate initial steps towards the System 2 type of thinking, we outlined a vision for the future, emphasizing the design of compound systems that better align with System 2 principles, featuring coordinated, logic-driven workflows capable of holistic reasoning and cross-document synthesis. While our work provides a foundational perspective for these advancements, there are still open questions about optimizing retrieval strategies, seamlessly integrating multiple data types, and fine-tuning decision-making modules. Addressing these challenges will be crucial for deploying robust, trustworthy AI systems that meet the high standards of reliability and precision required in enterprise contexts.

Acknowledgment

We thank the Editors of IEEE Data Engineering Bulletin for their valuable feedback: their suggestions immensely improved the quality of the article. Exploring these research problems required a significantly larger cast of characters than the author list in this paper does justice. We thank the co-authors of all the research papers cited in this article.

References

- [1] J. Maynez, S. Narayan, B. Bohnet, and R. McDonald, “On faithfulness and factuality in abstractive summarization,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020, pp. 1906–1919.
- [2] S. J. Semnani, V. Z. Yao, H. C. Zhang, and M. S. Lam, “Wikichat: Stopping the hallucination of large language model chatbots by few-shot grounding on wikipedia,” *arXiv preprint arXiv:2305.14292*, 2023.
- [3] Y. Feng, S. Rahman, A. Feng, V. Chen, and E. Kandogan, “Cmdbench: A benchmark for coarse-to-fine multimodal data discovery in compound ai systems,” in *Proceedings of the Conference on Governance, Understanding and Integration of Data for Effective and Responsible AI*, 2024, pp. 16–25.
- [4] M. Sclar, Y. Choi, Y. Tsvetkov, and A. Suhr, “Quantifying language models’ sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting,”

- in The Twelfth International Conference on Learning Representations, 2024. [Online]. Available: <https://openreview.net/forum?id=RIu5lyNXjT>
- [5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., “Language models are few-shot learners,” Advances in neural information processing systems, vol. 33, pp. 1877–1901, 2020.
 - [6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou et al., “Chain-of-thought prompting elicits reasoning in large language models,” Advances in neural information processing systems, vol. 35, pp. 24 824–24 837, 2022.
 - [7] P. Sahoo, A. K. Singh, S. Saha, V. Jain, S. Mondal, and A. Chadha, “A systematic survey of prompt engineering in large language models: Techniques and applications,” arXiv preprint arXiv:2402.07927, 2024.
 - [8] N. Hounsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, “Parameter-efficient transfer learning for nlp,” in International Conference on Machine Learning, 2019, pp. 2790–2799.
 - [9] Z. Qin et al., “Towards better parameter-efficient fine-tuning for large language models: A position paper,” arXiv preprint arXiv:2311.13126, 2023.
 - [10] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray et al., “Training language models to follow instructions with human feedback,” Advances in neural information processing systems, vol. 35, pp. 27 730–27 744, 2022.
 - [11] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, “FEVER: a large-scale dataset for fact extraction and VERification,” in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), M. Walker, H. Ji, and A. Stent, Eds. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 809–819. [Online]. Available: <https://aclanthology.org/N18-1074>
 - [12] R. Aly, Z. Guo, M. Schlichtkrull, J. Thorne, A. Vlachos, C. Christodoulopoulos, O. Cocarascu, and A. Mittal, “Feverous: Fact extraction and verification over unstructured and structured information,” arXiv preprint arXiv:2106.05707, 2021.
 - [13] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel et al., “Retrieval-augmented generation for knowledge-intensive nlp tasks,” Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020.
 - [14] D. Kahneman, Thinking, fast and slow. London: Penguin, 2012.
 - [15] Y. Bengio et al., “From system 1 deep learning to system 2 deep learning,” in Posner lecture at NeurIPS’2019, Neural Information Processing Systems, Vancouver, BC, 2019.
 - [16] S. Maekawa, H. Iso, S. Gurajada, and N. Bhutani, “Retrieval helps or hurts? a deeper dive into the efficacy of retrieval augmentation to language models,” in Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), K. Duh, H. Gomez, and S. Bethard, Eds., 2024, pp. 5506–5521.

- [17] H. Ding, L. Pang, Z. Wei, H. Shen, and X. Cheng, “Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models,” arXiv preprint arXiv:2402.10612, 2024.
- [18] Y. Zhang and Q. Yang, “A survey on multi-task learning,” IEEE transactions on knowledge and data engineering, vol. 34, no. 12, pp. 5586–5609, 2021.
- [19] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. Hruschka, and T. Mitchell, “Toward an architecture for never-ending language learning,” in Proceedings of the AAAI conference on artificial intelligence, vol. 24, no. 1, 2010, pp. 1306–1313.
- [20] B. Peng, M. Galley, P. He, H. Cheng, Y. Xie, Y. Hu, Q. Huang, L. Liden, Z. Yu, W. Chen et al., “Check your facts and try again: Improving large language models with external knowledge and automated feedback,” arXiv preprint arXiv:2302.12813, 2023.
- [21] A. Lazaridou, E. Gribovskaya, W. Stokowiec, and N. Grigorev, “Internet-augmented language models through few-shot prompting for open-domain question answering,” arXiv preprint arXiv:2203.05115, 2022.
- [22] F. Petroni, P. S. H. Lewis, A. Piktus, T. Rocktäschel, Y. Wu, A. H. Miller, and S. Riedel, “How context affects language models’ factual predictions,” in Conference on Automated Knowledge Base Construction, (AKBC), 2020.
- [23] D. Li, A. S. Rawat, M. Zaheer, X. Wang, M. Lukasik, A. Veit, F. Yu, and S. Kumar, “Large language models with controllable working memory,” in Findings of the Association for Computational Linguistics (ACL), 2023, pp. 1774–1793.
- [24] K. Sun, Y. Xu, H. Zha, Y. Liu, and X. L. Dong, “Head-to-tail: How knowledgeable are large language models (LLMs)? A.K.A. will LLMs replace knowledge graphs?” in Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL), 2024.
- [25] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, and H. Hajishirzi, “When not to trust language models: Investigating effectiveness of parametric and non-parametric memories,” in Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL), 2023.
- [26] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu, and A. Miller, “Language models as knowledge bases?” in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., 2019, pp. 2463–2473.
- [27] C. Sciavolino, Z. Zhong, J. Lee, and D. Chen, “Simple entity-centric questions challenge dense retrievers,” in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds., 2021, pp. 6138–6148.
- [28] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing,” ACM Computing Surveys, vol. 55, no. 9, pp. 1–35, 2023.
- [29] S. Maekawa, H. Iso, and N. Bhutani, “Holistic reasoning with long-context lms: A benchmark for database operations on massive textual data,” arXiv preprint arXiv:2410.11996, 2024.

- [30] Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, “Calibrate before use: Improving few-shot performance of language models,” in International Conference on Machine Learning. PMLR, 2021, pp. 12 697–12 706.
- [31] P. Pezeshkpour and E. Hruschka, “Large language models sensitivity to the order of options in multiple-choice questions,” in Findings of the Association for Computational Linguistics: NAACL 2024, 2024, pp. 2006–2017.
- [32] A. Mishra, S. Rahman, H. Kim, K. Mitra, and E. Hruschka, “Characterizing large language models as rationalizers of knowledge-intensive tasks,” Findings of the Association for Computational Linguistics ACL 2024, 2024.
- [33] R. R. Hoffman, S. T. Mueller, G. Klein, and J. Litman, “Metrics for explainable ai: Challenges and prospects,” arXiv preprint arXiv:1812.04608, 2018.
- [34] M. C. Stites, M. Nyre-Yu, B. Moss, C. Smutz, and M. R. Smith, “Sage advice? the impacts of explanations for machine learning models on human decision-making in spam detection,” in International Conference on Human-Computer Interaction. Springer, 2021, pp. 269–284.
- [35] E. Kandogan, S. Rahman, N. Bhutani, D. Zhang, R. Li Chen, K. Mitra, S. Gurajada, P. Pezeshkpour, H. Iso, Y. Feng et al., “A blueprint architecture of compound ai systems for enterprise,” arXiv e-prints, pp. arXiv–2406, 2024.
- [36] LinkedIn. Introducing hiring assistant for recruiter & jobs. [Online]. Available: <https://business.linkedin.com/talent-solutions/hiring-assistant>
- [37] Databricks. Musings on building a generative ai product. [Online]. Available: <https://www.microsoft.com/en-us/research/articles/magentic-one-a-generalist-multi-agent-system-for-solving-complex-tasks/>
- [38] Salesforce. Agentforce: Build the future with ai agents. [Online]. Available: <https://www.salesforce.com/agentforce/>
- [39] Microsoft. Magentic-one: A generalist multi-agent system for solving complex tasks. [Online]. Available: <https://www.microsoft.com/en-us/research/articles/magentic-one-a-generalist-multi-agent-system-for-solving-complex-tasks/>
- [40] Indeed. Indeed uses openai to deliver contextual job matching to millions of job seekers. [Online]. Available: <https://openai.com/index/indeed/>
- [41] LinkedIn. Musings on building a generative ai product. [Online]. Available: <https://www.linkedin.com/blog/engineering/generative-ai/musings-on-building-a-generative-ai-product>
- [42] A. M. Nambiar and D. Mundra, “An overview of data warehouse and data lake in modern enterprise data management,” Big Data Cogn. Comput., vol. 6, no. 4, p. 132, 2022. [Online]. Available: <https://doi.org/10.3390/bdcc6040132>
- [43] M. Armbrust, A. Ghodsi, R. Xin, and M. Zaharia, “Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics,” in Proceedings of CIDR, vol. 8, 2021, p. 28.
- [44] M. Bhandari, P. N. Gour, A. Ashfaq, P. Liu, and G. Neubig, “Re-evaluating evaluation in text summarization,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, Nov. 2020, pp. 9347–9359.

- [45] A. R. Fabbri, W. Kryściński, B. McCann, C. Xiong, R. Socher, and D. Radev, “SummEval: Re-evaluating summarization evaluation,” Transactions of the Association for Computational Linguistics (TACL), vol. 9, pp. 391–409, 2021.
- [46] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang, “Lost in the middle: How language models use long contexts,” arXiv preprint arXiv:2307.03172, 2023. [Online]. Available: <https://arxiv.org/abs/2307.03172>
- [47] Y. Wu, H. Iso, P. Pezeshkpour, N. Bhutani, and E. Hruschka, “Less is more for long document summary evaluation by llms,” in Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers), 2024, pp. 330–343.
- [48] H. Y. Koh, J. Ju, H. Zhang, M. Liu, and S. Pan, “How far are we from robust long abstractive summarization?” in Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2022, pp. 2682–2698.
- [49] K. Krishna, E. Bransom, B. Kuehl, M. Iyyer, P. Dasigi, A. Cohan, and K. Lo, “LongEval: Guidelines for human evaluation of faithfulness in long-form summarization,” in Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, 2023, pp. 1650–1669.
- [50] O. Khattab and M. Zaharia, “Colbert: Efficient and effective passage search via contextualized late interaction over bert,” in Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020, pp. 39–48.
- [51] S. Zhuang, H. Zhuang, B. Koopman, and G. Zuccon, “A setwise approach for effective and highly efficient zero-shot ranking with large language models,” arXiv preprint arXiv:2310.09497, 2023.
- [52] Z. Qin, R. Jagerman, K. Hui, H. Zhuang, J. Wu, J. Shen, T. Liu, J. Liu, D. Metzler, X. Wang et al., “Large language models are effective text rankers with pairwise ranking prompting,” arXiv preprint arXiv:2306.17563, 2023.
- [53] P. Pezeshkpour and E. Hruschka, “Multi-conditional ranking with large language models,” arXiv preprint arXiv:2404.00211, 2024.
- [54] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, S. Narang, A. Chowdhery, and D. Zhou, “Self-consistency improves chain of thought reasoning in language models,” in The Eleventh International Conference on Learning Representations.
- [55] Z. Zhao, W. S. Lee, and D. Hsu, “Large language models as commonsense knowledge for large-scale task planning,” Advances in Neural Information Processing Systems, vol. 36, 2024.
- [56] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, “Language models as zero-shot planners: Extracting actionable knowledge for embodied agents,” in International conference on machine learning. PMLR, 2022, pp. 9118–9147.
- [57] Y. Liang, C. Wu, T. Song, W. Wu, Y. Xia, Y. Liu, Y. Ou, S. Lu, L. Ji, S. Mao et al., “Taskmatrix.ai: Completing tasks by connecting foundation models with millions of apis,” Intelligent Computing, vol. 3, p. 0063, 2024.
- [58] Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian et al., “Toollm: Facilitating large language models to master 16000+ real-world apis,” arXiv preprint arXiv:2307.16789, 2023.

- [59] S. Kim, S. Moon, R. Tabrizi, N. Lee, M. Mahoney, K. Keutzer, and A. Gholami, “An llm compiler for parallel function calling,” arXiv, 2023.
- [60] S. Kambhampati, K. Valmeekam, L. Guan, K. Stechly, M. Verma, S. Bhambri, L. Saldyt, and A. Murthy, “Llms can’t plan, but can help planning in llm-modulo frameworks,” arXiv preprint arXiv:2402.01817, 2024.
- [61] K. Valmeekam, M. Marquez, S. Sreedharan, and S. Kambhampati, “On the planning abilities of large language models-a critical investigation,” Advances in Neural Information Processing Systems, vol. 36, pp. 75 993–76 005, 2023.

RAG-based Question Answering over Heterogeneous Data and Text

Philipp Christmann, Gerhard Weikum

Max Planck Institute for Informatics
Saarland Informatics Campus, Germany
{pchristm, weikum}@mpi-inf.mpg.de

Abstract

This article presents the QUASAR system for question answering over unstructured text, structured tables, and knowledge graphs, with unified treatment of all sources. The system adopts a RAG-based architecture, with a pipeline of evidence retrieval followed by answer generation, with the latter powered by a moderate-sized language model. Additionally and uniquely, QUASAR has components for question understanding, to derive crisper input for evidence retrieval, and for re-ranking and filtering the retrieved evidence before feeding the most informative pieces into the answer generation. Experiments with three different benchmarks demonstrate the high answering quality of our approach, being on par with or better than large GPT models, while keeping the computational cost and energy consumption orders of magnitude lower.

1 Introduction

Motivation and Problem. The task of question answering, QA for short, arises in many flavors: factual vs. opinions, simple lookups vs. multi-hop inference, single answer vs. list of entities, direct answers vs. long-form, one-shot questions vs. conversations, and other varieties (see, e.g., surveys [28, 29]). The state-of-the-art for this entire spectrum has been greatly advanced in the past decade. Most notably, incorporating deep learning into retriever-reader architectures (e.g., [2, 13, 18]) has boosted answering quality, and most recently, large language models (LLM) [25, 40] have pushed the envelope even further (e.g., [16]).

Today’s LLMs alone are capable of accurately answering many *factoid* questions, simply from their pre-trained parametric memory which latently encodes huge text corpora and other online contents. However, this critically depends on the frequency of evidence in the underlying contents and the complexity of the information need. For example, asking for the MVP of the 2024 NBA season would easily return the correct answer Nikola Jokic, but asking for the highest-scoring German NBA player or the MVP of the 2024 German basketball league pose a big challenge. The reason is that LLMs alone do not easily recall information about not so popular or even long-tail entities [17, 32], and that they are mainly geared for direct look-ups as opposed to connecting multiple pieces of evidence [24, 39].

[10, 12, 21, 41] known as RAG, address these bottlenecks. In addition to cleverly crafted prompts and few-shot examples, the LLM is provided with the top-ranked results of an explicit retrieval step, like web search or knowledge graph (KG) lookups. The former is often necessary for freshness of answers, and the latter may help with long-tail entities and also mitigate the notorious risk of hallucinations. Still, this

generation’s RAG architectures are limited in how broad and how deep they tap into external sources. Popular AI assistants like Gemini or ChatGPT seem to primarily retrieve from the text of web pages (incl. Wikipedia articles), and academic research has additionally pursued knowledge augmentation by enhancing prompts with facts from large KGs (e.g., Wikidata).

An additional content modality that is still underexplored are online tables: a wide range of tabular data including HTML tables in web pages, spreadsheets and statistics, all the way to CSV and JSON datasets that are abundant on the Internet. There is prior work on joint support for text and KGs and for text and tables, but very little on all of these together – some notable exceptions being [3, 5, 27, 38]. **Examples.** All three heterogeneous types of sources are crucial not only for answering different questions from different kinds of evidence, but also for combining multiple pieces of evidence of different modalities to infer correct and complete answers. To illustrate the need for tapping all sources, consider the following questions:

- Q1: Which Chinese basketballers have played in the NBA?*
- Q2: Who was the first Chinese NBA player?*
- Q3: Which Chinese NBA player has the most matches?*

Q1 can be cast into querying a KG, but the list there is not necessarily complete and up-to-date, so additional evidence from text or tables would be desired. Q2 needs information about who played in which seasons, found only in web pages or sports-statistics tables. Finally, Q3 may be lucky in finding prominent textual evidence (e.g., in biographies, Wikipedia etc.), but this often faces divergent statements, and resolving contradictions needs to dive into more evidence. Besides, when textual evidence is rare and hard to find or not trustworthy enough, then information from multiple tables and text snippets may have to be aggregated (e.g., totals of per-season counts). Some of this may perhaps become feasible for an industrial LLM’s RAG capabilities in the near future, but there are always harder scenarios by moving from Chinese NBA players deeper into the long tail, such as asking for Lithuanian players in the German handball league.

Approach and Contribution. This paper presents a simple but powerful and versatile RAG system with unified access to text, KG and tables. We call our method QUASAR (for Question Answering over Heterogeneous Sources with Augmented Retrieval). Its architecture is relatively straightforward: all heterogeneous content is verbalized and indexed for retrieval; a retriever finds top-ranked results for the given question (from different source types), and these are fed into the LLM for answer generation. This is the unsurprising bird-eye’s view. Specific details that are key factors for the strong performance of QUASAR are:

- i) automatically casting user questions into a structured representation of the information need, which is then used to guide
- ii) judicious ranking of search results, with multiple rounds of re-ranking and pruning, followed by
- iii) extracting faithful answers from an LLM in RAG mode, with answers grounded in tangible evidence.

The paper presents experiments with three different benchmarks, covering various flavors of questions. We focus on one-shot questions; conversational QA is out of scope here, but QUASAR itself is well applicable to this case, too. Our experiments demonstrate that our methods are competitive, on par with big GPT models and often better, while being several orders of magnitude lower in computational and energy cost. The experimental findings also highlight that question understanding, with structured representation of user intents, and iterative re-ranking of evidence are crucial for good performance.

Overall, our contribution lies in proposing a unified system architecture for RAG-based question answering over a suite of different data sources, with strong points regarding both effectiveness (i.e., answer quality) and efficiency (i.e., computational cost).

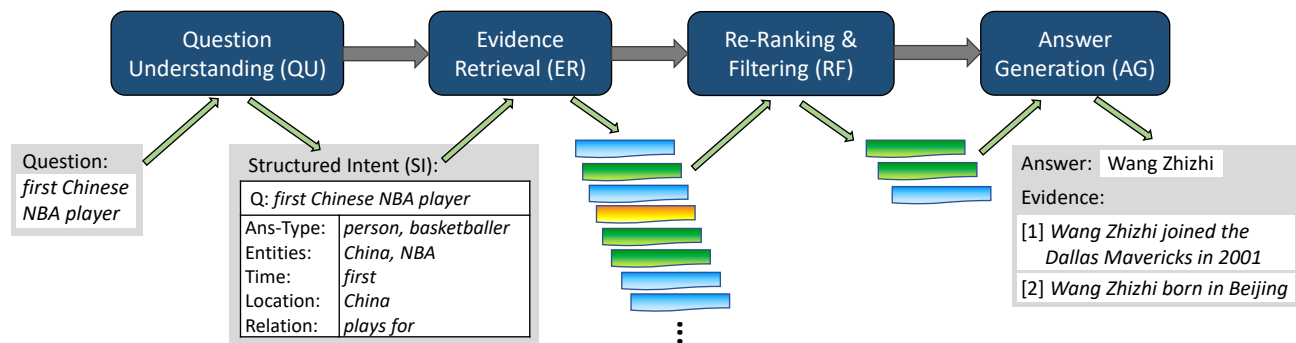


Figure 21: Overview of the QUASAR system.

2 Related Work

The RAG paradigm came up as a principled way of enhancing LLM factuality incl. provenance and mitigating the risk of hallucination [12, 21]. It is highly related to the earlier retriever-reader architectures for QA [2, 18], especially when the reader uses the fusion-in-decoder method [13, 27]. Since its invention, RAG methodology has been greatly advanced, introducing a wide suite of extensions, such as batched inputs, interleaving retrieval and generation steps, and more (see the recent surveys [10, 41]).

On question answering (QA), there is a vast amount of literature including a wealth of differently flavored benchmarks (see, e.g., [28]). The case of interest here is QA over heterogeneous sources, tapping into both unstructured content and structured data. A variety of works has pursued this theme by combining knowledge graphs with text sources, using graph-based methods, neural learning and language models (e.g., [30, 31, 37]).

Most relevant for this article is the research on jointly leveraging all different sources: text, KGs, and tables (incl. CSV and JSON files). This includes the UNIK-QA system [27], the SPAGHETTI/SUQL project [23, 38], the MATTER method [19], the STaRK benchmarking [34], and our own prior work [3, 5] (without claiming exhaustiveness). Out of these, we include UNIK-QA, SPAGHETTI and our own systems CONVINSE and EXPLAINNN as baselines in the experimental evaluation. Their architectures are similar to ours, but UNIK-QA and SPAGHETTI do not have our distinctive elements of question understanding and iterative re-ranking (originally introduced in EXPLAINNN [5]).

3 Methodology

The QUASAR system is a pipeline of four major stages, as illustrated in Figure 21. First, the input question is analyzed and decomposed, in order to compute a structured intent (SI) representation that will pass on to the subsequent steps, along with the original question. Second, the SI is utilized to retrieve pieces of evidence from different sources: text, KG and tables. Third, this pool of potentially useful evidence is filtered down, with iterative re-ranking, to arrive at a tractably small set of most promising evidence. The final stage generates the answer from this evidence, passing back the answer as well as evidence snippets for user-comprehensible explanation.

The second and fourth stage, Evidence Retrieval (ER) and Answer Generation (AG), are fairly standard. Such a two-phase architecture was called a retriever-reader architecture [42]. With a modern LLM replacing the earlier kinds of neural readers, this is the core of every RAG system [10].

Stages 1 and 3 are unique elements of our architecture, judiciously introduced to improve both effectiveness (i.e., answer quality) and efficiency (i.e., computational cost). Question Understanding (QU) provides the ER component with crisper and semantically refined input, and the Re-Ranking &

Filtering (RF) stage is beneficial for distilling the best evidence from the large pool of retrieved pieces. The following subsections elaborate on the four stages of the pipeline, emphasizing the QUASAR-specific steps QU and RF.

3.1 Question Understanding (QU)

To prepare the retrieval from different kinds of sources, including a KG, ad-hoc tables and text documents, it is useful to analyze and decompose the user question. In this work, we aim to cast a question into a structured intent (SI) representation: essentially a frame with faceted cues as slots, or equivalently, a concise set of key-value pairs. Figure 21 gives an idealized example for the question about the first Chinese NBA player. The facets or keys of potential interest here are:

- Ans-Type: the expected answer type (or types when considering different levels of semantic refinement),
- Entities: the salient entities in the question, and
- Relation: phrases that indicate which relation (between Q and A entities) the user is interested in.

In addition, as questions can have temporal or spatial aspects, the SI also foresees slots for:

- Time: cues about answer-relevant time points or spans, including relative cues (e.g., “before Covid”) and ordinal cues (e.g., “first”), and
- Location: cues about answer-relevant geo-locations.

The ideal SI for example question Q2 would look like:

Ans-Type: person, basketballer; Entities: China, NBA; Time: first; Location: China;
Relation: plays for.

Note that the values for these slots can be crisp like entity names or dates, but they can also take the form of surface phrases. The SI purpose and value lie in the decomposition. In practice, many questions would only lead to a subset of faceted cues, leaving some slots empty. For the example in Figure 21, an alternative SI could simply consist of

Ans-Type: person; Entities: China, NBA; Time: first.

Even this simplified SI can be highly beneficial in guiding the subsequent evidence retrieval.

To generate the SI from a user question, we employ a (small-scale) LM, specifically BART [20], a Transformer-based auto-encoder with 140M parameters.¹ BART is pre-trained for language representation; its power for our purpose comes from fine-tuning. To this end, we generate (question, SI) pairs by using an instruction-trained LLM like GPT-4, with few-shot in-context learning (following our earlier work [15]). Note that this is a one-time action; at inference-time we only use much smaller LMs. The generated silver-standard pairs are then used to fine-tune BART. In the experiments in this article, we leverage pre-existing collections of silver pairs, based on the training data of the CompMix benchmark [6], comprising 3,400 such pairs.

Although this paper focuses on single-shot questions, the QUASAR architecture is also geared for conversational QA. In that setting, the SI can play an even bigger role, as (follow-up) questions are often formulated in a rather sloppy manner – all but self-contained. For example, a conversation could start with a clear question When did Wang Zhizhi join the NBA?, followed a few dialog steps later, by a user utterance like Which teams did he play for? or simply Which teams?. In such an informal conversation,

¹<https://huggingface.co/facebook/bart-base>

the system needs to contextualize each user utterance based on the preceding turns in the dialog (e.g., inferring the relevant entities Wang Zhizhi and NBA from the conversational history). For details on conversational QA, based on our architecture, see our earlier works [3, 5].

3.2 Evidence Retrieval (ER)

The ER stage taps into a knowledge graph, a corpus of text documents, and a collection of web tables. Specifically, for the experiments, we use the Wikidata KG, all English Wikipedia articles, and all tables that are embedded in Wikipedia pages (incl. infoboxes, which can be seen as a special case of tables).

Retrieval from KG: To retrieve evidence from the KG, we utilize our earlier work CLOCQ [4], which provides entity disambiguations and a relevant KG-subgraph for a given query. Unlike most other works on QA-over-KG, CLOCQ fetches all KG-facts that are relevant for a given entity in a single step. For example, when querying for NBA players, it can traverse the KG neighborhood and pick up top teams, also considering so-called qualifier nodes in Wikidata which are often used for temporal scopes. As the disambiguation of entity names onto the KG can be tricky and noisy (e.g., China could be mapped to Chinese sports teams in all kinds of sports), CLOCQ considers several possible disambiguations [4] (typically in the order of 10 result entities). The queries for CLOCQ are constructed by concatenating all slots of the question’s SI. For the example query about the first Chinese NBA player, good result entities would be Dallas Mavericks, lists about NBA seasons, MVP awards etc., and their associated facts. These provide cues, but are likely insufficient to answer the question.

Retrieval from Text and Tables: The disambiguated entities returned by CLOCQ form anchors for tapping into text and tables. QUASAR first identifies relevant text documents and tables that refer to the anchor entities. With focus on Wikipedia, these are simply the articles for the respective entities. QUASAR then constructs a keyword query that concatenates all available fields of the SI. The query is evaluated against a linearized and verbalized representation (see below) of all sentences and all table rows in the selected documents. This returns a set of sentences and individual table rows, ranked by BM25 scores.

Evidence Verbalization: All results from the different data sources are uniformly treated by linearizing and verbalizing them into token sequences. For KG results, the entity-centric triple sets are linearized via breadth-first traversal of the mini-graph starting from the entity node. For tables, results are individual rows, which are contextualized by including labels from column headers and from the DOM-tree path of the article where the table comes from. For example, a table row about Wang Zhizhi playing for Dallas (Mavericks) in the 2000-2001 season, would be expressed as:

Wang Zhizhi / NBA Career / Season: 2000-2001, Team: Dallas, Games Played: 5 . . .

Finally, results from the text corpus are already in the form of token sequences, but we can additionally prefix these with the DOM-tree labels. We can think of this entire pool of evidence as an on-the-fly corpus of potentially relevant pseudo-sentences, forming the input of the subsequent RF stage.

Result Ranking: Overall, the ER stage compiles a substantial set of evidence, possibly many thousands of entities, text snippets and table rows. Therefore, we practically restrict the pool to a subset of high-scoring pieces, like the top-1000. For scoring, a simple BM25 model (a classical IR method) is applied. By default, we treat all evidence pieces uniformly with global scoring, no matter whether they come from KG, text or tables.

3.3 Re-Ranking and Filtering (RF)

With a pool of top-1000 evidence pieces, we could invoke an LLM for answer generation. However, that would face a large fraction of noise (i.e., misleading evidence) and incur high costs of computation and energy consumption.

For both of these reasons, we have devised light-weight techniques for iteratively reducing the top-1000 pieces to a small subset, say top-30 or top-10, that can be fed into an LLM at much lower cost (as LLM computations and pricing are at least linear in the number of input tokens). The difficulty is, of course, to do this without losing good evidence and reducing answer presence. Our techniques for this task are based on graph neural networks (GNNs) [35] or cross-encoders (CEs) [7, 22].

GNN-based RF. Given a large pool of evidence pieces from all sources, a bipartite graph is constructed:

- nodes being evidence pieces or entities that occur in these pieces, and
- edges connecting an evidence piece and an entity if the entity occurs in the evidence.

The task for the GNN is to jointly score the evidence and the entity nodes in a multi-task learning setup. The latter are the answer candidates, and the evidence should give faithful explanation for an answer. We build on our earlier work on explainable QA [5].

The node encodings are initialized with cross-encoder embeddings (see below) for node contents and the SI of the question. The inference iteratively adjusts the encodings based on message passing from neighboring nodes. The GNN is trained via weak supervision from question-answer pairs: evidence nodes are labeled as relevant if they are connected to a gold answer. More technical details are given in [5].

QUASAR invokes the GNN in multiple rounds, iteratively reducing top- k to top- k^* nodes with $k^* \ll k$. In practice, we would typically consider two rounds: re-ranking top-1000 and pruning to top-100, and then reducing to top-30 or top-10, which are passed to the answer generation stage. Note that this keeps the GNN at a tightly controlled size, so that its computational costs at inference-time are much smaller than those of an LLM.

CE-based RF. An alternative to the GNN inference is to employ a cross-encoder for scoring and re-ranking the evidence pieces. These are transformers (typically with a small LM like BERT) that are fine-tuned for scoring the relatedness between a query and a document [26]. In our case, the comparison is between the question SI and the evidence piece. In our experiments, we make use of two different cross-encoders, both trained on the MS-MARCO benchmark for passage retrieval [1], and fine-tuned on the respective benchmark (leveraging the same weak supervision data as for the GNNs), the difference being in model size.² We use the smaller model to reduce top-1000 to top-100, and the larger model to go further down from top-100 to top-30.

3.4 Answer Generation (AG)

The last stage follows mainstream practice to invoke an LLM in a retrieval-augmented manner. We call a ‘small-scale’ LLM, specifically a fine-tuned LLaMA-3.1 model (8B-Instruct)³, with a prompt⁴ consisting of:

- the concatenated SI of the original question, and
- the top-30 (or other top- k^* with small k^*) evidence pieces.

²<https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-4-v2> and <https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2>

³<https://huggingface.co/meta-llama/llama-3.1-8B-Instruct>

⁴The specific prompt is “SI: <concatenated SI> Evidence: <evidence pieces>”.

By the previous down-filtering of the original pool of evidence pieces, this last step has affordable cost in terms of computation time and energy consumption.

Fine-Tuning the LLM: We considered adding an instruction to the prompting, such as “answer this question solely based on the provided evidence snippets”. However, this turned out to be ineffective. The reason why the model works well without such instructions is our task-specific fine-tuning. We perform this by running the training data of benchmarks through the QUASAR pipeline, and training the AG stage with the top-30 evidence pieces as input. Thus, the fine-tuning makes the model learn the role of evidence for RAG-based QA.

Explanations: The top-30 evidence pieces can be used to provide users with explanation of answers. Optionally, these could be reduced further for comprehensibility. Alternatively, we can fine-tune the LLM to provide both answers and concise explanations. Since we can infer which evidences in the input mention the annotated ground-truth answers, our method could be fine-tuned to provide such *answering evidences* as well (cf. [11]).

4 Experiments

4.1 Experimental setup

Benchmarks. We run experiments on three benchmarks with different characteristics of questions.

- **COMP MIX.** COMP MIX [6] is a benchmark which was specifically designed for evaluating QA systems operating over heterogeneous sources. The dataset has 9,410 questions, out of which 2,764 are used for testing. Answers are crisp entity names, dates, or other literals.
- **CRAG.** We further evaluate on a subset of the CRAG [36] dataset, which was recently released as a testbed for RAG-based QA systems. We utilize the same pipeline and sources as outlined in Section 2, without using the web snippets or APIs provided with CRAG. This way we focus on entity-centric questions that do not require access to live web data (e.g., news feeds), and disregard cases where the results would be up-to-date quantities. This restricts the test data to 436 entity-centric questions, still enough for a proof of concept.
- **TIME QUESTIONS.** To showcase the generalizability of our pipeline, we conduct experiments on TIME QUESTIONS [14], a benchmark for temporal QA. The dataset requires temporal understanding and reasoning, which are well-known limitations of LLMs [8]. TIME QUESTIONS has 16,181 questions (3,237 for testing).

Typical examples for the questions in these three benchmarks are:

COMP MIX: *Which player won the most number of Man-of-the-Match titles in the FIFA world cup of 2006?*

CRAG: *What was the worldwide box office sales for little hercules?*

TIME QUESTIONS: *Which club did Cristiano Ronaldo play for before joining Real Madrid?*

Baselines. As competitors or reference points to QUASAR, we study the performance of the following methods:

- **Generative LLMs.** We compare QUASAR against out-of-the-box LLMs: **GPT-3 (text-davinci-003)**, **GPT-4 (gpt-4)** and **LLAMA3 (meta-llama/llama-3.1-8B-Instruct)**. The same prompt is used for all LLMs, consistent with previous work [6, 38]: “*Please answer the following question by providing the crisp answer entity, date, year, or numeric number. Q: <question>*”.

- **Heterogeneous QA methods.** CONVINSE [3], UNIK-QA [27], EXPLAIGNN [5] are QA methods designed to integrate heterogeneous sources: text, tables and KG. All of these integrate the exact same sources as QUASAR.
- **STATE-OF-THE-ART.** For COMPMIX and TIMEQUESTIONS, we also compare against state-of-the-art methods from the literature: SPAGHETTI [38] and UN-FAITH [15], which are among the best performing systems.

Results are taken from the literature whenever applicable. On CRAG, we use the models trained on COMPMIX for QUASAR and heterogeneous QA baselines.

Metrics. We measure *precision at 1* (**P@1**) as our main metric [29] on all benchmarks. On CRAG, we manually annotate answer correctness, as the ground-truth answer formats vary (e.g., entity name variants, lists, sentences).

We also compute the number of neural parameters aggregated over all sub-modules (**#Parameters**). Parameter counts for GPT-models are taken from [25] (GPT-4 might have less active parameters during inference).

For further analysis we measure *answer presence* (**AP@k**), i.e. whether the answer is present in the top- k ranked evidence pieces, and *mean reciprocal rank* within the top- k evidences (**MRR@k**).

Configuration. Our implementation uses the **Llama3.1-8B-Instruct** model for the AG stage. For the QU, ER and RF stages we adopt code from the EXPLAIGNN project.⁵ For the ER stage, we use CLOCQ, setting its specific parameters to $k = 10$ and $p = 1,000$.

As default, we use the GNN technique for the RF stage. For efficiency, we use light-weight models for initializing the GNN encoders – the same models used for the CE-based RF.⁶ The GNNs are trained for 5 epochs with an epoch-wise evaluation strategy, i.e. we choose the model with the best performance on the respective dev set. We train the GNNs on graphs with a maximum of 100 evidence and 400 entity nodes (as scored by BM25). During inference, the first GNN is applied on graphs with 1,000 evidence and 4,000 entity nodes, shrinking the pool of evidence pieces to the top-100. The second GNN then runs on graphs with 100 evidence and 400 entity nodes. The factor of 4 entities per evidence (on average) holds sufficient for the observed data, and enables batched inference. Other parameters are kept as is.

The AG model, based on **Llama3.1-8B-Instruct**, is fine-tuned for 2 epochs with a warm-up ratio of 0.01 and a batch size of 8, again with an epoch-wise evaluation strategy. Other parameters are set to the default Hugging Face training parameters.⁷

4.2 Main results

QUASAR is competitive on all benchmarks. Main results of our experiments are shown in Table 5. First of all, we note that QUASAR achieves competitive performance across all three benchmarks.

On COMPMIX, baselines for heterogeneous QA and LLAMA3 perform similarly, whereas GPT-based LLMs can answer more than 50% of the questions correctly. QUASAR exhibits substantially higher performance, on par with the state-of-the-art method SPAGHETTI [38] (which is based on GPT-4).

On the CRAG dataset, P@1 drops for all methods except for GPT-4. The benchmark includes realistic questions, which can be ambiguous/confusing (“*who was the director for the report?*”), on “exotic” entities with answers in social media (“*how many members does the teknoist have?*”), or require up-to-date

⁵<https://explaignn.mpi-inf.mpg.de>

⁶<https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-4-v2> and <https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-6-v2>

⁷https://huggingface.co/docs/transformers/v4.46.2/en/main_classes/trainer#transformers.TrainingArguments

information (“*when did chris brown release a song or album the last time?*”), and other cases that are challenging for all methods.

Finally, QUASAR establishes new state-of-the-art performance on the TIMEQUESTIONS benchmark. Interestingly, all of the tested LLMs show greatly reduced performance on this benchmark, which inherently requires temporal understanding and reasoning – a known weakness of stand-alone LLMs.

Method ↓ / Benchmark →	COMP MIX	CRAG	TIMEQUESTIONS	#Parameters
GPT-3	0.502	–	0.224	175,000 M
GPT-4	0.528	0.633	0.306	1,760,000 M
LLAMA3 [33] (8B-Instruct)	0.431	0.385	0.178	8,030 M
CONVINSE [3]	0.407	0.298	0.423	362 M
UNIK-QA [27]	0.440	0.280	0.424	223 M
EXPLAIGNN [5]	0.442	0.303	0.525	328 M
STATE-OF-THE-ART	0.565 (SPAGHETTI [38])	–	0.571 (UN-FAITH [15])	–
QUASAR (ours)	0.564	0.362	0.754	8,218 M

Table 5: End-to-end P@1 of QUASAR and baselines on three benchmarks. Results for GPT-3 and GPT-4 are taken from the literature [6, 15]. GPT-3 is not accessible anymore, hence no results on CRAG.

Integration of heterogeneous sources is vital. QUASAR integrates evidence from text, KG and tables into a unified framework. We aim to better understand how this affects the answering performance of the method. Table 6 shows end-to-end answering performance of QUASAR with different combinations of the input sources. The results clearly indicate that all types of sources contribute, with option Text+KG+Tables performing best, with a large margin over tapping only single source types.

4.3 Analysis

Unified retrieval enhances performance. In the RF stage, we re-rank and filter evidence from different source types, and feed the unified top- k^* into the AG stage. We conduct a comparison in which we consider the top-10 evidence pieces from each source type individually. This gives equal influence to KG, text and tables, whereas our default is based on global ranking. Table 7 shows the results for this analysis, showing our default choice performs better. The reason is that different questions require different amounts of evidence from each of the source types.

QUASAR works well with small amounts of evidence. We investigate the influence of the number of evidence pieces fed into the AG stage, varying it from 5 to 100. Results are shown in Figure 22. As the curve shows, there is a sharp increase in precision as we add evidence up to 30 or 40 pieces, which is around our default of top-30. This indicates that a certain amount of evidence is needed, to overcome the inherent noise and arrive at sufficient answer presence. As we increase the amount of evidence further, we observe a saturation effect, and eventually a degradation of performance. Too much evidence not only has diminishing returns, but can actually be confusing for the AG stage. This reconfirms our heuristic choice of top-30: enough for good answering while keeping computational costs reasonably low.

Ablation study on re-ranking. For more insight on the possible configurations of the RF stage, we conducted an ablation study with different options, including solely relying on the initial BM25 scoring without explicit re-ranking. The results are shown in Table 8. We observe that the iterative reduction in

Benchmark →	COMPMIX			TIMEQUESTIONS		
Input sources ↓ / Metric →	P@1	AP@100	AP@30	P@1	AP@100	AP@30
Text	0.455	0.563	0.531	0.539	0.515	0.487
KG	0.481	0.677	0.637	0.724	0.701	0.674
Tables	0.432	0.501	0.482	0.536	0.347	0.328
Text+KG	0.537	0.749	0.706	0.745	0.776	0.748
Text+Tables	0.503	0.632	0.594	0.567	0.578	0.549
KG+Tables	0.524	0.728	0.692	0.743	0.731	0.703
Text+KG+Tables	0.564	0.759	0.724	0.754	0.776	0.749

Table 6: Answer presence and answering precision of QUASAR with different combinations of input sources (on the respective test sets).

Input evidences ↓ / Metric →	P@1	AP@30
Top-30 Text+KG+Tables (ours)	0.574	0.710
Top-10 Text + Top-10 KG + Top-10 Tables	0.560	0.709

Table 7: Answer presence and precision of QUASAR for different choices of top-30 (on COMPMIX dev set).

two steps is slightly better than the single-step variants (going down from top-1000 to top-30 in one RF step). Between the two options of using a GNN or a CE, the differences are negligible. A notable effect is that our RF techniques retain the answer presence at a very high level, only a bit lower than for the initial top-1000. The last two rows of Table 8 demonstrate that RF is crucial: without explicit re-ranking, the technique of just picking smaller top- k from the original BM25 model leads to substantial degradation in both answer presence and precision.

Quality of SI. To assess the quality and robustness of the Structured Intents, we inspected a sample of questions and their SIs. Table 9 gives three anecdotic examples. We show SIs generated by QUASAR, which makes use of the pre-existing collection from the COMPMIX benchmark for training. This training data was obtained via different heuristics, which can be a limiting factor when user intents become more complex.

Therefore, we also looked at SIs derived via in-context learning (ICL) using GPT-4 with 5 handcrafted examples. As shown in our earlier work on temporal QA [15], such data can be used for training smaller models (e.g., BART), which can greatly boost the completeness and overall quality of the generated SIs.

From the sampled set, we observed that the ICL-based SIs are more complete with all slots filled, whereas the BART-based SIs focused more on the main slots Answer-type, Entities and Relation. However, both approaches achieve very high quality in filling the slots, capturing the user’s information need very well.

Interestingly, when questions get complicated, with nested phrases, the ICL-based variant succeeds in decomposing the questions, based on only 5 ICL examples. For example, for the question “which German state had the most Corona-related death cases in the first year after the outbreak?” the Time slot becomes “first year after Corona outbreak”, which can be resolved to identify the temporal scope. In general, we believe that such question decomposition, beyond simple temporal constraints, would be an interesting theme for future work.

Refraining from answering. We can train our model to refrain from answering in scenarios where the

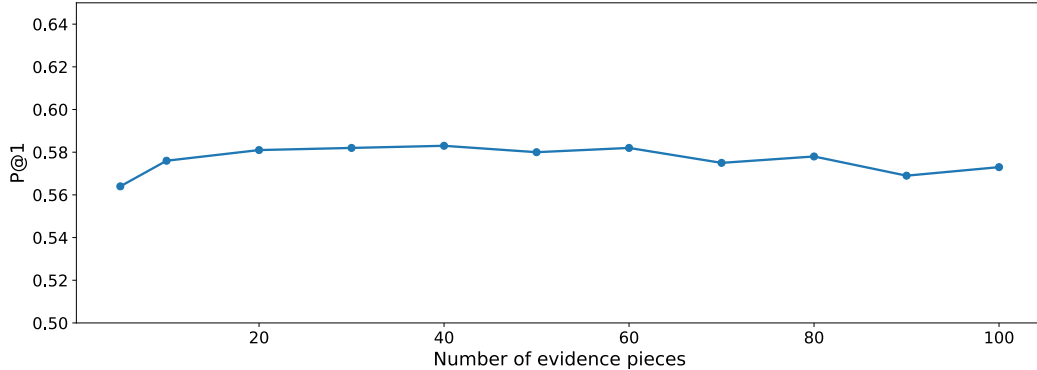


Figure 22: Performance of QUASAR on the COMPMIX dev set with different numbers of evidence.

RF Method ↓ / Metric →	COMPMIX (dev set)			
	P@1	AP@100	AP@30	MRR@100
GNN: 1000 → 100 → 30	0.574	0.738	0.710	0.572
CE: 1000 → 100 → 30	0.573	0.740	0.721	0.553
GNN: 1000 → 30	0.567	<i>n/a</i>	0.710	0.567
CE: 1000 → 30	0.570	<i>n/a</i>	0.715	0.558
BM25: 100 (w/o GNN or CE)	0.490	0.652	<i>n/a</i>	0.259
BM25: 30 (w/o GNN or CE)	0.468	<i>n/a</i>	0.534	0.259

Table 8: Ablation study for different RF strategies of QUASAR on the COMPMIX dev set. The answer presence in the RF input with top-1000 evidence pieces is 0.760.

provided evidence does not contain an answer to the question. Specifically, during training, when the answer is not present in the evidence, we change the target answer to unknown. This variant is referred to as QUASAR(faithful).

We measure the ratio of questions for which unknown is provided as answer, and the P@1 restricted to questions that are answered. The accuracy of refraining from answering is measured as well, based on whether the answer is present in the evidence or not. We conduct this experiment on COMPMIX and TIMEQUESTIONS, for which we can compute answer presence exactly. We also compute results for LLAMA3, which is already instructed with the option to answer “don’t know”. Table 10 shows the results. For COMPMIX, we observe that QUASAR has high accuracy on refraining when appropriate, whereas LLAMA3 tends to be overconfident with a very small rate of unknowns, leading to incorrect answers.

5 Insights, Limitations, and Challenges

Benchmark Performance. Our method, RAG-based QUASAR with an 8B LLaMA model, outperforms much larger LLMs like GPT-4 on two of the three benchmarks, with a very large margin for temporal questions. Obviously, pre-trained LLMs have only limited sense of properly positioning “remembered” facts on the timeline even with training data that exceeds ours by several orders of magnitude. This confirms our intuition that LLMs alone are not good at “recalling” higher-arity relations that require combining distant pieces of evidence. This is a sweet spot for RAG. Only for the CRAG benchmark, QUASAR is substantially inferior to a full-blown LLM. This is likely due to the nature of the questions: not necessarily the complexity of the information needs, but the need for more web sources (beyond what our experiments tap into).

Question	Current SI by QUASAR	SI via ICL
<i>what was disneys first color movie?</i>	Ans-Type: <i>animated feature film</i> Entities: <i>disneys</i> Relation: <i>was first color movie</i>	Ans-Type: <i>film, animated film</i> Entities: <i>Disney</i> Relation: <i>first color movie</i> Time: <i>first</i>
<i>at the oscars, who won best actor in 2018?</i>	Ans-Type: <i>human</i> Entities: <i>at the oscars</i> Relation: <i>who won best actor in 2018</i>	Ans-Type: <i>person, actor</i> Entities: <i>Oscars, 2018</i> Relation: <i>won best actor</i> Time: <i>2018</i>
<i>which German state had the most Corona-related death cases in the first year after the outbreak?</i>	Ans-Type: <i>state</i> Entities: <i>Germany, Corona</i> Relation: <i>which state had the most related death cases in the first year after the outbreak</i>	Ans-Type: <i>location, state</i> Entities: <i>Germany, Corona-related deaths</i> Relation: <i>highest count of death cases</i> Location: <i>Germany</i> Time: <i>first year after Corona outbreak</i>

Table 9: Examples for pairs of question and generated SI.

Metric → Method ↓	COMPMIX				TIMEQUESTIONS			
	P@1	P@1 (answered)	Refrain rate	Refrain accuracy	P@1	P@1 (answered)	Refrain rate	Refrain accuracy
LLAMA3	0.431	0.471	0.089	<i>n/a</i>	0.177	0.276	0.392	<i>n/a</i>
QUASAR (faithful)	0.497	0.713	0.303	0.838	0.597	0.804	0.257	0.864

Table 10: Performance of QUASAR with option to refrain from answering (“don’t know”).

Cost/Performance Ratio. The most important take-away from our experiments is that QUASAR achieves its competitive performance at a much lower cost than the full LLMs. Assuming that the consumed GFlops are proportional to the number of model parameters, QUASAR achieves a cost reduction by a factor of 200x for GPT-3 and 2000x for GPT-4. This does not only mean less computation, but also a massively lower electricity bill and climate impact.

Role of Question Understanding. We did not systematically investigate the influence of the Structured Intent in the QUASAR pipeline. However, the comparison to the big GPT models reflects the role of the SI, as we prompt the GPT models in their natural mode with the original questions. The linearized sequence of available SI slots does not always have major advantages, but there are enough cases where specific facets provide crucial cues. This holds especially for the Entities slot, as this drives the gathering of evidence in the ER stage (cf. [3]), and for the Time slot, as these cues are often decisive for temporal questions (cf. [15]).

Role of Re-Ranking. As our ablation studies show, merely using top- k evidence from an initial BM25-style ranking does not provide good performance. Also, there seems to be sweet spot in the choice of k : we need enough evidence for connecting the dots if the question requires multiple pieces of information, or for corroborating candidates if the question finds many useful but noisy pieces. In the experiments, $k = 30$ turns out to be good choice; much lower k results in insufficient evidence, and much larger k leads to saturation and ultimately degrading performance. Our argument for iteratively shrinking the candidate set in multiple rounds of re-ranking is substantiated in our experiments, but the gain of doing this, compared to GNN- or CE-based re-ranking from 1000 to 30, is not big. More

research is called to better understand the role of ranking in RAG.

Limitations of Evidence Retrieval. For ER, we adopted more or less standard techniques. The results showed very good answer presence, in the order of 75% in the top-100 or even top-30. An important case where this is insufficient are questions that require aggregating information over a large number of evidence pieces. An example is asking for the life-time total of 3-point scores of the basketball player Dirk Nowitzki. This requires collecting a set of per-season tables with NBA player statistics, but also other web sources with numbers for his career before he joined the NBA (including his youth teams). Of course, there are sometimes shortcuts like a Wikipedia article or biography mentioning the total number, but this cannot be universally assumed. The bottom line is that ER should be reconsidered as well, striving to improve the recall dimension.

Limitations of Answer Generation. For AG, we simply rely on a LLM, using it as an extractor (“reader”) from the given evidence. Despite the wide belief that LLMs can perform deep reasoning over many pieces of evidence, our experience is that the extraction works only well – robustly and faithfully – for relatively simple questions with a few multi-hop joins or simple aggregation over a few pieces. However, complicated questions such as asking for the top-100 NBA players with the largest number of life-time 3-point scores (again including their pre-NBA careers) are currently out of scope and will likely remain so for quite some time. This offers many opportunities for pushing the envelope further.

Trust in Data Sources. In our experiments, we considered all heterogeneous sources as trustworthy and unbiased. With focus on Wikidata and Wikipedia, this assumption has been well justified. In the wild, however, input data for RAG-based systems likely exhibit a wide spectrum of quality issues, in terms of stale information, biased positions, or simply false statements. Identifying trustworthy and up-to-date evidence and dealing with conflicting data, has been explored in other contexts (e.g., for KG curation [9]), but remains a major challenge for RAG-based QA.

Open Challenges and Future Work. The best-performing methods in our experiment, mostly QUASAR, reach P@1 values of 56% for COMPMIX and 75% for TIMEQUESTIONS. For the latter, the answer presence in the top-100 is only slightly higher; so the AG stage hardly misses anything. However, for COMPMIX, the answer presence is 75% – much higher than what our system can actually answer. Obviously, closing this gap is a major direction to pursue, with focus on the RF and AG stages. However, missing one fourth of the answers completely in the top-100 pool, is a big problem as well. This requires improving recall at the ER stage, possibly with better guidance by the QU, which in turn needs more sources beyond the scope of our experiments (currently limited to Wikidata and Wikipedia).

In general, we need to think beyond this kind of “benchmark mindset”. Even if we reached 80% or 90% precision and recall, we would still have a substantial fraction of questions that are answered incorrectly or not at all. The remaining errors may not be a problem for chatbots, but they would be a showstopper for the deployment of mission-critical applications in business or science. We believe that this big gap is a shortcoming of all methods, not an issue that comes from the data alone. For trivia-style QA, as looked at in this paper, a smart human in “open book” mode and no time limitation should be able to properly answer practically all questions, just by reading pieces of web contents and putting things together. Neither LLMs nor state-of-the-art RAG are the final solution; substantial research and creative ideas are needed to further advance QA.

References

- [1] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, Tong Wang. MS MARCO: A Human Generated MACHine Reading COMprehension Dataset. In arXiv 2018.
- [2] Danqi Chen, Adam Fisch, Jason Weston and Antoine Bordes. Reading Wikipedia to Answer Open-Domain Questions. In ACL 2017.
- [3] Philipp Christmann, Rishiraj Saha Roy, Gerhard Weikum. Conversational Question Answering on Heterogeneous Sources. In SIGIR 2022.
- [4] Philipp Christmann, Rishiraj Saha Roy, Gerhard Weikum. Beyond NED: Fast and Effective Search Space Reduction for Complex Question Answering over Knowledge Bases. In WSDM 2022.
- [5] Philipp Christmann, Rishiraj Saha Roy, Gerhard Weikum. Explainable Conversational Question Answering over Heterogeneous Sources via Iterative Graph Neural Networks. In SIGIR 2023.
- [6] Philipp Christmann, Rishiraj Saha Roy, Gerhard Weikum. CompMix: A Benchmark for Heterogeneous Question Answering. In WWW 2024.
- [7] Herve Dejean, Stephane Clinchant, Thibault Formal. A Thorough Comparison of Cross-Encoders and LLMs for Reranking SPLADE. In arXiv 2024.
- [8] Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. Time-Aware Language Models as Temporal Knowledge Bases. In TACL 2022.
- [9] Xin Luna Dong, Evgeniy Gabrilovich, Kevin Murphy, Van Dang, Wilko Horn, Camillo Lugaresi, Shaohua Sun, Wei Zhang. Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. In PVLDB 2015.
- [10] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, Haofen Wang. Retrieval-Augmented Generation for Large Language Models: A Survey. In arXiv 2023.
- [11] Tianyu Gao, Howard Yen, Jiatong Yu, Danqi Chen. Enabling Large Language Models to Generate Text with Citations. In EMNLP 2023.
- [12] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, Ming-Wei Chang. Retrieval Augmented Language Model Pre-Training. In ICML 2020.
- [13] Gautier Izacard, Edouard Grave. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In EACL 2021.
- [14] Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. Complex Temporal Question Answering on Knowledge Graphs. In CIKM 2021.
- [15] Zhen Jia, Philipp Christmann, Gerhard Weikum. Faithful Temporal Question Answering over Heterogeneous Sources. In WWW 2024.
- [16] Ehsan Kamaloo, Nouha Dziri, Charles L. A. Clarke, Davood Rafiei. Evaluating Open-Domain Question Answering in the Era of Large Language Models. In arXiv 2023.

- [17] Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, Colin Raffel. Large Language Models Struggle to Learn Long-Tail Knowledge. In ICML 2023.
- [18] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, Wen-tau Yih. Dense Passage Retrieval for Open-Domain Question Answering. In EMNLP 2020.
- [19] Dongkyu Lee, Chandana Satya Prakash, Jack FitzGerald, Jens Lehmann. MATTER: Memory-Augmented Transformer Using Heterogeneous Knowledge Sources. In ACL 2024.
- [20] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In ACL 2020.
- [21] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In NeurIPS 2020.
- [22] Jimmy Lin, Rodrigo Frassetto Nogueira, Andrew Yates. Pretrained Transformers for Text Ranking: BERT and Beyond. In Morgan & Claypool Publishers 2021.
- [23] Shicheng Liu, Jialiang Xu, Wesley Tjangnaka, Sina J. Semnani, Chen Jie Yu, Monica Lam. SUQL: Conversational Search over Structured and Unstructured Data with Large Language Models. In NAACL-HLT 2024.
- [24] Vaibhav Mavi, Anubhav Jangra, Adam Jatowt. Multi-hop Question Answering. In Foundations and Trends in Information Retrieval 2024.
- [25] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large Language Models: A Survey. In arXiv 2024.
- [26] Rodrigo Frassetto Nogueira, Kyunghyun Cho. Passage Re-ranking with BERT. In arXiv 2019.
- [27] Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, Scott Yih. UniK-QA: Unified Representations of Structured and Unstructured Knowledge for Open-Domain Question Answering. In NAACL-HLT 2022.
- [28] Anna Rogers, Matt Gardner, Isabelle Augenstein. QA Dataset Explosion: A Taxonomy of NLP Resources for Question Answering and Reading Comprehension. In ACM Computing Surveys 2023.
- [29] Rishiraj Saha Roy, Avishek Anand. Question Answering for the Curated Web: Tasks and Methods in QA over Knowledge Bases and Text Collections. In Synthesis Lectures on Information Concepts, Retrieval, and Services, Morgan & Claypool Publishers 2021.
- [30] Soumajit Pramanik, Jesujoba Alabi, Rishiraj Saha Roy, Gerhard Weikum. UNIQORN: Unified Question Answering over RDF Knowledge Graphs and Natural Language Text. In Journal of Web Semantics 2024.
- [31] Haitian Sun, Tania Bedrax-Weiss, William W. Cohen. PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text. In EMNLP/IJCNLP 2019.

- [32] Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, Xin Luna Dong. Head-to-Tail: How Knowledgeable are Large Language Models (LLMs)? A.K.A. Will LLMs Replace Knowledge Graphs? In NAACL-HLT 2024.
- [33] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, Guillaume Lample. Llama: Open and efficient foundation language models. In arXiv 2023.
- [34] Shirley Wu, Shiyu Zhao, Michihiro Yasunaga, Kexin Huang, Kaidi Cao, Qian Huang, Vassilis N. Ioannidis, Karthik Subbian, James Zou, Jure Leskovec. STaRK: Benchmarking LLM Retrieval on Textual and Relational Knowledge Bases. In arXiv 2024.
- [35] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. In IEEE Transactions on Neural Networks and Learning Systems 2021.
- [36] Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze D. Gui, Ziran W. Jiang, Ziyu Jiang, Lingkun Kong, Brian Moran, Jiaqi Wang, Yifan Ethan Xu, An Yan, Chenyu Yang, Eting Yuan, Hanwen Zha, Nan Tang, Lei Chen, Nicolas Scheffer, Yue Liu, Nirav Shah, Rakesh Wanga, Anuj Kumar, Wen-tau Yih, Xin Luna Dong. CRAG – Comprehensive RAG Benchmark. In arXiv 2024.
- [37] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, Jure Leskovec. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In NAACL-HLT 2021.
- [38] Heidi C. Zhang, Sina J. Semnani, Farhad Ghassemi, Jialiang Xu, Shicheng Liu, Monica S. Lam. SPAGHETTI: Open-Domain Question Answering from Heterogeneous Data Sources with Retrieval and Semantic Parsing. In ACL 2024.
- [39] Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Yong Liu, Shen Huang. End-to-End Beam Retrieval for Multi-Hop Question Answering. In NAACL-HLT 2024.
- [40] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, Ji-Rong Wen. A Survey of Large Language Models. In arXiv 2023.
- [41] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Bin Cui. Retrieval-Augmented Generation for AI-Generated Content: A Survey. In arXiv 2024.
- [42] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, Tat-Seng Chua. Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering. In arXiv 2021.

Evaluating the Factuality of Large Language Models using Large-Scale Knowledge Graphs

Xiaoze Liu[♣], Feijie Wu[♣], Tianyang Xu[♣], Zhuo Chen[♡], Yichi Zhang[♡],
Xiaoqian Wang[♣], Jing Gao[♣]

[♣] Purdue University, West Lafayette, IN 47907, USA

[♡] Zhejiang University, Hangzhou, China

{xiaoze, wu1977, xu1868, joywang, jinggao}@purdue.edu
{zhuo.chen, zhangyichi2022}@zju.edu.cn

Abstract

The advent of Large Language Models (LLMs) has significantly transformed the AI landscape, enhancing machine learning and AI capabilities. Factuality issue is a critical concern for LLMs, as they may generate factually incorrect responses. In this paper, we propose **GraphEval** to evaluate an LLM’s performance using a substantially large test dataset. Specifically, the test dataset is retrieved from a large knowledge graph with more than 10 million facts without expensive human efforts. Unlike conventional methods that evaluate LLMs based on generated responses, **GraphEval** streamlines the evaluation process by creating a judge model to estimate the correctness of the answers given by the LLM. Our experiments demonstrate that the judge model’s factuality assessment aligns closely with the correctness of the LLM’s generated outputs, while also substantially reducing evaluation costs. Besides, our findings offer valuable insights into LLM performance across different metrics and highlight the potential for future improvements in ensuring the factual integrity of LLM outputs. The code is publicly available at <https://github.com/xz-liu/GraphEval>.

1 Introduction

The rapid progress of Large Language Models (LLMs) has markedly boosted artificial intelligence and machine learning due to their strong contextual text generation capabilities. Despite these groundbreaking advancements, recent works [34, 35] have highlighted the significance of LLMs evaluation. LLMs are prone to producing seemingly authentic yet factually inaccurate responses, a phenomenon known as hallucination [19]. Such errors may stem from outdated or incorrect data during training or the model’s learned associations, impacting its reliability. The evaluation, therefore, helps identify instances of hallucination and understand the LLM’s ability to generate coherent and contextually relevant text, i.e., factuality of LLM outputs.

Recent efforts have been put into the factuality evaluation of LLM. For example, [26] introduce FELM, a benchmark comprising diverse factual samples across various domains. Moreover, [24] and [22] utilize external tools (e.g., search engines and a well-trained factual LLM) to estimate the factuality of the generated texts. Representing structured knowledge related to real-world objects, Knowledge Graphs (KGs) [11, 20, 28, 41] have gained prominence for LLM factuality assessments. They primarily originate from Wikipedia, encapsulate factual information for AI tasks, and form the knowledge base with datasets like Natural Questions [21].

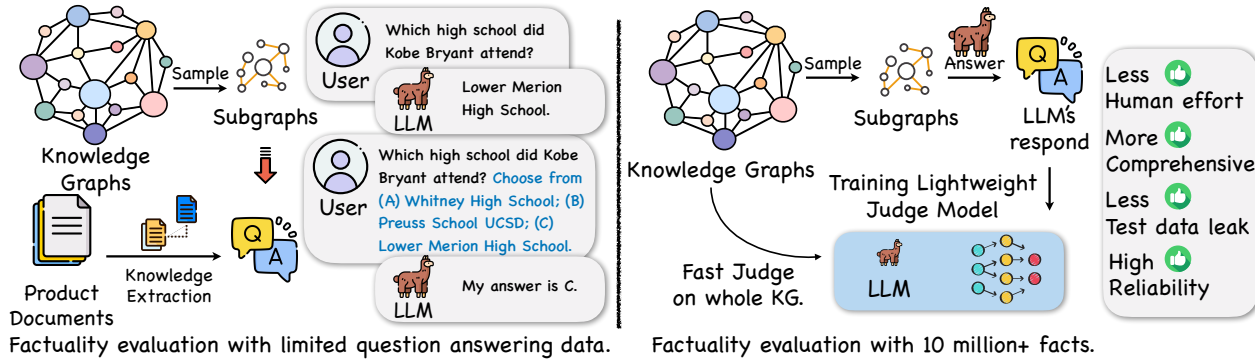


Figure 23: Existing works compared to the proposed GraphEval on factuality evaluation.

Several studies [1, 2] have focused on creating benchmarks from knowledge sources by posing factual questions derived from triples in KGs. These methods, as depicted in the left part of Figure 23, either (i) sample subgraphs from large KGs or (ii) extract a subset of knowledge from text documents to construct multiple-choice or text question-answer pairs. Those question pairs are then posed to LLMs to assess their factuality. However, the above-mentioned methods or evaluation strategies face challenges in comprehensively evaluating the factuality of LLMs. *Firstly*, the scope of evaluation data is often limited or incomplete, focusing predominantly on specific domains. This limitation restricts the evaluation’s breadth and undermines its applicability across various contexts, failing to cover the wide range of topics LLMs are expected to handle. The specialized nature of these datasets means that the evaluation may not accurately reflect the model’s performance in generating factual content across a broader spectrum of subjects. *Secondly*, the process of factuality evaluation itself is inherently time-consuming and costly. It necessitates that an LLM generate full texts, which must then be meticulously assessed for accuracy and reliability. This comprehensive generation and detailed review demand significant computational resources and extensive human effort [35] for validation. As a result, the process becomes less feasible for regular or large-scale applications, limiting the frequency and scope of practical evaluations. *Lastly*, due to the limited size of the evaluation data, there may be biases in the benchmarks [16], or risks of test data leakage [38], which might compromise the validity of the evaluations. Together, these challenges underscore the need for more scalable, efficient, and domain-agnostic approaches to evaluating the factuality of LLM-generated texts.

To this end, we propose GraphEval, which consists of two novel features in terms of the design, as presented on the right side of Figure 23. First, we utilize KGs that encapsulate factual information sourced from verifiable content like Wikipedia. With the KG, millions of prompts can be automatically generated, leading to a significant saving of human efforts in labeling the ground truth. From the data perspective, the KG gives a more diversified and comprehensive evaluation of the LLMs’ factuality. Second, the proposed method efficiently reduces the computation costs and speeds up the evaluation process. Specifically, we incorporate a highly reliable and lightweight judge model to decide whether an LLM can generate an accurate response to a designated question. Instead of generating the full text, the judge model returns three options (i.e., True, False, and I don’t know) to simulate LLMs’ responses to a given prompt. To ensure the reliability of the simulated results, the judge model is trained based on a few question-answer pairs, where the questions are sampled from KGs and the answers are generated by the target LLM. As a result, the judge model can serve as a replacement for the factuality evaluation of the facts extracted from large-scale KGs. In summary, our contributions are as follows:

- We propose GraphEval, a large-scale evaluation framework that assesses the factuality of LLMs using KGs. GraphEval evaluates the factuality of LLMs using the entire KGs, providing a more diversified and comprehensive evaluation of the LLMs’ factuality.
- We introduce a judge model to assist with the evaluation process, which reduces the computational

cost and enhances the efficiency of the evaluation. We also give a theoretical analysis of the judge model to demonstrate its validity.

- We conduct extensive experiments on a large-scale KG, i.e., DBpedia, to demonstrate the effectiveness and efficiency of **GraphEval** in evaluating the factuality of LLMs.
- We provide an in-depth analysis of the LLM’s performance on the KGs, including the LLM’s performance with respect to relation types, head entity types, tail entity types, and the relation of LLM performance to degree and pageviews.

2 Related Work

Factuality Issue of LLMs Factuality issue [19, 39], is the issue that LLMs may produce content inconsistent with established facts. As outlined in [19], this issue may be due to: *(i)* LLMs lacking expertise in specific domains [3, 7]; *(ii)* LLMs’ unawareness of recent developments or changes [4, 9]; *(iii)* LLMs not retaining [21, 35, 40] or forgetting [17, 31, 36, 37, 50] knowledge from its training corpus; and *(iv)* LLMs failing to reason with the knowledge they possess [27, 47, 51]. The factuality issue has been addressed by various works, by incorporating Retrieval Augmented Generation (RAG) [12, 14], fine-tuning [24], and knowledge-enhanced models [13, 22]. To summarize, these approaches integrate other knowledge sources into the LLMs’ training process or use them to augment the models’ knowledge base, thus alleviating the factuality issue. Our work differ from them in that we evaluate the factuality of LLMs, rather than providing factuality enhancement methods.

Factuality Evaluation of LLMs The expanding use of LLMs across various domains necessitates the assurance of their output’s accuracy and reliability. A range of benchmarks and evaluation methodologies for assessing large language models (LLMs) are proposed. These works primarily focus on evaluating the factuality, truthfulness, reasoning capabilities, and adaptability to new information of LLMs. MMLU [42] and TruthfulQA [5] aim to measure the factuality and truthfulness of LLMs across diverse tasks, while C-Eval [46] focuses on the Chinese context, assessing models’ knowledge of Chinese culture and laws. There are also works [1, 2] that propose factuality evaluation using subsets of KGs. However, selecting subsets of KGs to test LLMs can introduce selection bias. For example, random sampling can focus more on a few popular domains or subjects more densely connected with others, thus not showing LLMs’ factuality on diversified topics. Our work addresses this limitation by proposing a resource-efficient method to evaluate the factuality of LLMs which allows evaluations on whole KGs instead of subsets, thus providing a more diversified and comprehensive evaluation of the LLMs’ factuality, enabling an extensive assessment of LLM’s factuality and reasoning abilities in a way that existing individual benchmarks do not as they only focus on specific aspects.

Using KGs in LLMs KGs are structured representations of factual knowledge, typically in the form of (head, relation, tail) triples. There are lots of efforts in constructing [11], and reasoning [43] on KGs. This has made KGs an indispensable resource of factual knowledge for AI tasks. Currently, the most common way of integrating KGs with LLMs is using KGs as an external knowledge source to enhance LLM performance by pre-training, fine-tuning, or in-context learning [8, 23, 25, 33, 45]. Our work is different from these works in that we use KGs to evaluate the factuality of LLMs, rather than enhancing the LLMs with KGs.

3 Method

GraphEval is designed to measure the factuality of a language model in relation to a KG. As presented in Figure 24, the proposed work is divided into three steps:

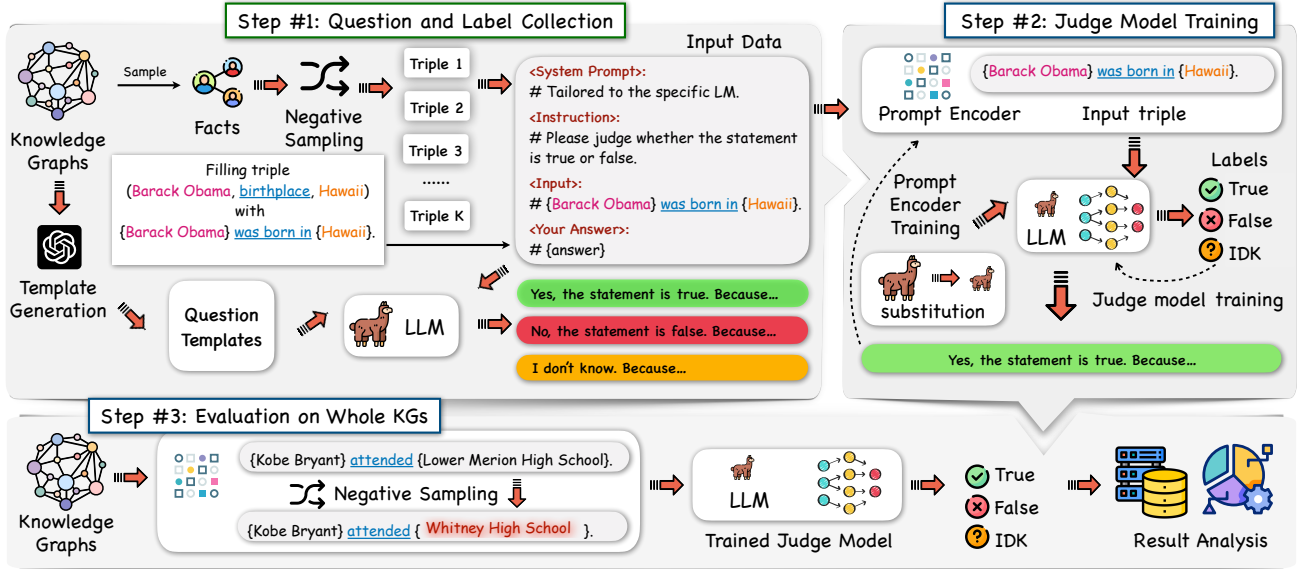


Figure 24: Overview of the GraphEval framework. **Step #1** retrieves KG statements and collect LLM judgments on them. **Step #2** trains the judge model which classifies LLM hidden states into three categories. **Step #3** evaluates the LLM on all KG statements with the judge model.

- **Step 1: Question and label collection from KGs and LLMs.** The model samples triples from KGs and converts each triple into a declarative statement with GPT-4-crafted templates. To prepare versatile statements, we employ *negative sampling*, where incorrect statements are intentionally generated. Afterward, those statements are posed to collect the labels answered by an LLM (i.e., Yes, No, and I don't know (IDK)).
- **Step 2: Judge model training.** With the triples collected in the first step, we train a judge model to avoid long-generated texts and conserve computational resources. In detail, inspired by [15], we train a classifier with LLMs' hidden states to make a selection within the above three options. We also apply p-tuning [18] to minimize the prompt/instruction size.
- **Step 3: Evaluation on whole KGs.** Similar to the first step, we retrieve all true/false statements from KGs. Subsequently, these statements are fed into the trained judge model to estimate the factuality of LLM. This process enables a thorough and multifaceted analysis of the LLM's performance in terms of factuality, drawing from a wide range of perspectives to provide a more comprehensive and diversified evaluation.

In the following sections, we will discuss the details of each step.

3.1 Question and Label Collection

Question Generation In order to evaluate the language model's ability to identify false statements, we directly construct a declarative sentence for each triple. This addresses the ineffectiveness of multiple-choice questions in our task. Firstly, multiple-choice prompts may cause misalignment with parametric knowledge in LLMs. Since LLMs mainly learn parametric knowledge through text data, in which knowledge facts are mostly represented as declarative sentences [6], employing multiple-choice questions may hinder the evaluation of factuality. Secondly, multiple-choice questions have more complex labels (i.e. A, B, C, D) than declarative sentences (i.e. True, False, IDK), which can complicate the tasks for the judge model, influencing the overall effectiveness. We use an example to illustrate this.

Example 3.1: For the triple (**Barack Obama**, **birthPlace**, **Hawaii**), a multi-choice question can be generated as **Where was Barack Obama born?** with choices **A. Hawaii B. Chicago C. New York D. Los Angeles**. Here, for the same triple, we can also generate another multi-choice question as **Where was Barack Obama born?** with choices **A. China, B. Hawaii, C. Japan, D. Russia**. The two questions represent the same triple, but the choices are different. As mentioned in the last paragraph, this can introduce complexity and potential misalignment with an LLM’s training and result in inconsistent responses.

To address the ineffectiveness of multiple-choice questions, we propose to directly ask the LLMs whether a statement is true or not. For instance, considering the triple (**Barack Obama**, **birthPlace**, **Hawaii**), we can formulate a fact **Obama was born in Hawaii** by integrating the entities **Barack Obama** and **Hawaii** into the template **{head} was born in {tail}**. Each template corresponds to the relation of a triple, and they are crafted to be clear and straightforward statements. GPT-4 is employed to generate these templates for all relations in the KG. These generated templates are then manually reviewed and refined to ensure their compatibility with the KG. Then, we can ask a question to the LLMs, such as **Is the statement "Barack Obama was born in Hawaii" true or false?**. Here, the templates are corresponding to the relations of the triples. This is because the number of relations in the KG is limited, while the number of triples is large. Therefore, we can use the relations to categorize the triples, and then use the templates to generate questions for each category. This can significantly reduce human labor, i.e., monitoring less than 1000 templates compared with monitoring more than 10 million triples. See the Appendix [A.5](#) for the detailed settings of the relation templates.

Negative sampling Although the declarative sentences simplify the training of the judge model, they alone are insufficient to evaluate the language model’s factual accuracy. LLMs can simply answer true for every question, and still get a high accuracy. To address this, we introduce negative sampling, a technique commonly used in KG completion tasks, to generate false statements. Specifically, we randomly replace one entity or relation in the original triple with another entity or relation sampled from the KG. For example, given the triple (**Barack Obama**, **birthPlace**, **Hawaii**), we can replace the tail entity **Hawaii** with another entity **Chicago** to form the false statement **Barack Obama was born in Chicago**. These false statements are then presented to the LLMs to evaluate their ability to identify falsehoods.

3.2 Judge Model

Normally, to evaluate the factual accuracy of a language model, we would generate questions from a KG and then pose these questions to the language model. However, given the expansive nature of KGs, it’s impractical to label every generated question by the LLM. A more efficient approach is to use the last token logits of the LLMs as their answers. However, recent research has highlighted discrepancies between these logits and the model’s actual text outputs [29]. Therefore, we introduce a novel judge model to assist with this task. The judge model, initially trained on a subset of labeled questions, is then employed to label the remaining questions. Uniquely, inspired by [15], the judge model utilizes the LLM’s hidden state as input, as a replacement of the LLM’s last layer with compressed output tokens. Specifically, three output classes are used: *True*, *False*, and *I don’t know*. The judge model is a two-layer feed-forward neural network, with a layer normalization and a ReLU activation function. This approach diverges from standard practices where LLMs generate answers, as here we only forward the transformer once. Consequently, this operation is significantly less resource-intensive than full answer generation, allowing the judge model to efficiently process a large number of questions with limited labeled data. With this model, we can glance at the correctness of an LLM, i.e., how likely the model

can answer a question relevantly and correctly. We evaluate the performance of the judge model using two metrics: (i) *Truthfulness*, i.e., the likelihood that the judge model prediction matches the LLM correctness under a given question; and (ii) *Informativeness*, i.e., the likelihood that the judge model does not give a prediction of ‘I don’t know.’ Since the evaluation is based on a general KG that spans multiple domains, other metrics such as “Relevance” would typically require a more specific contextual framework. Nonetheless, future research could explore the use of more context-specific metrics tailored to the LLM’s domain of application.

Efficiency To further enhance the judge model’s efficiency, we include 2 extra components. First, we found that the instruction prefix of the LLMs is too large for the judge model to process efficiently. We thus fine-tune a *prompt encoder* [32] to reduce the large input of the prompt prefix, which would be the same for all questions. Second, we found that our judge model, with the training process on the labeled dataset, is robust to the LLM’s hidden states. In experiments, we observed that our judge model can seamlessly utilize hidden states from distinct LLMs without significant differences in performance. For instance, within the LLaMA 2 model family, which contains 3 models with different parameters: 7B, 13B, and 70B, we found that the judge model’s performance is consistent regardless of whether the hidden states are from 7B, 13B, or 70B. Therefore, we can use the model with the least parameters, as a *substitute model* when computing the hidden states. This gives us a huge reduction in computational cost.

Analysis of Judge model In this part, we assume there are two datasets; one is for training the judge model, and the other is for evaluation, denoted by \mathcal{D}_S and \mathcal{D}_T , respectively. As the proposed judge model leads to a triple classification task, we assume a hypothesis portfolio $h = \{h_t, h_f, h_{idk}\}$, where these three hypotheses separately predict if a sample can be correctly answered by the LLM, i.e., True, False, and IDK. In other words, the hypothesis $\hat{h} \in h$ maps an input \mathbf{x} to $\{0, 1\}$, where 1 means the input satisfies the hypothesis conditions. For a given input \mathbf{x} , the equality $h(\mathbf{x}) = h_t(\mathbf{x}) + h_f(\mathbf{x}) + h_{idk}(\mathbf{x}) = 1$ always holds because the judge model provides an only output. Define the convex loss function for a hypothesis $\hat{h} \in h$ to be

$$L_{\mathcal{D}}(\hat{h}) = \sum_{(\mathbf{x}, y) \in \mathcal{D}} |\hat{h}(\mathbf{x}) - \mathbf{1}_{\hat{h}}(y)|,$$

where $\mathbf{1}_{\hat{h}}(y)$ indicates if the data indeed satisfies the hypothesis. Since a wrong prediction for data (\mathbf{x}, y) results in $\sum_{\hat{h} \in h} |\hat{h}(\mathbf{x}) - \mathbf{1}_{\hat{h}}(y)| = 2$, we define the misclassification rate as

$$L_{\mathcal{D}}(h) = \frac{1}{2} (L_{\mathcal{D}}(h_t) + L_{\mathcal{D}}(h_f) + L_{\mathcal{D}}(h_{idk}))$$

Below is a theoretical analysis to understand the bound of the misclassification rate, which is driven by Theorem 2 of [30].

Theorem 3.1: Let $\mathcal{H} = \{\mathcal{H}_t, \mathcal{H}_f, \mathcal{H}_{idk}\}$ be a set of hypothesis spaces of VC dimension d . If $\mathcal{U}_S, \mathcal{U}_T$ are the samples of size m each, drawn from \mathcal{D}_S and \mathcal{D}_T , respectively, then for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for every $h \in \mathcal{H}$, we have

$$L_{\mathcal{D}_T}(h) \leq L_{\mathcal{D}_S}(h) + \frac{3}{4} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) + 6\sqrt{\frac{2d \log(2m) + \log(2/\delta)}{m}} + \frac{1}{2}\lambda \quad (3)$$

where $\lambda = \inf_{h \in \mathcal{H}} (L_{\mathcal{D}_S}(h) + L_{\mathcal{D}_T}(h))$ is the optimal combined error, $d_{\mathcal{H}\Delta\mathcal{H}}$ measures the distribution discrepancy between two distributions.

The above theorem provides insights for the generalization bound of the judge model. The bound is associated with the discrepancy between training data \mathcal{D}_S and evaluation data \mathcal{D}_T , and the discrepancy can be measured by drawing samples from both training and evaluation datasets for an equivalent size. Moreover, the bound is affected by the optimal hypothesis over all the data, i.e., $\mathcal{D}_S \cup \mathcal{D}_T$, where a lower error leads to improved performance of the judge model.

3.3 Evaluation

For evaluating the LLM’s performance, we consider *Correctness*, which is defined as the proportion of questions for which the LLM’s response matches the true label (or false label if the question is generated from a negative triple). This captures the accuracy of the LLM in identifying correct information and distinguishing it from fabricated (negative) triples. We also adopt the metrics of *Truthfulness* and *Informativeness*, as defined in [5]. *Truthfulness* refers to the likelihood of the language model (LLM) providing an honest response. A response is considered *Truthful* if the LLM either provides the correct answer or opts for ‘I don’t know’. This criterion assesses the model’s ability to be honest about what it knows and to admit uncertainty rather than making false statements. *Informativeness* is the probability of the LLM offering any substantive information, irrespective of its accuracy. An answer is deemed *Informative* if it is anything other than ‘I don’t know’. This reflects the model’s capacity to provide substantial information without resorting to uncertainty or avoidance of an answer.

When considering multiple negative triples sampled, we combine the results for all negative triples sampled from a triple τ , as well as the results for their original positive triple τ , to calculate the overall performance of the LLM. Since correctly detecting a real triple from KG is much simpler than detecting a negative triple, we want to give a max penalty to the LLM’s wrong response to the real triple when designing the metric. Therefore, if a real triple is predicted as false, the LLM will score 0 across all metrics. Then, the negative triple results are averaged to give a fine-grained evaluation of the LLM’s performance. To achieve this, for each performance metric, we define functions \mathcal{F} which evaluates the LLM’s response to τ and \mathcal{F}' to each negative triple τ' sampled from τ . The overall performance metric for τ is then calculated as:

$$\text{Metric}(\tau) = \max \left(0, \mathcal{F}(\tau) - \frac{1}{|\mathcal{N}(\tau)|} \sum_{\tau' \in \mathcal{N}(\tau)} \mathcal{F}'(\tau') \right) \tag{4}$$

Here, $\mathcal{N}(\tau)$ represents the set of all negative triples generated from the positive triple τ . \mathcal{F} and \mathcal{F}' are defined as follows: **(i) Correctness.** \mathcal{F} is defined such that it is 1 if the judge model predicts that a real (positive) triple is True, and it is 0 otherwise; \mathcal{F}' is 0 if the judge model predicts a negative triple as False, and 1 otherwise. **(ii) Truthfulness.** When measuring *Truthfulness*, \mathcal{F} is set to 1 if the judge model’s prediction for the input τ is either True or IDK, and it is 0 otherwise. Similarly, \mathcal{F}' is set to 1 if the judge model’s prediction for the input τ' is True, and 0 otherwise; and **(iii) Informativeness.** For *Informativeness*, \mathcal{F} is defined as 1 if the judge model’s prediction for the input τ is anything other than "I don’t know", and it is 0 otherwise. \mathcal{F}' is set to $1 - \mathcal{F}$ on the informativeness metric. By applying this equation, we can systematically compute the *Correctness*, *Truthfulness*, and *Informativeness* of an LLM’s responses in a consistent and comprehensive manner, offering a detailed insight into its overall performance.

#Entities	#Relations	#Triples	Avg. degree	Density
4,928,232	633	16,915,848	6.80	7.18×10^{-7}

Table 11: Statistics of the DBpedia knowledge graph.

4 Experiments

4.1 Experiment Setup

Data We use DBpedia [11], a large-scale knowledge graph constructed from Wikipedia. We report the statistics of the DBpedia knowledge graph in Table 11. Note that there are “dummy” entities in DBpedia that represent a fact that is only true on a specific time period. An example is https://dbpedia.org/page/Kathy_Greenlee_Tenure_1. For simplicity, we remove these dummy entities and triples related from the knowledge graph. We refer to the remaining triples as the DBpedia knowledge graph. The DBpedia knowledge graph contains 4,928,232 entities, 633 relations, and 16,915,848 triples. The average node degree of the knowledge graph is 6.80, and the density of the knowledge graph is 7.18×10^{-7} .

LLMs In this paper, we evaluate the Meta LLaMA 2 family [49], including LLaMA-2-7B, LLaMA-2-13B, and LLaMA-2-70B, and Google’s Gemma [44] including Gemma-2B and Gemma-7B. For each language model, we first randomly sample 2000 triples, and perform a negative sampling to obtain another 2000 negative triples. For each triple, we ask the LLM 3 times the same question, on whether the triple is true, false, or the LLM doesn’t know. We use majority voting to determine the LLM’s final answer. When asking, we use huggingface’s pipeline with default settings and FP16 precision. This is to form a labeled dataset. We randomly sample 70% for the training set and 30% for the validation set, then train a judge model to classify the LLM’s hidden state into 3 classes: LLM correctly answering the question (True), LLM incorrectly answering the question (False), and LLM responding with I don’t know (IDK). We refer to Table 12 for the statistics of the labeled dataset.

Metrics For the LLM’s performance, we report the estimated factuality of the LLMs on the DBpedia knowledge graph. We report the LLM’s performance in terms of *Truthfulness*, *Informativeness*, and *Correctness*. For evaluating the judge model’s performance (See Appendix A.1), we seek to maximize the similarity between the judge model’s prediction and the LLM’s answer. Thus, we use the common metrics Precision (P), Recall (R), and F1 score (F) to evaluate the judge model’s accuracy; and the time it takes to predict to evaluate the judge model’s efficiency.

Hyperparameter Settings For the judge model classifier training, we train 100 epochs with a batch size of 8. We use the Adam optimizer with a learning rate of 1e-4. We use the same settings for all the evaluated LLMs. For LLaMA 2 7B, 13B, and 70B, we use LLaMA 2 7B as the judge model’s hidden state input. For Gemma 2B and 7B, we use Gemma 2B as the judge model’s hidden state input. The judge model is trained on a server with NVIDIA A6000 GPUs.

For the training of the prompt encoder, we use the same settings for all the evaluated LLMs. To be specific, we use 20 virtual tokens, 1 transformer submodule, 12 attention heads, 12 layers, MLP as the encoder reparameterization type, 4096 as the encoder hidden size, and 2e-5 as the learning rate. We train the prompt encoder for 5 epochs with a batch size of 8. We use the Adam optimizer with a weight decay of 0.01.

For the evaluation, we use two servers, one with NVIDIA A6000 GPUs and the other with NVIDIA A100 GPUs. For inference, we use Flash Attention 2 [10] as the attention implementation, and use FP16 precision.

Model	True	False	IDK	Truthful	Informative	Correct
LLaMA-2-7B	1901	1545	554	0.965	0.550	0.516
LLaMA-2-13B	2100	1796	104	0.979	0.980	0.959
LLaMA-2-70B	338	126	3536	0.993	0.007	0.006
Gemma-2B	1760	1786	454	0.056	0.867	0.024
Gemma-7B	1509	1751	740	0.206	0.657	0.056

Table 12: Statistics and performance metrics of LLMs. True, False, and IDK denote the number of labels from the LLMs in the labeled dataset. *Truthful*, *Informative*, and *Correct* represent performance metrics.

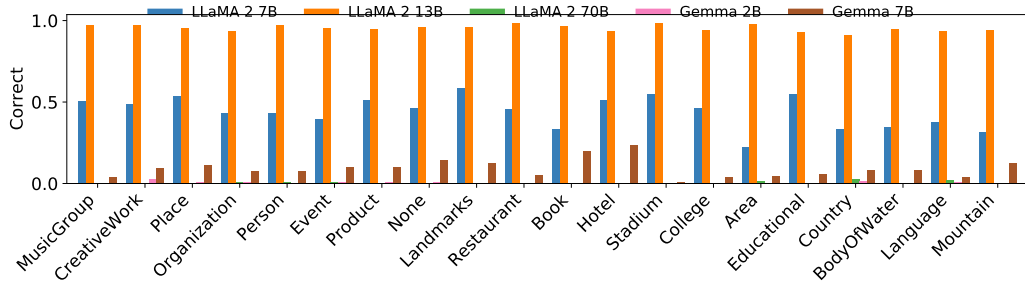
4.2 LLM’s Performance Analysis

We report the estimated factuality of the LLMs on DBpedia in Table 12. Overall, the LLaMA-2 series shows an increase in model size up to 13B, particularly in terms of balanced *truthfulness*, *informativeness*, and *correctness*. However, the 70B variant diverges, excelling in *truthfulness* but failing to provide useful or accurate information. We will discuss this phenomenon in the detailed LLaMA analysis. The Gemma series struggles with *truthfulness* and *correctness*, despite being *informative*. This might indicate that these models are better at generating detailed content but need careful consideration for tasks requiring high accuracy or reliability. The performance of these models highlights the complex trade-offs between being *truthful*, *informative*, and *correct*. We further provide a correlation analysis between the LLM’s performance and the degree/popularity of the entities in the Appendix A.3.

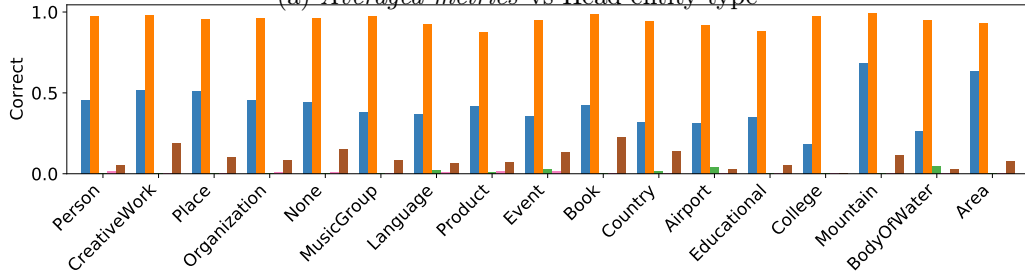
LLaMA-2 Analysis LLaMA-2-7B shows good *truthfulness* (.965) but is moderate in being *informative* (.550) and *correct* (.516). This suggests that while the model is generally reliable in its outputs, it may not always provide highly detailed or accurate information. LLaMA-2-13B significantly improves across all metrics compared to LLaMA-2-7B, with very high scores in *truthfulness* (.979), *informativeness* (.980), and *correctness* (.959). This indicates a strong overall performance, making it a very reliable and accurate model for generating information.

LLaMA-2-70B, despite its high *truthfulness* (.993), scores extremely low in both *informativeness* (.007) and *correctness* (.006), which is puzzling. We hypothesize that the model may have difficulty in making a decision, and thus selecting ‘I don’t know’ as the answer. This may be related to a more clear knowledge boundary of LLMs, as larger LMs tend to give up on more questions [48], meaning they have a better understanding on whether they know the answer or not. A detailed analysis of the knowledge boundary of LLMs can be found in Appendix A.4. This can also be confirmed by the fact that the model has the highest *truthfulness* score among all models, indicating that it is more likely to provide a correct answer when it knows the answer. However, it is still important to note that a high number of ‘I don’t know’ answers may indicate the model’s inability to answer factual questions.

Gemma Analysis Gemma-2B has an exceptionally low *truthfulness* score (.056) but is quite high in *informativeness* (.867). Its *correctness* score (.024) is also very low. This suggests that despite providing detailed responses, the model’s outputs are often neither *truthful* nor accurate. It might be generating detailed but misleading or incorrect information. Gemma-7B improves on *truthfulness* (.206) compared to Gemma-2B but still falls short of being considered reliable. Its *informativeness* (.657) is respectable, and its *correctness* (.056) remains low. Similar to Gemma-2B, while it can provide detailed responses, those are not often true or correct.



(a) Averaged metrics vs Head entity type



(b) Averaged metrics vs Tail entity type

Figure 25: The LLM’s *averaged metrics* with respect to head entity types and tail entity types

4.3 Relation Type Study

There are more than 600 different relation types in the DBpedia knowledge graph, and each relation type has different characteristics. It is unclear if we directly compare the performance of the LLMs on different relation types. Thus, to gain a better understanding of the LLM’s performance, we first analyze the LLM’s performance with respect to relation types. In DBpedia, most entities are associated with a <https://schema.org/> type. Thus, we can categorize the relations into different types by the triples they belong to. We denote a relation’s head/tail entity type as the most frequent schema type of the head/tail entity of the triples associated with the relation. For example, the relation **birthPlace** is associated with triples like (**Barack Obama**, **birthPlace**, **Hawaii**), and the head entity **Barack Obama** is associated with the schema type **Person**, and the tail entity **Hawaii** is associated with the schema type **Place**. Then, the relation’s head entity type is **Person**, and tail entity type is **Place**. We then analyze the LLM’s performance with respect to these relation types. We report the performance of the LLMs on different relation types, by taking the average of the 3 metrics, *correctness*, *truthfulness*, and *informativeness*, for each relation type. We present the results in Figure 25. Here, “None” refers to entities not linked to a schema type. We also present a detailed analysis of the LLM’s performance with respect to head and tail entity types in Appendix A.2. We can observe variability in model performance across relation types, such as “MusicGroup” and “CreativeWork” achieving high scores while “Area” and “Mountain” face lower performance, highlighting the diverse challenges in modeling different kinds of information. These performance differences suggest that the effectiveness of LLMs in handling structured knowledge heavily depends on the nature of the relations being modeled.

5 Conclusions

We introduce **GraphEval**, an innovative approach for appraising the efficacy of LLMs against a voluminous test dataset derived from an extensive knowledge graph containing over 10 million facts, significantly mitigating the necessity for costly human intervention. **GraphEval**, by embedding a judge module within

the LLM itself, not only refines the evaluation process but also establishes a new benchmark for assessing the veracity of the information presented by these models. The empirical evidence from our experiments substantiates the judge model’s proficiency in fact-checking, exhibiting a high degree of concordance with the accuracy of the LLM’s outputs, and simultaneously diminishing the resources required for evaluation. The insights gleaned from our study shed light on the multifaceted performance of LLMs and lay the groundwork for future endeavors aimed at enhancing the reliability of their generated content. Moreover, we consider extending this work to cross-lingual KGs to evaluate the performance of various LLMs in different languages.

References

- [1] Sun, K. et al.. Head-to-tail: How knowledgeable are large language models (llm)? AKA will llms replace knowledge graphs?. [arXiv preprint arXiv:2308.10168](#), , 2023.
- [2] Liang, P. et al.. Holistic evaluation of language models. [arXiv preprint arXiv:2211.09110](#), , 2022.
- [3] Lu, P. et al.. Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering. , 2022.
- [4] Yao, Y. et al.. Editing large language models: Problems, methods, and opportunities. [arXiv preprint arXiv:2305.13172](#), , 2023.
- [5] Lin, S. et al.. TruthfulQA: Measuring How Models Mimic Human Falsehoods. :3214–3252, 2022.
- [6] Weller, O. et al.. " According to..." Prompting Language Models Improves Quoting from Pre-Training Data. [arXiv preprint arXiv:2305.13252](#), , 2023.
- [7] Elliot Bolton, et al.. BioMedLM: A 2.7B Parameter Language Model Trained On Biomedical Text. , 2024.
- [8] Yasunoga, M. et al.. Deep bidirectional language-knowledge graph pretraining. [Advances in Neural Information Processing Systems](#), 35:37309–37323, 2022.
- [9] Jia, Z. et al.. Tempquestions: A benchmark for temporal question answering. :1057–1062, 2018.
- [10] Dao, T.. Flashattention-2: Faster attention with better parallelism and work partitioning. [arXiv preprint arXiv:2307.08691](#), , 2023.
- [11] Auer, S. et al.. Dbpedia: A nucleus for a web of open data. :722–735, 2007.
- [12] Haoyu Wang, et al.. BlendFilter: Advancing Retrieval-Augmented Large Language Models via Query Generation Blending and Knowledge Filtering. , 2024.
- [13] Shizhe Diao, et al.. Mixture-of-Domain-Adapters: Decoupling and Injecting Domain Knowledge to Pre-trained Language Models Memories. , 2023.
- [14] Lewis, P. et al.. Retrieval-augmented generation for knowledge-intensive nlp tasks. [Advances in Neural Information Processing Systems](#), 33:9459–9474, 2020.
- [15] Azaria, A., Mitchell, T.. The Internal State of an LLM Knows When It’s Lying. :967–976, 2023.
- [16] Gallegos, I.O. et al.. Bias and Fairness in Large Language Models: A Survey. [arXiv preprint arXiv:2309.00770](#), , 2023.
- [17] Kotha, S. et al.. Understanding catastrophic forgetting in language models via implicit inference. [arXiv preprint arXiv:2309.10105](#), , 2023.
- [18] Liu, X. et al.. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. [arXiv preprint arXiv:2110.07602](#), , 2021.
- [19] Wang, C. et al.. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. [arXiv preprint arXiv:2310.07521](#), , 2023.
- [20] Carlson, A. et al.. Toward an architecture for never-ending language learning. 24:1306–1313, 2010.
- [21] Kwiatakowski, T. et al.. Natural Questions: A Benchmark for Question Answering Research. [Transactions of the Association for Computational Linguistics](#), 7:453–466, 2019. MIT Press-Journals.
- [22] Feng, S. et al.. FactKB: Generalizable Factuality Evaluation using Language Models Enhanced with Factual Knowledge. :933–952, 2023.
- [23] Kim, J. et al.. KG-GPT: A general framework for reasoning on knowledge graphs using large language models. [arXiv preprint arXiv:2310.11220](#), , 2023.

- [24] Tian, K. et al.. Fine-tuning language models for factuality. [arXiv preprint arXiv:2311.08401](#), , 2023.
- [25] Luo, L. et al.. Reasoning on graphs: Faithful and interpretable large language model reasoning. [arXiv preprint arXiv:2310.01061](#), , 2023.
- [26] Shiqi Chen, et al.. FELM: Benchmarking Factuality Evaluation of Large Language Models. , 2023.
- [27] Berglund, L. et al.. The Reversal Curse: LLMs trained on " A is B" fail to learn " B is A". [arXiv preprint arXiv:2309.12288](#), , 2023.
- [28] Bollacker, K. et al.. Freebase: a collaboratively created graph database for structuring human knowledge. :1247–1250, 2008.
- [29] Wang, X. et al.. " My Answer is C": First-Token Probabilities Do Not Match Text Answers in Instruction-Tuned Language Models. [arXiv preprint arXiv:2402.14499](#), , 2024.
- [30] Ben-David, S. et al.. A theory of learning from different domains. [Machine learning](#), 79:151–175, 2010. Springer.
- [31] Wang, Y. et al.. Preserving In-Context Learning ability in Large Language Model Fine-tuning. [arXiv preprint arXiv:2211.00635](#), , 2022.
- [32] Liu, X. et al.. P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks. :61–68, 2022.
- [33] Jiang, J. et al.. ReasoningLM: Enabling Structural Subgraph Reasoning in Pre-trained Language Models for Question Answering over Knowledge Graph. :3721–3735, 2023.
- [34] Chang, Y. et al.. A survey on evaluation of large language models. [ACM Transactions on Intelligent Systems and Technology](#), , 2023. ACM New York, NY.
- [35] Cunxiang Wang, et al.. Evaluating Open-QA Evaluation. , 2023.
- [36] Goodfellow, I.J. et al.. An empirical investigation of catastrophic forgetting in gradient-based neural networks. [arXiv preprint arXiv:1312.6211](#), , 2013.
- [37] Zhai, Y. et al.. Investigating the Catastrophic Forgetting in Multimodal Large Language Models. [arXiv preprint arXiv:2309.10313](#), , 2023.
- [38] Zhou, K. et al.. Don't Make Your LLM an Evaluation Benchmark Cheater. [arXiv preprint arXiv:2311.01964](#), , 2023.
- [39] Zhang, Y. et al.. Siren's song in the AI ocean: a survey on hallucination in large language models. [arXiv preprint arXiv:2309.01219](#), , 2023.
- [40] Joshi, M. et al.. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. :1601–1611, 2017.
- [41] Suchanek, F.M. et al.. Yago: a core of semantic knowledge. :697–706, 2007.
- [42] Dan Hendrycks, et al.. Measuring Massive Multitask Language Understanding. [Proceedings of the International Conference on Learning Representations \(ICLR\)](#), , 2021.
- [43] Bordes, A. et al.. Translating embeddings for modeling multi-relational data. [Advances in neural information processing systems](#), 26, 2013.
- [44] Team, G. et al.. Gemma: Open models based on gemini research and technology. [arXiv preprint arXiv:2403.08295](#), , 2024.
- [45] Zhang, M. et al.. Knowledge Graph Enhanced Large Language Model Editing. [arXiv preprint arXiv:2402.13593](#), , 2024.
- [46] Huang, Y. et al.. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. [Advances in Neural Information Processing Systems](#), 36, 2024.
- [47] Liu, A. et al.. We're Afraid Language Models Aren't Modeling Ambiguity. [arXiv preprint arXiv:2304.14399](#), , 2023.
- [48] Ren, R. et al.. Investigating the factual knowledge boundary of large language models with retrieval augmentation. [arXiv preprint arXiv:2307.11019](#), , 2023.
- [49] Touvron, H. et al.. Llama 2: Open foundation and fine-tuned chat models. [arXiv preprint arXiv:2307.09288](#), , 2023.
- [50] Chen, S. et al.. Recall and Learn: Fine-tuning Deep Pretrained Language Models with Less Forgetting. :7870–7881, 2020.
- [51] Tan, Y. et al.. Can ChatGPT replace traditional KBQA models? An in-depth analysis of the question answering performance of the GPT LLM family. :348–367, 2023.

- [52] Talmor, A., Herzig, J., Lourie, N., and Berant, J. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. arXiv preprint arXiv:1811.00937, 2019. <https://arxiv.org/abs/1811.00937>.

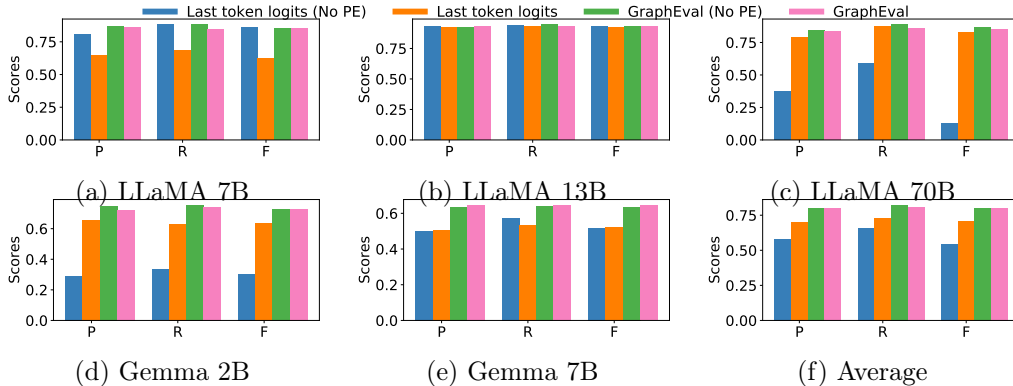


Figure 26: Evaluation scores on the judge model’s performance on the labeled validation set. P, R, and F are Precision, Recall, and F1 Score.

Substitute Model	LLaMA 2 7B			LLaMA 2 13B			LLaMA 2 70B		
	P	R	F	P	R	F	P	R	F
LLaMA 2 7B	.858	.845	.850	.928	.934	.930	.837	.861	.848
LLaMA 2 13B	.850	.868	.855	.930	.940	.932	.837	.851	.844
LLaMA 2 70B	.868	.883	.871	.924	.942	.931	.858	.876	.866

Table 13: Ablation on the LLaMA models as substitute models. The i -th row and j -th column denote the result of using i -th LLM as the substitute hidden state input for training on j -th model’s labels. P, R, and F are Precision, Recall, and F1 Score.

A Appendix

A.1 Judge Model Analysis

We analyze the judge model’s performance on the labeled validation set. We compare **GraphEval**’s judge model by using the last token logit as the judge model. This is a common practice in evaluating LLMs, as the last token logit is the most common way to extract the hidden state of the LLMs. We also analyze the judge model with or without the prompt encoder (PE), as it may have a negative impact on the judge model’s performance. We refer to Figure 26 for the judge model’s performance on the labeled validation set.

Accuracy Analysis The **GraphEval** model, both with and without Prompt Encoder (PE), consistently outperforms the score of using Last token logits in almost all configurations and metrics. This indicates the effectiveness of the **GraphEval** approach in capturing the nuances of the evaluation task.

Ablation Study *On Prompt Encoder:* As Figure 26 shows, the comparison between models with and without PE indicates a slight performance variation. For **GraphEval**, the presence of PE does not significantly alter the performance, suggesting that our method of evaluating LLMs is robust to the inclusion or exclusion of PE. For the Last token logits method, removing PE generally results in a perturbation in performance. However, the **GraphEval** approach’s consistency suggests a potentially different or more advanced mechanism of evaluation that is less dependent on PE. *On Substitute Models:* We also evaluate the judge model’s performance on different LLMs as hidden state input. We refer to Table 13 for the judge model’s performance on different LLMs as hidden state input. We can see that,

Models	LLaMA 2 7B		LLaMA 2 13B		LLaMA 2 70B		Gemma 2B		Gemma7B	
	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs	Speed	#GPUs
TG (A6000)	2.26	1	1.07	2	0.09	4	2.06 (1.82)	1	2.18 (1.28)	1
GraphEval (A6000)	121.34	1	120.10	1	117.90	1	388.61	1	389.04	1
TG (A100)	2.80	1	1.48	1	0.21	2	2.47	1	2.42	1
GraphEval (A100)	210.59	1	213.05	1	210.30	1	731.98	1	735.62	1

Table 14: Efficiency evaluation. Speed denotes the average number of triple facts on which a conclusion can be given in one second. #GPUs denotes the least number of GPUs to run without OOM. TG denotes text generation. The numbers in parentheses are the speed without Flash Attention 2.

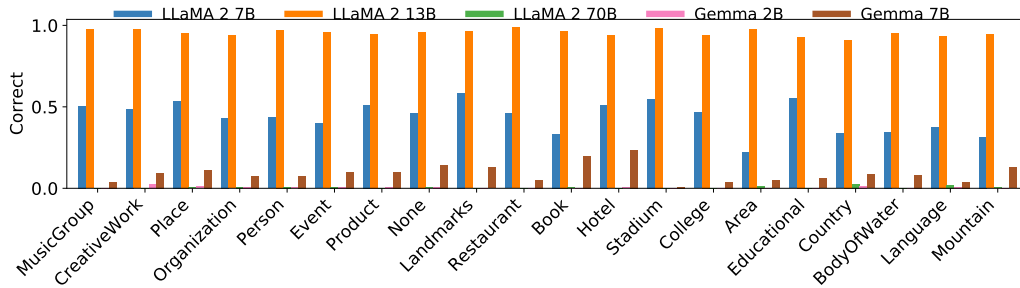
generally, when larger models are applied for feeding the hidden states, there is a slight increase in the fitting accuracy of the judge model. However, there is no significant difference in the judge model’s performance.

Efficiency study We also analyze the judge model’s efficiency by measuring the time it takes to make a prediction on one triple. The speed of text generation refers to the average rate at which the LLM completes generating a response consisting of one sentence derived from a triple. It’s important to recognize that the pace of text generation can vary with different prompts because the LLM may produce responses of varying lengths. Therefore, for a more consistent measure of text generation speeds, it’s advisable to consider the rate of token generation. Despite this, our evaluation framework, **GraphEval**, does not depend on text generation and operates on a triple-based unit. Consequently, we continue to use the triple as the unit of measurement for time. We use the same hardware and software environment for all the experiments. We compare the average speed of the judge model with text generation. We report the time it takes to make a prediction in Table 14. The attention implementation and precision are the same for text generation and for the judge model’s input model. We can see that the judge model is significantly faster than text generation. This indicates that the judge model is efficient in evaluating the LLMs. Also, benefiting from the substitute model, our evaluation speed and GPU requirement does not grow with the LLM size, which is an advantage for evaluating large LLMs. We also observe that, paradoxically, the Gemma 2B model operates slower than the Gemma 7B model, despite its smaller size. This counterintuitive result could be attributed to the implementation of Flash Attention 2. To draw a fair comparison, we documented the text generation speed on A6000 GPUs excluding Flash Attention 2, which is indicated within parentheses. The comparative data reveals that Gemma 2B is faster than Gemma 7B when Flash Attention 2 is not utilized. Notwithstanding this, Gemma 2B demonstrates enhanced performance when Flash Attention 2 is active. Therefore, for the sake of consistency, we have decided to maintain the results acquired with Flash Attention 2.

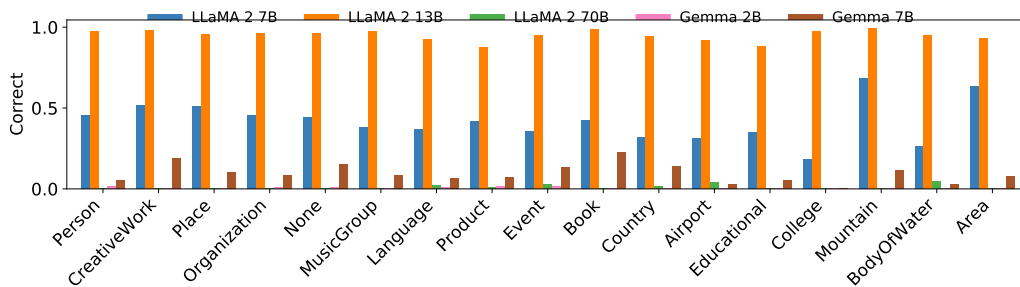
A.2 Detailed Relation Type Analysis

Llama Family Analysis Across the LLaMA family, a progressive improvement in performance is observed from 7b to 13b. The 7b model shows decent performance across categories with a particular strength in the *truthfulness*. However, its *informativeness* and *correctness* metrics show room for improvement, particularly in categories like Book, Hotel, and College, indicating a struggle to accurately provide informative and correct classifications in more nuanced or specific domains.

The LLaMA 13b model demonstrates a significant leap in performance, especially in *informativeness* and *correctness*, nearly reaching perfection across most categories. This jump can be attributed to the model’s increased capacity, enabling it to understand and process the nuances of various entities



(a) *Correctness* vs Head entity type



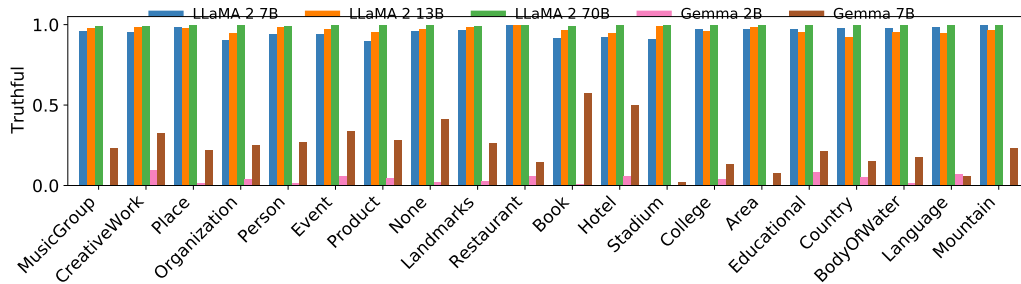
(b) *Correctness* vs Tail entity type

Figure 27: The LLM’s *correctness* with respect to head entity types and tail entity types

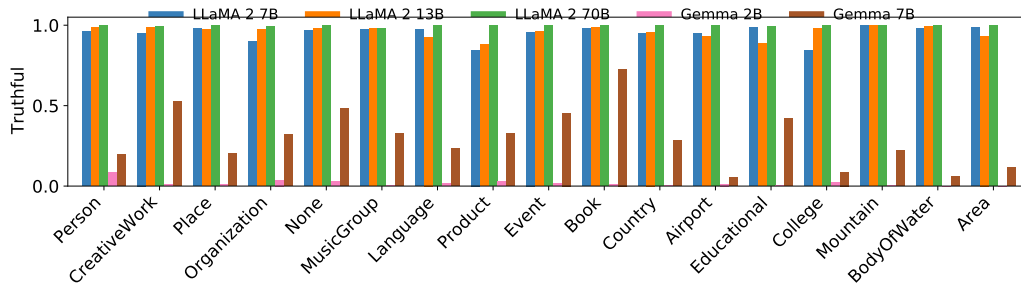
better, resulting in remarkably high scores in nearly all categories, especially noticeable in MusicGroup, CreativeWork, and Place.

The LLaMA 70b results appear anomalous with extremely high *truthfulness* scores but negligible *informativeness* and *correctness* across all categories. We suspect this discrepancy might be due to the model’s knowledge awareness [48], where the model might be less confident in its responses when the parameters are increased, leading to a higher proportion of “I don’t know” responses. This could explain the high *truthfulness* scores but low *informativeness* and *correctness* metrics, as the model might be too cautious to provide definitive answers.

Gemma Family Analysis The Gemma models present an interesting contrast. The Gemma 2b model shows a tendency towards high *informativeness* in certain categories like MusicGroup and Book but lacks behind significantly in *truthfulness* and *correctness* metrics. This suggests that while the model might be picking up on relevant information, it struggles to accurately validate the truth behind that information or its applicability to the queried entities. The Gemma 7b model shows improvement in the *truthfulness* metric compared to Gemma 2b, particularly noticeable in categories like Book and Hotel, and even surpasses LLaMA 7b in certain areas like None and Restaurant. However, it still significantly lags behind the LLaMA models, particularly LLaMA 13b, in both *informativeness* and *correctness*. The improved but still limited performance suggests that while Gemma 7b has a better grasp over the veracity of information compared to Gemma 2b, it still struggles with providing highly informative and correct outputs consistently across various entities.



(a) *Truthfulness* vs Head entity type



(b) *Truthfulness* vs Tail entity type

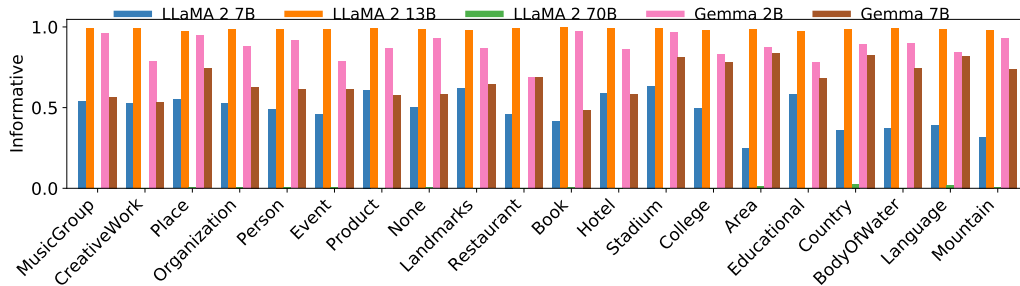
Figure 28: The LLM’s *truthfulness* with respect to head entity types and tail entity types

A.3 Correlation Analysis

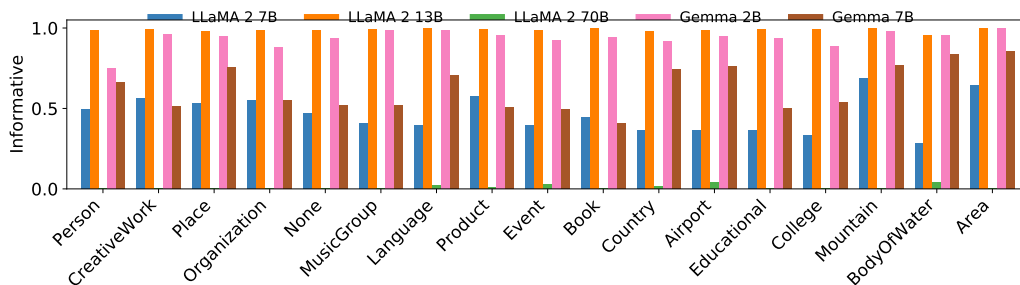
As the current language models are all exposed to Wikipedia knowledge during training, we are interested in how the LLM performance is correlated with the attributes of the triples in the knowledge graphs. As an example, if an entity has a higher degree, it may be linked to more documents, and the LLM may have more chances to learn about the entity during training. Another example is the popularity of the entity. If the entity is more popular, it may be linked to more external documents because it summarizes the relevant knowledge and provides high-level ideas to the general public, and the LLM may have more chances to learn about the entity during training. This raises the question of whether the LLM’s performance is correlated with the attributes of the triples in the knowledge graphs. For the entities in a knowledge graph, the degree of an entity is the number of edges connected to the entity. We also collect the *pageviews* of the entities in the knowledge graph from Wikimedia¹, which is the number of pageviews of the Wikipedia page of the entity. This can be seen as a measure of the popularity of the entity because a popular page should appeal to the significant attention of the readers. We collect the pageviews, in the time period of the entities in the knowledge graph from the Wikipedia page of the entity. After collecting the degree and pageviews of the entities in the knowledge graph, we can aggregate the degree and pageviews of the entities to the triples, by simply taking the average of the degree and pageviews of the head and tail entities of the triples.

Here, we analyze whether the LLM’s performance is correlated with the attributes of the triples in the knowledge graphs, such as the entity’s degree, and page views. We refer to Figure 30 for the correlation heatmap of the LLMs’ hidden states and the judge model’s predictions. Here, ‘T’ stands for *Truthful*, ‘I’ stands for *Informativeness*, ‘C’ stands for *Correctness*, ‘P’ stands for Pageviews, and ‘D’ stands for Degree. We can see that the LLM’s performance does not show a strong correlation with the attributes of

¹https://wikimedia.org/api/rest_v1/metrics/pageviews/per-article/en.wikipedia.org/all-access/all-agents



(a) *Informativeness* vs Head entity type



(b) *Informativeness* vs Tail entity type

Figure 29: The LLM’s *informativeness* with respect to head entity types and tail entity types

Models	Llama 3 1B	Llama 3 3B	Llama 3 8B	Llama 3 70B
CommonsenseQA	54.65	37.73	30.36	22.28
TruthfulQA	58.35	52.32	49.03	23.45

Table 15: Knowledge boundary analysis results for the Llama 3 series models on CommonsenseQA and TruthfulQA datasets. The table reports the Expected Calibration Error (ECE) values (%) measuring the alignment between model confidence and correctness. Lower is better. The best results are in bold.

the triples in the knowledge graphs. This indicates that the LLM’s performance is not directly correlated with the attributes of the triples in the knowledge graphs. However, the different metrics of LLMs may correlate with each other, such as *Truthful* and *Informativeness*, which is expected. This can be explained by the fact that certain attributes, like the degree of an entity in the knowledge graph, can be misleading. For example, degree is often correlated with popularity, but the popularity metric is 0 for many entities, particularly those in the long tail. This uneven distribution limits the usefulness of popularity as a reliable metric for evaluating LLM performance. In other words, while high-degree or popular entities may influence LLM performance to some extent, the vast majority of entities are long-tail, and their sparse or zero popularity values do not strongly correlate with performance outcomes. This highlights the need for more nuanced or domain-specific metrics to assess LLM performance effectively.

A.4 Knowledge Boundary Analysis

We analyze the knowledge boundaries of large language models (LLMs), as discussed in Section 4.2 and [48], which suggest that larger models have a better understanding whether they know an answer or not. To investigate this hypothesis, we conduct experiments using the Llama 3 model series on two

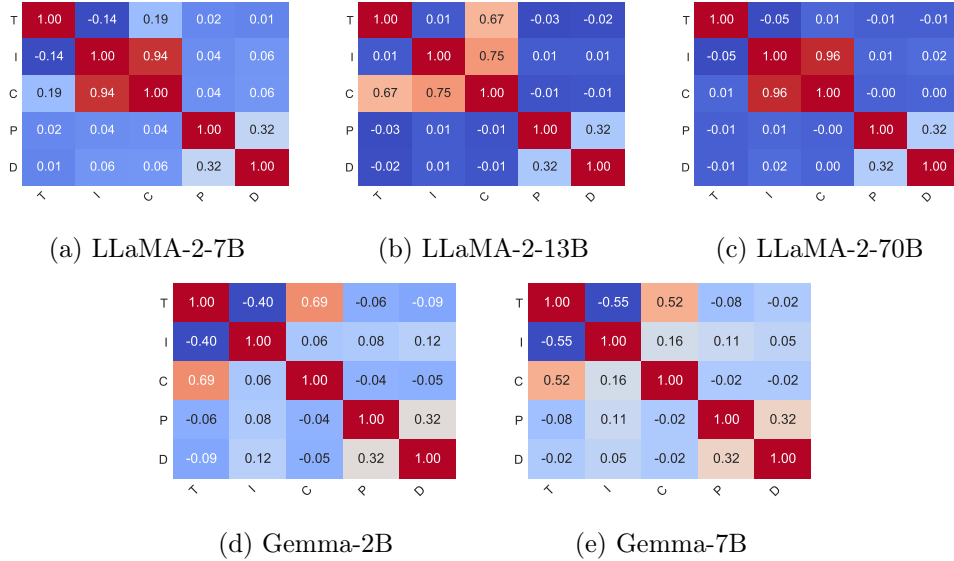


Figure 30: Correlation heatmap of the LLMs’ hidden states and the judge model’s predictions.

question-answering datasets: CommonsenseQA [52] and TruthfulQA [5]. To assess whether an LLM understands its own knowledge boundaries, we directly elicit confidence scores for each answer through prompting, then we calculate the Expected Calibration Error (ECE). ECE measures the misalignment between the correctness of answers and the models’ confidence. Mathematically, for LLM responses \mathcal{A} , ECE is defined as:

$$\text{ECE} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} |\mathbb{I}(a) - \text{conf}(a)|, \quad (5)$$

where $\mathbb{I}(a)$ is an indicator function that outputs 1 if a is correct and 0 otherwise, and $\text{conf}(a)$ denotes the confidence score assigned by the model. The experimental results are presented in Table 15.

The results in Table 15 align with our previous hypothesis across different Llama 3 model sizes. For both CommonsenseQA and TruthfulQA, the ECE values decrease as model size increases, indicating better alignment between confidence and correctness in larger models. Specifically, for CommonsenseQA, the Llama 3 70B model achieves the lowest ECE (22.28%), demonstrating superior calibration compared to smaller models like Llama 3 1B (54.65%). Similarly, on TruthfulQA, the Llama 3 70B model achieves an ECE of 23.45%, significantly outperforming the smaller Llama 3 1B model with an ECE of 58.35%.

These findings align with the hypothesis that larger models are better calibrated in estimating their confidence, which can partially explain why larger models are more likely to answer “I don’t know” when asked about a question as shown in Section 4.2.

Overall, GraphEval aligns well with the results on TruthfulQA and CommonsenseQA, demonstrating that it effectively captures the model’s factuality and informativeness across diverse knowledge domains. This alignment validates the robustness of our evaluation framework and confirms its consistency with established benchmarks for assessing LLM reasoning and truthfulness.

A.5 Detailed Settings

Relation templates We use the relation templates to create queries for evaluating the models. These templates are first generated by GPT with Web API in a few-shot manner, then manually curated to ensure the quality of the templates. We refer to Figure 31 for the prompt used for generating the

You are given a few samples of a relation in the format of <head, relation, tail>. You need to write a statement template about the relation, which will be used to generate a statement for a given head entity and the tail entity. I will give you 3 examples as below.

relation: "education"
triple samples:
<Mamphono Khaketla, education, National University of Lesotho>
...
<A.D. Frazier, education, University of North Carolina at Chapel Hill>

statement template: "{head} was educated at {tail}."

relation: "channel"
triple samples:
<Way Out, channel, CBS>
...
<19+, channel, TVN (Poland)>

statement template: "{head} is broadcasted on {tail}."

relation "curator"
triple samples:
<Alicante Museum of Contemporary Art, curator, Alicante>
...
<Baturyn Museum of Archeology, curator, Hetman's Capital>

statement template: "{head} is curated by {tail}."

according to the above 3 examples, please write a statement template for the following relation:

relation: {relation}
triple samples:
{triples}

Figure 31: The prompt to generate the relation template.

relation templates. The prompt is designed to ask the model to generate a query for a given relation type. The model is asked to generate a query that can be used to judge the factuality of the relation type. We then manually curate the generated templates to ensure the quality of the templates. We refer to Table 16 for the curated relation templates. Due to the large number of relation types in the DBpedia knowledge graph, we only showcase a few relation templates in the table, these templates are the most common relation types in the knowledge graph, sorted by the number of triples associated with the relation type. We can see that the relation templates are comprehensive and cover a wide range of topics. This can be seen as a source of multiple-domain knowledge for evaluating the LLMs.

Data and Model We download the DBpedia data dump from <https://www.dbpedia.org/>. We use the turtle format of the DBpedia knowledge graph. We directly use the LLaMA 2 and Gemma from the Hugging Face model hub. The model cards are meta-llama/Llama-2-7b-chat-hf, meta-llama/Llama-2-13b-chat-hf, meta-llama/Llama-2-70b-chat-hf, gemma-team/gemma-2b-chat-hf, and gemma-team/gemma-7b-chat-hf.

Instruction used for the LLaMA and Gemma models We report the instructions used for creating queries for the LLaMA and Gemma models. The instruction is designed to ask the model to

Relation	Template	Count
birthPlace	The birthplace of {head} is {tail}.	1,465,157
team	{head} is a part of the {tail} team.	1,265,483
subdivision	The subdivision of {head} is {tail}.	1,070,387
country	{head} is from the country {tail}.	766,844
starring	{head} is a character in a movie or play {tail}".	540,937
location	The location of {head} is {tail}.	523,283
type	The type of {head} is {tail}.	480,274
deathPlace	{head} passed away in {tail}.	435,869
timeZone	The time zone of {head} is {tail}.	433,915
genre	The genre of {head} is {tail}.	415,336
homepage	The homepage of {head} is {tail}.	366,745
position	The position of {head} is {tail}.	319,196
seeAlso	The related item to {head} under the ↔label 'seeAlso' is {tail}.	296,615
writer	The writer of {head} is {tail}.	249,017
almaMater	The alma mater of {head} is {tail}.	217,533
occupation	The occupation of {head} is {tail}.	200,615
award	The award won by {head} is {tail}.	181,521
recordLabel	The record label associated with {head} is {tail}.	178,657
party	The party that {head} is affiliated with is {tail}.	170,931
producer	The producer of {head} is {tail}.	169,628
formerTeam	{head} used to play for {tail} team.	151,374
family	{head} belongs to the {tail} family.	148,818
currentMember	The current member of {head} is {tail}.	148,739
battle	{head} participated in the following battles: {tail}.	148,188
nationality	The nationality of head is tail.	147,525
director	The director of {head} is {tail}.	145,621
associatedBand	The band associated with {head} is {tail}.	135,597
associatedMusical ↔ Artist	The musical artist associated with {head} ↔ in the music industry is {tail}.	135,582
class	The class of {head} is {tail}.	127,837
order	The order of {head} is {tail}.	123,626

Table 16: Relations templates.

judge whether the statement is true or false. We refer to Table 17 for the instruction used for creating queries. We use the same instruction for both the LLaMA and Gemma models with little modification to adjust the model’s instruction format. With this instruction, the most frequent responses of LLMs are *Yes, the statement is true*, *No, the statement is false*, and *I don’t know*, with some variations on the suffix, mainly explaining the reason for the answer. This is what we expect from the LLMs when using a judge model (or the first-token logit as well), since the judge model doesn’t use the LLM’s response, but the hidden state of the LLM, which makes the consistency of the response format important.

A.6 Language Setting

As a framework, GraphEval is not constrained by language, as long as the input is in the form of a knowledge graph. However, we did not conduct experiments on multilingual or cross-lingual datasets in the current work. Current experiments are conducted on English knowledge graphs.

Model	Instruction
LLaMA 2	Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\n You are given a statement. You are asked to judge whether the statement is true or false. Answer 'Yes, the statement is true.' if you know the statement is true. Answer 'No, the statement is false.' if you know the statement is false. Otherwise, answer 'I don't know.'\n\n### Input: Input \n\n### Response:\n\n
Gemma	start_of_turn>user Below is an instruction that describes a task, paired with an input that provides further context. Write a response that appropriately completes the request.\n\n### Instruction:\n You are given a statement. You are asked to judge whether the statement is true or false. Answer 'Yes, the statement is true.' if you know the statement is true. Answer 'No, the statement is false.' if you know the statement is false. Otherwise, answer 'I don't know.'\n\n### Input: Input <end_of_turn><start_of_turn>model\n\n The answer is "

Table 17: Instruction used for creating queries.

Increasing Accuracy of LLM-powered Question Answering on SQL databases: Knowledge Graphs to the Rescue

Juan Sequeda, Dean Allemang, Bryon Jacob
data.world AI Lab
{juan,dean.allemang,bryon}@data.world

Abstract

Large Language Models (LLMs) are promising technology to support Question Answering on enterprise SQL data (i.e. Text-to-SQL). Knowledge Graphs are also promising technology to enhance LLM-based question answering by providing business context that LLMs lack. However, it is not well understood to what extent Knowledge Graphs can increase the accuracy of LLM-powered question answering system on SQL databases. Our research aims to understand and quantify this extent. First, we introduce a benchmark comprising an enterprise SQL schema in the insurance domain, a range of enterprise queries encompassing reporting to metrics, and a contextual layer consisting of an ontology and mappings that define a Knowledge Graph. The experimental reveals that question answering using GPT-4, with zero-shot prompts directly on SQL databases, achieves an accuracy of 16%. Notably, this accuracy increases to 54% when questions are posed over a Knowledge Graph representation of the enterprise SQL database. Second, we present an approach that leverages the ontology of the Knowledge Graph to deterministically detect incorrect queries generated by the LLM and repair them. Experimental results show that the accuracy increases to 72.55%, including an additional 8% of "I don't know" unknown results. Thus, the overall error rate is 20%. The conclusion is that investing in Knowledge Graph provides higher accuracy for LLM powered question answering systems on SQL databases.

A Introduction

Business users and executives would like to have an AI assistant that understands their business, available to them at all times, in order to ask questions and receive accurate, explainable and governed answers. This challenge, known as Question Answering, which is the ability to interact with data using natural language questions and obtaining accurate results, has been a long-standing challenge in computer science dating back to the 1960s [12–14, 28]. The field has advanced throughout the past decades [6, 27, 31], through Text-to-SQL approaches, as a means of facilitating chatting with the data that is stored in SQL databases[9, 17, 21, 24, 29, 33]. With the rise of Generative AI and Large Language Models (LLMs) in early 2023, the interest increased dramatically. These question answering systems hold tremendous potential for transforming the way data-driven decision making is executed within enterprises.

Knowledge Graphs (KGs) have been identified as a promising solution to fill the business context gaps in order to reduce hallucinations, thus enhancing the accuracy of LLMs. The effective integration of LLMs and KGs started to gaining traction in academia in the past several years¹[23]. From an industry perspective, Gartner stated in July 2023 that, "*Knowledge graphs provide the perfect complement to*

¹<https://github.com/RManLuo/Awesome-LLM-KG>

LLM-based solutions where high thresholds of accuracy and correctness need to be attained."².

Our hypothesis is that Knowledge Graphs play a critical role in LLM powered Question Answering systems on SQL databases. However, at the time that we started this research in July 2023, it was not clear to what extent. The starting point of our work is to understand the role of Knowledge Graphs for accuracy, given that hallucinations became one of the largest concerns in the industry. Our work comes in two parts.

First, we seek to understand the accuracy of LLM-powered question answering systems with respect to enterprise questions, enterprise SQL databases and the role knowledge graphs play to improve the accuracy. Our first contribution [25, 26] is a benchmark with experimental results showing that by using GPT-4 and zero-shot prompting, enterprise natural language questions over enterprise SQL databases schema and generating a SQL query achieved 16.7% accuracy. This accuracy increased to 54.2% when a SPARQL query was evaluated over a Knowledge Graph representation of the SQL database in the form of an OWL ontology and R2RML mapping, thus an accuracy improvement of 37.5%. The benchmark can be found here: <https://github.com/datadotworld/cwd-benchmark-data>. This contribution has made an impact in the industry. The benchmark and the results were initially independently reproduced and validated by dbt Labs³. Several semantic layer vendors have further validated our results^{4 5 6 7 8 9}. The GraphRAG Manifesto by Neo4j argues that one of the benefits of GraphRAG relative to vector-only RAG is due to higher accurate responses, citing our benchmark and results¹⁰.

Leveraging the learnings from our first contribution, namely understanding what happened with inaccurate queries, our intuition is that accuracy can be further increased by 1) leveraging the ontology of the knowledge graph to check for errors in the LLM generated SPARQL queries and 2) using the LLM to repair incorrect queries. Our second contribution [3, 4] is a two-part approach consisting 1) Ontology-based Query Check (OBQC), which checks in a deterministic manner if the query is valid by applying rules based on the semantics of the ontology. If the OBQC detects an error, we could either determine to not return the result thus terminate or we could try to repair the query, and 2) LLM Repair, which repairs the detected incorrect SPARQL query generated by the LLM. The result is a new query which can then be passed back to the OBQC. By grouping all the questions in the benchmark, the OBQC and LLM Repair increased the accuracy 72.55%. If the repairs were not successful after three iterations, an unknown result was returned, which occurred 8% of the time. The result is an error rate of 20%.

The conclusion of our work is that Knowledge Graph provides higher accuracy for LLM powered question answering systems on SQL databases. Therefore, enterprises that are considering to use LLMs for question answering on their SQL databases must invest in knowledge graphs.

²Adopt a Data Semantics Approach to Drive Business Value," Gartner Report by Guido De Simoni, Robert Thanaraj, Henry Cook, July 28, 2023

³<https://roundup.getdbt.com/p/semantic-layer-as-the-data-interface>

⁴<https://www.atscale.com/blog/semantic-layers-make-genai-more-accurate/>

⁵<https://www.wisecube.ai/blog/optimizing-llm-precision-with-knowledge-graph-based-natural-language-qa-systems/>

⁶<https://blog.kuzudb.com/post/llms-graphs-part-1/>

⁷<https://delphiq.substack.com/p/delphi-at-100-dbt-semantic-layer>

⁸<https://cube.dev/blog/semantic-layers-the-missing-piece-for-ai-enabled-analytics>

⁹<https://www.stratio.com/blog/stratio-business-semantic-data-layer-delivers-99-answer-accuracy-for-llms/>

¹⁰<https://neo4j.com/blog/graphrag-manifesto/>

B Understanding the role of Knowledge Graphs on LLM’s Accuracy for Question Answering on SQL

While question answering systems have shown remarkable performance in several Text-to-SQL benchmarks [7, 8], such as Spider [30], WikiSQL[33], KaggleDBQA[19] their implications relating to enterprise SQL databases remain relatively obscure. We argue that existing Question Answering and Text-to-SQL benchmarks, although valuable, are often misaligned with real-world enterprise settings:

1. these benchmarks typically overlook complex database schemas representing enterprise domains, which likely comprise hundreds of tables,
2. they also often disregard questions that are crucial for operational and strategic planning in an enterprise, including questions related to business reporting, metrics, and key performance indicators (KPIs), and
3. a critical missing link is the absence of a business context layer – metadata, mappings, transformations, ontologies, that provides business semantics and knowledge about the enterprise.

Recent benchmarks [20, 22] are attempting to address the first challenge. However, the second and specially the third point have not been a focus of those new benchmarks. Without these vital components, LLMs for enterprise question answering on SQL databases risk being disconnected from the reality of enterprise data, leading to hallucinations and uncontrolled outcomes.

We investigate the following two research questions:

RQ1: To what extent Large Language Models (LLMs) can accurately answer enterprise natural language questions over enterprise SQL databases.

RQ2: To what extent Knowledge Graphs can improve the accuracy of Large Language Models (LLMs) to answer enterprise natural language questions over enterprise SQL databases.

The hypothesis is the following: *An LLM powered question answering system that answers a natural language question over a knowledge graph representation of the SQL database returns more accurate results than an LLM powered question answering system that answers a natural language question over the SQL database without a knowledge graph.*

Enterprise SQL Schema The enterprise SQL schema used in the benchmark comes from the P&C Data Model for Property And Casualty Insurance¹¹, a standard model created by Object Management Group (OMG), a standards development organization. This OMG specification addresses the data management needs of the Property and Casualty insurance community.

Enterprise Questions The benchmark comes with 43 Question-Answer pairs as evaluation criteria, where the input is the question, and the output is the corresponding answer to the question based on a data instance. The questions are written in English, and refer to concepts covered by the data. Since there can be multiple valid SQL queries for a given question, the determining accuracy factor is the final output instead of a generated SQL query. In order to score the execution accuracy of an LLM, we need to have a reference answer to each question. An "answer" in this situation is itself a query; it is a query that was written by a human expert, which gives the expected correct answer to the question. Each question has a reference query in SQL for the relational database, and SPARQL for the knowledge graph. Naturally, each query gives the same response when run against the data.

The questions are classified on a spectrum of low to high complexity:

¹¹<https://www.omg.org/spec/PC/1.0/About-PC>

- Low question complexity: Pertains to business reporting use cases, aimed at facilitating daily business operations. From a technical standpoint, these questions are translated into SELECT-FROM SQL queries.
- High question complexity: Arises in the context of Metrics and Key Performance Indicators (KPIs) within an organization. These questions are posed to make informed strategic decisions crucial for organizational success. From a technical standpoint, these questions are translated to SQL queries involving aggregations and mathematical functions.

Questions also depend on the number of tables required to provide an answer. Therefore, questions are also classified on a spectrum of low to high schema:

- Low schema complexity: Small number of tables (i.e. 0 - 4), denormalized schema
- High schema complexity: Larger number of tables (5+), normalized schema, many-to many join tables, etc.

By combining these two spectrums, four quadrants are defined which are used to classify the questions as shown in Figure 32:

High Question Complexity	<u>High Question/Low Schema Complexity</u> e.g. What is the average time to settle a claim by policy number? <ul style="list-style-type: none"> • Aggregation • Math • 4 tables 	<u>High Question/High Schema Complexity</u> e.g. What is the total loss of each policy where loss is the sum of loss payment, Loss Reserve, Expense Payment, Expense Reserve Amount <ul style="list-style-type: none"> • Aggregation • Math • 9 tables
	<u>Low Question/Low Schema Complexity</u> e.g. Return all the claims we have by claim number, open date and close date? <ul style="list-style-type: none"> • Projection 3 columns • 1 table 	<u>Low Question/High Schema Complexity</u> e.g. What are the loss payment, Loss Reserve, Expense Payment, Expense Reserve Amount by Claim Number <ul style="list-style-type: none"> • Projection 3 columns • 6 tables
	Low Schema Complexity	High Schema Complexity

Figure 32: Four quadrants to classify questions: (1) Low Question/Low Schema Complexity, (2) High Question/Low Schema Complexity, (3) Low Question/High Schema Complexity, and (4) High Question/High Schema Complexity

This 43 questions of the benchmark can be found in [25] and on Github¹². While 43 questions may be considered small, the benchmark ensures coverage of key scenarios that reflect real-world enterprise data usage and queries in the insurance domain. All the questions in the benchmark are building blocks to answer one of the most important key metrics in the insurance industry: Loss Ratio. While benchmarks with larger numbers of questions can provide generalizability to evaluate question answering systems and setup leaderboards, the goal of this benchmark is to understand the role of Knowledge Graphs and to

¹²<https://github.com/datadotworld/cwd-benchmark-data>

what extent the accuracy improves. The quadrant provides visibility on the type of extent. Furthermore, the question quadrant can be considered as a framework to be applied for other domains; instead of generating a *laundry list* of questions, categorize them in these quadrants.

Context Layer The context layer consists of two parts:

- **Ontology:** Business Concepts, Attributes, and Relationships that describe the insurance domain.
- **Mapping:** transformation rules from the source SQL schema to the corresponding Business Concepts, Attributes, and Relationships in the target ontology.

For this current version of the benchmark, the context layer is provided in machine readable as RDF: ontology in OWL and mapping in R2RML. The OWL ontology and R2RML mappings can be used to create the Knowledge Graph either in a virtualized or materialized way.

Scoring The benchmark reports three scores: Execution Accuracy, Overall Execution Accuracy and Average Overall Execution Accuracy.

- **Execution Accuracy (EA):** We follow the metric of Execution Accuracy (EA) from the Spider benchmark [30]. An execution is accurate if the result of the query matches the answer for the query. Note that the order or the labels of the columns are not taken in account for accuracy.
- **Overall Execution Accuracy (OEA):** Given the non-deterministic nature of LLMs, there is no guarantee that given an input question, the generated query will always be the same thus providing the same answer. Therefore, every question has a Overall Execution Accuracy (OEA) score which is calculated as (# of EA)/Total Number of runs.
- **Average Overall Execution Accuracy (AOEA):** The Average Overall Execution Accuracy is the average number of OEA scores for a given set of questions. This set could be for all the questions in the benchmark or all the questions in a quadrant.

The benchmark serves as a framework for the results to be reproduced in an enterprise’s own setting using their own enterprise schemas, questions and context.

B.1 Experimental Setup

The question answering system we evaluated was a zero-shot prompt to GPT-4, that is instructed to generate a query, which is executed against the database. The resulting response is compared to the response given by the reference query.

The particular parameters to the OpenAI API are as follows:

- `max_tokens = 2048`
- `n = 1`
- `temperature = 0.3`

Additionally, a timeout was set so that computations that take more than 60 seconds are considered to be failures.

Note that the goal of our experiment is to understand the role of Knowledge Graphs on accuracy. The focus of the experiment is not to understand how a certain LLM performs with a Knowledge Graph. That is why we select only one LLM for our experiment, namely GPT-4. Naturally, future work should include a comparison of multiple LLMs in order to understand how a Knowledge Graph can increase accuracy on different LLMs.

B.2 Question Answering System for SQL

The question answering system for SQL is shown in Figure 33. The question and the SQL DDL for the database are provided as zero-shot prompt to GPT-4. These are combined together using the following simple prompt template:

SQL Zero-shot Prompt

```
INSERT SQL DDL
Write a SQL query that answers the following question. Do not explain
the query. Return just the query, so it can be run verbatim from your
response.
Here's the question:
INSERT QUESTION
```

We kept the prompt simple for this experiment, because we wanted to focus on the ability of the contextual information (the DDL in the case of SQL) to provide necessary information for the formation of the query.

The resulting query is sent verbatim to the SQL processor of data.world, which returns an answer in a tabular form. This is converted into a Pandas DataFrame for comparison. At the same time, the reference query for the question is sent to data.world, and its result is also converted to a DataFrame. Once they are both in the form of DataFrames, it is a simple matter to compare them. Details of this comparison are available from the Spider project[30].

B.3 Question Answering System for Knowledge Graph

The question answering system for the Knowledge Graph is shown in Figure 34. The question and the OWL ontology are provided as zero-shot prompt to GPT-4. These are combined together using the following simple prompt template:

SPARQL Zero-shot Prompt

As in the SQL case, we kept the prompt simple. The extra line about the SERVICE allows the LLM to produce queries that invoke the data.world knowledge graph virtualization layer. In principle, this adds some complexity to the SPARQL prompt, but in practice, GPT-4 seemed to handle it very well.

The resulting query is sent verbatim to the SPARQL processor of data.world, and the result converted to a DataFrame, just as for the SQL case.

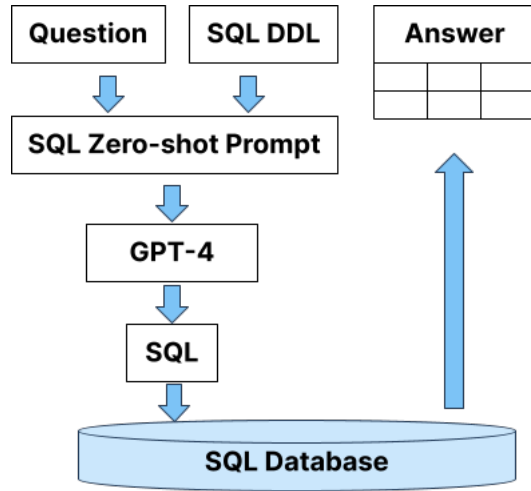


Figure 33: Question Answering System for SQL

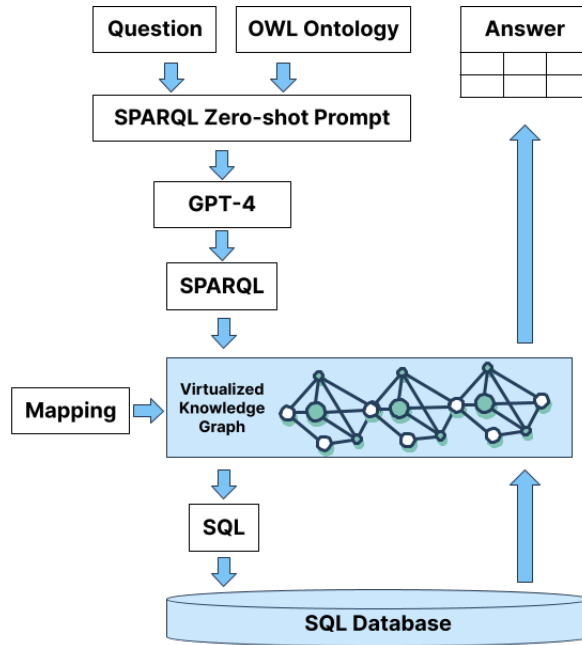


Figure 34: Question Answering System for Knowledge Graph

B.4 Results

The results are presented in four parts 1) overall, 2) question quadrant, 3) partial accuracy and 4) inaccurate results. In the results, we refer to

- SPARQL as question over Knowledge Graph representation of the SQL database and
- SQL as questions directly on the SQL databases without a Knowledge Graph.

Given that the OEA of a question is a percentage, the results are presented as a heatmap. Every cell corresponds to a generated query for the given question. The value in the cell is the OEA for that question. The green color corresponds to 100% OEA. The red color corresponds to 0% OEA. The color scale goes from green to red.

The Overall and Quadrant results are presented in Table 18.

B.4.1 Overall

By grouping all the questions in the benchmark, SQL achieves an AOEA of 16.7%. In comparison, SPARQL achieves an average OEA of 54.2% as shown in Figure 35. The heatmap that depicts the OEA for each question is shown in Figure 36: Therefore, overall SPARQL accuracy was 3x the SQL accuracy.

Overall, Natural Language questions translated to SPARQL over a Knowledge Graph representation of the SQL database achieved 3x the accuracy of natural language questions translated to SQL and executed directly over the SQL database. Combining all the questions into one overall result is not satisfactory because there are nuances to the types of questions. This is why we also present the results in each of the quadrants.

B.4.2 Quadrant

Figure 37 presents the AOEA scores for questions in each quadrant. Figure 38 presents the heatmap for each quadrant. We observe the following results:

- Low Question/Low Schema: SQL achieves an AOEA of 25.5%. In comparison, SPARQL achieves an AOEA of 71.1%. The SPARQL accuracy is 2.8X the SQL accuracy.
- High Question/Low Schema: SQL achieves an AOEA of 37.4%. In comparison, SPARQL achieves an AOEA of 66.9%. The SPARQL accuracy is 1.8X the SQL accuracy.
- Low Question/High Schema: SQL was not able to answer any question accurately. In comparison, SPARQL achieves an AOEA of 35.7%.
- High Question/High Schema: SQL was not able to answer any question accurately. In comparison, SPARQL achieves an AOEA of 38.7%.

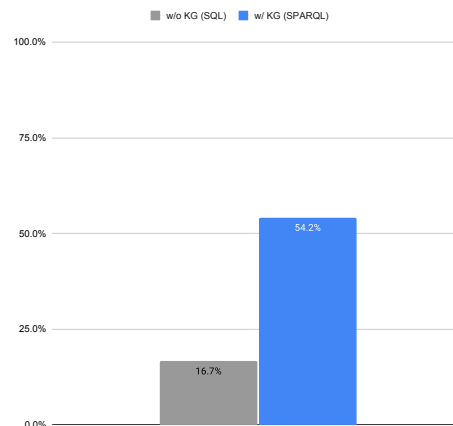


Figure 35: Average Overall Execution Accuracy (AOEA) of SPARQL and SQL for all the questions in the benchmark

Per the hypothesis, SPARQL achieves higher accuracy than SQL in every quadrant. Furthermore, it is surprising to observe that SQL was not able to answer any question in the High Schema Complexity quadrants. These results by quadrant sheds further light on understanding the extent. In each quadrant, SPARQL accuracy is higher than the SQL accuracy. While the SPARQL accuracy is 2.8X the SQL accuracy for Low Question/Low Schema and 1.8X for High Question/Low Schema, it was unforeseen that SQL was not able to accurately answer any questions for Low Question/High Schema and High Question/High Schema. The results also lead us to understand when SQL starts to fail. When a question requires more than 4 tables to provide then answer, the accuracy drops to zero.

	w/o KG (SQL)	w/ KG (SPARQL)	Improvement
All Questions	16.7%	54.2%	37.5%
Low Question/Low Schema	25.5%	71.1%	45.6%
High Question/Low Schema	37.4%	66.9%	29.5%
Low Question/High Schema	0%	35.7%	35.7%
High Question/High Schema	0%	38.5%	38.5%

Table 18: Average Overall Execution Accuracy (AOEA) of Overall and Quadrant Results

B.4.3 Partial Accuracy

We manually analyzed the generated SQL and SPARQL queries and observed that a subset of queries produced partially accurate results. We consider a partially accurate answer to be one where the returned answers are accurate but incomplete. During the manual analysis, the following patterns for partially accurate answers are observed:

- **Overlap:** the columns returned by the query are correct, however, they are a subset of the accurate answer. In some cases, they include other columns that are not part of the expected answer. This can be seen as a form of a semantic overlap[10].
- **Return Identifier:** An internal identifier was returned instead of the appropriate label.

Consider the question *Return all the claims we have by claim number, open date and close date?* and the following generated SQL and SPARQL query:

SQL

```
SELECT Claim_Identifier, Claim_Open_Date, Claim_Close_Date
FROM Claim
```

SPARQL

```
SELECT ?claim ?claimOpenDate ?claimCloseDate
WHERE {
  ?claim a in:Claim ;
    in:claimNumber ?claimNumber ;
    in:claimOpenDate ?claimOpenDate ;
    in:claimCloseDate ?claimCloseDate .
}
```

0.0%	0.0%
0.0%	0.0%
0.0%	0.0%
0.0%	0.0%
0.0%	0.0%
0.0%	0.0%
0.0%	0.6%
0.0%	3.2%
0.0%	9.4%
0.0%	10.6%
0.0%	22.1%
0.0%	22.7%
0.0%	23.6%
0.0%	27.0%
0.0%	29.1%
0.0%	31.8%
0.0%	35.5%
0.0%	36.4%
0.0%	36.4%
0.0%	43.3%
0.0%	48.5%
0.0%	58.1%
0.0%	58.2%
0.0%	59.6%
0.0%	62.2%
0.0%	65.5%
0.0%	75.7%
0.0%	88.5%
0.9%	94.5%
3.2%	96.7%
3.6%	97.1%
7.4%	98.2%
8.8%	99.1%
29.1%	99.1%
36.4%	99.4%
40.0%	99.4%
47.2%	99.7%
57.0%	100.0%
88.2%	100.0%
97.3%	100.0%
99.1%	100.0%
99.5%	100.0%
100.0%	100.0%
w/o KG (SQL)	w/ KG (SPARQL)

Figure 36: Overall Execution Accuracy (OEA) of SPARQL and SQL for all the questions as a heatmap

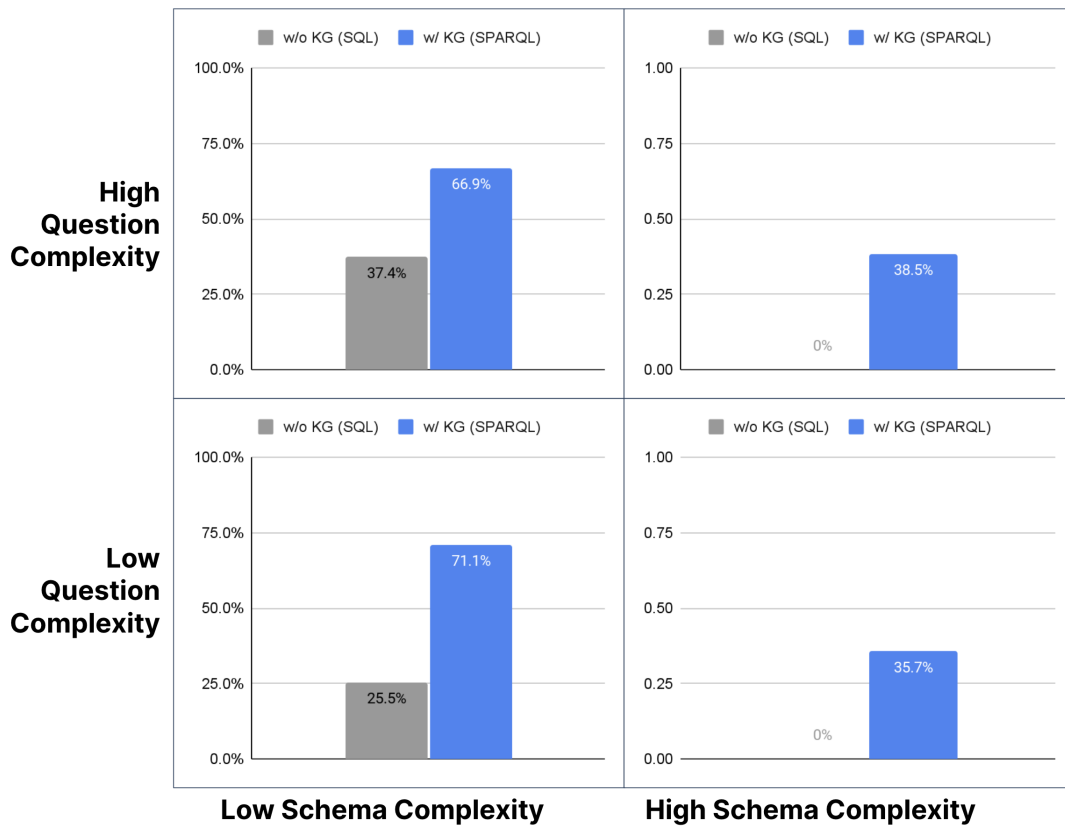


Figure 37: Average Overall Execution Accuracy (AOEA) of SPARQL and SQL for each quadrant

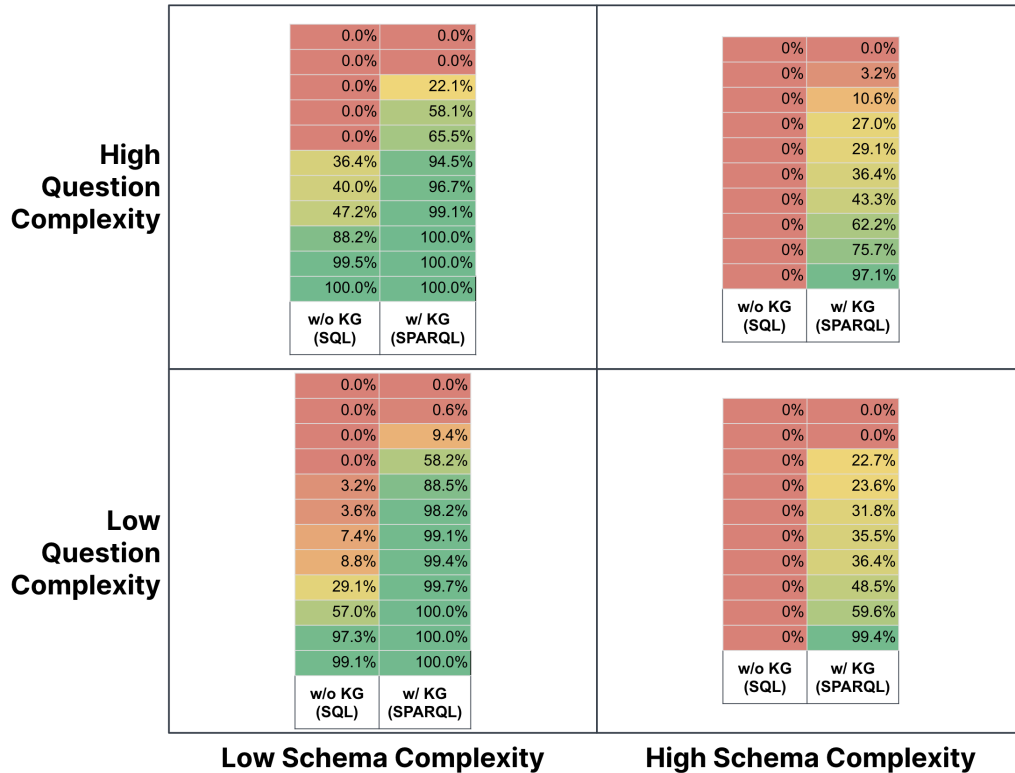


Figure 38: Overall Execution Accuracy (OEA) of SPARQL and SQL for each quadrant as a heatmap

The answer for claim open date and claim close date are accurate and is a subset of the correct answer. However in the SQL query, the `Claim_Identifier` column is being returned as the claim number, when in fact, the claim number is actually the column `company_claim_number`. In the SPARQL query case, the variable `?claim` is returned which binds to the IRI that uniquely identifies each claim. The claim number is not returned.

In practice, if a user is interacting with a system and the results are missing a column, they could ask for the missing column or provide a label instead of an identifier. Therefore partial accuracy may be acceptable for users. However this is an open question on how to define partial accuracy and how to score it.

B.4.4 Inaccuracy

During the manual analysis of the generated queries, we also observed query characteristics that generated the inaccurate answers. These characteristics were different for SQL and SPARQL.

SQL Inaccuracy The following three types of inaccuracies were observed:

- **Column Name Hallucinations:** Column names were generated that do not exist in the corresponding table.
- **Value Hallucinations:** Generated value applied as a filter on a column where that value does not exist in the database.
- **Join Hallucinations:** Generated joins that are not accurate.

SPARQL Inaccuracy

- **Incorrect Path:** The generated query does not follow the correct path of the properties in the ontology. The generated path goes from A to C when the correct path is A to B to C.
- **Incorrect Direction:** The generated query swaps the direction of a property. The generated direction is B to A, when the correct direction is A to B.

The inaccuracy of SQL queries are based on hallucination while the inaccuracy of SPARQL queries are based on path inconsistency. The SQL hallucinations are evident: column names that don't exist in a table and values that the LLM does not know if they exist in the data. The joins may seem plausible, but they are not how the database was designed, thus returning empty results. For SPARQL queries, the generated paths are indicative that the LLM knew what the correct starting and end node was and the error was on defining the correct path from the start node to the end node. One could even argue that the LLM appears to do some sort of reasoning but not always getting it correct. This observation is what led us to the second part of our work.

C Ontologies to the Rescue

Consider the following knowledge represented in an ontology of a knowledge graph: *a Policy is sold by an Agent*. If an LLM generated SPARQL query representing the statement *an Agent is sold by a Policy*, it would be inconsistent because it does not match the semantics of the ontology (i.e. this doesn't make sense). Our intuition is two-fold. First, by leveraging the ontology of knowledge graph, we can check the LLM generated SPARQL query and detect these types of errors. Second, we can also use the LLM to repair incorrect SPARQL queries.

For example, assume the following question “*return all the policies that an agent sold*”, resulted in the following SPARQL query:

```
SELECT ?agent ?policy
WHERE {
  ?agent :soldByAgent ?policy .
  ?agent rdf:type :Agent
}
```

and given the following snippet of an OWL ontology

```
:soldByAgent a owl:DatatypeProperty;
  rdfs:domain :Policy ;
  rdfs:range :Agent .
```

we could determine that the generated query should be correct if the domain of **:soldByAgent** is **:Policy**. However, per the query, the domain is **:Agent** and assuming they are disjoint, the generated query does not match the semantics of the ontology, thus it is incorrect. Given an explanation of this error, we could then prompt the LLM to try again.

We investigate the following research questions:

RQ3: To what extent can the accuracy increase by leveraging the ontology of a knowledge graph to detect errors of a SPARQL query and an LLM to repair the errors?

RQ4: What types of errors are most commonly presented in SPARQL queries generated by an LLM?

The hypothesis is the following: *An ontology can increase the accuracy of an LLM powered question answering system that answers a natural language question over a knowledge graph.*

This first part of our approach is the **Ontology-based Query Check** (OBQC), which checks if the query is valid by applying rules based on the semantics of the ontology. A set of rules checks the semantics of the body of the query (i.e. the WHERE clause). Another set of rules checks the head of query (i.e. the SELECT clause). It is important to clarify that the Ontology-based Query Check does not use an LLM. It is a deterministic rule-based approach based solely on the semantics of the ontology. If the OBQC detects an error, we could try to repair the query.

Repairing databases[1] and programs[11, 32] has been an long standing research area in computer science. Recently, LLMs have been applied to repair programs[5, 18]. Inspired by these approaches, consider the following example. Once a SPARQL query is detected to be incorrect, we can define an explanation for the reason why it is incorrect. Per our running example, an explanation is the following: *The property :soldByAgent has domain :Policy, but its subject ?agent is a :Agent, which isn't a subclass of :Policy..* What if we can pass the incorrect SPARQL query, with this explanation and prompt the LLM to rewrite the query?

The second part of our approach is **LLM Repair**, which repairs the SPARQL query generated by the LLM. It takes as input the incorrect query and the explanation coming from the rule(s) that was fired as an explanation to why the query is incorrect and re-prompts the LLM. The result is a new query which can then be passed back to the OBQC. Our approach gives us the opportunity to understand the capability of an LLM to repair a SPARQL query and thus further improve the accuracy. Figure 39 depicts the overview of our approach.

C.1 Ontology-based Query Check

Knowledge Graphs defined using the Semantic Web technology stack (specifically, RDF, RDFS, OWL and SPARQL) have been built on a rigorous logical foundation. The exact meaning of a statement (triple) in RDF is given in terms of predicate logic; the meaning of a model in RDFS or OWL is specified according to a logical foundation [2, 15, 16]¹³. The meaning of a SPARQL query is specified in terms of these logical foundations. A practical upshot of this theoretical framework is that it is possible to know exactly what constraints a model in RDFS or OWL places on the correctness of a SPARQL query, and these constraints can be described in an executable way in SPARQL. The approach takes two inputs: a SPARQL query and an ontology. The output consists of a list of sentences that describe ways in which the SPARQL query deviates from the specifications in the ontology.

The check system relies on the declarative nature of SPARQL, the structure of Basic Graph Patterns and, the ability to query the ontology via SPARQL itself. If the generated query deviates from the ontology, the approach outlines how. The approach to achieve this is threefold:

First, a SPARQL query consists of a pattern to be matched against the data (specified after the keyword WHERE in the query); known as a Basic Graph Pattern (BGP) of the query. The process begins with extraction of BGPs from the generated SPARQL query, replacing variables with resources from a reserved namespace (prefixed with qq:). Some portions of the original query logic, including the SELECT clause, subquery structures, filters, UNIONS, OPTIONAL and NOT clauses, aren't considered since the focus is on examining the compatibility of the BGP with the ontology structure. We leave that for future work. However, note that violations of BGPs in an OPTIONAL or FILTER NOT EXISTS / MINUS context are not ignored as they can also provide vital insights into the query understanding.

Second, a *conjunctive graph*¹⁴ is constructed by encapsulating two named graphs: **:query** and **:ontology**, representing the SPARQL query's BGP-turned-RDF and the ontology, respectively.

¹³For an introduction to knowledge graphs and semantic web technologies, we refer the reader to the textbooks "Semantic Web for the Working Ontologist" and "Knowledge Graphs" <https://kgbook.org/>

¹⁴using RDFLib nomenclature

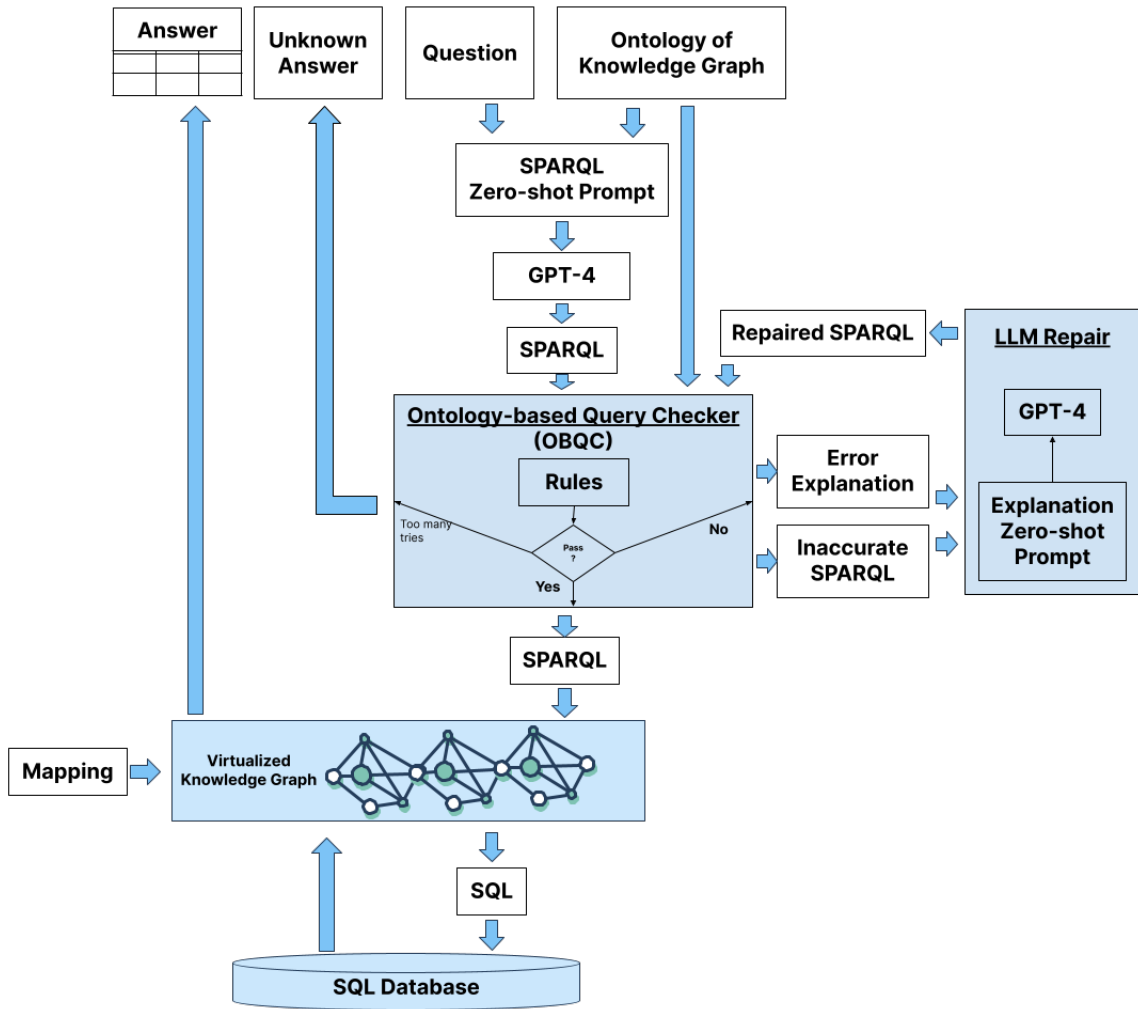


Figure 39: Overview of our Ontology-based Query Checker and LLM Repair approach

Third, the ontology consistency rules are applied guided by the formal logic of RDFS and OWL. The rules are implemented in SPARQL which query the `:query` and `:ontology` graphs and identify instances where the query diverges from the ontology.

Note that this ontology-based query check approach could be extended to other Knowledge Graphs that lack a rigorous semantic foundation, such as property graphs. However, property graphs do not have well-defined standardized schema language and an inference system. This would need to be explicitly defined in order for our approach to be applied.

C.1.1 Rules

We defined two set of rules based on the body and the head of a query. The body rules check that the Basic Graph Patterns of a SPARQL query (i.e. WHERE clause) matches the semantics of an ontology. Our approach follows a subset of the RDF Schema (RDFS) semantics¹⁵. The body rules are:

- Domain: If the domain of a property `p` is a class `C`, then the subject of any triple using `p` as a predicate must be a member of class `C`.
- Range: If the range of a property `p` is a class `C`, then the object of any triple using `p` as a predicate must be a member of class `C`.
- Double Range: If two triples make conflicting requirements on the range of a property then error.
- Double Domain: If two triples make conflicting requirements on the domain of a property then error
- Domain Range: If the object of a first triple is the subject of a second triple, then the range of the property of the first triple should be the same as the domain of the property of the second triple.
- Incorrect Property: All the properties in the query need to exist in the ontology.

The head rules check the head of a SPARQL query (i.e. SELECT clause). A common error for an LLM is to include extra values in the SELECT clause or to leave some out. These errors have nothing to do with the ontology. However, a very common error is to include a variable in the SELECT clause that will be bound to an IRI (an identifier). The head rules are:

- Subject Output: if a query selects a variable that is the subject of a basic triple pattern, then it is an IRI.
- IRI Output: if a predicate has a specified range which is a class, then the object of that triple is an IRI.

For simplicity, in this paper we only provide an example of the Domain Rule. The description of the implementation for all the rules can be found in [3, 4].

The Domain rule (`rdfs:domain` in RDF Schema) is defined in English as follows: If the domain of a property `p` is a class `C`, then the subject of any triple using `p` as a predicate must be a member of class `C`. The domain rule is formally defined as:

```
IF
  ?p rdfs:domain ?C .
  ?s ?p ?o .
THEN
  ?s rdf:type ?C .
```

¹⁵<https://www.w3.org/TR/rdf11-schema/>

The following SPARQL query is a representation of this domain rule which detects a violation:

```
SELECT ?p ?domain ?s ?class WHERE {
  GRAPH :query{
    ?s ?p ?o .
    ?s a ?class .
  }
  GRAPH :ontology{
    ?p rdfs:domain ?domain .
    FILTER (ISIRI (?domain))
  }
  FILTER NOT EXISTS {
    ?class rdfs:subClassOf* ?domain .
  }
}
```

The following example shows how the domain rule is used to check a BGP against an ontology. Suppose the LLM generated the following query:

```
SELECT ?agent WHERE {
  ?agent :soldByAgent ?policy .
  ?agent rdf:type :Agent
}
```

This query has a BGP consisting of two-triples:

```
?agent :soldByAgent ?policy .
?agent rdf:type :Agent
```

The BGP is turned into RDF graph by replacing the variables with resources from a reserved namespace (prefixed with qq:):

```
qq:agent :soldByAgent qq:policy .
qq:agent rdf:type :Agent
```

Now, suppose the ontology includes the following definition of `:soldByAgent`:

```
:soldByAgent
  rdfs:domain :Policy ;
  rdfs:range :Agent .
```

The conjunctive graph in nquads is the following:

```
:query {
  qq:agent :soldByAgent qq:policy .
  qq:agent rdf:type :Agent .
}
:ontology {
  :soldByAgent
    rdfs:domain :Policy ;
    rdfs:range :Agent .
}
```

The first clause, on the graph `:query`, finds the precondition for `rdfs:domain`; there is a triple with predicate (`?p`) whose subject is a member of some `class`. The second clause searches the ontology for a relevant domain definition; that is, that same property `?p` has a specified domain. This query ignores domain definitions that are not IRIs, which would typically include domains that are UNIONS or INTERSECTIONS of other classes. We have simplified the query for exposition in this paper, but it would be easy enough to extend the query to deal with other OWL constructs. We leave this as future work. Finally, the FILTER clause of this query checks to make sure that the class specified in the input query (`?class`) is not included in the domain (`?domain`). If all of these conditions in the evaluation query hold, then we have found a violation of the ontology in the query.

Continuing our example, in the `:query` graph, we have a match for the first clause, with the binding

```
?s -> qq:agent
?p -> :soldByAgent
?class -> :Agent
```

The second clause searches `:ontology` for a triple matching

```
?soldByAgent rdfs:domain ?domain
```

this matches, with `?domain` bound to `:Policy` (which is indeed an IRI). Finally, we test whether

```
:Agent rdfs:subClassOf* :Policy .
```

Notice that the meaning of the `*` in SPARQL implies that this would succeed if `:Agent` were the same as `:Policy`. But in this case, they are not the same, and there is no such triple, so the FILTER NOT EXISTS condition succeeds, and the check comes up with a match, with the following bindings

```
?p -> :soldByAgent
?domain -> :Policy
?s -> qq:agent
?class -> :Agent
```

This information is not very understandable to a human, and might not be usable to an LLM. But it can be formatted into a meaningful sentence in English as follows:

The property `:soldByAgent` has domain `:Policy`, but its subject `?agent` is a `:Agent`, which isn't a subclass of `:Policy`.

For each check rule, we provide a template that can create this explanation. In this case, the template is: *The property {p} has domain {dom}, but its subject {s} is a {class}, which isn't a subclass of {dom}*

C.2 LLM Repair

The LLM Repair is a prompt that takes as two inputs: 1) the list of issues for which the query is incorrect which is the output of the OBQC and 2) the incorrect SPARQL query. The prompt is the following:

Explanation Zero-shot Prompt

The output is a new LLM generated SPARQL query, which is passed again to the OBQC. This cycle repeats until the check pass, or an upper limit of cycles is reached. In our experiments, the limit is 3. In this latter case, the query generation is said to be unknown; there is no point in sending a query that is known to be faulty to the database.

It is instructive to note that the LLM repair focuses on that task; we do not repeat the question nor the ontology to the LLM. The ontology input was taken into consideration by the OBQC, and the question is reflected in the query so far.

It is also noteworthy that there are two possible outcomes; we can achieve a query that we have considerable confidence in (because it matches the semantics of the ontology), or we fail to create such a query. In the latter case, we are aware of the failure of the system. In contrast to a pure LLM-based system which is prone to *hallucinations*, when we get the wrong answer, we know it, and can report that to the user.

C.3 Results

The results consider three cases: 1) accurate queries (that get the right answer), 2) inaccurate queries (that get the wrong answer), and 3) *unknown* queries, queries that we know are incorrect and are not able to repair. As we summarize the results, we follow the metric of Execution Accuracy (EA) from the Spider benchmark [30].

We report the following metrics for a SPARQL query generated by an LLM:

- Execution Accuracy First Time: if the OBQC returns true the first time which results in an accurate execution.
- Execution Accuracy with Repairs: if the OBQC returns false the first time and the LLM Repair results in an accurate execution.
- Execution Unknown with Repairs: if the OBQC returns false the first time and the LLM Repair is unable to repair after three attempts.

C.3.1 Accuracy Results

By grouping all the questions in the benchmark, the Average Overall Execution Accuracy with Repairs is 72.55%. This is an increase of 29.67% based on the Average Overall Execution Accuracy First Time which is 42.88%. The Average Overall Execution Unknown with Repairs is 8% which implies that the LLM Repair is usually able to repair the queries and is still able to identify when queries can not be repaired. By combining the Average Overall Execution Accuracy with Repairs and Average Overall Execution Unknown with Repairs, the error rate is 19.44%. Based on the results shown in Table 19, we observe that the Ontology-based Query Check and LLM Repair favorably increased the accuracy and reduced the error rate in two areas:

- Questions on High Complex Schema: the Ontology-based Query Check and LLM Repair positively impacts the accuracy of all types of questions that are on a high schema complexity.
- Combining Accuracy and Unknowns: "I don't know" is a valid answer and arguable a better answer than an inaccurate answer. By combining accuracy and unknown, the error rate reduces, notably making a bigger impact in Low Question/Low Schema.

We observe the following details for each quadrant:

- Low Question/Low Schema: the Ontology-based Query Check and LLM Repair increased the accuracy by 25.48%. The Execution Unknown with Repairs was the highest in this quadrant, 12.87%. Combined, this implies that the error rate is 10.46%, the lowest of all the quadrants.

	Average Overall Execution Accuracy First Time	Average Overall Execution Accuracy with Repairs	Average Overall Execution Unknown with Repairs	Average Overall Execution Accuracy + Unknown with Repairs	Error Rate
All Questions	42.88%	72.55%	8%	80.56%	19.44%
Low Question / Low Schema	51.19%	76.67%	12.87%	89.54%	10.46%
High Question / Low Schema	69.76%	75.10%	6.02%	81.12%	18.88%
Low Question / High Schema	17.20%	76.33%	3.45%	79.79%	20.21%
High Question / High Schema	28.17%	60.62%	8.40%	69.03%	30.97%

Table 19: Average Overall Execution Accuracy (AOEA) of Overall and Quadrant Results

- High Question/Low Schema: the Ontology-based Query Check and LLM Repair increased the accuracy by 5.34% which was the lowest increase of the quadrants. It is not clear why. The final error rate is 18.88%.
- Low Question/High Schema: the Ontology-based Query Check and LLM Repair had a substantial impact by increasing the accuracy by 59.13% with an error rate of 20.21%.
- High Question/High Schema: the Ontology-based Query Check and LLM Repair had a meaningful impact by increasing the accuracy by 32.45% with an error rate of 30.97%.

Comparing to the results with our first contribution, the accuracy increase is notable. The Average Overall Execution Accuracy of the same zero-shot Text-to-SPARQL prompt on a Knowledge Graph representation of the SQL database, reported in our previous work, was 54.2%, indicating an error rate of 45.8%. With our Ontology-based Query Check and LLM Repair, the error rate is reduced to 19.44%. Figure 40 depicts these results for all the questions in the benchmark. Figure 41 depicts these results for all questions in each quadrant.

A follow up question to understand the *extent* is the following: *How much of the possible execution accuracy improvement was achieved?* In other words, given the number of times the system achieved an accurate answer on the first time, and the total number of runs, we know how much is left for improvement. Therefore, how much of that improvement was achieved? For example, given a total of 10 runs, where 2 of them were accurate on the first try achieving a First Time Execution Accuracy of 20%, means that there are 8 runs left where the OBQC and LLM Repair to repair a query in order to achieve 100% Execution Accuracy with Repairs. Let’s say that the system is able to accurately repair 4 times, therefore the Execution Accuracy with Repairs is 60%. The achievable improvement is 50% because the accurate repair occurred 4 times out of the 8 possible times. Achievable Improvement is calculated as

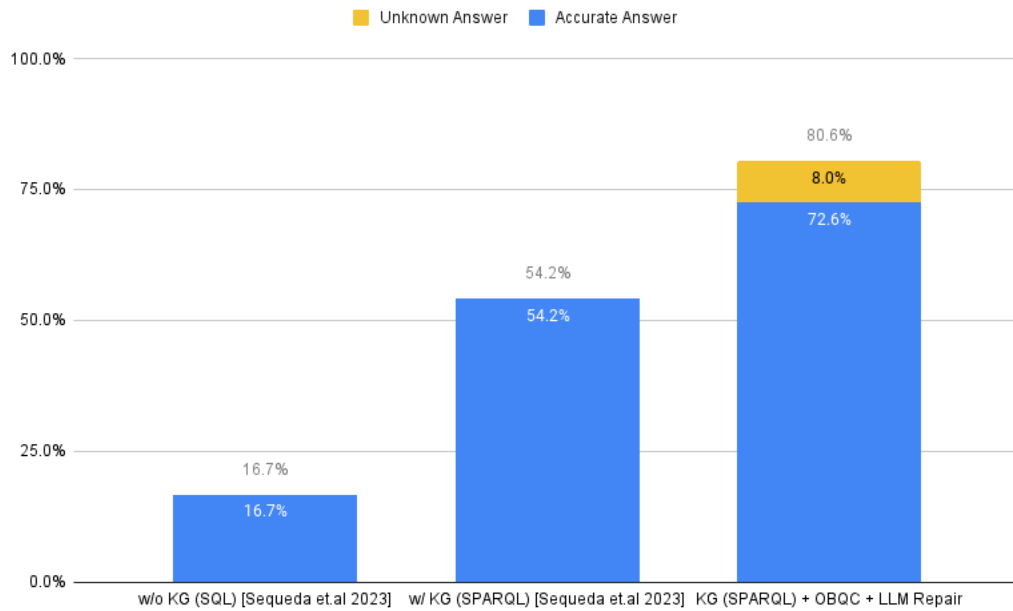


Figure 40: Average Overall Execution Accuracy (AOEA) of SPARQL and SQL for all the questions in the benchmark compared to OBQC and LLM Repair

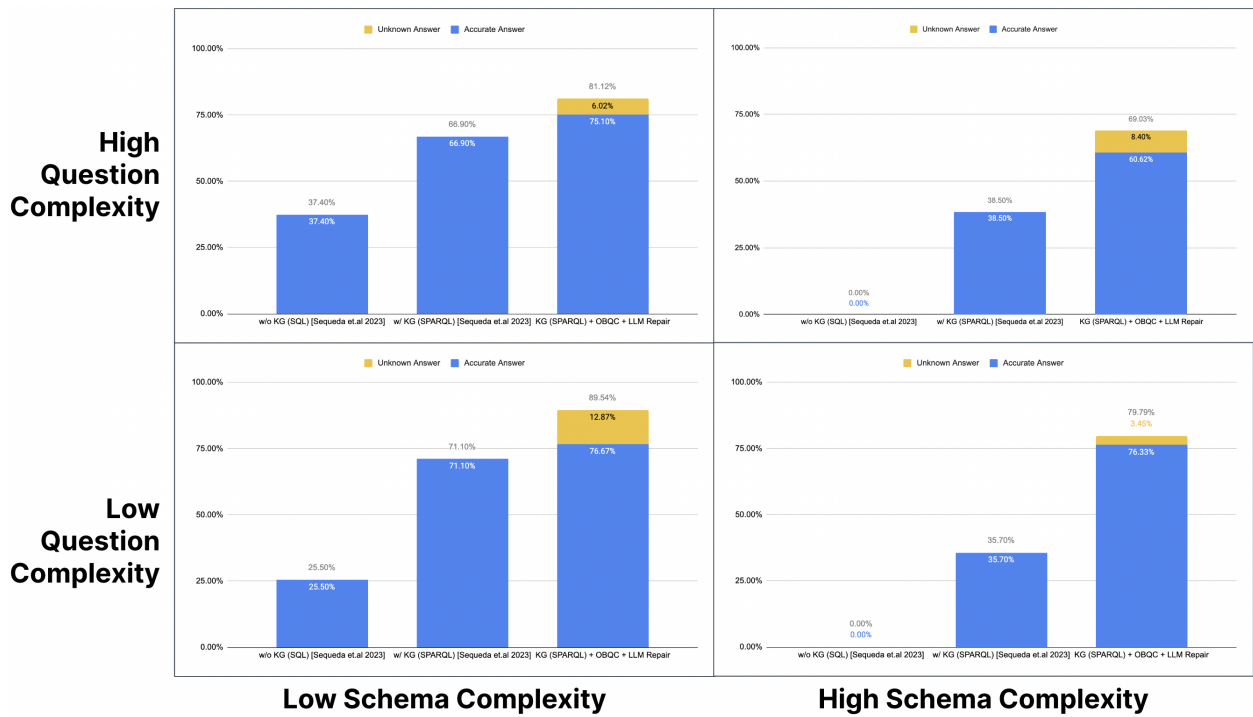


Figure 41: Average Overall Execution Accuracy (AOEA) of SPARQL and SQL for all questions in each quadrant compared to OBQC and LLM Repair

(Number of Accurately Repaired Queries) / (Total Number of runs - Number of First Time Executed Accurate queries). The results of achievable improvement are shown in Table 20

The results indicate that the OBQC is able to successfully repair queries half of the time. Thus, we are halfway there and there is still room for improvement. Recall that OBQC mainly checks the body of the query and just two checks in the SELECT clause determine if IRI identifiers are returned. We postulate that a set of inaccurate queries are due to the overlap type of partial accurate queries: the columns returned by the query are correct, however, they are a subset of the accurate answer. Therefore possible repair rules can be defined to check the head of the query. Additionally, this may indicate the need for more expressive ontologies.

		Average Achievable Improvement
All Questions		55.57%
Low Schema	Question/Low	49.30%
High Schema	Question/Low	40.45%
Low Schema	Question/High	72.23%
High Schema	Question/High	57.70%

C.3.2 Error Type Results

In our experiments, we kept count of the number of times a rule was invoked by the OBQC. Table 21 presents the percentage of usage of each rule in the OBQC. Notably, 70% of the repairs were done by the Body rules checks while 30% of the repairs were done by the Head rules. The rules exclusively related to domain were invoked 42.16% of the time and surprisingly, rules exclusively related to range were invoked less than 1% of the time. The Domain Range rule contributed to 22.78% of the repairs.

A surprising result is that the domain related rules had the largest impact in repairs. These results may shine some light on what is happening underneath the hood inside an LLM, namely the relationship between english language and triples of a graph. English is written and read from left to right. The domain of a property has a relationship to the left side of a triple. If the LLM writes a query which is wrong, it would most probably get it wrong at the beginning of a sentence/triple. This may be an explanation on why the domain related rules were the most impactful.

D Conclusion

We are now able to provide answers to our research questions:

RQ1: To what extent Large Language Models (LLMs) can accurately answer Enterprise Natural Language questions over Enterprise SQL databases.

Table 20: Average Achievable Improvement of Overall and Quadrant Results

Rule	Usage
Double Domain	37.47%
Domain Range	22.78%
IRI Output	18.40%
Subject Output	11.91%
Domain	4.69%
Incorrect Property	4.26%
Range	0.43%
Double Range	0.06%

Table 21: Rule usage in the Ontology-based Query Check

Answer: Using GPT-4 and zero-shot prompting, Enterprise Natural Language questions over Enterprise SQL databases achieved 16.7% Average Overall Execution Accuracy. For Low Question/Low Schema, the Overall Execution Accuracy was 25.5%. For High Question/Low Schema, the Overall Execution Accuracy was 37.%. However, for both Low Question/High Schema and High Question/High Schema, the accuracy was 0%.

RQ2: To what extent Knowledge Graphs can improve the accuracy of Large Language Models (LLMs) to answer Enterprise Natural Language questions over Enterprise SQL databases.

Answer: Using GPT-4 and zero-shot prompting, Enterprise Natural Language question over a Knowledge Graph representation of the enterprise SQL database achieved 54.2% Average Overall Execution Accuracy. The overall SPARQL accuracy was 3x the SQL accuracy and the accuracy improvement was 37.5%.

RQ3: To what extent can the accuracy increase by leveraging the ontology of a knowledge graph to detect errors of a SPARQL query and an LLM to repair the errors?

Answer: By using the same zero-shot prompting on GPT-4 and adding the Ontology-Based Query Check and LLM Repair, that new accuracy is 72.55%. That is over a 4x accuracy improvement compared to the zero-shot prompting with SQL. Given that we consider an unknown result in our approach because the LLM Repair is not able to repair and the Ontology-Based Query Check catches the error, the overall error rate is 19.44%.

RQ4: What types of errors are most commonly presented in SPARQL queries generated by an LLM?

Answer: 70% of the repairs were done by the Ontological checks while 30% of the repairs were done by the SELECT Clause checks. The domain related rules had the largest impact in repairs, being invoked 42.16% of the time. An interpretation of this result is that if the LLM writes a query which is wrong, it would most probably get it wrong at the beginning of a sentence/triple and the domain of a property has a relationship to the left side of a triple.

The answers to our research questions are evidence that supports the main conclusion of our work: Knowledge Graph provides higher accuracy for LLM powered question answering systems on SQL databases.

What is evident is that context is crucial for accuracy and these results further emphasizes the need to invest in business context. The point to be made here is a call to action that investing in context of SQL databases is required to increase the accuracy of LLMs for question answering over the SQL databases. In this work, the context was presented in the form of a knowledge graph consisting of an ontology that describes the semantics of the business domain and mappings that connect the physical schema with the ontology which are used to create the knowledge graph. The ontology can also include further semantics such as synonyms, labels in different languages, which are not expressible in a SQL DDL.

Arguably, our work has influenced the wider data industry to acknowledge the need to invest in semantics and knowledge graphs in this new AI era. Our work has been reproduced and validated by multiple independent vendors^{16 17 18 19 20 21 22}, and contributed to rise of GraphRAG²³.

The takeaway for enterprises is that in order to provide trustworthy question answering systems that results in highly accurate results, they must make an investment in business context and semantics.

¹⁶<https://roundup.getdbt.com/p/semantic-layer-as-the-data-interface>

¹⁷<https://www.atscale.com/blog/semantic-layers-make-genai-more-accurate/>

¹⁸<https://www.wisecube.ai/blog/optimizing-llm-precision-with-knowledge-graph-based-natural-language-qa-systems/>

¹⁹<https://blog.kuzudb.com/post/llms-graphs-part-1/>

²⁰<https://delphiq.substack.com/p/delphi-at-100-dbt-semantic-layer>

²¹<https://cube.dev/blog/semantic-layers-the-missing-piece-for-ai-enabled-analytics>

²²<https://www.stratio.com/blog/stratio-business-semantic-data-layer-delivers-99-answer-accuracy-for-llms/>

²³<https://neo4j.com/blog/graphrag-manifesto/>

This context needs to be effectively managed in metadata managements systems such as data catalog and governance platforms built on a knowledge graph architecture.

References

- [1] Foto N. Afrati and Phokion G. Kolaitis. Repair checking in inconsistent databases: algorithms and complexity. In Proceedings of the 12th International Conference on Database Theory, ICDT '09, page 31–41, New York, NY, USA, 2009. Association for Computing Machinery.
- [2] Dean Allemang, Jim Hendler, and Fabien Gandon. Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL, volume 33. Association for Computing Machinery, New York, NY, USA, 3 edition, 2020.
- [3] Dean Allemang and Juan Sequeda. Increasing the accuracy of llm question-answering systems on sql databases with ontologies. In Proceedings of the International Semantic Web Conference (ISWC), 2024.
- [4] Dean Allemang and Juan Sequeda. Increasing the LLM accuracy for question answering: Ontologies to the rescue! CoRR, abs/2405.11706, 2024.
- [5] Islem Bouzenia, Premkumar Devanbu, and Michael Pradel. Repairagent: An autonomous, llm-based agent for program repair, 2024.
- [6] Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriber. Expanding the scope of the ATIS task: The ATIS-3 corpus. Proceedings of the workshop on Human Language Technology, pages 43–48, 1994.
- [7] Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-SQL evaluation methodology. In Iryna Gurevych and Yusuke Miyao, editors, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 351–360, Melbourne, Australia, July 2018. Association for Computational Linguistics.
- [8] Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-sql evaluation methodology. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 351–360, July 2018.
- [9] Alessandra Giordani and Alessandro Moschitti. Automatic generation and reranking of sql-derived answers to nl questions. In Proceedings of the Second International Conference on Trustworthy Eternal Systems via Evolving Software, Data and Knowledge, pages 59–76, 2012.
- [10] Parke Godfrey and Jarek Gryz. Answering queries by semantic caches. In Trevor J. M. Bench-Capon, Giovanni Soda, and A Min Tjoa, editors, Database and Expert Systems Applications, 10th International Conference, DEXA '99, Florence, Italy, August 30 - September 3, 1999, Proceedings, volume 1677 of Lecture Notes in Computer Science, pages 485–498. Springer, 1999.
- [11] Claire Le Goues, Michael Pradel, and Abhik Roychoudhury. Automated program repair. Commun. ACM, 62(12):56–65, nov 2019.

- [12] Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: An automatic question-answerer. In Papers Presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference, IRE-AIEE-ACM '61 (Western), page 219–224, New York, NY, USA, 1961. Association for Computing Machinery.
- [13] C. Cordell Green and Bertram Raphael. The use of theorem-proving techniques in question-answering systems. In Proceedings of the 1968 23rd ACM National Conference, ACM '68, page 169–181, New York, NY, USA, 1968. Association for Computing Machinery.
- [14] Gary G. Hendrix, Earl D. Sacerdoti, Daniel Sagalowicz, and Jonathan Slocum. Developing a natural language interface to complex data. ACM Trans. Database Syst., 3(2):105–147, jun 1978.
- [15] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge Graphs. Synthesis Lectures on Data, Semantics, and Knowledge. Morgan & Claypool Publishers, 2021.
- [16] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. ACM Comput. Surv., 54(4):71:1–71:37, 2022.
- [17] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 963–973, 2017.
- [18] Matthew Jin, Syed Shahriar, Michele Tufano, Xin Shi, Shuai Lu, Neel Sundaresan, and Alexey Svyatkovskiy. Inferfix: End-to-end program repair with llms. In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2023, page 1646–1656, New York, NY, USA, 2023. Association for Computing Machinery.
- [19] Chia-Hsuan Lee, Oleksandr Polozov, and Matthew Richardson. KagleDBQA: Realistic evaluation of text-to-SQL parsers. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 2261–2273, Online, August 2021. Association for Computational Linguistics.
- [20] Fangyu Lei, Jixuan Chen, Yuxiao Ye, Ruisheng Cao, Dongchan Shin, Hongjin Su, Zhaoqing Suo, Hongcheng Gao, Wenjing Hu, Pengcheng Yin, Victor Zhong, Caiming Xiong, Ruoxi Sun, Qian Liu, Sida Wang, and Tao Yu. Spider 2.0: Evaluating language models on real-world enterprise text-to-sql workflows, 2024.
- [21] Fei Li and H. V. Jagadish. Constructing an interactive natural language interface for relational databases. Proceedings of the VLDB Endowment, 8(1):73–84, September 2014.
- [22] Jinyang Li, Binyuan Hui, Ge Qu, Jiayi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chen-Chuan Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM already serve as A database interface? A big bench for large-scale database grounded text-to-sqls. In Alice Oh, Tristan Naumann, Amir Globerson,

Kate Saenko, Moritz Hardt, and Sergey Levine, editors, Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023, 2023.

- [23] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. arXiv preprint arxiv:306.08302, 2023.
- [24] Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. Towards a theory of natural language interfaces to databases. In Proceedings of the 8th International Conference on Intelligent User Interfaces, pages 149–157, 2003.
- [25] Juan Sequeda, Dean Allemang, and Bryon Jacob. A benchmark to understand the role of knowledge graphs on large language model’s accuracy for question answering on enterprise SQL databases. CoRR, abs/2311.07509, 2023.
- [26] Juan Sequeda, Dean Allemang, and Bryon Jacob. A benchmark to understand the role of knowledge graphs on large language model’s accuracy for question answering on enterprise SQL databases. In Olaf Hartig and Zoi Kaoudi, editors, Proceedings of the 7th Joint Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), Santiago, Chile, 14 June 2024, pages 5:1–5:12. ACM, 2024.
- [27] Lappoon R. Tang and Raymond J. Mooney. Automated construction of database interfaces: Integrating statistical and relational learning for semantic parsing. In 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, pages 133–141, 2000.
- [28] W. A. Woods. Transition network grammars for natural language analysis. Commun. ACM, 13(10):591–606, oct 1970.
- [29] Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig, and Thomas Dillig. Sqlizer: Query synthesis from natural language. In International Conference on Object-Oriented Programming, Systems, Languages, and Applications, ACM, pages 63:1–63:26, October 2017.
- [30] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3911–3921, 2018.
- [31] John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2, pages 1050–1055, 1996.
- [32] Quanjun Zhang, Chunrong Fang, Yuxiang Ma, Weisong Sun, and Zhenyu Chen. A survey of learning-based automated program repair. ACM Trans. Softw. Eng. Methodol., 33(2), dec 2023.
- [33] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. CoRR, abs/1709.00103, 2017.

Symphony: Towards Trustworthy Question Answering and Verification using RAG over Multimodal Data Lakes

Nan Tang, Chenyu Yang, Zhengxuan Zhang, Yuyu Luo
HKUST(GZ), China

Ju Fan
Renmin University, China

Lei Cao
University of Arizona, USA

Sam Madden
MIT, USA

Alon Halevy
Google, USA

Abstract

Large Language Models (LLMs) have revolutionized access to multimodal data lakes, enabling users to query and analyze complex information across diverse data modalities using natural language. However, their generative nature unavoidably leads to hallucinations, resulting in inaccuracies and misinformation in models like GPT, Llama, and Gemini. To address this, we introduce **Symphony**, a system designed for trustworthy question answering and verification using multimodal data lakes. **Symphony** supports two core functions: reasoning and verification. In reasoning, **Symphony** retrieves relevant information from multimodal data sources, breaks down complex queries into manageable sub-questions, and uses specialized tools (e.g., LLMs or DBMS) to generate grounded answers. For verification, it cross-checks (LLM) generated answers against trusted sources, such as private or enterprise data lakes, to enhance accuracy and reliability. By integrating these processes, **Symphony** mitigates factual inaccuracies, aligns outputs with trusted data, and adapts to a wide range of applications.

A Introduction

The rise of **Large Language Models (LLMs)** has unlocked transformative **opportunities** across diverse domains, including natural language processing, data analysis, and creative design, among many others. By enabling interaction with complex systems through intuitive natural language queries, LLMs democratize access to knowledge and data, allowing users to obtain valuable information quickly and cost-effectively without requiring specialized expertise or manual data processing. This accessibility has driven widespread adoption across industries such as healthcare, finance, and customer service, empowering both professionals and non-experts to extract insights and make informed decisions with ease. By reshaping traditional workflows, LLMs have the potential to significantly enhance productivity and decision-making across a wide range of applications.

However, alongside these opportunities, significant **risks of LLMs** (or more generally, generative AI) have emerged. The phenomenon of hallucinations, i.e., the generation of inaccurate or misleading

Ju Fan is the corresponding author

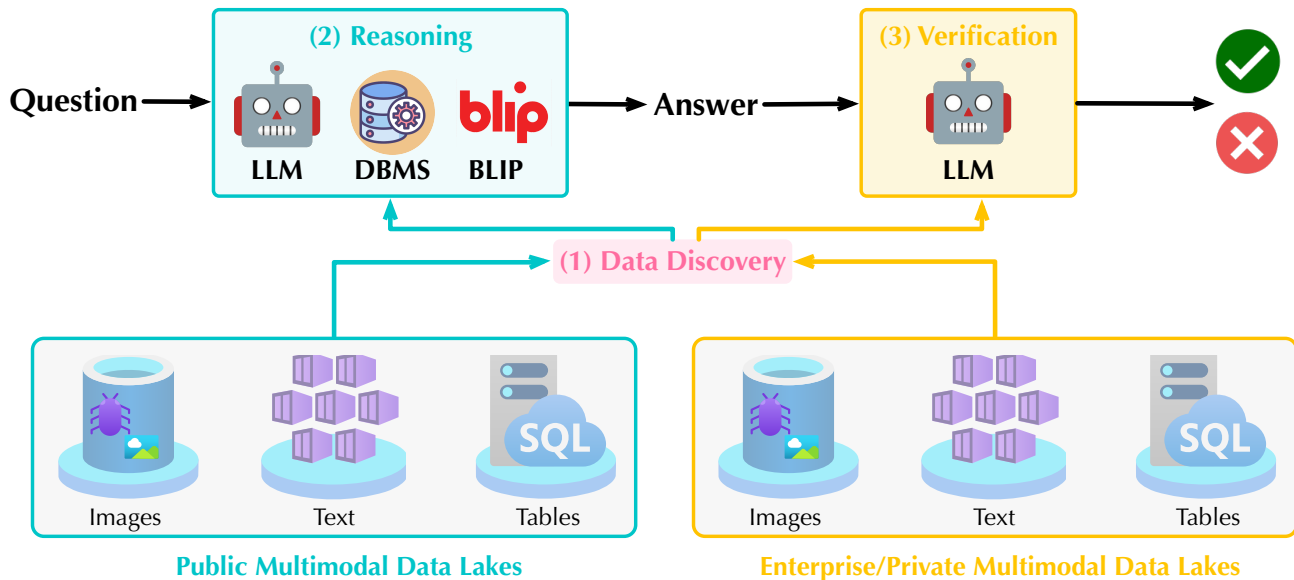


Figure 42: An Overview of **Symphony**.

information, poses a serious challenge to trust in LLM outputs. By 2023, analysts estimated that chatbots hallucinate as much as 27% of the time¹, and factual errors were present in 46% of generated texts [5], underscoring the prevalence of this issue. These inaccuracies can negatively impact various aspects, including decision-making, the spread of misinformation, privacy violations, and potential legal liabilities.

Developing **trustworthy solutions for question answering** is a priority for both academia and industry, with efforts focused on improving accuracy, reducing biases, and aligning models with human values. Companies like OpenAI and Google enhance model reliability through advanced training and Responsible AI principles². Initiatives like the Partnership on AI and collaborations by the AI Ethics Lab promote ethical guidelines and accountability in AI development³. However, challenges remain due to the probabilistic and generative nature of LLMs, which leads to unpredictability in outputs. While current efforts are valuable, they are insufficient to fully address issues such as inaccuracies, biases, and lack of explainability. More robust systems are needed, especially in high-stakes applications like data analysis over multimodal data, where trustworthiness is critical.

In this paper, we present **Symphony** [4, 17], a system designed for trustworthy question answering and data analysis over multimodal data lakes. Given a multimodal data lake L and a natural language question Q requiring factual or objective answers, the task of reasoning involves generating an answer to Q by retrieving relevant data from L and applying various reasoning tools such as LLMs, RDBMSs, or graph databases. Additionally, if an answer A is provided—whether from humans or LLMs (possibly enhanced with RAG)—the task of verification is to assess the correctness of A for Q , using the data lake L (either a public data lake or a private/enterprise data lake) to ensure factual accuracy and reliability.

As illustrated in Figure 42, **Symphony** comprises three core modules: (1) **Discovery** operates over multimodal data lakes and serves as the retrieval module; (2) **Reasoning** formulates answers to natural language questions using retrieved information and various tools; and (3) **Verification** assesses the correctness of provided answers utilizing LLMs and multimodal data lakes.

Roadmap. Section B describes the **Discovery** module. Section C discusses the **Reasoning** module.

¹<https://www.nytimes.com/2023/11/06/technology/chatbots-hallucination-rates.html>

²<https://openai.com/safety/>

³<https://aiethicslab.com/>

Section D presents the **Verification** module. Section E describes empirical findings. Section F identifies open problems. Section G discusses related work. Finally, we close this paper by concluding remarks in Section H.

B Symphony: Data Discovery over Multimodal Data Lakes

Data discovery is the process of identifying relevant data files from multimodal data lakes efficiently. As shown in Figure 43, **Symphony** provides two main categories of data discovery methods: (a) word-level similarity search and (b) holistic embedding-based similarity search. These methods are chosen based on a balance between efficiency and effectiveness, enabling retrieval of relevant data from data lakes, thus supporting both reasoning and verification.

Word-level similarity search breaks down queries and data items in the data lakes into words (or terms), retrieves and ranks data items based on combined word-level similarity, as shown in Figure 43 (a). Methods like BM25, TF-IDF, and Jaccard similarity are used. These approaches are simple, interpretable, and computationally efficient, making them ideal for scenarios with high word overlap between the query and data items.

Embedding-based similarity search encodes multimodal data items and queries into high-dimensional vectors in a shared embedding space, enabling fast and precise similarity calculations. As shown in Figure 43, in this process, multimodal data (e.g., text, tables or images) are encoded into dense vector embeddings. These embeddings are then stored in a vector database (e.g., Meta Faiss), and indexed for fast similarity-based retrieval, typically using efficient distance metrics like cosine similarity or dot product. To support various data representations, **Symphony** investigates the following two encoding strategies:

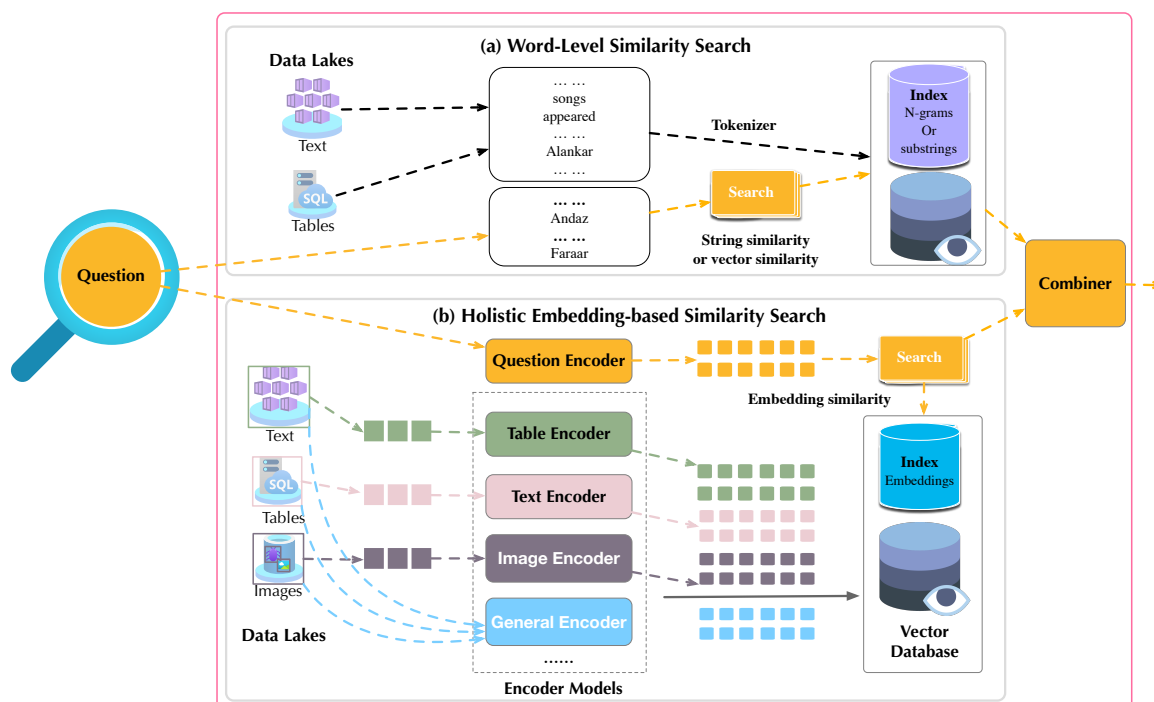


Figure 43: The **Discovery** Module.

1. **Modal-specific Representation Learning.** This strategy uses models tailored to each data type (e.g., text encoder [11], table encoder [16] or image encoder CLIP [14]), capturing unique features like intricate table structures or text nuances, making it ideal for precise retrieval within individual modalities.
2. **Modal-agnostic (Cross-Modal) Representation Learning.** This strategy learns a shared embedding space, using a General Encoder, through cross-modal representation learning (see [4] for more details), enabling similarity comparisons across modalities. This approach supports queries that can retrieve relevant items from different modalities, enhancing interoperability where cross-modal relationships are crucial.

C Symphony: Reasoning over Multimodal Data Lakes

In this section, we present the reasoning process of **Symphony**, which integrates Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) to reason over information from multimodal data lakes. Given a natural language (NL) question, **Symphony** first retrieves top- k relevant data items from the data lakes, as discussed in the previous section. **Symphony** then conducts a question decomposition strategy to address complex queries effectively, where a question can be decomposed into sub-questions, and each sub-question can be answered by different tools, such as LLMs, DBMSs, and so on.

C.1 Question Decomposition

We propose a Question Decomposition strategy to address complex questions requiring information from multiple sources. Here, a *data source* is defined as a collection of data items originating from the same origin, such as an isolated table, a database, or a text passage. When multiple data items, like a table and passage, come from the same Wikipedia page or structured document, they are also treated as a single data source. Similarly, if two tables d_i and d_j have a predefined primary-foreign key (PK-FK) relationship, as with tables from the same database, they are merged into a unified data source d'_k . Initially, we retrieve a set of data items $D = \{d_1, d_2, \dots, d_n\}$ from multimodal data lakes and process them using heuristic methods to form $D' = \{d'_1, d'_2, \dots, d'_m\}$, where $m \leq n$ and each $d_i \in D'$ represents a distinct data source.

The **objective** of our question decomposition strategy is to break down a complex query into simpler sub-questions to facilitate retrieval of relevant information. Ideally, each sub-question should focus on a primary data source. However, we allow flexibility for sub-questions that still require multiple sources after decomposition, accommodating scenarios where information from different sources complements or corroborates each other. This adaptable approach enhances reasoning effectiveness by balancing simplicity, which reduces each sub-question to its essential elements, with the integration of supportive data, enabling relevant details from multiple sources to strengthen the answer’s accuracy. Together, these elements minimize fusion errors and streamline the reasoning flow, creating a coherent, step-by-step resolution.

To achieve this, **Symphony** employs an iterative prompt-based approach with LLM to automate the decomposition process. In each round, the LLM generates a sub-question based on the previous sub-question and data source, or decides to terminate the process. First, an initial prompt is generated to identify the first sub-question and its corresponding data sources. **Symphony** then iteratively uses this information to generate subsequent prompts, guiding the LLM to create the next sub-question until the entire question is resolved. In cases where the LLM determines that a question cannot be effectively decomposed, **Symphony** bypasses decomposition. Instead, it directly leverages the information retrieved from multiple data sources to formulate a reasoning for the original question. For example, a question like “What is the population of France?” cannot be further decomposed into distinct sub-questions.

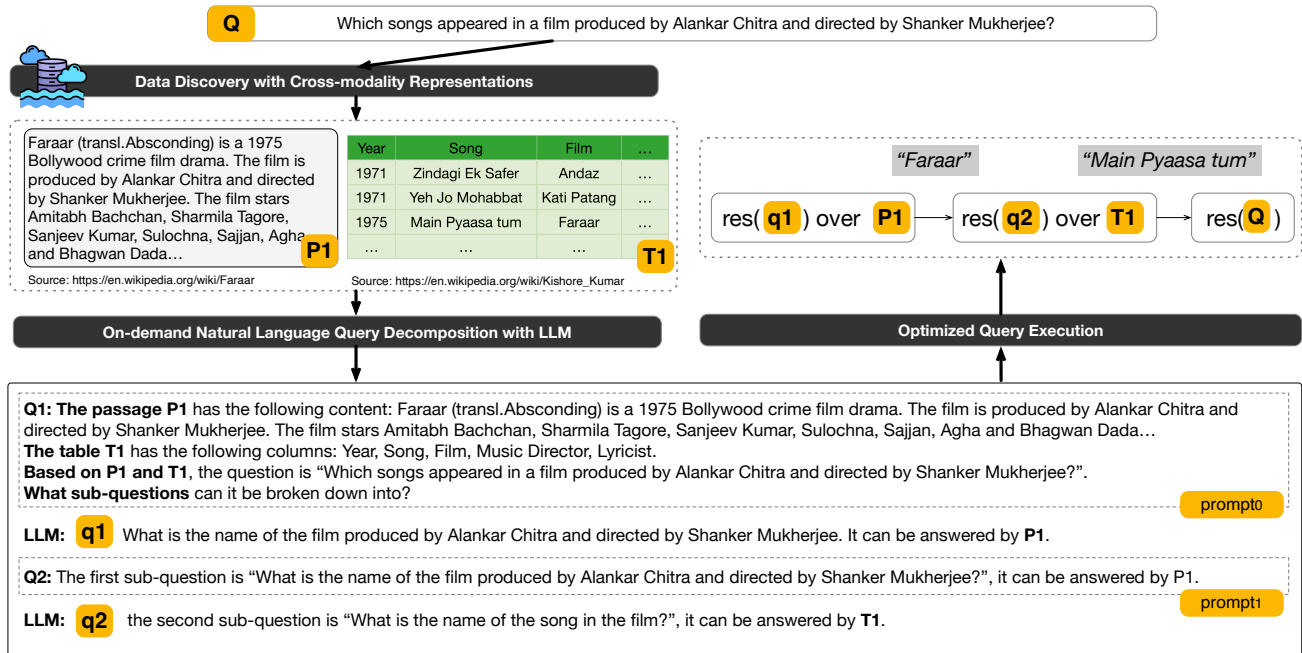


Figure 44: RAG-based Reasoning in **Symphony**

We illustrate the question decomposition process in Figure 44. Suppose we have a question **Q**: “Which song ...,” with two relevant data sources retrieved from multimodal data lakes, a passage **P1** and a table **T1**. Using a template-based approach, **Symphony** constructs an initial prompt, **prompt₀**, based on **Q**, **P1**, and **T1**. **Symphony** sends the prompt **prompt₀** to LLM, and LLM generate a sub-question **q1** as well as the data source on which it should be utilized. Building on the first sub-question **q1** and its data source **P1**, **Symphony** uses the next prompt template to generate **prompt₁**. Given **prompt₁**, the LLM generates the second sub-question, **q2**, and assigns table **T1** as its data source. At this point, the LLM decides to stop, as it considers the original query **Q** fully addressed.

C.2 Reasoning

Question Answering using Retrieval Augmented Generation (RAG). We leverage the powerful reasoning capabilities of large language models (LLMs) to address complex questions alongside relevant retrieved data sources. Using a prompt-based approach, we guide LLMs to conduct nuanced reasoning and generate coherent answers. If necessary, we can also prompt the LLM to provide detailed explanations of the reasoning, enhancing transparency and interpretability. **Symphony** offers NL2SQL as another way to support queries over a single table or a database. In addition to the existing NL2SQL techniques [9], **Symphony** leverages LLMs [12], as well as the prompting techniques to convert NL questions to SQL queries, using similar ideas we introduced in question decomposition.

Sub-Answers Aggregation. For complex questions, answers to each sub-question need to be combined accurately for a complete response. Using a prompt-based approach, the LLM sequentially integrates individual answers, rephrasing them into a coherent response to the original question. For instance, if the task involves summing values from sub-answers, the LLM aggregates these values directly, producing a reliable and context-aware final result.

Reasoning Optimization. To ensure both efficiency and accuracy, reasoning optimization is applied to streamline execution plans and reduce response times. **Symphony** employs a multi-objective optimizer

that balances speed with precision, choosing the best approach based on retrieved data type and question complexity. For instance, if an exact result is critical and the data is highly structured, **Symphony** prioritizes Natural Language to SQL (NL2SQL) for accuracy; otherwise, Table Question Answering (TableQA) can be used for faster, approximate answers.

This optimization framework also manages cost-performance trade-offs by evaluating the computational demands of various query methods. For high-priority questions, accurate methods are selected, while non-critical evaluating may utilize quicker, approximate options. This flexible approach enables **Symphony** to handle complex, multimodal evaluating efficiently, effectively decomposing and aggregating responses across data sources.

D Symphony: Answer Verification over Multimodal Data Lakes

This section describes our verification approach [17]. In **Symphony**, verification occurs when an answer is provided, with the objective of ensuring its correctness. Notably, **Reasoning** and **Verification** are loosely coupled, allowing **Verification** to validate answers regardless of whether they are generated by humans, large language models (LLMs), or other tools.

During verification, this answer is used as a query to retrieve supporting or contradicting data items from the multimodal data lake, with the aim of either validating or refuting the generated information. **Symphony** employs two types of verifiers. The first type is a one-size-fits-all model, such as an LLM, which can be conveniently utilized by sending prompts directly. The second type consists of task-specific models, designed for specialized scenarios, such as PASTA [9] for verifying facts based on tables. While using LLMs by default provides simplicity, we support task-specific models for two main reasons:

1. **Data privacy:** In sensitive domains like healthcare and government, using generic, externally hosted models can risk data exposure. Task-specific, localized models mitigate this by processing data internally, protecting privacy;
2. **Improved accuracy:** Our empirical results show that task-specific models, fine-tuned on relevant datasets, generally outperform generic models for specialized tasks, providing more reliable and precise results.

An example of this verification process is shown in Figure 45. Based on an answer generated by the LLM, **Symphony** retrieves a passage titled “Stomp the Yard” and a table “Meagan Good’s Filmography” from the data lake. Using these retrieved items as evidence, the verifier identifies the generated content as incorrect and provides an explanation for the error.

The multimodal data lakes used in verification can differ from those used in reasoning, such as private or enterprise-specific data lakes, which contain domain-specific knowledge and high-quality, reliable data tailored to organizational needs. Using such data sources in the verification process not only enhances reliability but also ensures that outputs adhere to specific content standards and align with user expectations. This is particularly valuable in enterprise applications where data quality and relevance are important.

E Experiments

We conduct preliminary experiments to demonstrate the effectiveness of **Symphony**. We evaluate its performance in two key processes: Reasoning and Verification.

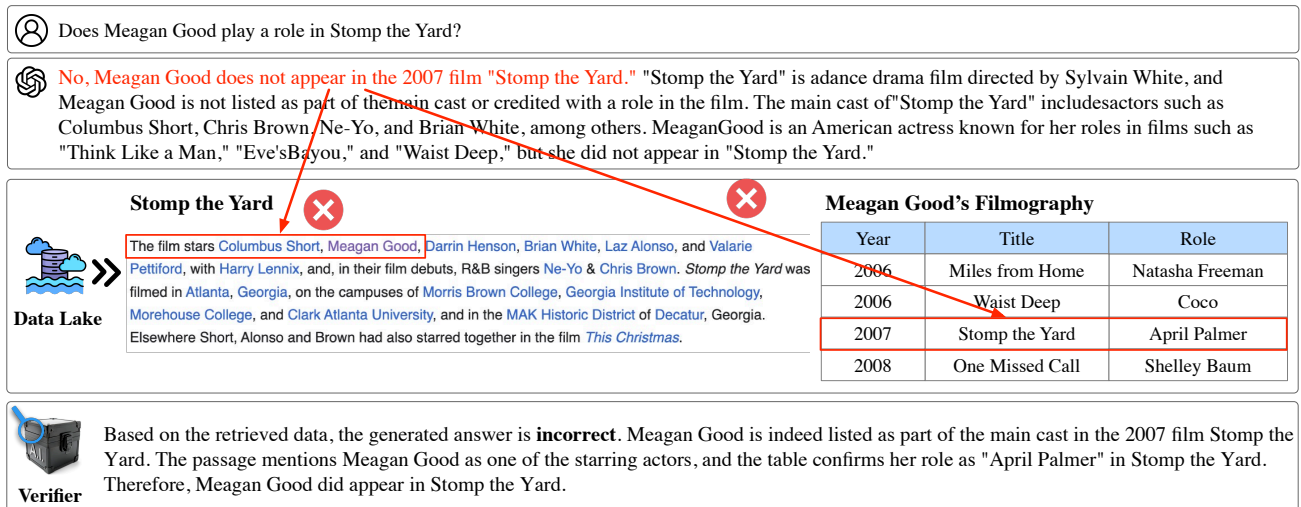


Figure 45: RAG-based Verification in **Symphony**

E.1 Question Answering

Experiment Setting. In this experiment, we focus on evaluating question answering performance using a multimodal data lake consisting of 400K web tables and 6M English passages extracted from Wikipedia. The data lake includes both tables and texts, and each query is designed to retrieve relevant data items to answer a given question. We use 18 manually crafted user queries, each with corresponding ground truth annotations specifying the required data items, sub-queries for decomposition, and final answers.

Data Discovery Evaluation. The effectiveness of data discovery is measured using the recall at K ($R@K$) metric, which calculates the proportion of relevant data items retrieved in the top- K recommendations. The experimental results show that when K is 5, 10, 15, and 20, the $R@K$ values are 40.8%, 46.3%, 59.3%, and 77.8%, respectively. For 12 out of the 18 queries, **Symphony** successfully discovers all the relevant items needed to answer the query. The remaining 6 queries show partial success. In total, 30 out of 38 related items are correctly discovered, demonstrating the potential of the proposed data discovery methodology, even though it is still in a preliminary stage.

Query Decomposition Evaluation. To decompose queries into manageable sub-queries, **Symphony** serializes the discovered data items and uses GPT-3 to generate sub-queries. The output includes the sub-queries and corresponding data item ids. Evaluation of the decomposition quality is based on two criteria: (1) whether each sub-query is useful for solving the original query, and (2) whether the sub-query can be answered correctly using the selected data item. The human evaluation results show that 77.8% of the queries scored 2 (both criteria met), 16.7% scored 1 (only the first criterion met), and 5.5% scored 0.

Table 22 shows the results of 8 instances. **Symphony** is able to handle different aggregation operations, such as sum (Instance 2) and comparison (Instance 3). Further, it correctly understands long sentences (Instance 1). However, **Symphony** has difficulty in dealing with sentences with complex syntactic structures (e.g., in Instance 4, GPT-3 mistakenly generates q_3 because it takes the subject of "released" as Academy Award for Best Foreign Language Film").

Instance 1: Q : Did the 2014 United States house of representatives elections take place on the same day in Rhode Island, South Carolina and Louisiana? $E = \{P_1, P_2, P_3\}$
 $\rightarrow q_1$: Did the 2014 United States House of Representatives elections take place on November 4, 2014 in Rhode Island? $e_1 = P_1$; q_2 : Did the 2014 United States House of Representatives elections take place on November 4, 2014 in South Carolina? $e_2 = P_2$; q_3 : Did the 2014 United States House of Representatives elections take place on November 4, 2014 in Louisiana? $e_3 = P_3$
★ Score: 2

Instance 2: Q : How many representatives were elected in the 2014 United States house of representatives elections in Rhode Island and South Carolina? $E = \{P_1, P_2\}$
 $\rightarrow q_1$: How many representatives were elected in the 2014 United States house of representatives elections in Rhode Island? $e_1 = P_1$; q_2 : How many representatives were elected in the 2014 United States house of representatives elections in South Carolina? $e_2 = P_2$
★ Score: 2

Instance 3: Q : Which number is larger, the number of living former prime ministers of Tunisia or the number of former presidents of Ghana? $E = \{T_1, P_1\}$
 $\rightarrow q_1$: What is the number of living former prime ministers of Tunisia? $e_1 = P_1$; q_2 : What is the number of former presidents of Ghana? $e_2 = T_1$
★ Score: 2

Instance 4: Q : What year was the first German film that won the Academy Award for Best Foreign Language Film released? $E = \{T_1, P_1\}$
 $\rightarrow q_1$: What year was the film The Tin Drum released? $e_1 = T_1$; q_2 : What was the first German film that won the Academy Award for Best Foreign Language Film? $e_2 = T_1$; q_3 : What year was the first Academy Award for Best Foreign Language Film given out? $e_3 = T_1$
★ Score: 0

Instance 5: Q : Did The Tin Drum and Kolya both win the Academy Awards for Best Foreign Language Film? $E = \{P_1, P_2\}$
 $\rightarrow q_1$: whether The Tin Drum has ever won the Academy Award for Best Foreign Language Film? $e_1 = P_1$; q_2 : whether Kolya has ever won the Academy Award for Best Foreign Language Film? $e_2 = P_2$
★ Score: 2

Instance 6: Q : How many years elapsed between the premiere of season 10 and the premiere of season 5 of Law & Order : Special Victims Unit? $E = \{T_1, T_2\}$
 $\rightarrow q_1$: What is the premiere date of season 10 of Law & Order : Special Victims Unit? $e_1 = T_2$; q_2 : What is the premiere date of season 5 of Law & Order : Special Victims Unit? $e_1 = T_1$
★ Score: 2

Instance 7: Q : Did You Let Me Walk Alone participate in the Eurovision Song Contest in 1996? $E = \{T_1, P_1\}$
 $\rightarrow q_1$: What year was the Eurovision Song Contest held in Lisbon, Portugal? $e_1 = T_1$; q_2 : What year was You Let Me Walk Alone released? $e_2 = P_1$
★ Score: 1

Instance 8: Q : Are the tallest building in the united kingdom and the tallest building in poland above 200 meters? $E = \{T_1, T_2\}$
 $\rightarrow q_1$: What is the height of the tallest building in the United Kingdom? $e_1 = T_1$; q_2 : What is the height of the tallest building in Poland? $e_2 = T_2$
★ Score: 2

Table 22: Example sub-queries generated by **Symphony**. q_i and e_i represent the i_{th} sub-query and its corresponding data item. T_i represents a table and P_i represents a text.


	<p>Claim: In 1954 u.s. open (golf), the cash prize for tommy bolt, fred haas, and ben hogan was 960 in total. (Ground Truth: a false claim that should be Refuted)</p>																																																																		
<p>Retrieved Evidence and Verification</p> <p>Table E1: 1954 u.s. open (golf)</p> <table border="1"> <thead> <tr> <th>place</th> <th>player</th> <th>country</th> <th>score</th> <th>to par</th> <th>money</th> </tr> </thead> <tbody> <tr> <td>t1</td> <td>ed furgol</td> <td>united states</td> <td>71 + 70 + 71 + 72 = 284</td> <td>+ 4</td> <td>1 6000</td> </tr> <tr> <td>t2</td> <td>gene littler</td> <td>united states</td> <td>70 + 69 + 76 + 70 = 285</td> <td>+ 5</td> <td>3600</td> </tr> <tr> <td>t3</td> <td>lloyd mangrum</td> <td>united states</td> <td>72 + 71 + 72 + 71 = 286</td> <td>+ 6</td> <td>1500</td> </tr> <tr> <td>t3</td> <td>dick mayer</td> <td>united states</td> <td>72 + 71 + 70 + 73 = 286</td> <td>+ 6</td> <td>1500</td> </tr> <tr> <td>t5</td> <td>bobby locke</td> <td>south africa</td> <td>74 + 70 + 74 + 70 = 288</td> <td>+ 8</td> <td>960</td> </tr> <tr> <td>t6</td> <td>tommy bolt</td> <td>united states</td> <td>72 + 72 + 73 + 72 = 289</td> <td>+ 9</td> <td>570</td> </tr> <tr> <td>t6</td> <td>fred haas</td> <td>united states</td> <td>73 + 73 + 71 + 72 = 289</td> <td>+ 9</td> <td>570</td> </tr> <tr> <td>t6</td> <td>ben hogan</td> <td>united states</td> <td>71 + 70 + 76 + 72 = 289</td> <td>+ 9</td> <td>570</td> </tr> <tr> <td>t6</td> <td>shelley mayfield</td> <td>united states</td> <td>73 + 75 + 72 + 69 = 289</td> <td>+ 9</td> <td>570</td> </tr> <tr> <td>t6</td> <td>billy joe patton (a)</td> <td>united states</td> <td>69 + 76 + 71 + 73 = 289</td> <td>+ 9</td> <td>1 0 1</td> </tr> </tbody> </table> <p>Verification result: Refuted. Explanation: The cash prize for Tommy Bolt, Fred Haas, and Ben Hogan was \$570 each, totaling \$1710.</p>		place	player	country	score	to par	money	t1	ed furgol	united states	71 + 70 + 71 + 72 = 284	+ 4	1 6000	t2	gene littler	united states	70 + 69 + 76 + 70 = 285	+ 5	3600	t3	lloyd mangrum	united states	72 + 71 + 72 + 71 = 286	+ 6	1500	t3	dick mayer	united states	72 + 71 + 70 + 73 = 286	+ 6	1500	t5	bobby locke	south africa	74 + 70 + 74 + 70 = 288	+ 8	960	t6	tommy bolt	united states	72 + 72 + 73 + 72 = 289	+ 9	570	t6	fred haas	united states	73 + 73 + 71 + 72 = 289	+ 9	570	t6	ben hogan	united states	71 + 70 + 76 + 72 = 289	+ 9	570	t6	shelley mayfield	united states	73 + 75 + 72 + 69 = 289	+ 9	570	t6	billy joe patton (a)	united states	69 + 76 + 71 + 73 = 289	+ 9	1 0 1
place	player	country	score	to par	money																																																														
t1	ed furgol	united states	71 + 70 + 71 + 72 = 284	+ 4	1 6000																																																														
t2	gene littler	united states	70 + 69 + 76 + 70 = 285	+ 5	3600																																																														
t3	lloyd mangrum	united states	72 + 71 + 72 + 71 = 286	+ 6	1500																																																														
t3	dick mayer	united states	72 + 71 + 70 + 73 = 286	+ 6	1500																																																														
t5	bobby locke	south africa	74 + 70 + 74 + 70 = 288	+ 8	960																																																														
t6	tommy bolt	united states	72 + 72 + 73 + 72 = 289	+ 9	570																																																														
t6	fred haas	united states	73 + 73 + 71 + 72 = 289	+ 9	570																																																														
t6	ben hogan	united states	71 + 70 + 76 + 72 = 289	+ 9	570																																																														
t6	shelley mayfield	united states	73 + 75 + 72 + 69 = 289	+ 9	570																																																														
t6	billy joe patton (a)	united states	69 + 76 + 71 + 73 = 289	+ 9	1 0 1																																																														
<p>Table E2: 1959 u.s. open (golf)</p> <table border="1"> <thead> <tr> <th>player</th> <th>country</th> <th>year (s)</th> <th>won</th> <th>total</th> <th>to par</th> <th>finish</th> </tr> </thead> <tbody> <tr> <td>ben hogan</td> <td>united states</td> <td>1948, 1950, 1951, 1953</td> <td>287</td> <td>+ 7</td> <td>t8</td> </tr> <tr> <td>cary middlecoff</td> <td>united states</td> <td>1949, 1956</td> <td>294</td> <td>+ 14</td> <td>t19</td> </tr> <tr> <td>liack fleck</td> <td>united states</td> <td>1955</td> <td>294</td> <td>+ 14</td> <td>t19</td> </tr> <tr> <td>liulus boros</td> <td>united states</td> <td>1952</td> <td>297</td> <td>+ 17</td> <td>t28</td> </tr> <tr> <td>tommy bolt</td> <td>united states</td> <td>1958</td> <td>301</td> <td>+ 21</td> <td>t38</td> </tr> </tbody> </table> <p>Verification result: Not related.</p>		player	country	year (s)	won	total	to par	finish	ben hogan	united states	1948, 1950, 1951, 1953	287	+ 7	t8	cary middlecoff	united states	1949, 1956	294	+ 14	t19	liack fleck	united states	1955	294	+ 14	t19	liulus boros	united states	1952	297	+ 17	t28	tommy bolt	united states	1958	301	+ 21	t38																													
player	country	year (s)	won	total	to par	finish																																																													
ben hogan	united states	1948, 1950, 1951, 1953	287	+ 7	t8																																																														
cary middlecoff	united states	1949, 1956	294	+ 14	t19																																																														
liack fleck	united states	1955	294	+ 14	t19																																																														
liulus boros	united states	1952	297	+ 17	t28																																																														
tommy bolt	united states	1958	301	+ 21	t38																																																														

Figure 46: Verifying a textual claim using retrieved tables.

E.2 Answer Verification

We showcase preliminary experimental results that highlight the initial achievements of **Symphony** in facilitating the verification of generative AI.

Experiment Setting. We perform a controlled study to assess textual claims, employing 1,300 textual claims from the TabFact [3] benchmark, which is currently the most advanced benchmark for verifying the credibility of textual hypotheses by utilizing a given table. The data lake consists of 16,573 tables from the TabFact and 2,925 tables sourced from WikiTable-TURL [6].

Evaluation for Retrieval. We use Elasticsearch [8] to retrieve the top-5 tables for each textual claim. Given the limited amount of relevant data, we focus on the recall metric for evaluation. Each textual claim is associated with a corresponding table in the original dataset, which we consider relevant evidence, while other retrieved tables are deemed irrelevant. The retrieval performance, measured by R@5, is 0.88.

Evaluation for Verification. We evaluate the verification process using two different verifiers: GPT-3.5, the default verifier for both data types, and PASTA [9], a specialized model for text verification. The performance of the verifiers is measured by accuracy. When the retrieved data cannot support or refute a claim, the verifier outputs “not related”. However, in this case, since PASTA that only offers two

different answers: “true” or “false”, we consider it’s also correct when PASTA outputs “false”.

We conduct experiments in two settings. When a relevant table is retrieved and provided as evidence to the verifier, PASTA achieves higher accuracy than GPT-3.5 (0.89 vs. 0.75) in verifying the textual claim based on the table. However, in cases where many of the retrieved tables are irrelevant to the claim, the verifier must accurately determine which tables are not related. In this setting, PASTA’s accuracy drops to 0.72 because it has not encountered this scenario during training, while GPT-3.5 improves to 0.91. Thus, when the retrieved data is highly related to the generative data, local models like PASTA have higher accuracy while protecting privacy. In contrast, GPT-3.5 is better at generalizing and providing explanations for further judgments. Users can select the appropriate model based on their requirements.

In Figure 46, we present a case of verifying a textual claim based on retrieved tables using GPT-3.5. **Symphony** retrieves two tables E_1 and E_2 , where E_1 can be used with an aggregation query to refute the claim while E_2 is not related because it is for the year 1959. The red boxes in Figure 46 show that GPT-3.5 can provide not only a verification result but also some explanation.

F Open Problems

Cross-Modal Data Discovery. Data discovery presents a significant challenge within data preparation, especially when dealing with data lakes that store diverse types of data across various formats, including structured data (e.g., tables), semi-structured data (e.g., graphs), and unstructured data (e.g., images and videos). Unlike data lakes containing only relational tables, discovering relevant data across multiple modalities requires addressing the inherent heterogeneity of these data types. One promising direction for tackling this challenge is to explore cross-modal representation learning, which encodes data from different modalities into a unified vector space. This approach can enable a streamlined data discovery process by supporting embedding-based similarity search. While we have made initial strides in cross-modal representation, our current work has not touched the surface of modeling relationships across different data modalities. Further research is needed to deepen our understanding and improve cross-modal data discovery methods.

Cross-Modal Data Reasoning and Verification. One of the complexity of cross-modal reasoning and verification stems from the intricate relationships between different data modalities, such as text, images, and structured data (e.g., tables and knowledge graphs). Each modality often possesses unique characteristics and contextual information that can complicate the verification process. For instance, verifying a claim made in textual data may require correlating it with relevant knowledge graph entities or structured data, where mismatches in representation and interpretation can lead to inaccuracies. Current large language models, such as GPT, demonstrate reasonable performance in reasoning across diverse data types; however, there remains significant room for improvement, particularly regarding privacy and accuracy. To address these challenges, promising directions include the development of domain-specific models that focus on the interactions between specific modalities, improved representation learning techniques for better alignment of data types, and hybrid approaches that combine local and large language models. Additionally, privacy-preserving techniques, such as federated learning and iterative feedback mechanisms, could enhance the robustness and reliability of cross-modal reasoning and verification. These strategies aim to create a more effective framework for ensuring the accuracy and trustworthiness of generative AI outputs across different modalities.

Trustworthiness of Data Sources. The accuracy of discovering and verifying data across different modalities in a data lake can be influenced by the quality and reliability of the underlying data sources. Therefore, it is crucial to assess the trustworthiness of different sources accurately to enhance the overall

accuracy and reliability of the entire verification process.

G Related Works

Retrieval Augmented Generation Question Answering. Large language models sometimes generate factually incorrect or misleading information, often due to a lack of real-time knowledge or limited access to external facts beyond their training data. RAG-based Question Answering addresses this by integrating external knowledge retrieval into the generation process. By retrieving relevant document chunks through semantic search, RAG ensures that the model’s responses are grounded in accurate, real-world information, effectively reducing the likelihood of hallucinations. Early approaches focused on jointly training the retriever and generator, ensuring that the retrieved content aligned with the generation model’s intent to provide more accurate answers [10]. With the success of in-context learning, more recent work has treated the retriever as a separate module, directly providing retrieved information to the model via prompts [18]. As retrieval technologies have advanced, RAG-based systems now support multimodal retrieval, enabling answers that draw from diverse data sources [1, 2, 13].

Trustworthiness of Large Language Models. The trustworthiness of LLMs is essential for their effective deployment in real-world applications. To assess LLM trustworthiness, researchers have proposed various approaches. For example, TrustLLM [7] provides a comprehensive framework for evaluating LLMs across different trust dimensions. However, evaluating LLM trustworthiness remains challenging, with gaps in holistic assessment approaches. Some studies suggest that self-evaluation, where LLMs assess their confidence in the generated outputs, can help improve selective generation and mitigate inaccuracies [15]. Additionally, understanding the internal mechanisms of LLMs, such as the use of local intrinsic dimension (LID) for predicting truthfulness, has been proposed as a way to measure model reliability [19]. In our work, we aim to improve the trustworthiness of LLMs through post-verification, ensuring that generated outputs are validated against reliable sources after generation to minimize inaccuracies and enhance their overall reliability.

H Concluding Remarks

In conclusion, **Symphony** represents a significant advancement in the pursuit of trustworthy question answering over multimodal data lakes. By harnessing the power of RAG, **Symphony** effectively addresses the critical challenge of hallucinations inherent in LLMs. Its dual functionality caters to diverse user needs, facilitating both reasoning and verification processes. Through the decomposition of complex queries and the retrieval of relevant information from various data sources, **Symphony** generates grounded answers that can be rigorously cross-checked against reliable datasets. This collaborative approach not only enhances the accuracy of responses but also fosters confidence in the decision-making processes that rely on such information. As we continue to explore the potential of multimodal data and LLMs, **Symphony** stands out as a versatile tool that can adapt to a wide range of applications, paving the way for more reliable and informed use of LLMs in various domains.

References

- [1] W. Chen, M.-W. Chang, E. Schlinger, W. Y. Wang, and W. W. Cohen. Open question answering over tables and text. In International Conference on Learning Representations, 2021.
- [2] W. Chen, H. Hu, X. Chen, P. Verga, and W. Cohen. MuRAG: Multimodal retrieval-augmented generator for open question answering over images and text. In EMNLP, 2022.

- [3] W. Chen, H. Wang, J. Chen, Y. Zhang, H. Wang, S. Li, X. Zhou, and W. Y. Wang. Tabfact: A large-scale dataset for table-based fact verification. arXiv preprint arXiv:1909.02164, 2019.
- [4] Z. Chen, Z. Gu, L. Cao, J. Fan, S. Madden, and N. Tang. Symphony: Towards natural language query answering over multi-modal data lakes. In CIDR, 2023.
- [5] A. de Wynter, X. Wang, A. Sokolov, Q. Gu, and S.-Q. Chen. An evaluation on large language model outputs: Discourse and memorization. Natural Language Processing Journal, 4:100024, 2023.
- [6] X. Deng, H. Sun, A. Lees, Y. Wu, and C. Yu. Turl: Table understanding through representation learning. ACM SIGMOD Record, 51(1):33–40, 2022.
- [7] Y. H. et al. Trustllm: Trustworthiness in large language models, 2024.
- [8] C. Gormley and Z. Tong. Elasticsearch: The Definitive Guide. O’Reilly Media, Inc., 1st edition, 2015.
- [9] Z. Gu, J. Fan, N. Tang, P. Nakov, X. Zhao, and X. Du. Pasta: Table-operations aware fact verification via sentence-table cloze pre-training. In EMNLP, 2022.
- [10] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave. Atlas: Few-shot learning with retrieval augmented language models. Journal of Machine Learning Research, 24(251):1–43, 2023.
- [11] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense passage retrieval for open-domain question answering. In EMNLP, pages 6769–6781, Nov. 2020.
- [12] B. Li, Y. Luo, C. Chai, G. Li, and N. Tang. The dawn of natural language to sql: Are we fully ready? Proc. VLDB Endow., 17(11):3318–3331, Aug. 2024.
- [13] H. Luo, Y. Shen, and Y. Deng. Unifying text, tables, and images for multimodal question answering. In EMNLP (Findings), 2023.
- [14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PMLR, 2021.
- [15] J. Ren, Y. Zhao, T. Vu, P. J. Liu, and B. Lakshminarayanan. Self-evaluation improves selective generation in large language models. In ICBINB, 2023.
- [16] N. Tang, J. Fan, F. Li, J. Tu, X. Du, G. Li, S. Madden, and M. Ouzzani. RPT: relational pre-trained transformer is almost all you need towards democratizing data preparation. Proc. VLDB Endow., 14(8):1254–1261, 2021.
- [17] N. Tang, C. Yang, J. Fan, L. Cao, Y. Luo, and A. Y. Halevy. Verifai: Verified generative AI. In CIDR, 2024.
- [18] X. Wang, Q. Yang, Y. Qiu, J. Liang, Q. He, Z. Gu, Y. Xiao, and W. Wang. Knowledgept: Enhancing large language models with retrieval and storage access on knowledge bases, 2023.
- [19] F. Yin, J. Srinivasa, and K.-W. Chang. Characterizing truthfulness in large language model generations with local intrinsic dimension. ArXiv, abs/2402.18048, 2024.

REFIT: Reranker Relevance Feedback during Inference

Revanth Gangi Reddy¹ Pradeep Dasigi² Md Arafat Sultan³ Arman Cohan^{2,4}
Avirup Sil³ Heng Ji¹ Hannaneh Hajishirzi^{2,5}

¹University of Illinois at Urbana-Champaign ²Allen Institute for AI

³IBM Research AI ⁴Yale University ⁵University of Washington

{revanth3,hengji}@illinois.edu {pradeepd,armanc,hannah}@allenai.org
arafat.sultan@ibm.com avi@us.ibm.com

Abstract

Retrieve-and-rerank is a prevalent framework in neural information retrieval, wherein a bi-encoder network initially retrieves a pre-defined number of candidates (*e.g.*, $K=100$), which are then reranked by a more powerful cross-encoder model. While the reranker often yields improved candidate scores compared to the retriever, its scope is confined to only the top K retrieved candidates. As a result, the reranker cannot improve retrieval performance in terms of Recall@K. In this work, we propose to leverage the reranker to improve recall by making it provide relevance feedback to the retriever at *inference-time*. Specifically, given a test instance during inference, we distill the reranker’s predictions for that instance into the retriever’s query representation using a lightweight update mechanism. The aim of the distillation loss is to align the retriever’s candidate scores more closely with those produced by the reranker. The algorithm then proceeds by executing a second retrieval step using the updated query vector. We empirically demonstrate that this method, applicable to various retrieve-and-rerank frameworks, substantially enhances the retrieval recall across multiple domains, languages, and modalities.

A Introduction

Information Retrieval (IR) involves retrieving a set of candidates from a large document collection given a user query. The retrieved candidates may be further reranked to bring the most relevant ones to the top, constituting a typical retrieve-and-rerank (R&R) framework [1, 2]. Reranking generally improves the ranks of relevant candidates among those retrieved, thus improving on metrics such as Mean Reciprocal Rank (MRR) [3] and Normalized Discounted Cumulative Gain (nDCG) [4], which assign better scores when relevant results are ranked higher. However, retrieval metrics like Recall@K, which mainly evaluate the presence of relevant candidates in the top K retrieved results, remain unaffected. Increasing Recall@K can be key, especially when the retrieved results are used in downstream knowledge-intensive tasks [5] such as open-domain question answering [6–8], fact-checking [9], entity linking [10–12] and dialog generation [13, 14].

Most existing neural IR methods use a dual-encoder retriever [15, 16] and a subsequent cross-encoder reranker [17]. Dual-encoder¹ models leverage separate query and passage encoders and perform a late interaction between the query and passage output representations. This enables them to perform inference at scale as passage representations can be pre-computed. Cross-encoder models, on the other hand, accept the query and the passage together as input, leaving out scope for pre-computation. The cross-encoder typically provides better ranking than the dual-encoder—thanks to its more elaborate

¹We use the terms bi-encoder and dual-encoder interchangeably in this paper.

computation of query-passage similarity informed by cross-attention—but is limited to seeing only the retrieved candidates in an R&R framework.

Since the more sophisticated reranker often generalizes better at passage scoring than the simpler, but more efficient retriever, here we propose to use relevance feedback from the former to improve the quality of query representations for the latter directly *at inference*. Concretely, after the R&R pipeline is invoked for a test instance, we update the retriever’s corresponding query vector by minimizing a distillation loss that brings its score distribution over the retrieved passages closer to that of the reranker. The new query vector is then used to retrieve documents for the second time. This process effectively teaches the retriever how to rank passages like the reranker—a stronger model—for the given test instance. Our approach, REFIT², is lightweight as only the output query vectors (and no model parameters) are updated, ensuring comparable inference-time latency when incorporated into the R&R framework. Figure 47 shows a schematic diagram of our approach, which introduces a distillation and a second retrieval step into the R&R framework. By operating exclusively in the representation space—as we only update the query vectors—our framework yields a parameter-free and architecture-agnostic solution, thereby providing flexibility along important application dimensions, e.g., the language, domain, and modality of retrieval. We empirically demonstrate this effect by showing improvements in retrieval on multiple English domains, across 26 languages in multilingual and cross-lingual settings, and in different modalities such as text and video retrieval.

Our main contributions are as follows:

- We propose REFIT, an inference-time mechanism to improve the recall of retrieval in IR using relevance feedback from a reranker.
- Empirically, REFIT improves retrieval performance in multi-domain, multilingual, cross-lingual and multi-modal evaluation.
- The proposed distillation step is fast, considerably increasing recall without any loss in ranking performance over a standard R&R pipeline with comparable latency.

B Related Work

Pseudo-relevance feedback: Our method has similarities with Pseudo-Relevance Feedback (PRF) [18–20] in IR: [21, 22] use the retrieved documents to improve sparse approaches via query expansion or query term reweighting, [23, 24] score similarity between a target document and a top-ranked feedback document, while [25] train a separate query encoder that computes a new query embedding using the retrieved documents as additional input. In contrast, our approach does not require customized training

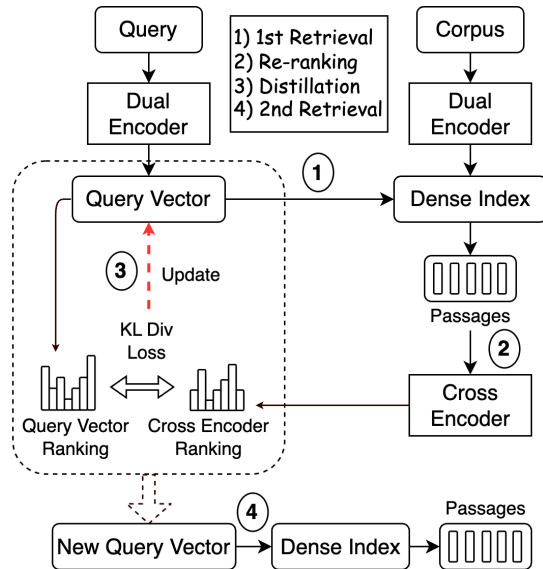


Figure 47: REFIT: The proposed method for reranker relevance feedback. We introduce an inference-time distillation process (step 3) into the traditional retrieve-and-rerank framework (steps 1 and 2) to compute a new query vector, which improves recall when used for a second retrieval step (step 4).

²REFIT stands for **R**eranker **F**eedback at **I**nference **T**ime

feedback models or availability of explicit feedback data, as we improve the query vector by directly distilling from the reranker’s output within an R&R framework.

Further, previous approaches to PRF have been dependent on the choice of retriever architecture and language; [25]’s PRF model is tied to the retriever used, [26] explore cross-lingual relevance feedback, but require feedback documents in target language and thereby could only apply to three languages, while [27] explore interpolating relevance feedback between dense and sparse approaches. On the other hand, our approach is independent of the choice of the retriever and reranker architecture, and can be used for neural retrieval in any domain, language or modality.

Distillation in Neural IR: Existing approaches primarily leverage reranker feedback *during training* of the dual-encoder retriever, to sample better negatives [28], for standard knowledge distillation of the cross-attention scores [29], to train smaller and more efficient rankers by distilling larger models [30], or to align the geometry of dual-encoder embeddings with that from cross-encoders [31]. Instead, we leverage distillation at inference time, updating only the query representation to replicate the cross-encoder’s scores for the corresponding test instance. A key implication of this design choice is that unlike existing methods, we keep the retriever parameters unchanged, meaning REFIT can be incorporated out-of-the-box into any neural R&R framework. In contrast, extending training-time distillation to new languages or modalities would require re-training the bi-encoder.

More recently, TOUR [32] has proposed test-time optimization of query representations with two variants: TOUR_{hard} and TOUR_{soft}. TOUR_{hard} optimizes the marginal likelihood of a small set of (pseudo) positive contexts. REFIT shares similarities with TOUR_{soft}, which uses the normalized scores of a cross-encoder over the retrieved results as soft labels. Crucially, TOUR relies on multiple iterations of relevance feedback via distillation, where each iteration runs until the top-1 retrieval result has the highest reranker score (in TOUR_{soft}) or is a pseudo-positive (in TOUR_{hard}). This makes inference highly computationally expensive, as each additional iteration involves labeling top- K retrieval results with a reranker and then retrieving again. REFIT improves efficiency over TOUR by requiring only a single iteration of feedback that simply updates the query vector for longer, foregoing additional retrieval and reranking steps. More specifics on the inference process of the two methods can be found in §E.4. TOUR was evaluated only on English phrase and passage retrieval tasks, while we demonstrate REFIT’s effectiveness in multidomain, multilingual and multimodal settings, with an empirical comparison with TOUR in §E.4.

C Method

Here we discuss the standard retrieve-and-rerank (R&R) framework for IR (§C.1) and how our proposal fits into it (§C.2). While our approach can be applied to any R&R framework, we consider a text-based retriever and reranker for simplicity while elaborating our method. A multi-modal R&R framework is described in §E.3.

C.1 Retrieve-and-Rerank

R&R for IR consists of a first-stage retriever and a second-stage reranker. Modern neural approaches typically use a dual-encoder model as the retriever and a cross-encoder for reranking.

The Retriever: The dual-encoder retriever model is based on a Siamese neural network [33], containing separate Bert-based [34] encoders $E_Q(\cdot)$ and $E_P(\cdot)$ for the query and the passage, respectively. Given a query q and a passage p , a separate representation is obtained for each, such as the CLS output or a pooled

Algorithm 1: REFIT

Input: Query q and its representation Q_q , retrieved passages P and their representations \hat{P} .

Output: Updated query representation $Q_{q,n}$

- 1: Initialize query vector $Q_{q,0} = Q_q$
 - 2: Compute reranker distribution $D_{CE}(q, P)$ (Eq. 10) **for** i **in** 0 **to** n **do**
 - 3: Compute retriever distribution $D_{Q_{q,i}}(\hat{P})$ (Eq. 11)
 - 4: Compute loss \mathcal{L} (Eq. 12)
 - 5: Update $Q_{q,i+1} = Q_{q,i} - \alpha \frac{\partial}{\partial Q_{q,i}} \mathcal{L}$
 - 6:
 - 7: **return** $Q_{q,n}$
-

representation of the individual token outputs from $E_Q(q)$ and $E_P(p)$. The question-passage similarity $sim(q, p)$ is computed as the dot product of their corresponding representations: query/passage.

$$Q_q = Pool(E_Q(q)) \tag{6}$$

$$P_p = Pool(E_P(p)) \tag{7}$$

$$sim(q, p) = S(Q_q, P_p) = Q_q^T P_p \tag{8}$$

Since Eq. 8 is decomposable, the representations of all passages in the retrieval corpus can be pre-computed and stored in a dense index [35]. During inference, given a new query, the top K most relevant passages are retrieved from the index via approximate nearest-neighbor search.

The Reranker: The cross-encoder reranker model uses a Bert-based encoder $E_R(\cdot)$, which takes the query q and a corresponding retrieved passage p together as input and outputs a similarity score. A feed-forward layer F is used on top of the CLS output from $E_R(\cdot)$ to compute a single logit, which is used as the final reranker score $R(q, p)$. The top K retrieved passages are then ranked based on their corresponding reranker scores.

$$R(q, p) = F(CLS(E_R(q, p))) \tag{9}$$

C.2 Reranker Relevance Feedback

The main idea underlying our proposal is to compute an improved query representation for the retriever using feedback from the more powerful reranker. More specifically, we perform a lightweight inference-time distillation of the reranker’s knowledge into a new query vector.

Given an input query q during inference, we use the following output provided by the R&R pipeline:

- Query representation Q_q from the retriever.
- Retrieved passages $P = \{p_1, p_2, \dots, p_K\}$ and their representations $\hat{P} = [P_{p_1}, P_{p_2}, \dots, P_{p_K}]$ from the retriever.
- The reranking scores $R(q, P) = [R(q, p_1), \dots, R(q, p_K)]$.

Step (Device)	Retrieve & Rerank		REFIT (K=100)
	K=100	K=125	
1st Retrieval (CPU)	40ms	40ms	40ms
Rerank (CPU)	1540ms	1925ms	1540ms
Rerank (GPU)	360ms	450ms	360ms
Distillation (CPU)	-	-	30ms
2nd Retrieval (CPU)	-	-	40ms
Total (CPU)	1580ms	1965ms	1650ms
Total (GPU)	400ms	490ms	470ms

Table 23: Comparison of inference times (in milliseconds) for different approaches, utilizing both CPU-only and GPU configurations (when reranking K passages).

Note that \hat{P} above is directly obtained from the passage index and is not computed during inference.

The proposed reranker feedback mechanism begins with using the reranking scores $R(q, P)$ to compute a cross-encoder ranking distribution $D_{CE}(q, P)$ over passages P as follows:

$$D_{CE}(q, P) = \text{softmax}([R(q, p_1), \dots, R(q, p_K)]) \quad (10)$$

The query and passage representations from the retriever are used to compute a similar distribution $D_{Q_q}(\hat{P})$ over P :

$$D_{Q_q}(\hat{P}) = \text{softmax}([Q_q^T P_{p_1}, \dots, Q_q^T P_{p_K}]) \quad (11)$$

Next, we compute the loss as the KL-divergence between the retriever and reranker distributions:

$$\mathcal{L} = D_{KL}(D_{CE}(q, P) || D_{Q_q}(\hat{P})) \quad (12)$$

which is then used to update the query vector via gradient descent. The query vector update process is repeated for n times, where n is a hyper-parameter. A schematic description of the process can be found in Algorithm 1.

Finally, the updated query vector $Q_{q,n}$ is used for a second-stage retrieval from the passage index. From dual-encoder retrieval with the updated $Q_{q,n}$, we aim to achieve better recall than with the initial Q_q , while obtaining a ranking performance that is comparable with that of the reranker.

D Experimental Setup

D.1 Distillation Process

We observe that the output scores from the dual-encoder and the cross-encoder models are not bounded to specific intervals. Hence, we do min-max normalization separately on the query vector’s scores $Q_q^T \hat{P}$ (from the dual-encoder) and the cross encoder’s scores $R(q, P)$ to bring the two scoring distributions closer. Further, the cross-encoder tends to have peaky scoring distributions, hence we use a temperature T ($= 2$ after tuning) while computing the softmax $D_{CE}(q, P)$ over the cross-encoder scores. After tuning on the MS Marco dev set, we set the number of updates $n=100$ with learning rate $\alpha=0.005$.

	BM25	ANCE	RocketQA v1	RocketQA v2	RocketQA v1+Rerank	RocketQA v1+REFIT	Contriever	Contriever + Rerank	Contriever + REFIT
MS MARCO	65.8	85.2	88.4	88.7	89.4	90.0*	89.1	89.9	90.5*
Trec-COVID	49.8	45.7	48.5	46.4	52.0	52.9	40.7	43.8	51.5*
NFCorpus	25.0	23.2	26.9	25.9	27.4	29.2*	30.0	29.5	31.9*
NQ	76.0	83.6	91.1	89.8	91.8	92.7*	92.5	93.3	94.2*
HotpotQA	74.0	57.8	69.8	67.7	71.4	73.3*	77.7	78.6	80.4*
FiQA	53.9	58.1	63.6	61.2	64.3	63.8	65.6	65.9	65.6
DBPedia	39.8	31.9	45.7	43.4	47.6	50.2*	54.1	56.0	57.3*
Scidocs	35.6	26.9	31.8	29.3	33.1	35.5*	37.8	38.3	40.1*
FEVER	93.1	90.0	92.6	92.5	92.8	93.7*	94.9	95.3	95.5*
Climate-FEVER	43.6	44.5	47.4	48.7	49.3	53.6*	57.4	59.0	59.5
Scifact	90.8	81.6	88.1	85.4	89.0	89.9*	94.7	94.4	95.2*
Average	58.9	57.1	63.1	61.7	64.4	65.9*	66.8	67.6	69.0*

Table 24: Recall@100 (in %) on the English BEIR benchmark. Performance of REFIT is shown for different choices of underlying retrievers. RocketQAv2 [39] corresponds to a training-time distillation baseline. Improvements marked with * are statistically significant at $p < 0.05$ as per paired t-test.

D.2 Rerank Baseline

REFIT introduces the additional overhead of distillation and a second retrieval step into the R&R framework. We note that distillation latency (in Algorithm 1) is linear in the number of updates n . Table 23 compares the inference latency of our method with that of standard R&R, assuming $K=100$ passages are to be reranked and $n=100$ updates are used during distillation. We highlight that our distillation process is lightweight and takes just 30ms on a CPU. We see that the additional distillation and retrieval steps increase the latency of inference by roughly 17.5% when using a GPU (or 4.4% for CPU);³ in that same amount of time, vanilla R&R can process a total of 125 passages on the GPU (see Table 23), to potentially increase Recall@100. Hence, and for fair comparison, we evaluate against a *Rerank* baseline that is allowed to retrieve and rerank 125 passages. We note that both REFIT and the *Rerank* baseline use the same retriever and reranker, and are evaluated on Recall@100.

D.3 Retriever and Reranker

We use Contriever [36] as the underlying retriever (unless otherwise mentioned), which has been pre-trained with an unsupervised contrastive learning objective on a large-scale collection of Wikipedia and CCNet documents. Contriever is a dual-encoder retriever that outperforms traditional term-matching methods, BM25 and recent dense approaches e.g. DPR [15] and ANCE [37]. For retrieval in both English and other languages, we use the publicly available version of Contriever, fine-tuned on MS MARCO [38]. Our English⁴ and multilingual⁵ rerankers are based on sentence transformers.

E Results

E.1 English Retrieval in Multiple Domains

We evaluate English retrieval performance on the BEIR benchmark [40], comprising training and in-domain test instances from MS MARCO and out-of-domain evaluation data from a number of scientific,

³24-core AMD EPYC 7352 CPU and 80GB A100 GPU.

⁴cross-encoder/ms-marco-MiniLM-L-6-v2

⁵cross-encoder/mmarco-mMiniLMv2-L12-H384-v1

	mBERT	XLM-R	Contriever	Rerank	REFIT
Arabic	81.1	79.9	88.7	89.5	90.9*
Bengali	88.7	84.2	91.4	91.4	95.9*
English	77.8	73.1	77.2	78.7	81.8*
Finnish	74.2	81.6	88.1	88.9	91.0*
Indonesian	81.0	87.4	89.8	90.5	93.7*
Japanese	76.1	70.9	81.7	82.5	85.2*
Korean	66.7	71.1	78.2	81.0	80.2
Russian	77.6	74.1	83.8	85.7	87.3
Swahili	74.1	73.9	91.4	92.0	90.5
Telugu	89.5	91.2	96.6	97.0	97.5
Thai	57.8	89.5	90.5	91.6	93.3*
Average	76.8	79.7	87.0	88.1	89.7*

Table 25: Recall@100 (in %) on the multilingual Mr.TyDi benchmark. Rerank and REFIT use Contriever as the underlying retriever. * corresponds to statistical significance at $p < 0.05$ (paired t-test).

biomedical, financial, and Wikipedia-based retrieval datasets⁶.

Firstly, we compare our inference-time distillation approach against a training-time distillation method. We use RocketQAv1 [41] as the underlying retrieval model and RocketQAv2 [42] as the retriever distilled at training time from the cross-encoder. We also compare with a *Rerank* (K=125) baseline, which improves Recall@100 by reranking the top 125 passages (retrieved by RocketQAv1). Moreover, we also demonstrate the effectiveness of REFIT with a different underlying retrieval model, in this case, Contriever.

Table 24 shows Recall@100 results on the BEIR benchmark. Firstly, we see that REFIT consistently outperforms all baselines. Next, RocketQAv2 shows improvement over RocketQAv1 on MS MARCO, which is the dataset used for training-time distillation of RocketQAv2. However, RocketQAv2’s performance degrades on out-of-domain datasets from the BEIR benchmark. This is unsurprising, since the training-time distillation approach is limited to the bi-encoder seeing the cross-encoder’s relevance labels only in the source domain, i.e. the domain used for training (MS MARCO in this case). As a result, the training-time distillation approach may not generalize well to unseen domains (BEIR in this case). In contrast, REFIT offers the key advantage of learning from target-domain pseudo labels provided by the reranker *at inference*, which yields improved out-of-domain generalization.

E.2 Retrieval in More Languages

Multilingual Retrieval

We also evaluate on Mr. TyDi [43], a multilingual IR benchmark derived from TyDi QA [44], where given a question in one of 11 languages, the goal is to retrieve candidates from a pool of Wikipedia documents in the same language. Our underlying retriever is the multilingual version of Contriever. Other baseline retrieval models are mBERT and XLM-R [45], in addition to the *Rerank* (K=125) baseline. Table 25 shows Recall@100 for the different systems on Mr.TyDi. Here again, REFIT yields significant improvement over all baselines on most languages.

Cross-lingual Retrieval

For our cross-lingual experiments, we used the MKQA benchmark [46]. MKQA involves retrieving passages from the English Wikipedia corpus for questions that are posed in 26 different languages.

⁶We omit some datasets due to license & versioning issues.

	avg	en	ar	fi	ja	ko	ru	es	sv	he	th	da	de	fr
mBERT	57.9	74.2	44.0	51.7	55.7	48.2	57.4	63.9	62.7	46.8	51.7	63.7	59.6	65.2
XLM-R	59.2	73.4	42.4	57.7	53.1	48.6	58.5	62.9	67.5	46.9	61.5	66.9	60.9	62.4
Contriever	65.6	75.6	53.3	66.6	60.4	55.4	64.7	70.0	70.8	59.6	63.5	72.0	66.6	70.1
Rerank	66.4	76.0	54.5	67.5	61.5	56.7	65.8	70.5	71.6	60.8	64.9	72.7	67.5	70.6
REFIT	68.2	76.6	58.0	68.8	64.7	59.3	68.4	72.5	73.1	62.9	66.5	74.1	70.1	72.5
	it	nl	pl	pt	hu	vi	ms	km	no	tr	zh-cn	zh-hk	zh-tw	
mBERT	64.1	66.7	59.0	61.9	57.5	58.6	62.8	32.9	63.2	56.0	58.4	59.3	59.3	
XLM-R	58.1	66.4	61.0	62.0	60.1	62.4	66.1	46.6	65.9	60.6	55.8	55.5	55.7	
Contriever	70.3	71.4	68.8	68.5	66.7	67.8	71.6	37.8	71.5	68.7	64.1	64.5	64.3	
Rerank	70.8	72.0	69.9	69.3	67.5	68.7	72.0	38.6	72.3	69.3	65.1	65.4	65.2	
REFIT	72.4	73.6	71.1	71.5	68.9	70.5	73.3	39.9	73.3	70.7	67.5	67.4	66.9	

Table 26: Recall@100 (in %) on the cross-lingual MKQA benchmark. Rerank and REFIT use Contriever as the underlying retriever. All improvements are statistically significant at $p < 0.05$ (paired t-test).

Following [36], we discard unanswerable questions and questions with a yes/no answer or a long answer, leaving 6,619 queries per language in the final test set. Table 26 compares Recall@100 of different models on MKQA. REFIT again outperforms, leading the nearest baseline (*Rerank*) by about 2 points on average, and with improvements on all 26 MKQA languages.

E.3 Multi-modal Retrieval

A key advantage of REFIT is that it can operate independently of the choice of architecture for the bi-encoder and the cross-encoder, and is therefore not limited to working on only text input. To demonstrate this, we apply our method to retrieval in a multi-modal setting. Specifically, we consider text-to-video retrieval, which involves retrieving videos that are relevant to a given textual query.

The retriever and reranker for this experiment are based on BLIP [47], a state-of-the-art vision-language model that comprises two unimodal encoders and an image-grounded text encoder. The unimodal encoders encode image and text separately, akin to dual-encoders in text-to-text retrieval, and are trained with an Image-Text Contrastive (ITC) loss. The image-grounded text encoder injects visual information into the text encoder by incorporating a cross-attention layer, similar to a text-to-text cross-encoder, and is trained with an Image-Text Matching (ITM) loss. We refer the reader to [47] for a more detailed description of BLIP’s architecture and the pre-training objectives. BLIP can thus be used for retrieval with the unimodal encoders (which we refer to as $BLIP_{ITC}$), and for reranking with the image-grounded text encoder (which we refer to as $BLIP_{ITM}$). We use the output from $BLIP_{ITM}$ as the reranker distribution, which is then used to compute the distillation loss for updating the query representation that is output by $BLIP_{ITC}$.

We evaluate using Recall@100 on the MSRVT [48] text-to-video retrieval dataset, with $BLIP_{ITC}$ [47] being our primary retrieval-only baseline along with other baselines taken from [47]. The *Rerank* baseline uses $BLIP_{ITM}$ to rerank $K=125$ videos retrieved by $BLIP_{ITC}$. Table 27 compares performance on the 1k test split of MSRVT. We see that $BLIP_{ITM}$ yields better ranking (as evident from higher Recall@10) than $BLIP_{ITC}$ as expected, but shows only minor gains in Recall@100. Crucially, REFIT improves Recall@100 over the already strong $BLIP_{ITC}$ retriever, without a noticeable drop in Recall@10 compared to the $BLIP_{ITM}$ reranker.

Method	R@10	R@100
MIL-NCE	32.4	-
VideoCLIP	30.0	-
FiT	51.6	-
BLIP _{ITC}	69.0	92.1
Rerank (BLIP _{ITM})	74.6	92.3
REFIT	74.7	92.9

Table 27: Recall of text-to-video retrieval methods on the MSRVTT benchmark.

	NQ	EntityQ	BEIR	Mr.TyDi
Retrieve	86.1	70.1	66.8	87.0
Rerank	86.8	71.2	67.6	88.1
TOUR _{hard}	87.0 ⁺	71.9 ⁺	68.4	88.7
TOUR _{soft}	87.2 ⁺	72.5 ⁺	68.1	88.1
REFIT	87.6	72.6	69.2	89.7

Table 28: Recall@100 numbers for comparison of REFIT with both variants of TOUR. ⁺ corresponds to numbers directly taken from [32].

E.4 Comparison with TOUR

In this section, we compare the performance of REFIT with TOUR on the passage retrieval benchmarks used in [32], NQ [49] and EntityQuestions [50] as well as the multidomain BEIR and multilingual Mr.TyDi benchmarks. For NQ and EntityQuestions, we use the same retriever [15] and reranker [51] as in [32]. The retriever and the reranker for BEIR and Mr.TyDi are the same as described in §D.3. In Table 28, we can see that REFIT consistently outperforms both TOUR variants across various datasets. We believe that the lower performance of TOUR can be attributed to its early stopping criterion for distillation updates. Specifically, TOUR performs relevance feedback for 3 iterations, wherein in each iteration, distillation into the query vector continues until the top-1 retrieval result has the highest reranker score (for TOUR_{soft}) or is a pseudo-positive (for TOUR_{hard}). In contrast, REFIT makes more distillation updates but for only one iteration (we do $n = 100$ updates, which has been tuned). This makes it also considerably faster than TOUR, as each additional iteration of relevance feedback in TOUR comes with a high computational overhead (§B). We show in §F.4 that REFIT can further benefit from multiple rounds of relevance feedback with continuous improvements over the course of three iterations.

E.5 REFIT for multi-vector dense retrieval

Our experiments thus far have been focused on single-vector dense retrieval, where queries and passages are encoded as individual vectors. Multi-vector retrieval models like ColBERT [16, 52], on the other hand, compute token-level query and passage representations, subsequently employing a late-interaction mechanism for scoring. This section explores the application of REFIT to multi-vector retrieval, specifically, ColBERTv2 [52]. In this case, distillation (Step 3 in Figure 47) updates embeddings of individual tokens in the query. We present results in Table 29 on a subset⁷ of the BEIR dataset, which clearly show that REFIT can be effectively extended to multi-vector dense retrieval, as it consistently surpasses the performance of the ColBERTv2 retriever and outperforms the Rerank baseline (with $K = 125$) in most cases. Notably, the training of ColBERTv2 [52] involved the use of a reranker’s scores for supervision; our results in this section thus reinforce the finding of §E.1 that REFIT’s inference-time distillation can be superior to ordinary knowledge distillation during training.

	ColBERTv2	Rerank	REFIT
NFCorpus	27.7	28.0	28.8
FiQA	62.8	63.7	64.3
Scidocs	35.8	36.6	38.5
Scifact	89.4	90.2	90.1

Table 29: Recall@100 (in %) on a subset of the English BEIR benchmark, with Rerank and REFIT using ColBERTv2 as the underlying retriever.

⁷Owing to the substantially larger index size inherent to multi-vector dense retrieval, we restrict this study to subsets of BEIR with $< 100k$ passages in the retrieval corpus.

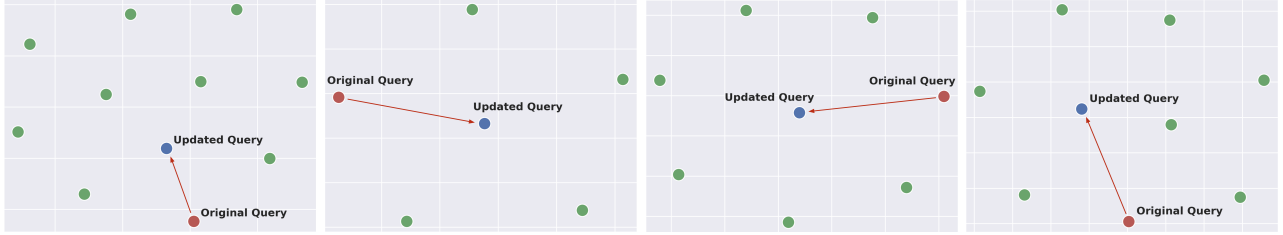


Figure 48: t-SNE plots for some examples from BEIR, with the query vectors shown alongside the corresponding positive passages. The updated query vectors after REFIT are now closer to the positive passages (in green).

Query	Initial Retrieval (within Reranker Top-5)	Newly Retrieved Positive
treating tension headaches without medication	Most intermittent tension-type headaches are easily treated with over-the-counter medications, including: 1 Aspirin. 2 Ibuprofen (Advil, Motrin IB, others) 3 Acetaminophen (Tylenol, others)	Instead of popping a pill when you get a headache, toss some almonds. For everyday tension-type headaches , almonds can be a natural remedy and a healthier alternative to other medicine.
who drives the number 95 car in nascar	On October 2013, it was announced that McDowell would be moving to Leavine Family Racing's No. 95 Ford for the 2014 NASCAR Sprint Cup Series season. McDowell failed to qualify for the Daytona 500.	Michael Christopher McDowell is an American professional stock car racing driver. He currently competes full-time in the Monster Energy NASCAR Cup Series , driving the No. 95 Chevrolet SS for Leavine Family Racing .
who plays addison shepherd on grey's anatomy	In 2005, she was cast in her breakout role in the ABC series Grey's Anatomy, as Dr. Addison Montgomery , the estranged spouse of Derek Shepherd.	Kathleen Erin Walsh is an American actress and businesswoman. Her roles include Dr. Addison Montgomery on the ABC television dramas Grey's Anatomy and Private Practice.

Table 30: Examples of how initial retrievals highly ranked (top-5) by the reranker (middle) helps retrieve new positives (right) via the updated query vector, due to important lexical and semantic overlap (highlighted in green). The text that contains the answer to the query is shown in red.

F Discussion and Analysis

This section describes additional experiments, providing further insights into REFIT.

F.1 Query vectors: the original and the new

To better understand how the updated query vector after reranker relevance feedback improves recall, we take a closer look at the query and passage vectors computed for a set of BEIR examples. Figure 48 shows t-SNE plots for four such examples, where each dot represents a vector, and the distance between any two points is their cosine distance. As the figure shows, the reranker feedback brings the query vector in each case closer to the corresponding positive passage vectors, making the query align with an increased number of relevant passages and consequently improving recall. Across different datasets in BEIR, we observed that the new query vector is also closer to the initially retrieved positives by 5-16%.

We observe that the new positives discovered by the updated query vector are closest to a passage in the reranker’s top 5 in 26% of the cases (38% for top 10; 55% for top 20), confirming an effective transfer of the reranker’s knowledge into the query vector. Table 30 provides some examples, showing how specific words and phrases in a passage within the reranker top-5 help retrieve additional candidates with lexical/semantic overlap (highlighted in green) via relevance feedback. Interestingly, in the fourth example, an incorrect passage highly scored by the reranker leads to the subsequent retrieval of an actual positive candidate.

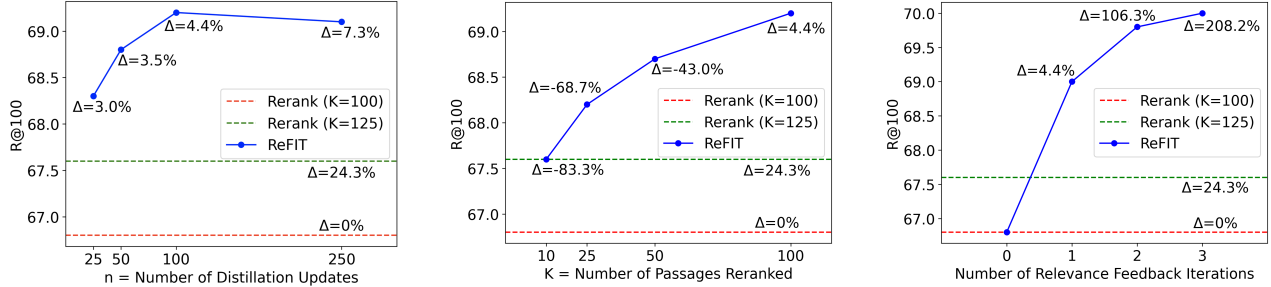


Figure 49: Variation of REFIT performance with distillation updates n (left), reranked passages K used for distillation supervision (center) and relevance feedback iterations (right). Δ corresponds to change in latency with respect to the standard R&R framework with $K=100$ (CPU-only configuration).

F.2 How much additional latency does our approach introduce?

Our proposed method introduces a distillation and an additional retrieval step into the standard R&R framework. While retrieval takes constant time with respect to the number of updates n in Algorithm 1, the latency of distillation is directly proportional to n . Figure 49 (left) demonstrates the effect of varying n on both the latency and performance of our approach. The extra latency is computed with respect to a standard R&R framework that runs with $K=100$. With a mere 4.4% increase in latency (for when $n=100$), our method produces a gain that is significantly larger than a more computationally expensive reranking of $K=125$ candidates which in turn corresponds to 24.3% increase in latency on a CPU. Thereby, we demonstrate that, under latency constraints, our approach can be made faster by simply lowering the number of updates, while still surpassing the conventional strategy of reranking a larger pool of candidates for improving recall.

F.3 How do smaller K values affect results?

Our experiments described thus far are run in the standard setting of $K=100$: 100 passages are retrieved, reranked and subsequently used to distill the reranker score distribution into the new query. Here we investigate how REFIT performs as we vary K . Smaller values of K correspond to a faster R&R pipeline (as lower number of candidates are reranked), but it comes at the expense of the target teacher distribution now providing lesser supervision. Figure 49 (center) shows Recall@100 of the post-relevance feedback retrieval step on BEIR for different values of K . While a higher K expectedly leads to a higher recall in general, we observe performance improvements over directly reranking 125 passages, even when considerably smaller number of passages are used for distillation. Our approach can thus be easily tuned to achieve different accuracy-speed trade-offs depending on the requirements of the target application.

F.4 Can multiple iterations of relevance feedback further improve results?

Our relevance feedback approach improves recall when the updated query vector is used for a second retrieval step. Here we examine if further improvements are possible from more iterations of relevance feedback, i.e., running the following operations in a loop: (1) rerank the retrieval results from the previous iteration, (2) update the query vector via distillation from the reranker distribution, and (3) retrieve again. We note that this experiment operates under the assumption of a relaxed time budget, as the computationally expensive reranker must be executed N times. Figure 49 (right) shows performance on BEIR from N iterations of relevance feedback, with $N = 0$ corresponding to baseline retrieval. We can see that recall improves with each additional round of relevance feedback; the biggest gain comes in the first round ($N = 1$) and performance saturates after $N = 2$.

F.5 Further Discussion

The curious case of zero initial positives:

In §F.1, we presented an example where our method leverages a close negative among the initially retrieved candidates to later retrieve a positive passage. We find that in 24% of the cases where the first-stage retriever retrieves no positive passages, our method can improve recall in a similar fashion. Among all cases where recall improves, however, 75% have at least one positive in the top retrieved results. These results indicate that while the presence of positive candidates in the initial retrieval is useful, our relevance feedback approach can also generally leverage informative negatives to update the query vector in the right direction.

Choice of Reranker:

In the experiments comparing our approach to the R&R framework, we used an efficient (yet high-performing) reranker both in the baseline model and as the teacher model for distillation. Would the results have been different if we used a more powerful (but computationally expensive) reranker instead? To find an answer, it is essential to note that the final recall of an R&R engine is inherently limited by the underlying retriever. For instance, the Recall@100 of an R&R pipeline with $K=125$ cannot exceed the Recall@125 of the underlying retriever, irrespective of the quality of the reranker. The Recall@100 of REFIT (BEIR: 69.2, Mr. TyDi: 89.7, MKQA: 68.2 and MSRVT: 92.9) is consistently higher than the Recall@125 of the baseline retriever (BEIR: 68.9, Mr. TyDi: 88.2, MKQA: 66.9 and MSRVT: 92.8). These results clearly suggest that even the best reranker baseline would fail to attain the recall of our method. Further, we can expect a better reranker to improve the recall of REFIT since leveraging a stronger teacher model for distillation should lead to a better student (retriever query vector).

G Conclusion and Future Work

We demonstrate that query representations can be improved using feedback from a cross-encoder reranker *at inference time* for better performance of dual-encoder retrieval. This work proposes for distillation using relevance feedback from the reranker as a better and faster alternative to the traditional strategy of reranking a larger pool of candidates for improving recall. REFIT is lightweight and improves retrieval accuracy across different domains, languages and modalities over a state-of-the-art retrieve-and-rerank pipeline with comparable latency. Future work will focus on the potential integration of textual relevance feedback from large language models (LLMs). Additionally, a promising area of exploration lies in enhancing the interpretability by examining how relevance feedback influences the significance of individual query terms within the query representation.

H Limitations

REFIT introduces an additional latency into a traditional retrieve-and-rerank framework. The distillation time is only dependent on the number of updates, and is unaffected by the model architecture and number of retrieved passages; the overall additional latency (as per Table 23) amounts to an extra 17.5% on GPU (or 4.4% on CPU) when the number of retrieved passages $K=100$. However, it is noteworthy that REFIT remains faster and exhibits superior performance compared to the standard approach of reranking a larger pool of candidates for improving recall. Moreover, the efficacy of our approach is contingent upon the reranker providing a better ranking than the retriever. We anticipate that our method might provide minimal gains in situations where the retriever performs similar to the reranker.

References

- [1] S. WANG, M. YU, J. JIANG, W. ZHANG, X. GUO, S. CHANG, Z. WANG, T. KLINGER, G. TESAURO, and M. CAMPBELL, “Evidence aggregation for answer re-ranking in open-domain question answering.(2018),” in Proceedings of the 6th International Conference on Learning Representation, Vancouver, Canada, 2018 April 30-May, vol. 3, pp. 1–14.
- [2] M. Hu, Y. Peng, Z. Huang, and D. Li, “Retrieve, read, rerank: Towards end-to-end multi-document reading comprehension,” in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2285–2295, 2019.
- [3] N. Craswell, Mean Reciprocal Rank, pp. 1703–1703. Boston, MA: Springer US, 2009.
- [4] K. Järvelin and J. Kekäläinen, “Cumulated gain-based evaluation of ir techniques,” ACM Transactions on Information Systems (TOIS), vol. 20, no. 4, pp. 422–446, 2002.
- [5] F. Petroni, A. Piktus, A. Fan, P. Lewis, M. Yazdani, N. De Cao, J. Thorne, Y. Jernite, V. Karpukhin, J. Maillard, et al., “Kilt: a benchmark for knowledge intensive language tasks,” in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2523–2544, 2021.
- [6] D. Chen, A. Fisch, J. Weston, and A. Bordes, “Reading wikipedia to answer open-domain questions,” in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1870–1879, 2017.
- [7] D. Chen and W.-t. Yih, “Open-domain question answering,” in Proceedings of the 58th annual meeting of the association for computational linguistics: tutorial abstracts, pp. 34–37, 2020.
- [8] R. Gangi Reddy, B. Iyer, M. A. Sultan, R. Zhang, A. Sil, V. Castelli, R. Florian, and S. Roukos, “Synthetic target domain supervision for open retrieval qa,” in Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1793–1797, 2021.
- [9] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, “Fever: a large-scale dataset for fact extraction and verification,” in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 809–819, 2018.
- [10] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum, “Robust disambiguation of named entities in text,” in Proceedings of the 2011 conference on empirical methods in natural language processing, pp. 782–792, 2011.
- [11] A. Sil and A. Yates, “Re-ranking for joint named-entity recognition and linking,” in Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 2369–2374, 2013.
- [12] A. Sil, G. Kundu, R. Florian, and W. Hamza, “Neural cross-lingual entity linking,” in Proceedings of the AAI Conference on Artificial Intelligence, vol. 32, 2018.
- [13] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, “Wizard of wikipedia: Knowledge-powered conversational agents,” in International Conference on Learning Representations, 2018.

- [14] M. Komeili, K. Shuster, and J. Weston, “Internet-augmented dialogue generation,” in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 8460–8478, 2022.
- [15] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, “Dense passage retrieval for open-domain question answering,” in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6769–6781, 2020.
- [16] O. Khattab and M. Zaharia, “Colbert: Efficient and effective passage search via contextualized late interaction over bert,” in Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pp. 39–48, 2020.
- [17] R. Nogueira and K. Cho, “Passage re-ranking with bert,” arXiv preprint arXiv:1901.04085, 2019.
- [18] J. Rocchio, “Relevance feedback in information retrieval,” The Smart retrieval system-experiments in automatic document processing, pp. 313–323, 1971.
- [19] Y. Lv and C. Zhai, “Adaptive relevance feedback in information retrieval,” in Proceedings of the 18th ACM conference on Information and knowledge management, pp. 255–264, 2009.
- [20] H. Li, A. Mourad, B. Koopman, and G. Zuccon, “How does feedback signal quality impact effectiveness of pseudo relevance feedback for passage retrieval?,” arXiv preprint arXiv:2205.05888, 2022.
- [21] M. Bendersky, D. Metzler, and W. B. Croft, “Parameterized concept weighting in verbose queries,” in Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp. 605–614, 2011.
- [22] J. Xu and W. B. Croft, “Query expansion using local and global document analysis,” in Acm sigir forum, vol. 51, pp. 168–175, ACM New York, NY, USA, 2017.
- [23] C. Li, Y. Sun, B. He, L. Wang, K. Hui, A. Yates, L. Sun, and J. Xu, “Nprf: A neural pseudo relevance feedback framework for ad-hoc information retrieval,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4482–4491, 2018.
- [24] Z. Zheng, K. Hui, B. He, X. Han, L. Sun, and A. Yates, “Bert-qe: Contextualized query expansion for document re-ranking,” in Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 4718–4728, 2020.
- [25] H. Yu, C. Xiong, and J. Callan, “Improving query representations for dense retrieval with pseudo relevance feedback,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3592–3596, 2021.
- [26] R. Chandradevan, E. Yang, M. Yarmohammadi, and E. Agichtein, “Learning to enrich query representation with pseudo-relevance feedback for cross-lingual retrieval,” in Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1790–1795, 2022.
- [27] H. Li, S. Wang, S. Zhuang, A. Mourad, X. Ma, J. Lin, and G. Zuccon, “To interpolate or not to interpolate: Prf, dense and sparse retrievers,” arXiv preprint arXiv:2205.00235, 2022.

- [28] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, “Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering,” in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5835–5847, 2021.
- [29] G. Izacard and E. Grave, “Distilling knowledge from reader to retriever for question answering,” in International Conference on Learning Representations, 2020.
- [30] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, and A. Hanbury, “Improving efficient neural ranking models with cross-architecture knowledge distillation,” arXiv preprint arXiv:2010.02666, 2020.
- [31] Y. Wang, J. Bai, Y. Wang, J. Zhang, W. Rong, Z. Ji, S. Wang, and J. Xiao, “Enhancing dual-encoders with question and answer cross-embeddings for answer retrieval,” in Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 2306–2315, 2021.
- [32] M. Sung, J. Park, J. Kang, D. Chen, and J. Lee, “Optimizing test-time query representations for dense retrieval,” in Findings of the Association for Computational Linguistics: ACL 2023 (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 5731–5746, Association for Computational Linguistics, July 2023.
- [33] D. Chicco, “Siamese neural networks: An overview,” Artificial Neural Networks, pp. 73–94, 2021.
- [34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186, 2019.
- [35] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with gpus,” IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535–547, 2019.
- [36] G. Izacard, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, “Unsupervised dense information retrieval with contrastive learning,” arXiv preprint arXiv:2112.09118, 2021.
- [37] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk, “Approximate nearest neighbor negative contrastive learning for dense text retrieval,” in International Conference on Learning Representations, 2020.
- [38] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, “Ms marco: A human generated machine reading comprehension dataset,” in CoCo@ NIPs, 2016.
- [39] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, “Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking,” in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 2825–2835, 2021.
- [40] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, “BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models,” in Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), 2021.
- [41] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, “RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering,” in Proceedings of the 2021 Conference of the North American Chapter of the Association for

Computational Linguistics: Human Language Technologies (K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, eds.), (Online), pp. 5835–5847, Association for Computational Linguistics, June 2021.

- [42] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, “RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking,” in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Online and Punta Cana, Dominican Republic), pp. 2825–2835, Association for Computational Linguistics, Nov. 2021.
- [43] X. Zhang, X. Ma, P. Shi, and J. Lin, “Mr. tydi: A multi-lingual benchmark for dense retrieval,” in Proceedings of the 1st Workshop on Multilingual Representation Learning, pp. 127–137, 2021.
- [44] J. H. Clark, E. Choi, M. Collins, D. Garrette, T. Kwiatkowski, V. Nikolaev, and J. Palomaki, “Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages,” Transactions of the Association for Computational Linguistics, vol. 8, pp. 454–470, 2020.
- [45] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, É. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, “Unsupervised cross-lingual representation learning at scale,” in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451, 2020.
- [46] S. Longpre, Y. Lu, and J. Daiber, “Mkqa: A linguistically diverse benchmark for multilingual open domain question answering,” Transactions of the Association for Computational Linguistics, vol. 9, pp. 1389–1406, 2021.
- [47] J. Li, D. Li, C. Xiong, and S. Hoi, “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation,” in ICML, 2022.
- [48] J. Xu, T. Mei, T. Yao, and Y. Rui, “Msr-vtt: A large video description dataset for bridging video and language,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5288–5296, 2016.
- [49] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al., “Natural questions: a benchmark for question answering research,” Transactions of the Association for Computational Linguistics, vol. 7, pp. 453–466, 2019.
- [50] C. Sciavolino, Z. Zhong, J. Lee, and D. Chen, “Simple entity-centric questions challenge dense retrievers,” in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, (Online and Punta Cana, Dominican Republic), pp. 6138–6148, Association for Computational Linguistics, Nov. 2021.
- [51] M. Fajcik, M. Docekal, K. Ondrej, and P. Smrz, “R2-D2: A modular baseline for open-domain question answering,” in Findings of the Association for Computational Linguistics: EMNLP 2021, (Punta Cana, Dominican Republic), pp. 854–870, Association for Computational Linguistics, Nov. 2021.
- [52] K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia, “ColBERTv2: Effective and efficient retrieval via lightweight late interaction,” in Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, eds.), (Seattle, United States), pp. 3715–3734, Association for Computational Linguistics, July 2022.

KDD Cup CRAG competition: Systems, Findings and Learnings

Xiao Yang^{††1}, Yifan Ethan Xu^{*1}, Kai Sun^{*1}, Jiaqi Wang^{*1}, Lingkun Kong¹,
Wen-tau Yih², Xin Luna Dong¹
¹Meta Reality Labs, ²FAIR, Meta
{xiaoyangfb, ethanxu, sunkaicn, jqwang, klk,
scottyih, lunadong}@meta.com

Abstract

The KDD Cup 2024 CRAG Challenge aims to provide a foundation for evaluating Retrieval Augmented Generation (RAG) systems. RAG systems take natural language questions as input, decide whether and how to use the facilitating information such as web pages and mock Knowledge graphs, and return natural language answers. A good RAG system provides correct and useful answers to questions without bringing in hallucinated information. In this report, we describe the motivation for organizing this challenge, review the design for the benchmark and the competition, discuss observations from the submissions, and reflect the learnings from hosting the challenge.

A Introduction to RAG and the CRAG benchmark

The rise of Large Language Models (LLMs) has revolutionized the field of natural language processing [22, 24, 53, 58], but these models still struggle with providing accurate and reliable answers to complex questions. One major challenge is the tendency for LLMs to “hallucinate” or generate responses that are not grounded in factual information [18, 34, 40, 42]. To address this issue, researchers have turned to Retrieval-Augmented Generation (RAG) [6, 10, 13, 20], a technique that involves searching external sources to gather relevant information and then using that information to generate more accurate answers (Figure 50).

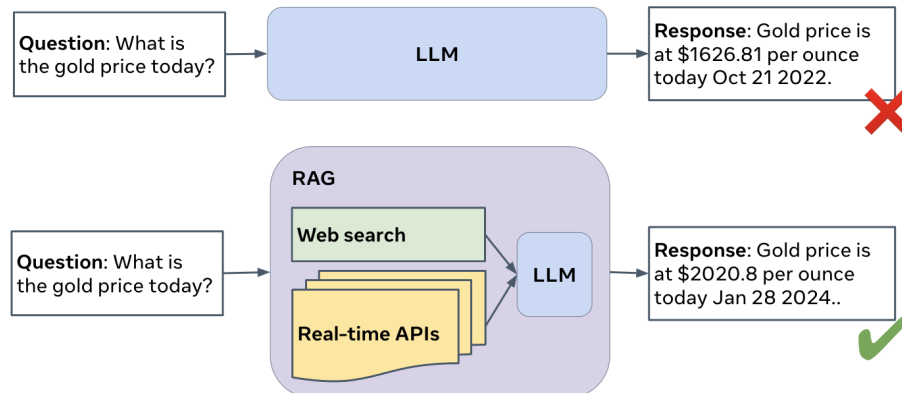


Figure 50: Given a question, a RAG system searches external sources to retrieve relevant information and then provides grounded answers.

^{††}The first four authors contributed equally.

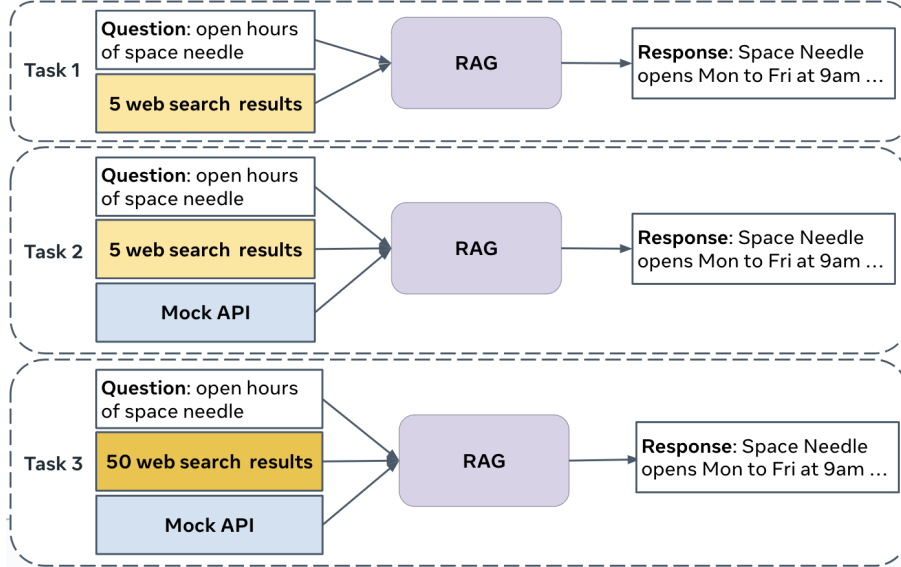


Figure 51: KDD Cup 2024 Meta CRAG challenge tasks.

However, existing datasets for evaluating RAG systems are limited in their scope and diversity, failing to capture the complexity and nuance of real-world question-answering tasks. To address this limitation, we developed the Comprehensive RAG (CRAG) Benchmark, a comprehensive evaluation framework designed to assess the capabilities of RAG systems. The benchmark comprises over 4.4K question-answer pairs, accompanied by mock APIs that mimic the process of searching the web and accessing knowledge graphs. CRAG is across five domains (finance, sports, music, movie, and encyclopedia open domain), and a diverse range of question types, including simple-fact questions, questions that require aggregation and reasoning. CRAG further encompasses varied entity popularity from popular to long-tail and temporal spans ranging from seconds to years. As such, it covers a broad spectrum of real user queries, providing a realistic and challenging testbed for RAG systems [52]. By incorporating both web and knowledge graph search, CRAG simulates realistic information retrieval scenarios, reflecting the common practice of users seeking answers from multiple sources. Our evaluations using this benchmark revealed substantial performance gaps in even the most advanced LLMs and RAG systems, highlighting the need for continued research and development in this area.

The CRAG benchmark served as the foundation for the KDD Cup 2024 competition, which drew over 2.5K participants and 5.6K submissions from around the world. The winning solutions demonstrated significant improvements over baseline methods. In this paper, we present key insights and takeaways from the competition, including our learnings on the RAG landscape (Section C), learnings from the CRAG winning solutions (Section D), and learnings from hosting the CRAG challenge (Section E), suggesting directions for future RAG research.

B Challenge Overview

B.1 Challenge Tasks

A RAG QA system takes a question Q as input and outputs an answer A ; the answer is generated by LLMs according to information retrieved from external sources or directly from the knowledge internalized in the model. The answer should accurately address the question while avoiding any hallucinations.

We designed three tasks, which share the same set of (question, answer) pairs but differ in the external data accessible for retrieval to augment QA, as shown in Figure 51. Here, we provide the content that can be leveraged in QA to ensure fair comparisons.

Task 1: Retrieval Summarization. In Task 1, we provide up to five web pages for each question. These web pages are likely, but not guaranteed, to be relevant to the question. This task aims to test the answer generation capability of a RAG system.

Task 2: KG and Web Retrieval Augmentation. In Task 2, we in addition provide mock APIs to access information from underlying mock KGs. The mock KGs store structured data relevant to the questions; answers to the questions may or may not exist in the mock KGs. The mock APIs take input parameters, oftentimes parsed from the question, and provide structured data from the mocked KGs to support answer generation. This task tests how well a RAG system 1) queries structured data sources and 2) synthesizes information from different sources.

Task 3: End-to-end RAG. Similar to Task 2, Task 3 also provides both web search results and mock APIs as candidates for retrieval but provides 50 web pages, instead of 5, as candidates. The larger set of web pages increases the likelihood of providing relevant information to answer the question, but they also tend to contain more noise. As such, Task 3 in addition tests how a RAG system ranks a larger number of retrieval results.

The three tasks, each adding upon the previous one, allow testing different capabilities of the end-to-end RAG systems.

B.2 Data and System Requirement

The challenge used the CRAG benchmark as described in Section A. We split the data randomly into validation, public test, and private test at 30%, 30%, and 40%, and released the validation and public test sets for the challenge. The final winners were determined by evaluating against the held-out private test set.

In order to make the systems more comparable and to encourage open source development, we require all submitted systems to be based on Llama 2 [44] or Llama 3 [3] models in this challenge. Participants can also fine-tune their models using publicly available data as long as the data were not generated by proprietary models.

The challenge was hosted on the AICrowd competition platform, and results were posted on a leaderboard. All submissions were run on AWS G4dn.12xlarge instances equipped with 4 NVIDIA T4 GPUs with 16GB GPU memory. Since neither the Llama 2 70B or Llama 3 70B models in full precision can be directly run on these T4 GPUs, participants needed to use quantization or other techniques to make their system runnable on the inference platform. Also, network connection was disabled during the challenge to ensure all participants can access an equal set of retrieved contents, and to prevent leak of the private test set data. Moreover, each example had a time-out limit of 30 seconds and was truncated to 75 BPE tokens [38] in the auto-evaluation. In human-evaluation, graders examined the first 75 bpe tokens to find valid answers, but reviewed the whole response to judge for hallucination. See section B.3 for more details about the evaluation.

B.3 Evaluation

Metrics

We use a scoring method to assess the performance of RAG systems. For each question in the evaluation set, we first label the answer with **perfect**, **acceptable**, **missing**, or **hallucination**, according to the following criteria.

Perfect. The response correctly answers the user’s question and contains no hallucinated content.

Acceptable. The response provides a useful answer to the user’s question but may contain minor errors that do not harm the usefulness of the answer.

Missing. The response is “I don’t know”, “I’m sorry I can’t find ...”, a system error such as an empty response, or a request from the system to clarify the original question.

Hallucination. The response provides wrong or irrelevant information to answer the user’s question.

We then use a scoring method Score_h with score 1, 0.5, 0, and -1 for each *perfect*, *acceptable*, *missing*, and *hallucinated* answer, respectively, where we penalize hallucinated answers because users would prefer the model to admit “I don’t know” than providing an hallucinated answer when it does not know how to answer the question. We then define **Truthfulness** as the average score from all examples in the evaluation set for a given RAG system. Concretely,

$$\text{Truthfulness}_h = \text{Perfect rate} + 0.5 * \text{Acceptable rate} - \text{Hallucination rate}.$$

Evaluation Method

We employ a hybrid evaluation system that includes both human evaluation (**human-eval**) and model-based automatic evaluation (**auto-eval**). In the former, we use manual grading to judge perfect, acceptable, missing, and hallucinated for each answer. In the latter, we merge perfect and acceptable, call it **accurate**, and use a three-way scoring Score_a with 1, -1 , 0 for accurate, hallucinated, and missing answers. And in this case,

$$\text{Truthfulness}_a = \text{Accuracy} - \text{Hallucination rate}.$$

We design a two-step method for automatic evaluation: if the answer matches the ground truth exactly, it is considered accurate; otherwise, we use LLMs to determine whether the response is accurate, hallucinated, or missing. To avoid the self-preference problem [31], we experimented with two families of LLM evaluators: ChatGPT (**gpt-3.5-turbo-0125**) [29] and Llama 3 (**llama-3-70B-instruct**) [3] and reported the average accurate, hallucinated, missing rates, and scores from the two models for benchmarking the straightforward solutions [52]. This two-step method yields an accuracy of 94.5% for ChatGPT and 99% for Llama 3 compared to human-eval. We adopted ChatGPT as the auto-evaluator in the challenge due to its low cost.

B.4 Challenge Schedule and Outcomes

The challenge was announced on March 15, 2024. Submissions were accepted starting on April 1, 2024, simultaneously with the release of the starter kit. Baseline models were released a week after. The challenge had two phases: in Phase 1, each team can make up to six submissions per week for all three tasks together during a two-month period, and check their results directly on the leaderboard based on auto-eval; in Phase 2, each team can submit up to six times for the three tasks together in total. The submitted solutions in Phase 2 were first evaluated by auto-eval. Then the top-15 teams from each task were sent for human-eval to determine the final winners. Phase 2 ended on June 22, 2024, and the final results were announced on July 28, 2024.

The challenge awarded the top-3 teams that obtained the highest Truthfulness scores in each task. It also provided seven additional prizes for teams that achieved the highest scores for each of the seven complex question types. In the end, twelve teams won the prizes (See Table 32 for the list of the winning teams.), among which six winning teams were invited to present in the KDD 2024 CRAG workshop¹. All participating teams were encouraged to submit a technical report for their solutions.

¹Please see more information about the workshop at: <https://kddcup24.github.io/pages/benchmark.html>

The challenge attracted more than 2,500 participants, 384 teams and more than 5,600 submissions. After the challenge was complete, we held a workshop during the ACM KDD 2024 conference. The workshop featured three keynote speeches, six winning-team presentations, and attracted more than 130 onsite audiences. It also published 11 technical reports from the participating teams afterwards.

C Learnings on RAG landscape

	Solution	Accuracy	Hallucination	Missing	Truthfulness	Latency (ms)
LLM-Only	Llama 3	32.3%	28.9%	38.8%	3.4%	
	GPT-4	33.5%	13.5%	53.0%	20.0%	
Straightforward RAG	Llama 3	40.6%	31.6%	27.8%	9.1%	
	GPT-4	43.6%	30.1%	26.3%	13.5%	
CRAG KDD Cup Winners *	Top-3: vsluy-team	43%	24.9%	30.1%	18.1%	
	Top-2: APEX	48.6%	13.7%	36.3%	34.9%	
	Top-1: db3	53.3%	17.1%	28.0%	36.2%	
Industry SOTA **	Perplexity.ai	60.2%	25.3%	10.1%	34.9%	4,634
	Meta SG	57.4%	16.0%	21.8%	41.4%	3,431
	ChatGPT Plus	66.5%	25.0%	1.9%	41.5%	6,195
	Gemini Advanced	65.9%	16.6%	12.5%	49.3%	5,246
	Copilot Pro	68.5%	17.9%	7.8%	50.6%	11,596

Table 31: Truthfulness scores on CRAG. (*: the score is obtained by human evaluation and the others are obtained by automatic evaluation; **: in addition, different accesses to retrieval contents. [52].)

Table 31 summarizes the performance in truthfulness score of baselines (the LLM only and straightforward RAG solutions), the KDD Cup winner solution, and the best-performing industry state-of-the-art (SOTA) system on CRAG. We have the following observations and details can be found in [52].

1. The LLM-only solution performs poorly on CRAG, with truthfulness ranging from 3% to 20%.
2. Although RAG can help answer more questions, adding RAG in a straightforward manner does not necessarily reliably outperform the LLM-only solution (truthfulness ranging from 9% to 13%). This is because careless summarization in baseline RAG solutions can introduce more hallucinations generated from irrelevant retrieval results.
3. The CRAG KDD Cups winning solutions improve the truthfulness significantly to 36%, but still falls short of perfection in terms of answer accuracy and latency. We will describe the solutions in more detail in Section D.
4. Industry SOTA systems, which likely leverages better models and retrieval systems (e.g., web and KG search engines), achieved 51% truthfulness. The systems that achieved the highest truthfulness and that had the lowest latency were different, reflecting the quality-latency tradeoff. There is still much room for improvement on quality and latency.

D Learnings from the winning solutions

More than 2500 participants took part in the CRAG competition. Four teams won top-three positions across three tasks, and seven additional teams scored high points for individual question types (see Table 32).

Table 32: CRAG competition winning teams and team placements. For each task we awarded top-3 teams for overall scores, and top-1 team for each question type.

Team name	Task 1 placement	Task 2 placement	Task 3 placement
db3 [49]	1st place	1st place	1st place
APEX [30]		2nd place	2nd place
md_dh [11]	2nd place	3rd place	Set Aggregation Post-processing
vslyu-team [51]			3rd place
ElectricSheep [55]	3rd place	Simple with condition Set Aggregation Multi-hop Post-processing	
dRAGonRAnGers [32]	Comparison Post-processing	Comparison	Comparison
ETSLab	False premise		Multi-hop
dummy model [16]	Simple with condition Set Aggregation		
bumblebee7 [56]	Multi-hop		
Future [9]		False premise	
StarTeam [48]			Simple with condition
Riviera4 [43]			False premise

Most winning solutions adapt a general RAG system design that comprises two main stages (see Figure 52): (1) Knowledge Retrieval components that retrieve and process relevant information from web or knowledge graphs, and (2) an Augmented Generation component that leverages an LLM to incorporate retrieved information and provide an answer to the question.

No team trained the overall pipeline in an end-to-end fashion. Instead, each component is optimized separately, where sub-systems are tailored to the specific tasks or question types. The teams leverage both off-the-shelf solutions (e.g., BeautifulSoup for HTML parsing, BGE encoders for ranking) and customized solutions (e.g., fine-tuned LLM to reduce hallucination). In the following we deep dive to learnings from each component.

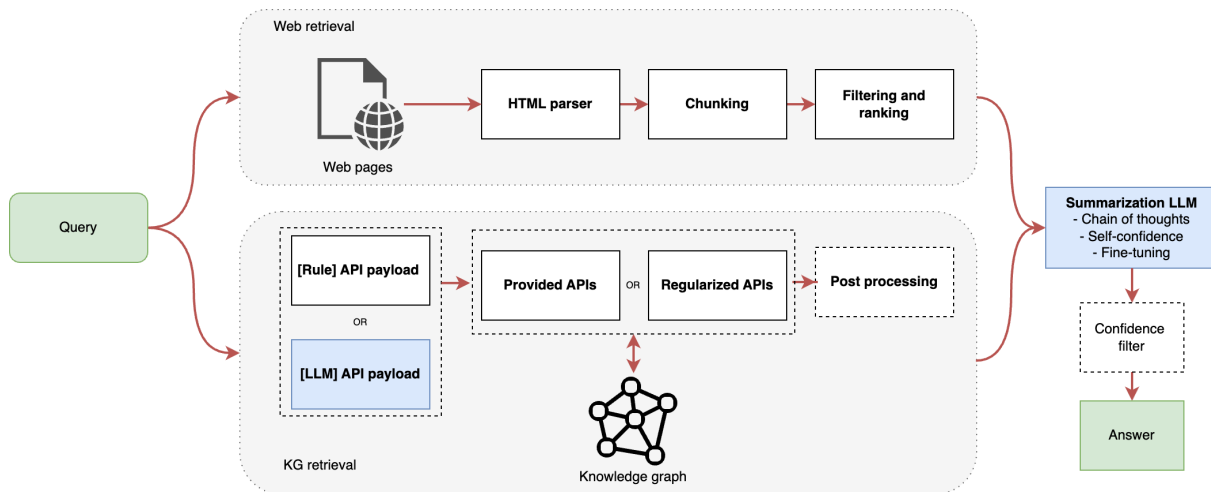


Figure 52: Conceptual overall architecture from winning solutions.

D.1 Knowledge Retrieval from Web

Given a query, this component retrieves supporting material from the web repository (task 1, 2 and 3). An ideal retrieval system finds relevant knowledge efficiently (high recall) without including too many noises (high precision). There are several key challenges: 1) raw HTML contains inconsistent structures, making it challenging to extract clean text; 2) long web pages exceed LLM’s input length constraints; 3) webpages often contain contents that are irrelevant to the question that dilute signals and potentially lead to hallucinations from summarizations.

To address these challenges, most winning solutions follow a workflow consisting of four steps to obtain clean relevant information from webpages: HTML parsing, content chunking, relevance scoring, and re-ranking. The teams heavily utilize off-the-shelf libraries as building blocks as listed in Table 33 below.

Table 33: Off-the-shelf libraries the winning solutions leveraged for web retrieval.

Component	Libraries
HTML Parsing	BeautifulSoup [35], Trafilatura [5]
Chunking	CharacterTextSplitter [26] , ParentDocumentRetriever [26]
Relevance scoring	bge-base [50], ms-marco-miniLM [27] , sentence-t5 [28], BM25 [36]
Re-ranking	bge-reranker [8]

The teams leveraged a variety of chunking strategies to partition web content into smaller segments. Beyond the basic token-level chunking method, three solutions stand out in ensuring related contexts are preserved in the same chunk, allowing the summarization model to access coherent information:

1. Db3’s [49] solution maintains a parent-child chunk relationship using the ParentDocumentRetriever library. Once they identify smaller children chunks that are relevant to the question, they include their associated parent chunks in the pipeline.
2. Md_dh’s [11] solution traverses the tree structure of HTML to identify chunks at node level, minimizing segmentation of texts under a common tag such as header, table, sections.
3. ElectricSheep’s [55] solution first identifies questions in web pages by matching sentences that start with “5W1H” key words (i.e. "Who", "What", "Where", "When", "Why" and "How") and end with a question mark, and then ensures subsequent texts are grouped together with the questions.

For relevance scoring and re-ranking, the winning solutions mainly rely on off-the-shelf sparse (e.g. BM25) and dense (e.g. bge-base) retrieval models and re-ranker models (e.g. bge-reranker) without customized fine-tuning. Ablation study (md_dh [11]) shows that dense retrieval via cross-encoders has a slight edge over sparse retrieval in terms of accuracy.

D.2 Knowledge Retrieval from KG

CRAG provides various domain specific APIs to access underlying KG data (task 2,3). The main tasks for efficient retrieval from KG are three folds: 1) identifying the appropriate API(s), 2) generating correct API parameters, and 3) processing API results to find relevant information.

The winning solutions for KG retrieval fall into three broad categories.

1. Build each retrieval step separately. Using APEX’s [30] solution as an example, this approach invokes the following steps. It first leverages an LLM to identify named entities, followed by entity linking via string matches or BM25 fuzzy matches. Then it identifies an appropriate API and its parameters via domain-specific rules and calls the API. Afterwards, it filters the API results based on the date of the query. Finally it converts the filtered results to markdown format and sent to the final generation model.

This approach provides explicit control and insights of each step at the cost of integration complexity.

2. Generate e2e API call directly. A few teams (db3 [49], md_dh [11], ElectricSheep [55] etc.) prompt an LLM to directly generate API payload, where fine-tuned LLMs with API specific data further improve the calling accuracy.
3. Regularized API. Another innovative solution came from team db3 [49]. Instead of building a retrieval system around the original APIs provided by CRAG, the authors develop a new set of “regularized” APIs that enhances the expressiveness of the original APIs. More specifically, the solution built API wrappers to encapsulate filtering conditions (e.g. “year” == “2012”) and aggregation logics (e.g. maximum, average). The following is an example comparing the original API and the regularized API.

$$get_movie(movie_name) \rightarrow get_movie(movie_name, condition)[key_name] \quad (13)$$

where “condition” is from a set of predefined rules, and “key_name” are indices to search in the results. The expressive regularized APIs allow the team to obtain concise information via a single API call generated by LLM.

D.3 LLM Augmented Generation

LLM Augmented Generation incorporates upstream retrieved knowledge to generate a final answer. All winning solutions put significant efforts in reducing hallucination in this step, partially because hallucination is penalized more severely than a missing (e.g. "I don't know") answer in the CRAG scoring criteria.

The main strategies that the winning solutions adapt to reduce hallucinations are a combination of chain-of-thought prompting, explicit confidence estimation, supervised fine-tuning, and manual rules.

1. Chain-of-thought. Teams (e.g., ElectricSheep [55], APEX [30]) show that prompting the LLM to articulate intermediate thinking steps improve answering accuracy for complex questions and significantly reduce hallucination. For instance, ElectricSheep [55] prompts the model to first identify if a question has false premise, then whether the question can be answered by each retrieved evidence, before generating the final answer.
2. Explicit confidence inference. Another approach is to estimate LLM answer's confidence, then only retain the answer at high confidence, and answer "I don't know" otherwise. Winning solutions either prompt the LLM directly to self produce a confidence level, or sample several answering paths and approximate the confidence as the answer consistency. Both methods are effective in their respective pipeline, but the self-consistency approach demands more computational resources.
3. Supervised fine-tuning. Several teams (e.g., db3 [49], md_dh [11], dRAGonRAnGers [32]) fine-tune an Llama-3-8b model specifically targeting reducing hallucinations. The teams generated training labels by first running the pipeline with a non-fine-tuned LLM generation model, then prompts a separate LLM to automatically identify questions that are not answerable by retrieved knowledge passages and labeled "I don't know" for fine-tuning. The resulting fine-tuned model answers "I don't know" more frequently on questions that it tends to hallucinate originally.
4. Manual rules. Since dynamic questions are much more difficult to answer correctly than static questions. Some teams opt to avoid answering dynamic questions all together to prevent getting penalized with hallucinated answers.

D.4 Observations by Question Categories

CRAG categorizes questions along four dimensions: Topic domain (Finance, Sports, Music, Movie, Open), Dynamism (Real-time, Fast-changing, Slow-changing, Static), Popularity (Web, Head, Torso, Tail) and Question Type (Simple, Simple with condition, Set, Comparison, Aggregation, Multi-hop, Post-processing, False premise). The strategies and performance of winning solutions vary significantly across different dimensions, as shown in Figure 53 from task 3 winning teams.

Domain dimension Since each domain has its own set of APIs, most winning solutions develop router to detect the topic domain of each question then use the results to select appropriate API calls and construct domain specific prompts accordingly.

Among the domains, Finance questions prove to be the most challenging. This is mainly because many financial questions are real-time or fast-changing (e.g., "can you tell me the opening stock price of landp on the last Friday (question asked on 02/28/2024, 07:58:48 PT)?"), thus requiring finding results precisely corresponding to the reference time. To tackle date-time parsing, APEX [30] leveraged regex rules and Python libraries (pytz and datetime) to parse absolute datetime objects. One other challenge is that Finance questions often require numeric reasoning. ElectricSheep [55] delegates numeric calculations to an external Python interpreter, which reduces hallucination from LLM.

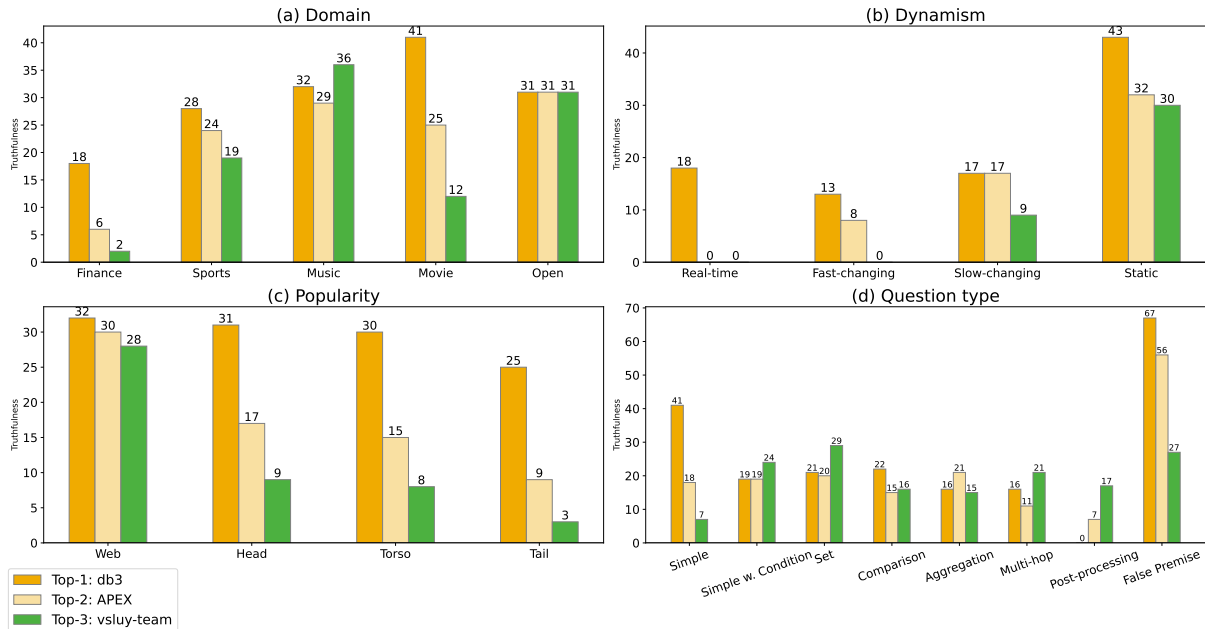


Figure 53: Human-evaluated Truthfulness scores from Task 3 winners on the private test set across the four question categories. The truthfulness scores reflect different strategies taken by different winning teams. First-place winner for all three tasks Db3 [49], as an example, put in significant efforts for improving the results on using the APIs, which gained them large margins in handling those KG-supported questions.

Dynamism dimension Overall, winning solutions perform better in static and slow-changing questions than fast-changing and real-time questions, mainly because of difficulties in temporal reasoning in both the retrieval step and the summarization step. Some teams even implemented solutions to blankly answer “I don’t know” for all dynamic questions to reduce hallucination penalties. One promising approach was team db3’s [49] “regularized” APIs (see Section D.2), which improved the APIs’ expressiveness in handling temporal information.

Dynamic questions are an area where RAG should excel over vanilla LLM. The fact that teams struggle with dynamic questions suggests further opportunities in this area.

Popularity dimension In this dimension, the performance is largely correlated with entity popularity as expected, due to sparsity of information for torso to tail questions.

Question Type dimension In this dimension, comparison questions (e.g., “who started performing earlier, Adele or Ed Sheeran?”), post-processing questions (e.g., “How many days did Thurgood Marshall serve as a Supreme Court justice?”), and multi-hop questions (e.g., “what school does Lebron James’s youngest son attend?”) are the most difficult types, mainly because they require retrieving information of multiple entities and nontrivial reasoning over the retrieved results.

Team dRAGonRAnGers’s [32] solution won first place for comparison questions in all three tasks. The authors hypothesized that their explicit chain-of-thought prompt contributed to the success by generating rationals for each entities involved in the comparison then synthesizing a final answer. Similarly, carefully engineered chain-of-thought prompts also improved reasoning for multi-hop and post-processing questions, as Team ElectricSheep [55] and bumblebee7 [56] demonstrated.

E Learnings from hosting the challenge

In this section, we reflect the experience from creating the benchmark, designing the challenge, and finally hosting the challenge and workshop, and summarize our learnings from this one year journey.

E.1 How can we design a fair, informative and feasible challenge?

We strive to create an informative and engaging challenge, but also need to take into consideration the fairness, accessibility, as well as time and budget constraints. We made several design decisions for the challenge, which turned out to be critical for its success. We narrate these design choices next.

Creating equally accessible retrieval contents. Providing equally accessible retrieval contents eliminates the unfair comparison induced by having access to different resources among the participants. A RAG system theoretically can retrieve from the universe of all public and proprietary information sources. Although it is interesting to test RAG systems in an end-to-end manner by allowing participants to use all accessible APIs, it inevitably creates an advantage for large companies or entities that have more resources. We therefore made a decision to provide equally accessible retrieval contents in the benchmark for the challenge, so that all systems can be evaluated against the same set of facilitating information, eliminating the potential unfairness introduced by having access to different amount of retrieval sources. This design choice, although not completely testing the retrieval of the RAG systems, still allows users to evaluate it by combining all the retrieved webpages as the retrieval pool.

Another advantage of using fixed retrieval contents is to be able to compare over time. Search engines evolve over time. People who use the benchmark questions later on can get better results just because the search engine improves its results. Meanwhile, CRAG retrieval content is available for experiments. Researchers who want to conduct a fair comparison do not need to reproduce the previous results, when the results were reported on CRAG retrieval content. Otherwise, they would need to re-implement existing SOTA systems to make sure the same version of search engines were used. This can be very consuming and even infeasible (if the system involves proprietary implementation or data). We believe providing equally accessible retrieval content can substantially facilitate the research for RAG, and are striving to extend this practice to future studies. See Section G for more discussions.

Setting model constraints. We put the constraints on which model can be used to make the challenge focus on RAG, instead of the base models. The participants can use vanilla or fine-tuned Llama 2 or Llama 3 models as long as the data were publicly available and were not generated from any proprietary model. This decision also ensures the results from the built systems are more comparable.

E.2 How can we ensure the cost is affordable?

There are a lot of cost involved in running a large-scale challenge. To ensure its sustainability, we carefully planned to control two major costs: computation and evaluation.

Setting submission limit and hardware constraints. The computation cost mainly comes from using GPUs to run the inference. We set limits on the number of submissions from each team during the challenge to make sure the computation cost can be controlled. The challenge allows up to six submissions for each team each week in phase 1, and up to six submissions for each team in total in phase 2.

We also selected a more affordable GPU option: the AWS G4dn.12xlarge instance equipped with 4 NVIDIA T4 GPUs with 16GB GPU memory as the supported hardware for the inference. These GPUs

have the limitation that they cannot run the Llama 2 70B or Llama 3 70B full precision models directly. However, the price of those more powerful GPUs were several times higher and would not bring much additional value. Therefore, we decided to choose the more economical option.

These submission limits, along with the selected hardware configurations, ensured that we can provide enough room for the participants to develop and test their solutions while keeping the computation cost under the budget.

Using auto-eval in Phase-1 and in Phase-2 initial selection. Although human-eval is the most comprehensive way for evaluating responses from LLMs, it would be too slow and expensive to support challenges that receive thousands of submissions. Therefore we made an early decision to develop auto-eval solutions for the challenge. Phase 1 of the challenge was purely relying on auto-eval to support the leaderboard. For phase 2, we first used auto-eval to select the top-15 teams for each task, and then employed human manual evaluation to determine the final winners. This design substantially reduced the time and expense (more than 100 times) needed for the evaluation.

E.3 How can we ensure the evaluation is reliable?

In order to guarantee the evaluation quality, we still need to ensure the auto-eval mechanism has high accuracy, which we discuss next.

Ensuring auto-eval quality. We solve the auto-evaluation problem with an LLM task, where the prompt asks the LLM to determine whether the generated answer is accurate, hallucinated, or missing based on the ground truth information. This is a simple version of the NLP task —consistency check.

We conducted extensive experiments to make sure the auto-eval solution can have high accuracy [52]. We have two observations when developing the auto-eval solution. First, it is critical to provide high quality ground truth answers for the auto-evaluator to work effectively. During the experiment, we observed that some examples may not have a unique correct answer (e.g., a question about height can be answered in either feet or meters.). Hence, we also provided alternative ground truth answers for the benchmark questions whenever needed. It turned out that these alternative answers were very helpful for reducing evaluation errors. Second, we don't need the most powerful models to serve as the auto-evaluator. Our experiment shows that the accuracy of using the Llama 3 (`llama-3-70B-instruct`) model and the ChatGPT (`gpt-3.5-turbo`) are 99% and 94.5% respectively (for predicting accurate, missing and hallucinated responses) [52]. See Table 34 for more detailed results.

In the final evaluation, among the nine top-3 winning teams selected by manual evaluation, six of them were also selected by our auto-evaluation. Generally speaking, using LLM-as-a-Judge is a common practice in evaluating language models' performance. Procedures for measuring and resolving the self-enhancement bias, position bias [57] and preventing prompt attacks play vital roles in ensuring the reliability of the evaluation results.

Preventing evaluation attacks. We found three decisions very helpful for protecting the evaluation from hacks. The first one was to keep a held-out test set for the final competition (phase 2). We initially ran the leaderboard in phase 1 using the released public test set, but soon found the leaderboard was filled with 100% accuracy. This was because some teams tried to test the evaluation system and created a map with (query, answer) pairs based on the released data, from which they looked up answers for the test queries during the submission. This issue was remedied by replacing the leaderboard test set with a subset of the held-out dataset. The second decision was to hold out the exact prompt used in the auto-eval to avoid prompt attack. We released an example evaluation prompt to illustrate how the auto-eval works in the starter kit, and later on received questions from the challenge forum calling out

Model	F1 Score			Accuracy
	Correct	Missing	Hallucinated	
ChatGPT-3.5 Turbo	92.0%	100.0%	92.0%	94.5%
Llama 3.0 8B Instruct	94.6%	100.0%	90.7%	95.3%
Llama 3.0 70B Instruct	98.9%	100.0%	97.9%	99.0%
Llama 3.1 70B Instruct	98.2%	100.0%	96.8%	98.4%

Table 34: Auto-eval performance for using different models as the evaluator. The first three columns show the F1 scores on the accurate, missing and hallucinated responses. The last column shows the overall accuracy on all examples in the CRAG public test set. Llama 3.0 70B and Llama 3.1 70B models achieve accuracy around 99%. Llama 3.0 8B, although being much smaller, attains accuracy 95.3%, on par with ChatGPT-3.5 Turbo. All models have 100% on the missing answers because we require all missing answers to be answered as “I don’t know”, and we use exact match to judge for missing responses during the evaluation.

ways to hack the prompt for obtaining high scores. During the final competition, we also noticed some submissions that tried to fool the auto-evaluator achieved very high scores. Although the final winners were selected by the manual evaluation, recall that a team needed to be among the top 15 in the auto-eval to be eligible for human-eval. Keeping the evaluation prompt private significantly reduced the risk of prompt attack and also reduced the overhead needed for manual checking after the evaluation. The third one was to disable network connection during the inference. The choice of providing pre-fetched retrieval contents (discussed in Section E.1) allowed us to avoid using live web connection during the challenge. This also helped reduce the risk of leaking the private test data during the evaluation stage.

F Related work

F.1 RAG Benchmarks

We compare and highlight the strengths and limitations of existing retrieval-augmented generation (RAG) benchmarks across several critical dimensions, as demonstrated in table 35 in CRAG [52]. QALD-10 [46] focuses primarily on knowledge graph search over Wikidata but does not incorporate web retrieval, limiting its capacity to test models in unstructured environments. It lacks mock APIs or dynamic question capabilities, reducing its ability to simulate real-world, evolving knowledge scenarios. MS MARCO [4], while widely used for open-domain question answering, primarily emphasizes passage retrieval and lacks integration with knowledge graphs. It also fails to test for long-tail facts, as it predominantly handles popular, factoid-based information sourced from common queries, limiting its coverage of less frequent, tail facts. Natural Questions [19] focuses on more complex and long-form queries but remains heavily constrained to Wikipedia-based knowledge and lacks dynamic or API-driven retrieval. RGB [7], on the other hand, introduces a focus on long-tail facts but does not fully test retrieval beyond specific domains, missing integration with KG searches or dynamic question handling. FreshLLMs [47] excels in testing models’ ability to retrieve and update real-time knowledge but remains narrow, lacking domain diversity and focus on structured retrieval such as knowledge graphs.

Despite having smaller question size than MS MARCO and NQ, CRAG [52] stands out by combining web retrieval, knowledge graph search, and mock APIs, simulating diverse retrieval environments. It goes beyond Wikipedia, tackling dynamic questions and ensuring comprehensive coverage of both torso and tail facts across multiple domains, making it a more robust and versatile benchmark for evaluating

Table 35: Comparing CRAG to existing benchmarks for factual question answering.

Benchmark	Web retrieval	KG search	Mock API	Dynamic question	Torso and tail facts	Beyond Wikipedia	Question size
QALD-10 [46]	✗	✓	✗	✗	✗	✗	0.8K
MS MARCO [4]	✓	✗	✗	not explicitly	not explicitly	✓	100K
NQ [19]	✓	✗	✗	not explicitly	not explicitly	✗	323K
RGB [6]	✓	✗	✗	✗	✗	✓	1K
FreshLLM [47]	✗	✗	✗	✓	✗	✓	0.6K
CRAG [52]	✓	✓	✓	✓	✓	✓	4.4K

next-generation RAG systems.

F.2 RAG Systems

RAG systems have evolved significantly over time, focusing on improving large language models (LLMs) by integrating retrieval mechanisms for enhanced knowledge access. Talmor et al. [41] pioneered the use of the web as a knowledge base to answer complex questions, emphasizing retrieval for reasoning tasks. Lewis et al. [21] formally introduced RAG, combining dense retrieval with generative models to solve knowledge-intensive NLP tasks, setting a strong foundation for subsequent models. REALM [14] improved this by incorporating retrieval into pre-training, making retrieval part of both the learning and fine-tuning process, which was further extended by Fusion-in-Decoder (FiD) [17], fusing multiple retrieved documents into the generation process for improved question answering. Mallen et al. [25] explored the effectiveness of parametric and non-parametric memory mechanisms to determine when models should rely on internal versus external knowledge. Similarly, Sun et al. [39] investigated whether LLMs could replace traditional knowledge graphs, assessing how well LLMs store and retrieve structured knowledge. QA-GNN [54] integrated retrieval-augmented methods with reasoning from knowledge graphs, combining structured and unstructured knowledge for better question answering. Meanwhile, Mallen et al. [25] focused on trust in LLMs, analyzing the limitations of parametric knowledge and advocating for non-parametric memory integration. Recently, FreshLLMs [47] refreshed LLM knowledge using search engine augmentation to maintain up-to-date responses. These works collectively contribute to a growing body of research that refines LLM performance by bridging the gap between parametric knowledge and external, retrievable information.

F.3 RAG System Evaluation

In recent developments for evaluating RAG systems, multiple frameworks have proposed solutions aimed at addressing key challenges like retrieval relevance, truthfulness, and the efficiency of the evaluation process. ARES [37] offers an automated evaluation approach by generating synthetic data and fine-tuning lightweight language models, allowing for scalable assessments of retrieval relevance, answer faithfulness and answer relevance. However, ARES may face issues with domain adaptation, as its synthetic data may not always represent the nuances of diverse real-world datasets. RAGAS [12] introduces a reference-free evaluation method, providing metrics for context relevance without relying on ground truth annotations, reducing human labeling costs. Nevertheless, its heuristic-based measures may struggle with capturing deeper semantic nuances and can exhibit biases in complex scenarios. CoFE-RAG [23] performs an exhaustive evaluation across the entire RAG pipeline, employing multi-granularity keyword-based assessment of the retrieved context. RAG-QA Arena [15] proposes a pairwise preference evaluation framework with truthfulness as the primary criterion, leveraging ground truth to gauge the evaluation

system quality.

F.4 RAG Competitions

More recently, the TREC 2024 RAG Track is proposed, featuring Ragnarök [33], an open-source, end-to-end evaluation framework with MS MARCO V2.1 collection and industrial baselines like OpenAI’s GPT-4o and Cohere’s Command R+. The framework provides a web-based interface for benchmarking pairwise RAG systems through a RAG battle arena.

Various domain-specific competitions, such as the Zindi RAG challenge [2] for public services, the FinanceRAG competition [1], and the Trustbit Enterprise RAG Challenge [45], push the boundaries of RAG applications. These challenges emphasize tailored RAG systems that retrieve domain-specific information and generate context-aware responses, demonstrating the potential of RAG to enhance service delivery in public administration, financial analysis, and business document comprehension.

G Discussion and future work

The KDD Cup CRAG 2024 competition established a benchmark for evaluating Retrieval-Augmented Generation (RAG) systems with three challenging tasks: retrieval summarization, knowledge graph and web retrieval, and end-to-end RAG. Questions covered various domains, dynamism levels, types, and entity popularity. Key learnings from hosting the challenge (Section E) emphasized the need for standardized retrieval content and retrieval access, controlled model constraints, maintaining a private test set, and mixed automated and human evaluation strategies to ensure fairness and manage cost.

CRAG highlighted persistent challenges in RAG systems. First, reducing retrieval noises from the web and from KG is critical as irrelevant or contradicting information degrades answer quality. Second, handling real-time or fast-changing questions remains difficult due to the need for nuanced temporal reasoning. Third, questions that require aggregation, multi-hop reasoning or post-processing are challenging as they require complex logic to integrate knowledge across multiple sources. Lastly, reducing hallucinations in summarization remains essential. To address the above challenges, winning teams leveraged a variety of solutions (Section D). For example, developing multi-step pipelines to parse, chunk, and rank web contents reliably, creating regularized APIs to enhance KG retrieval expressiveness, incorporating external libraries to improve temporal reasoning, leveraging chain-of-thought techniques to break down complex queries, and fine-tuning LLM and implementing confidence estimation component to reduce hallucinations. The practical applications of these findings have significant implications for improvements in industry-grade RAG systems.

The CRAG competition highlights several promising directions for future RAG research: Enhancing web retrieval quality via better handling of structured and semi-structured data, improving temporal reasoning for real-time queries, and refining methods for multi-hop and aggregation tasks can improve response reliability. Additionally, targeted fine-tuning may be essential to effectively reduce hallucinations. Looking ahead, future CRAG-style competitions could expand the question sets to include multimodal, multi-turn questions, integrating text with images or structured data for a more realistic RAG environment. As researchers pursue these directions, they will also need to address key scalability challenges in building RAG systems that can handle real-time and multimodal data, including ensuring data consistency across diverse sources and achieving low-latency processing of real-time multimodal data. By tackling these challenges, researchers can unlock the full potential of RAG systems and improve response reliability.

References

- [1] Financerag challenge: Retrieval-augmented generation (rag) for financial documents. <https://finance-rag.com/>. Accessed: 2024-11-10.
- [2] Retrieval-augmented generation (rag) for public services and administration tasks. <https://zindi.africa/competitions/retrieval-augmented-generation-rag-for-public-services-and-administration-tasks>. Accessed: 2024-11-10.
- [3] AI@Meta. Llama 3 model card. 2024.
- [4] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen, et al. Ms marco: A human generated machine reading comprehension dataset. ArXiv preprint, abs/1611.09268, 2016.
- [5] A. Barbaresi. Trafilatura: A web scraping library and command-line tool for text discovery and extraction. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations, pages 122–131, 2021.
- [6] J. Chen, H. Lin, X. Han, and L. Sun. Benchmarking large language models in retrieval-augmented generation. arXiv preprint arXiv:2309.01431, 2023.
- [7] J. Chen, H. Lin, X. Han, and L. Sun. Benchmarking large language models in retrieval-augmented generation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 17754–17762, 2024.
- [8] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. arXiv preprint arXiv:2402.03216, 2024.
- [9] X. Chen, L. Wang, W. Wu, Q. Tang, and Y. Liu. Honest ai: Fine-tuning "small" language models to say "i don't know", and reducing hallucination in rag. 2024 KDD Cup Workshop for Retrieval Augmented Generation, 2024.
- [10] Z. Chen, Z. Gu, L. Cao, J. Fan, S. Madden, and N. Tang. Symphony: Towards natural language query answering over multi-modal data lakes. In CIDR, 2023.
- [11] M. DeHaven. Marags: A multi-adapter system for multi-task retrieval augmented generation question answering. 2024 KDD Cup Workshop for Retrieval Augmented Generation, 2024.
- [12] S. Es, J. James, L. Espinosa Anke, and S. Schockaert. RAGAs: Automated evaluation of retrieval augmented generation. In N. Aletras and O. De Clercq, editors, Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 150–158, St. Julians, Malta, Mar. 2024. Association for Computational Linguistics.
- [13] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, Q. Guo, M. Wang, and H. Wang. Retrieval-augmented generation for large language models: A survey. 2024.
- [14] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. Retrieval augmented language model pre-training. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of Proceedings of Machine Learning Research, pages 3929–3938. PMLR, 2020.

- [15] R. Han, Y. Zhang, P. Qi, Y. Xu, J. Wang, L. Liu, W. Y. Wang, B. Min, and V. Castelli. Rag-qa arena: Evaluating domain robustness for long-form retrieval augmented question answering. [arXiv preprint arXiv:2407.13998](#), 2024.
- [16] L. He, R. Li, S. Shen, J. Lu, L. Zhu, Y. Su, and Z. Huang. A simple yet effective retrieval-augmented generation framework for the meta KDD cup 2024. In [2024 KDD Cup Workshop for Retrieval Augmented Generation](#), 2024.
- [17] G. Izacard and E. Grave. Leveraging passage retrieval with generative models for open domain question answering. In [Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume](#), pages 874–880, Online, 2021. Association for Computational Linguistics.
- [18] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung. Survey of hallucination in natural language generation. [ACM Comput. Surv.](#), 55(12), mar 2023.
- [19] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M.-W. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural questions: A benchmark for question answering research. [Transactions of the Association for Computational Linguistics](#), 7:452–466, 2019.
- [20] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021.
- [21] P. S. H. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, [Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual](#), 2020.
- [22] V. Liévin, C. E. Hother, A. G. Motzfeldt, and O. Winther. Can large language models reason about medical questions? [Patterns](#), 5(3), 2024.
- [23] J. Liu, R. Ding, L. Zhang, P. Xie, and F. Huang. Cofe-rag: A comprehensive full-chain evaluation framework for retrieval-augmented generation with enhanced data diversity. [arXiv preprint arXiv:2410.12248](#), 2024.
- [24] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. [ACM Computing Surveys](#), 55(9):1–35, 2023.
- [25] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, and H. Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. [ArXiv preprint, abs/2212.10511](#), 2022.
- [26] V. Mavroudis. Langchain. 2024.
- [27] ms-marco MiniLM-L-12-v2.
- [28] J. Ni, G. H. Abrego, N. Constant, J. Ma, K. B. Hall, D. Cer, and Y. Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. [arXiv preprint arXiv:2108.08877](#), 2021.

- [29] OpenAI. ChatGPT. <https://openai.com/index/chatgpt/>, 2023. Accessed: 2024-06-04.
- [30] J. Ouyang, Y. Luo, M. Cheng, D. Wang, S. Yu, Q. Liu, and E. Chen. Revisiting the solution of meta kdd cup 2024: Crag. 2024 KDD Cup Workshop for Retrieval Augmented Generation, 2024.
- [31] A. Panickssery, S. R. Bowman, and S. Feng. Llm evaluators recognize and favor their own generations. ArXiv preprint, abs/2404.13076, 2024.
- [32] S. Park, J. Seok, J. Lee, J. Yun, and W. Lee. KDD cup meta CRAG 2024 technical report: Three-step question-answering framework. In 2024 KDD Cup Workshop for Retrieval Augmented Generation, 2024.
- [33] R. Pradeep, N. Thakur, S. Sharifmoghaddam, E. Zhang, R. Nguyen, D. Campos, N. Craswell, and J. Lin. Ragnar\": ok: A reusable rag framework and baselines for trec 2024 retrieval-augmented generation track. arXiv preprint arXiv:2406.16828, 2024.
- [34] V. Rawte, S. Chakraborty, A. Pathak, A. Sarkar, S. Tonmoy, A. Chadha, A. P. Sheth, and A. Das. The troubling emergence of hallucination in large language models—an extensive definition, quantification, and prescriptive remediations. arXiv preprint arXiv:2310.04988, 2023.
- [35] L. Richardson. Beautiful soup documentation, 2007.
- [36] S. Robertson, H. Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval, 3(4):333–389, 2009.
- [37] J. Saad-Falcon, O. Khattab, C. Potts, and M. Zaharia. Ares: An automated evaluation framework for retrieval-augmented generation systems, 2023.
- [38] R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany, 2016. Association for Computational Linguistics.
- [39] K. Sun, Y. E. Xu, H. Zha, Y. Liu, and X. L. Dong. Head-to-tail: How knowledgeable are large language models (llm)? aka will llms replace knowledge graphs? ArXiv preprint, abs/2308.10168, 2023.
- [40] Y. Sun, H. Xin, K. Sun, Y. E. Xu, X. Yang, X. L. Dong, N. Tang, and L. Chen. Are large language models a good replacement of taxonomies? Proc. VLDB Endow., 17(11):2919–2932, aug 2024.
- [41] A. Talmor and J. Berant. The web as a knowledge-base for answering complex questions. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 641–651, New Orleans, Louisiana, 2018. Association for Computational Linguistics.
- [42] N. Tang, C. Yang, J. Fan, L. Cao, Y. Luo, and A. Y. Halevy. Verifai: Verified generative AI. In CIDR, 2024.
- [43] Y. Taya, D. Ito, S. Maeda, and Y. Hamano. RAG approach enhanced by category classification with BERT. In 2024 KDD Cup Workshop for Retrieval Augmented Generation, 2024.

- [44] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.
- [45] Trustbit. Enterprise rag challenge, 2024. Accessed: 2024-11-10.
- [46] R. Usbeck, X. Yan, A. Perevalov, L. Jiang, J. Schulz, A. Kraft, C. Möller, J. Huang, J. Reineke, A.-C. Ngonga Ngomo, et al. Qald-10—the 10th challenge on question answering over linked data. *Semantic Web*, (Preprint):1–15, 2023.
- [47] T. Vu, M. Iyyer, X. Wang, N. Constant, J. Wei, J. Wei, C. Tar, Y.-H. Sung, D. Zhou, Q. Le, et al. Freshllms: Refreshing large language models with search engine augmentation. *ArXiv preprint*, abs/2310.03214, 2023.
- [48] C. Wu, T. Shen, R. Yan, H. Wang, Z. Liu, Z. WANG, D. Lian, and E. Chen. Knowledge graph integration and self-verification for comprehensive retrieval-augmented generation. In *2024 KDD Cup Workshop for Retrieval Augmented Generation*, 2024.
- [49] Y. Xia, J. Chen, and J. Gao. Winning solution for meta kdd cup’ 24. *2024 KDD Cup Workshop for Retrieval Augmented Generation*, 2024.
- [50] S. Xiao, Z. Liu, P. Zhang, and N. Muennighoff. C-pack: Packaged resources to advance general chinese embedding, 2023.
- [51] W. Xie, X. Liang, Y. Liu, K. Ni, H. Cheng, and Z. Hu. Weknow-rag: An adaptive approach for retrieval-augmented generation integrating web search and knowledge graphs, 2024.
- [52] X. Yang, K. Sun, H. Xin, Y. Sun, N. Bhalla, X. Chen, S. Choudhary, R. D. Gui, Z. W. Jiang, Z. Jiang, et al. Crag-comprehensive rag benchmark. *ArXiv preprint*, abs/2406.04744, 2024.
- [53] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec. QA-GNN: Reasoning with language models and knowledge graphs for question answering. *Association for Computational Linguistics*, 2021.
- [54] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online, 2021. Association for Computational Linguistics.
- [55] Y. Yuan, C. Liu, J. Yuan, G. Sun, S. Li, and M. Zhang. A hybrid rag system with comprehensive enhancement on complex reasoning. *2024 KDD Cup Workshop for Retrieval Augmented Generation*, 2024.
- [56] J. Zhao and X. Liu. TCAF: a multi-agent approach of thought chain for retrieval augmented generation. In *2024 KDD Cup Workshop for Retrieval Augmented Generation*, 2024.

- [57] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in Neural Information Processing Systems, 36:46595–46623, 2023.
- [58] Y. Zhu, S. Du, B. Li, Y. Luo, and N. Tang. Are large language models good statisticians? In NeurIPS, 2024.



Data Engineering

It's FREE to join!

TCDE

tab.computer.org/tcde/

The Technical Committee on Data Engineering (TCDE) of the IEEE Computer Society is concerned with the role of data in the design, development, management and utilization of information systems.

- Data Management Systems and Modern Hardware/Software Platforms
- Data Models, Data Integration, Semantics and Data Quality
- Spatial, Temporal, Graph, Scientific, Statistical and Multimedia Databases
- Data Mining, Data Warehousing, and OLAP
- Big Data, Streams and Clouds
- Information Management, Distribution, Mobility, and the WWW
- Data Security, Privacy and Trust
- Performance, Experiments, and Analysis of Data Systems

The TCDE sponsors the International Conference on Data Engineering (ICDE). It publishes a quarterly newsletter, the Data Engineering Bulletin. If you are a member of the IEEE Computer Society, you may join the TCDE and receive copies of the Data Engineering Bulletin without cost. There are approximately 1000 members of the TCDE.

Join TCDE via Online or Fax

ONLINE: Follow the instructions on this page:

www.computer.org/portal/web/tandc/joinatc

FAX: Complete your details and fax this form to **+61-7-3365 3248**

Name _____

IEEE Member # _____

Mailing Address _____

Country _____

Email _____

Phone _____

<p>TCDE Mailing List</p> <p>TCDE will occasionally email announcements, and other opportunities available for members. This mailing list will be used only for this purpose.</p>	<p>Membership Questions?</p> <p>Xiaoyong Du Key Laboratory of Data Engineering and Knowledge Engineering Renmin University of China Beijing 100872, China duyong@ruc.edu.cn</p>	<p>TCDE Chair</p> <p>Xiaofang Zhou School of Information Technology and Electrical Engineering The University of Queensland Brisbane, QLD 4072, Australia zxf@uq.edu.au</p>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

IEEE Computer Society
10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314

Non-profit Org.
U.S. Postage
PAID
Los Alamitos, CA
Permit 1398