# SRAG: Structured Retrieval-Augmented Generation for Multi-Entity Question Answering over Wikipedia Graph

**Anonymous ACL submission**

## Abstract

Multi-entity question answering (MEQA) poses significant challenges for large language models (LLMs), which often struggle to consolidate scattered information across multiple documents. An example question might be "What is the distribution of IEEE Fellows among various fields of study?", which requires retrieving information from diverse sources *e.g.,* Wikipedia pages. The effectiveness of current retrieval-augmented generation (RAG) methods is limited by the LLMs' capacity to aggregate insights from numerous pages. To address this gap, this paper introduces a structured RAG (SRAG) framework that systematically organizes extracted entities into relational tables (*e.g.,* tabulating entities with schema columns like "name" and "field of study") and then apply table-based reasoning techniques. Our approach decouples retrieval and reasoning, enabling LLMs to focus on structured data analysis rather than raw text aggregation. Extensive experiments on Wikipedia-based multi-entity QA tasks demonstrate that SRAG significantly outperforms state-of-the-art RAG solutions, achieving a 29.6% improvement in accuracy. The results underscore the efficacy of structuring unstructured data to enhance LLMs' reasoning capabilities.

## 1 Introduction

Recent progress in Retrieval-Augmented Generation (RAG) has enhanced how language models access external knowledge, improving applications like question answering and document integration (Fan et al., 2024). By merging advanced retrieval methods with powerful language models, these systems have shown strong performance. However, challenges persist in accurately retrieving entity information from multi-document and heterogeneous knowledge bases. This challenge becomes especially apparent in Multi-Entity Question Answering, the challenge lies not only in recognizing and extracting relevant entities precisely from data but also in understanding the properties of these entities within the context of the query. Consider answering questions such as "What are the capitals of countries bordering France?" or "How many Turing Award Winners are Canadian" (the query $Q$ in Figure 1-a). Answering these questions requires the extraction of multiple documents, unless specific statistical analysis has been carried out by hand in advance. Existing RAG methods struggle with MEQA tasks, because useful information required to these tasks is badly scattered. This characteristic makes it difficult for existing RAG methods to accurately identify key information and perform global reasoning with noisy

retrieved content.

To tackle the challenge, we propose an innovative **Structured RAG System (SRAG)** which includes two main parts: (1) **Multi-entity Semantic Retrieval** as illustrated in Figure 1-a, and (2) **Structured Question Answering (SQA)** as illustrated in Figure 1-c, to effectively address multi-entity QA.

**Contributions** Our notable contributions are summarized as follows.

- **MEBench: Specialized Benchmark for MEQA.** We curate MEBench, a standardized benchmark based on Wikipedia to evaluate the effectiveness of various approaches to address the complexities of information extraction and reasoning involving multiple entities.
- **Multi-entity Semantic Retrieval.** Multi-entity Semantic Retrieval innovatively enhances semantic retrieval by integrating language models with structured database. This integration enables the creation of precise SPARQL queries to effectively retrieve relevant entities and web pages from Wikipedia. Our key contribution lies in improving retrieval accuracy by leveraging contextual brilliance of language models with the rigor of structured data validation.
- **SQA: Module for Structured Entity Information and Reasoning.** We propose the SQA, a module for managing vast and unstructured data by extracting properties of entities and organizing information into structured tables. This module transforms textual information of entities into a format with a rigorous and accurate schema, which facilitates analysis. Our experiments demonstrate its remarkable performance, achieving SOTA results and outperforming the strongest baselines by 29.6% in overall accuracy, while leading across all eight subtasks.

## 2 Related Work

### 2.1 Retrieval-Augmented Generation

Advanced RAG systems have evolved to incorporate pre-retrieval, retrieval, and post-retrieval strategies aimed at mitigating the limitations of basic RAG methods. In parallel, Modular RAG systems have introduced frameworks for iterative and dynamic cycles of intertwined retrieval and generation processes (Gao et al., 2023). Community summaries serve as a form of self-memory (Cheng et al., 2023), enhancing generation-augmented retrieval to support future cycles of information generation. Additionally, the parallel generation of community answers from these summaries reflects an iterative (Shao et al., 2023) or federated (Wang et al., 2024) retrieval-generation approach. Furthermore, using a hierarchical index and summarization aligns with alternative methods, such as generating a hierarchical index of text chunks through vector clustering (Sarthi et al., 2024). These methods concentrate on semantic aggregation and overlook the structured views of multiple entities.

### 2.2 Graph RAG

Recently, several studies have explored the incorporation of graph structures into RAG systems to enhance LLMs in addressing complex question-answering tasks (Edge et al., 2024) (Panda et al., 2024). One approach involves utilizing pre-existing knowledge graphs, from which subgraphs are extracted based on specific queries. These subgraphs are then encoded as soft prompts or converted into plain

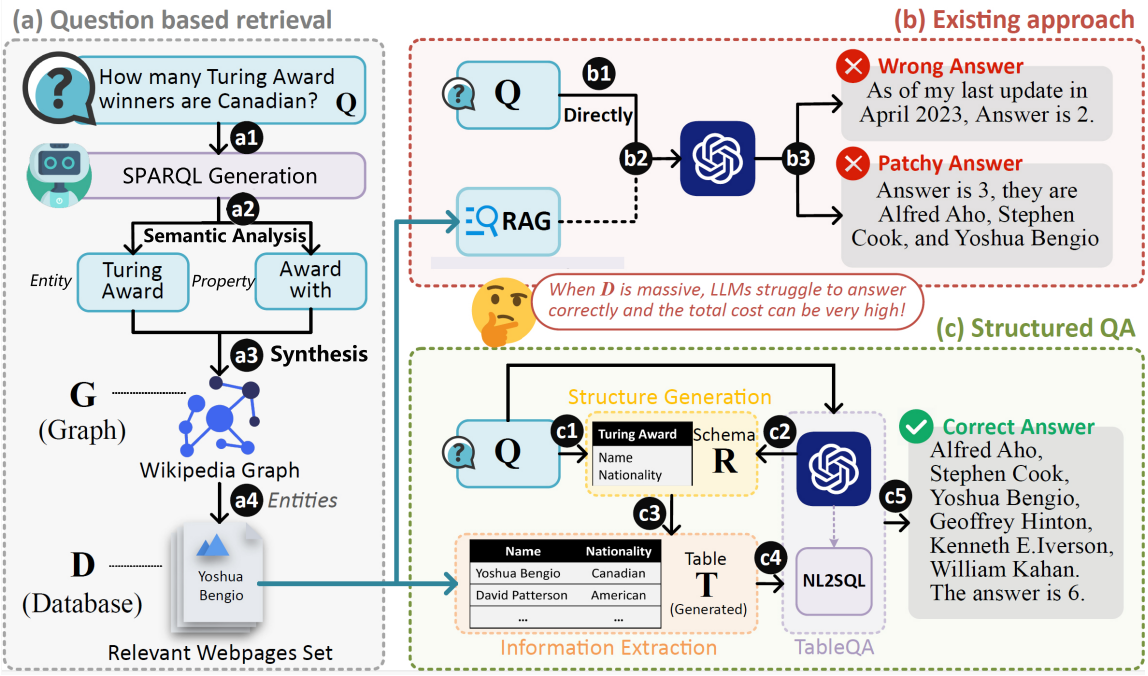


Figure 1: An Overview of Multi-entity QA Solutions over Wikipedia Graph. (a) **Multi-Entity Retrieval**: In a1, a rough SPARQL query is generated using language model (GPT-4). In a2, we integrate the LLM's semantic parsing with Wikipedia's API and utilize verifiable query accuracy on structured Wikidata to accurately identify entities and properties. In step a3, we synthesize an exact SPARQL query. Finally, in a4, the refined SPARQL query is used to retrieve the relevant entities and web pages. (b) **Existing Reasoning Solutions**: b1 represents direct responses from LLMs, while b2 combines LLMs with RAG. (c) Our proposal: **Structured QA**. Initially, in step c1, a language model (GPT-4) is employed to analyze the question and determine the **table schema**. In c3, we utilize an **information extraction module** to populate the form. Finally, in step c4, the **TableQA** module is used to derive the final answer.

text for the generation module GraphGPT (Tang et al., 2024) and other methods (He et al., 2024) (Guan et al., 2023). Another method focuses on extracting entity-relation triples from relevant text documents according to query needs, allowing for the construction of graph structures that facilitate knowledge augmentation, like GraphRAG (Edge et al., 2024) and RQ-RAG (Chan et al., 2024). These models are inadequate for addressing multi-entity questions. In response, we introduce a comprehensive Multi-Entities QA benchmark MEBench based on Wikipedia Graph and provide a first-of-its-kind original system SRAG.

## 3 Problem Statement

### 3.1 Wikipedia Graph

Wikipedia graph is represented as $G=(V, E, P)$, where $V = \{v_1, v_2, ..., v_n\}$ is the set of nodes in the graph, with each node ($v_i \in V$) representing an entity; $E$ is the set of direct edges in the graph, with each edge

$(e_j(v_i, v_k) \in E)$ representing a connection (or relationships) between two nodes. An edge is a tuple $(v_a, v_b, r)$, where $(v_a, v_b \in V)$ and $r$ is the type of relationship. For example, $E = \{(v_1, v_2, r_1), (v_2, v_3, r_2), ..., (v_m, v_n, r_k)\}$. $P$ represents the set of properties associated with both nodes and edges. Figure 2 is a sample of a Wikipedia graph snippet.

### 3.2 Multi-Entities Question

A Multi-Entities Question can be formally defined as $Q = (t_Q, V_Q, P_Q)$, where $t_Q \in T$ denotes the query type, with $T = \{t_1, t_2, \ldots, t_8\}$ representing the set of eight predefined types. Details can be seen in Table 1. $V_Q$ denotes the collection of entities directly associated with the question. $P_Q$ represents the comprehensive set of properties pertinent to the question, encompassing both node and edge properties.

## 4 System Design

In our system, the **Multi-entity Semantic Retrieval** involves conducting a SPARQL retrieval across the Wikipedia graph to obtain relevant Wikipedia pages. Secondly, in SQA module, **table generation** begins with "guessing" a table schema based on the given query, followed by the extraction of information from the identified entities to populate the table. Finally, we implement an **executor** that processes the generated table to respond to the query. Next, we will elaborate on the details of each step.

### 4.1 Composite SPARQL Retrieval

Initially, we utilize GPT-4 to parse the question to construct rough SPARQL.The entities ID and properties ID contained in the rough SPARQL frequently turn out to be inaccurate. To make SPARQL valid, we deploy GPT-4 as Semantic Analysis Model to identify entities and properties. Integrating with the Wikipedia API, we get right `(entityID, propertyID)` pair to replacing the wrong IDs in the rough SPARQL, as illustrated in Figure 1-a2. Consequently, our system is capable of performing entity and property identification without ambiguities. For multi-hop queries, the Semantic Analysis process initially deconstructs the queries into sub-queries, allowing composite SPARQL retrieval step to be applied sequentially to each sub-query to identify named entities and extract properties until all sub-queries have been processed.

### 4.2 Table Generation

Although SPARQL provides aggregation functions such as "SUM, AVG, COUNT", etc. they are insufficient for complex statistical problems. Therefore, we build tables instead of extended graphs to support more complex algorithms and analysis. The table generation consists of two steps: (1) Generation of schema; (2) Extracting entity information to fill the table.

**Schema Generation.** We employ GPT-4 to systematically parsing the question to identify critical entities, attributes, and their interrelationships, which are then formalized into a structured schema. The generated schema may necessitate adjustments or refinements to align more closely with the intent of the question. For example, for the question "How many Nobel Prizes in Physics laureates have been awarded for discoveries in Particle Physics?", the LLM produces a schema `(name, YearAwarded, field)`, there are two issues, first is that `(field)` is oversimplified, it should be `(field in Physics)`. The second issue is that columns `YearAwarded` are redundant. Therefore, We prompt GPT-4 to critically review content to minimize oversimplification, omission of essential elements, and redundancy, and the prompt is shown in Appendix A.2.1.

**Entity Information Extraction.** In our data


Figure 2: Illustration of a Wikipedia graph snippet.

processing workflow, we use the LLM (Mistral-7B) to extract information from the retrieved data, populating a table where each row represents a unique entity, as shown in Figure 1-c3. This step transforms entities into structured tables for downstream table-based reasoning tasks.

### 4.3 Executor

We utilize GPT-4 to generate SQL according to the question. To increase accuracy, we include relevant information in the prompt, such as the table schema and data samples. The generated SQL are executed on the generated table to obtain results, which could be a single value or a subset of the table.

## 5 Experiment

### 5.1 Experiment Setup

**MEBench Benchmark.** It is a specialized benchmark designed to evaluate systems addressing multi-entity QA. The benchmark comprises 4,780 methodically structured questions partitioned into two subsets: a training set (3,406 questions) for model fine-tuning and a test set (1,374 questions) for rigorous evaluation. These questions are systematically categorized into three primary categories, further divided into eight distinct types (see Table 1), ensuring broad coverage of real-world multi-entity reasoning scenarios. Table 2 details comprehensive statistics of the benchmark.

**Baselines.** For open-source LLMs, we conduct experiments using the representative Meta-Llama-3-8B-Instruct (Meta Llama3, 2024) and apply QLoRA (Dettmers et al., 2023) to fine-tune it with the training set of MEBench. For proprietary LLMs, we select the widely recognized GPT models, including GPT-3.5-turbo (Ouyang et al., 2022) and GPT-4 (Achiam et al., 2023). Additionally, we incorporate RAG across all vanilla baseline models for comparative analysis and evaluation of the model's capacity to integrate and leverage external data sources.

**Evaluation Metrics.** We adopt Accuracy ($Acc$) as the primary metric to assess the performance of LLMs on MEBench tasks. For the subcategories of Variance Analysis, Correlation Analysis, and Distribution Compliance within the Statistics tasks shown in Table 1, we focus solely on prompting LLMs to identify relevant columns and applicable methods, evaluating the accuracy of their selections instead of the computational results, as LLMs' abilities in precise calculations are not the central focus of this study.

Table 1: Examples of multi-entities queries.

| Categories | Types | Examples |
|---|---|---|
| Comparison | Intercomparison | Which has more ACM fellow, UK or USA? |
| | Superlative | Which city has the highest population? |
| Statistics | Aggregation | How many ACM fellow are from MIT? |
| | Distribution Compliance | Does the nationality of ACM fellows follow a normal distribution? |
| | Correlation Analysis | Is there a linear relationship between number of events and records broken in Olympic Games? |
| | Variance Analysis | Do the variances in the number of participating countries and total events in the Summer Olympics differ significantly? |
| Relationship | Descriptive Relationship | Is there a relationship between the year of ACM fellowship induction and the fellows' areas of expertise? |
| | Hypothetical Scenarios | If China wins one more gold medal, will it overtake the US in the gold medal tally at the 2024 Olympics? |

Table 2: Statistics of MEBench benchmark.

| Categories | MEBench-train | MEBench-test | MEBench-total |
|---|---|---|---|
| #-Queries | 3406 | 1374 | 4780 |
| #-one-hop Q | 1406 | 606 | 2012 |
| #-multi-hop Q | 1322 | 768 | 2090 |
| Ave. #-entities /Q | 460 | 391 | 409 |
| #-Topics | 165 | 76 | 241 |
| #-Comparison | 1107 | 438 | 1545 |
| #-Statistics | 1440 | 585 | 2025 |
| #-Relationship | 859 | 351 | 1210 |

### 5.2 Results and Analysis

Various models exhibit notable variations in performance on MEBench. Table 3 presents experimental results alongside overall accuracy on MEBench, and Figure 3 shows accuracy on eight further-divided types.

**Performance of SRAG and Baselines.** Compared to baselines, our SRAG significantly im-

Table 3: Experimental results for MEBench.

| Models | Accuracy | | | |
|---|---|---|---|---|
| | Comparison | Statistics | Relationship | Overall |
| GPT-3.5-turbo | 0.105 | 0.198 | 0.476 | 0.239 |
| GPT-3.5-turbo + RAG | 0.605 | 0.260 | 0.476 | 0.425 |
| GPT-4 | 0.199 | 0.289 | 0.507 | 0.316 |
| GPT-4 + RAG | 0.763 | 0.410 | 0.687 | 0.593 |
| Llama-3-Instruct | 0.046 | 0.118 | 0.256 | 0.130 |
| Llama-3-Instruct + RAG | 0.447 | 0.181 | 0.410 | 0.325 |
| FT Llama-3-Instruct | 0.046 | 0.253 | 0.259 | 0.189 |
| FT Llama-3-Instruct + RAG | 0.687 | 0.448 | 0.573 | 0.556 |
| **SRAG (Ours)** | **0.934** | **0.908** | **0.803** | **0.889** |

proves overall accuracy, reaching 88.9% and increasing the best baseline (GPT-4 + RAG) by **29.6%**. Our approach outperforms the accuracy in the relational and comparative query types by 11.6% and 17.1%, respectively, while achieving a remarkable improvement of 46% for statistical query types.

**Fine-grained Performance on Sub-tasks.** Figure 3 shows that vanilla LLMs perform well in correlation analysis and descriptive relationship sub-tasks, while RAG significantly improves intercomparison and superlative tasks. However, neither fine-tuning nor RAG overcomes challenges in variance analysis and aggregation tasks, while our proposed SRAG achieves superior accuracy of 87.3% and 97.9%.

**Errors Analysis for SRAG.** We sample and analyze the output of the SRAG system. It faces two challenges which are listed below.

- **Relation semantic parsing.** The semantic parsing model in SPARQL retrieval effectively recognizes entities but struggles with relationship identification, leading to challenges in graph retrieval and negatively affecting the performance of RAG-based approaches, including SQA. For example, in the query "How many US presidents have served more than one term in office?" The model incorrectly identifies the relationship as "instance of" rather than the correct "position held", leading to erroneous results.
- **Insufficient information extraction.** We also identified errors in SRAG's information extraction during the table-filling phase. An analysis of more than 2,000 table filling instances reveals that these errors occur primarily as omissions (albeit with a low probability of approximately 0.1%). A new challenge is the appearance of multi-word synonyms within the same column, such as "US" and "America", which negatively affects the accuracy of SQL execution such as "SELECT".

## 6 Conclusion

Our research presents a novel framework, Structured RAG system (SRAG), to address the complexities involved in multi-entity question an-

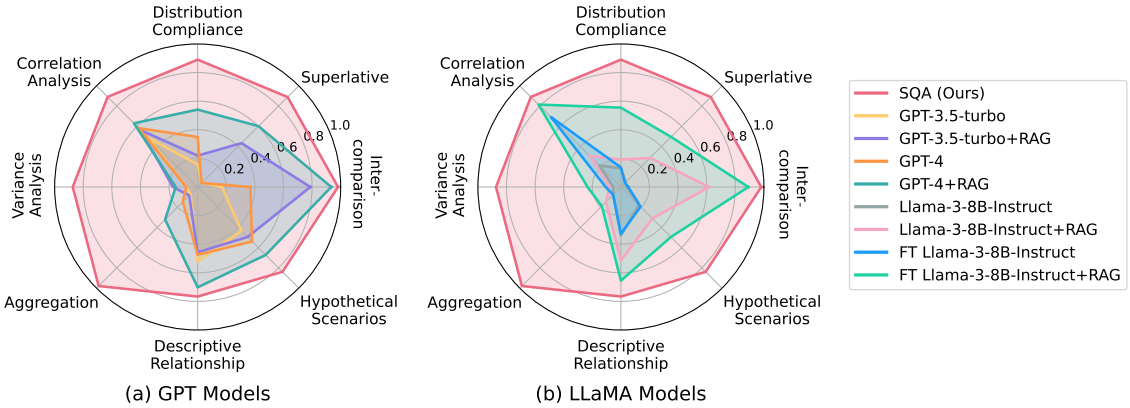


Figure 3: Experimental results for eight types queries of each model.

swering (QA) from Wikipedia. Existing methods, particularly those employing RAG alongside LLMs, often fall short in effectively aggregating and reasoning over information scattered across multiple Wikipedia pages. By leveraging the inherent structure of wiki-graph for multi-entity retrieval and introducing a system to organize extracted entities into a relational table format, SRAG significantly enhances the performance of multi-entity questions answering. The exhaustive experiments conducted underscore the superior performance of SRAG in overcoming the limitations of traditional RAG-based solutions. This research not only presents a more effective methodology for multi-entity QA but also sets the stage for future explorations into improving the accuracy and efficiency of information mining from large, unstructured knowledge bases.

## 7 Limitations

The limitations identified in the error analysis in Section 5.2 emphasize the need for better semantic understanding of queries and the use of large language models (LLMs) to improve data extraction accuracy. Another key limitation of the system is its ability to handle ambiguous queries, which in real-world scenarios can vary significantly in complexity. This requires advanced semantic inference to accurately grasp user intent, demanding the system to comprehend both explicit content and implicit context. The challenge is to balance precision and recall, allowing diverse query handling while minimizing errors. Also, we did not compare the SQL query-based table method with simpler techniques such as keyword extraction and summarization. This was based on the assumption that SQL's advantages in structured query handling are apparent. However, this overlooks the potential research value such a comparison could provide.

## References

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

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yi-Ting Guo, and Jie Fu. 2024.

Rq-rag: Learning to refine queries for retrieval augmented generation. *ArXiv*, abs/2404.00610.

Xin Cheng, Di Luo, Xiuying Chen, Lemao Liu, Dongyan Zhao, and Rui Yan. 2023. Lift yourself up: Retrieval-augmented text generation with self memory.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2023. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering.

Chunwei Liu, Matthew Russo, Michael Cafarella, Lei Cao, Peter Baille Chen, Zui Chen, Michael Franklin, Tim Kraska, Samuel Madden, and Gerardo Vitagliano. 2024. A declarative system for optimizing ai workloads. *arXiv preprint arXiv:2405.14696*.

Meta Llama3. 2024. Meta llama3. `https://llama.meta.com/llama3/`. Accessed: 2024-04-10.

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

Pranoy Panda, Ankush Agarwal, Chaitanya Devaguptapu, Manohar Kaul, and Prathosh Ap. 2024. HOLMES: Hyper-relational knowledge graphs for multi-hop question answering using LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13263–13282, Bangkok, Thailand. Association for Computational Linguistics.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Íñigo Goiri, Aashaka Shah, Saeed Maleki, and Ricardo Bianchini. 2023. Splitwise: Efficient generative llm inference using phase splitting. *arXiv preprint arXiv:2311.18677*.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 491–500, New York, NY, USA. Association for Computing Machinery.

Shuai Wang, Ekaterina Khramtsova, Shengyao Zhuang, and Guido Zuccon. 2024. Feb4rag: Evaluating federated search in the context of retrieval augmented generation.

# A Appendix

## A.1 Hops

In terms of Wikipedia graph systems, the term 'hop' refers to a step taken along the edges of a graph from one node to another, so we consider 'hop' as a tuple $(v_a, v_b, t)$. For no hop, there is no relationship (edge) to track. Single hop track all entities have relationship $t$ to $v_a$. Multi-hop involves traversing multiple edges (hops) to find connections between nodes that are not directly linked, like track $(v_a, v_b, t_1)$ $(v_b, v_c, t_2)$.

## A.2 Prompt

### A.2.1 Prompt for schema

> Create a table schema that comprehensively captures information about {.....}. Ensure the schema is detailed and structured, avoiding over-simplification, missing elements, and redundancy. This schema should be structured so each row represents a unique instance, with each column capturing a distinct aspect of property details. Ensure there is no overlap in content between columns to avoid repetition.

## A.3 Optimization

Two aspects of optimization are included in SRAG system to enhance the overall performance:

**Model Selection.** Model selection is straightforward yet highly effective for optimization (Liu et al., 2024). The SRAG system comprises multiple tasks, necessitating the selection of the most suitable model for different tasks. For basic tasks, more affordable and faster LLMs can suffice, while utilization of the most advanced LLMs is essential in more complex tasks to ensure optimal performance. Specifically, SRAG system employs powerful yet resource-intensive GPT-4 for tasks such as semantic analysis or generation of table schemas and SQL queries. In contrast, for more basic information extraction, we utilize open-source Mistral-7B, thereby achieving a balance between cost efficiency and functional performance.

**LLM Input/Output Control**. SplitWise (Patel et al., 2023) shows that LLM inference time is generally proportional to the size of input and output tokens. Since GPT models decide the cost based on the input token, we try to minimize the input of large models. Meanwhile, we use the instructive prompt to reduce the size of the outputs generated by LLM without changing the quality of these outputs. The example of prompt is in Appendix A.3.1.

### A.3.1 Prompt for Output Control

> ...review your output to ensure it meets all the above criteria. Your goal is to produce a clear, accurate, and well-structured output. Just output the {}, no other word or symbol.

## A.4 Tables

Table 4 shows examples of topics and their entities' properties.

Table 5 shows examples of question templates to synthesize queries.

## A.5 Automated QA Generation and Validation

We extract the introductory paragraph of textual content for each entity from Wikipedia, akin to an abstract of the entity's page, to derive relevant property values. The preprocessing of graph node properties is conducted using GPT-4. GPT-4 is deployed to generate the essential properties of key entities for each topic, and subsequently, property values are extracted from the respective web pages of these entities. This process

Table 4: Example Topics and Their Entities Properties.

| Topics | Entities Properties | #-Entities |
|---|---|---|
| ACM fellow | nationality, field of study, affiliation | 1115 |
| Cities of the World | population, geographic coordinates, altitude, GDP | 7040 |
| Presidents of the US | term lengths, political parties, vice-presidents, birth states, previous occupations | 55 |
| Chemical Elements | atomic number, atomic mass, boiling point, melting point, electron configuration | 166 |
| Summer Olympic Games | host cities, number of participating countries, total number of events, medal tally, records broken | 35 |
| Nobel Prize in Chemistry | categories, year of award, country of origin, field of contribution. | 194 |

culminates in the formation of property tables. An illustrative example of the topics and entities' properties is provided in Appendix Table 4.

When questions or queries are posed, the SRAG system efficiently navigates the graph by utilizing both the connections (edges) and the nodes along with the associated property tables to retrieve relevant information. The property tables, which contain attributes and values related to the entities within the graph, serve as a comprehensive and structured data source that can be queried alongside the graph structure. This dual approach facilitates thorough analysis, as it takes into account both the relational context (the connections among entities) and the specific properties of the entities involved. Moreover, such automated process benefits from low labor costs due to automation and optimization within the graph database system, reducing the need for time-consuming and error-prone manual data processing and analysis.

### A.6 Quality Control of Questions

We devise several strategies to ensure the integrity and effectiveness of questions.

- **Question Templates.** The use of templates ensures that every question is crafted with a clear structure, making it easier for respondents to understand and answer them accurately. For relationship and complex statistic questions, we turn the questions in a closed-ended style, as they require a specific response of either "yes" or "no", which makes the answer in a standardized format. We meticulously prepare all question templates, with examples in the Appendix Table 5.
- **Question Refinement.** After the initial development phase, each question undergoes a refinement process utilizing GPT-3.5-turbo. This stage is essential for improving the clarity, relevance, and neutrality of the questions. It also includes a thor-

Table 5: Template example for queries generated by the LLM (GPT-4).

| Categories | Types | Template Examples |
|---|---|---|
| Comparison | Intercomparison | Which has high [property], [entity A] or [entity B]? |
| | Superlative | Which [entity] has the highest/lowest [property]? |
| Statistics | Aggregation | How many [entities] have [specific property value]? |
| | Distribution Compliance | Does [property] follow a normal distribution? |
| | Correlation Analysis | Is there a linear relationship between [property A] and [property B]? |
| | Variance Analysis | Are the variances in [property A] and [property B] significantly different? |
| Relationship | Descriptive Relationship | How is [entity A] related to [entity B]? |
| | Hypothetical Scenarios | What would be the impact if [entity A] collaborates with [entity B]? |

ough review to identify and mitigate any potential bias, contributing to minimizing misunderstandings and elevating the overall quality of the questions.

- **Manual review.** We assess the questions for accuracy, ensuring they are factually correct and relevant to our purpose. Manual reviews can also provide insights into whether the questions are likely to effectively elicit the intended information from answers, thereby contributing to the reliability and validity of the benchmark.

### A.7 Baseline Performance.

Introducing RAG significantly improves overall performance, particularly in comparison tasks, while fine-tuning LLaMA-3-Instruct alone does not yield substantial gains without RAG. On MEBench, open-source models like LLaMA-3-Instruct, even with RAG, can't match proprietary models like GPT-4, which achieves a 59.3% accuracy compared to LLaMA-3-Instruct's 31.6%.