From Local to Global: A Graph RAG Approach to Query-Focused Summarization

Darren Edge[1][†]   **Ha Trinh**[1][†]   **Newman Cheng**[2]   **Joshua Bradley**[2]   **Alex Chao**[3]

Apurva Mody[3]   **Steven Truitt**[2]

Jonathan Larson[1]

[1]Microsoft Research
[2]Microsoft Strategic Missions and Technologies
[3]Microsoft Office of the CTO

{daedge,trinhha,newmancheng,joshbradley,achao,moapurva,steventruitt,jolarso}
@microsoft.com

[†]These authors contributed equally to this work

Abstract

The use of retrieval-augmented generation (RAG) to retrieve relevant information from an external knowledge source enables large language models (LLMs) to answer questions over private and/or previously unseen document collections. However, RAG fails on global questions directed at an entire text corpus, such as "What are the main themes in the dataset?", since this is inherently a query-focused summarization (QFS) task, rather than an explicit retrieval task. Prior QFS methods, meanwhile, fail to scale to the quantities of text indexed by typical RAG systems. To combine the strengths of these contrasting methods, we propose a Graph RAG approach to question answering over private text corpora that scales with both the generality of user questions and the quantity of source text to be indexed. Our approach uses an LLM to build a graph-based text index in two stages: first to derive an entity knowledge graph from the source documents, then to pregenerate community summaries for all groups of closely-related entities. Given a question, each community summary is used to generate a partial response, before all partial responses are again summarized in a final response to the user. For a class of global sensemaking questions over datasets in the 1 million token range, we show that Graph RAG leads to substantial improvements over a naïve RAG baseline for both the comprehensiveness and diversity of generated answers. An open-source, Python-based implementation of both global and local Graph RAG approaches is forthcoming at https://aka.ms/graphrag.

1 Introduction

Human endeavors across a range of domains rely on our ability to read and reason about large collections of documents, often reaching conclusions that go beyond anything stated in the source texts themselves. With the emergence of large language models (LLMs), we are already witnessing attempts to automate human-like sensemaking in complex domains like scientific discovery (Microsoft, 2023) and intelligence analysis (Ranade and Joshi, 2023), where sensemaking is defined as
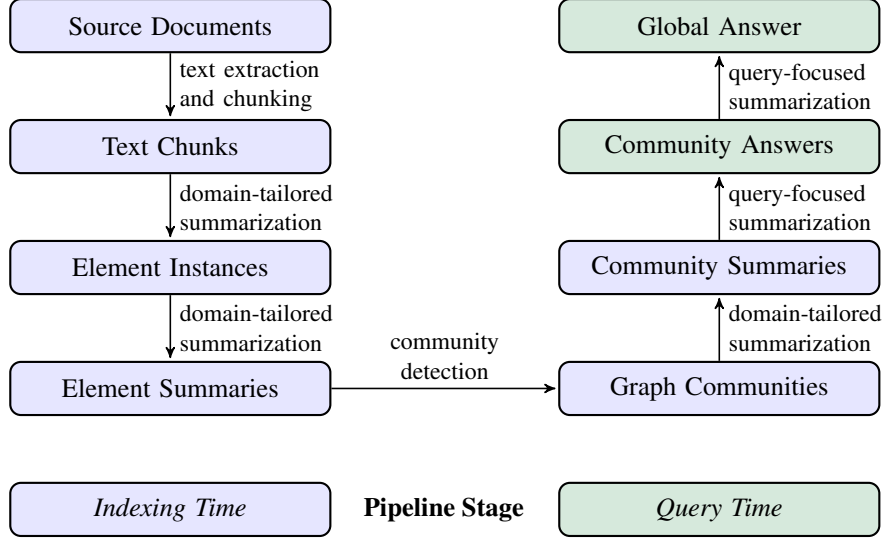
Preprint. Under review.

Figure 1: Graph RAG pipeline using an LLM-derived graph index of source document text. This index spans nodes (e.g., entities), edges (e.g., relationships), and covariates (e.g., claims) that have been detected, extracted, and summarized by LLM prompts tailored to the domain of the dataset. Community detection (e.g., Leiden, Traag et al., 2019) is used to partition the graph index into groups of elements (nodes, edges, covariates) that the LLM can summarize in parallel at both indexing time and query time. The "global answer" to a given query is produced using a final round of query-focused summarization over all community summaries reporting relevance to that query.

"*a motivated, continuous effort to understand connections (which can be among people, places, and events) in order to anticipate their trajectories and act effectively*" (Klein et al., 2006a). Supporting human-led sensemaking over entire text corpora, however, needs a way for people to both apply and refine their mental model of the data (Klein et al., 2006b) by asking questions of a global nature.

Retrieval-augmented generation (RAG, Lewis et al., 2020) is an established approach to answering user questions over entire datasets, but it is designed for situations where these answers are contained locally within regions of text whose retrieval provides sufficient grounding for the generation task. Instead, a more appropriate task framing is query-focused summarization (QFS, Dang, 2006), and in particular, query-focused *abstractive* summarization that generates natural language summaries and not just concatenated excerpts (Baumel et al., 2018; Laskar et al., 2020; Yao et al., 2017) . In recent years, however, such distinctions between summarization tasks that are abstractive versus extractive, generic versus query-focused, and single-document versus multi-document, have become less relevant. While early applications of the transformer architecture showed substantial improvements on the state-of-the-art for all such summarization tasks (Goodwin et al., 2020; Laskar et al., 2022; Liu and Lapata, 2019), these tasks are now trivialized by modern LLMs, including the GPT (Achiam et al., 2023; Brown et al., 2020), Llama (Touvron et al., 2023), and Gemini (Anil et al., 2023) series, all of which can use in-context learning to summarize any content provided in their context window.

The challenge remains, however, for query-focused abstractive summarization over an entire corpus. Such volumes of text can greatly exceed the limits of LLM context windows, and the expansion of such windows may not be enough given that information can be "lost in the middle" of longer contexts (Kuratov et al., 2024; Liu et al., 2023). In addition, although the direct retrieval of text chunks in naïve RAG is likely inadequate for QFS tasks, it is possible that an alternative form of pre-indexing could support a new RAG approach specifically targeting global summarization.

In this paper, we present a *Graph RAG* approach based on global summarization of an LLM-derived knowledge graph (Figure 1). In contrast with related work that exploits the structured retrieval and traversal affordances of graph indexes (subsection 4.2), we focus on a previously unexplored quality of graphs in this context: their inherent *modularity* (Newman, 2006) and the ability of community detection algorithms to partition graphs into modular communities of closely-related nodes (e.g., Louvain, Blondel et al., 2008; Leiden, Traag et al., 2019). LLM-generated summaries of these
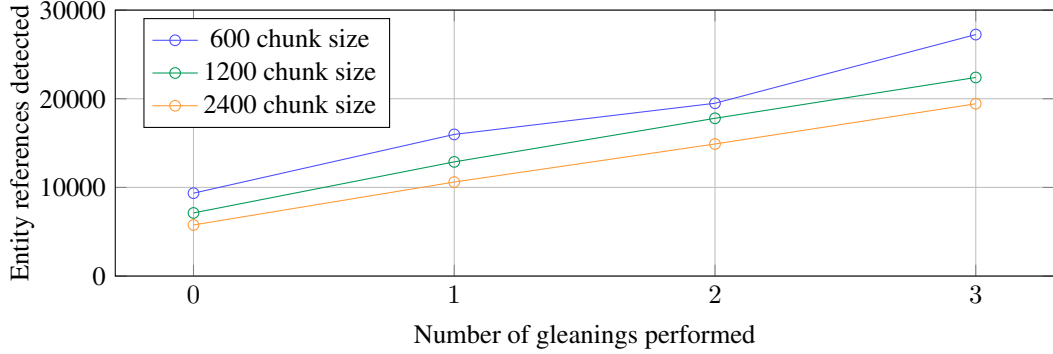
Figure 2: How the entity references detected in the HotPotQA dataset (Yang et al., 2018) varies with chunk size and gleanings for our generic entity extraction prompt with `gpt-4-turbo`.

community descriptions provide complete coverage of the underlying graph index and the input documents it represents. Query-focused summarization of an entire corpus is then made possible using a map-reduce approach: first using each community summary to answer the query independently and in parallel, then summarizing all relevant partial answers into a final global answer.

To evaluate this approach, we used an LLM to generate a diverse set of activity-centered sense-making questions from short descriptions of two representative real-world datasets, containing podcast transcripts and news articles respectively. For the target qualities of comprehensiveness, diversity, and empowerment (defined in subsection 3.4) that develop understanding of broad issues and themes, we both explore the impact of varying the the hierarchical level of community summaries used to answer queries, as well as compare to naïve RAG and global map-reduce summarization of source texts. We show that all global approaches outperform naïve RAG on comprehensiveness and diversity, and that Graph RAG with intermediate- and low-level community summaries shows favorable performance over source text summarization on these same metrics, at lower token costs.

2 Graph RAG Approach & Pipeline

We now unpack the high-level data flow of the Graph RAG approach (Figure 1) and pipeline, describing key design parameters, techniques, and implementation details for each step.

2.1 Source Documents → Text Chunks

A fundamental design decision is the granularity with which input texts extracted from source documents should be split into text chunks for processing. In the following step, each of these chunks will be passed to a set of LLM prompts designed to extract the various elements of a graph index. Longer text chunks require fewer LLM calls for such extraction, but suffer from the recall degradation of longer LLM context windows (Kuratov et al., 2024; Liu et al., 2023). This behavior can be observed in Figure 2 in the case of a single extraction round (i.e., with zero gleanings): on a sample dataset (HotPotQA, Yang et al., 2018), using a chunk size of 600 token extracted almost twice as many entity references as when using a chunk size of 2400. While more references are generally better, any extraction process needs to balance recall and precision for the target activity.

2.2 Text Chunks → Element Instances

The baseline requirement for this step is to identify and extract instances of graph nodes and edges from each chunk of source text. We do this using a multipart LLM prompt that first identifies all *entities* in the text, including their name, type, and description, before identifying all *relationships* between clearly-related entities, including the source and target entities and a description of their relationship. Both kinds of element instance are output in a single list of delimited tuples.

The primary opportunity to tailor this prompt to the domain of the document corpus lies in the choice of few-shot examples provided to the LLM for in-context learning (Brown et al., 2020).

For example, while our default prompt extracting the broad class of "named entities" like people, places, and organizations is generally applicable, domains with specialized knowledge (e.g., science, medicine, law) will benefit from few-shot examples specialized to those domains. We also support a secondary extraction prompt for any additional covariates we would like to associate with the extracted node instances. Our default covariate prompt aims to extract *claims* linked to detected entities, including the subject, object, type, description, source text span, and start and end dates.

To balance the needs of efficiency and quality, we use multiple rounds of "gleanings", up to a specified maximum, to encourage the LLM to detect any additional entities it may have missed on prior extraction rounds. This is a multi-stage process in which we first ask the LLM to assess whether all entities were extracted, using a logit bias of 100 to force a yes/no decision. If the LLM responds that entities were missed, then a continuation indicating that "MANY entities were missed in the last extraction" encourages the LLM to glean these missing entities. This approach allows us to use larger chunk sizes without a drop in quality (Figure 2) or the forced introduction of noise.

2.3 Element Instances → Element Summaries

The use of an LLM to "extract" descriptions of entities, relationships, and claims represented in source texts is already a form of abstractive summarization, relying on the LLM to create independently meaningful summaries of concepts that may be implied but not stated by the text itself (e.g., the presence of implied relationships). To convert all such instance-level summaries into single blocks of descriptive text for each graph element (i.e., entity node, relationship edge, and claim covariate) requires a further round of LLM summarization over matching groups of instances.

A potential concern at this stage is that the LLM may not consistently extract references to the same entity in the same text format, resulting in duplicate entity elements and thus duplicate nodes in the entity graph. However, since all closely-related "communities" of entities will be detected and summarized in the following step, and given that LLMs can understand the common entity behind multiple name variations, our overall approach is resilient to such variations provided there is sufficient connectivity from all variations to a shared set of closely-related entities.

Overall, our use of rich descriptive text for homogeneous nodes in a potentially noisy graph structure is aligned with both the capabilities of LLMs and the needs of global, query-focused summarization. These qualities also differentiate our graph index from typical knowledge graphs, which rely on concise and consistent knowledge triples (subject, predicate, object) for downstream reasoning tasks.

2.4 Element Summaries → Graph Communities

The index created in the previous step can be modelled as an homogeneous undirected weighted graph in which entity nodes are connected by relationship edges, with edge weights representing the normalized counts of detected relationship instances. Given such a graph, a variety of community detection algorithms may be used to partition the graph into communities of nodes with stronger connections to one another than to the other nodes in the graph (e.g., see the surveys by Fortunato, 2010 and Jin et al., 2021). In our pipeline, we use Leiden (Traag et al., 2019) on account of its ability to recover hierarchical community structure of large-scale graphs efficiently (Figure 3). Each level of this hierarchy provides a community partition that covers the nodes of the graph in a mutually-exclusive, collective-exhaustive way, enabling divide-and-conquer global summarization.

2.5 Graph Communities → Community Summaries

The next step is to create report-like summaries of each community in the Leiden hierarchy, using a method designed to scale to very large datasets. These summaries are independently useful in their own right as a way to understand the global structure and semantics of the dataset, and may themselves be used to make sense of a corpus in the absence of a question. For example, a user may scan through community summaries at one level looking for general themes of interest, then follow links to the reports at the lower level that provide more details for each of the subtopics. Here, however, we focus on their utility as part of a graph-based index used for answering global queries.

Community summaries are generated in the following way:

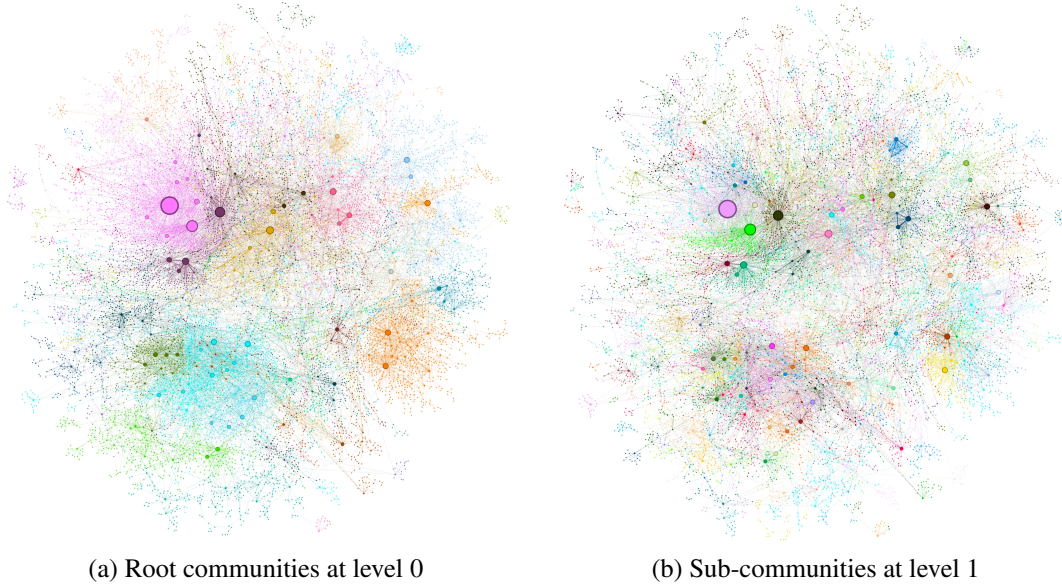(a) Root communities at level 0   (b) Sub-communities at level 1

Figure 3: Graph communities detected using the Leiden algorithm (Traag et al., 2019) over the MultiHop-RAG (Tang and Yang, 2024) dataset as indexed. Circles represent entity nodes with size proportional to their degree. Node layout was performed via OpenORD (Martin et al., 2011) and Force Atlas 2 (Jacomy et al., 2014). Node colors represent entity communities, shown at two levels of hierarchical clustering: (a) Level 0, corresponding to the hierarchical partition with maximum modularity, and (b) Level 1, which reveals internal structure within these root-level communities.

- *Leaf-level communities*. The element summaries of a leaf-level community (nodes, edges, covariates) are prioritized and then iteratively added to the LLM context window until the token limit is reached. The prioritization is as follows: for each community edge in decreasing order of combined source and target node degree (i.e., overall prominence), add descriptions of the source node, target node, linked covariates, and the edge itself.

- *Higher-level communities*. If all element summaries fit within the token limit of the context window, proceed as for leaf-level communities and summarize all element summaries within the community. Otherwise, rank sub-communities in decreasing order of element summary tokens and iteratively substitute sub-community summaries (shorter) for their associated element summaries (longer) until fit within the context window is achieved.

2.6 Community Summaries → Community Answers → Global Answer

Given a user query, the community summaries generated in the previous step can be used to generate a final answer in a multi-stage process. The hierarchical nature of the community structure also means that questions can be answered using the community summaries from different levels, raising the question of whether a particular level in the hierarchical community structure offers the best balance of summary detail and scope for general sensemaking questions (evaluated in section 3).

For a given community level, the global answer to any user query is generated as follows:

- *Prepare community summaries*. Community summaries are randomly shuffled and divided into chunks of pre-specified token size. This ensures relevant information is distributed across chunks, rather than concentrated (and potentially lost) in a single context window.

- *Map community answers*. Generate intermediate answers in parallel, one for each chunk. The LLM is also asked to generate a score between 0-100 indicating how helpful the generated answer is in answering the target question. Answers with score 0 are filtered out.

- *Reduce to global answer*. Intermediate community answers are sorted in descending order of helpfulness score and iteratively added into a new context window until the token limit is reached. This final context is used to generate the global answer returned to the user.

5

Dataset	Example activity framing and generation of global sensemaking questions
Podcast transcripts	*User*: A tech journalist looking for insights and trends in the tech industry *Task*: Understanding how tech leaders view the role of policy and regulation *Questions*: 1. Which episodes deal primarily with tech policy and government regulation? 2. How do guests perceive the impact of privacy laws on technology development? 3. Do any guests discuss the balance between innovation and ethical considerations? 4. What are the suggested changes to current policies mentioned by the guests? 5. Are collaborations between tech companies and governments discussed and how?
News articles	*User*: Educator incorporating current affairs into curricula *Task*: Teaching about health and wellness *Questions*: 1. What current topics in health can be integrated into health education curricula? 2. How do news articles address the concepts of preventive medicine and wellness? 3. Are there examples of health articles that contradict each other, and if so, why? 4. What insights can be gleaned about public health priorities based on news coverage? 5. How can educators use the dataset to highlight the importance of health literacy?

Table 1: Examples of potential users, tasks, and questions generated by the LLM based on short descriptions of the target datasets. Questions target global understanding rather than specific details.

3 Evaluation

3.1 Datasets

We selected two datasets in the one million token range, each equivalent to about 10 novels of text and representative of the kind of corpora that users may encounter in their real world activities:

- *Podcast transcripts*. Compiled transcripts of podcast conversations between Kevin Scott, Microsoft CTO, and other technology leaders (Behind the Tech, Scott, 2024). Size: 1669 × 600-token text chunks, with 100-token overlaps between chunks (∼1 million tokens).

- *News articles*. Benchmark dataset comprising news articles published from September 2013 to December 2023 in a range of categories, including entertainment, business, sports, technology, health, and science (MultiHop-RAG; Tang and Yang, 2024). Size: 3197 × 600-token text chunks, with 100-token overlaps between chunks (∼1.7 million tokens).

3.2 Queries

Many benchmark datasets for open-domain question answering exist, including HotPotQA (Yang et al., 2018), MultiHop-RAG (Tang and Yang, 2024), and MT-Bench (Zheng et al., 2024). However, the associated question sets target explicit fact retrieval rather than summarization for the purpose of data sensemaking, i.e., the process though which people inspect, engage with, and contextualize data within the broader scope of real-world activities (Koesten et al., 2021). Similarly, methods for extracting latent summarization queries from source texts also exist (Xu and Lapata, 2021), but such extracted questions can target details that betray prior knowledge of the texts.

To evaluate the effectiveness of RAG systems for more global sensemaking tasks, we need questions that convey only a high-level understanding of dataset contents, and not the details of specific texts. We used an activity-centered approach to automate the generation of such questions: given a short description of a dataset, we asked the LLM to identify N potential users and N tasks per user, then for each (user, task) combination, we asked the LLM to generate N questions that require understanding of the entire corpus. For our evaluation, a value of $N = 5$ resulted in 125 test questions per dataset. Table 1 shows example questions for each of the two evaluation datasets.

3.3 Conditions

We compare six different conditions in our analysis, including Graph RAG using four levels of graph communities (**C0**, **C1**, **C2**, **C3**), a text summarization method applying our map-reduce approach directly to source texts (**TS**), and a naïve "semantic search" RAG approach (**SS**):

- **CO**. Uses root-level community summaries (fewest in number) to answer user queries.
- **C1**. Uses high-level community summaries to answer queries. These are sub-communities of C0, if present, otherwise C0 communities projected down.
- **C2**. Uses intermediate-level community summaries to answer queries. These are sub-communities of C1, if present, otherwise C1 communities projected down.
- **C3**. Uses low-level community summaries (greatest in number) to answer queries. These are sub-communities of C2, if present, otherwise C2 communities projected down.
- **TS**. The same method as in subsection 2.6, except source texts (rather than community summaries) are shuffled and chunked for the map-reduce summarization stages.
- **SS**. An implementation of naïve RAG in which text chunks are retrieved and added to the available context window until the specified token limit is reached.

The size of the context window and the prompts used for answer generation are the same across all six conditions (except for minor modifications to reference styles to match the types of context information used). Conditions only differ in how the contents of the context window are created.

The graph index supporting conditions **C0**-**C3** was created using our generic prompts for entity and relationship extraction only, with entity types and few-shot examples tailored to the domain of the data. The graph indexing process used a context window size of 600 tokens with 1 gleaning for the Podcast dataset and 0 gleanings for the News dataset.

3.4 Metrics

LLMs have been shown to be good evaluators of natural language generation, achieving state-of-the-art or competitive results compared against human judgements (Wang et al., 2023a; Zheng et al., 2024). While this approach can generate reference-based metrics when gold standard answers are known, it is also capable of measuring the qualities of generated texts (e.g., fluency) in a reference-free style (Wang et al., 2023a) as well as in head-to-head comparison of competing outputs (LLM-as-a-judge, Zheng et al., 2024). LLMs have also shown promise at evaluating the performance of conventional RAG systems, automatically evaluating qualities like context relevance, faithfulness, and answer relevance (RAGAS, Es et al., 2023).

Given the multi-stage nature of our Graph RAG mechanism, the multiple conditions we wanted to compare, and the lack of gold standard answers to our activity-based sensemaking questions, we decided to adopt a head-to-head comparison approach using an LLM evaluator. We selected three target metrics capturing qualities that are desirable for sensemaking activities, as well as a control metric (directness) used as a indicator of validity. Since directness is effectively in opposition to comprehensiveness and diversity, we would not expect any method to win across all four metrics.

Our head-to-head measures computed using an LLM evaluator are as follows:

- *Comprehensiveness*. How much detail does the answer provide to cover all aspects and details of the question?
- *Diversity*. How varied and rich is the answer in providing different perspectives and insights on the question?
- *Empowerment*. How well does the answer help the reader understand and make informed judgements about the topic?
- *Directness*. How specifically and clearly does the answer address the question?

For our evaluation, the LLM is provided with the question, target metric, and a pair of answers, and asked to assess which answer is better according to the metric, as well as why. It returns the winner if one exists, otherwise a tie if they are fundamentally similar and the differences are negligible. To account for the stochasticity of LLMs, we run each comparison five times and use mean scores. Table 2 shows an example of LLM-generated assessment.

Question	Which public figures are repeatedly mentioned across various entertainment articles?
Graph RAG	**Overview of Prominent Public Figures in Entertainment** The entertainment industry is vast and diverse, encompassing film, television, music, sports, and digital media. Certain public figures stand out due to their significant contributions and influence across these sectors. The following summary highlights key individuals who are repeatedly mentioned in various entertainment articles, reflecting their impact and presence within the industry. **Actors and Directors** [...] **Public Figures in Controversy** [...] **Musicians and Executives** [...] **Athletes and Coaches** [...] **Influencers and Entrepreneurs** [...] The repeated mention of these figures in entertainment articles signifies their ongoing relevance and the public's interest in their work. Their influence spans across various aspects of entertainment, from shaping cultural narratives in film and television to driving trends in music and digital media. These individuals not only contribute to their respective fields but also influence the broader cultural landscape, often becoming central figures in social discussions and public discourse.
Naïve RAG	Public figures who are repeatedly mentioned across various entertainment articles include Taylor Swift, Travis Kelce, Britney Spears, and Justin Timberlake. These individuals have been highlighted for various reasons, ranging from their professional achievements to their personal lives. **Taylor Swift** [...] **Travis Kelce** [...] **Britney Spears** [...] **Justin Timberlake** [...] These figures are frequently covered due to their high-profile status and the public's interest in their careers and personal lives. Their activities, whether in music, sports, or personal relationships, have significant cultural and economic impacts, as evidenced by the media coverage and public reactions.
LLM Decision	**Comprehensiveness: Winner=1 (Graph RAG)** Answer 1 is better because it provides a more comprehensive and detailed list of public figures from a wider range of entertainment sectors, including film, television, music, sports, gaming, and digital media. It also includes specific examples of their contributions and the impact they have on their respective fields, as well as mentions of controversies and their implications. Answer 2, while detailed in its coverage of a few individuals, is limited to a smaller number of public figures and focuses primarily on their personal lives and relationships rather than a broad spectrum of their professional influence across the entertainment industry. **Diversity: Winner=1 (Graph RAG)** Answer 1 is better because it provides a more varied and rich response by covering a wide range of public figures from different sectors of the entertainment industry, including film, television, music, sports, gaming, and digital media. It offers insights into the contributions and influence of these figures, as well as controversies and their impact on public discourse. The answer also cites specific data sources for each mentioned figure, indicating a diverse range of evidence to support the claims. In contrast, Answer 2 focuses on a smaller group of public figures, primarily from the music industry and sports, and relies heavily on a single source for data, which makes it less diverse in perspectives and insights. **Empowerment: Winner=1 (Graph RAG)** Answer 1 is better because it provides a comprehensive and structured overview of public figures across various sectors of the entertainment industry, including film, television, music, sports, and digital media. It lists multiple individuals, providing specific examples of their contributions and the context in which they are mentioned in entertainment articles, along with references to data reports for each claim. This approach helps the reader understand the breadth of the topic and make informed judgments without being misled. In contrast, Answer 2 focuses on a smaller group of public figures and primarily discusses their personal lives and relationships, which may not provide as broad an understanding of the topic. While Answer 2 also cites sources, it does not match the depth and variety of Answer 1. **Directness: Winner=2 (Naïve RAG)** Answer 2 is better because it directly lists specific public figures who are repeatedly mentioned across various entertainment articles, such as Taylor Swift, Travis Kelce, Britney Spears, and Justin Timberlake, and provides concise explanations for their frequent mentions. Answer 1, while comprehensive, includes a lot of detailed information about various figures in different sectors of entertainment, which, while informative, does not directly answer the question with the same level of conciseness and specificity as Answer 2.

Table 2: Example question for the News article dataset, with generated answers from Graph RAG (C2) and Naïve RAG, as well as LLM-generated assessments.

Podcast transcripts

	SS	TS	C0	C1	C2	C3
SS	50	17	28	25	22	21
TS	83	50	50	48	43	44
C0	72	50	50	53	50	49
C1	75	52	47	50	52	50
C2	78	57	50	48	50	52
C3	79	56	51	50	48	50

Comprehensiveness

	SS	TS	C0	C1	C2	C3
SS	50	18	23	25	19	19
TS	82	50	50	50	43	46
C0	77	50	50	50	46	44
C1	75	50	50	50	44	45
C2	81	57	54	56	50	48
C3	81	54	56	55	52	50

Diversity

	SS	TS	C0	C1	C2	C3
SS	50	42	57	52	49	51
TS	58	50	59	55	52	51
C0	43	41	50	49	47	48
C1	48	45	51	50	49	50
C2	51	48	53	51	50	51
C3	49	49	52	50	49	50

Empowerment

	SS	TS	C0	C1	C2	C3
SS	50	56	65	60	60	60
TS	44	50	55	52	51	52
C0	35	45	50	47	48	48
C1	40	48	53	50	50	50
C2	40	49	52	50	50	50
C3	40	48	52	50	50	50

Directness

News articles

	SS	TS	C0	C1	C2	C3
SS	50	20	28	25	21	21
TS	80	50	44	41	38	36
C0	72	56	50	52	54	52
C1	75	59	48	50	58	55
C2	79	62	46	42	50	59
C3	79	64	48	45	41	50

Comprehensiveness

	SS	TS	C0	C1	C2	C3
SS	50	33	38	35	29	31
TS	67	50	53	45	44	40
C0	62	47	50	40	41	41
C1	65	55	60	50	50	50
C2	71	56	59	50	50	51
C3	69	60	59	50	49	50

Diversity

	SS	TS	C0	C1	C2	C3
SS	50	47	57	49	50	50
TS	53	50	58	50	50	48
C0	43	42	50	42	45	44
C1	51	50	58	50	52	51
C2	50	50	55	48	50	50
C3	50	52	56	49	50	50

Empowerment

	SS	TS	C0	C1	C2	C3
SS	50	54	59	55	55	54
TS	46	50	55	53	52	52
C0	41	45	50	48	48	47
C1	45	47	52	50	49	49
C2	45	48	52	51	50	49
C3	46	48	53	51	51	50

Directness

Figure 4: Head-to-head win rate percentages of (row condition) over (column condition) across two datasets, four metrics, and 125 questions per comparison (each repeated five times and averaged). The overall winner per dataset and metric is shown in bold. Self-win rates were not computed but are shown as the expected 50% for reference. All Graph RAG conditions outperformed naïve RAG on comprehensiveness and diversity. Conditions C1-C3 also showed slight improvements in answer comprehensiveness and diversity over TS (global text summarization without a graph index).

3.5 Configuration

The effect of context window size on any particular task is unclear, especially for models like `gpt-4-turbo` with a large context size of 128k tokens. Given the potential for information to be "lost in the middle" of longer contexts (Kuratov et al., 2024; Liu et al., 2023), we wanted to explore the effects of varying the context window size for our combinations of datasets, questions, and metrics. In particular, our goal was to determine the optimum context size for our baseline condition (**SS**) and then use this uniformly for all query-time LLM use. To that end, we tested four context window sizes: 8k, 16k, 32k and 64k. Surprisingly, the smallest context window size tested (8k) was universally better for all comparisons on comprehensiveness (average win rate of 58.1%), while performing comparably with larger context sizes on diversity (average win rate = 52.4%), and empowerment (average win rate = 51.3%). Given our preference for more comprehensive and diverse answers, we therefore used a fixed context window size of 8k tokens for the final evaluation.

3.6 Results

The indexing process resulted in a graph consisting of 8564 nodes and 20691 edges for the Podcast dataset, and a larger graph of 15754 nodes and 19520 edges for the News dataset. Table 3 shows the number of community summaries at different levels of each graph community hierarchy.

Global approaches vs. naïve RAG. As shown in Figure 4, global approaches consistently outperformed the naïve RAG (**SS**) approach in both comprehensiveness and diversity metrics across datasets. Specifically, global approaches achieved comprehensiveness win rates between 72-83% for Podcast transcripts and 72-80% for News articles, while diversity win rates ranged from 75-82% and 62-71% respectively. Our use of directness as a validity test also achieved the expected results, i.e., that naïve RAG produces the most direct responses across all comparisons.

9

	Podcast Transcripts					News Articles				
	C0	**C1**	**C2**	**C3**	**TS**	**C0**	**C1**	**C2**	**C3**	**TS**
Units	34	367	969	1310	1669	55	555	1797	2142	3197
Tokens	26657	225756	565720	746100	1014611	39770	352641	980898	1140266	1707694
% Max	2.6	22.2	55.8	73.5	100	2.3	20.7	57.4	66.8	100

Table 3: Number of context units (community summaries for **C0-C3** and text chunks for **TS**), corresponding token counts, and percentage of the maximum token count. Map-reduce summarization of source texts is the most resource-intensive approach requiring the highest number of context tokens. Root-level community summaries (**C0**) require dramatically fewer tokens per query (9x-43x).

Community summaries vs. source texts. When comparing community summaries to source texts using Graph RAG, community summaries generally provided a small but consistent improvement in answer comprehensiveness and diversity, except for root-level summaries. Intermediate-level summaries in the Podcast dataset and low-level community summaries in the News dataset achieved comprehensiveness win rates of 57% and 64%, respectively. Diversity win rates were 57% for Podcast intermediate-level summaries and 60% for News low-level community summaries. Table 3 also illustrates the scalability advantages of Graph RAG compared to source text summarization: for low-level community summaries (**C3**), Graph RAG required 26-33% fewer context tokens, while for root-level community summaries (**C0**), it required over 97% fewer tokens. For a modest drop in performance compared with other global methods, root-level Graph RAG offers a highly efficient method for the iterative question answering that characterizes sensemaking activity, while retaining advantages in comprehensiveness (72% win rate) and diversity (62% win rate) over naïve RAG.

Empowerment. Empowerment comparisons showed mixed results for both global approaches versus naïve RAG (**SS**) and Graph RAG approaches versus source text summarization (**TS**). Ad-hoc LLM use to analyze LLM reasoning for this measure indicated that the ability to provide specific examples, quotes, and citations was judged to be key to helping users reach an informed understanding. Tuning element extraction prompts may help to retain more of these details in the Graph RAG index.

4 Related Work

4.1 RAG Approaches and Systems

When using LLMs, RAG involves first retrieving relevant information from external data sources, then adding this information to the context window of the LLM along with the original query (Ram et al., 2023). Naïve RAG approaches (Gao et al., 2023) do this by converting documents to text, splitting text into chunks, and embedding these chunks into a vector space in which similar positions represent similar semantics. Queries are then embedded into the same vector space, with the text chunks of the nearest k vectors used as context. More advanced variations exist, but all solve the problem of what to do when an external dataset of interest exceeds the LLM's context window.

Advanced RAG systems include pre-retrieval, retrieval, post-retrieval strategies designed to overcome the drawbacks of Naïve RAG, while Modular RAG systems include patterns for iterative and dynamic cycles of interleaved retrieval and generation (Gao et al., 2023). Our implementation of Graph RAG incorporates multiple concepts related to other systems. For example, our community summaries are a kind of self-memory (Selfmem, Cheng et al., 2024) for generation-augmented retrieval (GAR, Mao et al., 2020) that facilitates future generation cycles, while our parallel generation of community answers from these summaries is a kind of iterative (Iter-RetGen, Shao et al., 2023) or federated (FeB4RAG, Wang et al., 2024) retrieval-generation strategy. Other systems have also combined these concepts for multi-document summarization (CAiRE-COVID, Su et al., 2020) and multi-hop question answering (ITRG, Feng et al., 2023; IR-CoT, Trivedi et al., 2022; DSP, Khattab et al., 2022). Our use of a hierarchical index and summarization also bears resemblance to further approaches, such as generating a hierarchical index of text chunks by clustering the vectors of text embeddings (RAPTOR, Sarthi et al., 2024) or generating a "tree of clarifications" to answer multiple interpretations of ambiguous questions (Kim et al., 2023). However, none of these iterative or hierarchical approaches use the kind of self-generated graph index that enables Graph RAG.

4.2 Graphs and LLMs

Use of graphs in connection with LLMs and RAG is a developing research area, with multiple directions already established. These include using LLMs for knowledge graph creation (Trajanoska et al., 2023) and completion (Yao et al., 2023), as well as for the extraction of causal graphs (Ban et al., 2023; Zhang et al., 2024) from source texts. They also include forms of advanced RAG (Gao et al., 2023) where the index is a knowledge graph (KAPING, Baek et al., 2023), where subsets of the graph structure (G-Retriever, He et al., 2024) or derived graph metrics (GraphToolFormer, Zhang, 2023) are the objects of enquiry, where narrative outputs are strongly grounded in the facts of retrieved subgraphs (SURGE, Kang et al., 2023), where retrieved event-plot subgraphs are serialized using narrative templates (FABULA, Ranade and Joshi, 2023), and where the system supports both creation and traversal of text-relationship graphs for multi-hop question answering (Wang et al., 2023b). In terms of open-source software, a variety a graph databases are supported by both the LangChain (LangChain, 2024) and LlamaIndex (LlamaIndex, 2024) libraries, while a more general class of graph-based RAG applications is also emerging, including systems that can create and reason over knowledge graphs in both Neo4J (NaLLM, Neo4J, 2024) and NebulaGraph (GraphRAG, NebulaGraph, 2024) formats. Unlike our Graph RAG approach, however, none of these systems use the natural modularity of graphs to partition data for global summarization.

5 Discussion

Limitations of evaluation approach. Our evaluation to date has only examined a certain class of sensemaking questions for two corpora in the region of 1 million tokens. More work is needed to understand how performance varies across different ranges of question types, data types, and dataset sizes, as well as to validate our sensemaking questions and target metrics with end users. Comparison of fabrication rates, e.g., using approaches like SelfCheckGPT (Manakul et al., 2023), would also improve on the current analysis.

Trade-offs of building a graph index. We consistently observed Graph RAG achieve the best head-to-head results against other methods, but in many cases the graph-free approach to global summarization of source texts performed competitively. The real-world decision about whether to invest in building a graph index depends on multiple factors, including the compute budget, expected number of lifetime queries per dataset, and value obtained from other aspects of the graph index (including the generic community summaries and the use of other graph-related RAG approaches).

Future work. The graph index, rich text annotations, and hierarchical community structure supporting the current Graph RAG approach offer many possibilities for refinement and adaptation. This includes RAG approaches that operate in a more local manner, via embedding-based matching of user queries and graph annotations, as well as the possibility of hybrid RAG schemes that combine embedding-based matching against community reports before employing our map-reduce summarization mechanisms. This "roll-up" operation could also be extended across more levels of the community hierarchy, as well as implemented as a more exploratory "drill down" mechanism that follows the information scent contained in higher-level community summaries.

6 Conclusion

We have presented a global approach to Graph RAG, combining knowledge graph generation, retrieval-augmented generation (RAG), and query-focused summarization (QFS) to support human sensemaking over entire text corpora. Initial evaluations show substantial improvements over a naïve RAG baseline for both the comprehensiveness and diversity of answers, as well as favorable comparisons to a global but graph-free approach using map-reduce source text summarization. For situations requiring many global queries over the same dataset, summaries of root-level communities in the entity-based graph index provide a data index that is both superior to naïve RAG and achieves competitive performance to other global methods at a fraction of the token cost.

An open-source, Python-based implementation of both global and local Graph RAG approaches is forthcoming at `https://aka.ms/graphrag`.

Acknowledgements

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. (2023). Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Baek, J., Aji, A. F., and Saffari, A. (2023). Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Ban, T., Chen, L., Wang, X., and Chen, H. (2023). From query tools to causal architects: Harnessing large language models for advanced causal discovery from data.

Baumel, T., Eyal, M., and Elhadad, M. (2018). Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Cheng, X., Luo, D., Chen, X., Liu, L., Zhao, D., and Yan, R. (2024). Lift yourself up: Retrieval-augmented text generation with self-memory. *Advances in Neural Information Processing Systems*, 36.

Dang, H. T. (2006). Duc 2005: Evaluation of question-focused summarization systems. In *Proceedings of the Workshop on Task-Focused Summarization and Question Answering*, pages 48–55.

Es, S., James, J., Espinosa-Anke, L., and Schockaert, S. (2023). Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*.

Feng, Z., Feng, X., Zhao, D., Yang, M., and Qin, B. (2023). Retrieval-generation synergy augmented large language models. *arXiv preprint arXiv:2310.05149*.

Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3-5):75–174.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., and Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Goodwin, T. R., Savery, M. E., and Demner-Fushman, D. (2020). Flight of the pegasus? comparing transformers on few-shot and zero-shot multi-document abstractive summarization. In *Proceedings of COLING. International Conference on Computational Linguistics*, volume 2020, page 5640. NIH Public Access.

He, X., Tian, Y., Sun, Y., Chawla, N. V., Laurent, T., LeCun, Y., Bresson, X., and Hooi, B. (2024). G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Jacomy, M., Venturini, T., Heymann, S., and Bastian, M. (2014). Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLoS ONE 9(6): e98679. https://doi.org/10.1371/journal.pone.0098679.*

Jin, D., Yu, Z., Jiao, P., Pan, S., He, D., Wu, J., Philip, S. Y., and Zhang, W. (2021). A survey of community detection approaches: From statistical modeling to deep learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):1149–1170.

Kang, M., Kwak, J. M., Baek, J., and Hwang, S. J. (2023). Knowledge graph-augmented language models for knowledge-grounded dialogue generation. *arXiv preprint arXiv:2305.18846*.

Khattab, O., Santhanam, K., Li, X. L., Hall, D., Liang, P., Potts, C., and Zaharia, M. (2022). Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*.

Kim, G., Kim, S., Jeon, B., Park, J., and Kang, J. (2023). Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. *arXiv preprint arXiv:2310.14696*.

Klein, G., Moon, B., and Hoffman, R. R. (2006a). Making sense of sensemaking 1: Alternative perspectives. *IEEE intelligent systems*, 21(4):70–73.

Klein, G., Moon, B., and Hoffman, R. R. (2006b). Making sense of sensemaking 2: A macrocognitive model. *IEEE Intelligent systems*, 21(5):88–92.

Koesten, L., Gregory, K., Groth, P., and Simperl, E. (2021). Talking datasets–understanding data sensemaking behaviours. *International journal of human-computer studies*, 146:102562.

Kuratov, Y., Bulatov, A., Anokhin, P., Sorokin, D., Sorokin, A., and Burtsev, M. (2024). In search of needles in a 11m haystack: Recurrent memory finds what llms miss.

LangChain (2024). Langchain graphs. https://python.langchain.com/docs/use_cases/graph/.

Laskar, M. T. R., Hoque, E., and Huang, J. (2020). Query focused abstractive summarization via incorporating query relevance and transfer learning with transformer models. In *Advances in Artificial Intelligence: 33rd Canadian Conference on Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, May 13–15, 2020, Proceedings 33*, pages 342–348. Springer.

Laskar, M. T. R., Hoque, E., and Huang, J. X. (2022). Domain adaptation with pre-trained transformers for query-focused abstractive text summarization. *Computational Linguistics*, 48(2):279–320.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. (2023). Lost in the middle: How language models use long contexts. arXiv:2307.03172.

Liu, Y. and Lapata, M. (2019). Hierarchical transformers for multi-document summarization. *arXiv preprint arXiv:1905.13164*.

LlamaIndex (2024). LlamaIndex Knowledge Graph Index. https://docs.llamaindex.ai/en/stable/examples/index_structs/knowledge_graph/KnowledgeGraphDemo.html.

Manakul, P., Liusie, A., and Gales, M. J. (2023). Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*.

Mao, Y., He, P., Liu, X., Shen, Y., Gao, J., Han, J., and Chen, W. (2020). Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*.

Martin, S., Brown, W. M., Klavans, R., and Boyack, K. (2011). Openord: An open-source toolbox for large graph layout. *SPIE Conference on Visualization and Data Analysis (VDA)*.

Microsoft (2023). The impact of large language models on scientific discovery: a preliminary study using gpt-4.

NebulaGraph (2024). Nebulagraph launches industry-first graph rag: Retrieval-augmented generation with llm based on knowledge graphs. https://www.nebula-graph.io/posts/graph-RAG.

Neo4J (2024). Project NaLLM. https://github.com/neo4j/NaLLM.

Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582.

Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., and Shoham, Y. (2023). In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Ranade, P. and Joshi, A. (2023). Fabula: Intelligence report generation using retrieval-augmented narrative construction. *arXiv preprint arXiv:2310.13848*.

Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., and Manning, C. D. (2024). Raptor: Recursive abstractive processing for tree-organized retrieval. *arXiv preprint arXiv:2401.18059*.

Scott, K. (2024). Behind the Tech. https://www.microsoft.com/en-us/behind-the-tech.

Shao, Z., Gong, Y., Shen, Y., Huang, M., Duan, N., and Chen, W. (2023). Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Su, D., Xu, Y., Yu, T., Siddique, F. B., Barezi, E. J., and Fung, P. (2020). Caire-covid: A question answering and query-focused multi-document summarization system for covid-19 scholarly information management. *arXiv preprint arXiv:2005.03975*.

Tang, Y. and Yang, Y. (2024). MultiHop-RAG: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391*.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Traag, V. A., Waltman, L., and Van Eck, N. J. (2019). From Louvain to Leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1).

Trajanoska, M., Stojanov, R., and Trajanov, D. (2023). Enhancing knowledge graph construction using large language models. *ArXiv*, abs/2305.04676.

Trivedi, H., Balasubramanian, N., Khot, T., and Sabharwal, A. (2022). Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

Wang, J., Liang, Y., Meng, F., Sun, Z., Shi, H., Li, Z., Xu, J., Qu, J., and Zhou, J. (2023a). Is chatgpt a good nlg evaluator? a preliminary study. *arXiv preprint arXiv:2303.04048*.

Wang, S., Khramtsova, E., Zhuang, S., and Zuccon, G. (2024). Feb4rag: Evaluating federated search in the context of retrieval augmented generation. *arXiv preprint arXiv:2402.11891*.

Wang, Y., Lipka, N., Rossi, R. A., Siu, A., Zhang, R., and Derr, T. (2023b). Knowledge graph prompting for multi-document question answering.

Xu, Y. and Lapata, M. (2021). Text summarization with latent queries. *arXiv preprint arXiv:2106.00104*.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., and Manning, C. D. (2018). HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yao, J.-g., Wan, X., and Xiao, J. (2017). Recent advances in document summarization. *Knowledge and Information Systems*, 53:297–336.

Yao, L., Peng, J., Mao, C., and Luo, Y. (2023). Exploring large language models for knowledge graph completion.

Zhang, J. (2023). Graph-toolformer: To empower llms with graph reasoning ability via prompt augmented by chatgpt. *arXiv preprint arXiv:2304.11116*.

Zhang, Y., Zhang, Y., Gan, Y., Yao, L., and Wang, C. (2024). Causal graph discovery with retrieval-augmented generation based large language models. *arXiv preprint arXiv:2402.15301*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. (2024). Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.