IJHSR
ijhighschoolresearch.org
RESEARCH ARTICLE

Empowering Large Language Model Reasoning : Hybridizing Layered Retrieval Augmented Generation and Knowledge Graph Synthesis

Vedanth Aggarwal

GEMS Modern Academy; Nad Al Sheba - Nad Al Sheba 3 - Dubai, United Arab Emirates; vedanth.aggarwal@gmail.com

ABSTRACT: Retrieval Augmented Generation has improved LLM question answering significantly. However, this mechanism still produces hallucinations and structural incoherence in knowledge-intensive tasks. Additionally, many existing techniques neither holistically leverage multiple properties of text nor integrate diverse prompting and agenting frameworks. To address these limitations, this paper proposes a novel methodology that extracts and utilizes unstructured and structured properties of text to construct layered RAG pipelines designed to enhance complex LLM reasoning. Our approach synthesizes three distinct RAG methodologies, each specialized in various aspects: textual entity knowledge graph extraction (Textual Entity RAG); community summary and entity generation (Microsoft GraphRAG), and structural link navigation (MetaWiki RAG). By cumulatively layering these techniques along with advanced prompting and agentic evaluation, we aim to capture a more comprehensive context, enabling the model to generate well-structured responses that reflect all relevant attributes of the text. The proposed framework not only enhances existing RAG mechanisms but also demonstrates the effective integration of knowledge graphs. Additionally, it showcases the application of this framework to advanced answer generation using Wikipedia, with extensions to similar knowledge networks. This novel approach offers a robust solution for social recommender systems and other practical applications, delivering holistic outcomes by synthesizing diverse RAG techniques.

KEYWORDS: Robotics and Intelligent Machines, Machine Learning, Artificial Intelligence, Retrieval Augmented Generation; Knowledge Graphs, LLM Reasoning.

Introduction

The field of Natural Language Processing (NLP) has undergone a revolutionary shift with the arrival of large language models (LLMs), which demonstrate a remarkable ability to generate, interpret and engage with human language. They have been trained on vast amounts of text corpus, which allows them to capture inherent patterns of text such as syntax and semantics. Owing to this progress, LLM reasoning has emerged as a key focus area for recent research.[1] This field aims to explore and improve the ability of models to draw logical inferences and solve complex, multi-step problems in analytical tasks. Effective reasoning demands LLMs to leverage contextual understanding, integrate diverse pieces of information, and follow structured thought processes. However, when integrating large source documents or text corpus, LLMs face challenges like logical inconsistency, generation of contradictory information, and difficulty in tracking long passages of text.[2] Thus, they are unable to perform well in knowledge-intensive applications. In a nutshell, enhancing LLM reasoning is an important direction the global research community is now moving towards and our paper aims to address how to build better techniques that leverage graph context to achieve this.

In response to this, Retrieval Augmented Generation (RAG) (Figure 1) has emerged as a key NLP tool. These models access relevant information from external knowledge bases or document repositories, enabling more accurate and contextually appropriate responses.[3] RAG pipelines achieve this by indexing large datasets, embedding documents into vector spaces, and using similarity search to dynamically retrieve most relevant content to the question. Thus, they ground outputs in factual content in tasks such as question answering, summarization, and content creation. Furthermore, empirical studies show nearly 30-50% gains in metrics like exact match (EM) and F1 scores on datasets such as Natural Questions and TriviaQA when using RAG.[4] As the field progresses, modern research is focusing on optimizing retrieval algorithms, indexing techniques, and query generation pipelines - all aimed at question answering quality enhancement.

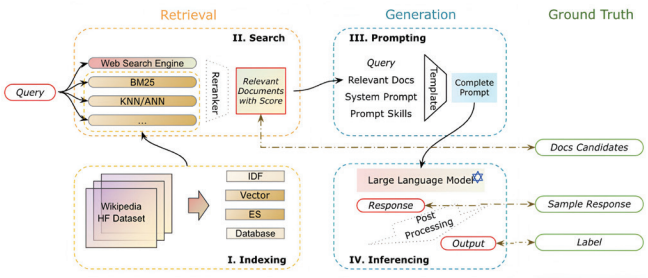


Figure 1: This is a retrieval augmented generation pipeline demonstrating stages of document indexing, vector embeddings, similarity retrieval, querying and generation followed at a high-level in majority of RAG pipelines.[2]

As the next step of LLM evolution, the use of unstructured information for context in LLMs has become increasingly evident. While RAG provides an expansive breadth of infor-

DOI: 10.36838/v6i12.11

mation that forms the base of material generated, structured graphs (Figure 2) provide a framework for coherent reasoning that improves chain of thought, context arrangement and idea integration.[5] Structured data augmented RAG thus provides a more concise relational representation of the most important concepts in a text. This recent evolution in LLM information processing has fostered an emerging interest in the integration of knowledge graphs - networks where nodes represent entities, and edges capture relationships. Extracting relationships from text allows reasoning over interconnected ideas and provides a framework to link and structure essential components of domain specific information.[6] In fact, studies using the COVID-19 Open Research Dataset (CORD-19) for knowledge graph-based reasoning reported an 18% improvement in precision and reduction in irrelevant answers.[7,8]

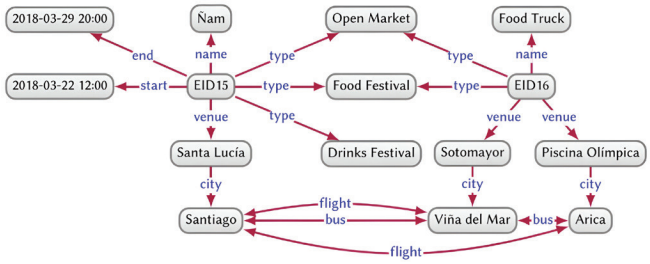


Figure 2: This network is a knowledge graph based on flight and country data. They represent nodes as entities and relationships as edges such as vehicle type. Nodes can be heterogeneous and have multiple descriptive attributes.[3]

In addition, to structure all these pipelines together, LLMs require sophisticated prompting and agent-based methodologies.[9] Frameworks such as chain of thought, few shot reasoning have helped improve LLM reasoning thought-process and improve the utilization of every aspect of contextual information, specifically when prompts are lengthy and require specific instructions.[10,11] On the other hand, self-evaluation and decision making agentic frameworks have primarily aided LLMs to evaluate their answers and decide approaches to multi-step queries.[12] This cycle of thinking ensures final responses are well revised and well throughout.

Despite all this progress, modern retrieval models face several limitations. Firstly, structural relevance. The unorganized integration of document chunks can lead to unsystematic arrangement of ideas in terms of timeline and logical flow of thought. Secondly, the widespread occurrence of hallucinations, which is phenomenon wherein models make major logical leaps and false conclusions that are not factually supported.[9] Thirdly, conventional RAG fails at high level questions that require a global overview of the database as they tend to rely on localized chunks as opposed to interconnected data.[13] Finally, most retrievals are unable to extract inherent connections in text i.e identify key relationships between major entities organized in information networks. These issues underscore the need for a robust mechanism that leverages all properties of text to achieve significant advancement in these issues.

To tackle this, our research introduces a novel approach that integrates Textual Entity RAG, Microsoft GraphRAG, and MetaWiki GraphRAG techniques applied on the Wikipedia dataset. Through our novel framework and empirical analysis we present a structured approach focusing on three critical problem statements:

1. How to efficiently leverage, automatically construct, extract, and traverse knowledge graphs to enhance information retrieval and integration? To maximize the exploitation of graph information for RAG structures, we combine advanced graph creation frameworks with hybrid RAG schemas. This is achieved by semantic chunking of text paired with graph embeddings to create a multi-dimensional schema for traversing subgraphs and extracting nodes. This ensures that all textually relevant nodes and edges are returned, allowing the RAG model to generate contextually supported and logically consistent results. These pipelines are layered with prompting and agenting to create an assimilated system.

2. How to improve reasoning, coherence, and reduce hallucinations of LLMs in Retrieval-Augmented Generation? Enabling logical consistency and hallucination suppression in LLM-equipped RAG systems requires a two-pronged solution that combines state-of-the-art retrieval mechanisms with structure enhancement graph techniques.To tackle this, our framework carefully builds and structures the input context to guide the LLM to closely follow factual information and avoid large deviations which may result in hallucinations. In short, we improve the structured thought process of this model and improve information context at various levels during retrieval and generation, so the model can reason with logical analysis.

3. How to leverage the underlying Wikipedia network structure for global contextual understanding? Utilizing the interconnected hyperlinked nature of Wikipedia's network for RAG systematically exploits high level graph properties to enable large-scale reasoning. We build a knowledge graph based on the dense article connectivity of this Wikipedia graph. This is enhanced with multi-level node attributes to provide additional context on each Wikipedia article. Then, we use sophisticated link analysis algorithms to discover important nodes as well as contextually coherent subgraphs. Overall, the augmented graph structure provides a comprehensive high level context.

Related Work:

The body of related work pertinent to our research can be categorized into four principal domains. This paper utilizes various existing techniques in prompting and agenting; enhances and explores diverse RAG pipelines; leveraged knowledge graph concepts to develop retrieval methods; and overall aims to test LLM reasoning on the improved context by amalgamating all fundamental concepts from these domains.

- ***Retrieval Augmented Generation:***

Retrieval Augmented Generation has emerged as an initial solution to address Large Language Model challenges such as hallucination, outdated knowledge, and non-transparent reasoning processes by incorporating knowledge from external databases.[2,3] The success of RAG lies in its ability to compensate for the inherent limitations of closed LLMs, such as

outdated information and the lack of access to real-time data. It has to an extent enhanced generation accuracy and credibility in knowledge-intensive tasks and synergistically merged with LLMs. However, retrieval operations often rely on traditional document similarity metrics, which are insufficient when dealing with complex interdependencies. The real challenge is not just retrieving documents but retrieving the right knowledge for nuanced queries. Furthermore, RAG pipelines frequently struggle with query-focused summarization (QFS) tasks that demand nuanced global contextualization beyond basic retrieval.[13] Therefore, to solve this there is a growing interest in knowledge grounded generation.[8] Another significant research gap is developing dynamic retrieval mechanisms in cases like scientific article networks, recommender systems, and knowledge graphs - these aim to build intricate pipeline for processing domain specific knowledge.[14] Our paper aims to leverage existing RAG pipelines and enhance them to address existing challenges in LLM reasoning. The paper synthesizes diverse RAG methodologies and creates multi-layered retrieval pipelines integrating various retrieval elements.

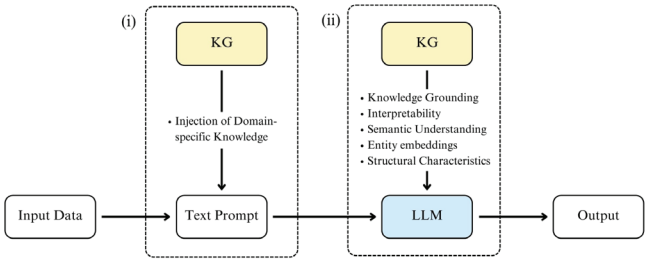


Figure 3: This pipeline demonstrates how knowledge graph extraction can potentially be applied to integrate domain specific knowledge and node embeddings for improving text prompts and LLM output generation. Additionally, it shows the presence of structural, semantic and knowledge grounding.[15]

- ***LLM Reasoning in context of graph structured data:***

The rise of LLMs having massive context-aware knowledge and semantic comprehension capabilities has made significant advancement in QA generation and human-centric tasks; models like GPT-4 and BERT have demonstrated exceptional performance in general NLP tasks and become indispensable to artificial general intelligence.[15] However, in domain-intensive tasks, they produce hallucinations due to their reliance on unstructured text. Their key weakness is weak precise relational reasoning and their inability to interpret graph structures into meaningful text representations, producing incorrect or unsupported information in 21-35% of responses in such cases. Thus, knowledge graphs and graph structured data have emerged as promising solutions; however, application of LLMs to rich structural/relational/textual graphs is still relatively less explored and faces challenges on accessing and precisely manipulating knowledge, encoding graphs to language, and knowledge formatting.[16,17] A promising field to address this is augmenting LLMs with graph embeddings or graph-to-text generation tools. These include developing a framework where KGs directly influence generation through graph-enhanced prompting strategies. Our paper serves to evaluate LLM reasoning on graph structured data through various metrics and provide frameworks to encode graphs for maximum efficiency in LLMs answer generation.

- ***Knowledge Graphs:***

In real-world scenarios, text data is associated with rich structure graph information (e.g., academic networks, scientific research, finance, e-commerce networks) or graph data is paired with textual information (e.g., molecules with descriptions). They are a powerful tool for representing and analyzing complex relationships in applications such as social networks, recommender systems, and computational finance.[18] Hence, reasoning on graphs (Figure 3) is essential for drawing inferences about the relationships between entities in a complex system, identifying hidden patterns/trends and avoiding hallucinations by leveraging explicitly network representations.[19,20] The 2 relevant frameworks for our paper are LLM-augmented KGs that leverage LLMs for different KG (Knowledge Graph) tasks such as embedding, completion, construction, graph-to-text generation, and question answering; Synergized LLMs + KGs, in which LLMs and KGs play mutually beneficial roles for bidirectional reasoning.[21] Having said that, most techniques restrict KGs to tasks like entity linking or knowledge embedding, often missing the opportunity to exploit their full potential for relational reasoning and multi-hop inference. In fact, fewer than 10% of studies actively apply LLMs to structural graph tasks.[22] To improve this, our paper specifically focuses on Wikipedia, which is a rich network of human knowledge connected by hyperlinks.[23] Furthermore, one potential mechanism we are utilizing is the bidirectional interaction framework allowing continuous knowledge exchange between the KG and LLM throughout the generation process.

- ***Role of Prompting & Agenting:***

Our paper uses prompt engineering to formulate and structure context to improve LLM understanding. Chain of Thought: By providing intermediate reasoning steps as guidelines, performance improves on arithmetic, common sense and symbolic reasoning. This is especially significant as COT provides examples of graph data formats and graph reasoning methodologies which the LLM can use as guidance during reasoning processes.[24] Contrastive Chain of Thought : This improves CoT by providing valid and invalid reasoning demonstrations to explain mistakes and erroneous logic - thus preventing LLMs from making incongruous connections from graph data and RAG chunks.[25] Graph of Thought is another mechanism we are testing to see its comparative performance against other techniques : GoT models information as a graph, where units of "LLM thoughts" are vertices, and edges are dependencies representing networks of thoughts. Other methodologies include few-shot prompting, zero-shot prompting, directional-stimulus prompting.[26,27] Secondly, Agenting is a crucial tool employed in our research to integrate several RAG pipelines and evaluate answers. ReACT : A decision making LLM framework demonstrates improved interpretability and trustworthiness over baselines like HotpotQA and Fever (Fact verification). It reduces hallucinations and error propagation

DOI: 10.36838/v6i12.11

by generating human-like task-solving trajectories, which are extremely beneficial for hybridized multi-step retrieval.[28] Reflexion : This framework converts binary/scalar feedback into verbal textual summary feedback. This self-reflective feedback is crucial to our paper as it acts as a 'semantic' gradient signal for future episodes, which allows LLMs to enhance their final responses with maximum contextual and knowledge graph understanding.[29]

Preliminaries:

This section gives a high level overview of the dataset and key terminologies that illuminate the fundamental aspects of our methodology.

Dataset: Our dataset belongs to SNAP, Stanford Network Analysis Project, datasets. These datasets are employed for their extensive network data, critical for analyzing graph-based retrieval and extracting knowledge graphs within RAG frameworks.[30] Specifically, we are using the Wikispeedia Path Datasets - The dataset was specifically chosen as our research aims to target the Wikipedia knowledge graph to leverage its rich article network and properties. Although our research isn't centered around LLM wikispeedia gameplay, this dataset provides useful data like path sequences (~76000), plaintext articles, article names (~4600) and hyperlink connections (~119000).[31] Additionally, the text articles form the source corpus for all the baseline retrieval. Alternatively the Wikipedia Article Network dataset, focused on community detection and network visualization, can also satisfy base demands of this research.

These following are the primary components of the database: wikispeedia_plaintext_articles - raw wikipedia articles with entire page content, forming the text corpus for our vector databases and RAG pipeline; articles.tsv - list of all article names, which are used as nodes for our graphs including dead end and orphaned nodes as well; categories.tsv - hierarchical list of categories and themes associated with each node, which are beneficial for community detection and thematically linking nodes; links.tsv - list of all hyperlink connections between one article to another (to create edges); paths_finished.tsv - utilized in sub-experiment testing ability of LLM to navigate wikipedia graph.

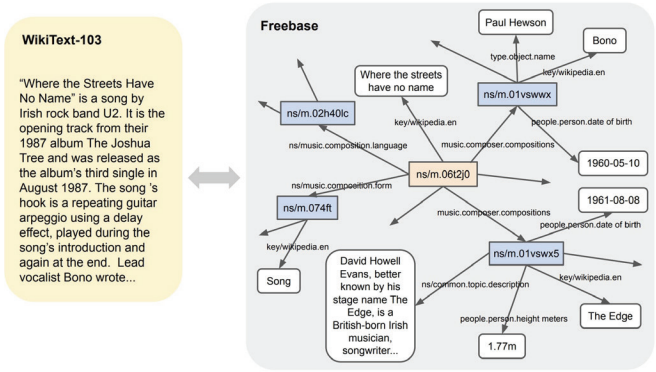


Figure 4: This illustration is a sample representation of knowledge graph integration for Wikipedia articles from the WikiGraphs dataset. One key aspect of this mechanism is that each article features a network of key textual elements, such as names, dates, and numerical figures, connected by structured hierarchical category edges.[32]

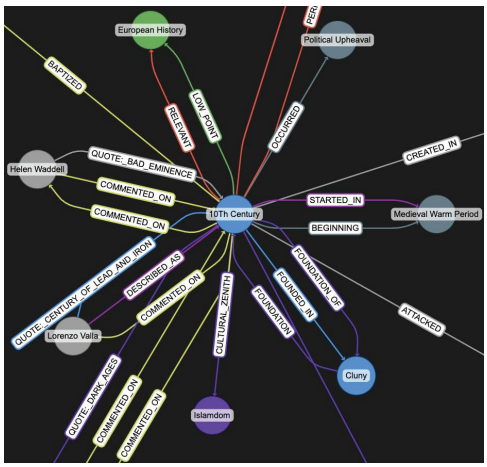
Retrieval-Augmented Generation Pipeline: This system combines retrieval methods to generate outputs that are relevant to the context. Firsty, it splits down documents into smaller, easier-to-handle text chunks.[33] After this, the sections are vectorized using embedding techniques (Word2Vec, Transformer models,etc.), which change text into numerical representations that capture word meanings. A vector database is set up to perform approximate nearest neighbor (ANN) searches (k-d trees, HNSW graphs, etc.), which quickly compare how similar a query is to stored documents. The last part of the pipeline is generating content, wherein relevant text documents are retrieved using search algorithms and formatted with the assistance of LLMs to generate the final output that directly aligns with source information only.

Knowledge Graphs: Knowledge graphs are complex systems that represent entities and their interrelationships in a graph structure with nodes and edges (Figure 4). In Retrieval-Augmented Generation (RAG), textual knowledge graphs use NLP techniques like Named Entity Recognition (NER) and relation extraction (finding connections between ideas).[34] Alternatively, knowledge graphs can be high level networks depicting the linkage between metadata and broader structures. Potential operations include subgraph extraction based on query relevance, node ranking to prioritize important entities, and relationship traversal. Several graph exploration methods like breadth-first or depth-first can also search efficiently examine the graph and extract a detailed subgraph rich in context.[35,36] These operations leverage a rich, interconnected context, enhancing the accuracy and depth of the generated responses.[37]

Methods

This section details the methodology behind our integrated framework of retrieval techniques. We will systematically break down the procedures behind the three major techniques: Textual Entity Graph RAG, Microsoft GraphRAG, and MetaWiki RAG. Each technique leverages unique approaches to integrate graph-based structures, advanced language models, and meta-information, aiming to optimize the accuracy and relevance of retrieval and generation processes.

Textual Entity Graph RAG:



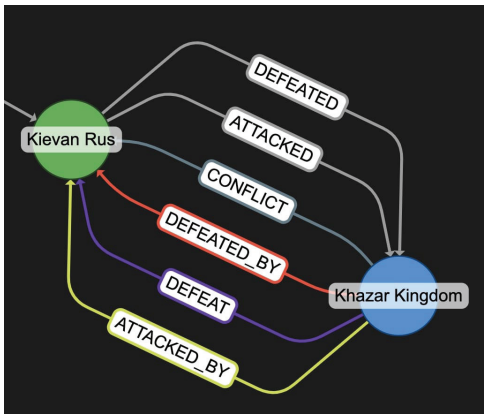


Figure 5: Knowledge graph extracted from Wikipedia article using a large language model graph transformer. Nodes represent all important events, names, organizations, etc. and bidirectional edges represent direct and indirect relationships mentioned in the text. The graph structure also demonstrates thematic similarity and connections at micro and macro level.

The Textual Entity Graph Retrieval-Augmented Generation (RAG) architecture starts with loading the source documents from Wikipedia articles dataset. It uses a new powerful language model based graph transformer to subsequently read and group key entities, along with all the connections between them from unstructured text data. The process starts by parsing the source documents for which NLP (Natural Language Processing) techniques like Named Entity Recognition (NER) and relationship extraction algorithms are used.[34] They are methods by which you can identify entities (e.g., person, organization and event) as well establish linguistic relationships between them: for example, "affiliation with" or "born in". The entities and relationships are then joined together to form a complete graph (Figure 5), where the nodes represent different things/entity classes and edges define their connections.[8] Additionally, each node has an attribute of entity type (Ex: person, country, organization, etc.). After constructing the graph, a cutting-edge vector database[38] of the source document chunks is created and integrated within the graph. This is hosted on a graph visualization platform like Neo4J, which is used to extract and perform search queries on the interconnected nodes and verify all information.[39]

Textual Entity Graph RAG has a complex, multi-stage architecture for processing user queries through the retrieval pipeline. First, it extracts important entities from the query by pruning nonessential words to extract a subset of essential keywords. A query language for graphs is then used (like Cypher or SPARQL) to run queries that are expressed as path patterns where source and target nodes contain the entities gleaned from text data, enabling complex multi-hop extraction (nodes which are more than 1 link apart) of all relevant relationships within the knowledge-graph.[40] Through this, we can extract tuples of all graph paths where the relevant entities are present. To improve precision of the search results, we combine keyword-based retrieval with a vector-based similarity search in a hybrid RAG fashion. Such a dual approach helps in not only returning documents that are semantically similar to the user query but also include important keywords.[41] The final step actualizes chain-of-thought prompting to help the model use the structure of graph data in order to generate exact and contextually relevant responses. Specifically, through prompting we format the knowledge graph in SOURCE - RELATIONSHIP - TARGET pairs and explain how to utilize these structural connections to strengthen the answer.[42]

Example: Take a Wikipedia article on World War I. It is likely to have a sentence like "At the end of World War 1, countries like the United Kingdom signed the historic Treaty of Versailles in 1919". After parsing the text, key entities include : "World War 1" (event), "United Kingdom" (country), "1919" (date), and "Treaty of Versailles" (thing). These become the nodes and their type is the attribute. Then using relationship extraction, we can find the following connections: "United Kingdom" → "country in" → "World War I", "Treaty of Versailles" → "date" → "1919", "United Kingdom" → "signed" → "Treaty of Versailles". These relations become edges. Thus, in this graph the United Kingdom is a node connected to the Treaty of Versailles which is then connected to the year 1919. Finally, for a sample question like "What year was the treaty signed?", we extract the paths with entities similar to "treaty" (including its neighbours) and relations related to time and dates. Using this, the edge "Treaty of Versailles" → "date" → "1919" is retrieved for LLM context. After combining this with the retrieved text sentences from the source document, the LLM formats this into the final answer "The Treaty of Versailles was signed in 1919."

Microsoft Graph RAG:

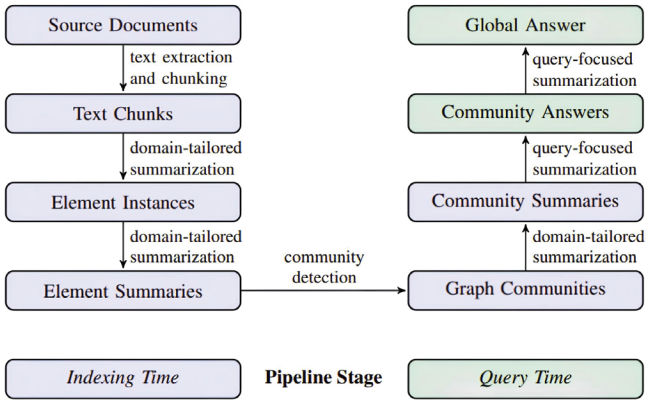


Figure 6: Microsoft graph RAG indexing and query time pipeline. The indexing includes source chunking and domain tailored summarization. The query phase involves graph community detection, domain tailored summarization and query focused summarization.[10]

This open-source GraphRAG framework, developed by Microsoft, can be used to generate a complete knowledge graph from given text documents by using LLMs. This framework uses a series of LLM calls to identify complex entities such as people, organizations, and events along with their relationships.[15] The framework formats this in various tuples and structural sequences of information pairs. Another key feature is the "bottom-up clustering", which groups data hierarchically into semantic clusters or communities, which represent major themes. For example, concepts like machine learning, neural networks, and artificial intelligence will be grouped into one community.

DOI: 10.36838/v6i12.11

Community-based summary generation (Figure 6) is a crucial technique as it captures essential graph information of each community of nodes in one holistic summary. The community summaries are hierarchically ordered from broader to more narrow communities. When a question is asked, the system searches for relevant concepts and the community summaries.[43] It generates partial answers from each summaries and combines all of these partial generations into one holistic answer in a mechanism called query-focused summarization.[13] Microsoft graph RAG also uses the dual search strategy that enables global search, which provides general, broader and high level information like themes, and local searches, which focus on specific entities and particular relationships such as character traits.

MetaWiki RAG:

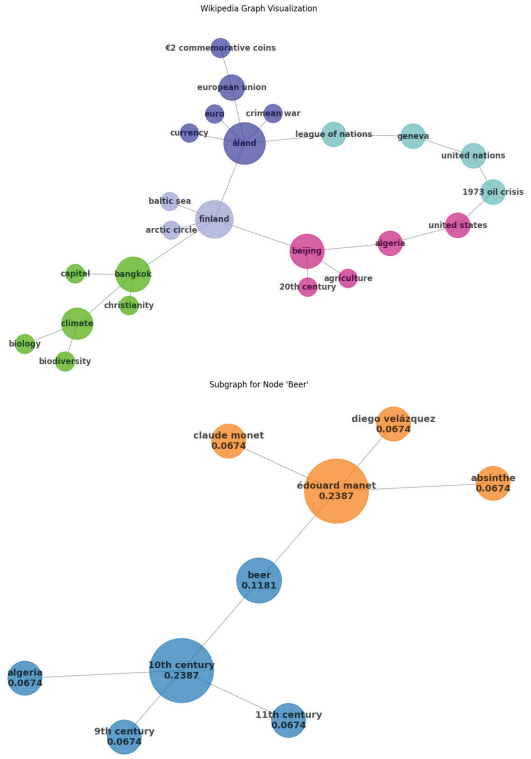


Figure 7: Graphs generated from Wikipedia articles and metadata for a small subset of hyperlink connections. The nodes represent decoded article titles, with their size indicating PageRank (significance based on connections). The colors denote community clusters identified using a greedy modularity algorithm. All node connections, community clusters, and PageRank scores are relative and specific to this subgraph.

The MetaWiki RAG novel framework is a sophisticated methodology designed to leverage Wikipedia's expansive network of articles for knowledge graph creation and metadata retrieval. Firstly, using the SNAP dataset, which provides several node tsv files, we decode all articles names and create an edgeless graph (a graph with only nodes - considering there are orphan and dead end articles).[30] Then we add bidirectional edges (connections between nodes) by loading all article link relationships. Thus each node is an article name and each edge is the hyperlink connection (Figure 7).[31] The entire graph is built using the 'networkx' library and at this stage it is a high level structural graph that is able to capture the network information of Wikipedia.

Then, we create a parameter and text rich graph by developing several node attributes.[44] Firstly, by loading the article text file corresponding to the node we create a node summary attribute. These summaries are generated using Ollama LLMs and condense the article's information into concise, yet informative, summaries. Then, by processing category data we assign a hierarchical list of article categories to each node (Ex: History -> US History -> WW1), this helps in categorical classification of nodes.[45] Thirdly, page rank scores are calculated for each node by evaluating the number and depth of neighbor connections. This reflects the relative importance of each article in a network. Finally, by using the edges we create primary and secondary neighbor attributes (nodes the article is connected to) by traversing the entire graph. These are lists of all one hop and 2 hop connecting nodes (secondary nodes is a dictionary where connecting nodes also has to be mentioned). As an experiment, using Textual Entity RAG a sample of nodes is given a knowledge graph attribute containing a graph of textual relationships for that article page. Thus, during query time this attribute can be utilized for all relevant nodes to provide more direct structured information.

The process of retrieval used in MetaWiki RAG is very efficient and optimized for relevance and precision. The source documents are queries using MMR (Maximum Marginal Relevance search type) to produce semantically relevant yet diverse/distinct text chunks. Then, by performing a similarity search between the user query and the node summaries (a vector database and index was created for node summaries), we retrieve k-most relevant nodes to the query (Ex: Nodes 'Climate change,' 'Renewable energy,' 'Carbon footprint,' and 'Sustainable development' are retrieved when the prompt is about 'Global warming'). For each node we print text attributes such as article title, article summary, page rank, categories, neighbors, etc. Then for the most relevant node, the entire neighbor structure along with node attributes is encoded into linguistic connections for the LLM. This was inspired by Google's Talk Like A Graph mechanism; therefore we are using a combination of incident and social network encoding. This encodes graph data in an understable manner for LLMs and helps them get a high level overview of which concepts and events are connected to each other and how.[21] Finally, by using advanced prompting and reasoning we format all the node and graph information along with instructions on how to leverage this to enhance the traditional context. This helps the LLM prioritize entities whose nodes are most relevant, use the neighbor information to link concepts, use node summaries for content generation especially in cross or multi-concept queries combining multiple concepts. This metadata helps the LLM enhance its ability to generate responses that are both contextually rich and accurate.

As an extension this can be fed to vision models like GPT-4 to gain insights that visualization may better offer. By using plotting libraries, we can extract a sub-graph of all related nodes and feed that image to a vision model along with existing information. Additionally, we have experimented with a ReACT

agent. We give this agent several functions like neighbour_info, node_info, rag_info and based on the user prompt (Ex: if the question is content heavy, requires pathfinding, needs connecting themes) it aims to utilize these functions iteratively to gain observations and decide the next course of action. This is in development, but besides wikispeedia it can be a great tool for dynamically extracting relevant info specific to a user prompt.

Hybridization Mechanism:

We implement an object-oriented framework that encapsulates three graph-based RAG techniques within a LLM generator class. The class has specialized functions to call each retrieval technique. There are additional methods that combine all the outputs to generate a final, comprehensive response. This design enables flexible retrieval and generation pipelines that adjust dynamically based on task complexity.

Sequential Answer Generation. The pipeline begins by querying the traditional baseline RAG, MetaWiki RAG, Textual RAG, Microsoft RAG for each of their individual answers. Then, it asks the LLM to leverage the baseline RAG and MetaWiki's response to generate a combined answer. Similarly, it then asks the LLM to integrate the combined answer and Microsoft's response for an integrated response. Finally, it prompts the LLM to combine Textual Entity's answer with this integrated response for one final answer. Thus it sequentially integrates these models along with sophisticated prompting on how to leverage these techniques. The order is deliberately selected as MetaWiki specialized in high level network coverage and node identification which is key for the initial step. Then Microsoft adds the 2nd tier of community response generation and knowledge graphs. Finally, Textual Entity provides explicit direct graph connections to structure the final response with stronger coherence.

Cumulative Generation with Chain-of-Thought Synthesis. In this strategy, each RAG module operates independently, and all the outputs are collated into one large prompt template. We use contrastive and normal chain of thought reasoning to explain how to leverage each technique's answer properties. The prompt provides a step-by-step reasoning example on how an LLM should navigate all this information and extract key elements from all answers. Textual Entity is used to organize the arguments and ensure factual relevance; Microsoft's is a source of key ideas, characters and themes related to the question; MetaWiki highlights how broader level concepts relate and which ideas to put emphasis on. Thus, this is a synthesis strategy wherein the LLM is fed all the answers cumulatively, augmented using prompting mechanisms

Context Retrieval and Graph Traversal. This system compiles all the graph and text property information generated by each method. The system extracts complete graph structures from the Textual Entity Graph RAG (entity relationships), Microsoft Graph RAG (thematic clusters), and MetaWiki RAG (node summaries and metadata). By combining only the graph and retrieval context each technique internally uses (rather than their final answers), we can get even better responses in some situations. The LLM can structurally utilize the various knowledge graphs, metadata, and community structures to frame its response without being influenced by the individual responses of each technique. Therefore, we are amalgamating the entire graph and RAG context into 1 prompt template to be used by the LLM.

Agent-Based Dynamic Retrieval and Iterative Refinement. This approach involves a dynamic, agent-driven system that selects retrieval strategies based on query requirements. The agent evaluates the nature of the query and determines the appropriate retrieval method to use. For high-level thematic exploration, the agent may prioritize Microsoft RAG, while detailed entity extraction might trigger the Textual Entity Graph RAG. During the retrieval and generation process, the agent continuously monitors the response quality. If necessary, it recalls the different techniques to retrieve missing information or refine the answer. For example, it may extract subgraphs from the Textual Entity Graph or adjust the prompts for Microsoft RAG to improve precision. The agent may adapt the prompts dynamically, targeting specific nodes or relationships to retrieve relevant data. Furthermore, it can systematically keep using the retrieval methods and refine its answer or target different nodes each time. This is the most advanced hybridization framework as an LLM agent can deconstruct questions into multi-level reasoning steps. For example, if a prompt is about "renewable energy", it can perform retrieval on related concepts not directly mentioned ("fossil fuels"). Moreover, based on the output of technique, it can leverage information in the answer as a basis for further inquiry. For example, if "treaty of versailles" is a neighbor node of WW1, then it can inquire further related to this treaty specifically. This model is still under development and still requires heavy optimization and refinement; however, it demonstrates the maximum potential.

Experimentation:

This section outlines the detailed experimental pipelines designed to evaluate and compare the performance of various Retrieval-Augmented Generation (RAG) techniques.

Q1: How do advanced RAG techniques like Textual Entity-based RAG, Microsoft RAG, and MetaWIKI RAG compare to traditional methods, and what are the key strengths and weaknesses of each? The baseline RAG uses classic methods for finding and generating answers, while Textual Entity-based RAG uses textual graph relationships (extracted through special queries) along with hybrid similarity search. Microsoft RAG combines a knowledge graph with community summaries and query focused summarization whereas MetaWIKI RAG uses a large network hyperlink graph based on Wikipedia. The aim is to analyze the answers generated by each technique and get a comparative standing of the various models and their strengths. Specifically, do these three techniques outscore traditional RAG and then amongst them which technique is overall superior.

Q2: How does the combined MetaWIKI and Textual Graph RAG stack up against Microsoft RAG in generating detailed answers, and how does answer structure impact their effectiveness? We use two methods to combine the results from MetaWIKI

DOI: 10.36838/v6i12.11

and Textual Graph: one includes combining the individual answers to create a final answer while the other utilizes answers and the entire context of both techniques to create a more detailed and rich answer. The aim is to measure how the combination of our two techniques compare to Microsoft's GraphRAG. Furthermore, the results help substantiate whether meta data and textual graphs extracted in our technique are comparative to community response generation.

Q3: What are the advantages and challenges of integrating Textual Graph RAG, Microsoft RAG, and MetaWIKI RAG, and how does this integration enhance answer quality? Our goal is to see if amalgamating all three together can create better answers than using any one method alone. We have 3 pipelines for this : combining individual answers of each technique, providing answers and context of all 3 techniques, and only providing context of all 3 techniques. Our study aims to find out if combining these methods improves the quality of answers. Specifically, we want to analyze what aspect of each technique's answer is most unique and useful for the LLM"s final answer. This will indicate to us the current ability of LLMs with regard to diverse graph information and how effective they are at amalgamating the entire context to produce the highest quality response. This also highlights which components of context are futile and thus can be deprioritized.

Q4: How do agentic frameworks and advanced prompting improve RAG techniques like Textual Entity RAG and MetaWIKI RAG, and can they rival Microsoft RAG in complex query handling? In this study, we investigate the improvements of agenting and prompting frameworks for MetaWiki and Textual Entity. The aim is to analyze the advantages provided by various self-evaluation and decision-making frameworks to improve answer coherence. More importantly, this experiment also analyzes which prompting techniques enable most efficient context utilization and answer framing.

Evaluation Framework:

Prompting. Our first few experiments centered on verifying basic prompts which needed an adequate domain level understanding and understanding of cross-connections across various subjects. The aim was to test a broad range of question answering such as similarity and differences,timeline synthesis, impact assessment, high level overviews and cross text links. Advanced prompts required connecting, contrasting, and inferring relationships across complex domains. This included envisioning different scenarios, exploring perspectives of characters, integrating events in hypothetical situations and analyzing multi-dimensional sub-questions. These are some examples:

- "What are the similarities between the 9th, 10th, and 11th centuries?"
- "Create a timeline of history, focusing specifically on India."
- "Write a story about a day in the life of a scholar in the Tang Dynasty during the 9th century. What challenges and opportunities do they face, and how do they contribute to the rich cultural and intellectual landscape of the time?"
- "Envision a brewing competition in a medieval European town in the 9th century. What ingredients and techniques are used, and what social and economic impacts does this event have on the community?"
- "Write an emotive descriptive essay exploring the life of a healer in a small European village during the Dark Ages. How do they navigate the challenges of limited medical knowledge, superstition, and the harsh realities of their time?"

Evaluation. Using the Microsoft Graph RAG evaluation framework as a benchmark, we evaluated answers using LLM linguistic feedback on 4 criteria: comprehensiveness, coverage of all aspects and details of question; diversity, richness and variety of perspectives; empowerment, ability to aid the reader to make informed judgements; and directness, clarity and specificity to concisely address question. To make decisions more transparent, the framework requires the LLM to provide overall rankings, numerical scores and analysis of strengths and weaknesses of each technique for more nuanced reasoning. This feedback process is known as self-evaluation and is an increasingly popular feedback mechanism. Essentially, by utilizing a detailed set of criteria the LLM critiques and provides feedback on its own answers. Furthermore, it is required to justify its ranking of techniques with detailed reasoning using specific references to the responses. One advantage of this , besides the qualitative feedback, is that the linguistic ability of the answer generator and evaluator are similar since they are from the same LLM. Furthermore, the methods can be evaluated on datasets like Natural Questions, HotPotQA, and RAGEval.[46-48]

■ Result and Discussion

This section details the results of our experiments across various Retrieval-Augmented Generation (RAG) techniques and their combinations.

Experiment 1: Individual Performance of Techniques:

After multiple rounds of testing, MetaWiki emerged as the victor with Textual Entity in very close competition (Table 1). MetaWiki's strengths include thorough and multi-dimensional explorations, multi-perspective analysis, which allows informed judgment for the reader by explaining context and directly addressing the question in a clear manner. Textual Entity gives a well rounded overview of micro and macro concepts, with specific focus on character's perspective and directness through perspectives of related entities. However, this fails to address perspectives of multiple actors and emphasize the broader context. Microsoft has a straightforward answer and gives a clear understanding covering essential points. However, it relies on singular perspective and lacks nuanced depth. Baseline RAG ranks last as it is solely text chunk retrieval based without specialized graph information. While the answer is direct, it is rather narrow focused. The baseline ranked last in 95% of the trials and was on average 20-40% lower in scores. This proves

that graph and structured information allows more multifaceted and comprehensive answering.

Table 1: Individual ranking of all techniques on LLM evaluation criteria from a scale of 1 - 10, with 10 being the highest.

Technique\Criteria	Comprehensiveness	Diversity	Empowerment	Directness
Microsoft	7	6	7	9
MetaWiki	8	8	9	9
Textual Entity	9	7	8	8
Baseline	5	4	5	8

Experiment 2: Combined Techniques vs. Microsoft RAG:

The combined answer generated using the entire context and answers emerged as the best technique. Each section of this answer was well-developed covering all sub-components of the question. In nearly 80% of our trials the LLM mentioned this technique provided a superior and "richer variety of insights" along with a more robust analysis (Table 2). Furthermore, "comprehensiveness" and "diversity" were consistently 15-25% higher. However, it could be sharpened for a more direct focus on the main themes and actors in a concise manner. The 2nd combined answer generated only by integrating MetaWiki and Textual Entity's final answers ranked 2nd overall. The responses were clear and had a direct focused perspective, providing a thorough understanding of concepts and their implications. However, it lacked diversity in thought and could benefit from contrasting perspectives. Finally, Microsoft was a formidable opponent. It had a thorough clear understanding explaining all key aspects of the question and concisely covering important themes. However, deeper exploration and analysis was the major flaw. From these observations, we can see the combined answer which has more diverse graph information and various structural and RAG info to use leads to a more holistic answer. That being said, there is a trade off between diversity and directness. The combined answers were longer and therefore less direct in nature.

Table 2: Ranking of combined answers with comparison to Microsoft on LLM evaluation criteria from a scale of 1 - 10, with 10 being the highest.

Technique\Criteria	Comprehensiveness	Diversity	Empowerment	Directness
MetaWiki + TextualEntity (Full Context)	9	8	8	8
MetaWiki + Textual Entity (Only Answers)	8	7	9	9
Microsoft	7	7	8	8

Experiment 3: Combining All Three Techniques:

When combining the answers of 3 techniques with the entire context and answers, we achieve the best performance of any model out of all 3 experiments. It has a 40-60% increase across all metrics compared to the baseline and consistently beats Microsoft RAG with 15-25% score margin. Nearly 30%+ boosts were consistently seen in "diversity", "comprehensiveness", and "empowerment". "Directness" was also consistently high in 100% of trials. The answer thoroughly detailed concepts in context of wider implications and range of perspectives in a well structured and concise manner (Table 3). The only critique is some density could be more broken up. Ranking second is the combined answer integrating only final answers of all 3 techniques. It covers multiple aspects and discusses different angles providing an easy to follow and succinct understanding. However, it could include more specific examples and personal anecdotes while being concise. Third is Microsoft RAG; it covers key areas bridging areas of influence, and it directly relates ideas and clearly outlines impacts. However, it could be streamlined with more broader context and specific examples. Finally, traditional RAG ranked last. Although it provides a straightforward and direct answer to questions, the lack of depth or breadth was its biggest pitfall. Thus, by amalgamating all 3 techniques we holistically capture all properties of text to create the most efficient answer.

Table 3: Ranking of combined answers with reference to Microsoft and Baseline on LLM evaluation criteria from a scale of 1 - 10, with 10 being the highest.

Technique\Criteria	Comprehensiveness	Diversity	Empowerment	Directness
Textual Entity + MetaWiki + Microsoft (Full Context)	10	9	10	9
Textual Entity + MetaWiki + Microsoft (Only Answers)	9	8	9	8
Microsoft	9	8	8	9
Baseline	6	5	6	8

Experiment 4: Impact of Agenting and Prompting Techniques:

The final experiment assessed the impact of agenting and prompting techniques on the performance of MetaWiki RAG and Textual Entity Graph RAG in specific. They both achieved nearly 20-40% performance increases in performance in 90% of experiment trials. Firstly, agenting techniques like self evaluation based on the linguistic feedback were highly effective in improving weak areas in initial outputs. By using this feedback and generating answers the areas of strength increased and achieved much higher rankings and comparative performance to Microsoft RAG - a consistent increase of 30%+ in the refined answer was seen in 95% of trials. Similarly, when agenting like ReACT is used when combining techniques the decision making process helps better integrate diverse components of both techniques. In prompting, I evaluated Few Shot, CoT, Contrastive CoT, Tree of Thoughts, Meta Prompting, Directional Stimulus Prompting, which all saw boosts in performance. With MetaWiki, Meta Prompting (focused on structure and broader patterns of questions) and modified directional prompting (provided hints on information utilization) helped the LLM leverage Wikipedia's high-level node connections and effectively use this graph information in the right manner. With Textual Entity RAG, chain of thought, tree of thought and contrastive chain of thought saw huge benefits. These focused on providing examples and how to utilize entity relationships and keyword pairs to structure and formulate

DOI: 10.36838/v6i12.11

answers. Overall, using prompting and agenting improved performance significantly (Table 4).

Table 4: Relative ranking of MetaWiki & Textual Entity with and without addition of prompting and agenting on LLM evaluation criteria from a scale of 1 - 10, with 10 being the highest.

Technique\Criteria	Comprehensiveness	Empowerment	Diversity	Directness
MetaWiki	7	8	8	7
MetaWiki with Prompting/Agenting	9	8	9	8
Textual Entity	7	7	7	8
Textual Entity with Prompting/Agenting	9	9	8	8

Sub-Experiment: Using MetaWiki to Simulate Wikispeedia Gameplay:

In this experiment, the MetaWiki RAG framework tests the LLM's ability to navigate through Wikipedia's article network by simulating the Wikispeedia game, where the goal is to move from a starting article to a target article through a series of hyperlink jumps (Ex: "sports" -> "football" -> "goalkeeper" -> "Manuel Neuer"). As explained in our technique, each node (article) in the graph comes with rich attributes—including a summary, categories, page rank scores, and lists of primary and secondary neighbors—providing the LLM with essential information for decision-making. The LLM uses an agentic approach that leverages functions we built to receive node information, article summaries, neighbor connections inspired by Google's "Talk Like a Graph" method. By examining the neighboring nodes, categories, and PageRank, the LLM can choose optimal paths to reach the final target article . For example, an article with high page rank has a high number of hyperlink connections - this can be a good node or high-degree hub connecting many articles; categories can help LLM thematically navigate by analyzing which subject-domains the target node belongs to. We successfully found in 60-70% of cases the LLM can effectively navigate 1-3 hop nodes. In nodes with 4 or more connections, the accuracy was limited to 15-20%. However, this is a promising insight into the model's graph-based reasoning and ability to use structured knowledge to make informed jumps, testing its proficiency in multi-hop navigation. Thus, this experiment was instrumental in building test ground that informed our intuition on how we can build reasoning engines with graph based RAG.

Discussion:

This paper set out to prove that leveraging and assimilating network structures and knowledge graphs to augment baseline retrieval augmented general improves LLM reasoning and reduces issues like hallucinations and incoherence. The experimental results successfully substantiate this and underscore the critical role of structured information in enhancing RAG models. Firstly, all graph augmented techniques outperformed baseline RAG in all 4 criteria, which clearly establishes that specialized graph information prevents narrow focused answers and increased depth and coverage of all essential answer components. Our results are supported by popular frameworks such as HybridRAG or KG-FiD, integrating KGs, that have faithfulness (factual accuracy), recall, and answer relevance scores around 0.96, which are consistently higher than traditional VectorRAG by 10-15%.[49] Secondly, techniques like Textual Entity and especially the novel MetaWiki framework performed exceptionally well, highlighting the benefits of incorporating diverse and structured data sources. MetaWiki's neighbor and node attribute information facilitated a multi-dimensional and multi-perspective exploration of text. Furthermore, Textual Entity's graph relationships ensured directness while enhancing organization of micro and macro concepts. The efficacy of our techniques is further substantiated by empirical studies which demonstrate that incorporating knowledge graphs improves performance by 20-30% performance improvements in answer relevance and factual correctness across datasets such as SQuAD and TriviaQA.[50]

Moreover, holistically combining all 3 techniques significantly boosts performance to maximum levels in all criteria, with an average 30% increase in all performance metrics. This suggests that by assimilating various properties and graph formats the utility of the context to provide nuanced and contextually rich answers increases. The results prove that the combination of techniques yielded superior performance compared to individual methods, demonstrating the benefit of hybrid approaches. Moreover, empirical studies, such as COKG-QA, demonstrate that graph-enhanced language models that leverage direct graph information for relation extraction demonstrate improved accuracy and reasoning in generating contextualized answers.[51] By merging MetaWiki, Textual Entity, and Microsoft RAG techniques, the experiments achieved a more holistic and detailed understanding of queries. Microsoft provided a base of community-based summaries; MetaWiki gives high level neighbor structures and relevant node attributes; and Textual Entity gives direct tuples of graph paths - existing empirical analysis has also proven that direct structured graphs are 85% better at capturing implicit textual knowledge representation.[52] Thus majority properties of the text, its concepts and entities are covered through each technique. This ensures the context is multi-dimensional and covers all potential attributes inherent to the text. In fact, reviews in this domain support the idea that diverse knowledge graphs lead to 40% greater breadth of information which improves reasoning in complex queries.[53] Additionally, an integrated pipeline not only improved the answers' breadth and diversity but also addressed trade-offs between directness and comprehensiveness.

While we haven't leveraged traditional metrics of answer evaluation, we can still frame results in terms of these popular benchmarking metrics. While all techniques had high accuracy (similarity to ground truth) scores, knowledge graph techniques generally had higher quality and more nuanced answers. In multi-dimensional queries - quality of response was actually more relevant than pure accuracy. Secondly, there is a certain trade-off between precision (retrieved content that is relevant) and recall (relevant information retrieved). Graph techniques have higher recall as the structural information was highly relevant. However, this risked a slight decrease in precision as directness can decrease due to excessive context. Nonetheless,

using advanced prompting there was a positive overall increase in both these metrics and relevance was higher. Finally, our techniques boasted high diversity scores owing to the variety of perspectives in the answer. Having said this, our paper aims to highlight the qualitative evaluation metrics more and puts more emphasis on the LLM textual feedback received.

Finally, advanced prompting and agenting techniques showed significant improvement, especially in guiding the LLM to efficiently leverage the complex context. This assists LLMs in effectively navigating and utilizing a variety of answers and information. In a nutshell, the results highlight the importance of focusing on dynamic, context-aware hybridization of RAG and knowledge graphs, which can be augmented with prompting.

■ Conclusion

This research demonstrated the integral role of structured information in enhancing the performance of Retrieval-Augmented Generation (RAG) models. Techniques such as MetaWiki and Textual Entity RAG, which leveraged multi-faceted data and explored diverse perspectives, outperformed less structure-driven methodologies. The findings also reveal that while MetaWiki and Textual Entity performed strongly individually, layering these techniques generates more comprehensive and contextually rich responses. Therefore this mechanism encapsulates deeper contextual understanding, information coherence and a wider range of narratives.

Thus, this study also demonstrates that integrating various RAG techniques yields superior outcomes, suggesting that the amalgamation of structured information sources, like MetaWiki and Textual Entity, can significantly improve the quality of generated content. Although advanced prompting and agenting techniques provided significant enhancements, the fundamental takeaway is the importance of structured data and pipeline integration in enhancing the effectiveness of RAG models. Finally, as a practical application this study paves the way for an application for advanced question answering on Wikipedia to transform information navigation and user question answering. This can be further extended to other dataset like TREC-COVID, wherein existing hybrid systems have demonstrated state-of-the-art results in precision (up to 0.96) and recall (1.0). Therefore, our research is an effective step in demonstrating the application of layered graph pipelines in real-world context datasets - Wikipedia especially offers limitless potential to test hybrid mechanisms on a variety of knowledge domains.[54]

Future Works:

Future efforts can focus on developing improved dynamic advanced RAG frameworks that adaptively combine techniques like Textual Entity Graph RAG, Microsoft RAG, and MetaWiki RAG based on the query requirements; this can include multiple agents collaborating to optimize information retrieval. Furthermore, expanding these models across different domains, such as healthcare and law, can reveal domain-specific characteristics unique to enhancing contextual awareness. Moreover, synthesizing Textual Entity with MetaWiki is a very promising approach that leverages high level networks and direct graph connections as individual node attributes.

Additionally, graph retrieval methods can work on prioritizing meaningful connections and exploring thematic retrieval techniques. This includes identifying indirect graph connections that do not contain direct keywords but are thematically relevant. Finally, practical applications such as an advanced Wikipedia-based search tool, to revolutionize information navigation and data assimilation for answering user queries, presents significant commercial potential. Another exciting direction is the development of a capable (reinforcement-learning based) LLM framework that utilizes MetaWiki graph properties to participate in the Wikispeedia game and simulate human navigation for shortest path identification.

■ Acknowledgments

I would like to thank and express my deepest gratitude towards Dr. Siddharth Krishnan, University of North Carolina at Charlotte, for his invaluable support and guidance through this research endeavor. However, I attest that the ideas and writing in this paper are entirely my own.

■ References

1. IEEE, "The Recent Large Language Models in NLP," *IEEE Xplore*, 2023. Available: https://ieeexplore.ieee.org/document/10376050
2. Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang, "Retrieval-augmented generation for large language models: A survey," arXiv, Mar. 27, 2024. doi: 10.48550/arXiv.2312.10997.
3. H. Li, Y. Su, D. Cai, Y. Wang, and L. Liu, "A survey on retrieval-augmented text generation," arXiv, Feb. 13, 2022. doi: 10.48550/arXiv.2202.01110.
4. C. Alberti, K. Lee, and M. Collins, "A BERT baseline for the natural questions," *arXiv*, vol. 1901.08634, 2019. [Online]. Available: https://paperswithcode.com/paper/a-bert-baseline-for-the-natural-questions
5. A. Hogan *et al.*, "Knowledge graphs," ACM Comput. Surv., vol. 54, no. 4, pp. 1–37, 2022. doi: 10.1145/3447772.
6. J. Huang, X. Zhang, Q. Mei, and J. Ma, "Can LLMs effectively leverage graph structural information through prompts, and why?" arXiv, Jun. 15, 2024. doi: 10.48550/arXiv.2309.16595.
7. J. Cowls, M. Kejriwal, and P. Szekely, "Knowledge Graphs and COVID-19: Opportunities, Challenges, and Implementation," Data Intelligence, vol. 4, no. 3, pp. 471–492, 2020, doi: 10.1162/dint_a_00154.
8. D. Li and F. Xu, "Synergizing knowledge graphs with large language models: A comprehensive review and future prospects," arXiv, Jul. 25, 2024. doi: 10.48550/arXiv.2407.18470.
9. G. Perković, A. Drobnjak, and I. Botički, "Hallucinations in LLMs: Understanding and addressing challenges," in *2024 47th International Convention on Information, Communication and Electronic Technology (MIPRO)*, 2024, pp. 2084–2088. doi: 10.1109/MIPRO60963.2024.10569238.
10. C. Cao *et al.*, "Prompting is all you need: LLMs for systematic review screening," medRxiv, Jun. 3, 2024, p. 2024.06.01.24308323. doi: 10.1101/2024.06.01.24308323.
11. "Prompt engineering in consistency and reliability with the evidence-based guideline for LLMs," *npj Digital Medicine*. Available:

DOI: 10.36838/v6i12.11

https://www.nature.com/articles/s41746-024-01029-4 (accessed Aug. 14, 2024).
12. J. Zhang *et al.*, "Exploring collaboration mechanisms for LLM agents: A social psychology view," *arXiv*, May 27, 2024. doi: 10.48550/arXiv.2310.02124. [Online]. Available: https://doi.org/10.48550/arXiv.2310.02124
13. D. Edge *et al.*, "From local to global: A graph RAG approach to query-focused summarization," *arXiv*, Apr. 24, 2024. doi: 10.48550/arXiv.2404.16130. [Online]. Available: https://doi.org/10.48550/arXiv.2404.16130
14. A. Tamkin, M. Brundage, J. Clark, and D. Ganguli, "Understanding the capabilities, limitations, and societal impact of large language models," arXiv, Feb. 4, 2021. doi: 10.48550/arXiv.2102.02503.
15. A. Kau, X. He, A. Nambissan, A. Astudillo, H. Yin, and A. Aryani, "Combining knowledge graphs and large language models," arXiv, Jul. 9, 2024. doi: 10.48550/arXiv.2407.06564.
16. Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, and L. Zhao, "GRAG: Graph retrieval-augmented generation," arXiv, May 26, 2024. doi: 10.48550/arXiv.2405.16506.
17. R. Chen, T. Zhao, A. Jaiswal, N. Shah, and Z. Wang, "LLaGA: Large language and graph assistant," arXiv, Apr. 11, 2024. doi: 10.48550/arXiv.2402.08170.
18. J. Guo, L. Du, H. Liu, M. Zhou, X. He, and S. Han, "GPT4Graph: Can large language models understand graph-structured data? An empirical evaluation and benchmarking," arXiv, Jul. 11, 2023. doi: 10.48550/arXiv.2305.15066.
19. P. 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 Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 9459–9474.
20. B. Jin, G. Liu, C. Han, M. Jiang, H. Ji, and J. Han, "Large language models on graphs: A comprehensive survey," arXiv, Feb. 1, 2024. doi: 10.48550/arXiv.2312.02783.
21. B. Fatemi, J. Halcrow, and B. Perozzi, "Talk like a graph: Encoding graphs for large language models," arXiv, Oct. 6, 2023. doi: 10.48550/arXiv.2310.04560.
22. M. Kejriwal and Y. Wang, "Trends and challenges in applying large language models to structural data tasks," IEEE Access, vol. 11, pp. 10234–10250, 2023, doi: 10.1109/ACCESS.2023.3260784.
23. R. West and J. Leskovec, "Human wayfinding in information networks," 2012.
24. J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," arXiv, Jan. 10, 2023. doi: 10.48550/arXiv.2201.11903.
25. Y. K. Chia, G. Chen, L. A. Tuan, S. Poria, and L. Bing, "Contrastive chain-of-thought prompting," arXiv, Nov. 15, 2023. doi: 10.48550/arXiv.2311.09277.
26. M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, and T. Hoefler, "Graph of thoughts: Solving elaborate problems with large language models," in Proc. AAAI Conf. Artif. Intell., vol. 38, no. 16, 2024, pp. 17682–17690. doi: 10.1609/aaai.v38i16.29720.
27. S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," arXiv, Dec. 3, 2023. doi: 10.48550/arXiv.2305.10601.
28. S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," arXiv, Mar. 9, 2023. doi: 10.48550/arXiv.2210.03629.
29. N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," Adv. Neural Inf. Process. Syst., vol. 36, 2023, pp. 8634–8652.
30. J. Kamps and M. Koolen, "The importance of link evidence in Wikipedia," in Advances in Information Retrieval, C. Macdonald, I. Ounis, V. Plachouras, I. Ruthven, and R. W. White, Eds., vol. 4956, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2008, pp. 270–282. doi: 10.1007/978-3-540-78646-7_26.
31. "A graph-structured dataset for Wikipedia research," Companion Proc. The 2019 World Wide Web Conf. Available: https://dl.acm.org/doi/abs/10.1145/3308560.3316757 (accessed Aug. 15, 2024).
32. L. Wang, Y. Li, O. Aslan, and O. Vinyals, "WikiGraphs: A Wikipedia text - knowledge graph paired dataset," arXiv. Available: https://arxiv.org/abs/2107.09556v1 (accessed Aug. 25, 2024).
33. H. Sharma, "Survey of research on chunking techniques," unpublished.
34. R. Sharnagat, "Named entity recognition: A literature survey," unpublished.
35. C. Wang, X. Ma, J. Chen, and J. Chen, "Information extraction and knowledge graph construction from geoscience literature," Comput. Geosci., vol. 112, pp. 112–120, 2018. doi: 10.1016/j.cageo.2017.12.007.
36. M. Gardner and T. Mitchell, "Efficient and expressive knowledge base completion using subgraph feature extraction," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisbon, Portugal, 2015, pp. 1488–1498. doi: 10.18653/v1/D15-1173.
37. "Knowledge graphs: Opportunities and challenges," *Artificial Intelligence Review*. [Online]. Available: https://link.springer.com/article/10.1007/s10462-023-10465-9 (accessed Aug. 15, 2024).
38. Y. Han, C. Liu, and P. Wang, "A comprehensive survey on vector database: Storage and retrieval technique, challenge," *arXiv.* [Online]. Available: https://arxiv.org/abs/2310.11703v1 (accessed Aug. 15, 2024).
39. J. J. Miller, Graph Database Applications and Concepts with Neo4j, 2013.
40. R. Angles, M. Arenas, P. Barceló, A. Hogan, J. Reutter, and D. Vrgoč, "Foundations of modern query languages for graph databases," ACM Comput. Surv., vol. 50, no. 5, pp. 68:1–68:40, 2017. doi: 10.1145/3104031.
41. W. 文继军 and S. 王珊, "SEEKER: Keyword-based information retrieval over relational databases," J. Softw., vol. 16, no. 7, pp. 1270–1281.
42. B. Perozzi, B. Fatemi, D. Zelle, A. Tsitsulin, M. Kazemi, R. Al-Rfou, and J. Halcrow, "Let your graph do the talking: Encoding structured data for LLMs," arXiv. Available: https://arxiv.org/abs/2402.05862v1 (accessed Aug. 15, 2024).
43. W. Zhang, S. Vakulenko, T. Rajapakse, Y. Xu, and E. Kanoulas, "Tackling query-focused summarization as a knowledge-intensive task: A pilot study," arXiv, Jul. 31, 2023. doi: 10.48550/arXiv.2112.07536.
44. Y. Li, Z. Yu, and D. He, "Text-rich graph neural networks with subjective-objective semantic modeling," IEEE Trans. Knowl. Data Eng., vol. 36, no. 9, pp. 4956–4967, 2024. doi: 10.1109/TKDE.2024.3378914.
45. Q. Wang, X. Wang, Z. Chen, and R. Wang, "The category structure in Wikipedia: To analyze and know how it grows," in Chinese Lexical Semantics, P. Liu and Q. Su, Eds., vol. 8229, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2013, pp. 538–545. doi: 10.1007/978-3-642-45185-0_56.
46. "Natural questions: A benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, MIT Press. Available: https://direct.mit.edu/tacl/article/

doi/10.1162/tacl_a_00276/43518/Natural-Questions-A-Benchmark-for-Question (accessed Aug. 17, 2024).
47. Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, "HotpotQA: A dataset for diverse, explainable multi-hop question answering," arXiv, Sep. 25, 2018. doi: 10.48550/arXiv.1809.09600.
48. K. Zhu *et al.*, "RAGEval: Scenario specific RAG evaluation dataset generation framework," arXiv, Aug. 2, 2024. doi: 10.48550/arXiv.2408.01262.
49. B. Sarmah, B. Hall, R. Rao, S. Patel, S. Pasquali, and D. Mehta, "HybridRAG: Integrating Knowledge Graphs and Vector Retrieval Augmented Generation for Efficient Information Extraction," *arXiv preprint*, arXiv:2408.04948, 2024. Available: https://ar5iv.labs.arxiv.org
50. B. Sarmah, B. Hall, R. Rao, S. Patel, S. Pasquali, and D. Mehta, "HybridRAG: Integrating Knowledge Graphs and Vector Retrieval Augmented Generation for Efficient Information Extraction," arXiv preprint, arXiv:2408.04948, 2024.
51. H. Du, Z. Le, H. Wang, Y. Chen, and J. Yu, "COKG-QA: Multi-hop Question Answering over COVID-19 Knowledge Graphs," Data Intelligence, vol. 4, no. 3, pp. 471-492, 2022, doi: 10.1162/dint_a_00154.
52. W. Jiang, Y. Liu, and M. Gao, "Think-on-Graph 2.0: Deep and Interpretable Large Language Model Reasoning with Knowledge Graph-guided Retrieval," arXiv preprint, arXiv:2407.10805, 2024.
53. M. Kejriwal, "Knowledge Graphs: A Practical Review of the Research Landscape," Information, vol. 13, no. 4, pp. 161, 2022, doi: 10.3390/info13040161.
54. W. R. Hersh and J. S. Chen, "A comparative analysis of system features used in the TREC-COVID information retrieval challenge," Journal of Biomedical Informatics, vol. 117, p. 103745, 2021. Available: https://dmice.ohsu.edu/publications.html.

Author

Vedanth Aggarwal is a high schooler who aims to major in computer science. He has a huge passion for exploring novel domains such as machine learning, natural language processing, and artificial intelligence. As the founder of global STEM NGOs with 2000+ members, he aims to promote accessible technology education and build socially impactful projects that solve global issues. Additionally, he is an aspiring entrepreneur and enjoys exploring emerging startups in GenAI and Fintech.

DOI: 10.36838/v6i12.11