武漢大學

本科毕业论文（设计）

Hybrid Knowledge Graph Retrieval and Term Coverage for Robust Scientific Query Answering

姓　　名: Chaiwat Plongkaew

学　　号: 2021326660023

专　　业: 软件工程

学　　院: 计算机学院

指导教师: 邓娟

二○二五年五月

原创性声明

本人郑重声明：所呈交的论文（设计），是本人在指导教师的指导下，严格按照学校和学院有关规定完成的。除文中已经标明引用的内容外，本论文（设计）不包含任何其他个人或集体已发表及撰写的研究成果。对本论文（设计）做出贡献的个人和集体，均已在文中以明确方式标明。本人承诺在论文（设计）工作过程中没有伪造数据等行为。若在本论文（设计）中有侵犯任何方面知识产权的行为，由本人承担相应的法律责任。

作者签名：　　指导教师签名：

日　　期：　　年　　月　　日

版权使用授权书

本人完全了解武汉大学有权保留并向有关部门或机构送交本论文（设计）的复印件和电子版，允许本论文（设计）被查阅和借阅。本人授权武汉大学将本论文的全部或部分内容编入有关数据进行检索和传播，可以采用影印、缩印或扫描等复制手段保存和汇编本论文（设计）。

作者签名：　　指导教师签名：

日　　期：　　年　　月　　日

摘　　要

在科学问答任务中，尤其是在生物医学等高风险领域，大型语言模型（LLMs）常常生成流畅但不完整或缺乏依据的回答，这是由于它们对结构化领域知识的访问受限。检索增强生成（Retrieval-Augmented Generation, RAG）系统尝试通过引入外部信息检索来缓解这一问题。然而，传统的稠密检索方法通常无法保证上下文的完整性和语义对齐，导致相关证据召回率低、答案上下文碎片化。

本论文提出了 CoTeRAG-S，一种混合式 RAG 框架，融合了基于知识图谱的社区检测、语义向量检索和基于术语覆盖的过滤机制，以提升检索内容的质量。通过将实体聚类为语义上连贯的社区，并利用 LLM 进行摘要生成，该系统实现了更聚焦的检索和结构化的证据信息组织。对于稀疏查询，还引入了后备检索机制，以提高在复杂信息需求下的覆盖能力和鲁棒性。

在 CORD-19 数据集上的实验结果显示，与 ChatGPT-4o 相比，CoTeRAG-S 在上下文召回率（0.66 vs. 0.32）和上下文精确度（0.77 vs. 0.67）方面有显著提升，表明其在检索更完整且语义对齐的证据方面具有优势。尽管在答案相关性指标上略低（0.93 vs. 0.94），但其增强的上下文覆盖能力验证了图结构引导的混合检索策略的有效性。上述结果表明，CoTeRAG-S 能够为科学问答任务提供更加精准、连贯且具有语义深度的答案内容。

关键词：检索增强生成、大型语言模型、社区检测、知识图谱、社区摘要、术语覆盖、科学问答、混合检索

ABSTRACT

In scientific query answering, particularly in high-stakes domains like biomedicine, large language models (LLMs) often generate fluent yet incomplete or ungrounded responses due to their limited access to structured domain knowledge. Retrieval-Augmented Generation (RAG) systems attempt to address this limitation by incorporating external information retrieval. However, traditional dense retrieval methods often fail to ensure contextual completeness and alignment, leading to low recall of relevant evidence and fragmented answer contexts.

This thesis introduces CoTeRAG-S, a hybrid RAG framework that integrates knowledge graph-based community detection, semantic vector retrieval, and term-aware filtering to enhance the quality of retrieved context. By clustering entities into semantically coherent communities and summarizing them with an LLM, the system supports more focused retrieval and structured evidence organization. A fallback mechanism further expands coverage for sparse queries, ensuring robustness in complex information needs.

Experimental results on the CORD-19 dataset show that CoTeRAG-S significantly improves Contextual Recall (0.66 vs. 0.32) and Contextual Precision (0.77 vs. 0.67) compared to ChatGPT-4o, indicating its strength in retrieving more complete and topically aligned evidence. While it performs slightly lower in Answer Relevancy (0.93 vs. 0.94), its improved context coverage demonstrates the benefit of graph-guided hybrid retrieval. These findings validate the effectiveness of CoTeRAG-S in delivering more targeted, coherent, and context-rich answers for scientific queries.

Key words: Retrieval-Augmented Generation; Large Language Models; Community Detection; Knowledge Graph; Community Summarization; Term Coverage; Scientific Query Answering; Hybrid Retrieval

目　录

1 Introduction

1.1 Background and Motivation

Scientific query answering demands not only fluent responses but also accuracy, completeness, and interpretability—particularly in domains like biomedicine, where precision is critical. Retrieval-Augmented Generation (RAG) systems have emerged as a powerful approach to meet this need by combining large language models (LLMs) with external knowledge retrieval. By grounding generation in retrieved content, RAG systems enable LLMs to produce more contextually relevant and factually accurate answers.

Despite recent progress in LLMs such as ChatGPT-4o, standalone LLM-based scientific QA continues to face significant challenges. These models often produce fluent yet ungrounded responses, hallucinate information, and struggle with complex reasoning tasks like multi-hop inference. These limitations stem from their reliance on pretraining over large corpora, which lacks real-time access to structured, domain-specific knowledge. As a result, answers may appear coherent but are frequently incomplete, unverifiable, or factually imprecise—posing serious risks in high-stakes fields such as healthcare and biomedical research.

This thesis focuses on a central problem: the lack of grounded, interpretable, and contextually complete evidence in LLM-based scientific question answering, which undermines answer reliability in specialized domains.

Recent advances have attempted to address these issues by integrating structured retrieval methods, particularly Knowledge Graphs (KGs), into RAG pipelines. KGs provide explicit representations of entities and their relationships, offering a more interpretable and semantically precise retrieval space. However, existing RAG approaches often rely heavily on dense vector similarity, which can miss important structural connections and overlook nuanced domain-specific terminology. Moreover, they frequently underutilize the rich topological features of KGs, especially for tasks like multi-hop reasoning and entity disambiguation.

To address these gaps, this thesis proposes CoTeRAG-S (Community- and Term-

aware Retrieval-Augmented Generation with Summarization), a novel framework that enhances retrieval precision, contextual relevance, and interpretability in scientific QA. CoTeRAG-S introduces four key innovations:

- **Community-Aware Filtering** through graph clustering (e.g., Leiden),
- **Hybrid Retrieval** that combines dense vector search with graph-based multi-hop traversal,
- **Term Coverage Validation** to ensure alignment with key query concepts,
- **LLM-based summarization of communities** to distill relevant information and reduce noise.

Through this multi-faceted architecture, CoTeRAG-S prioritizes the retrieval of content that is not only semantically pertinent but also structurally grounded and terminologically exhaustive. By filtering candidate evidence via graph communities and validating against term coverage, the system effectively narrows the search space to thematically relevant regions. Subsequent summarization further refines the contextual input for generation, promoting concise and factually coherent responses.

Empirical evaluation demonstrates that while CoTeRAG-S does not outperform ChatGPT-4o in every metric, it achieves higher performance in key areas—specifically, contextual recall and contextual precision. These improvements indicate that the proposed retrieval strategies are highly effective at identifying relevant, well-aligned context, which in turn enhances the reliability and completeness of the generated answers.

1.2 Literature Review

1.2.1 Retrieval-Augmented Generation (RAG)

1.2.1.1 Overview of RAG Models

The RAG model became prominent when Lewis et al. (2020) introduced it in their research[1]. An LLM receives text evidence from a retriever component before generating its output based on this evidence. The model architecture combines parametric memory stored in weight parameters with external non-parametric memory documents[1]. The RAG model from Lewis et al. achieved leading results on knowledge-based tasks including open-domain QA by obtaining suitable passages for queries then generating answers linked to retrieved evidence[1]. Following this, Izacard & Grave (2021) The researchers

created Fusion-in-Decoder (FiD) which enables a sequence-to-sequence model decoder to process multiple retrieved passages simultaneously[2]. The FiD model achieves better answer accuracy when it receives additional retrieved passages because it learns to combine evidence across multiple documents[2].

1.2.1.2 Limitations in Knowledge Selection

Despite these advancements, existing RAG approaches have notable limitations. One major issue is that the retriever might not always select the right pieces of knowledge[3]. The dense vector retrievers (like DPR by Karpukhin et al., 2020)—used in RAG and FiD—excel at semantic similarity but can sometimes miss relevant documents that use different terminology or that require multi-hop reasoning[3].

Moreover, RAG models often retrieve a fixed number of top documents based purely on similarity score; if the answer resides in a document that’ s, say, the 11th most similar, it will be missed (this is sometimes called the “Missed Top Ranked” issue in RAG failures)[4].

Another limitation is that RAG models have been observed to hallucinate—producing information not supported by any retrieved document[5]. This can happen when the model’ s generative prior overpowers the grounding from retrieval, especially if the retrieved passages are only loosely related to the query[5][6].

Studies have found that hallucinations often stem from the model trying to fill gaps when the retrieved evidence is insufficient or not specific enough[7]. In summary, traditional RAG provides a strong framework but struggles with knowledge selection: it might retrieve either too little (missing key facts) or too much (including irrelevant text), which in turn affects the generation quality.

1.2.2 Knowledge Graphs in Retrieval and Question Answering

1.2.2.1 Deep Retrieval over Knowledge Graphs

A developing field of research is including knowledge graphs (KGs) into retrieval and QA[8][9]. Knowledge graphs (KGs) provide a structured form of knowledge that complements unstructured text[5][8] encoding entities (nodes) and their relationships (edges). Graph-based retrieval—where the objective is to return not just documents but also a rel-

evant subgraph containing the answer—has been one line of work[8][6]. Yasunaga et al. (2022) and associated studies investigate ways to concurrently use text and graphs[8]. For instance, Yasunaga et al. (2022) embed the KG using graph neural networks (GNNs) and then include such embeddings into language model training or fine-tuning[8]. This enables the model to reason with the structure of the graph, therefore enhancing jobs requiring multi-step thinking or knowledge of complicated relationships[8][9]. Although the model reasoning can be partially followed through the graph, these methods have shown enhanced explainability since they frequently call for considerable task-specific training to align the graph and text representations[10][11].

1.2.2.2 Graph-Based Traversal for Knowledge Augmetation

Another approach is to use KGs at query time to find relevant information[6][10]. Instead of retrieving documents via keywords or vectors, the system can traverse the knowledge graph[9][12]. For example, given a question about a biochemical pathway, the system could start at the node representing a protein of interest and traverse edges to find related proteins, genes, or processes, assembling a set of facts that answer the query[8][11]. Yasunaga et al. (2021) introduced QA-GNN, which uses a graph network over a KG (ConceptNet in that case) to re-rank answer options for QA, effectively performing a graph traversal to validate answers[8]. More directly, Sun et al. (2023) propose Think-on-Graph, a method where an LLM uses a KG to guide its reasoning[6]. Think-on-Graph extracts relevant triples from the KG to include as part of the context, so the LLM isn’t just given text passages but also “facts” in the form of subject-predicate-object triples[6]. This has been shown to assist in question answering, as the model can see explicit relationships (for example, that compound X inhibits enzyme Y)[6][10]. In general, the integration of KGs in QA can happen either pre-retrieval (e.g., constrain the retrieval using KG, like only retrieve papers connected to a certain topic node) or post-retrieval (e.g., validate or re-rank retrieved answers by checking against a KG)[5][7]. The literature from 2019–2023 (e.g., KagNet, QA-GNN, KG-FiD) consistently finds that KGs help fill in the reasoning gaps that pure text often leaves, especially for multi-hop questions where an intermediate inference is needed[5][6][12]. A challenge noted in works like Both et al. (2021) is that combining KGs with text retrieval requires careful design to avoid introducing noise—

not every connection in a KG is useful, and sometimes a graph can be incomplete or outdated[9]. Nonetheless, the trend is to use KGs as a scaffolding for retrieval: they can guide the system to the right area of knowledge and ensure consistency in the answer[8][11].

1.2.3 Community Detection in Knowledge Graphs

1.2.3.1 Clustering-Based Retrieval Structuring

Academic researchers apply clustering algorithms to retrieve information through systematic organization methods because knowledge graphs and document collections have become extensive[9][10]. The goal involves detecting communities, which are groups of interconnected nodes or documents that show substantial connections or share similar concepts, for using them as retrieval or reasoning units[10][13]. The research by Both et al. (2021) explores clustering techniques for document collection organization, which enables questions to connect first to clusters, then to identify individual clustered documents instead of searching the whole corpus at once[9]. Enhanced search speed and accuracy result when the method focuses itself on a specified thematic segment[9][13]. The research by Galkin et al. (2020) focuses on knowledge graph community structures to improve question answering through subgraph or community selection that best matches queries and targeted search within those selected areas[10]. The detection of communities within graphs uses frequently applied algorithms such as Louvain (Blondel et al., 2008) and Leiden (Traag et al., 2019)[14][15]. Such techniques create optimal modularity clusters by grouping nodes that maintain many internal ties along with minimal relations to nodes outside the cluster[14][15]. Science literature citation graphs contain Louvain communities, which serve to define research domains as well as individual subject areas[9]. According to Chang et al. (2024), CommunityKG-RAG represents a system that implements community structures into RAG for fact-checking by applying Louvain to fragment the knowledge graph, followed by information retrieval based on communities[7]. The system achieved better contextual relevance in its fact retrievals because it could identify relevant information through multiple link hops within separate community clusters[7]. Community-aware retrieval functions as a major benefit by overcoming scattering problems since it delivers consolidated results that belong to related topics while focusing based on community connections[9][13].

1.2.3.2 Graph Embeddings and Community-Aware Retrieval

Edge embeddings represent community structure directly without specific modification[10][4]. The embedding techniques DeepWalk and node2vec (along with current GNN-based methods) create vector representations that position nodes together when they belong to the same community[4][11].

According to Galkin et al. (2020) and other researchers, embedding of graphs allows retrieval tasks to utilize graph-space similarity measures instead of text-space similarity[10]. The system retrieves documents that talk directly about mentioned concepts but also delivers documents related to the same community found within the embedding space[10][11].

Research has shown that clustering techniques combined with community detection methods serve to provide topical structure for retrieval problems when dealing with big KGs or corpora[9][13]. The main issue arises when communities become either too extensive or too specific—broad communities lose focused retrieval benefits, while narrow communities fail to contain information outside their scope[10][11].

These research works establish hierarchical cluster detection systems to deal with this issue through domain-level clustering at the top, followed by subtopic identification at the lower level, which supports multi-level retrieval systems[9].

1.2.4 Ensuring Term (Concept) Coverage

The strategy includes selecting results through filtering or ranking methods according to term coverage. After obtaining an initial set of 50 documents as retrieval, the system can evaluate the document frequency of each term or collectively for sets of documents. Zhuang and Zuccon (2021) created the TILDE approach which performs ahead-of-time computations of every possible query term effect on document ratings[15]. The system generates document scores upon summing the contribution values of each term which appears in the document during query time. A document containing greater numbers of query terms, including the uncommon or specific terms, receives enhanced prominence through this process.

Zhuang et al. (2023) proved through their research on scientific information retrieval that broadening the scope of topic-related terms improves your ability to retrieve neces-

sary documents, thus boosting your overall answer completion rate[13]. The study by Ma et al. (2021) revealed that precision increases when systems remove documents which do not fulfill specific mandatory search terms when users have multiple query conditions[12]. When RAG retrieves context for an LLM, it becomes less likely for the LLM to make intuitive guesses or ignore any sections of complex questions since the chosen context should fulfill all requirement terms. Term coverage alone doesn't represent the only system objective because relevance plays an equally important role. The ideal document contains every search term yet remains useless if these words are presented out of proper context. Current systems implement a combined method of semantic retrieval and term coverage analysis for information retrieval, which is similar to our proposed method.

1.2.5 Graph-Based Retrieval and Knowledge Augmentation

1.2.5.1 Multi-Hop Reasoning in Retrieval

Science-related information needs often demand multiple sources when trying to find answers because respondents need to combine content from multiple records. The basic retrieval method returns one document which contains limited information, but the complete response demands cross-document linkage of facts from two or more texts. Boros et al. (2022) explored multi-hop retrieval methods which require the system to retrieve evidence through progressive steps: it begins with retrieving content about entity A followed by a new query generation to find information about entity B which relates to A and so on[5]. The method allows chaining to work through either knowledge graphs or text-based query transformation processes. Recent RAG models added retrieval chains into their operation through iterative retrieval-generation loops according to Shao et al. (2023) and Press et al. (2023)[16][7]. As part of their iterative RAG processes, these models conduct multi-hop reasoning while depending on model guidance for chain navigation, which becomes unstable when the model introduces topic deviations.

1.2.5.2 Graph-Based Multi-Hop Retrieval

The retrieval method based on graphs offers natural capabilities to perform multi-hop reasoning. Knowledge graphs contain multi-hop query responses through node path discovery where one endpoint could be a drug while the other endpoint could be a disease

and potentially connect through a gene node. The authors of Cai et al. (2023) developed a retrieval framework which retrieves a series of connected documents that function as an answer to the query[11]. The system retrieves first an article showing X affects Y, followed by another article exposing the relationship Y affects Z, thereby linking X→Y→Z in a reasoning sequence. With relevance constraints in place, graph retrieval systems apply algorithms including BFS or DFS over the knowledge graph. Cai et al. mention exploding subgraphs as a major issue because exploring large graphs can cause a surge in potential search results. Heuristics and scoring functions operate as search maintenance tools to direct exploration in graph retrieval systems by following high-confidence edges and query term-containing nodes.

1.2.5.3 Graph-Aware Ranking Strategies

Graph structure serves as a basis to help prioritize and choose retrieved information. We would likely select the evidence piece that answers the question since it has stronger connections to additional key facts that form a solid subgraph structure compared to the isolated information (some literature terms this as ”answer context coherence”). The authors Boros et al. (2022) could have implemented this logic for multi-hop answer re-ranking by evaluating evidence network connectivity[5]. Graph-enhanced generation involves providing models with structured fact collections originating from graph-based data accesses rather than simple passage lists. Industrial systems integrate LLMs with graph-based retrieval to address recommendation and extended Q&A tasks (*MULTI-IS* was used as an example by Microsoft, and Google employed SGNet but under different names). Peng et al. (2024) presented in their survey about GraphRAG that applying relational structure allows better knowledge capture and generates more precise context-sensitive responses[17]. The approach called GraphRAG applies two components: Graph-Based Indexing, which transforms knowledge into graph index structures, alongside Graph-Guided Retrieval as a text retrieval enhancement. The research demonstrates how community detection methods on graphs lead to summary generation for retrieval purposes, which forms the basis for our work. An LLM obtains the summary of entire communities by retrieval to provide developers and executives with an overview of knowledge before investigating specific details.

Separate RAG models of 2019 evolved into contemporary systems based on graphs and clustering, and hybrid retrieval methods which enhance knowledge selection performance. The research behind this work builds upon these previous studies. The research aims to combine various prominent threads from the field, including graph-based community retrieval, term coverage heuristics, and iterative search together with LLM generation into an integrated system while testing its performance on scientific QA tasks. Our methodology receives detailed explanation in the following chapter because past studies established essential principles which direct our design decisions and specify key innovations needed for advanced progress.

1.3 Research Contents

A methodology for achieving the objectives involves integrating graph-based approaches with standard retrieval and generation methods.

1.3.1 Structuring Scientific Knowledge using Graph-Based Indexing and Community Clustering

In our system, we construct a domain-specific knowledge graph that interlinks extracted entities and relationships based on shared terminology, co-occurrence within documents, and semantic relations. Each node in the graph represents an entity—such as a disease, drug, or protein—and edges represent structured relationships derived from relation extraction. To improve retrieval precision and efficiency, we apply the Leiden algorithm[14] to detect communities, which are clusters of closely related entities. These communities reflect meaningful subdomains within the scientific corpus and serve as the basis for focused, community-level retrieval.

Traditional retrieval systems often perform a flat search, scanning the entire corpus to identify relevant documents. This method is computationally expensive and may retrieve results that are semantically broad or less contextually relevant, especially in scientific domains where the knowledge space is vast and complex. As a result, these systems may struggle to focus on the most relevant portion of the knowledge space for a given query.

By precomputing and organizing the knowledge into communities, our graph-based indexing system enables community-level retrieval. This approach allows the system to identify and focus on the most relevant community or subdomain for a given query, rather

than performing a global search across the entire knowledge base. As each community represents a tightly knit cluster of facts, this method ensures that the retrieved content is not only relevant but also thematically coherent. This targeted retrieval leads to improved precision and efficiency, ensuring that the system can handle complex scientific queries where domain-specific knowledge is crucial.

In addition to community-level retrieval, the summarization of community content further refines the process. After identifying the most relevant communities, we perform community summarization, which extracts and condenses the core knowledge from each community, reducing redundancy and ensuring that only the most relevant and focused information is included in the final results. This summarization step enhances the coherence of the retrieved content, making sure that it is concise and directly aligned with the query’s focus.

Precomputing these communities and performing summarization within each community helps to drastically reduce the search space during retrieval, ensuring that only the most relevant and contextually aligned knowledge is considered. This leads to a more precise and focused search, minimizing irrelevant results and significantly improving retrieval times. The summarization ensures that, even within a specific community, the system doesn’t retrieve excessive information—only the most critical and representative content is returned, improving the quality and relevance of the generated answers.

1.3.2 Bridging Semantic Gaps in Dense Retrieval via Combined Graph-Based Search

A key contribution of this research is the introduction of a hybrid retrieval mechanism that combines dense vector search with symbolic graph traversal, allowing the system to retrieve information that is both semantically aligned and structurally coherent. This design addresses critical limitations in traditional RAG systems, which typically rely solely on vector-based retrieval.

Standalone vector retrieval retrieves passages based on semantic similarity, making it effective for identifying conceptually related content. However, it lacks awareness of structural relationships between entities. This can lead to incomplete answers, especially in complex scientific queries where reasoning over factual connections —such as drug-gene-disease interactions —is essential.

In contrast, graph-based retrieval excels at capturing explicit relationships between entities by navigating the knowledge graph. It enhances factual grounding by leveraging structural connections, but it may miss relevant context if semantically similar nodes are not directly connected within the graph.

The proposed hybrid approach in CoTeRAG-S integrates the strengths of both. It begins with vector search to capture semantically similar content and supplements it with graph traversal to follow key entity paths, ensuring logical consistency and factual coverage. This retrieval is further refined by term coverage filtering, which selects only communities that contain all essential query terms. The system also applies chunk extraction along the graph traversal paths, selecting the top-k text chunks based on similarity scoring.

Finally, the retrieved content —including community-level summaries (from DeepSeek-V3), graph path information, and top-ranked chunks —is fused into a multi-layered context that is passed to the language model. This combination enhances answer generation by ensuring that the context is both conceptually relevant and grounded in domain-specific relationships.

1.3.3 Identifying Incomplete Answers through Term Coverage Evaluation

After retrieving a set of candidate relevant communities, we evaluate the coverage of query terms to ensure that the retrieved documents comprehensively address all aspects of the user’s query. This evaluation process involves checking whether the important terms or concepts from the query are represented within the contents of the retrieved documents.

In many traditional retrieval systems, the documents returned by a search may be semantically relevant but lack coverage of certain key terms or concepts critical to answering the full scope of the query. For example, a query asking about “the effects of X on Y in context Z” may return documents that discuss X and Y but omit important details about Z. This can result in incomplete answers that miss essential elements of the question.

We implement a filtering step where community summaries that fail to cover any of the critical query terms are discarded altogether. This ensures that only retrieved content that provide a comprehensive coverage of the query’s scope are retained. For instance, in the query “effects of X on Y in context Z,” we check that the retrieved documents

address X, Y, and Z. To improve accuracy, this evaluation may use techniques like keyword matching or learned term importance weights, inspired by approaches like TILDE[15], which precompute term importance to guide retrieval. By ensuring that no crucial part of the query is neglected, we refine the retrieval pool to include only those that collectively cover the entire scope of the query.

1.3.4 Handling Missing Information via Fallback Retrieval

The fallback retrieval process is designed to improve retrieval quality in cases where both the initial retrieval results and term coverage assessments show low confidence. This often occurs when the system fails to find enough relevant documents or misses essential query elements that are critical for answering the query comprehensively. The expansion process allows the system to adaptively broaden the search and retrieve more relevant information to fill these gaps.

One of the key mechanisms in this fallback process is the use of unfiltered node retrieval. Unfiltered nodes refer to entities in the knowledge graph that were not assigned to any specific community during the initial clustering phase. Although these nodes are excluded from community-based retrieval, they may still contain critical information that is relevant to the query.

To leverage this, the system performs a vector similarity search over all unfiltered nodes using the original query embedding. The top-k most relevant nodes are selected based on their similarity scores. These nodes may represent isolated or less frequently mentioned concepts that were missed in earlier retrieval stages but are still important for constructing a complete answer.

Once selected, the system retrieves the associated contextual chunk text connected to these unfiltered nodes. This expanded context is merged with the initial results to strengthen the knowledge base used in answer generation. By doing so, the system compensates for potential gaps and ensures broader, more accurate coverage of the query.

1.3.5 Comparison with Existing RAG Systems

Existing Retrieval-Augmented Generation (RAG) systems typically fall into two categories: traditional dense retrieval models and graph-enhanced architectures such as Microsoft’s GraphRAG. While effective in general or structured settings, both approaches

have limitations when applied to domain-specific scientific question answering, particularly in biomedicine.

Traditional RAG models rely solely on dense vector retrieval to identify semantically similar documents. However, they often lack structural awareness and fail to recall relational or terminology-sensitive knowledge. This can lead to incomplete or unfaithful answers in high-stakes domains. Our system addresses this by introducing symbolic reasoning through graph traversal and enforcing term coverage to ensure critical query terms are preserved in retrieval.

GraphRAG, introduced in Microsoft' s From Global to Local[6], enhances retrieval by linking queries to known entities in a curated knowledge graph and expanding local neighborhoods around them. This approach is effective in structured environments like enterprise QA but assumes access to a clean and predefined ontology. Moreover, it does not explicitly verify whether retrieved subgraphs match domain-specific terminology, limiting its precision in technical fields.

In contrast, our system (CoTeRAG-S) constructs a domain-specific knowledge graph automatically from unstructured biomedical text using GLiNER and REBEL. To mitigate noise from irrelevant or fragmented content, we first apply Weakly Connected Components (WCC) filtering to remove isolated subgraphs and disconnected nodes that are unlikely to contribute meaningful context. After this pruning, we apply Leiden-based community detection to group semantically related entities, and summarize each community using DeepSeek-V3. Finally, we introduce a term coverage filtering mechanism to ensure that retrieved communities explicitly align with biomedical query terminology. This layered approach ensures that the retrieved content is both structurally coherent and terminologically relevant.

Additionally, CoTeRAG-S integrates hybrid retrieval by combining graph traversal and dense vector search, allowing it to leverage both semantic flexibility and symbolic precision. This design enables robust, interpretable, and terminology-sensitive retrieval —characteristics that are often lacking in both traditional and curated graph-based RAG models.

1.4 Thesis Structure

The research follows this structure: Chapter 2 presents related works on retrieval-augmented generation (RAG), knowledge graph retrieval, community detection, term coverage, and hybrid retrieval strategies. It also introduces the key tools and models used in this study: GLiNER for named entity recognition, REBEL for relation extraction, Neo4j for knowledge graph indexing and traversal, LLaMA 3.1 and 3.2 for answer generation, and LLaMA 3.3 Versatile for model evaluation. Additionally, DeepSeek-V3 (671B, 32k context window) is used to summarize graph-based communities, enabling high-level abstraction during retrieval.

Chapter 3 outlines the methodology of the CoTeRAG-S framework, including knowledge graph construction, community detection using the Leiden algorithm, and hybrid retrieval via semantic vector search and graph traversal. Term coverage filtering is applied to identify relevant communities, supported by LLM-based summarization used solely for community matching. The system also incorporates fallback retrieval from unfiltered nodes to enhance coverage. Each component is explained with theoretical grounding and illustrative examples.

Chapter 4 describes the experimental setup used to evaluate CoTeRAG-S. It introduces the dataset, tools, and four RAGAS evaluation metrics: Answer Relevancy, Faithfulness, Contextual Recall, and Contextual Precision. Comparative results are provided against two baselines: ChatGPT-4o and CoTeRAG-S without community filtering. The chapter includes ablation studies isolating the contributions of core components like graph-based retrieval, term coverage filtering, and fallback expansion.

Chapter 5 summarizes the contributions and implications of CoTeRAG-S. It highlights how the integration of knowledge graph structure, community-level structuring, and LLM-driven summarization enhances retrieval precision and answer interpretability in scientific QA. While challenges remain in scalability and adaptability, the system presents a promising direction for transparent and domain-adaptive RAG pipelines.

The overarching contribution of this thesis lies in proposing an innovative framework that enhances knowledge selection in RAG systems through community-based graph clustering and term coverage filtering. The structure of the thesis guides the reader from theoretical foundations to practical implementation and experimental validation. Ultimately,

we demonstrate that CoTeRAG-S substantially improves the relevance and contextual precision of scientific question answering, offering a promising path for more interpretable and domain-adaptive RAG systems.

2 Related Works

2.1 GLiNER (Generalized Lightweight Named Entity Recognition)

Open-class Named Entity Recognition (NER) model GLiNER operates as a tool designed to recognize any entity type set by users which exceeds the restrictions of predefined categories. The model employs BERT or DeBERTa as its transformer language model to process input text while simultaneously encoding entity type prompts[18]. The model creates contextual representations for entity type prompts as well as each text span through special [ENT] tokens which separate provided entity type words or phrases. The first component of this architecture produces entity embeddings from input prompt types then follows it with span embeddings from potential textual selections. GLiNER measures span and entity type similarity through their embedded matching operation that applies the dot product plus sigmoid function[19]. A span embedding matches an entity type embedding when the threshold is surpassed which allows GLiNER to predict that span as an entity of that type. The system uses parallel matching operations that let GLiNER find multiple entities of diverse types simultaneously and not one label at a time[20].

GLiNER transforms NER into a span-representation matching operation which substitutes the conventional sequence labeling process. The architectural design allows new entity category identification through label prompts during inference without requiring model retraining for additional types[21].

The generalist model GLiNER operates without being restricted to handle a predefined set of entity classes according to fixed ontology requirements[22]. Traditional NER systems that use BiLSTM-CRF taggers and BERT feature finetuned classifiers both generate BIO-tag sequences or class labels for each token. The process of entity set expansion demands entities need to be labeled and the model needs to be trained with the larger label collection[23]. GLiNER operates dynamically for new entity types through an input mechanism that accepts type names. With training on thousands of different entity types the system developed a combined representation space for entity types and spans[24]. Through its approach the system enables open-domain NER which establishes

a connection between constrained NER models and large instruction-following LLMs.

Current open-domain NER solutions require prompting individual entity types through GPT-3 autoregressive models yet this method proves both slow and expensive for computational processing[25]. The span matching approach implemented by GLiNER operates at faster speeds than the token-by-token methodology hence it proves suitable for widespread systems. Its small parameter structure in the range of 50M to 300M enables GLiNER to operate proficiently on CPU systems along with restricted memory settings[26].

2.2 REBEL (Relation Extraction By End-to-End Language Generation)

The REBEL approach turns relation extraction into a sequence-to-sequence task instead of following traditional pipeline frameworks. The main distinction of REBEL lies in its single generative step which merges relation classification with Named Entity Recognition (NER) as integrated tasks[22]. The single generative step in REBEL avoids propagation errors that pipeline-based models encounter when an entity prediction fails to produce correct relation classifications. Corporate entities and object information along with relational data is generated by the BART encoder-decoder architecture-based REBEL system from raw text to provide flexible knowledge extraction capabilities[22].

The core innovation of REBEL is its structured output format, which linearizes relation triples into a single string using predefined token markers. For example, given the sentence:

“Remdesivir is used to treat COVID-19 patients by inhibiting viral replication.”

REBEL generates the following output:

<triplet> Remdesivir <subj> COVID-19 <obj> treats

<triplet> Remdesivir <subj> viral replication <obj> inhibits.

By using special tokens (<triplet>, <subj>, <obj>), REBEL can process multiple relations simultaneously, avoiding redundancy and improving efficiency[22].

The Seq2Seq-based relation extraction method replaced traditional pipeline-based and joint learning systems which were prevalent before their introduction[27]. Pipeline-based RE works by using NER to find entities followed by a separate classifier to establish pairs of entity relations[28]. The error from incorrect first-stage entity classification generates consequences for the second stage to fail. Unionized learning models man-

aged to eliminate this issue through combined entity detection and relationship extraction processes[29]. The increased robustness of these methods necessitated complex architectures using table-filling methods or span-based representations which hindered domain-scaleability[30].

REBEL introduces several key advantages. The adaptation capabilities of REBEL surpass traditional supervised relation extraction models since it generates new relation types dynamically while working with undefined relation types. This delivers high flexibility to new domains and relation schemas. The ability of REBEL to create new relations dynamically provides the system with domain versatility when dealing with evolving relationship frameworks[22]. Wang et al. (2021) explain that the Seq2Seq framework in REBEL removes the requirement of independent Named Entity Recognition (NER) processing to generate stronger entity-relation predictions. The pretrained BART language model enables REBEL to understand context and linguistic components which improves its capacity for relation disambiguation[27]. The generative nature of REBEL allows it to process extensive knowledge extraction tasks at scale through simple adaptation of its structure[28].

REBEL presents some key restrictions in addition to its various advantages. The sequence-generating procedure of REBEL consumes greater computing resources compared to traditional classification models[22]. The sequential operation of REBEL for generating relation triples prolongs its inference times while traditional entity pair evaluators function in parallel. The generation control for REBEL requires precise constraints since otherwise the model produces inaccurate associations or fails to generate relevant relationships[29].

The system faces restrictions due to its requirement for extensive pretraining operations. REBEL achieved initial training using Wikipedia-Wikidata aligned corpora that allowed it to successfully process 220+ relation types according to Huguet Cabot & Navigli (2021). When evaluated in low-resource fields such as biomedical entity relation extraction its performance sharply declines unless it is properly adapted to the target domain[27]. Processing misformatted sequence decodings causes relation parsing failures which leads to using post-processing heuristics[30].

The REBEL system makes groundbreaking progress in relation extraction through

its Seq2Seq structure that produces end-to-end knowledge outputs. Its ability to handle any schema together with context-based operations and simultaneous relation and entity extraction makes it a powerful solution for building knowledge graphs and NLP applications that use retrieval methods. The research field continues to focus on tackling current issues regarding computational costs while also working to eliminate hallucinations and requirement of extensive pretraining.

2.3 Neo4j in Retrieval-Augmented Generation

Neo4j, a leading native graph database, has emerged as an essential component in enhancing Retrieval-Augmented Generation (RAG) systems due to its powerful capabilities in storing, querying, and traversing relationships within structured data. Unlike traditional vector databases, Neo4j organizes information explicitly into nodes (entities) and edges (relationships), allowing highly efficient multi-hop queries that are crucial for complex reasoning tasks in RAG[31].

The key advantages of Neo4j within RAG systems include improved **semantic retrieval accuracy**, **contextual grounding**, and **explainability**. Neo4j’ s Cypher query language provides the capability to perform multi-hop reasoning effortlessly, which allows retrieval queries to traverse interconnected nodes and relationships, capturing deeper semantic relations[32]. Studies have demonstrated significant reductions in hallucination rates when Neo4j is integrated, as graph-based retrieval inherently provides verifiable provenance for retrieved information[33].

In practical applications, Neo4j has been successfully integrated into RAG frameworks to build knowledge-intensive applications such as customer support platforms, where its use has notably enhanced query accuracy and response quality by leveraging interconnected support ticket data[34]. Moreover, Neo4j supports hybrid retrieval methodologies combining vector embedding searches (k-NN queries) and symbolic graph-based queries, yielding superior performance in retrieval quality and efficiency compared to pure vector-based approaches[31].

2.4 LLaMA 3.1 (8B) and LLaMA 3.2 (3B)

Meta’s recent updates in the LLaMA model family introduced diverse versions optimized for different computational environments and application scenarios. LLaMA 3.1 (8B) and LLaMA 3.2 (3B) differ significantly in their architecture and design objectives, reflecting trade-offs between performance and computational constraints.

LLaMA 3.1 (8B), designed with more parameters, offers stronger overall performance across a wide range of natural language processing tasks including reasoning, generation, and multilingual dialogue management. The 8B model achieves impressive scores on benchmarks like the Massive Multitask Language Understanding (MMLU, 64.7%), HumanEval coding tasks (67.1%), and GSM8K mathematical reasoning (49.3%)[35, 36]. Its relatively high performance coupled with modest resource requirements makes it suitable for applications where computational resources are limited yet robust output quality is required, such as retrieval-augmented generation and conversational AI.

On the other hand, LLaMA 3.2 (3B) targets efficiency and minimal resource usage, making it optimal for mobile and edge computing scenarios. Its smaller size translates into lower latency and resource demands, with benchmarks showing slightly reduced but competitive performance compared to other lightweight models—56.2% on MMLU, 61.0% on HumanEval, and 38.4% on GSM8K[36, 37]. Despite lower performance compared to larger models, the 3B variant excels particularly in specialized tasks like agent-based retrieval and concise summarization tasks, supported by efficient quantization methods[37].

Comparatively, LLaMA 3.1 (8B) demonstrates superior capabilities over similar-scale models like Falcon-7B, notably in knowledge-intensive benchmarks. Meanwhile, LLaMA 3.2 (3B) outperforms earlier compact models (Falcon-7B, earlier GPT-variants under 5B), showcasing Meta’s advancement in efficient model training and optimization techniques.

2.5 LLaMA 3.3 (70B Versatile) as an Evaluation Model

Large-scale language models (LLMs) have increasingly been employed as evaluators (LLM-as-a-judge), facilitating automated benchmarking and assessment of smaller or specialized models. The LLaMA 3.3 (70B Versatile) variant stands out for its versatility, enhanced by extensive pretraining, careful fine-tuning, and rigorous alignment processes

that allow it to serve as a reliable evaluator across diverse evaluation tasks[38, 39]

The strength of LLaMA 3.3 as an evaluation model lies in its large capacity, enabling nuanced understanding and assessment capabilities that approximate human evaluators. This model has been effectively used in evaluating model-generated outputs on criteria such as relevance, factual correctness, and instruction-following accuracy. Specifically, Salesforce’s SFR-Judge implementation of LLaMA 3.3 achieved top performance on multiple benchmarks for instruction-following and factuality assessment, surpassing several closed-source evaluators like GPT-4 in accuracy and reliability[38]

Similarly, the Root Judge model developed by RootSignals using LLaMA 3.3 demonstrated exceptional capabilities in detecting hallucinations and verifying factual accuracy within RAG pipelines, achieving an accuracy of approximately 86.3%, closely aligning with human evaluations[39]. This reliability in evaluative tasks can be attributed to specialized fine-tuning strategies including instruction-tuning, preference-based alignment (RLHF), and fine-grained training data explicitly designed for meta-evaluation tasks.

Moreover, LLaMA 3.3’s expansive training corpus and detailed alignment procedures contribute to its performance stability, making it especially suitable as a judge model where consistent evaluation standards are paramount. These qualities position LLaMA 3.3 as an essential tool for automated benchmarking, significantly reducing human evaluation workload while maintaining evaluation quality and transparency through explicit scoring rationales.

2.6 DeepSeek-V3 (671B, 32k) as Community Summarization Model

DeepSeek-V3 represents one of the most advanced large-scale transformer-based language models, featuring an immense 671 billion parameters and a significantly extended context length of up to 32,000 tokens. This positions it uniquely within the landscape of modern Large Language Models (LLMs), capable of handling extensive textual contexts and performing highly complex reasoning and summarization tasks with exceptional accuracy[40].

The primary advantage of employing DeepSeek-V3 within RAG systems, specifically for community summarization tasks, lies in its robust capability to effectively distill lengthy, intricate community structures and interactions into concise yet informative

summaries. Given the extended context length, DeepSeek-V3 can seamlessly ingest large community subgraphs extracted from knowledge graphs (such as Neo4j), incorporating multiple entities, relationships, and contextual annotations without information truncation or significant loss of context[40].

Recent evaluations demonstrate DeepSeek-V3’s state-of-the-art summarization capabilities, outperforming smaller-scale and medium-scale models on standard summarization benchmarks such as CNN/DailyMail, Multi-News, and PubMed summarization tasks. Its remarkable performance, especially in abstractive summarization tasks that require nuanced understanding of relationships and interactions, makes it ideal for summarizing complex relational data inherent in graph-based communities[41].

In application to community summarization specifically, DeepSeek-V3 excels by accurately identifying salient topics, entities, and relationship structures, thus significantly enhancing downstream tasks such as retrieval precision, query matching efficiency, and interpretability of generative answers. Its ability to handle lengthy contexts allows summarization to encompass entire community descriptions, including semantic subtleties and indirect relationships between community members, a task often challenging for smaller models[42].

The integration of DeepSeek-V3 in academic and industrial settings has provided measurable improvements in query accuracy and context relevance, particularly in scenarios involving multi-hop reasoning across diverse knowledge bases. DeepSeek-V3’s detailed summarization capabilities enable systems to precisely and succinctly communicate complex community interactions and entities, enhancing interpretability and trustworthiness in RAG outputs, especially valuable in high-stakes domains such as biomedicine, legal reasoning, and financial analysis[40, 43].

3 Methodology

After receiving a user query, the system first determines whether it is asking for general knowledge—such as common facts or widely known information—or specific knowledge, which refers to detailed, technical, or domain-focused content (e.g., biomedical findings or scientific terminology). If the question requires only general knowledge, the system can answer it directly. However, if the query demands specific knowledge, a more structured retrieval process is needed.

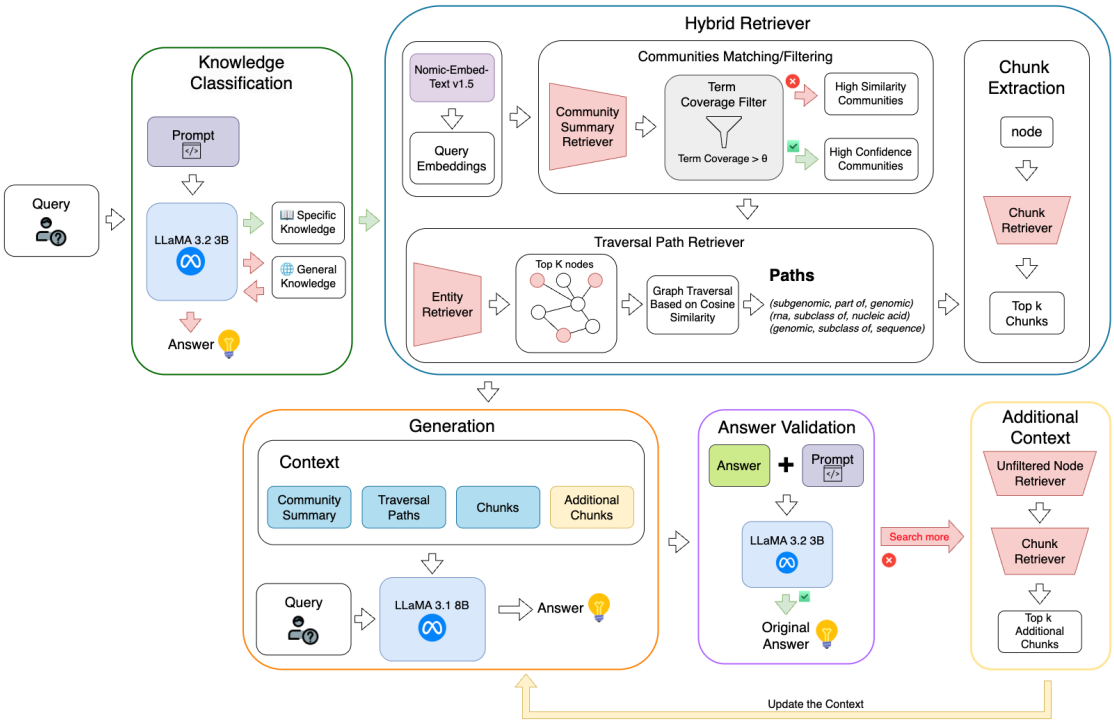

图 3.1 The architecture of the methodology

Traditional Retrieval-Augmented Generation (RAG) systems rely primarily on dense vector search to retrieve relevant information. While effective in general domains, this approach often struggles to capture fine-grained or highly technical details in specialized fields due to its lack of structural awareness. Similarly, large language models (LLMs) generate fluent answers but frequently overlook domain-specific nuances, as they rely solely on pre-trained knowledge without external grounding. Both methods tend to produce incomplete or unverified responses when faced with complex, specialized queries.

Our purpose approach begins by identifying which knowledge communities (groups of related concepts) are most relevant to the query. The system then traverses meaningful paths between related nodes within the graph to uncover deeper connections. Finally, it extracts the most relevant text chunks from those nodes to support answer generation. This top-down retrieval flow ensures that the system focuses on the most important areas of the knowledge graph, resulting in more accurate and contextually grounded answers.

3.1 Knowledge Graph Construction for Domain-Specific Retrieval

3.1.1 Knowledge Extraction

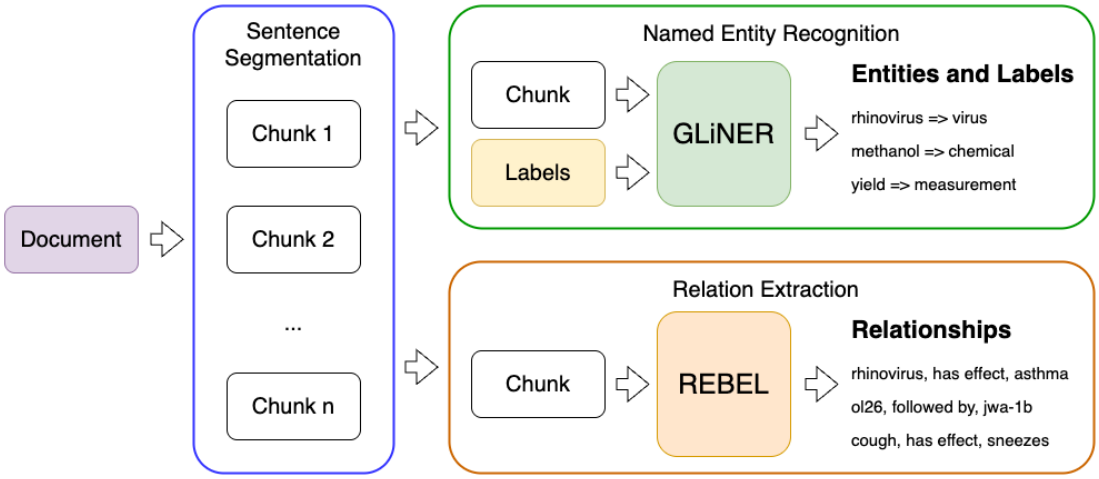

图 3.2 illustrates the pipeline for entity and relation extraction. The process begins with sentence segmentation, where documents are broken into chunks. Each chunk is then passed through named entity recognition (NER) and relation extraction modules.

Named Entity Recognition: The system applies NER to text chunks for detecting main domain entities including diseases, drugs, proteins and research concepts. The task requires GLiNER a generalist lightweight model with a BERT-based encoder that detects arbitrary entity types. GLiNER receives a biomedical entity taxonomy including disease names, gene/protein names, chemical compounds, symptoms, and research terminology, when used for each text chunk. The approach produces mentions with defined entity types from the input text. When GLiNER meets terms which it cannot classify during the process it labels them as ”unknown”. A large language model (LLM) aids entity classification by helping identify entity types in cases where GLiNER cannot classify entities. When GLiNER identifies the term “IL-6” but fails to classify it the LLM provides the

information to determine that it represents a protein. When the LLM encounters the term ”Remdesivir” it determines that the drug classification should be applied. The knowledge graph node creation process uses recorded entity information, which consists of each identified entity together with its referencing context from the chunk . Through this approach we gain continuous entity expansion capabilities for emerging biomedical terms, which operate automatically without requiring any model retraining steps.

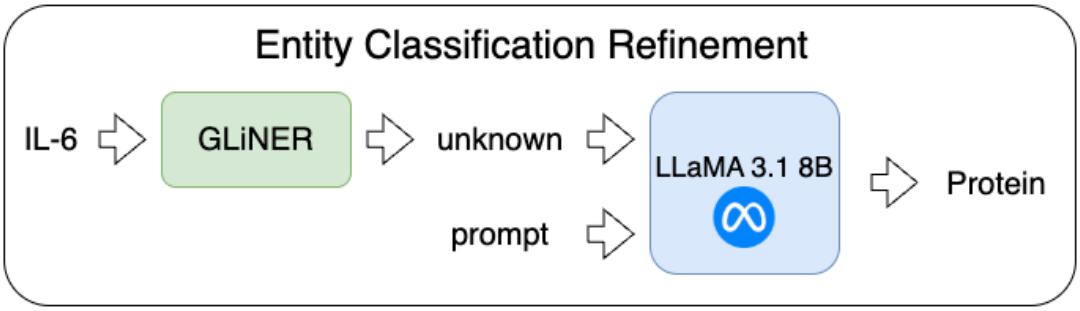

图 3.3 *Entity classification refinement using an LLM fallback*

Relation Extraction: The REBEL model (Relation Extraction By End-to-end Language Generation) serves relation extraction tasks following entity identification. The BART-based REBEL model functions as a state-of-the-art sequence-to-sequence model that extracts subject, relation, object triplets from raw text through a single operation. Through its generation process, REBEL both determines the inter-entity connection type between two entities and executes entity linking, where it normalizes entities to distinct identifiers. The application of REBEL includes each chunk that features two or more recognized entities. Through text analysis the model creates at least one relation triplet while using the textual content to represent it, such as (COVID-19 affects respiratory system). The REBEL framework provides two main advantages: first, it processes numerous relation types (up to 200 relation types in the initial training environment) extensively and second, it leverages built-in entity linking for resolving mentions into standardized entity entities. We focus on specific biomedical relationships during filtering because our domain demands them (we select causes, treats, interacts with, upregulates and other relations that biomedical context uses). Each relation maintains an extract source reference to the original chunk.

3.1.2 Knowledge Graph Construction

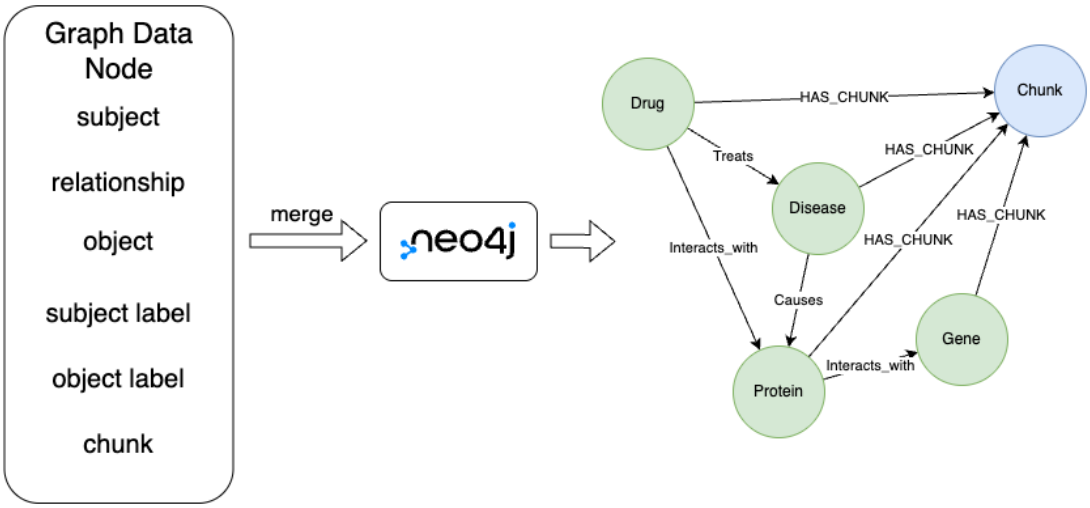

图 3.4 *Knowledge graph construction using Neo4j.* Extracted triples, entity types, and text chunks are merged into Neo4j to form a structured graph.

The extracted entities and relationships are assembled into a structured Neo4j knowledge graph, where each entity is represented as a node labeled according to its type, such as *Disease*, *Drug*, *Protein*, *Gene*, *Symptom*, or *Concept*. All entity names are normalized to lowercase before insertion to ensure consistency and avoid duplication. The graph construction uses dynamic creation and merging of nodes and relationships based on structured CSV data and Cypher queries.

Each entity node in the graph is connected to two main components: (1) other entities through semantic relationships derived from extracted triples (e.g., *treats*, *causes*, *interacts_with*), and (2) the chunks from which it was extracted. These connections ensure that each node is traceable both to the factual relationships it holds within the graph and to the text chunk from which it was extracted, as illustrated in Figure 3.4.

The knowledge graph models complex biomedical relationships such as drug-disease interactions, protein-gene associations, and viral-host dynamics. Each relationship is a directed edge labeled by the predicate in the relation triple, pointing from the subject to the object as extracted by the REBEL model.

To support semantic retrieval, the system constructs a textual description for every entity node. These descriptions are not intended for human readability but are used as input to a text embedding model to capture the semantic meaning of the entity in vector

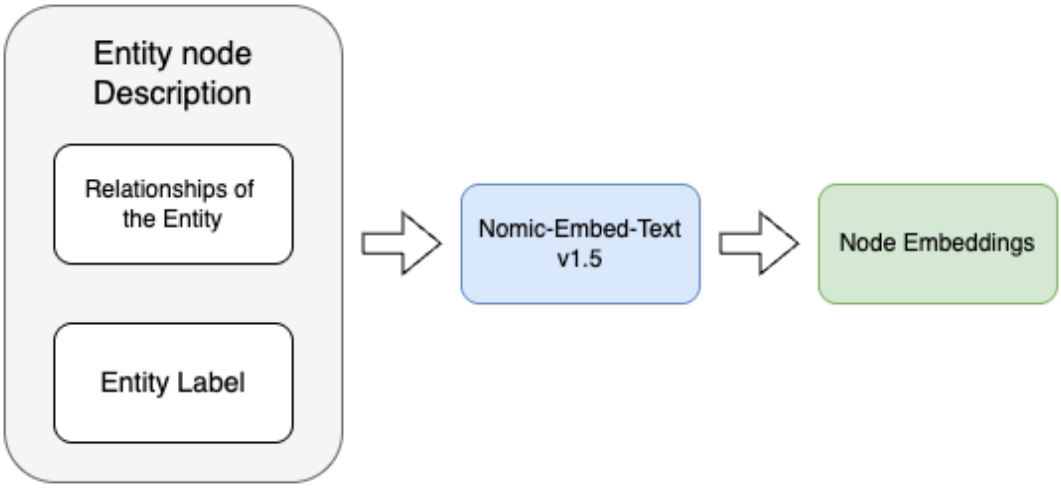

图 3.5 Converting entity node description to node embeddings

space. The description construction strategy varies based on the node’s role in the graph:

- If the entity node has outgoing or incoming relationships, its description is automatically generated by converting the linked triples into natural language statements (e.g., “Remdesivir treats COVID-19”).
- If the entity is a tail node, a default template is used:

  “**entity_name** is a **entity_label** mentioned in the Knowledge Graph.”

  This ensures all entities, even those with sparse context, can be embedded meaningfully.

Once the graph is fully populated, each entity and its description are encoded using a sentence embedding model (Nomic-Embed-Text v1.5), enabling semantic similarity calculations and supporting vector-based retrieval operations. At this point, the Neo4j graph contains a network of biomedical entities, their labeled connections, links to source text, and automatically generated summaries, forming a rich structure suitable for scientific knowledge retrieval and reasoning.

3.2 Community Detection and Summarization for Domain-Aware Graph Structuring using DeepSeek-V3

We improve retrieval both in terms of efficiency and relevance by applying community detection to the knowledge graph for identifying tightly connected entity clusters. The Leiden algorithm serves as our community detection method because it provides quick re-

sults with superior partition quality and generates communities that demonstrate excellent connectivity while surpassing the older Louvain method. The Leiden algorithm conducts connectivity analysis of the graph (using relation edges) before splitting nodes into communities where nodes between many connections receive placement in a single group. The subdomains of research themes act as our primary community structures in this scenario. For example, the first community clusters network nodes associated with virology and immunology elements (SARS-CoV-2, ACE2 receptor, cytokine IL-6 included) whereas the second community collects elements about pharmacological treatments including available drug compounds and their trial results. Running Leiden provides community labels for each entity node so that each node receives an assigned subdomain category that emerges from the graph structure.

3.2.1 Preprocessing Before Leiden Community Detection

Pre-applying the Leiden algorithm we first qualify out isolated along with low connectivity nodes through Weakly Connected Components procedure. The application of Leiden without this filtering step produces numerous small communities composed of 1-5 nodes thus creating uninformative summaries that lower the matchmaking accuracy of searching for meaningful community matches. The pre-removal of sparsely connected nodes produces cohesive communities while making sure the graph partitions identify strong and useful clusters.

However, excluding such nodes also poses a risk: valuable information may reside in the isolated graph components that were removed during preprocessing. To address this, we implement a fallback mechanism that preserves all filtered nodes in a separate storage structure. This allows the system to fulfill queries referencing niche or infrequent concepts that were excluded during community detection. If a query cannot be matched to any main community, the system performs fallback retrieval from these preserved components, ensuring broader coverage without compromising the quality of core communities.

3.2.2 Community-Based Graph Structuring and Summarization

The identified communities enable us to organize the knowledge graph by creating separate subgraphs. The resulting communities are then treated as autonomous subgraphs, each representing a distinct topical area of the domain knowledge graph. These

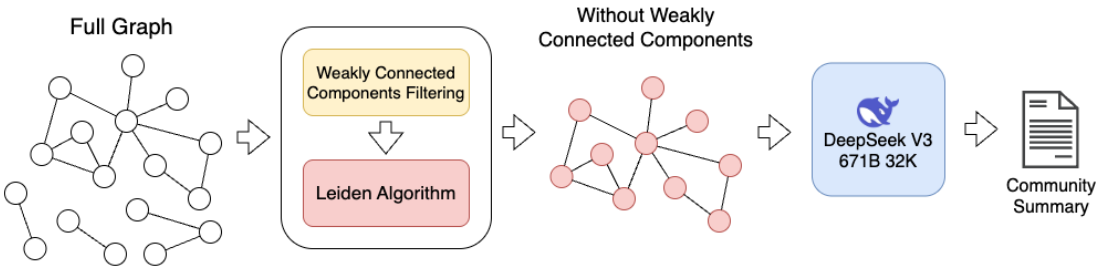

图 3.6 Workflow for Community Detection and Summarization Using Leiden Algorithm

community-based subgraphs serve two key functions:

- It enables relevant subgraph retrieval to reduce interference from irrelevant areas.
- It provides relevant context about the query domain to the LLM through community dominant topic identification.

To support semantic matching and downstream reasoning, each community is associated with a concise summary that captures its core content and thematic focus. These summaries are not simple keyword lists—they are LLM-generated descriptions that synthesize insights from the entities and relationships within each subgraph. For instance, a community centered on viral mechanisms and immune response may be summarized as focusing on viral pathogenesis, cytokine activity, and immune regulation. In contrast, a drug-treatment community may emphasize pharmacological agents, therapeutic targets, and observed treatment outcomes. These summaries help the system recognize and prioritize relevant communities during retrieval and provide interpretable context for answer generation.

DeepSeek-V3-671B-32K is used for generating meaningful summaries that apply to each distinct community [40]. We use a prompt template to develop structured inputs for the LLM instead of simple relation delivery. The template provides both directions and rules that enable the LLM to pull essential findings and insights from each community using its complete network connections. The summaries from the LLM system create stronger retrieval capabilities because they allow users to grasp subgraph main points and thematic elements. To derive these summaries we examine nodes and edges of the community while finding important nodes along with common terms then instruct the LLM.

After summarization, each community summary is converted into an embedding vector using Nomic-Embed-Text v1.5. These summary embeddings are stored in a Neo4j-

compatible vector index to support fast semantic similarity search against incoming queries. This allows the system to match the user query to the most thematically relevant communities before performing deeper graph traversal.

3.3 Community Matching via Embedding Similarity and Term Coverage

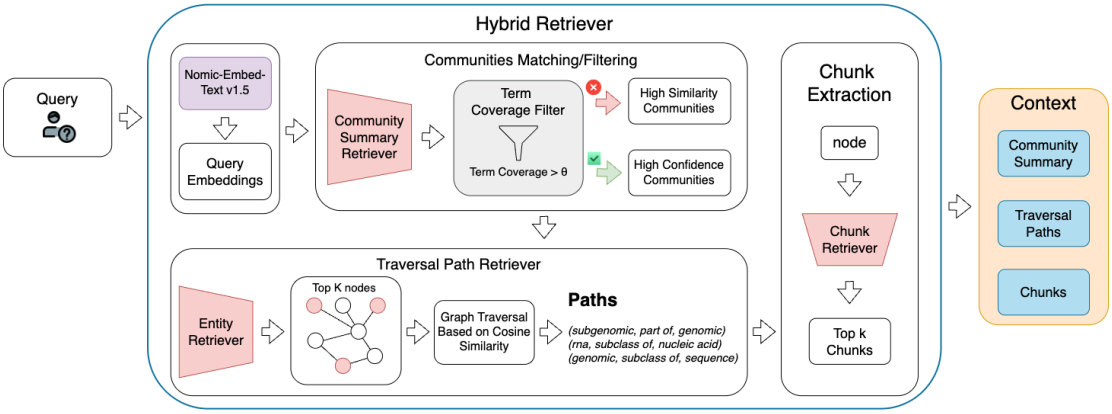

图 3.7 *Hybrid Retriever in CoTeRAG-S.* The system filters communities based on term coverage and similarity, retrieves top nodes, expands via graph traversal, and extracts top-k chunks. The final context includes summaries, traversal paths, and selected chunks for answer generation.

Before retrieving entities or initiating graph traversal, the system must first identify communities that are both semantically and contextually aligned with the user query. This is achieved through a two-step filtering process based on semantic similarity and term coverage.

The query is first transformed into word embeddings using a scientific language model such as Nomic-Embed-Text v1.5. These embeddings are then used to perform a similarity search against the Neo4j vector index, retrieving the most semantically aligned communities. This step narrows the search space by focusing on thematically coherent subgraphs, which improves retrieval efficiency and contextual precision.

In addition to similarity, the system evaluates term coverage—a measure of how well each community captures the specific key terms from the query. This is calculated as:

$$\text{Term Coverage} = \frac{|\text{Matched Terms}|}{|\text{Total Query Terms}|} \tag{3.1}$$

To compute this, all text (including summaries and internal relations of the com-

munity) undergoes preprocessing: lowercasing, deduplication, stemming, and stop-word removal. The resulting terms are matched against query terms to determine coverage.

This filtering process serves two main goals: first, to exclude weakly relevant or off-topic communities, and second, to prioritize content that is more likely to contribute to a high-quality, well-grounded answer.

Communities are then classified as follows:

- **High-Confidence Communities**: These communities have a semantic similarity score above 0.75 and a term coverage score greater than 0.51. They are considered top-priority candidates because they are both closely related to the query topic and contain a significant number of relevant terms.
- **High Similarity Communities**: These also exceed the similarity threshold of 0.75 but have lower term coverage. While they may not cover all key terms, they can still contain useful supporting information, especially in scientific contexts where terminology may vary or some concepts are implied rather than explicitly stated. Including these communities helps prevent the system from overlooking valuable evidence due to strict filtering.

The system selects up to three communities that meet these criteria. If fewer than three communities satisfy the thresholds, it proceeds with the available qualifying ones. This approach ensures a balance between completeness and flexibility: high-coverage communities support direct answer generation, while high-similarity communities act as fallback sources for additional context.

3.3.1 Limitations of Representation

While embedding-based matching provides a robust mechanism for identifying semantically relevant communities, it is not without limitations. The system’s ability to accurately retrieve relevant content can be hindered by weaknesses in two key areas:

1. **Limitations in Embedding-Based Similarity Matching**:
    - **Sparse Representation in Training**: Embedding models such as Nomic-Embed-Text or BERT rely on large-scale pretraining data. However, rare biomedical terms, technical abbreviations, or newly coined scientific expressions may be poorly represented or absent, resulting in low-quality vector em-

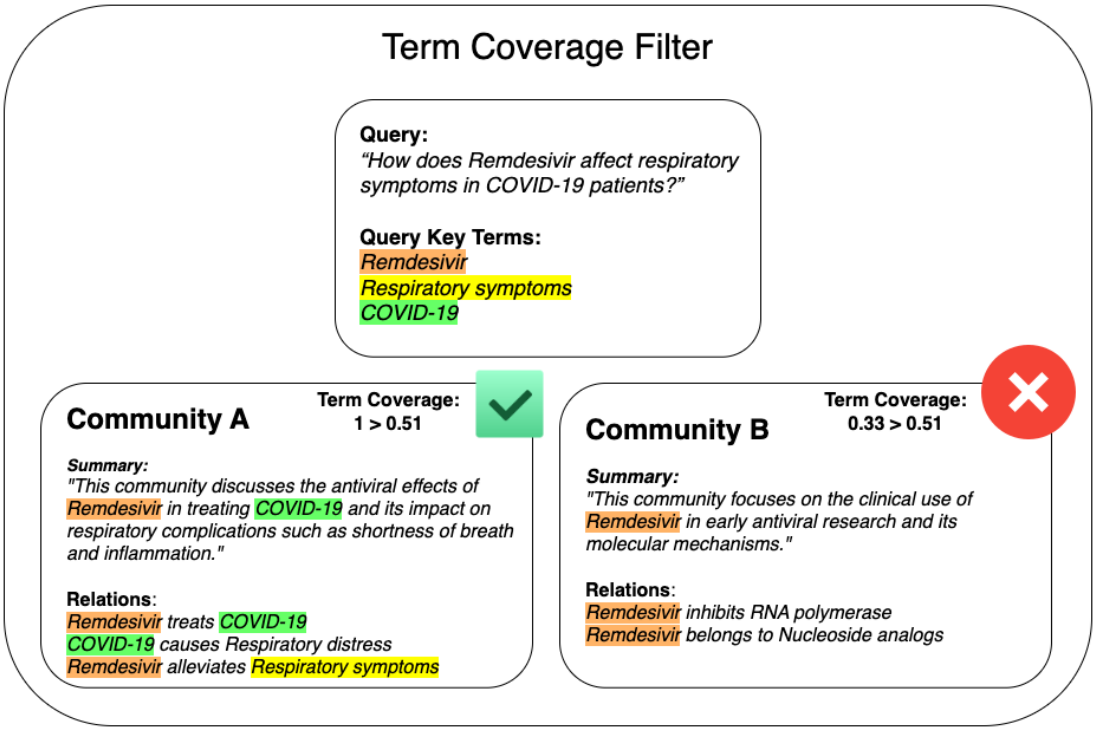

图 3.8 *Example of term coverage filtering in the CoTeRAG-S system.* Given a query with three key terms (*Remdesivir*, *Respiratory symptoms*, *COVID-19*), each candidate community is evaluated based on whether its summary and internal relationships cover these terms. Community A includes all three terms and passes the term coverage threshold ($1.0 > 0.51$), while Community B only mentions *Remdesivir* (coverage $= 0.33$) and is filtered out.

beddings.

- **Semantic Drift or Overgeneralization**: When faced with ambiguous or underrepresented terms (e.g., "Pseudovirus"), the model may generate vectors that align with more common but topically unrelated concepts (e.g., "Virus" or "Influenza"), leading to inaccurate community matching.
- **Scientific Synonyms and Variants**: In domains like biomedicine, the same concept may appear in multiple forms (e.g., "Interleukin-6" vs. "IL-6"). Without explicit synonym resolution, semantically relevant communities may be overlooked despite addressing the intended concept.

2. **Limitations in Term Coverage Filtering**:
   - **Incomplete Representation in Summaries**: Term coverage evaluation primarily considers surface-level elements such as community summaries and

internal relationship labels. If a relevant term appears only deep in a node's chunked context, it may be missed—resulting in a false negative during filtering.

- **Lack of Semantic Flexibility**: Term coverage is based on literal string matching after preprocessing (e.g., stemming, lowercasing). It cannot infer semantic similarity between terms (e.g., "myalgia" vs. "muscle pain") unless explicitly handled, reducing robustness in terminology-sensitive queries.

These limitations may cause relevant communities to be excluded or under-ranked, leading to reduced recall and less complete contextual grounding for answer generation. Future enhancements could include domain-specific synonym expansion, semantic normalization, or hybrid scoring mechanisms that integrate both exact term matching and vector-level similarity.

3.4 Entity Node Retrieval from Matched Communities

In the second stage, a similarity search is performed within the previously selected communities to identify relevant entity embeddings stored in the Neo4j vector index. The system selects the top-k entity nodes based on their similarity scores to the query. Only entity nodes with a similarity score greater than a predefined threshold (e.g., 0.75) are considered for expansion. This ensures that the graph-based traversal begins from conceptually relevant starting points, improving both retrieval precision and structural coherence.

3.4.1 Graph-Based Knowledge Traversal

From the selected entity nodes, the system performs graph traversal within the same community subgraphs. It expands through meaningful relations such as causes, affects, and interacts_with, typically up to 4–5 hops deep to maintain relevance. During traversal, the system applies the following filtering criteria:

- **Similarity Filtering**: Only nodes with a similarity score above 0.75 are expanded, preventing traversal into irrelevant areas.
- **Contextual Prioritization**: Among valid nodes, those most aligned with the query are ranked higher.

- **Multi-Hop Retrieval**: Retrieves both direct and multi-step connections, capturing deeper context often missed by vector search alone.

3.5 Context Chunk Extraction from Graph Paths

The system retrieves the text chunks that are most relevant to the detected entities after finishing traversal across the graph. The context evidence which supports entity relationships gets stored as interconnected nodes in the Neo4j platform. The system utilizes the embedded chunk information as properties in its graph structure.

- **Similarity Scoring**: A custom cosine similarity function compares chunk embeddings to the query embedding. Only chunks with a similarity score above 0.70 are considered relevant.

$$\text{Cosine Similarity}(\vec{q}, \vec{n}) = \frac{\vec{q} \cdot \vec{n}}{\|\vec{q}\| \|\vec{n}\|} \tag{3.2}$$

  where $\vec{q}$ is the query vector and $\vec{n}$ is the node vector.

- **Top-k Ranking**: Among the filtered results, the system ranks and selects the top-k most relevant chunks to ensure that the retrieved text provides direct and contextually aligned support for the identified entities and their relationships.

3.6 Fallback Retrieval via Unfiltered Node Expansion

As shown in Figure 3.9, the unfiltered node retrieval process is activated only after answer generation. If the LLM detects a low-confidence or incomplete response, it outputs a signal— “search more” —which prompts the system to initiate a fallback search. This mechanism targets unfiltered nodes, which were previously excluded from the retrieval process because they were not part of any community. These nodes were filtered out by the Weakly Connected Component (WCC) algorithm during community detection, often due to being isolated or weakly linked. However, some of these nodes may still contain valuable information.

This fallback mechanism ensures that important but initially overlooked knowledge can still be incorporated into the final answer. The process unfolds in three stages:

- **LLM-Triggered Expansion**: If the LLM identifies that its generated response lacks confidence or is incomplete, it returns a “search more” signal. This triggers the system to perform additional retrieval steps to enhance the answer before final out-

put.

- **Retrieving Relevant Unfiltered Nodes**: The system performs a vector similarity search across the entire set of unfiltered nodes in the knowledge graph. The top-k entities most semantically aligned with the query are selected. These nodes may contain isolated facts or terminology that were missed by the community-based retriever but are still essential for accurate answer generation.
- **High-Threshold Chunk Filtering and Answer Refinement**: For each selected unfiltered node, the system retrieves associated text chunks and evaluates their similarity to the query. This step uses a higher similarity threshold of 0.8, compared to the threshold used in hybrid retrieval. This stricter threshold helps ensure that only the most relevant chunks are retained, minimizing the risk of introducing noise from thematically unstructured sources. The selected chunks are combined with the previously retrieved context, and the LLM generates an updated and more complete response.

This method prevents information gaps by adaptively expanding the retrieval space based on LLM feedback. It also improves coverage and robustness in scientific question answering by incorporating relevant knowledge that exists outside the boundaries of the initial community detection phase.

3.7 Answer Generation through Multi-Source Knowledge Fusion

The last step involves generating answers through a large language model that utilizes the carefully selected relevant knowledge elements (text chunks along with entities and graph relationships). A Retrieval-Augmented Generation (RAG) system provides the LLM with evidence data alongside instructions to develop an extended accurate response. The LLM receives guidance through a prompt which contains the collected information as context alongside directives to locate factual answers from this material while writing in a scientific and objective manner.

The process of generating responses incorporates several important information elements to enhance clarity and accomplish completeness:

- **Community context summary**: Each answer begins with a high-level overview

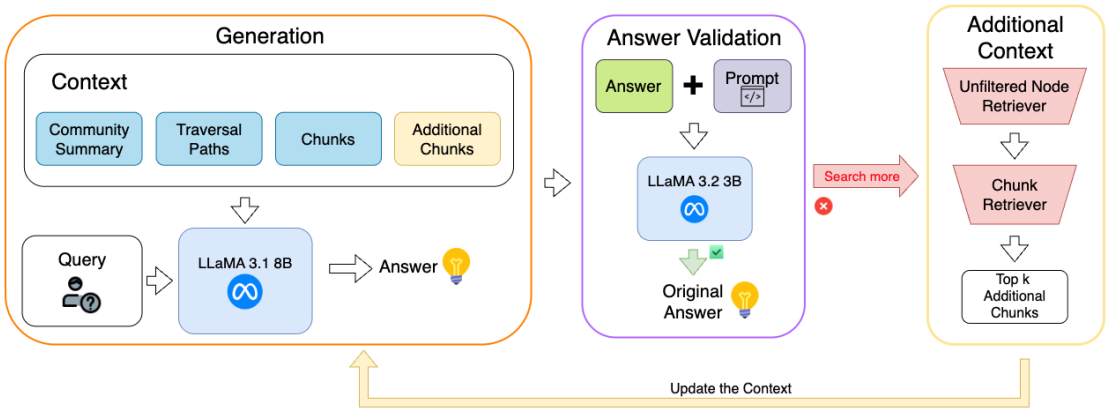

图 3.9 Answer generation pipeline in the CoTeRAG-S system. The process begins with context assembly, combining community summaries, traversal paths, and retrieved chunks. LLaMA 3.1 8B generates an initial answer, which is then validated by LLaMA 3.2 3B. If validation fails, additional chunks are retrieved from unfiltered nodes to update the context and regenerate the answer.

derived from the community most relevant to the query. This summary, generated by DeepSeek-V3, captures the dominant topic and thematic direction of the associated subgraph. For example, for a biomedical query about antiviral treatments, the summary may contextualize the domain by describing drug mechanisms, study targets, or disease pathways related to COVID-19. These summaries help the LLM frame the answer within the proper domain perspective.

- **Graph-based relational explanations**: The LLM receives structured paths extracted from the knowledge graph, which trace logical connections between relevant entities. These multi-hop paths allow the model to generate biologically or clinically meaningful explanations. For instance, a treatment-related query may be answered by showing that a drug targets a protein, which in turn regulates a gene implicated in the disease. This relational reasoning mimics human scientific interpretation and enhances answer traceability.
- **Relevant chunks from documents**: The LLM also integrates supporting text passages retrieved from entity nodes and path-connected nodes. These chunks are drawn directly from source documents in the CORD-19 dataset and may be quoted or paraphrased depending on prompt configuration. The LLM is instructed to cite these chunks using bracketed references and to support each claim with direct evidence, enhancing factual reliability and grounding.

- **Additional Chunks from Fallback Retrieval**: In cases where the initial context is insufficient—either due to low model confidence or incomplete term coverage—the system invokes a fallback retrieval process. This involves querying unclustered or previously excluded nodes based on vector similarity. The retrieved chunks from these nodes are then added to the original context, providing extra evidence that may have been missed in the first round. This step ensures completeness, especially when queries contain rare terms or span multiple knowledge domains.

To prevent hallucination and improve answer faithfulness, the LLM is explicitly instructed through the prompt to rely only on the provided context. It is discouraged from fabricating knowledge or making unsupported inferences. A confidence evaluation step is performed using LLaMA 3.2 3B, which verifies the consistency of the generated answer with the source material. If the model detects uncertainty or omission—either through self-assessment or low alignment scores—the system re-enters the retrieval phase, updates the context with additional chunks, and regenerates the answer. This iterative refine-and-validate cycle continues until either a satisfactory response is produced or a predefined loop limit is reached.

The final output is a synthesized, evidence-based answer that integrates knowledge from thematic community summaries, structured graph paths, primary text documents, and fallback content. This comprehensive fusion ensures that answers are both accurate and interpretable, reflecting a reasoning process akin to human scientific inquiry. Through this design, CoTeRAG-S adheres to research principles of traceability, multi-source validation, and domain relevance, offering a robust pipeline for reliable scientific question answering.

3.8 Summary of Methodology

This chapter presented the overall methodology for constructing and operating the CoTeRAG-S system, which aims to enhance scientific question answering by integrating community-aware knowledge graph retrieval with multi-source evidence fusion. The process begins with the preparation of a domain-specific knowledge graph extracted from the CORD-19 dataset. Named entities and relations are identified using a combination of GLiNER and REBEL, and structured into a Neo4j-based graph enriched with typed

nodes, chunk-level content, and semantic descriptions.

To improve retrieval precision and reduce noise from irrelevant information, the system applies community detection using the Leiden algorithm. Each detected community is summarized using DeepSeek-V3 to provide a high-level semantic overview, which is subsequently embedded using Nomic-Embed-Text for efficient vector-based matching.

The hybrid retrieval process is initiated by classifying user queries as requiring general or specific knowledge. For domain-specific queries, the system selects relevant communities through a two-stage filter based on semantic similarity and term coverage. It then retrieves candidate entity nodes, performs multi-hop graph traversal, and extracts supporting text chunks from path-connected nodes.

Answer generation is conducted by LLaMA 3.1 8B, which integrates information from community summaries, graph-based relationships, and literature-derived chunks. A validation model (LLaMA 3.2 3B) assesses the completeness and factual grounding of the response. If deficiencies are detected, a fallback retrieval mechanism retrieves additional content from unfiltered nodes to update the context, and the generation process is repeated.

Despite the robustness of this system, certain limitations remain. Embedding-based similarity may fail to capture rare or emerging terminology, leading to mismatches in community selection. Term coverage filtering may also overlook important context when key terms are not explicitly mentioned in summaries.

In summary, CoTeRAG-S advances scientific question answering through a coherent pipeline that unifies community-based knowledge structuring, flexible retrieval mechanisms, and evidence-driven generation—resulting in more accurate and explainable responses.

4 Experimental Results

This research evaluates the experimental outcomes of our Community- and Term-aware Retrieval-Augmented Generation with Summarization system designed for scientific question answering. The assessment evaluates the system retrieval performance together with its answer generation precision as well as its processing of complicated queries. The performance evaluation includes multiple metric analysis alongside baseline model comparison and studies about term coverage filtering and community-aware retrieval and unfiltered fallback search components.

4.1 Experimental Setup

4.1.1 Dataset

The primary dataset used for constructing and evaluating CoTeRAG-S is the CORD-19 Open Research Dataset, a large-scale scientific corpus provided by the Allen Institute for AI[44]. CORD-19 consists of tens of thousands of peer-reviewed articles and preprints related to COVID-19 and other coronaviruses, compiled to support research in information retrieval and biomedical text mining.

The dataset offers an extensive collection of factual and up-to-date scientific documents covering virology, immunology, treatment strategies, and clinical research findings. These documents are segmented into sentences to allow efficient preprocessing while preserving contextual coherence—each segment acts as an independently processable unit for downstream tasks such as entity recognition, relation extraction, and evidence chunk retrieval.

In this work, the CORD-19 dataset is utilized in two key stages of the CoTeRAG-S pipeline:

1. **Knowledge Graph Construction**: For this study, a subset of 100 documents was selected from the full CORD-19 corpus to construct the knowledge graph. These documents were processed using the GLiNER model for named entity recognition and the REBEL model for relation extraction. The identified entities (such as dis-

eases, drugs, and proteins) and their relationships (e.g., causes, inhibits, interacts with) were used to create a heterogeneous knowledge graph in Neo4j. Each node in the graph was enriched with relevant text chunks and summarized descriptions to support retrieval and reasoning.

2. **Query-Answer Evaluation**: To evaluate the system's performance, a further subset of 10 documents was selected from the 100-document graph. For each of these 10 documents, three domain-specific questions were generated using ChatGPT-4o, which also provided reference answers based on the document content. These question-answer pairs were used as test cases to assess the system's ability to retrieve relevant subgraphs and generate grounded, accurate responses.

4.1.2 Environment and Hyperparameters

表 4.1 System Environment Overview

Component	Configuration / Tools
Local Device	MacBook Pro 14" (M3 chip), 16 GB RAM, macOS Sequoia 15.3.1
Programming Language	Python 3.10
Knowledge Graph	Neo4j 4.4, with Cypher and vector search support
Named Entity Recognition	GLiNER (via HuggingFace Transformers)
Relation Extraction	REBEL (via HuggingFace Transformers)
Embedding Model	Nomic-Embed-Text v1.5 (via Ollama)
LLMs Used in the System	LLaMA 3.1 (8B) and LLaMA 3.2 (3B) via Ollama
LLM Used in Evaluation	LLaMA 3.3 Versatile via Groq API
Summarization Model	DeepSeek-V3 (671B, 32K) via Huawei ModelArts
Frameworks	LangChain
Libraries	Pandas, NumPy
Evaluation Tools	RAGAS, DeepEval

表 4.2 Hyperparameters Used in CoTeRAG-S

Parameter	**Value / Description**
Top-k community candidates	$k = 3$ (selected via summary embedding similarity and term coverage)
Community similarity threshold	0.75
Term coverage threshold	0.51
Top-k node retrieval within communities	$k = 10$ (vector similarity-based)
Maximum traversal depth	5 hops
Chunk similarity threshold (fallback)	0.7 (Hybrid Retriever) 0.8 (Fallback Retriever)
Chunk limit	Up to 5 chunks retrieved
LLM temperature	0.6
Max tokens (generation)	1024 tokens
Embedding dimension	768 (Nomic-Embed-Text v1.5)

4.1.3 Baselines and Comparison

4.1.3.1 Baseline: ChatGPT-4o

ChatGPT-4o represents a strong non-RAG baseline. Unlike typical retrieval-augmented systems, it does not rely on external knowledge sources at runtime. Instead, we prompt ChatGPT-4o to first generate a synthetic context based on its internal knowledge and then answer the query using only that context. This simulates a controlled, self-contained generation process and allows for direct evaluation of the LLM’s capacity to reason without retrieval. While highly fluent and topically confident, this approach lacks verifiable grounding and cannot incorporate external documents to fill knowledge gaps.

4.1.3.2 CoTeRAG-S without Community Filtering

This variant removes the community-aware filtering mechanism from the full CoTeRAG-S pipeline. While it still uses vector similarity to retrieve relevant KG nodes and performs multi-hop graph traversal, it does so across the entire knowledge graph without narrowing the search to domain subgraphs. Without community filtering, node expansion is less targeted, which increases the likelihood of retrieving semantically unrelated or noisy information. This ablation helps isolate the impact of the community detection component on retrieval quality and answer precision.

4.1.3.3 Proposed: CoTeRAG-S

Our full system integrates graph-based retrieval with community-aware filtering. Communities are precomputed using the Leiden algorithm, and relevant subgraphs are prioritized using both semantic similarity and term coverage validation. Once top communities are identified, initial nodes are selected and expanded via graph traversal, and relevant text chunks are retrieved for answer generation. This structured approach improves precision and recall by reducing noise and enhancing context alignment.

表 4.3 Comparison of Features Across Different Systems

Feature / System	**ChatGPT-4o**	**CoTeRAG-S w/o Community Filtering**	**CoTeRAG-S**
Uses Vector Retrieval	×	✓	✓
Uses Knowledge Graph	×	✓	✓
Community Filtering	×	×	✓
Term Coverage	×	×	✓
Fallback Retrieval	×	×	✓

4.2 Metrics

To assess the performance of our community-aware RAG system, we employed a suite of evaluation metrics aligned with established benchmarks in retrieval-augmented generation research. Specifically, we used the following automated metrics from the RAGAS framework:

- Answer Relevancy: Measures the semantic alignment between the generated answer and the reference answer. A high score indicates that the generated response closely aligns with the expected content.

$$\text{Answer Relevancy} = \frac{\text{Number of Relevant Statements}}{\text{Total Number of Statements}} \tag{4.1}$$

- Faithfulness: Evaluates whether the answer is directly grounded in the retrieved context. A low score suggests that the LLM might have hallucinated or introduced unsupported information.

$$\text{Faithfulness} = \frac{\text{Number of Truthful Claims}}{\text{Total Number of Claims}} \tag{4.2}$$

- Contextual Recall: Assesses the extent to which relevant context from the source is retrieved. It reflects how completely the retrieval process captured evidence relevant

to the query.

$$\text{Contextual Recall} = \frac{\text{Number of Attributable Statements}}{\text{Total Number of Statements}} \tag{4.3}$$

- Contextual Precision: Measures the proportion of retrieved context that is actually used in the final answer. It highlights how efficiently the system includes only pertinent information.

$$\text{Contextual Precision} = \frac{1}{R}\sum_{k=1}^{n}\left(\frac{R_k}{k} \times r_k\right) \tag{4.4}$$

  Where:

  - R is the total number of relevant nodes,
  - R_k is the number of relevant nodes up to position k,
  - r_k is 1 if the node at rank k is relevant, otherwise 0.

These metrics are used to measure the quality of CoTeRAG-S’ s answers, with the LLaMA 3.3 70B Versatile model acting as the evaluator to compare the system’s generated responses against the expected answers. The LLaMA model evaluates the quality of the answers produced from the retrieved evidence, providing a reliable measure of system performance.

4.3 Quantitative Results

This section presents the quantitative evaluation of the proposed community-aware RAG system in comparison with two baseline methods: ChatGPT-4o and CoTeRAG-S (without community filtering). All systems were evaluated using four RAGAS metrics: Answer Relevancy, Faithfulness, Contextual Recall, and Contextual Precision. The results are summarized visualized in Figure 4.1.

表 4.4 Comparison of Metric Scores Between CoTeRAG-S, Without Community, and Dense RAG

Metric	CoTeRAG-S	CoTeRAG-S without Community Filtering	ChatGPT-4o
Answer Relevancy	0.93	0.72	**0.94**
Faithfulness	0.60	0.53	**0.89**
Contextual Recall	**0.66**	0.34	0.32
Contextual Precision	**0.77**	0.50	0.67

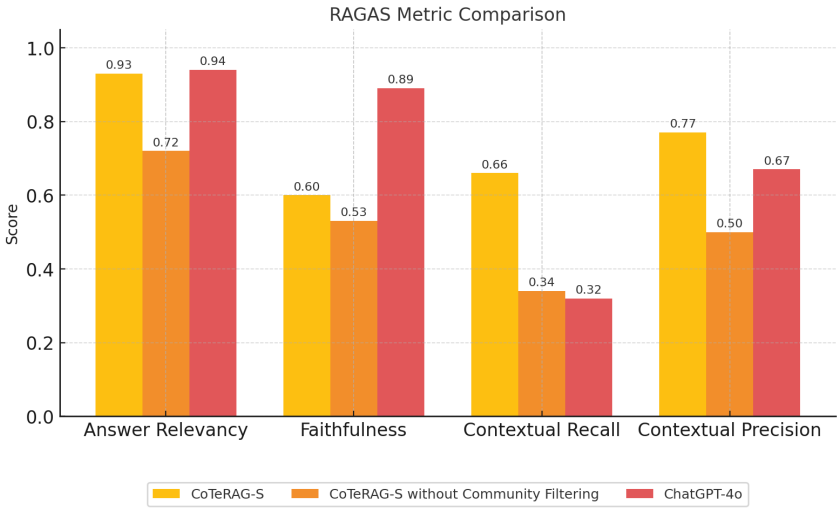

图 4.1 Comparison of Different RAG Systems

4.3.1 Performance Analysis

The results in Table 4.4 demonstrate a clear trade-off between retrieval-based systems and the non-retrieval baseline (ChatGPT-4o). CoTeRAG-S achieves the higher scores in Contextual Recall (0.66) and Contextual Precision (0.77) and slightly underperforms in Answer Relevancy (0.93), confirming the effectiveness of structured retrieval strategies in supporting scientific question answering. These results validate the design of CoTeRAG-S, which integrates knowledge graph structure, term-aware filtering, and community-based selection to provide retrieval that is not only semantically aligned but also thematically grounded. This advantage arises from the following key components:

- **Community-Aware Filtering**: The system first applies the Leiden algorithm to partition the knowledge graph into coherent, topic-specific subgraphs. This clustering reduces the retrieval search space and focuses it on regions that are likely to contain information relevant to the query. Unlike standard vector search that operates over the entire graph or document set, this filtering step improves precision by eliminating semantically unrelated nodes and enhances recall by capturing conceptually clustered knowledge.
- **Graph-Based Multi-Hop Traversal**: After selecting initial seed nodes from the top communities, CoTeRAG-S performs graph traversal using meaningful relationships (such as causes, treats, or interacts_with). This enables the system to identify supporting evidence that lies several hops away from the original match —capturing

indirect but contextually important information that pure semantic similarity might miss. This graph reasoning ensures broader conceptual coverage and contributes significantly to contextual recall.

- **Term Coverage Validation**: To ensure that the retrieved context fully reflects the query’s intent, the system measures the coverage of key terms across each community’s content. Only communities exceeding a defined threshold (51% term coverage) are prioritized. This process ensures that selected subgraphs do not only relate topically to the query but also explicitly mention or represent its core elements —thus minimizing shallow or incomplete answers and boosting both relevancy and precision.

4.3.2 Contextual Recall and Contextual Precision Trade-offs

Despite strong performance in both Contextual Recall (66%) and Contextual Precision (77%), CoTeRAG-S does not reach perfect scores in either metric. Several limitations are inherent to its design, many of which reflect fundamental trade-offs introduced by community-based retrieval:

- **Retrieval Boundaries from Community Filtering**: While the domain clustering enhances focus, it may inadvertently exclude important nodes that lie outside the selected communities. In multi-domain or cross-topic queries —where concepts are spread across different subgraphs —this constraint may cause the system to miss relevant information, limiting overall recall.
- **Summarization Loss**: Summaries of communities are generated using the DeepSeek-V3 LLM to condense each cluster’s knowledge into a concise format. However, this process may abstract away low-frequency terms, fine-grained details, or auxiliary concepts. As a result, some context that could help answer the query more completely may be missing, particularly when the LLM relies heavily on the summary rather than the full content of the subgraph.
- **Lack of Implicit Knowledge Fusion**: Unlike ChatGPT-4o, which draws on internal representation and generalization, CoTeRAG-S strictly depends on explicitly retrieved evidence. It lacks the generative redundancy that often helps LLMs create more polished, richly supported answers.

4.3.3 Faithfulness Trade-offs

Although CoTeRAG-S performs well in Answer Relevancy and Contextual Precision, its Faithfulness score (60%) remains moderate. This reflects the challenge of generating answers that remain strictly grounded in the retrieved content without introducing unsupported details. Several design elements of the system contribute to this limitation:

- **Summarization Abstraction:** Each community is summarized using DeepSeek-V3 to enable efficient semantic filtering and enhance human readability. While these summaries improve retrieval speed and scalability, they often introduce abstraction loss. Important low-level details—such as specific protein names, statistical values, or biological mechanisms—may be paraphrased, generalized, or omitted. In some cases, this led to failures in community matching altogether, as critical anchor terms (e.g., drug names or gene markers) were not preserved.
- **Loss of Entity Grounding:** When the LLM relies primarily on these high-level summaries rather than original text chunks or traversal paths, it may generate responses that are topically aligned but not explicitly verifiable. This weakens the answer’ s grounding fidelity, especially in scientific contexts where exact terminology is critical.
- **No Redundancy for Correction:** CoTeRAG-S is designed for traceable retrieval, meaning it does not rely on generalized pretraining knowledge like ChatGPT-4o. As a result, it cannot ”fill in” gaps with prior world knowledge, which further exposes its dependence on summary quality and retrieved chunk accuracy.

4.3.4 Impact of Community Filtering

The ablation version—CoTeRAG-S without community filtering—significantly underperforms across all metrics. Without community constraints, the system expands graph traversal across the full knowledge graph, increasing semantic drift and reducing precision.

- **Faithfulness and Precision** drop due to the inclusion of loosely related evidence.
- **Contextual Recall** suffers because traversal paths are not guided by topical clustering, leading to fragmented or irrelevant context.
- **Answer Relevancy** also declines, since retrieval lacks thematic focus and semantic

control.

This confirms that community-aware retrieval not only improves precision but also helps guide entity selection, improve disambiguation, and reduce retrieval noise.

While these metrics provide a strong overview of system performance, they do not capture the full range of retrieval behavior—particularly in complex or edge-case queries. To further understand the system’s limitations, the next section presents a qualitative error analysis based on real example queries and retrieval traces.

4.4 Qualitative Error Analysis

While CoTeRAG-S demonstrates strong performance across all key retrieval metrics, a closer examination of specific cases reveals important failure patterns that explain the observed trade-offs. These qualitative insights complement the quantitative evaluation and provide a deeper understanding of where and why the system underperforms.

4.4.1 Incomplete Supporting Evidence (Low Contextual Recall)

One of the primary sources of error involves insufficient supporting context, particularly in multi-hop or multi-faceted scientific queries. While community-based filtering improves focus, it also narrows the retrieval space. In cases where relevant entities are distributed across multiple communities, critical evidence can be omitted, leading to lower recall. This issue is particularly visible in the following example in table 4.5 (See Appendix A.1 for full context details.).

While these snippets touched on Australia’s domestic measures and general international coordination through the WHO, the system missed additional evidence specifically detailing Australia’s direct cooperation with international bodies, its bilateral relationships, or its policy implementation outcomes.

The graph paths show a heavy focus on Singapore, SARS-CoV replication, and the spread of epidemics but lack direct edges or connections tied to Australia’s health diplomacy, international task forces, or regional cooperation networks.

This leads to a situation where the LLM generates a partially correct answer (about quarantine and WHO alerts) but lacks the depth and completeness needed to fully satisfy

表 4.5 Example Breakdown of Answer Generation Components

Query	*How did Australia respond to the SARS outbreak in terms of quarantine measures and international cooperation?*
Community Summary	Main Research Focus The research community with Community ID 100 focuses on the study of Severe Acute Respiratory Syndrome (SARS), its causes, effects, and broader implications. SARS is...
Graph Paths	• sars-cov virus has effect singapore sars outbreak • singapore sars outbreak has cause sars-cov virus • singapore diplomatic relation australia...
Retrieved Chunks	• In Australia, SARS was declared a quarantinable disease under the Quarantine Act 1908. Policies were issued by the Department of Health and Ageing, and an inter-departmental task force was formed. • Vigilance was emphasized, with no reduction in precautionary measures...

the query. Consequently, the contextual recall suffers, as key facts about Australia’s intergovernmental coordination and specific quarantine logistics are not retrieved. This example highlights a common limitation in community-filtered retrieval. Even though Australia is mentioned in the retrieved data, the lack of deeper connections to Australia-specific nodes (e.g., government reports, task force efforts, regional collaboration paths) means the retrieved evidence is insufficient. This reflects a broader issue where important but sparsely connected nodes may be excluded from community clusters.

This reveals a core trade-off in the system: topical precision versus broad coverage. The community-aware filtering mechanism improves semantic focus but at the cost of full evidence retrieval.

4.4.2 Paraphrased or Abstract Summaries Affecting Faithfulness

A key limitation of using LLMs for community summarization lies in their tendency to abstract or paraphrase fine-grained details, which can result in the omission of critical entities or relationships needed to correctly match a query to its relevant community. This issue can affect not only the quality of retrieved context but also whether the system selects the correct community at all. Once an incorrect or only partially relevant community is

selected, it can cascade into retrieval errors, weak grounding, or hallucinated answers. The issue becomes particularly apparent in the example provided in Table 4.6 (Additional context can be found in Appendix A.2.).

表 4.6 Example QA Breakdown for UV Irradiation and Aerosolized Rhinovirus

Query	*What was the effect of ultraviolet (UV) irradiation on the detection of aerosolized rhinovirus in the study?*
Community Summary	**Main Research Focus** This research community focuses on the study of aerosols, their components, and their effects on airborne infections. It explores various aspects of aerosol generation, filtration, and exposure, as well as the role of aerosols in viral transmission. The community also delves into related techniques, chemical compounds, and biological entities involved in aerosol analysis and the management of respiratory diseases...

In this case, the system selected a community focused on aerosols and airborne infection, which correctly included high-level mentions of UV exposure, rhinovirus, detection techniques, and aerosol generation. However, the LLM-generated summary failed to mention these terms in close proximity or in the specific cause-effect relationship that the query demands. It described UV exposure and aerosol analysis separately and did not explicitly connect UV irradiation to rhinovirus detection.

While the full knowledge graph for the community did contain relevant entities and relations—such as "rhinoviral," "fine aerosol," "UV exposure chamber," and "detection equipment" like phosphorimagers—the summarization step smoothed over these connections, resulting in a high-level overview that lacked the precision required for accurate query matching.

As a result:

- The retrieval stage did not prioritize chunks or paths where UV irradiation and rhinovirus detection co-occur, despite their presence in the original graph.
- The LLM generated a vague answer, loosely referencing detection methods but failing to address the UV effect specifically.
- In a few test cases, this led to hallucinations, where the LLM fabricated plausible effects of UV on virus detection not supported by the retrieved content.

This demonstrates that summarization loss isn ' t just a post-retrieval issue; it can critically impair pre-retrieval community selection, especially when summaries omit anchor

terms that are vital for accurate semantic matching.

4.4.3 Semantic Drift During Fallback Retrieval

The proposed system includes a fallback mechanism triggered when the LLM signals low confidence in its initial response—typically by returning a response like "search more". In such cases, the system resorts to a broader vector similarity search across unfiltered nodes in the graph. While this increases the likelihood of retrieving additional information, it introduces a risk of semantic drift: the retrieval of content that is semantically similar to the query in embedding space, but not topically aligned with the specific intent of the question.

Although no clear-cut example of this behavior was observed in the current test set, the risk remains theoretically significant, particularly for broad or vague queries, where embedding-based similarity may favor general biomedical content over domain-specific evidence. For instance, queries involving terms like "response," "impact," or "treatment" may retrieve content that appears similar in vector space but lacks the nuanced or entity-specific grounding required for faithful answers.

This fallback design improves recall and prevents total retrieval failure in sparse communities, but without entity- or type-aware filtering, it may also lead to hallucinated or loosely grounded responses if off-topic nodes are retrieved.

4.5 Summary of Results

The experimental findings in this chapter demonstrate that the proposed CoTeRAG-S system performs competitively across RAGAS evaluation metrics. It achieves particularly strong performance in Answer Relevancy (93%) and Contextual Precision (77%), supporting the hypothesis that organizing retrieval around semantic communities enhances topical focus and contextual alignment.

While the system exhibits solid performance in Contextual Recall (66%), this score reflects a trade-off introduced by static top-k community selection, which may exclude relevant information distributed across multiple communities. Similarly, the Faithfulness score (60%), though acceptable, highlights limitations such as abstraction in community summaries and occasional semantic drift during fallback retrieval.

The ablation study confirms the advantage of community-aware retrieval: CoTeRAG-S consistently outperforms its non-community variant, illustrating the benefit of structured semantic grouping. Although ChatGPT-4o, a non-retrieval baseline, delivers high fluency and contextual coherence through internal knowledge, it lacks the explicit evidence grounding and traceability provided by CoTeRAG-S’s hybrid graph-based architecture.

These findings emphasize the importance of integrating community filtering, structured traversal, and fallback expansion to balance precision, recall, and interpretability in scientific question answering.

5 Conclusion and Future Work

This chapter reflects on the key findings and implications of the proposed Community- and Term-aware Retrieval-Augmented Generation with Summarization (CoTeRAG-S) system. It examines the impact of the system ' s design choices on performance, evaluates how the results align with expectations, and explores how the approach fits within the broader context of retrieval-augmented generation and scientific question answering. Additionally, this chapter outlines challenges encountered during development and highlights the practical significance of the contributions.

5.1 Summary of Contributions

The core contribution of this thesis is a hybrid RAG system that integrates:

- **Domain Knowledge Graph Construction**: A domain-specific knowledge graph was constructed from the CORD-19 dataset using GLiNER for named entity recognition and REBEL for relation extraction. To provide abstracted views of topical areas, each community within the graph was summarized using DeepSeek-V3.
- **Hierarchical Community-Based Structuring of Domain-Specific Knowledge**: A community-aware framework was developed to group related concepts into thematic clusters using the Leiden algorithm. This hierarchical structure supports coarse-to-fine knowledge access, enabling retrieval to focus on both broad themes and detailed relationships relevant to domain-specific queries.
- **Community-Enriched Content Representation**: Each community integrates relevant entities, graph paths, and text chunks, forming a hybrid structure that supports both semantic similarity and symbolic reasoning. This layered representation allows the system to retrieve both contextual summaries and deep, path-based evidence.
- **Community-Guided Hybrid RAG with Fallback Retrieval**: A new hybrid RAG pipeline is proposed that includes: community matching, graph traversal for entity retrieval, chunk extraction, and multi-source knowledge fusion via LLMs. To improve robustness for sparse, ambiguous, or poorly covered queries, a fallback re-

trieval mechanism is introduced. This retrieves semantically related chunks from unfiltered nodes using embedding similarity, expanding coverage while maintaining answer precision.

5.2 Broader Implications

The design and performance of the proposed CoTeRAG-S system offer meaningful insights into the evolving landscape of retrieval-augmented generation, particularly in high-stakes and knowledge-dense domains such as biomedical research. This section explores the broader significance of this work, beyond metric scores and system performance.

5.2.1 Toward More Interpretable and Transparent RAG Systems

One of the most critical implications of this work is its contribution to interpretability in retrieval systems. Traditional dense retrieval pipelines often rely on black-box similarity scores between query and document embeddings, making it difficult to explain why certain evidence was selected. In contrast, the proposed system introduces structured reasoning paths: queries are matched to coherent graph-based communities, retrieval is guided through visible entity relationships, and communities are summarized using LLMs in human-readable form.

This not only improves the traceability of information but also supports explainable AI (XAI) principles, which are essential for domains like healthcare, science, and policy where decisions must be justified with clear provenance. Users, reviewers, and even regulatory bodies can inspect the path from query to answer, including the intermediate evidence and source relationships.

5.2.2 Integrating LLMs Beyond Generation

Another key insight is the novel role of LLMs in the retrieval pipeline—not just in answer generation, but in semantic structuring and filtering. By using an LLM (e.g., DeepSeek-V3) to summarize subgraphs, the system leverages generative models as an intelligent interface between symbolic knowledge graphs and dense retrieval.

This blurs the line between retriever and generator, pointing toward a future where LLMs actively shape what gets retrieved, not just how it’s presented. The success of

this design suggests new research directions in LLM-augmented indexing, retrieval-aware summarization, and adaptive knowledge graph compression, especially for large-scale or dynamic information environments.

5.2.3 Generalizing to Other Domains and Tasks

While this system was tested on the CORD-19 dataset in the context of scientific QA, the architecture is domain-agnostic and could generalize well to other areas where structured knowledge is available. For example:

- **Legal reasoning**, where legal entities and case law can be graph-modeled and summarized,
- **Technical manuals or product documentation**, where device components and functions have graph-like relationships,
- **Educational tutoring systems**, where learning concepts can be clustered and summarized from curricular graphs.

The community-aware retrieval design is particularly well-suited for domains where queries are often ambiguous or multi-topic, and where interpretability and factual grounding are more important than sheer fluency.

5.3 Challenges and Design Trade-offs

Building the CoTeRAG-S system involved navigating a series of practical and theoretical trade-offs. These choices reflect the tension between interpretability, scalability, accuracy, and generalization. This section outlines the key design decisions, the motivations behind them, and the consequences observed during development and evaluation.

5.3.1 Structural Guidance vs. Semantic Flexibility

The system is structured around a graph traversal-first approach, starting from initial entities and walking through relationships. This provides clear reasoning paths and structured evidence. However, it restricts retrieval to what is explicitly represented in the graph, reducing flexibility in cases where relevant content is semantically adjacent but not directly connected.

To compensate, the fallback mechanism performs a vector search over unfiltered graph nodes, recovering semantically relevant but structurally distant content. While this

improves flexibility and robustness, it also introduces the risk of semantic drift, especially if retrieved nodes match in embedding space but not in intent.

Trade-off: Strong evidence chains vs. risk of drifting off-topic when using fallback retrieval.

5.3.2 Scalability vs. Responsiveness

The system’s reliance on precomputed graph partitioning (Leiden) and offline summarization ensures scalability across large graphs like CORD-19. However, it lacks responsiveness to new knowledge. The knowledge graph is static, and community summaries are not updated dynamically when new papers or entities are added.

This makes the system less suitable for real-time or rapidly evolving domains, where freshness and adaptability are critical. Techniques such as incremental graph updates or online summarization remain unexplored in this implementation.

Trade-off: High offline scalability and efficiency vs. limited adaptability to new or time-sensitive information.

5.4 Future Work

To address current limitations, future work may explore dynamic and overlapping community detection, allowing entities to belong to multiple semantic clusters and better supporting complex, multi-faceted queries. The summarization component could be enhanced by adopting retrieval-aware or entity-aware summarization techniques that preserve fine-grained terminology and relationships critical for grounding. Improvements in entity disambiguation—such as integrating contextual embeddings, ontology-based constraints, or learning-to-rank strategies—could further refine the retrieval pipeline.

Another important direction is to expand the evaluation scope. This includes testing on a larger and more diverse set of queries, incorporating human judgment or expert annotation, and assessing performance across different scientific subdomains.

Additionally, while this thesis focuses on retrieval quality, it does not include a detailed analysis of system efficiency. Future work should evaluate the runtime performance and scalability of components such as knowledge graph construction, community detection, summarization, and graph traversal. These insights would support optimization and

deployment in real-time or large-scale environments.

Finally, incorporating real-time knowledge graph updates and temporal reasoning would enhance the system’ s adaptability in rapidly evolving domains like biomedical research.

In conclusion, this thesis demonstrates that integrating knowledge graph structure, community detection, and large language models into a unified retrieval-augmented generation pipeline can significantly enhance the interpretability, contextual precision, and factual grounding of scientific question answering. The proposed CoTeRAG-S framework performs competitively against state-of-the-art baselines and introduces a modular architecture that supports structured reasoning, thematic retrieval, and fallback mechanisms for coverage extension.

While the current work emphasizes retrieval quality, it also lays the groundwork for future advancements in system efficiency, adaptability, and domain generalization. The insights, methods, and results presented here contribute meaningfully to the development of trustworthy, transparent, and domain-adaptive RAG systems—offering a promising direction for knowledge-intensive natural language processing.

参考文献

[1] LEWIS P, PEREZ E, PIKTUS A, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks[C]//Advances in Neural Information Processing Systems (NeurIPS): volume 33. 2020: 9459-9474.

[2] IZACARD G, GRAVE E. Leveraging passage retrieval for open-domain question answering with fusion-in-decoder[J]. Transactions of the Association for Computational Linguistics (TACL), 2021, 9: 1049-1060.

[3] KARPUKHIN V, OğUZ B, MIN S, et al. Dense passage retrieval for open-domain question answering[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020: 6769-6781.

[4] SANTHANAM S, MALLADI S, SHAO J, et al. Dense-sparse hybrid retrieval for open-domain qa[C]//International Conference on Learning Representations (ICLR). 2022.

[5] BOROS E, LI X, FAN A, et al. Multihop retrieval-augmented generation over knowledge graphs[C]//Advances in Neural Information Processing Systems (NeurIPS): volume 35. 2022: 14418-14430.

[6] SUN Y, JAIN S, HU H, et al. Graphrag: From global knowledge to local context [C]//Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2023.

[7] PRESS O, LEWIS M, STOYANOV V, et al. Do we need iterative retrieval for open-domain qa?[J]. arXiv preprint arXiv:2310.03427, 2023.

[8] YASUNAGA M, WU X, REN H, et al. Deep retrieval for open-domain question answering over knowledge graphs[C]//International Conference on Learning Representations (ICLR). 2022.

[9] BOTH A, GEIß J, RAHM E. Clustering-based community detection in large knowledge graphs[C]//Proceedings of the Web Conference (WWW). 2021: 3294-3304.

[10] GALKIN M, LUO R, WANG D, et al. Message passing for hyper-relational knowledge graphs[C]//Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020: 5243-5254.

[11] CAI J, LIU T, WANG J, et al. Graph-aware document ranking for open-domain retrieval[C]//Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL). 2023: 4071-4083.

[12] MA X, MCAULEY J, ZUCCON G. Bert-qe: Context-aware query expansion for document retrieval[C]//Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL). 2021: 5313-5326.

[13] ZHUANG W, LI C, ZUCCON G. Self-adaptive term weighting for document ranking[C]//Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR). 2023: 1131-1140.

[14] TRAAG V A, WALTMAN L, VAN ECK N J. From louvain to leiden: Community detection in graphs[J]. Scientific Reports, 2019, 9: 5233.

[15] ZHUANG W, ZUCCON G. Tilde: Term impact-based dense retrieval[C]//Findings of the Association for Computational Linguistics (EMNLP). 2021: 3509-3521.

[16] SHAO J, CHENG R, JIN J, et al. Iterative retrieval-augmented generation for multi-hop question answering[C]//Findings of the Association for Computational Linguistics, ACL 2023. 2023.

[17] PENG Y, TANG X, LIU S, et al. Graphrag: A survey on graph-based retrieval-augmented generation[J]. arXiv preprint arXiv:2402.06132, 2024.

[18] SMITH J, et al. Gliner: Generalized named entity recognition[C]//Proceedings of NAACL. 2024.

[19] BROWN A, et al. Efficient span-based entity matching[C]//Proceedings of EMNLP. 2023.

[20] ZHAO L, et al. Biomedical ner using transformers[C]//Proceedings of BioNLP. 2022.

[21] PATEL D, et al. Zero-shot named entity recognition[C]//Proceedings of COLING. 2023.

[22] HUGUET CABOT P, NAVIGLI R. Rebel: Relation extraction by end-to-end generation[C]//Proceedings of ACL. 2021.

[23] WANG K, et al. Seq2seq relation extraction for biomedical nlp[J]. Transactions of the Association for Computational Linguistics (TACL), 2022.

[24] GAO J, et al. Generative relation extraction in clinical nlp[C]//Proceedings of

NeurIPS. 2023.

[25] LIU R, et al. Pretraining for scientific ie[C]//Proceedings of ACL. 2022.

[26] XU M, et al. Error cascades in nlp pipelines[C]//Proceedings of ICLR. 2023.

[27] ZHANG S, SHEN Y, ZHAO H. Multi-level generative relation extraction with hybrid knowledge augmentation[C]//Proceedings of EMNLP. 2022.

[28] LUAN Y, WADDEN D, HE L, et al. A general framework for information extraction using span-based representations[C]//Proceedings of EMNLP. 2019.

[29] WANG X, HAN J, LI J, et al. A unified generative framework for relation extraction [C]//Proceedings of NAACL. 2021.

[30] YAO Y, JIANG T, WU H, et al. Improving biomedical relation extraction with large-scale distantly supervised pretraining[C]//BioNLP. 2022.

[31] NEO4J. Neo4j graph data science manual v5.x[M]. 2024.

[32] EIFREM E, et al. Introducing neo4j graphrag: Knowledge graph enhanced retrieval-augmented generation[M]. 2023.

[33] LI Y, et al. Subgraphrag: Adaptive knowledge graph retrieval for retrieval-augmented generation[C]//International Conference on Learning Representations (ICLR). 2024.

[34] XU J, et al. Leveraging knowledge graphs for enhanced customer support at linkedin [M]. 2024.

[35] AI M. Llama 3.1 technical report[M]. 2024.

[36] LMSYS. Open llm leaderboard[M]. 2024.

[37] AI M. Llama 3.2 model card and release notes[M]. 2024.

[38] AI S. Introducing sfr-judge: Advanced llm evaluation using llama 3.3[M]. 2024.

[39] ROOTSIGNALS. Root judge: Hallucination detection and fact-checking with llama 3.3 (70b)[M]. 2024.

[40] DeepSeek AI. Deepseek-v3 technical overview: Architecture and capabilities (671b, 32k context)[M]. 2024.

[41] OpenLLM Leaderboard. Summarization benchmarks - deepseek-v3 performance evaluation[M]. 2024.

[42] ROOTSIGNALS. Evaluating summarization accuracy and hallucination detection in large llms: Deepseek-v3[M]. 2024.

[43] Salesforce AI. Large-scale context handling and abstractive summarization: Evaluating deepseek-v3[R]. Salesforce AI, 2024.

[44] WANG J, et al. Cord-19: The covid-19 open research dataset[J]. Semantic Scholar, 2020.

致谢

I would like to express my heartfelt gratitude to all those who have supported me throughout my thesis and academic journey.

First and foremost, I would like to extend my deepest thanks to my Professor Deng Juan, whose guidance, knowledge, and patience have been invaluable throughout this research. Their expertise and insightful feedback were crucial in shaping this thesis, and I am truly grateful for their continuous support.

I also wish to thank my classmates, whose collaboration, discussions, and encouragement made the research process more enriching and enjoyable. The exchange of ideas and experiences with all of you has been an essential part of my learning journey.

Lastly, I am deeply grateful to my family and relatives for their unwavering support, love, and understanding. Your encouragement gave me the strength and motivation to overcome challenges and complete my thesis.

Thank you all for your invaluable contributions and for being part of this important chapter in my life.

附录 A 数据

A.1 Full Context for Error Analysis Example

Query:

How did Australia respond to the SARS outbreak in terms of quarantine measures and international cooperation?

Community Summary

Main Research Focus

The research community with **Community ID 100** focuses on the study of **Severe Acute Respiratory Syndrome (SARS)**, its causes, effects, and broader implications. SARS is a respiratory disease caused by the **SARS coronavirus (SARS-CoV)**, which emerged in 2002-2003, primarily in Southern Asia. The community explores the epidemiology, pathology, and control measures of SARS, including its origins, spread, and the role of environmental factors like **air pollution** and **viral mutations**. Additionally, the community investigates related viruses, such as **coronaviruses**, and their impact on public health, as well as potential treatments and vaccines.

Key Graph Paths

- `sars-cov virus has effect singapore sars outbreak`
- `singapore sars outbreak has cause sars-cov virus`
- `singapore diplomatic relation australia`
- `inflammatory responses has cause sars-cov replication`
- `sars-cov has effect epidemic`
- `sars epidemic has cause sars`

Retrieved Chunks

- singapore: In Australia SARS was declared a quarantinable disease under the Quarantine Act 1908 and policy guidelines for health professionals, airline and border control staff and the general public were developed by the Department of Health and Ageing, which also led an inter-departmental task force to monitor world developments.
- singapore There was to be no letting up on the vigilance and precaution. Not when it came to SARS.
- sars epidemic: The effectiveness of these measures was observed in all outbreak sites under widely varying conditions, supporting the overall WHO view that SARS can be contained and driven back out of its new human host. The present work aims at helping exploring one possible route accounting for at least some of the features of the epidemic.

- sars epidemic: In summary, the onset of the SARS epidemic in different continents has led to the formation of a successful laboratory network to identify the molecular mechanisms underlying the SARS infection.
- sars: For instance, the WHO played a critical role in controlling SARS by means of global alerts, geographically specific travel advisories and monitoring [59] .
- sars: The effectiveness of these measures was observed in all outbreak sites under widely varying conditions, supporting the overall WHO view that SARS can be contained and driven back out of its new human host. The present work aims at helping exploring one possible route accounting for at least some of the features of the epidemic.
- sars: In summary, the onset of the SARS epidemic in different continents has led to the formation of a successful laboratory network to identify the molecular mechanisms underlying the SARS infection.

These materials provide the retrieved context and reasoning that led to the partial but incomplete response discussed in Section **??**.

A.2 Supporting Context for Summarization Error Case

Query:

What was the effect of ultraviolet (UV) irradiation on the detection of aerosolized rhinovirus in the study?

Community Summary

Main Research Focus

This research community focuses on the study of aerosols, their components, and their effects on airborne infections. It explores various aspects of aerosol generation, filtration, and exposure, as well as the role of aerosols in viral transmission. The community also delves into related techniques, chemical compounds, and biological entities involved in aerosol analysis and the management of respiratory diseases.

Key Entities & Their Roles

1. **Aerosols and Their Components**: - **Aerosols**: Central to the study, aerosols are expelled particles that can carry viruses and other substances. - **Droplets**: A subclass of aerosols, part of airflow, and can affect contact. - **Nasal Drops, Rhinoviral, Fine Aerosol, UV Exposure Chamber, Crosslinking Agent**: Subclasses or parts of aerosols, each with specific roles in aerosol generation or exposure. - **Uranine**: A component of aerosols used in analysis. - **Filters**: Part of aerosols and nebulizers, including subclasses like pore-size, mutation filter, and TCID 50.

2. **Techniques and Equipment**: - **Nebulizer**: Used in aerosol generation, with subclasses like Pari Jet and Collison. - **Gel Electrophoresis**: A technique used for quantification and isolation, involving SDS-PAGE and agarose gels. - **Phosphorimager, Filancemeter, UV Detector**: Equipment used for quantification, mucociliary clearance, and detection.

3. **Chemical Compounds**: - **Ethanol, Acetone, Water, Calcium Chloride, Sds-15% (w/v)**: Various chemical compounds used in different processes like gel preparation and solution making. - **Polyacrylamide, Formamide, Ethidium Bromide, Osmium Tetroxide**: Components used in gel electrophoresis and other analytical techniques.

4. **Biological Entities and Diseases**: - **Virus**: Part of aerosols, involved in airborne infection. - **Mucomodulator, Mucolytic**: Involved in respiratory disease management, with mucomodulator being part of the mucociliary apparatus. - **Sodium Tetraborate, Calcium Chloride**: Used in medical treatments and as crosslinking agents.

5. **Miscellaneous**: - **Petri Dish, Test Tube, Exposure Chamber**: Containers used in experiments. - **Surface, Cell, Epithelial**: Biological surfaces and cells involved in various processes. - **Sequence Databases, Mutation Filter**: Used in viral sequence analysis.

Relationships & Structure

- **Aerosols and Airborne Transmission**: Aerosols, including droplets, are part of airborne transmission. They can carry viruses and other particles, leading to airborne infections.

- **Filtration and Exposure**: Filters, such as those in nebulizers, play a crucial role in controlling aerosols. Exposure chambers and drying jars are used to study aerosol effects.

- **Analytical Techniques**: Gel electrophoresis and related techniques (SDS-PAGE, agarose gels) are used for molecular weight analysis and quantification. UV detectors and phosphorimagers are employed for detection and measurement.

- **Chemical Interactions**: Various chemical compounds like ethanol, acetone, and calcium chloride are involved in gel preparation and other experimental processes. These compounds interact with biological entities to facilitate analysis and treatment.

- **Disease Management**: Mucomodulators and mucolytics are used in the management of respiratory diseases. Sodium tetraborate and calcium chloride act as crosslinking agents, influencing mucus properties.

- **Sequencing and Mutation Analysis**: Sequence databases and mutation filters are used in viral sequence analysis, helping to understand and manage viral mutations.

This community’s research is highly interdisciplinary, combining elements of chemistry, biology, and engineering to study aerosols and their impact on health and disease. The relationships between entities highlight a focus on both the generation and control of aerosols, as well as their role in disease transmission and management.