

Master's Programme in ICT Innovation

AcademicRAG: Knowledge Graph Enhanced Retrieval-Augmented Generation for Academic Resource Discovery

Enhancing Educational Resource Discovery through Knowledge Graph-Based RAG Framework

Zhuchenyang Liu

© 2025

This work is licensed under a [Creative Commons](#)
“Attribution-NonCommercial-ShareAlike 4.0 International” license.



Author Zhuchenyang Liu

Title AcademicRAG: Knowledge Graph Enhanced Retrieval-Augmented Generation for Academic Resource Discovery — Enhancing Educational Resource Discovery through Knowledge Graph-Based RAG Framework

Degree programme ICT Innovation

Major Data Science

Supervisor Dr. Petter Holme

Advisors Dr. Fredrik Heintz , Dr. Elias Zea, Dr. Shiva Sander Tavallaey

Collaborative partner KTH Royal Institute of Technology, ABB Group

Date 25 May 2025 **Number of pages** 62+11 **Language** English

Abstract

This thesis presents AcademicRAG, a novel knowledge graph-enhanced framework for academic resource discovery that addresses critical limitations in traditional information retrieval systems. By integrating graph-based retrieval mechanisms with large language models, the framework significantly improves the capture, representation, and utilization of complex semantic relationships inherent in academic knowledge. The research introduces two key technical innovations: clue-guided keyword generation and subgraph-based retrieval, which together enhance query performance across diverse academic domains.

Comprehensive evaluations demonstrate that AcademicRAG consistently outperforms existing state-of-the-art approaches in response comprehensiveness, diversity, and user empowerment. The practical implementation of a Course Discovery System validates the framework's utility in educational contexts, effectively addressing distinct information needs for both students and faculty. For students, the system provides personalized learning pathways and prerequisite analyses; for faculty, it offers curriculum optimization insights by identifying content overlaps and progression patterns across departmental boundaries.

Despite notable achievements, AcademicRAG and its application acknowledges limitations in entity extraction consistency and relationship standardization, suggesting future work in hybrid extraction approaches and multimodal content integration. AcademicRAG represents a significant advancement toward more intuitive, context-aware academic information systems, enabling richer exploration of educational resources and more informed academic decision-making.

Keywords Knowledge Graph, Retrieval-Augmented Generation, Academic Resource Discovery, Large Language Models, Graph Retrieval, Course Discovery System, Educational Technology

Preface

I wish to express my heartfelt gratitude to my parents and family for their unwavering support and encouragement throughout my Master's studies. Their belief in my academic pursuits has been a constant source of motivation.

I sincerely thank my advisors, Dr. Shiva Sander Tavallaey, Dr. Elias Zea, and Dr. Fredrik Heintz, for their invaluable guidance, insightful feedback, and scholarly mentorship throughout this research. Their expertise and dedication have been instrumental in shaping this thesis.

I would also like to thank my supervisor, Dr. Petter Holme from Aalto University, for his supervision and evaluation of this work.

Special appreciation goes to my collaborative partner, Shuhua Chen, for the productive partnership that made this ambitious project possible. The synergy of our combined efforts has significantly enriched the quality and scope of this work.

I would like to extend my gratitude to SUPR (Swedish National Infrastructure for Computing) for providing the essential computational resources that enabled the extensive experimentation and evaluation presented in this research.

I am also grateful to Dr. Yunfan Gao for graciously permitting the use of the figure from his research paper, which has enhanced the clarity of our presentation.

Finally, I wish to acknowledge the friends I have met during my Master's journey, whose companionship has made this academic adventure truly memorable. I am deeply grateful to Aalto University, KTH Royal Institute of Technology, ABB Group, and the EIT (European Institute of Innovation and Technology) Master's program platform for providing the exceptional academic environment, industrial collaboration opportunities, and international perspective that have shaped my research and personal growth.

Otaniemi, 25 May 2025

Zhuchenyang Liu

Contents

Abstract	3
Preface	4
Contents	5
Symbols and abbreviations	7
1 Introduction	8
1.1 Background and motivation	8
1.2 Research questions and scope	8
1.3 Structure of the thesis	9
1.4 Division of the work	10
2 Overview of Knowledge Graph-based RAG	12
2.1 LLMs	12
2.2 Definition of knowledge graph	12
2.3 RAG	13
2.4 Graph-based RAG	15
2.5 Related works on Graph-based RAG	16
2.6 Potential improvements on existing Graph-based RAG frameworks	18
3 AcademicRAG Framework Architecture	19
3.1 Overview of the framework	19
3.2 Graph-based text indexing	20
3.2.1 Document chunking	20
3.2.2 Entity and relationship extraction	20
3.3 Graph-Guided query retrieval	22
3.3.1 Keywords retrieve using clues	22
3.3.2 Local information extraction based on subgraph	23
3.3.3 Global information extraction	24
3.4 Graph-enhanced generation	25
3.5 AcademicRAG framework evaluation	25
3.5.1 Experiment settings	25
3.5.2 Evaluation result compared with baselines	28
3.5.3 Computing resources analysis	29
3.6 Summary of academicRAG framework architecture	31
4 Data Pipeline for AcademicRAG	32
4.1 Document process flow	33
4.2 Document index flow	34
4.3 Element extraction flow	34
4.4 Data pipeline evaluation	35
4.4.1 Experiment settings	35

4.4.2	Knowledge graph evaluation for research articles	36
4.4.3	Knowledge graph evaluation for course syllabi	38
4.5	Summary of AcademicRAG data pipeline	40
5	Application: Course Discovery System using AcademicRAG	42
5.1	Adaptation of AcademicRAG for course syllabi	42
5.2	System design	43
5.3	Evaluation	44
5.3.1	Evaluation setting	44
5.3.2	Knowledge graph evaluation	45
5.3.3	Student perspective evaluation	47
5.3.4	Faculty perspective evaluation	48
6	Discussion	50
6.1	Effectiveness of AcademicRAG framework and pipeline	50
6.1.1	Framework performance and evaluation considerations . . .	50
6.1.2	Technical advantages of subgraph retrieval and clue mechanisms	51
6.1.3	Versatility and robustness of the AcademicRAG data pipeline	51
6.2	Educational applications and impact	52
6.2.1	Student-centered learning support and pathway planning . .	52
6.2.2	Curriculum analysis and academic program optimization . .	53
6.3	Limitations and future work	54
6.3.1	Current limitations	54
6.3.2	Potential improvements and extensions	55
6.3.3	Integration of multimodal content	56
7	Conclusion	58
A	Github Repository	63
B	LLM Judge Evaluation Protocol	64
B.1	System instruction	64
B.2	Evaluation template	64
B.3	Implementation details	65
C	Academic Framework Ablation Studies	66
D	Academic Framework Case Study	68
E	Entity Types Design for Course Syllabi	72
F	Detailed Evaluation Results for Application	73

Symbols and abbreviations

Symbols

c_i	text chunk
d_j	document
\mathcal{D}	set of documents
C	set of text chunks
E	set of entities
R	set of relationships
$\mathcal{K}^{\text{content}}$	set of content keywords
$\mathcal{K}^{\text{clues}}$	set of clue keywords
\mathcal{K}_h	set of high-level keywords
\mathcal{K}_l	set of low-level keywords
Q	user query
τ	similarity threshold
$f_{\text{embed}}(\cdot)$	embedding function
$\text{sim}(\cdot, \cdot)$	similarity function
\mathcal{G}	knowledge graph

Operators

$\text{LLM}_{\text{Extraction}}(\cdot, \cdot)$	LLM configured for entity-relationship extraction
$\text{LLM}_{\text{keyword}}(\cdot, \cdot)$	LLM configured for keyword generation
$\text{LLM}_{\text{response}}(\cdot, \cdot)$	LLM configured for response generation
$\text{pipeline}_{\text{local}}(\cdot, \cdot)$	local information extraction function
$\text{pipeline}_{\text{global}}(\cdot, \cdot)$	global information extraction function
$\text{Concat}(\cdot, \cdot)$	contextual fusion operator

Abbreviations

KG	Knowledge Graph
LLM	Large Language Model
RAG	Retrieval-Augmented Generation
NLP	Natural Language Processing
RLHF	Reinforcement Learning from Human Feedback
GRPO	Group Relative Policy Optimization
QA	Question Answering

1 Introduction

1.1 Background and motivation

The rapid expansion of academic information in digital repositories has created both opportunities and challenges for researchers, educators, and students [1]. While vast amounts of knowledge are now accessible, effectively navigating these resources remains problematic. This research addresses this challenge through the development of a Graph-based RAG framework specifically designed for semantic search and discovery in academic contexts.

Traditional information retrieval systems rely heavily on keyword matching, which often fails to capture the complex semantic relationships between academic concepts [2], leading to incomplete search results and missed connections across disciplines. These systems struggle particularly with contextual understanding, synonym recognition, and identifying conceptual linkages that extend beyond lexical similarity. Consequently, valuable academic resources may remain undiscovered, hindering knowledge transfer and interdisciplinary collaboration.

While recent advances in LLMs [3] have improved text processing capabilities, standard RAG approaches still present limitations when applied to academic contexts [4]. Conventional RAG methods, despite enhancing LLMs with external knowledge retrieval [5], typically utilize flat document structures that fail to represent the intricate relationships inherent in academic knowledge. These relationships—such as prerequisite connections between courses, methodological similarities across research papers, or conceptual hierarchies within disciplines—are crucial for comprehensive academic resource discovery [6].

By integrating knowledge graph structures [7] with advanced embedding techniques from NLP, this research aims to develop a framework that preserves and leverages these complex relationships. The proposed AcademicRAG framework addresses the limitations of both traditional search systems and standard RAG implementations [8] by explicitly modeling the interconnected nature of academic knowledge, enabling more intuitive navigation and discovery of resources.

This research is significant as educational institutions increasingly seek AI-enhanced tools to improve knowledge accessibility and learning experiences. The framework has potential applications across university course discovery systems, research literature navigation platforms, and interdisciplinary learning tools. The potential collaborating organization envisions deploying this framework to enhance academic productivity, facilitate cross-disciplinary connections, and ultimately transform how academic resources are discovered and utilized.

1.2 Research questions and scope

This master’s thesis project explores the intersection of knowledge graphs, retrieval systems, and language models through three principal research questions:

1. How can graph databases and language models be optimally integrated in a graph-based RAG architecture to enable efficient academic resource discovery?

2. What architectural patterns enable a flexible data pipeline capable of processing both unstructured and semi-structured academic content while preserving semantic relationships?
3. How can the graph-based framework be effectively applied to downstream academic tasks, and what is its performance in real-world applications?

The scope of this research encompasses the design, implementation, and evaluation of the AcademicRAG framework, with particular focus on two practical applications: academic literature navigation and university course discovery. While the framework aims to be adaptable across various academic contexts, this thesis limits its experimental validation to these two use cases. The research addresses both technical aspects and practical considerations. The evaluation will measure both system performance and user experience to provide a comprehensive assessment of the framework in real-world academic scenarios.

1.3 Structure of the thesis

This thesis is organized into seven chapters that systematically present the research, development, and evaluation of the AcademicRAG framework as a collaborative project between two master's students.

Chapter 1 introduces the research context, outlines the motivation behind developing a graph-based RAG framework for academic applications, presents the key research questions, and defines the scope of the investigation.

Chapter 2 provides a comprehensive overview of knowledge graph-based RAG, establishing the theoretical foundation for the research. This chapter defines key concepts including knowledge graphs, RAG, and LLMs, followed by a critical literature review of existing graph-based RAG approaches.

Chapter 3 presents the proposed AcademicRAG Framework Architecture in detail, which was collaboratively developed by both master's students. The chapter begins with a high-level overview before examining each component: graph-based text indexing, graph-guided query retrieval (including keyword retrieval using clues and information extraction based on subgraphs), and graph-enhanced generation processes. The chapter also includes comprehensive evaluation of the framework across various metrics and use cases.

Chapter 4 explores the data pipeline developed for AcademicRAG, detailing the processes for data acquisition, preprocessing, knowledge graph construction, and maintenance that enable the framework's functionality. This pipeline was jointly developed to support both application domains.

Chapter 5 demonstrates the practical implementation of the Course Discovery System, which represents my specific contribution to this collaborative project. This chapter details the system architecture, implementation challenges, and specific adaptations of the AcademicRAG framework for educational content navigation.

Chapter 6 provides a critical discussion of the research findings, addressing limitations, comparing results, and exploring implications for academic resource

discovery. This discussion synthesizes insights from my Course Discovery System and highlights the effectiveness of the framework.

Chapter 7 concludes the thesis by summarizing the key contributions, presenting final thoughts on the research outcomes, and suggesting directions for future work.

The thesis ends with a comprehensive references section and appendices covering additional details on the framework’s evaluation, case studies, and implementation specifics.

1.4 Division of the work

This paper serves as a summary report for a two-person project. As the project involved collaborative efforts in literature review, framework design, and data pipeline development, we carefully divided the workload to ensure balanced contributions and efficient progress.

Furthermore, given that a master’s thesis is attributed to a single author, the jointly completed sections naturally resulted in similar content in the initial parts of our respective papers. To uphold academic rigor and meet the requirements of thesis writing, we have clarified the division of writing responsibilities for these shared sections. The details of this division are presented in Table 1.

Table 1: Division of the Labour for the Report Writing

Section	Subsection	Contributor(s)
Introduction	1.1 Background and motivation	S.C., Z.L.
	1.2 Research questions and scope	S.C., Z.L.
	1.3 Structure of the thesis	Z.L., S.C.
Overview	2.1 LLMs	S.C., Z.L.
	2.2 Definition of knowledge graph	S.C., Z.L.
	2.3 RAG	S.C., Z.L.
	2.4 Graph-based RAG	S.C., Z.L.
	2.5 Related works on Graph-based RAG	S.C., Z.L.
	2.6 Potential improvements on existing Graph-based RAG frameworks	Z.L., S.C.
Framework	3.1 Overview of the framework	Z.L., S.C.
	3.2 Graph-based text indexing	Z.L., S.C.
	3.3 Graph-guided query retrieval	S.C., Z.L.
	3.4 Graph-enhanced generation	Z.L., S.C.
	3.5 AcademicRAG framework evaluation	S.C., Z.L.
	3.6 Summary	Z.L., S.C.
Pipeline	4.1 Document process flow	S.C., Z.L.
	4.2 Document index flow	S.C., Z.L.
	4.3 Element extraction flow	S.C., Z.L.
	4.4 Data pipeline evaluation	Z.L., S.C.
	4.5 Summary	Z.L., S.C.
Application	All Subsections	Z.L.
Discussion	All Subsections	Z.L.
Conclusion	All Subsections	Z.L.

Note: S.C. indicates the cooperator Shuhua Chen (shuhua@kth.se). Names in front indicate the main writer; names in back indicate the person responsible for refinement.

2 Overview of Knowledge Graph-based RAG

2.1 LLMs

LLMs are a class of deep learning models trained on vast amounts of textual data to understand and generate human-like text. Built primarily on the Transformer architecture [9], modern LLMs leverage self-attention mechanisms to process sequential data while capturing long-range dependencies, enabling unprecedented performance in NLP tasks.

The evolution of LLMs began with bidirectional models like BERT [10], which introduced masked language modeling for contextual representation learning. Subsequent autoregressive models such as GPT [11] demonstrated the effectiveness of unidirectional transformers for text generation. The scaling of model parameters (e.g., GPT-3 with 175 billion parameters [3]) and dataset sizes revealed emergent capabilities, including few-shot learning and chain-of-thought reasoning [12].

Most advanced LLMs like GPT-4 [13] and DeepSeek-R1 [14] are built on subsequent autoregressive models and achieve state-of-the-art performance across diverse benchmarks, enabling applications ranging from conversational AI to code generation. The generation process can be formulated as the following optimization problem:

$$y_t = \arg \max_y \mathcal{P}_\theta(y_{t-1}, \dots, y_1 | x), \quad (1)$$

where \mathcal{P}_θ denotes the probability distribution, x is the input, and y_1, \dots, y_t are the output sequence of LLMs. However, LLMs face limitations including hallucination of factual inaccuracies [2], temporal knowledge cutoffs, and insufficient reasoning robustness [15].

Recent research focuses on alignment techniques like RLHF [16] and GRPO [17]. The development of open-source alternatives (e.g., LLaMA [18]) has further democratized LLM research while raising discussions about ethical deployment.

2.2 Definition of knowledge graph

A Knowledge Graph (KG) is a structured representation of real-world entities and their relationships, organized as a directed graph $G = (E, R, T)$, where E denotes entities (e.g., people, concepts, or events), R represents relation types (e.g., "is-a," "part-of," or "located-in"), and $T \subseteq E \times R \times E$ is a set of triples [1]. Each triple $(h, r, t) \in T$ encodes the fact that a head entity h is connected to a tail entity t via relation r . For example, (Paris, capitalOf, France) captures geopolitical knowledge. This graph-based structure allows for efficient reasoning, querying, and integration of heterogeneous data sources, making KGs indispensable for semantic understanding and decision-making systems.

Building a KG involves extracting knowledge from unstructured text (via Named Entity Recognition and Relation Extraction) and structured data integration (e.g., Wikidata [7]), and completing existing KGs through link prediction or rule-based reasoning. For example, KnowEdu, an education-focused KG system, leverages neural

sequence labeling and probabilistic association rules to derive pedagogical concepts and their prerequisite relationships from curriculum data [6]. Specifically, because LLMs show excellent performance on semantic understanding and entity recognition, [19] builds the KG by utilizing LLMs to extract entities and relationships.

KGs have found extensive applications across various domains, leveraging their ability to model complex relationships and integrate heterogeneous data sources:

- Search Engines and Information Retrieval: Commercial systems like Google’s Knowledge Graph leverage entity relationships to disambiguate queries and surface contextual information, improving search relevance significantly in benchmark evaluations.
- Recommendation Systems: KGs power recommendation engines by identifying connections between user preferences and available content, leading to more personalized suggestions.
- Healthcare and Life Sciences: KGs integrate vast amounts of biomedical data, aiding in drug discovery, disease research, and personalized medicine by uncovering hidden relationships among biological entities.

Emerging applications combine KGs with LLMs for enhanced reasoning and generation. DRAGON [20] achieved state-of-the-art performance on complex QA benchmarks including CSQA [21], OBQA [22] and so on by training LLM on KG data. Future directions include dynamic KG construction from multimodal inputs and quantum-enhanced graph embeddings for real-time decision systems.

2.3 RAG

RAG is a hybrid approach that combines the power of large pre-trained language models with external knowledge retrieval mechanisms to enhance performance on knowledge-intensive tasks [4], [5]. Unlike traditional language models that rely solely on parametric memory—where knowledge is stored within model weights—RAG incorporates non-parametric memory by retrieving relevant documents from an external database before generating responses. This integration mitigates the common limitations of LLMs, such as hallucinations, outdated knowledge, and limited interpretability, making RAG a critical advancement in NLP.

The mechanisms of RAG enable LLMs to leverage external knowledge dynamically, improving their ability to generate accurate, informative, and up-to-date responses. Unlike standard LLMs that rely solely on memorized knowledge from pretraining, RAG retrieves relevant information from external sources, making it highly adaptable to rapidly evolving domains such as news and scientific literature. One of RAG’s most significant advantages is its ability to mitigate hallucinations, a common issue where LLMs generate seemingly plausible but factually incorrect information. By grounding responses in retrieved, verifiable documents, RAG improves factual consistency and reliability. Additionally, RAG enhances interpretability and transparency, as users can inspect the external sources that contribute to the generated responses. A typical

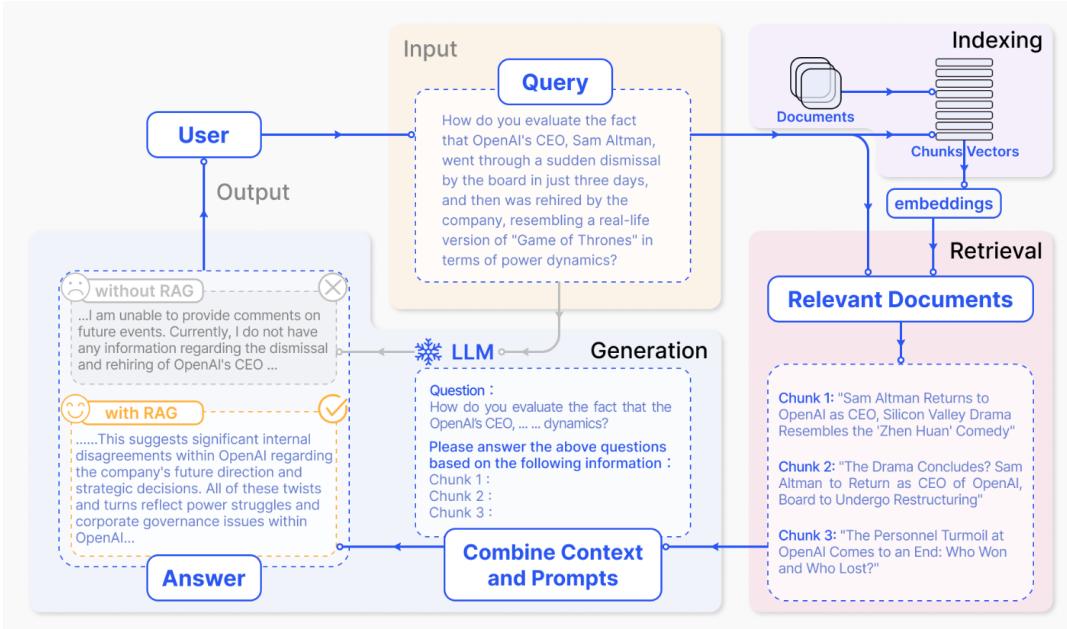


Figure 1: RAG process applied to QA [5]

Note: This image is used with the permission of the author.

RAG application is illustrated in Figure 1. Consider a scenario where a user asks ChatGPT about a recently trending news topic. Since ChatGPT’s knowledge is limited to its pretraining data, it may lack updated information on recent developments. The RAG framework bridges this gap by retrieving relevant news articles from external databases. These retrieved documents, combined with the original query, form a comprehensive prompt that enables the LLM to generate a factually grounded and well-informed response.

The traditional RAG framework consists of three primary steps:

- **Step 1. Indexing:** The first step involves preparing and structuring raw data from diverse formats, such as PDF, HTML, Word, and Markdown. These documents are processed through a cleaning and extraction pipeline to ensure uniform plain-text representation. Given the context length limitations of LLMs, the text is segmented into smaller, manageable chunks. Each chunk is then encoded into a vector representation using an embedding model and stored in a vector database, enabling efficient similarity searches in the retrieval phase.
- **Step 2. Retrieval:** When a user submits a query, the system first encodes it into a vector representation using the same embedding model used during indexing. The RAG framework then computes similarity scores between the query vector and the stored text chunks in the database. The top K most relevant chunks are retrieved based on their similarity to the query, ensuring that the response is grounded in relevant external knowledge. These retrieved chunks are incorporated into the expanded context of the prompt.
- **Step 3. Generation:** In the final stage, the retrieved documents and the original

query are combined into a structured prompt, which is then processed by an LLM, such as GPT [13]. The model generates a response based on the retrieved information while also leveraging its internal parametric knowledge when necessary. Depending on the application, the LLM can be restricted to only use the retrieved documents to enhance factual accuracy. Additionally, in cases involving multi-turn dialogue, the conversation history can be incorporated into the prompt, allowing the model to maintain coherence across interactions.

RAG has been applied in various fields with notable success: (1) **Open-Domain Question Answering:** RAG significantly improves performance on QA tasks by fetching relevant passages from Wikipedia or specific datasets. It has outperformed traditional extractive QA models on datasets such as Natural Questions and TriviaQA [23]. (2) **Legal and Medical NLP:** Due to the necessity of precise and up-to-date knowledge, RAG has been applied in legal and medical domains, where it retrieves case law or medical literature to assist decision-making [24]. (3) **Chatbots and Virtual Assistants:** By integrating RAG, conversational agents such as ChatGPT [13] can provide more accurate responses by retrieving real-time information, such as financial data, technical documentation, or recent news [5].

2.4 Graph-based RAG

Graph-based RAG is an extension of RAG that enhances retrieval mechanisms by incorporating structured knowledge from graphs, such as KGs and other relational databases [8]. Unlike traditional RAG systems that primarily retrieve and use unstructured text, Graph-based RAG retrieves structured entities, relationships, and subgraphs to enrich language model responses with contextual and relational knowledge. This approach enables more precise reasoning, improved factual accuracy, and enhanced interpretability, making it particularly useful in domains requiring strong logical consistency and entity relationships.

The Graph-based RAG framework can be decomposed into three primary stages [8]:

- **Step 1. Graph-based Indexing (G-Indexing):** This step involves the construction or selection of a structured graph database that serves as the retrieval corpus. Graph-based indexing facilitates efficient querying by organizing nodes (entities) and edges (relationships) in a way that optimizes search operations. The graph data can come from open knowledge graphs such as Wikidata [7] or be built from private datasets. Indexing constitutes the initial and the most important phase of Graph-based RAG as it determines the granularity of the subsequent retrieval stage, playing a crucial role in enhancing query efficiency.
- **Step 2. Graph-Guided Retrieval (G-Retrieval):** Instead of retrieving isolated text snippets, Graph-based RAG aims to retrieve the most relevant graph elements such as nodes, triples (subject-predicate-object), paths, or subgraphs relevant to a given query. Retrieval methods may include traditional graph traversal

algorithms, semantic similarity searches, or graph neural network (GNN)-based retrieval models.

- **Step 3. Graph-Enhanced Generation (G-Generation):** Retrieved graph data is integrated into the response generation process. This can be done by converting graph structures into textual formats or directly using structured data representations through graph-aware transformers or hybrid models that combine GNNs with LLMs. In this stage, the generator takes the query, retrieved graph elements, and an optional prompt as input to generate a response.

These three stages work together to improve the contextual depth of generated responses, ensuring that relational knowledge is incorporated effectively.

Graph-based RAG enhances traditional RAG by integrating structured knowledge, leading to improved reasoning, accuracy, and interpretability. Compared to text-based retrieval, the incorporation of structured relational data brings several notable advantages:

(1) **Preservation of Relational Knowledge:** Unlike RAG, which primarily focuses on retrieving semantically similar text, Graph-based RAG explicitly models relationships between entities. This structured retrieval enables more precise reasoning, especially for multi-hop inference tasks where multiple entities and their relationships need to be considered together.

(2) **Efficient Knowledge Integration:** Graph structures enable a more hierarchical and contextual organization of knowledge, making it easier to integrate diverse sources of information into LLMs. Particularly, Graph-based RAG allows selective retrieval of only relevant knowledge (e.g., specific relationships rather than full documents), reducing noise and irrelevant information.

(3) **Reduced Information Redundancy:** By retrieving subgraphs instead of long textual passages, Graph-based RAG minimizes irrelevant content and improves response efficiency. Since graphs store knowledge in an explicit, compressed format, Graph-based RAG also mitigates the “Lost in the Middle” [25] problem where transformers fail to attend to relevant context due to input length constraints.

Despite its advantages, Graph-based RAG still faces some challenges. Building and maintaining high-quality knowledge graphs require significant effort and domain expertise, which introduces extra burden. Additionally, there is not universal retrieval paradigm that can efficiently retrieve relevant subgraphs from large-scale graph databases, which remains an open research problem. Converting structured graph data into formats suitable for LLMs without losing relational integrity is also an ongoing challenge.

2.5 Related works on Graph-based RAG

Graph-based RAG extends standard RAG by leveraging structured knowledge graphs to enhance retrieval and reasoning. One of the pioneering approaches in this domain was introduced by Microsoft team, which employs LLMs to extract entities and relationships from unstructured text and constructs a knowledge graph [19]. To facilitate efficient

retrieval, Microsoft’s GraphRAG uses community detection algorithms to partition the knowledge graph into distinct communities and then applies LLMs to generate community reports summarizing each community’s key information. During retrieval, the model fetches the most relevant community reports to guide response generation. However, this approach has several key limitations [26]:

1. **High computational cost:** Generating and storing community reports for all communities requires extensive computational resources.
2. **Scalability issues:** When new knowledge is inserted, all community reports must be recomputed, making it inefficient for dynamic and evolving datasets.
3. **Noise from irrelevant nodes:** Since the community reports summarize large portions of the knowledge graph, they often include unrelated nodes, introducing unnecessary noise into the retrieval process.

LightRAG proposes a dual-level retrieval framework that replaces communities and reports with local and global keyword extraction [26]. Instead of generating a community report for each partition, LightRAG extracts two levels of keywords: Local Keywords, capturing entity-specific information, and Global Keywords, providing a broader conceptual context. By indexing keywords instead of full community summaries, LightRAG significantly reduces retrieval latency and storage requirements while minimizing retrieval noise.

Medical GraphRAG optimizes Microsoft’s GraphRAG for medical applications by integrating external domain-specific knowledge sources such as medical textbooks and controlled medical vocabularies [24]. Instead of relying on community detection, Medical GraphRAG categorizes graph nodes based on predefined medical terminology. Relying on terminology is a powerful classification method but it does not generalize well beyond the medical field.

GRAG introduces ego-graph retrieval, where retrieval focuses on a local subgraph centered on an entity of interest [27]. While this approach improves retrieval relevance, it lacks strong inter-node connections and requires additional graph embedding models, increasing implementation complexity.

Unlike LLM-generated knowledge graphs, some methods, such as CG-RAG and the scholarly knowledge graph framework by Jia et al. [28], [29], construct knowledge graphs using manual rule-based techniques. These methods rely on explicitly defined citation relationships between academic documents and allow for precise, curated knowledge graphs that maintain citation integrity. However, these approaches have several drawbacks. They are effective for scientific papers and literature-based queries but do not extend well to other academic texts like course syllabi or textbooks. Additionally, since the knowledge graph relies on manually inserted relationships, many implicit entity connections may remain undiscovered.

2.6 Potential improvements on existing Graph-based RAG frameworks

Microsoft’s GraphRAG framework demonstrates notable domain adaptability and streamlined knowledge graph construction pipelines [19]. Nevertheless, several avenues warrant exploration to enhance its efficacy while preserving these foundational strengths.

Although community detection facilitates interpretable graph partitioning, its computational intensity and static community representations ultimately constrain scalability for dynamic knowledge repositories. Therefore, exploring indexing methodologies that transcend rigid community boundaries—such as adaptive subgraph sampling techniques like SimGRAG [30]—could substantially reduce maintenance overhead while simultaneously preserving structural awareness and semantic coherence.

Furthermore, LightRAG’s hierarchical keyword extraction presents a computationally efficient alternative to community reports [26]. However, its reliance on LLM-generated keywords inevitably introduces potential semantic drift concerns. Consequently, supplementing this approach with content-anchored keywords explicitly linked to source text segments could significantly enhance semantic grounding while still maintaining computational efficiency and retrieval performance.

Moreover, while subgraph-based retrieval exemplified by GRAG’s ego-graphs demonstrates promise for focused knowledge aggregation, it nonetheless risks oversimplifying complex connectivity patterns [27]. Thus, a hierarchical subgraph strategy integrating proximity-based traversal with calibrated content relevance scoring offers more nuanced contextual capture without necessitating computationally intensive graph embedding operations.

In conclusion, these proposed refinements address identified limitations while honoring the original frameworks’ architectural principles and design objectives. Comprehensive empirical evaluation across diverse domains therefore remains essential to quantify their impact on retrieval precision, computational efficiency, and overall system performance.

3 AcademicRAG Framework Architecture

3.1 Overview of the framework

AcademicRAG is a novel framework collaboratively developed to address the limitations of traditional retrieval systems in academic contexts. By integrating knowledge graph databases with RAG techniques, this framework creates a more semantically aware system capable of understanding and leveraging complex relationships within academic contexts.

Similar to other graph-based RAG architectures [8], [19], [24], [27], [28], [31]–[33], AcademicRAG operates through three primary components, which support automatic graph database generation, graph information extraction based on query statements and final answer generation, respectively.

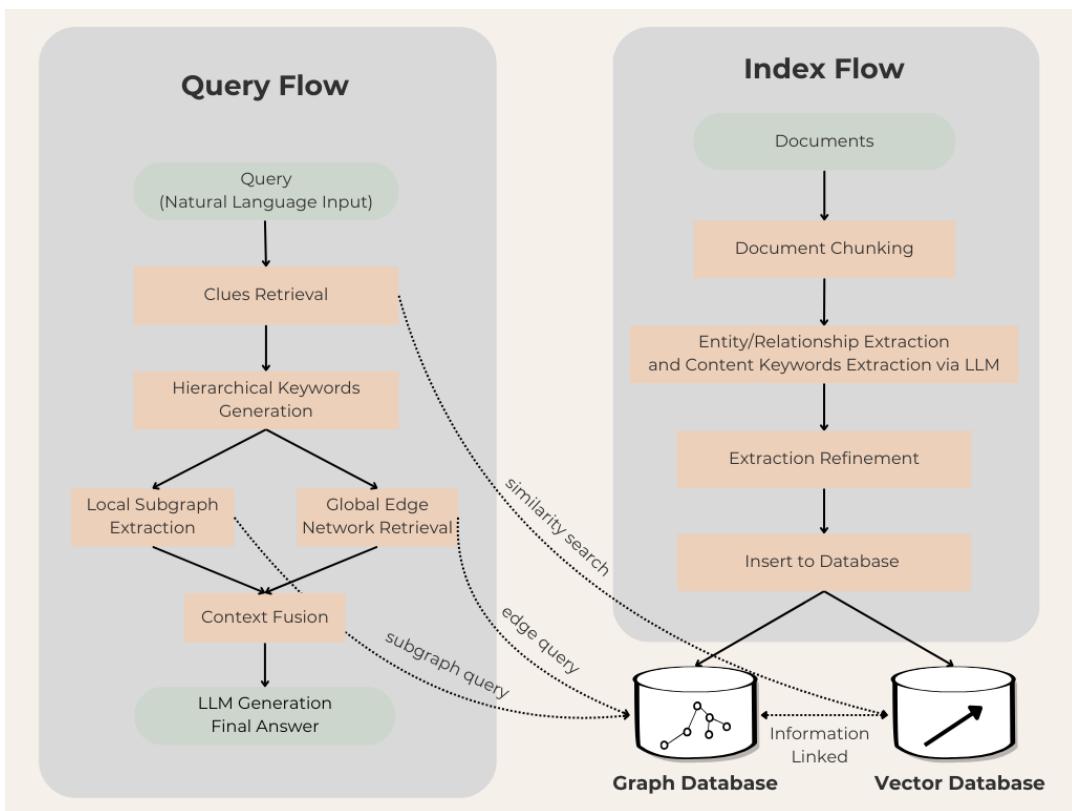


Figure 2: AcademicRAG Framework Flowchart

The AcademicRAG framework, as illustrated in Figure 2, consists of two processing flows that work in conjunction to deliver contextually rich and semantically aware responses to academic queries.

The Index Flow, shown on the right side, focuses on automatic graph database generation. It begins with document ingestion, followed by chunking to create manageable text segments. These chunks undergo entity and relationship extraction using LLMs, which identify key concepts, academic terms, and the connections between them. After extraction refinement to ensure accuracy, this structured information is

inserted into a dual-database architecture: a Graph Database that preserves relational information between academic entities, and a Vector Database that stores semantic embeddings of content.

The Query Flow, depicted on the left, handles user interactions through a multi-stage process. It starts with natural language query input, which undergoes clues retrieval to identify key academic concepts. These concepts are then organized into hierarchical keywords, enabling two parallel retrieval paths: Local Subgraph Extraction, which retrieves relevant concept networks from the Graph Database, and Global Edge Network Retrieval, which identifies broader relationship patterns. The Context Fusion stage integrates these retrieval results, providing a comprehensive knowledge foundation for the final LLM-based answer generation.

In the following sections, we will examine each of these components in detail, exploring their design principles, implementation considerations, and specific adaptations for academic resource discovery. We will outline how these components work together to create a unified framework that enhances both the retrieval precision and contextual awareness of academic information systems.

3.2 Graph-based text indexing

Graph-based text indexing is the first component of the AcademicRAG framework, transforming unstructured academic documents into a structured, queryable knowledge representation. The process is meticulously designed to capture the semantic richness of academic content through three stages: document chunking, entity and relationship extraction, and content keyword extraction.

3.2.1 Document chunking

The initial stage of the indexing process involves segmenting academic documents $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$ into text chunks $C = \{c_i \mid c_i \subseteq d_j, \forall d_j \in \mathcal{D}\}$ for processing. In the entity extraction stage, these text chunks will be passed to an LLM with prompts designed to extract the various elements of a graph index. Our approach employs a fixed-size chunking strategy optimized for academic content, the chunk length $|c_i|$ is a hyperparameter that can be varied in different application scenarios.

3.2.2 Entity and relationship extraction

The second stage of the indexing framework involves the systematic extraction of entities and relationships from document chunks through the application of LLMs. This phase is structured to ensure precision and adaptability to diverse academic domains.

LLMs are deployed with domain-specific prompts engineered to optimize entity and relationship extraction from academic texts. The input prompts are meticulously designed to embed chunked content, thereby directing the model to generate contextually relevant entities and relationships. Entities are identified and classified according to a domain-adaptive schema, which can be tailored to accommodate discipline-specific

terminologies and conceptual hierarchies. Relationships, represented as undirected edges between entities, are formulated as natural language statements that encapsulate semantic associations, such as causal links, functional dependencies, or categorical affiliations.

Inspired by Medical graph RAG [24], which uses external terminologies to work as global connections, we propose content keywords to capture the global themes of text units. During the extraction process, content keywords are extracted from each chunk. These content keywords refer to global level keywords that summarize the main concepts, themes, or topics of the entire text chunks. These should capture the overarching ideas present in the document.

To facilitate this extraction, we utilize a LLM specifically configured for entity-relationship extraction. This model operates based on a domain-specific prompt template, $P_{\text{domain}}(d)$, which simultaneously instructs both entity identification and relationship extraction. The prompt is tailored to a particular domain d , ensuring that the extraction process aligns with domain-specific requirements.

The entity extraction process can be formulated as follows:

$$(E_i, R_i, \mathcal{K}_i^{\text{content}}) = \text{LLM}_{\text{Extraction}}(P_{\text{domain}}(d), c_i), \forall i \quad (2)$$

where $\text{LLM}_{\text{Extraction}}$ is the LLM configured for extraction, c_i is an input document chunk, $E_i = \{e_1, e_2, \dots, e_n\}$ is the set of all recognized entities within the chunk, $R_i = \{(e_j, r_{jk}, e_k) \mid e_j, e_k \in E_i\}$ represents relationships between entity pairs with r_{jk} connecting entities e_j and e_k , and $\mathcal{K}_i^{\text{content}}$ represents global-level terms that capture the core concepts and themes of the document chunk.

To enable efficient retrieval from the graph database, we utilize an LLM-powered function $F(\cdot)$ to generate key-value pairs (K, V) for each entity e_i and relationship r_i . Each index key is a word or short phrase that facilitates fast and relevant retrieval, while the corresponding value is a textual summary containing relevant snippets from external data to support knowledge-enhanced text generation. Entities use their names as unique index keys, whereas relationships use the names of the two connected entities as their index keys.

A multi-pass iterative refinement mechanism is employed to maximize entity and relationship extraction across data chunks. The text chunks $\{c_i\}_{i=1,2,\dots}$ are repeatedly processed through multiple extraction rounds to capture as many entities and relations as possible. However, the multi-pass iterative extraction results in an exponential increase in the number of LLM calls. Therefore, this represents a trade-off between performance and cost. Empirically, performing a single re-extraction achieves the optimal balance between performance and cost.

During the extraction phase, duplicate entities are systematically consolidated to ensure the structural consistency of the graph. Concurrently, duplicate relationships are processed using an LLM, which generates a synthesized and integrative description aimed at preserving both the integrity and completeness of relational information. This methodology minimizes redundancy while enhancing the overall coherence and semantic fidelity of the resulting knowledge graph.

The refined entities and relationships are persistently stored in a graph database,

which preserves their structured interconnections and enables efficient traversal for query resolution. Concurrently, dense vector embeddings are generated for both entities and relationships using embedding models, and these embeddings, along with the extracted content keywords, are indexed in a vector database. This dual-storage architecture facilitates hybrid retrieval capabilities: the graph database supports structured queries based on explicit relationships, while the vector embeddings enable similarity-based searches, enhancing the system’s ability to retrieve contextually relevant information.

Another key advantage of our storage structure is its adaptability to incremental data updates. Unlike the community structure proposed in Microsoft’s GraphRAG [19], which requires recomputing entire community structures and reports when the graph updates, our framework eliminates this overhead by leveraging key-value pairs and content keywords for information retrieval. When a new document is added, our indexing strategy ensures that only the newly extracted entities, relationships, and content keywords are incorporated into the database after deduplication, rather than re-generating all communities and reports. This design significantly enhances efficiency and scalability, making the framework more suitable for dynamic and continuously evolving knowledge bases.

By integrating LLM-driven extraction, iterative refinement, and dual-format storage, this approach establishes a robust foundation for advanced academic knowledge retrieval, ensuring high fidelity in entity-relationship mapping and scalability across heterogeneous scholarly corpora.

3.3 Graph-Guided query retrieval

Graph-guided query retrieval enables efficient and context-aware information retrieval by leveraging structured relationships and semantic embeddings. This process is divided into three key stages.

3.3.1 Keywords retrieve using clues

Upon query initiation, the system employs a multi-stage retrieval strategy to optimize keyword relevance and contextual alignment. First, the content keywords vector database is queried to identify the content keywords that exhibit maximal semantic congruence with the user’s query. These content keywords, synthesized from both the query and the indexed corpus, function as retrieval clues to guide subsequent processing. The clues are retrieved via the equation:

$$\mathcal{K}^{\text{clues}} = \{k \in \mathcal{K}^{\text{content}} \mid \text{sim}(f_{\text{embed}}(Q), f_{\text{embed}}(k)) \geq \tau\}, \quad (3)$$

where Q represents the user query in natural language form, while $\mathcal{K}^{\text{content}}$ denotes the content keyword vector database. The embedding model, $f_{\text{embed}}(\cdot)$, projects text into a vector space, enabling the computation of semantic similarity through a predefined metric, $\text{sim}(\cdot, \cdot)$, such as cosine similarity. A threshold parameter, τ , determines the minimum required semantic congruence for a keyword k to be included in the retrieved set $\mathcal{K}^{\text{clues}}$, which subsequently guides further processing.

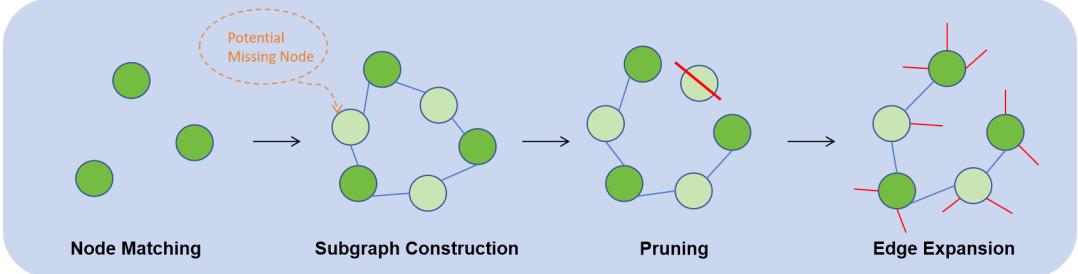


Figure 3: The Process of Subgraph Building

An LLM is then invoked to generate a hierarchical keyword set, comprising high-level (conceptual/thematic) and low-level (specific/contextual) terms derived from the original query and the identified clues. Low-level keywords target localized information retrieval, prioritizing granular details within narrow contexts, while high-level keywords facilitate broad-scope exploration of conceptual relationships. The process can be written as:

$$(\mathcal{K}_h, \mathcal{K}_l) = \text{LLM}_{\text{keyword}}(Q, \mathcal{K}^{\text{clues}}). \quad (4)$$

\mathcal{K}_h corresponds to high-level keywords that encapsulate conceptual or thematic aspects, whereas \mathcal{K}_l comprises low-level keywords focusing on specific contextual details. The model $\text{LLM}_{\text{keyword}}$ is specifically designed to generate these dual-level keywords by leveraging both the original query Q and the retrieved content keywords $\mathcal{K}^{\text{clues}}$. This dual-level keyword generation mechanism ensures adaptive alignment with user intent, enabling hybrid retrieval strategies that balance specificity and generality.

3.3.2 Local information extraction based on subgraph

For queries requiring localized insights, our framework executes a structured subgraph extraction protocol driven by low-level keywords \mathcal{K}_l . The workflow is formalized as follows:

1. **Node Matching:** Graph nodes are matched against low-level keywords \mathcal{K}_l using vector similarity metrics, prioritizing nodes with higher cosine similarity to the keyword embeddings.
2. **Subgraph Construction:** A subgraph is generated by identifying the shortest paths between matched nodes, ensuring topological coherence while preserving contextual proximity.
3. **Pruning:** Edges are filtered based on semantic relevance thresholds, retaining only those whose descriptive statements align with the query keywords. Some isolated nodes after filtering edges will also be removed.
4. **Text Unit Extraction:** Textual chunks that are associated with retained edges and appear in the nodes' one-hop ego-graph are ranked by frequency within the subgraph, with top-ranked segments prioritized for retrieval.

5. **Edge Expansion:** The subgraph is iteratively expanded by incorporating one-hop neighbors of matched nodes, introducing additional edges that satisfy relevance criteria.
6. **Data Truncation:** To adhere to computational constraints, all extracted data—including nodes, edges, and text units—are truncated iteratively until the total token count complies with predefined thresholds.

This pipeline outputs entities, relationships, and original text segments, ensuring that contextual fidelity is preserved while effectively capturing potential relationships. Figure 3 shows how the subgraph is retrieved. Unlike LightRAG [26], which restricts retrieval to one-hop neighbors, and GRAG [27], which relies on k-hop ego-graphs, our approach retrieves a complete subgraph for local information extraction, enabling the construction of richer connections between different nodes. This method enhances retrieval depth and knowledge integration, providing a more comprehensive representation. The whole local information extraction pipeline can be formulated as:

$$\mathcal{L}_{\text{local}} = (E_{\text{local}}, R_{\text{local}}, T_{\text{local}}) = \text{pipeline}_{\text{local}}(\mathcal{K}_l, \mathcal{G}), \quad (5)$$

where $\mathcal{L}_{\text{local}}$ represents the local information extracted, $\text{pipeline}_{\text{local}}$ denotes the local extraction function that processes low-level keywords \mathcal{K}_l against the knowledge graph \mathcal{G} , producing a set containing local entities E_{local} , relationships R_{local} , and associated text units T_{local} .

3.3.3 Global information extraction

For broad-scope queries, the framework leverages high-level keywords \mathcal{K}_h to retrieve global knowledge structures through the following steps:

1. **Edge Retrieval:** The vector database is queried to identify edges whose embeddings correlate with high-level keywords, prioritizing edges that encapsulate conceptual relationships (e.g., theoretical frameworks, domain-wide trends).
2. **Node Retrieval:** Nodes related to the retrieved edges are extracted and sorted by their degree.
3. **Edge Data Processing:** Retrieved edges are analyzed to extract metadata, including degree (sum of degree between source and target nodes) and weight (semantic connection strength), quantifying their structural and contextual significance.
4. **Integration and Ranking:** Edges are aggregated and ranked using a composite metric combining relevance scores (based on keyword alignment) and connection weights, ensuring that dominant conceptual linkages are prioritized.
5. **Text Unit Extraction:** Text chunks that are associated with edges are ranked by the similarity between edges and high-level keywords and retrieved.

6. **Data Truncation:** To adhere to computational constraints, all extracted data—including nodes, edges, and text units—are truncated iteratively until the total token count complies with predefined thresholds.

The final output includes interconnected entities, relationships, and supporting textual evidence, enabling holistic knowledge discovery across academic domains. By incorporating high-level keywords derived from content keywords, our framework effectively retrieves global information from the graph without relying on the community structure used in Microsoft’s GraphRAG [19]. This approach optimizes retrieval efficiency while maintaining a balance between performance and computing resource consumption, providing a scalable and efficient solution. The whole global information extraction pipeline can be formulated as:

$$\mathcal{L}_{\text{global}} = (E_{\text{global}}, R_{\text{global}}, T_{\text{global}}) = \text{pipeline}_{\text{global}}(\mathcal{K}_h, \mathcal{G}), \quad (6)$$

where $\mathcal{L}_{\text{global}}$ represents the global information extracted, $\text{pipeline}_{\text{global}}$ denotes the global extraction function that processes high-level keywords \mathcal{K}_h against the knowledge graph \mathcal{G} , producing a set containing global entities E_{global} , relationships R_{global} , and associated text units T_{global} .

3.4 Graph-enhanced generation

The final stage of the AcademicRAG pipeline synthesizes localized and global information into a coherent, contextually grounded response. Local subgraphs, extracted via low-level keywords, and global relationship networks, retrieved via high-level keywords, are concatenated into a unified contextual framework. This integration combines dual-level entities, relationships, and source text anchors.

Eventually, this integrated information is embedded in a prompt predefined for academic texts, generating the final answer through a large model. The graph-enhanced generation process is shown as Eq. 7:

$$A_{\text{final}} = \text{LLM}_{\text{response}} \left(P_{\text{final}}, \text{Concat}(\mathcal{L}_{\text{local}}, \mathcal{L}_{\text{global}}) \right), \quad (7)$$

where $\text{LLM}_{\text{response}}$ refers to the LLM configured to generate the final answer based on the given input. P_{final} represents the prompt used to format the input for the model. The information extracted by low-level keywords is denoted as $\mathcal{L}_{\text{local}}$, while information retrieved by high-level keywords is denoted as $\mathcal{L}_{\text{global}}$. The $\text{Concat}(\cdot, \cdot)$ operator performs the contextual fusion of these dual-level components, combining entities, relationships, and source text anchors into a cohesive structure. Finally, the final answer A_{final} is generated after this retrieval and synthesis process.

3.5 AcademicRAG framework evaluation

3.5.1 Experiment settings

This section presents the experimental setup for evaluating the AcademicRAG framework. To assess its performance in an academic context, we utilize a subset of

the UltraDomain benchmark [34] as the evaluation dataset. This benchmark comprises 428 college textbooks spanning 18 academic domains, providing a diverse and comprehensive knowledge base. For our experiments, we focus on two specific domains—agriculture and computer science—which include a total of 22 college textbooks (10 from computer science and 12 from agriculture). The considerable amount of textbooks ensures that the dataset can cover enough domain knowledge to conduct the query test and ensure the validity of the evaluation.

To assess the effectiveness of RAG systems in high-level sensemaking tasks, we aggregate all textual content from each domain as context and adopt the generation methodology described in Microsoft’s GraphRAG [19]. Specifically, we first instruct an LLM to generate five distinct RAG users, each representing a unique academic user, such as a "PhD Student Specializing in NLP" or a "Farmer Transitioning to Organic Practices." This approach allowed us to capture diverse information needs across the academic spectrum. Each synthetic user was assigned five distinct tasks that reflected common academic activities, such as "Identify prerequisite courses for ML" or "Adopt soil-building techniques without synthetic inputs". These tasks were deliberately designed to emphasize different cognitive intents, including exploratory, comparative, and problem-solving dimensions. For question synthesis, we generated five questions for each user-task pair, resulting in a total of 125 questions per domain (5 users \times 5 tasks \times 5 questions). These questions were crafted to require holistic understanding of the corpus rather than simple fact retrieval.

To establish the effectiveness of our approach, we compared AcademicRAG against the following state-of-the-art models on our evaluation dataset:

- **NaiveRAG** [4]: This model serves as a standard baseline in existing RAG systems. It segments input contexts into smaller text chunks and stores them in a vector database using text embeddings. During retrieval, NaiveRAG directly embeds the user’s query into a vector representation and retrieves the most relevant text chunks based on similarity scores, ensuring an efficient and straightforward retrieval process.
- **GraphRAG** [19]: This is a graph-based RAG system that leverages LLMs to extract entities and relationships from input text, structuring them as nodes and edges within a graph database. The system further generates descriptions for these elements, clusters nodes into communities for retrieval, and produces community reports to encapsulate global contextual information. When handling queries, GraphRAG enhances retrieval quality by traversing communities, enabling access to more comprehensive and interconnected knowledge.
- **LightRAG** [26]: An adaptation of GraphRAG, LightRAG follows a similar approach by utilizing LLMs to construct graph-based representations. However, instead of clustering elements into communities, it employs keywords to classify them. When processing queries, LightRAG retrieves one-hop neighbors and their relationships by keywords, enabling faster response generation while maintaining relevant contextual knowledge.

All of the evaluated frameworks use the same configuration. For graph construction, we employed Qwen2.5-72B to extract entities and relationships from the corpus, which were subsequently stored in the graph database. For the generation phase, we continue to employ Qwen2.5-72B to ensure consistent and relevant outputs. To maintain consistency across all datasets, chunk size is uniformly set to 1200. Additionally, the gleaning parameter is fixed at 1 for Microsoft’s GraphRAG, LightRAG, and our framework, ensuring a standardized retrieval process across different models.

Defining ground truth for those queries involving complex high-level semantics is challenging, so we implemented a pairwise comparison methodology utilizing a powerful reasoning LLM judge like Microsoft’s GraphRAG [19]. Specifically, we employed DeepSeek-R1 [14] to evaluate AcademicRAG’s responses against those of the baseline models. The details for LLM judge are described in Appendix B. The evaluation employed four critical dimensions:

1. **Comprehensiveness:** How much detail does the answer provide to cover all aspects and details of the question?
2. **Diversity:** How varied and rich is the answer in providing different perspectives and insights on the question?
3. **Empowerment:** How well does the answer help the reader understand and make informed judgements about the topic?
4. **Overall:** This dimension assesses the cumulative performance across the three preceding criteria to identify the best overall answer.

This methodological approach enables the calculation of a robust final score based on averaged winning percentages, ensuring reliable and unbiased comparative assessments. The LLM first evaluates both answers across the first three assessment dimensions, determining a preferred answer for each individual criterion. Following this dimensional analysis, the LLM synthesizes the results to identify the overall superior response as the winner.

Specifically, for each domain, we construct 125 question-answer pairs (Answer 1 and Answer 2), where Answer 1 and Answer 2 are generated by two distinct Graph-based RAG frameworks. By leveraging a LLM as a judge, each question-answer pair is evaluated across three evaluation dimensions as well as an overall assessment. This process enables us to determine the number of winning cases for each framework in relation to the questions. Ultimately, we derive the win rates for two Graph-based RAG frameworks.

To maintain evaluation fairness and mitigate potential positional bias introduced by the order in which responses appear in the prompt, we alternate the placement of each answer and compute the final average win rates based on three trials. This multi-faceted evaluation approach provides a nuanced understanding of each framework’s strengths and limitations in academic information retrieval contexts.

Table 2: AcademicRAG Win Rates (%) compared with other frameworks

Metrics	Framework	Win Rate	
		Agriculture	CS
Comprehensiveness	vs. NaiveRAG	62.8%↑	67.1%↑
	vs. LightRAG	50.0%	56.2%↑
	vs. GraphRAG	52.4%↑	51.2%↑
Diversity	vs. NaiveRAG	58.0%↑	83.2%↑
	vs. LightRAG	55.2%↑	57.8%↑
	vs. GraphRAG	50.8%↑	46.8%↓
Empowerment	vs. NaiveRAG	51.6%↑	75.7%↑
	vs. LightRAG	58.4%↑	59.8%↑
	vs. GraphRAG	54.0%↑	55.2%↑
Overall	vs. NaiveRAG	57.2%↑	77.5%↑
	vs. LightRAG	54.8%↑	56.6%↑
	vs. GraphRAG	52.4%↑	53.6%↑

Note: ↑ indicates AcademicRAG outperforming the compared framework, ↓ indicates AcademicRAG underperforming. Values are AcademicRAG win rates.

3.5.2 Evaluation result compared with baselines

Table 2 presents a comparison of our AcademicRAG framework with other baseline frameworks in the agriculture and computer science (CS) domains. As shown, our method consistently outperforms nearly all other state-of-the-art (SOTA) frameworks across all dimensions in both domains. From these results, we can draw the following conclusions:

- **Wide applicability of AcademicRAG:** This consistent advantage highlights the robustness and adaptability of AcademicRAG. In agriculture, where domain-specific expertise and nuanced contextual understanding are essential, AcademicRAG effectively captures and processes complex information. In computer science, where the rapid pace of innovation and the wide distribution of knowledge present significant challenges, our framework demonstrates remarkable flexibility and accuracy.

These results underscore the broad applicability and strength of AcademicRAG in handling diverse and dynamic knowledge domains. This strong performance across different academic domains highlights the versatility and adaptability of our framework, which indicates that AcademicRAG can provide more comprehensive, contextually relevant, and enriched responses compared to other RAG approaches.

- **Response Diversity:** AcademicRAG retrieves data in the form of entities and relationships, which inherently allows it to extract and utilize richer, more structured information compared to traditional RAG methods that rely on

retrieving entire text chunks. This structured retrieval approach not only enhances contextual understanding but also improves the accuracy and relevance of the generated responses.

Furthermore, by retrieving a subgraph rather than simply fetching one-hop neighbors from the knowledge graph, AcademicRAG can generate responses with greater diversity than LightRAG, as it incorporates a broader range of interconnected knowledge. However, while AcademicRAG improves upon LightRAG in this aspect, it does not have an advantage compared with GraphRAG in terms of diversity. We hypothesize that this is due to excessive noise present in GraphRAG’s community reports, which, despite increasing diversity, also introduce irrelevant or redundant information. This observation is further supported by the empowerment win rate, indicating that while GraphRAG retrieves a wider range of data, its responses may not always be as informative or useful as those generated by AcademicRAG.

- **Superiority over Graph-based RAG methods:** AcademicRAG demonstrates notable advantages over other graph-based RAG frameworks, including GraphRAG and LightRAG, particularly within the evaluated computer science corpus. AcademicRAG employs subgraph retrieval and clue-guided keyword generation, enabling it to precisely locate relevant elements while also uncovering hidden relationships and contextual information within the knowledge graph.

This enhanced retrieval mechanism allows AcademicRAG to extract deeper insights, leading to more informative, contextually rich, and well-structured responses within the scope of the evaluated corpus. By effectively synthesizing retrieved knowledge, AcademicRAG can help users grasp complex topics more efficiently and make well-informed decisions based on the available data.

To further validate the effectiveness of our approach, we conducted ablation studies on our subgraph retrieval and clues-guided keywords generation methods. These experiments systematically assessed the impact of each component by removing one while maintaining the other. The results, presented in Appendix C, demonstrate that both components contribute significantly to the framework’s overall performance, with notable performance degradation observed when either component is removed.

Additionally, we performed detailed case studies comparing AcademicRAG against other frameworks through specific queries, with comprehensive analysis and response examples provided in Appendix D. These case studies offer qualitative insights into how our framework achieves superior performance in real-world academic information retrieval scenarios.

3.5.3 Computing resources analysis

Compared to GraphRAG [19], one of the most significant advantages of AcademicRAG is its efficient utilization of computing resources. To quantify this efficiency, we compare the two frameworks based on two key metrics: the number of tokens consumed and the number of API calls required.

First, we analyze token consumption and API calls during the indexing and retrieval processes. Second, we examine how each framework handles data updates in dynamic environments, focusing on their adaptability and computational overhead. The results of this evaluation, conducted on the legal dataset, are presented in Table 3. In this context:

- $n_{\text{community}}$ is the number of detected communities during indexing time.
- T_{extract} represents the token overhead for entity and relationship extraction.
- C_{\max} denotes the maximum number of tokens allowed per API call.
- C_{extract} indicates the number of API calls required for extraction.

Table 3: Comparison of Indexing, Retrieval, and Incremental Text Update Phases between GraphRAG and Our Framework

Stage	Metric	GraphRAG	Ours
Indexing	Tokens	$n_{\text{community}} \times 5,000 + T_{\text{extract}}$	T_{extract}
	API Calls	$(n_{\text{community}} \times 5000/C_{\max}) + C_{\text{extract}}$	C_{extract}
Retrieval	Tokens	$610 \times 1,000$	< 500
	API Calls	$(610 \times 1,000/c_{\max})$	1
Incremental Text Update	Tokens	$n_{\text{community}} \times 5,000 + T_{\text{extract}}^{\text{new}}$	$T_{\text{extract}}^{\text{new}}$
	API Calls	$(n_{\text{community}} \times 5000/C_{\max}) + C_{\text{extract}}^{\text{new}}$	$C_{\text{extract}}^{\text{new}}$

During indexing, GraphRAG generates community reports for $n_{\text{community}}$ detected communities in addition to extracting entities and relationships. Each community report consumes approximately 5,000 tokens, and in our experiment, about 7,000 communities were identified within the knowledge graph. This means GraphRAG incurs an additional overhead of over 35 million tokens and 1,000 API calls per domain compared to AcademicRAG.

In the retrieval process, GraphRAG retrieves around 610 community reports per query, with each report containing 1,000 tokens. In contrast, AcademicRAG optimizes retrieval, consuming fewer than 500 tokens to extract keywords instead of processing entire community reports. This efficiency is achieved through our dual-storage architecture, which seamlessly integrates graph structures and vector-based representations, eliminating the need to process many community reports while maintaining retrieval accuracy.

For incremental updates, both GraphRAG and AcademicRAG require $T_{\text{extract}}^{\text{new}}$ tokens to extract newly added entities and relationships. However, GraphRAG suffers from a major inefficiency: it must delete outdated community structures and regenerate all community reports when handling new data. This leads to significant computational overhead and poor scalability in dynamic environments. In contrast, AcademicRAG

efficiently integrates newly extracted entities and relationships into the existing graph without requiring any reconstruction. Our method ensures minimal overhead during both indexing and incremental updates, making AcademicRAG a more cost-effective and scalable solution for real-world applications.

3.6 Summary of academicRAG framework architecture

The AcademicRAG framework integrates knowledge graph structures with retrieval-augmented generation to enhance academic information discovery. Its architecture comprises three key components:

- **Graph-based Text Indexing**, which transforms academic content into structured knowledge representations stored in dual graph and vector databases;
- **Graph-Guided Query Retrieval**, which leverages clue-guided keyword generation to extract both local subgraphs and global relationship networks; and
- **Graph-Enhanced Generation**, which synthesizes this information into contextually rich responses.

Evaluation results demonstrate AcademicRAG’s effectiveness across two typical academic domains, consistently outperforming baseline frameworks in comprehensiveness, diversity, and user empowerment. The framework’s subgraph retrieval mechanism and clue-guided keyword generation are particularly significant, enabling the system to identify complex relationships that more basic retrieval methods often miss.

AcademicRAG also achieves noteworthy efficiency in computational resource utilization, especially during retrieval and incremental updates. Unlike systems that require regenerating entire knowledge structures when new information is added, AcademicRAG’s architecture allows for seamless integration of new entities and relationships, making it both effective and scalable for dynamic academic environments.

The framework provides a robust foundation for diverse downstream academic applications, including course discovery systems and research literature assistants. By capturing complex academic relationships while remaining domain-adaptable, AcademicRAG enables the development of specialized tools that can navigate intricate educational pathways, uncover cross-disciplinary connections, and personalize information discovery. This versatility makes the framework particularly valuable in addressing domain-specific challenges while maintaining consistent performance across varying academic contexts.

4 Data Pipeline for AcademicRAG

In the AcademicRAG framework, the data pipeline is a key component to ensure the efficient operation of the system and the accurate generation of knowledge graphs. In order to provide more accurate semantic understanding and retrieval capabilities in a complex academic environment, AcademicRAG requires an automated and scalable data processing mechanism. By designing a complete end-to-end data pipeline, the system can extract structured knowledge from original documents and store it in various databases, ultimately providing a strong knowledge foundation for semantic retrieval and question-answer generation.

Our data pipeline design is inspired by and references GraphRAG [19] and LightRAG [26]. The data pipeline is a full-process automated processing workflow from document to database. Unlike GraphRAG and LightRAG, which only support text input, our data pipeline supports multiple input formats (such as PDF, HTML, JSON, TXT), and gradually converts documents into structured knowledge graphs through a series of modular workflows.

Furthermore, because we removed the community structure feature from GraphRAG, our data pipeline supports incremental input without requiring a complete reconstruction of the storage system. The data pipeline integrates functions such as document processing, index generation, and entity-relationship extraction. It ensures efficient data storage and retrieval by combining the graph database, vector database, and key-value database.

As shown in figure 4, AcademicRAG’s data pipeline consists of three main sublevel workflows that connect to each other and are processed in a hierarchical relationship:

- **Document Process Flow** is the first layer of the pipeline. It is responsible for processing and managing the raw documents input. Documents are controlled under a unified management framework to be passed to the **Document Index Flow** in an orderly manner, ensuring the robustness of the entire pipeline at the file management level.
- **Document Index Flow** is the second layer of the pipeline, which focuses on single document processing for the subsequent chunk indexing. This stage ensures that individual documents are processed in a structured manner while maintaining semantic information.
- **Element Extraction Flow** is the third layer of the pipeline and the most critical step, which is responsible for processing a single chunk. At this stage, entities, relationships and keywords are extracted from the chunk and stored in both graph and vector databases. It forms the basis for the query process.

The following sections provide detailed discussions on the design principles and implementation methods for each of these workflows, highlighting how AcademicRAG leverages its efficient data pipeline to construct an advanced semantic retrieval system tailored for academic contexts.

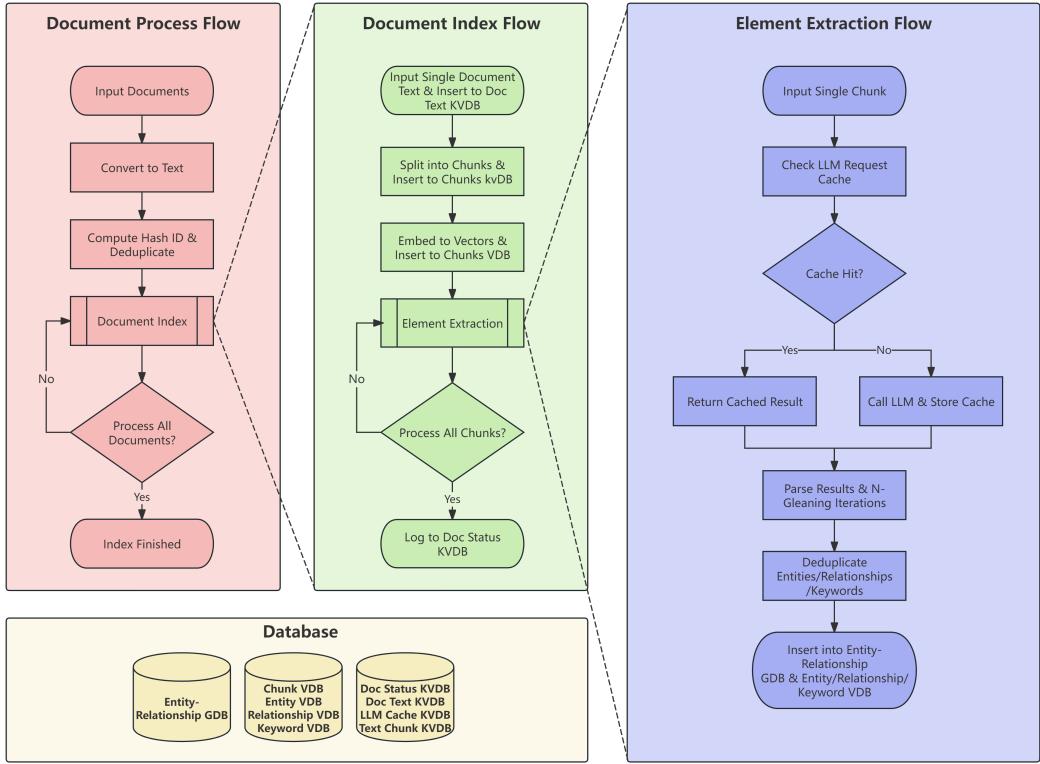


Figure 4: AcademicRAG Data Pipeline

4.1 Document process flow

In AcademicRAG’s data pipeline, the input academic documents are under the control of **Document Process Flow**. Firstly, heterogeneous academic documents are converted to standard text content. Then, a unique MD5 content hash value is computed based on each document’s content, serving as an indexing key in the database and avoiding duplicate input.

GraphRAG’s absence of file status persistence hinders users from verifying correct processing and causes significant overhead upon restarting interrupted workflows. We addressed this by introducing a file status database “Doc Status KVDB”, drawing inspiration from LightRAG [26]. After extracting basic metadata, the system sets the initial state of the document (such as PENDING) in the “Doc Status KVDB” based on the content hash value and records its metadata in the “Doc Status KVDB”. Through this state management mechanism, the system can maintain data consistency and integrity during large-scale document processing.

After the state setting is completed, the system filters the documents in the PENDING state and introduce them into the **Document Index Flow**. In this process, the single document will be inserted to the semantic databases. At the same time, the state management system updates the processing state to PROCESSING in order to avoid redundant processing of files in multi-threaded operations.

If the document is successfully processed, the system stores the segmented text

chunks and the generated embedding vectors in databases such as “Text Chunks KVDB” and “Chunk VDB”. Upon successful completion, the system will update the document status to PROCESSED. If processing fails, the system will set the status to FAILED and record detailed error logs for troubleshooting and recovery.

When all documents are processed, the system terminates the current document queue processing flow. At this point, successfully indexed documents can be efficiently retrieved through AcademicRAG’s query flows. Through strict state tracking and fault-tolerant design, this process ensures efficient processing of large-scale academic materials.

4.2 Document index flow

The **Document Index Flow** is responsible for converting input documents into structured semantic chunks for subsequent knowledge extraction. The process first receives the single document to be indexed, then it is split into semantic chunks, which serve as the basic units for **Element Extraction Flow** to facilitate refined knowledge extraction and vectorization.

After the documents are chunked, the system generates embeddings for each semantic chunk, which converts text into vector representations through embedding models. These vectors can capture the semantic information of the document content and are stored in the “Chunk VDB” vector database. Then, the system calls the **Element Extraction Flow** to perform in-depth indexing of the content in each semantic chunk.

After element extractions are completed, the system will update the document status information to PROCESSED in “Doc Status KVDB”. If a problem occurs during processing, the system will set the document status to FAILED and record the error log in detail for troubleshooting and recovery. Through strict state management and exception handling mechanisms, the system ensures that the document indexing process can be completed efficiently and reliably.

4.3 Element extraction flow

The **Element Extraction Flow** is the third layer and responsible for extracting entities, relationships and keywords from semantic chunks of document to build the underlying vector and graph database for AcademicRAG. This process receives single-chunked text from the **Document Index Flow** as input and calls an LLM to identify the semantic elements.

In order to reduce the consumption of computational resources, we introduced LLM cache storage like LightRAG [26]. Firstly, the system checks the LLM cache to determine whether there are extraction results for the same text chunk that have been processed before. If the cache hits, the system directly returns the cached results to avoid redundant computation. If there is no hit, the system initiates a request to the LLM to extract entities, relationships and keywords. After the first extraction, the system performs multiple rounds of gleaning optimization by parsing the results from the LLM to further improve the accuracy and completeness of the elements extraction.

After the extraction of elements is completed, the system will deduplicate the results. It ensures the uniqueness of the extracted entities, relationships and keywords. The extracted entities and relationships then are inserted into the pre-defined “Entity-Relationship GDB” graph database to ensure efficient data storage and fast query. In addition, the system will generate vector embeddings for each entity, relationship, and keyword, storing them in the “Entities VDB”, “Relationship VDB”, and “Keyword VDB” vector databases to provide rich semantic information for subsequent semantic retrieval and query.

This element extraction process forms the foundation of AcademicRAG’s knowledge representation, enabling precise semantic mapping of academic content while maintaining computational efficiency through strategic caching and deduplication mechanisms.

4.4 Data pipeline evaluation

4.4.1 Experiment settings

In order to evaluate the effectiveness of the AcademicRAG data pipeline in processing various types of academic content, we designed a comprehensive set of experimental methods to analyze its performance. Since the core question of the study is how to build a data pipeline that can process diverse academic content while preserving semantic relationships, we chose two types of representative academic texts as input: research articles and course syllabi.

For research papers, we selected 3 articles from the Machine Learning and Acoustics areas, which have citation relationships with each other. These papers are on average 6 pages (about 7000 tokens) in length and contain complex concepts, formulas, and citation structures. For course syllabi, we collected 5 course syllabi for acoustics and its fundamentals from KTH Royal Institute of Technology, covering multiple levels from entry-level to advanced graduate programs.

For all entity and relationship extraction processes in the data pipeline evaluation, we utilized DeepSeek-V3-0324 [35] as the large language model component of AcademicRAG. This model was selected for its robust performance in academic content analysis and specialized knowledge extraction capabilities, ensuring high-quality knowledge graph construction from both research articles and course syllabi.

The expected output of the AcademicRAG data pipeline is a series of structured databases, the core of which is the Knowledge Graph database, which is automatically generated from a given document processed through the pipeline by means of an LLM. Our evaluation therefore focuses on analyzing the quality of the knowledge graphs generated from these two types of academic texts.

Considering the shortage of widely applicable standardized assessment frameworks for knowledge graph evaluation, we adopted a qualitative approach to assess the quality of knowledge graphs generated from these two types of academic texts. Our qualitative methodology includes the following aspects:

- **Knowledge graph visualization and analysis:** The generated knowledge graphs

are visualized to intuitively observe the structural features, connectivity and conceptual hierarchical relationships of the graphs.

- **Semantic structure assessment:** Assess the accuracy and completeness of the semantic relationships in the knowledge graph through expert review, with special attention to whether the associations between domain-specific concepts are correctly captured.
- **Cross-document association analysis:** Examine how the data pipeline establishes conceptual connections between different documents and assess its ability to integrate dispersed knowledge.
- **Case study:** Select a specific academic concept, track its representation in the knowledge graph, and analyze how the data pipeline extracts and constructs relevant knowledge structures for the concept from the raw text.

Finally, we comparatively analyze the differences in structure and content between the knowledge graphs generated from research articles and course syllabi, assessing the adaptability of the data pipeline to handle different text formats.

Through this qualitative analysis approach, we are able to comprehensively assess the ability of the AcademicRAG data pipeline to process different academic texts, especially its performance in preserving semantic relationships. The experimental results will provide us with insights on how to design flexible data pipelines to process diverse academic content, therefore answering the second research question.

In addition, the results of the evaluation of these two different academic texts will also directly inform our practical application scenarios. The knowledge graph evaluation of research paper generation will guide us to optimize the academic literature assistant application, especially in handling complex citation relationships and cross-domain conceptual connections; while the knowledge graph analysis of course syllabi generation will directly influence the design of course discovery system, helping us to improve the prerequisite course recommendation algorithms and learning path planning functions.

4.4.2 Knowledge graph evaluation for research articles

In order to systematically assess the ability of the pipeline to handle unstructured academic data, academic papers are selected as the evaluation object in this study. Although academic papers have some fixed structural elements such as abstracts and references, these structures are different between papers. Furthermore, the whole paper is often regarded as a continuous text rather than a structured segmentation from the text processing perspective of the LLMs. Therefore, it is reasonable to treat academic papers as unstructured data. The long length of the academic papers is also a significant test of the pipeline's ability to capture complex relationships holistically.

In terms of specific sample selection, we adopt three papers with citation links in the field of machine learning and acoustics to construct the evaluation dataset: The landmark "Attention Is All You Need" [36] (P1), which proposes the Transformer

architecture and serves as a basic reference paper in this study. The application paper in the field of acoustics, "Transformer Transducer" [37] (P2), which directly cites paper P1. Another acoustics paper, "Transformer-based Streaming ASR with Cumulative Attention" [38] (P3), cites both P1 and P2. This progressive citation relationship provides a multi-level testing scenario for the evaluation process to deal with complex academic associations.

In order to effectively match the textual features of academic papers and accurately evaluate the performance of the pipeline, the following settings are carried out during the experiment:

(1) In view of the semantic characteristics of academic papers, the key entity categories to be extracted are clearly defined, including "Method", "Concept", "Paper", "Author" and "Model", etc., in order to ensure that the extraction task is in line with the semantic structure of academic literature.

(2) Considering that academic papers are usually long in length, this study adopts a larger chunk size to ensure that the LLM can process the complete context of the whole paper at one time. To alleviate the problem of information extraction performance degradation that may be brought about by increasing chunk size, we increased the number of iterations for extraction, which improves the accuracy and completeness of entity recognition.

These settings are designed to ensure the accuracy of information extraction to eliminate the influence of other factors in the experiment.

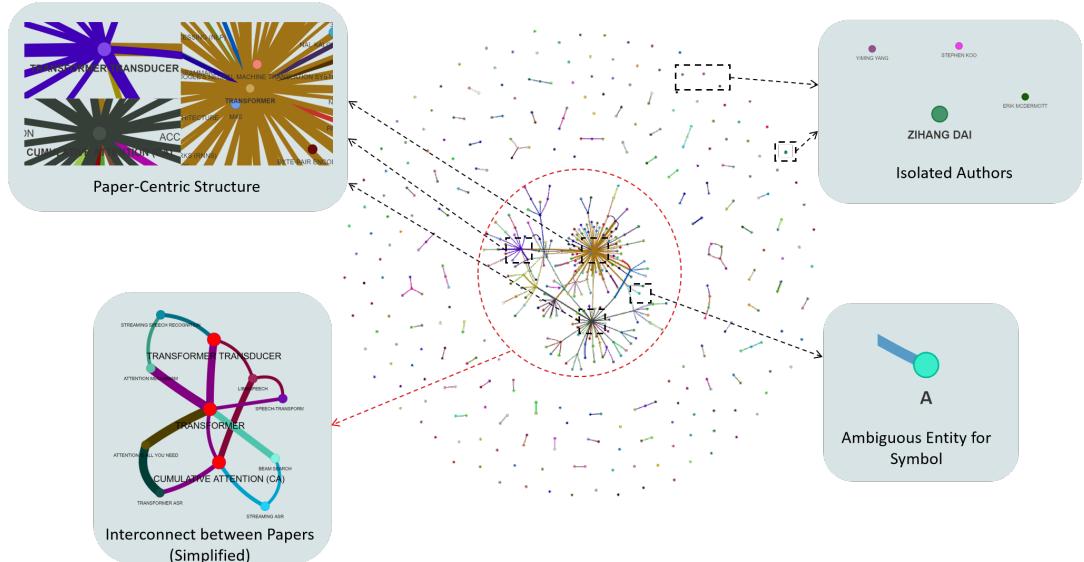


Figure 5: Knowledge Graph Visualization and Analysis for Academic Articles

Figure 5 shows the generated knowledge graph and some areas of interest. The knowledge graph successfully realizes a structured paper-centered representation, which completely covers the core content of the three target documents (P1, P2, and P3). One can see that the graph presents clear clustering features, in which the key elements of each of the three papers form core nodes and are interconnected through semantic associations. It is worth noting that P1 ("TRANSFORMER"), which is

the base paper, exhibits significant pivotal characteristics, and its node connection density is significantly higher than that of P2 and P3. This phenomenon is in line with our expectation because the nodes in P2 ("TRANSFORMER TRANSDUCER") and P3 ("CUMULATIVE ATTENTION") involving the Transformer architecture are associated with P1, confirming the fundamental role of P1 in this research area.

Considering the details in the knowledge graph, we have the following findings:

- **Links between Papers:** The simplified subgraph in the lower left corner of Figure 5 shows that the system has successfully captured the deep connections between the three papers. The core nodes "TRANSFORMER", "TRANSFORMER TRANSDUCER" and "CUMULATIVE ATTENTION" are directly connected through the shared Transformer architecture. Moreover, the system also identifies potential implicit associations, e.g., both P2 and P3 are experimented with the LibriSpeech dataset, and this common feature creates additional connections in the graph, demonstrating the framework's ability to discover potential associations.
- **Isolated Entity Problem:** There are a number of isolated nodes in the graph (e.g., "ZIHANG DAI" labeled in the figure), which mainly originate from the reference section. When the entities of cited papers are split and categorized into other clusters, their corresponding author nodes are isolated due to the lack of a title node.
- **Ambiguous Mathematical Symbol:** Semantic disambiguation of formula symbols (e.g., N , A , etc.) remains challenging. Since the same symbols may refer to different concepts in different papers, this leads to entity confusion and introduces noise. Such ambiguous symbols can harm the performance when generating answers.

4.4.3 Knowledge graph evaluation for course syllabi

The selection of data sources for course syllabi is an important part of validating the AcademicRAG data pipeline's ability to process semi-structured academic texts. Five acoustics and its foundation courses from the KTH Royal Institute of Technology (Sweden) were selected for evaluation. These courses represent a clear knowledge pathway as shown in Table 4.

Table 4: Selected KTH Courses for Knowledge Graph Evaluation

Course Level	Course Code	Course Name
Foundation Courses	SF1624	Algebra and Geometry
Foundation Courses	SF1625	Calculus in One Variable
Basic Acoustics Course	SK1120	Waves
Basic Acoustics Course	DT1175	Sound
Advanced Acoustics Course	DT2213	Musical Communication and Music Technology

There are clear knowledge dependencies and lines of learning between these courses, which are clearly noted in the course syllabi, making them well suited for knowledge graph construction and assessment. It is worth noting that these course syllabi are in HTML format and the information is organized in a more dispersed and fragmented manner, which provides a good test scenario for the knowledge extraction capability of the data pipeline.

When initializing the data pipeline, we configured it specifically for the structural characteristics of the course syllabi. First, multiple entity types were defined, including 'Course', 'Knowledge Concept', 'Skill', 'Person', 'Assessment', 'Material', 'Learning Tool', and 'Grading Scale'. Second, we increased chunk size to ensure that each course outline is not overly segmented and that each chunk contains the course name information at the beginning of the outline. At the same time, we increased the gleaning parameter appropriately to improve the completeness of information extraction.

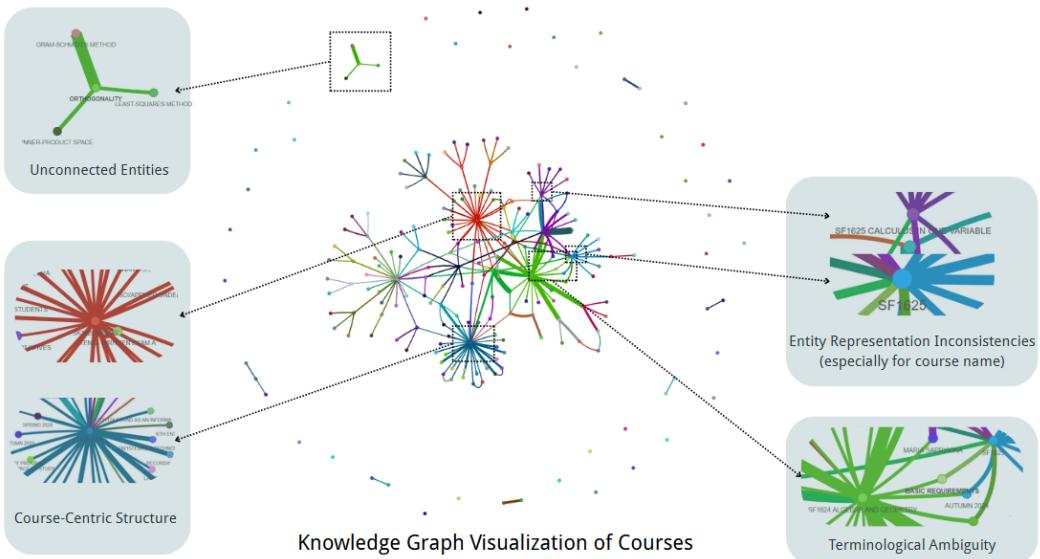


Figure 6: Knowledge Graph Visualization and Analysis for Course Syllabi

The results of the knowledge graph visualization are presented in Figure 6. Firstly, the knowledge graph exhibits a clear star-shaped structure centered on the course. Specifically, there are dense connections between the course entity and multiple related entities. This reflects the central role of the course in the knowledge graph. It demonstrates well-structured knowledge representation in graph databases.

However, the visualization presented in Figure 6 also reveals several key challenges for data pipelines when dealing with course syllabi:

- **Inconsistent entity representations:** The figure illustrates the problem that the same course (e.g. SF1625) may be represented differently in different contexts. Sometimes it contains the full name and sometimes only the course code, which makes entity association more difficult.

- **Terminology ambiguity:** The data pipeline is ambiguous when dealing with terms such as 'BASIC REQUIREMENTS'. It can be interpreted as course knowledge requirements or prerequisites, which affects the accuracy of the relationship.
 - **Unconnected entities:** The figure shows isolated clusters of entities such as 'GRAM-SCHMIDT METHOD', 'ORTHOGONALITY' and 'LEAST-SQUARES METHOD'. They are conceptually related but not connected to the main curriculum network.

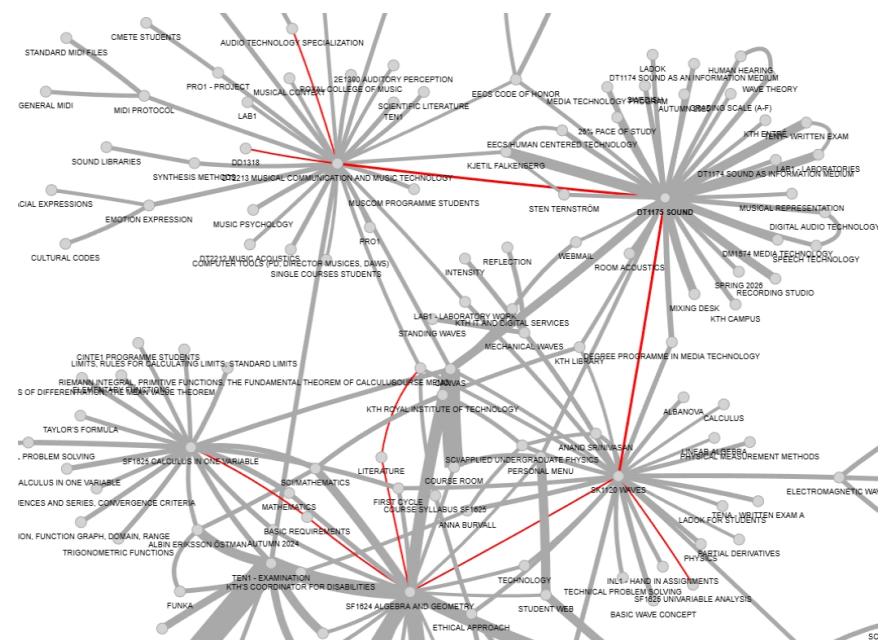


Figure 7: Visualization of Course Prerequisite Chain

Despite these challenges, the data pipeline successfully extracted key knowledge structures from course syllabi. To illustrate this achievement, we merged entities representing the same course but with different names, (such as "SF1625" and "SF1625 CALCULUS IN ONE VARIABLE") into unified representations. The resulting knowledge graph visualization is shown in the figure. We highlight relationships containing terms like "prerequisite," "required," and "fundamental" in red. This visualization clearly reveals the prerequisite chain—the sequential relationships between well-defined connections. A clear representation of prerequisite relationships is essential for constructing a well-structured course knowledge graph database. Such a database can effectively support educational pathway planning and course recommendation systems.

4.5 Summary of AcademicRAG data pipeline

The AcademicRAG data pipeline implements a hierarchical, three-tier architecture designed to transform heterogeneous academic content into structured knowledge

representations. Its principal components—Document Process Flow, Document Index Flow, and Element Extraction Flow—work in concert to process raw documents, segment them into semantic chunks, and extract entities, relationships, and keywords utilizing large language models. This architecture enables fault-tolerant processing through comprehensive state management, prevents duplicate entries via content hashing, and minimizes computational overhead through strategic LLM caching mechanisms.

Experimental evaluations conducted on diverse academic corpora—including research articles with citation networks and course syllabi with prerequisite relationships—demonstrated the pipeline’s capacity to construct semantically rich knowledge graphs that preserve domain-specific relationships. While successfully capturing complex interconnections between papers and course learning pathways, the system exhibited certain limitations, including entity naming inconsistencies, isolated reference nodes, and occasional terminological ambiguities that impacted graph connectivity.

The pipeline’s most significant contribution lies in its support for incremental knowledge integration without requiring complete database reconstruction—a substantial improvement over previous graph-based frameworks like GraphRAG [19] that necessitated comprehensive regeneration of community structures. This capability, coupled with its adaptability to multiple document formats and semantic extraction patterns, establishes a robust foundation for downstream applications such as course discovery systems and research literature assistants, while maintaining computational efficiency through targeted retrieval strategies rather than exhaustive community report generation.

5 Application: Course Discovery System using AcademicRAG

The course discovery system addresses critical information retrieval challenges in academic institutions by leveraging the AcademicRAG knowledge graph framework to serve two distinct user groups: students and educational administrators. Students confront multifaceted obstacles in navigating complex curricular structures, requiring semantic course matching for academic-goal alignment, prerequisite relationship mapping for optimized degree progression, and knowledge gap identification to ensure program readiness. Concurrently, faculty and administrators require comprehensive curriculum analytics to detect content redundancies, evaluate interdisciplinary connections, and validate logical knowledge progression across degree programs—needs unmet by traditional static catalogs and keyword-based search systems that fail to capture latent course relationships.

To overcome these limitations, the system implements knowledge graph-enhanced search capabilities that transform course discovery into a semantically rich, context-aware process. Core objectives include:

- Enhancing discovery through natural language query processing that reveals implicit course connections.
- Generating personalized learning trajectories by integrating student backgrounds with curricular requirements.
- Providing curriculum optimization tools for identifying content overlaps and strengthening pedagogical sequencing.

These objectives directly address inefficiencies in conventional methods while enabling data-driven educational decision-making.

Anticipated outcomes manifest in two interrelated domains. For students, the system facilitates systematic knowledge gap mitigation and accelerated career-path alignment through optimized learning sequences, reducing resource waste from suboptimal course selection. For educators, it offers analytical tools to uncover implicit course dependencies, evaluate curricular coherence, and implement evidence-based revisions to improve teaching quality and resource allocation. This dual benefit enhances both individual learning outcomes and institutional educational efficacy.

5.1 Adaptation of AcademicRAG for course syllabi

Course syllabi in higher education, such as DT1175 "Sound" from KTH [39], are typically semi-structured documents combining clearly labeled metadata (e.g., course codes, titles, credits) with natural language descriptions. While some components, like course identifiers and assessment formats, follow predictable patterns, their presentation can vary across institutions, posing challenges for standardized extraction.

Key sections of a syllabus include learning outcomes, course content, prerequisites, and assessment methods. Learning objectives often follow an implicit hierarchy, which

benefits knowledge graph construction but requires accurate parsing. Prerequisites convey inter-course dependencies but are described inconsistently, making semantic interpretation more complex. Similarly, assessment structures differ widely in composition and grading formats.

Additional information—such as textbooks, teaching language, and course equivalency notes—is highly unstructured and context-dependent. These characteristics necessitate the integration of structured data extraction and natural language understanding within the AcademicRAG framework to support reliable entity recognition and relationship modeling for educational knowledge graphs. Therefore, it is also crucial to make corresponding structural refinements based on the characteristics of the course syllabi.

Firstly, to preserve the structural and semantic integrity of course syllabi, the chunking strategy in AcademicRAG was enhanced to segment documents based on recognizable section boundaries rather than fixed token windows. This preserves context within each chunk and strengthens inter-entity associations. Prefixes indicating course identity are systematically added to each chunk to support disambiguation and cross-chunk reasoning. A fallback mechanism reverts to traditional chunking when structure is insufficient, ensuring robustness across varied formats.

Secondly, prompt engineering was customized to better extract and organize educational content. In the entity-relation extraction phase, prompts were designed to encourage explicit inter-entity linking and to anchor all entities to the main course. Domain-specific examples were included to guide the model in recognizing typical structures like prerequisites or grading methods. In the query phase, prompts leverage the hierarchical structure of syllabi to help the model interpret user intent and generate semantically consistent answers aligned with syllabus content.

Finally, the entity schema was optimized for course syllabi by replacing generic types with nine domain-specific categories, such as "Course Name", "Knowledge Concept", and "Grading Scale". This design ensures comprehensive coverage of course-relevant information and supports fine-grained knowledge representation. It enables accurate reasoning over course content, such as recommending learning paths or identifying prerequisite structures, thereby improving the system's ability to retrieve and organize educational knowledge.

Table E1 illustrates the nine core entity types specifically designed for course syllabi content, each of which targets a different conceptual dimension in the course syllabi and together form a complete framework for representing educational knowledge.

5.2 System design

The course discovery system is built on the AcademicRAG framework and operates in two phases: an offline phase for data ingestion and knowledge graph construction, and an online phase for real-time query processing. University course syllabi in HTML format are crawled and cleaned to extract structured and semi-structured information. The processed text is then segmented into meaningful chunks with course identifiers prefixed, enabling precise context tracking during knowledge extraction.

The knowledge graph is built using AcademicRAG’s pipeline for entity and relation extraction, with a dual-database architecture. A graph database stores structured entities and interconnections to support logical reasoning and traversal, while a vector database stores semantic embeddings to enable similarity-based retrieval. This hybrid storage design allows the system to handle both exact and fuzzy queries efficiently.

During online querying, the system applies prompt-guided retrieval and reasoning. User intent is first classified, then matched against a keyword vector index to extract relevant context. Guided by these cues, the system performs layered retrieval from the graph and vector databases, feeding results to a language model for answer generation. A natural language conversational interface and a graphical exploration interface are provided for user interaction, both supported by a unified back-end to ensure consistent, semantically coherent responses.

5.3 Evaluation

5.3.1 Evaluation setting

To effectively assess the performance of the course discovery system, we adopted a case study approach involving two primary user groups: students and faculty members. The evaluation was grounded in real institutional data—specifically, twenty-nine acoustics-related course syllabi from KTH Royal Institute of Technology—spanning various academic levels and specialization areas within the field of acoustics.

Throughout the course discovery system evaluation, DeepSeek-V3-0324 [35] served as the underlying large language model for AcademicRAG’s query processing and response generation components. This advanced model was particularly effective at interpreting complex academic queries and generating pedagogically sound responses that addressed the distinct information needs of both student and faculty user groups.

The methodology was designed to emulate authentic academic scenarios. To this end, we conducted a series of interviews with current students and faculty members in the acoustics program at KTH. These conversations revealed common challenges and information needs related to course selection, curriculum navigation, and program development. Based on the insights gathered, we constructed a set of representative queries that reflect the distinct objectives and perspectives of each user group.

Student perspective For the student evaluation, we considered a prototypical background typical of those enrolled in acoustics-related programs at KTH. Such students generally possess a strong foundation in mathematics, physics, and digital signal processing, often acquired through undergraduate studies in Electrical Engineering, Media Technology, or Engineering Physics. They are proficient in MATLAB and have completed relevant coursework in wave physics, linear algebra, and Fourier analysis. Exposure to introductory acoustics modules and laboratory-based learning equips them to articulate structured academic queries and navigate advanced course offerings.

Given this profile, the student-focused evaluation addressed the following key tasks:

1. **Learning path planning:** "I'm interested in audio engineering. Which courses should I take, in what order, to build expertise in this area while meeting all prerequisites?"
2. **Knowledge gap identification:** "What additional courses and knowledge do I need before enrolling in DT2300 Sound in Interaction?"
3. **Course recommendation:** "Which courses cover both acoustic theory and signal processing with practical laboratory components that would complement my current knowledge of basic wave physics?"

Faculty perspective For faculty and educational administrators, the evaluation focused on more analytical and structural queries pertaining to curriculum design and optimization:

1. **Program structure analysis:** "Classify all courses based on their offering term (Spring or Autumn). Then further categorize them by academic level (Bachelor's or Master's) and present the classification in a 2×2 matrix format."
2. **Content overlap assessment:** "How many intended learning outcomes are common between each pair of courses? Present the response in a matrix for each of the three aspects."
3. **Curriculum integration potential:** "Based on the answers to questions 1-2, can you suggest viable and feasible ways to integrate/merge courses with high overlapping relevance?"

By employing this multi-faceted evaluation framework—rooted in authentic user needs and institutional data—we aim to rigorously assess the system's capabilities and limitations in supporting data-driven educational decision-making from both student and faculty perspectives.

5.3.2 Knowledge graph evaluation

The course syllabi-derived knowledge graph exhibits a semantically rich and structurally coherent network, featuring clear clusters and hierarchical layers. As shown in Figure 8, the graph contains a densely connected core with peripheral satellite clusters, aligning with the natural organization of academic curricula—where foundational concepts connect with specialized or supporting elements.

The graph comprises 1,211 nodes and 2,252 edges, with an average degree of 3.72. Despite a low global density of 0.0031, indicating overall sparsity, the graph demonstrates strong local cohesion. The density is computed as:

$$\text{Density} = \frac{2E}{N(N - 1)}, \quad (8)$$

where E is the number of edges and N is the number of nodes.

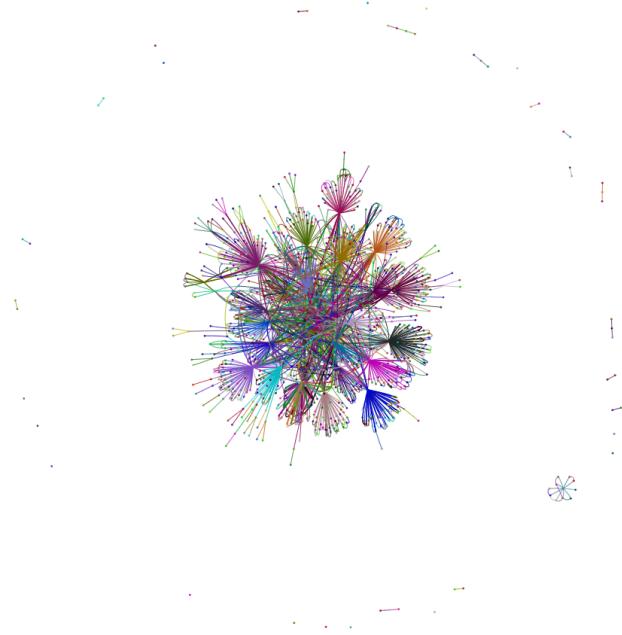


Figure 8: Visualization of Course Knowledge Graph

To assess local connectivity, we calculate the average clustering coefficient:

$$\bar{C} = \frac{1}{N} \sum_{i=1}^N \frac{2e_i}{k_i(k_i - 1)}, \quad (9)$$

where k_i is the degree of node i , and e_i is the number of links between its neighbors. Nodes with $k_i < 2$ are assigned $C_i = 0$ by convention. The computed average clustering coefficient is 0.3971, confirming the presence of tightly knit local structures.

Among the 34 connected components, the largest encompasses 1,146 nodes—94.63% of the graph—indicating a well-integrated knowledge structure across course documents.

Entity and Relation Statistics. Table 5 shows the entity type distribution. "Knowledge Concept" dominates, accounting for nearly half of all nodes, while "Skill" and "Assessment" reflect learning outcomes and evaluation criteria. The "Others" category suggests flexibility in capturing schema-external but semantically useful entities.

Out of 2,252 edges, 2,177 possess unique relation descriptions—such as prerequisite chains, instructor mappings, and assessment dependencies—highlighting high-resolution relation extraction.

Furthermore, all nodes are enriched with complete attribute annotations, yielding 100% attribute completeness and demonstrating the robustness of the entity extraction module.

Table 5: Entity type distribution in the course syllabi knowledge graph.

Entity Type	Count	Percentage
Knowledge Concept	552	45.58%
Skill	225	18.58%
Assessment	129	10.65%
Person	61	5.04%
Learning Tool	59	4.87%
Learning Material	52	4.29%
Credits	45	3.72%
Course Name	44	3.63%
Grading Scale	6	0.50%
Others	38	3.14%

Centrality Metrics. To examine graph dynamics, we compute two key node-level centrality metrics. The degree centrality for a node v is given by:

$$C_D(v) = \frac{\deg(v)}{N - 1}, \quad (10)$$

and measures how well a node is connected.

Betweenness centrality assesses a node's role in information flow:

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}, \quad (11)$$

where σ_{st} is the number of shortest paths between s and t , and $\sigma_{st}(v)$ is the count passing through v .

The graph's average degree centrality and betweenness centrality are 0.0031 and 0.0021 respectively, indicating a balanced topology without over-centralization—beneficial for downstream applications like recommendation and retrieval.

The evaluation demonstrates AcademicRAG's effectiveness in constructing accurate and comprehensive knowledge graphs from educational syllabi. The framework successfully preserves the semantic complexity of course relationships while maintaining structural coherence. This knowledge representation provides a robust foundation for downstream applications such as personalized learning path recommendations and curriculum optimization tools.

5.3.3 Student perspective evaluation

We conducted a comprehensive evaluation of the system's effectiveness from the student perspective, focusing on three core challenges that represent critical pain points in curriculum navigation. The case studies, summarized in Tables F1–F3, demonstrate how the system leverages its knowledge graph architecture to address complex educational needs.

In the learning path planning scenario (Table F1), the system demonstrated sophisticated curriculum mapping capabilities by generating a logical progression

that respects educational hierarchies. Beginning with foundational courses, the path systematically advanced through increasing levels of specialization. The system exhibited contextual reasoning by providing pedagogical justifications that connected course concepts across the curriculum sequence, revealing its ability to model not just what courses should be taken, but why they build upon each other in meaningful ways.

The knowledge gap identification capabilities (Table F2) revealed the system's analytical strengths in differentiating between requirement categories and contextualizing abstract requirements. By distinguishing between mandatory and recommended prerequisites while explaining their relevance to course content, the system transformed institutional requirements into personalized guidance. This capability demonstrates how the knowledge graph approach enables educational systems to bridge the gap between administrative data and student-centered learning support.

The multi-dimensional recommendation task (Table F3) highlighted the system's capacity for balanced criteria evaluation across theoretical and practical dimensions. Rather than performing simple keyword matching, the system identified offerings that authentically integrated multiple academic elements, revealing its understanding of deeper educational relationships. This ability to evaluate courses across multiple dimensions simultaneously represents a significant advancement over traditional catalog systems that rely on one-dimensional filtering.

Across all scenarios, the system consistently transformed educational data into contextually meaningful guidance. Its strength lies in synthesizing relational knowledge - connecting course sequences, mapping conceptual hierarchies, and aligning theoretical foundations with practical applications. These capabilities demonstrate how graph-based educational systems can address the multifaceted challenges of academic planning by modeling the complex interconnections inherent in curriculum structures.

5.3.4 Faculty perspective evaluation

The evaluation demonstrates the system's capacity to support curriculum management and optimization through both broad structural analysis and detailed content comparison. The assessment focused on three critical administrative functions that traditionally require extensive manual review and domain expertise. The evaluation results are illustrated in Tables F4–F7.

When tasked with program structure analysis, the system demonstrated advanced categorization capabilities by generating a comprehensive classification matrix as shown in Table F4. This analysis revealed significant distribution patterns across different academic levels and terms, highlighting potential gaps in curricular pathways and scheduling imbalances. The system effectively processed and organized complete attribute information for each course, providing administrators with a solid foundation for strategic planning.

The system exhibited even more sophisticated capabilities in content analysis when evaluating relationships between curriculum components. As shown in Tables F5 and F6, the system identified specific pedagogical alignments that would be more time-consuming to detect through manual review. This analysis went beyond simple keyword

matching to identify conceptual equivalences despite variations in terminology. This capability offers curriculum committees unprecedented insight into content redundancies and progression patterns across traditionally separated academic units.

The system's most advanced administrative function emerged in its ability to synthesize insights from both structural and content analyses to generate actionable reorganization recommendations, as demonstrated in Table F7. Rather than simply identifying overlaps, the system produced specific integration proposals with clear rationales and implementation methodologies. These recommendations demonstrate how the system transforms raw data into structured academic planning strategies that respect both content relationships and practical implementation constraints.

Across all administrative scenarios, the system consistently converted complex curricular relationships into actionable insights. Its strength lies in multilevel analysis—simultaneously mapping broad organizational structures while identifying detailed content connections. This dual capability helps administrators identify both strategic gaps and operational redundancies. Additional administrative and institutional constraints, such as faculty availability, budget limitations, accreditation requirements, or classroom space, could further inform the context of query prompts. Including such constraints would potentially enhance the practical relevance of generated responses by factoring in implementation feasibility, resource allocation implications, and compliance requirements. Most importantly, the implementation recommendations demonstrate how knowledge graph-based reasoning can bridge the gap between analytical insights and practical administrative action, providing not just what should change, but concrete pathways for institutional implementation that respect both academic coherence and operational realities.

6 Discussion

The Discussion section critically examines the empirical findings and theoretical contributions of the AcademicRAG framework, contextualizing its performance within both technical and educational domains. By synthesizing results from the framework evaluation in Chapter 3, data pipeline assessment in Chapter 4, and course discovery system implementation in Chapter 5, this chapter identifies key patterns that illustrate the framework's effectiveness while acknowledging its limitations. The discussion is structured around three main themes: the technical effectiveness of the AcademicRAG architecture and data pipeline, the educational impact of the Course Discovery System implementation, and current limitations with corresponding avenues for future research. This multifaceted analysis not only validates the framework's contributions to academic resource discovery but also provides a foundation for advancing knowledge graph-enhanced retrieval systems in educational contexts.

6.1 Effectiveness of AcademicRAG framework and pipeline

6.1.1 Framework performance and evaluation considerations

The evaluation results presented in Section 3.5 demonstrate that AcademicRAG consistently outperforms baseline frameworks across four dimensions and two domains. According to Table 2, in the agriculture domain, AcademicRAG achieved win rates of 57.2%, 54.8%, and 52.4% against NaiveRAG, LightRAG, and GraphRAG respectively, while in computer science, these advantages were even more pronounced at 77.5%, 56.6%, and 53.6%. This pattern of superior performance across diverse academic fields suggests that the framework's architectural innovations—particularly its subgraph-based retrieval mechanism and clue-guided keyword generation—effectively address the complex semantic relationships inherent in academic knowledge structures.

However, these encouraging results must be contextualized within the limitations of current RAG evaluation methodologies. While our LLM-based pairwise comparison approach represents the state-of-the-art for assessing high-level semantic tasks, the subjective nature of dimensions like "comprehensiveness" and "empowerment" introduces interpretive challenges. Unlike traditional information retrieval metrics such as precision and recall, these qualitative dimensions rely on the reasoning capabilities of the evaluator model itself. Furthermore, the synthetic nature of our user personas and queries, though carefully designed to reflect authentic academic information needs, may not capture the full complexity of real-world scenarios or the evolving requirements of genuine academic users.

The observed performance disparity between agriculture and computer science domains warrants further investigation. The significantly higher win rates in computer science suggest that domain characteristics—perhaps related to terminology standardization, conceptual hierarchy, or citation patterns—may influence the framework's effectiveness. This domain-dependent performance variation highlights the need for adaptive tuning of the framework's parameters, particularly in chunk size, keyword extraction thresholds, and graph construction methodologies, to accommodate the

unique semantic structures of different academic fields.

6.1.2 Technical advantages of subgraph retrieval and clue mechanisms

The ablation studies in Appendix C offer compelling evidence for the individual contributions of both subgraph retrieval and clue-guided keyword generation to AcademicRAG’s performance. The removal of clues (-clues variant) resulted in diminished diversity scores, with win rates against GraphRAG dropping from 50.8% to 48.4% in agriculture and from 46.8% to 43.0% in computer science. This performance degradation could be attributed to the hallucination effect observed when keywords are generated without contextual anchoring. Without clues extracted from the knowledge graph, the LLM relies entirely on its parametric knowledge to generate keywords, resulting in semantic drift where the generated keywords may have limited correspondence with the actual content indexed in the knowledge graph.

Subgraph retrieval offers an intuitive solution to the connective shortcomings of one-hop neighbor approaches. By exploring the complete set of paths between matched nodes, AcademicRAG captures both direct and higher-order semantic relationships that might otherwise remain undiscovered. Consider a practical example from our case study in Appendix D, where the query about hybrid models for user segmentation benefited from subgraph retrieval by connecting disparate methods like hierarchical clustering and ensemble techniques. The -subgraph variant, which relied on one-hop neighbors, surfaced more immediate connections but missed these valuable methodological bridges, resulting in less comprehensive responses despite potentially greater surface-level diversity.

The synergistic relationship between these mechanisms creates a virtuous cycle of retrieval precision. Clues first narrow the semantic search space to relevant domains within the knowledge graph, increasing the likelihood that low-level keywords will match meaningful entities. The subsequent subgraph construction then expands this focused retrieval laterally, exploring emergent connections between these high-confidence entity matches. This balanced approach—constraining, then expanding—mitigates both the narrow tunnel vision of strict keyword matching and the diffuse irrelevance of unconstrained graph exploration, resulting in more coherent and contextually appropriate information retrieval.

6.1.3 Versatility and robustness of the AcademicRAG data pipeline

The AcademicRAG data pipeline demonstrates remarkable versatility in processing diverse academic content types, as evidenced by our comprehensive knowledge graph evaluations in Section 4.4. The pipeline successfully processed both unstructured research articles and semi-structured course syllabi, generating semantically rich knowledge graphs with well-defined entity-relationship structures. For research papers, the pipeline effectively captured citation networks and conceptual relationships between papers, with clear clustering around fundamental papers like "Attention Is All You Need." For course syllabi, it reconstructed prerequisite chains and learning outcome relationships, enabling sophisticated educational pathway analysis.

This cross-domain adaptability stems from several architectural decisions. First, the pipeline's modular design, with distinct document processing, indexing, and extraction flows, allows for targeted customization at each processing stage without disrupting the overall workflow. Second, the LLM-driven entity-relationship extraction with domain-specific prompt engineering enables flexible adaptation to different academic contexts without requiring extensive rule-based reconfiguration. Third, the incremental processing capability, achieved by eliminating community-based structures present in GraphRAG, allows the system to efficiently incorporate new knowledge without rebuilding existing database structures.

The pipeline demonstrates robust performance even when confronted with challenging semantic features common in academic texts. For research papers, it successfully navigated complex citation structures and technical terminology, though some limitations emerged in handling mathematical symbols and isolated reference nodes. For course syllabi, while entity naming inconsistencies and terminology ambiguities occasionally affected graph connectivity, the system nevertheless captured core educational relationships accurately enough to support sophisticated query responses. These results suggest that the pipeline's fundamental architecture is sufficiently robust to accommodate domain-specific extensions while maintaining high performance across diverse academic contexts.

6.2 Educational applications and impact

6.2.1 Student-centered learning support and pathway planning

The Course Discovery System demonstrates considerable effectiveness in addressing complex academic information needs through its knowledge graph-enhanced retrieval mechanism. As evidenced by the case studies presented in Section 5.3 and Appendix F, the system successfully transformed raw syllabus data into structured, semantically rich responses that directly addressed multifaceted educational queries. For learning path planning, which is shown in Table F1, the system constructed coherent course sequences that not only respected prerequisite chains but also articulated the pedagogical relationships between courses—explaining. This capacity for educational rationale generation surpasses traditional catalog systems, which typically provide only flat listings without contextualizing how courses relate to one another in knowledge progression.

The system's performance in prerequisite identification further illustrates its effectiveness in translating institutional requirements into actionable guidance. As shown in Table F2, when analyzing prerequisites for "DT2300 Sound in Interaction", the system accurately distinguished between mandatory and recommended requirements while providing contextual explanations for why specific competencies were necessary. This dual-layered approach—combining quantitative requirements (e.g., "6 credits of programming") with qualitative justifications (e.g., "necessary for interactive sound design projects")—transforms administrative data into meaningful learning guidance. The integration of course content analysis with prerequisite relationships enables students to understand not just what courses they need, but why they need them—a

significant advancement over conventional prerequisite checking systems.

The multi-dimensional recommendation capabilities demonstrated in Table F3 further highlight the system's sophisticated understanding of curriculum structures. When tasked with identifying courses that combined both acoustic theory and signal processing with practical laboratory components, the system went beyond simple keyword matching to generate recommendations that authentically integrated these dimensions. For each recommended course, such as DT2212 Music Acoustics and SD2111 Engineering Acoustics, the system provided a comprehensive analysis of how theoretical concepts connected with specific laboratory applications. This ability to evaluate courses across multiple technical dimensions simultaneously represents a significant advancement over traditional course catalog systems that typically rely on single-dimension filtering. By identifying these complex integration patterns, the system enables students to make more informed decisions aligned with both their theoretical interests and practical skill development goals.

From a student perspective, this comprehensive approach to educational guidance significantly enhances the learning experience by providing personalized, contextually relevant information that connects individual course selection to broader educational objectives. By transforming fragmented institutional data into coherent knowledge structures, the system empowers students to make informed decisions that optimize their academic pathways while building meaningful connections between theoretical foundations and practical applications.

6.2.2 Curriculum analysis and academic program optimization

The Course Discovery System demonstrates particular strength in surfacing implicit curriculum relationships that remain invisible to traditional information systems. As shown in Table F4, the system performed a comprehensive structural analysis that classified courses by both academic level and term offering, revealing significant organizational patterns across the acoustics curriculum. The accuracy of this classification was verified against the official KTH course catalog, confirming the system's reliability in extracting and organizing institutional data.

As illustrated in the content overlap assessment (Tables F5 and F6), the system demonstrated its ability to identify conceptual similarities across course descriptions despite terminological variations. Rather than relying on exact text matching, the system synthesized thematic categories of learning outcomes, such as "Application of mathematical models and methods to describe sound and vibration quantities" - a concept it determined was present in eight different course pairs, though expressed with different terminology in each syllabus. This semantic abstraction capability enables the system to recognize conceptually equivalent learning objectives even when courses use different vocabularies to describe similar competencies—a significant advancement over conventional catalog systems that typically depend on exact keyword matching. While the accuracy of these synthesized categorizations presents its own evaluation challenges, the approach demonstrates a promising direction for identifying cross-curricular connections that remain invisible in traditional analyses.

The curriculum integration recommendations presented in Table F7 demonstrate

the system’s ability to translate analytical insights into actionable implementation strategies. By synthesizing both structural and content analyses, the system generated five specific course integration proposals that address identified overlaps while respecting institutional constraints. Each recommendation includes a clear rationale grounded in specific learning outcome similarities and a tailored integration methodology. For example, the system proposed combining DT2212 (Music Acoustics) and DT1175 (Sound) through a sequential integration that leverages their shared focus on sound behavior in spaces, while suggesting a modular approach for SD2111 (Engineering Acoustics) and SD2180 (Non-linear Acoustics) based on their progression from linear to non-linear phenomena. The system further demonstrates its practical orientation by providing implementation guidelines that include curriculum review processes, student feedback mechanisms, and faculty collaboration strategies—transforming abstract analyses into executable administrative workflows. This translation of data analysis into actionable planning represents a significant advancement over traditional curriculum mapping tools that typically provide only descriptive documentation.

From a faculty perspective, the system’s ability to uncover these hidden curriculum relationships represents a transformative advancement in educational governance. By revealing conceptual connections and learning pathways that typically remain obscured by administrative boundaries and terminological variations, the Course Discovery System enables evidence-based curriculum development that aligns with actual knowledge structures rather than institutional divisions. The application’s outputs were validated against official course documentation from KTH Royal Institute of Technology, confirming the accuracy of both structural classifications and content relationship identification. This capacity for discovering and visualizing implicit relationships empowers faculty to design more coherent, efficient, and pedagogically sound educational experiences that optimize both teaching resources and student learning outcomes.

6.3 Limitations and future work

6.3.1 Current limitations

The AcademicRAG framework, despite its demonstrated effectiveness, exhibits several limitations that warrant acknowledgment. Foremost among these is the framework’s significant dependency on the capabilities of the underlying large language model. The quality of entity extraction, relationship identification, and response generation is inherently constrained by the LLM’s reasoning capabilities and domain knowledge. While Qwen2.5-72B and DeepSeek-V3 performed admirably in our experiments, variations in LLM performance directly impact system reliability. This dependency creates a fundamental interpretability challenge. When the system provides unexpected or suboptimal responses, the error may originate from the knowledge graph structure, the retrieval mechanism, or the generative model. As these components interact in complex ways, identifying the exact source of the error becomes increasingly difficult. With growing system complexity, this challenge becomes even more pronounced.

Entity recognition instability represents another significant limitation, particu-

larly evident in our knowledge graph evaluations. As demonstrated in Section 4.4, the LLM-based extraction process occasionally produced inconsistent entity representations for identical concepts (e.g., "SF1625" versus "SF1625 CALCULUS IN ONE VARIABLE"), impeding effective relationship mapping and reducing graph connectivity. This instability manifests more prominently in academic domains with specialized notation and complex terminological structures. Mathematical expressions proved especially challenging, with identical symbols (e.g., " N " or " A ") sometimes extracted as distinct entities despite representing the same concept across documents. These inconsistencies cascade through the knowledge graph construction process, creating structural fragmentation that impairs retrieval effectiveness.

Relationship ambiguity further compromises the system's semantic precision. The current implementation employs undirected relationships described in natural language rather than standardized, typed edges with formal semantics. While this approach offers flexibility in capturing diverse academic associations, it introduces considerable interpretive ambiguity. For instance, the relationship between two courses might be variously described as "builds upon," "requires knowledge from," or "prerequisite for"—all semantically related but lacking standardized representation. This ambiguity complicates graph traversal during retrieval, as the system cannot reliably distinguish between different relationship types (hierarchical, sequential, or associative) using formal reasoning rules. Consequently, the retrieval process must rely on text similarity rather than logical inference, potentially missing important structural patterns in the academic knowledge landscape.

6.3.2 Potential improvements and extensions

Several promising avenues exist for enhancing the entity-relationship extraction process that forms the foundation of the AcademicRAG framework. One particularly promising approach involves implementing a hybrid extraction system that combines LLM-based generation with rule-based validation. This approach would leverage domain-specific ontologies to standardize extracted entities against established academic taxonomies, thereby reducing naming inconsistencies and improving graph connectivity. For instance, course codes could be validated against institutional catalogs, while academic concepts could be mapped to disciplinary thesauri. This validation layer would operate as a post-processing filter on LLM-generated entities, preserving the flexibility of natural language extraction while imposing structural consistency on the resulting knowledge graph.

Relationship standardization represents another critical improvement opportunity. Rather than relying solely on free-text relationship descriptions, the system could implement a hierarchical relationship schema that categorizes academic associations into standardized types (prerequisite, conceptual hierarchy, methodological similarity, etc.) with well-defined semantics. This standardization would enable more sophisticated reasoning during retrieval—for example, automatically inferring transitive prerequisites or identifying conceptual subsumption relationships. The implementation could retain natural language descriptions for human interpretability while encoding formal relationship types as metadata, enabling both semantic richness and logical reasoning

within the same framework.

The extension of AcademicRAG to additional domains and document types presents compelling opportunities for future research. Legal education materials, with their complex case citation networks and hierarchical doctrine structures, would benefit significantly from the framework’s ability to identify implicit connections. Similarly, medical education documents, characterized by complex conceptual prerequisites and clinical pathway progressions, represent a promising application domain. The framework could also be adapted to handle more specialized academic document types beyond those examined in this study—including laboratory manuals, assessment instruments, and curriculum standards documents. Each extension would require domain-specific prompt engineering and entity schema adaptations, but the core architecture could remain largely intact, demonstrating the framework’s inherent versatility as a foundation for specialized academic knowledge systems.

6.3.3 Integration of multimodal content

The integration of multimodal content—including images, mathematical formulas, and technical diagrams—represents perhaps the most significant frontier for enhancing the AcademicRAG framework. Academic knowledge is inherently multimodal, with visual representations often communicating concepts that are difficult to express in text alone. In acoustics education, for example, waveform visualizations, room impulse response graphs, and circuit diagrams carry critical information that complements textual descriptions. The current implementation, however, processes only textual content, discarding this rich visual information during knowledge extraction. Expanding the framework to incorporate multimodal understanding would substantially enhance its capabilities in domains where visual representations are pedagogically essential.

Mathematical formula integration presents particularly compelling opportunities for academic knowledge systems. The current implementation treats mathematical expressions as plain text, losing their structural semantics and computational properties. By incorporating specialized formula extraction and representation techniques, the framework could preserve both the symbolic structure and semantic meaning of mathematical content. This would enable more sophisticated reasoning about mathematical relationships in academic contexts—for instance, recognizing when different notational forms represent equivalent concepts or identifying conceptual relationships expressed through mathematical formalism. Such capabilities would be especially valuable in quantitative disciplines like physics, engineering, and economics, where mathematical expressions often constitute the primary language of conceptual communication.

A comprehensive multimodal extension would require significant architectural enhancements. First, the document processing pipeline would need modification to extract and index visual content alongside text. This might involve specialized image segmentation techniques for diagrams and figures, and computer vision models to classify visual content types. Second, the entity extraction process would need to incorporate multimodal entity types that span textual and visual representations—for example, linking a textual description of a circuit to its corresponding diagram. Finally,

the retrieval and response generation components would require adaptation to integrate multimodal content into coherent responses, potentially incorporating visualization techniques to generate explanatory diagrams dynamically. While ambitious, such extensions would significantly advance the framework's capacity to represent and reason about the full spectrum of academic knowledge.

7 Conclusion

This thesis has presented AcademicRAG, a novel knowledge graph-enhanced framework for academic resource discovery that addresses critical limitations in traditional information retrieval systems. By integrating graph-based retrieval mechanisms with LLMs, the framework demonstrates a significant advancement in capturing, representing, and leveraging the complex semantic relationships inherent in academic knowledge.

The AcademicRAG framework introduces two key technical innovations that enhance retrieval performance: clue-guided keyword generation and subgraph-based retrieval. Our comprehensive evaluation across diverse academic domains demonstrates that these mechanisms consistently outperform existing state-of-the-art approaches, with particularly notable improvements in response comprehensiveness and user empowerment. The dual-level keyword extraction process effectively balances local detail with global context, while the subgraph retrieval mechanism captures multifaceted relationships that remain invisible to conventional approaches.

The AcademicRAG data pipeline employs a three-tier architecture that efficiently transforms diverse academic content into structured knowledge. Its three-stage flow components work together to process documents, create semantic chunks, and extract key relationships. The pipeline features fault-tolerant processing through state management and reduces computational costs via LLM caching. Its most significant advantage is supporting incremental knowledge integration without rebuilding the entire database—a major improvement over GraphRAG [19] requiring complete regeneration of knowledge structures.

The implementation of the Course Discovery System demonstrates the framework’s practical utility in educational contexts. By transforming unstructured syllabus content into semantically rich knowledge representations, the system addresses distinct information needs for both students and faculty. For students, it provides personalized learning pathways and prerequisite analyses that connect academic requirements to skill development objectives. For faculty, it offers curriculum optimization insights by identifying content overlaps and progression patterns across departmental and school boundaries.

Despite these achievements, we acknowledge several limitations in the current implementation. The system’s reliance on large language models introduces interpretability challenges and entity extraction inconsistencies. Future work should focus on hybrid extraction approaches that combine LLM generation with rule-based validation, standardized relationship schemas that enable logical inference, and multimodal extensions that incorporate visual content.

The AcademicRAG framework represents a significant step toward more intuitive, context-aware academic information systems. By preserving and leveraging the interconnected nature of scholarly knowledge, it enables richer exploration of educational resources and more informed academic decision-making. As educational institutions increasingly seek AI-enhanced tools to improve knowledge accessibility, AcademicRAG offers promising foundations for the next generation of academic discovery systems.

References

- [1] A. Hogan, E. Blomqvist, M. Cochez, *et al.*, “Knowledge graphs”, *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–37, May 31, 2022, issn: 0360-0300, 1557-7341. doi: [10.1145/3447772](https://doi.org/10.1145/3447772). arXiv: [2003.02320\[cs\]](https://arxiv.org/abs/2003.02320). [Online]. Available: <http://arxiv.org/abs/2003.02320> (visited on 03/04/2025).
- [2] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, “A survey on knowledge graphs: Representation, acquisition and applications”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 494–514, Feb. 2022, issn: 2162-237X, 2162-2388. doi: [10.1109/TNNLS.2021.3070843](https://doi.org/10.1109/TNNLS.2021.3070843). arXiv: [2002.00388\[cs\]](https://arxiv.org/abs/2002.00388). [Online]. Available: <http://arxiv.org/abs/2002.00388> (visited on 03/03/2025).
- [3] T. B. Brown, B. Mann, N. Ryder, *et al.*, *Language models are few-shot learners*, Jul. 22, 2020. doi: [10.48550/arXiv.2005.14165](https://doi.org/10.48550/arXiv.2005.14165). arXiv: [2005.14165\[cs\]](https://arxiv.org/abs/2005.14165). [Online]. Available: <http://arxiv.org/abs/2005.14165> (visited on 03/03/2025).
- [4] P. Lewis, E. Perez, A. Piktus, *et al.*, *Retrieval-augmented generation for knowledge-intensive NLP tasks*, Apr. 12, 2021. doi: [10.48550/arXiv.2005.11401](https://doi.org/10.48550/arXiv.2005.11401). arXiv: [2005.11401\[cs\]](https://arxiv.org/abs/2005.11401). [Online]. Available: <http://arxiv.org/abs/2005.11401> (visited on 03/03/2025).
- [5] Y. Gao, Y. Xiong, X. Gao, *et al.*, *Retrieval-augmented generation for large language models: A survey*, Mar. 27, 2024. doi: [10.48550/arXiv.2312.10997](https://doi.org/10.48550/arXiv.2312.10997). arXiv: [2312.10997\[cs\]](https://arxiv.org/abs/2312.10997). [Online]. Available: <http://arxiv.org/abs/2312.10997> (visited on 03/04/2025).
- [6] P. Chen, Y. Lu, V. W. Zheng, X. Chen, and B. Yang, “KnowEdu: A system to construct knowledge graph for education”, *IEEE Access*, vol. 6, pp. 31 553–31 563, 2018, Conference Name: IEEE Access, issn: 2169-3536. doi: [10.1109/ACCESS.2018.2839607](https://doi.org/10.1109/ACCESS.2018.2839607). [Online]. Available: <https://ieeexplore.ieee.org/document/8362657> (visited on 03/03/2025).
- [7] D. Vrandečić and M. Krötzsch, “Wikidata: A free collaborative knowledgebase”, *Commun. ACM*, vol. 57, no. 10, pp. 78–85, 2014, issn: 0001-0782. doi: [10.1145/2629489](https://doi.org/10.1145/2629489). [Online]. Available: <https://dl.acm.org/doi/10.1145/2629489> (visited on 03/04/2025).
- [8] B. Peng, Y. Zhu, Y. Liu, *et al.*, *Graph retrieval-augmented generation: A survey*, Sep. 10, 2024. doi: [10.48550/arXiv.2408.08921](https://doi.org/10.48550/arXiv.2408.08921). arXiv: [2408.08921\[cs\]](https://arxiv.org/abs/2408.08921). [Online]. Available: <http://arxiv.org/abs/2408.08921> (visited on 01/23/2025).
- [9] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, Aug. 2, 2023. doi: [10.48550/arXiv.1706.03762](https://doi.org/10.48550/arXiv.1706.03762). arXiv: [1706.03762\[cs\]](https://arxiv.org/abs/1706.03762). [Online]. Available: <http://arxiv.org/abs/1706.03762> (visited on 03/03/2025).

- [10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, May 24, 2019. doi: [10.48550/arXiv.1810.04805](https://doi.org/10.48550/arXiv.1810.04805). arXiv: [1810.04805\[cs\]](https://arxiv.org/abs/1810.04805). [Online]. Available: <http://arxiv.org/abs/1810.04805> (visited on 03/03/2025).
- [11] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, “Improving language understanding by generative pre-training”, 2018.
- [12] J. Wei, X. Wang, D. Schuurmans, *et al.*, *Chain-of-thought prompting elicits reasoning in large language models*, Jan. 10, 2023. doi: [10.48550/arXiv.2201.11903](https://doi.org/10.48550/arXiv.2201.11903). arXiv: [2201.11903\[cs\]](https://arxiv.org/abs/2201.11903). [Online]. Available: <http://arxiv.org/abs/2201.11903> (visited on 03/04/2025).
- [13] OpenAI, J. Achiam, S. Adler, *et al.*, *GPT-4 technical report*, Mar. 4, 2024. doi: [10.48550/arXiv.2303.08774](https://doi.org/10.48550/arXiv.2303.08774). arXiv: [2303.08774\[cs\]](https://arxiv.org/abs/2303.08774). [Online]. Available: <http://arxiv.org/abs/2303.08774> (visited on 03/03/2025).
- [14] DeepSeek-AI, D. Guo, D. Yang, *et al.*, *DeepSeek-r1: Incentivizing reasoning capability in LLMs via reinforcement learning*, Jan. 22, 2025. doi: [10.48550/arXiv.2501.12948](https://doi.org/10.48550/arXiv.2501.12948). arXiv: [2501.12948\[cs\]](https://arxiv.org/abs/2501.12948). [Online]. Available: <http://arxiv.org/abs/2501.12948> (visited on 03/04/2025).
- [15] S. Bubeck, V. Chandrasekaran, R. Eldan, *et al.*, *Sparks of artificial general intelligence: Early experiments with GPT-4*, Apr. 13, 2023. doi: [10.48550/arXiv.2303.12712](https://doi.org/10.48550/arXiv.2303.12712). arXiv: [2303.12712\[cs\]](https://arxiv.org/abs/2303.12712). [Online]. Available: <http://arxiv.org/abs/2303.12712> (visited on 03/03/2025).
- [16] L. Ouyang, J. Wu, X. Jiang, *et al.*, *Training language models to follow instructions with human feedback*, Mar. 4, 2022. doi: [10.48550/arXiv.2203.02155](https://doi.org/10.48550/arXiv.2203.02155). arXiv: [2203.02155\[cs\]](https://arxiv.org/abs/2203.02155). [Online]. Available: <http://arxiv.org/abs/2203.02155> (visited on 03/03/2025).
- [17] Z. Shao, P. Wang, Q. Zhu, *et al.*, *DeepSeekMath: Pushing the limits of mathematical reasoning in open language models*, Apr. 27, 2024. doi: [10.48550/arXiv.2402.03300](https://doi.org/10.48550/arXiv.2402.03300). arXiv: [2402.03300\[cs\]](https://arxiv.org/abs/2402.03300). [Online]. Available: <http://arxiv.org/abs/2402.03300> (visited on 03/03/2025).
- [18] H. Touvron, T. Lavril, G. Izacard, *et al.*, *LLaMA: Open and efficient foundation language models*, Feb. 27, 2023. doi: [10.48550/arXiv.2302.13971](https://doi.org/10.48550/arXiv.2302.13971). arXiv: [2302.13971\[cs\]](https://arxiv.org/abs/2302.13971). [Online]. Available: <http://arxiv.org/abs/2302.13971> (visited on 03/03/2025).
- [19] D. Edge, H. Trinh, N. Cheng, *et al.*, *From local to global: A graph RAG approach to query-focused summarization*, Apr. 24, 2024. doi: [10.48550/arXiv.2404.16130](https://doi.org/10.48550/arXiv.2404.16130). arXiv: [2404.16130\[cs\]](https://arxiv.org/abs/2404.16130). [Online]. Available: <http://arxiv.org/abs/2404.16130> (visited on 01/23/2025).
- [20] M. Yasunaga, A. Bosselut, H. Ren, *et al.*, *Deep bidirectional language-knowledge graph pretraining*, Oct. 19, 2022. doi: [10.48550/arXiv.2210.09338](https://doi.org/10.48550/arXiv.2210.09338). arXiv: [2210.09338\[cs\]](https://arxiv.org/abs/2210.09338). [Online]. Available: <http://arxiv.org/abs/2210.09338> (visited on 03/04/2025).

- [21] A. Saha, V. Pahuja, M. M. Khapra, K. Sankaranarayanan, and S. Chandar, *Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph*, 2018. arXiv: [1801.10314](https://arxiv.org/abs/1801.10314) [cs.CL]. [Online]. Available: <https://arxiv.org/abs/1801.10314>.
- [22] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, “Can a suit of armor conduct electricity? a new dataset for open book question answering”, in *EMNLP*, 2018.
- [23] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, *TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension*, May 13, 2017. doi: [10.48550/arXiv.1705.03551](https://doi.org/10.48550/arXiv.1705.03551). arXiv: [1705.03551](https://arxiv.org/abs/1705.03551)[cs]. [Online]. Available: <http://arxiv.org/abs/1705.03551> (visited on 03/04/2025).
- [24] J. Wu, J. Zhu, Y. Qi, et al., *Medical graph RAG: Towards safe medical large language model via graph retrieval-augmented generation*, Oct. 15, 2024. doi: [10.48550/arXiv.2408.04187](https://doi.org/10.48550/arXiv.2408.04187). arXiv: [2408.04187](https://arxiv.org/abs/2408.04187)[cs]. [Online]. Available: <http://arxiv.org/abs/2408.04187> (visited on 02/12/2025).
- [25] N. F. Liu, K. Lin, J. Hewitt, et al., “Lost in the middle: How language models use long contexts”, *Transactions of the Association for Computational Linguistics*, vol. 12, pp. 157–173, 2024. doi: [10.1162/tacl_a_00638](https://doi.org/10.1162/tacl_a_00638). [Online]. Available: <https://aclanthology.org/2024.tacl-1.9/>.
- [26] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, *LightRAG: Simple and fast retrieval-augmented generation*, Nov. 7, 2024. doi: [10.48550/arXiv.2410.05779](https://doi.org/10.48550/arXiv.2410.05779). arXiv: [2410.05779](https://arxiv.org/abs/2410.05779)[cs]. [Online]. Available: <http://arxiv.org/abs/2410.05779> (visited on 02/13/2025).
- [27] Y. Hu, Z. Lei, Z. Zhang, B. Pan, C. Ling, and L. Zhao, *GRAG: Graph retrieval-augmented generation*, Oct. 21, 2024. doi: [10.48550/arXiv.2405.16506](https://doi.org/10.48550/arXiv.2405.16506). arXiv: [2405.16506](https://arxiv.org/abs/2405.16506)[cs]. [Online]. Available: <http://arxiv.org/abs/2405.16506> (visited on 02/13/2025).
- [28] Y. Hu, Z. Lei, Z. Dai, et al., *CG-RAG: Research question answering by citation graph retrieval-augmented LLMs*, Jan. 25, 2025. doi: [10.48550/arXiv.2501.15067](https://doi.org/10.48550/arXiv.2501.15067). arXiv: [2501.15067](https://arxiv.org/abs/2501.15067)[cs]. [Online]. Available: <http://arxiv.org/abs/2501.15067> (visited on 02/12/2025).
- [29] R. Jia, B. Zhang, S. J. R. Méndez, and P. G. Omran, *Leveraging large language models for semantic query processing in a scholarly knowledge graph*, May 24, 2024. doi: [10.48550/arXiv.2405.15374](https://doi.org/10.48550/arXiv.2405.15374). arXiv: [2405.15374](https://arxiv.org/abs/2405.15374)[cs]. [Online]. Available: <http://arxiv.org/abs/2405.15374> (visited on 02/13/2025).
- [30] Y. Cai, Z. Guo, Y. Pei, W. Bian, and W. Zheng, *Simrag: Leveraging similar subgraphs for knowledge graphs driven retrieval-augmented generation*, 2024. arXiv: [2412.15272](https://arxiv.org/abs/2412.15272) [cs.CL]. [Online]. Available: <https://arxiv.org/abs/2412.15272>.

- [31] X. He, Y. Tian, Y. Sun, *et al.*, *G-retriever: Retrieval-augmented generation for textual graph understanding and question answering*, May 27, 2024. doi: [10.48550/arXiv.2402.07630](https://doi.org/10.48550/arXiv.2402.07630). arXiv: [2402.07630\[cs\]](https://arxiv.org/abs/2402.07630). [Online]. Available: <http://arxiv.org/abs/2402.07630> (visited on 02/19/2025).
- [32] M. Huang, C. Bu, Y. He, and X. Wu, *How to mitigate information loss in knowledge graphs for GraphRAG: Leveraging triple context restoration and query-driven feedback*, Jan. 26, 2025. doi: [10.48550/arXiv.2501.15378](https://doi.org/10.48550/arXiv.2501.15378). arXiv: [2501.15378\[cs\]](https://arxiv.org/abs/2501.15378). [Online]. Available: <http://arxiv.org/abs/2501.15378> (visited on 02/12/2025).
- [33] B. Sarmah, B. Hall, R. Rao, S. Patel, S. Pasquali, and D. Mehta, *HybridRAG: Integrating knowledge graphs and vector retrieval augmented generation for efficient information extraction*, Aug. 9, 2024. doi: [10.48550/arXiv.2408.04948](https://doi.org/10.48550/arXiv.2408.04948). arXiv: [2408.04948\[cs\]](https://arxiv.org/abs/2408.04948). [Online]. Available: <http://arxiv.org/abs/2408.04948> (visited on 02/19/2025).
- [34] H. Qian, P. Zhang, Z. Liu, K. Mao, and Z. Dou, *Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery*, 2024. arXiv: [2409.05591\[cs.CL\]](https://arxiv.org/abs/2409.05591). [Online]. Available: <https://arxiv.org/abs/2409.05591>.
- [35] DeepSeek-AI, *Deepseek-v3 technical report*, 2024. arXiv: [2412.19437\[cs.CL\]](https://arxiv.org/abs/2412.19437). [Online]. Available: <https://arxiv.org/abs/2412.19437>.
- [36] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, 2023. arXiv: [1706.03762\[cs.CL\]](https://arxiv.org/abs/1706.03762). [Online]. Available: <https://arxiv.org/abs/1706.03762>.
- [37] Q. Zhang, H. Lu, H. Sak, *et al.*, *Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss*, 2020. arXiv: [2002.02562\[eess.AS\]](https://arxiv.org/abs/2002.02562). [Online]. Available: <https://arxiv.org/abs/2002.02562>.
- [38] M. Li, S. Zhang, C. Zorila, and R. Doddipatla, *Transformer-based streaming asr with cumulative attention*, 2022. arXiv: [2203.05736\[eess.AS\]](https://arxiv.org/abs/2203.05736). [Online]. Available: <https://arxiv.org/abs/2203.05736>.
- [39] KTH Royal Institute of Technology, *Dt1175 sound 7.5 credits*, Accessed: 28 April 2025, 2025. [Online]. Available: <https://www.kth.se/student/kurser/kurs/DT1175?l=en>.

A Github Repository

The GitHub repository for our AcademicRAG framework: <https://github.com/shua-chen/academicRAG>

B LLM Judge Evaluation Protocol

The following prompt was used to configure the large language model (DeepSeek-R1) for conducting pairwise evaluations of responses generated by different RAG frameworks. The evaluation was structured to assess three key dimensions—**Comprehensiveness**, **Diversity**, and **Empowerment**—along with an overall judgment.

B.1 System instruction

—Role—

You are an expert tasked with evaluating two answers to the same question based on three criteria: Comprehensiveness, Diversity, and Empowerment.

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

For each criterion, choose the better answer (either Answer Petter or Answer Bob) and explain why. Then, select an overall winner based on these three categories.

B.2 Evaluation template

Here is the question:

{query}

Here are the two answers:

Answer Petter:

{answer1}

Answer Bob:

{answer2}

Evaluate both answers using the three criteria listed above and provide detailed reasoning for each criterion.

Output your evaluation in the following JSON format:

```
{
  "Comprehensiveness": {
    "Reasoning": "[Provide reasoning here]",
    "Winner": "[Answer Petter or Answer Bob]"
  },
  "Diversity": {
    "Reasoning": "[Provide reasoning here]",
    "Winner": "[Answer Petter or Answer Bob]"
  },
}
```

```
"Empowerment": {  
    "Reasoning": "[Provide reasoning here]",  
    "Winner": "[Answer Petter or Answer Bob]"  
},  
"Overall Winner": {  
    "Reasoning": "[Provide reasoning here]",  
    "Winner": "[Answer Petter or Answer Bob]"  
}  
}
```

B.3 Implementation details

For each query-response pair evaluation, the framework names (“Petter” and “Bob”) served as placeholders for the actual RAG systems being compared (AcademicRAG, GraphRAG, LightRAG, or NaiveRAG). To mitigate potential positional bias, we conducted three evaluations per pair, alternating the order of presentation, and averaged the results to determine final win rates. The structured JSON output facilitated automated aggregation of win/loss statistics across the evaluation corpus of 125 questions per domain.

C Academic Framework Ablation Studies

To validate the effectiveness of our proposed methods, we conduct ablation studies to assess the impact of subgraph retrieval and clue-guided keyword generation. The results, presented in Table C1, illustrate the contributions of these components to the overall performance of the AcademicRAG framework. We analyze the effects of subgraph and clues by omitting one of them during the retrieval stage, then comparing the ablated models against GraphRAG on the evaluation dataset. Our observations for the different variants are listed as follows:

- **Impact of Clues:** The -clues model eliminates the clue-guided keyword generation process during keyword extraction. Instead, it relies solely on LLM-generated keywords, derived directly from the user’s query, to retrieve relevant nodes and edges from the KG. Without the guidance of contextual clues, the LLM generates keywords based solely on its internal knowledge, which often leads to hallucinations—producing keywords that are irrelevant to the actual KG. This misalignment negatively impacts the retrieval stage, reducing the amount of retrieved information and ultimately diminishing the diversity of the generated answers.
- **Impact of Subgraph:** The -subgraph model modifies the retrieval stage by fetching one-hop neighbors instead of retrieving entire subgraph. While this approach introduces a greater number of neighboring nodes, thereby allowing the model to generate more diverse responses, it also comes with significant drawbacks. Many of the retrieved neighbor nodes are irrelevant to the query, adding unnecessary noise to the retrieval process. Additionally, some crucial interconnected nodes that could provide deeper contextual insights are overlooked, resulting in responses that are less comprehensive.

Table C1: Ablation Studies on Win Rates (%) Compared with GraphRAG

Metrics	Framework	Win Rate	
		Agriculture	CS
Comprehensiveness	vs. AcademicRAG	52.4%↑	51.2%↑
	vs. -clues	53.6%↑	51.0%↑
	vs. -subgraph	50.8%↑	49.2%↓
Diversity	vs. AcademicRAG	50.8%↑	46.8%↓
	vs. -clues	48.4%↓	43.0%↓
	vs. -subgraph	51.6%↑	48.0%↓
Empowerment	vs. AcademicRAG	54.0%↑	55.2%↑
	vs. -clues	60.8%↑	54.6%↑
	vs. -subgraph	52.8%↑	59.3%↑
Overall	vs. AcademicRAG	52.4%↑	53.6%↑
	vs. -clues	57.2%↑	51.8%↑
	vs. -subgraph	53.6%↑	55.2%↑

Note: ↑ indicates the noted framework outperforming the GraphRAG, ↓ indicates underperforming. Values are win rates.

The results of our ablation studies highlight the critical role of both subgraph retrieval and clue-guided keyword generation in the AcademicRAG framework. The removal of either

component leads to noticeable declines in some dimensions, demonstrating their effectiveness and necessity. Subgraph retrieval ensures that the model captures potential relationships and retrieves comprehensive contextual information, while clue-guided keyword generation prevents hallucinations by aligning keyword selection with actual KG content. These findings reinforce the importance of these mechanisms in enhancing retrieval precision, maintaining contextual relevance, and improving overall response diversity within the AcademicRAG framework.

D Academic Framework Case Study

Beyond the advantages reflected in statistical data, to provide a more intuitive analysis of AcademicRAG’s answer generation capability on academic texts, this section will present two cases, each offering a direct comparison of our framework with LightRAG and GraphRAG in actual evaluations.

The case study in Table D1 demonstrates AcademicRAG’s ability to generate structured, actionable responses through its subgraph-aware retrieval mechanism. For the query regarding hybrid models in ride-sharing segmentation, GraphRAG produces a conventional answer focusing on basic model combinations (regression + clustering) and dataset applications. While technically correct, its response exhibits three critical limitations: (1) Linear presentation of methods without hierarchical organization, (2) Over-reliance on textbook case studies rather than operational frameworks, and (3) Limited connection between technical solutions and practical implementation.

Table D2 reveals similar limitations in LightRAG’s approach, despite its graph-based foundation. LightRAG organizes its response into five discrete components—regression, clustering, combination, evaluation, and data transformation—but fails to establish meaningful connections between these elements. For instance, while it mentions performance evaluation metrics (cross-validation, WCSS), it does not specify how these metrics should be applied iteratively to optimize hybrid models. The "Application in Ride-Sharing" section merely lists generic objectives ("segment users", "predict behavior") without detailing implementation workflows. This fragmented structure stems from LightRAG’s reliance on keyword-based one-hop neighbor retrieval, which captures isolated concepts ("regression", "clustering") but misses higher-order relationships between methodological components and operational requirements. Consequently, LightRAG’s answer resembles a textbook summary rather than an executable solution framework.

AcademicRAG overcomes these limitations through its unique retrieval architecture. The clue-guided keyword generation first identifies core concepts ("hybrid models", "user segmentation", "dynamic refinement") and latent requirements ("implementation steps", "scalability considerations"). This triggers a targeted subgraph retrieval that connects regression techniques (e.g., decision trees) with advanced clustering methods (hierarchical clustering) and system design principles. The retrieved subgraph reveals four distinct hybrid approaches through entity relationships that both GraphRAG and LightRAG missed, including the hierarchical clustering-regression combination and ensemble methods.

The LLM evaluation highlights AcademicRAG’s superiority across all dimensions. In comprehensiveness, the framework’s ability to retrieve interconnected concepts from the subgraph allows coverage of 4 hybrid approaches versus GraphRAG’s 2 and LightRAG’s 3. The diversity advantage stems from synthesizing cross-domain knowledge – while baseline models focus on machine learning components, AcademicRAG incorporates software design patterns from the computer science corpus through subgraph relationships. Empowerment is achieved through operational guidance, where AcademicRAG converts retrieved knowledge into implementable workflows (e.g., explicit evaluation phase integration), directly addressing the implementation gaps observed in both GraphRAG and LightRAG responses.

Table D2 extends our comparative analysis by examining the performance differential between LightRAG and AcademicRAG in response to the identical query on hybrid models for user segmentation in ride-sharing. The juxtaposition of these frameworks reveals notable distinctions in their respective approaches and output quality. While both systems present structured responses with enumerated methodologies, AcademicRAG demonstrates superior

Table D1: Case Study: Comparison Between GraphRAG and AcademicRAG

Query	What hybrid models (e.g., regression + clustering) enhance user segmentation in ride-sharing?
GraphRAG	<p>Hybrid Models for User Segmentation in Ride-Sharing User segmentation in ride-sharing platforms is critical for enhancing user experience and optimizing business strategies. Hybrid models combining regression and clustering techniques can significantly improve segmentation accuracy.</p> <p>Regression Models: Used to predict metrics like ride frequency, duration, and distance. For example, the **Linear Model** and **Decision Tree Model** were applied to the **Bike Sharing Dataset**, with the Decision Tree achieving better performance (RMSLE: 0.6259 vs. 1.4974).</p> <p>Clustering Models: **K-Means Clustering** groups users based on behavior and characteristics. User vectors encapsulate attributes for clustering.</p> <p>Hybrid Approach: Combines initial clustering with regression for predictive insights and refinement. For example, K-Means segments users, and regression predicts metrics for each segment.</p> <p>Case Study: The **Bike Sharing Dataset** demonstrates the effectiveness of hybrid models, with decision trees outperforming linear models.</p> <p>Conclusion: Hybrid models provide deep insights into user behavior, enabling personalized services and improved business performance.</p>
AcademicRAG	<p>Hybrid Models for Enhancing User Segmentation in Ride-Sharing Hybrid models combining regression and clustering offer comprehensive user segmentation for ride-sharing platforms. Four approaches are highlighted:</p> <ol style="list-style-type: none"> Regression + Clustering: Predict user behaviors (e.g., ride frequency) and cluster based on predictions. Benefits include personalization and operational efficiency. Clustering + Regression: Segment users initially with clustering, then refine with regression for dynamic insights. Hierarchical Clustering + Regression: Create multi-level segmentation for granular insights and scalability. Ensemble Methods: Combine multiple models for robust and accurate predictions. <p>Practical Example: A ride-sharing company uses hybrid models to segment users based on ride frequency and preferred times. Steps include data collection, clustering, regression, and implementation.</p> <p>Conclusion: Hybrid models enable deeper insights into user behavior, leading to personalized services and improved operational efficiency.</p>
LLM Decision	<p>Comprehensiveness: "Winner": "Answer AcademicRAG" Reasoning: AcademicRAG explores multiple hybrid approaches and includes a practical example, while GraphRAG focuses on one approach.</p> <p>Diversity: "Winner": "Answer AcademicRAG" Reasoning: AcademicRAG presents diverse methodologies (e.g., hierarchical clustering, ensemble methods), while GraphRAG centers on a single approach.</p> <p>Empowerment: "Winner": "Answer AcademicRAG" Reasoning: AcademicRAG provides actionable steps (e.g., model evaluation, implementation), while GraphRAG lacks concrete guidance.</p> <p>Overall Winner: "Winner": "Answer AcademicRAG" Reasoning: AcademicRAG excels in comprehensiveness, diversity, and empowerment, making it the superior response.</p>

Table D2: Case Study: Comparison Between LightRAG and AcademicRAG

Query	What hybrid models (e.g., regression + clustering) enhance user segmentation in ride-sharing?
LightRAG	<p>Hybrid Models for Enhancing User Segmentation in Ride-Sharing User segmentation in ride-sharing can be significantly enhanced by combining regression and clustering models. Key approaches include:</p> <ul style="list-style-type: none"> 1. Regression Models for Predictive Analytics: Predict bike usage patterns using datasets like bike sharing, incorporating features like weather and seasonal information. [...] 2. Clustering Models for Segmentation: Apply K-means clustering to group users based on behavior, such as spending patterns or ride preferences. [...] 3. Combining Regression and Clustering: Use regression to derive features for clustering, then apply regression within clusters for deeper insights. [...] 4. Performance Evaluation: Use cross-validation and WCSS to evaluate models and ensure optimal performance. [...] 5. Data Transformation: Normalize and log-transform data to improve model accuracy. [...] Application in Ride-Sharing: Segment users, predict behavior, optimize services, and monitor performance. [...]
AcademicRAG	<p>Hybrid Models for Enhancing User Segmentation in Ride-Sharing Hybrid models combining regression and clustering offer comprehensive user segmentation for ride-sharing platforms. Four approaches are highlighted:</p> <ul style="list-style-type: none"> 1. **Regression + Clustering:** Predict user behaviors (e.g., ride frequency) and cluster based on predictions. Benefits include personalization and operational efficiency. [...] 2. **Clustering + Regression:** Segment users initially with clustering, then refine with regression for dynamic insights. [...] 3. **Hierarchical Clustering + Regression:** Create multi-level segmentation for granular insights and scalability. [...] 4. **Ensemble Methods:** Combine multiple models for robust and accurate predictions. [...] <p>Practical Example: A ride-sharing company uses hybrid models to segment users based on ride frequency and preferred times. Steps include data collection, clustering, regression, and implementation. [...]</p> <p>Conclusion: Hybrid models enable deeper insights into user behavior, leading to personalized services and improved operational efficiency. [...]</p>
LLM Decision	<p>Comprehensiveness: "Winner": "Answer AcademicRAG"</p> <p>Reasoning: AcademicRAG provides a structured, cohesive explanation with clear benefits and a practical example. [...]</p> <p>Diversity: "Winner": "Answer AcademicRAG"</p> <p>Reasoning: AcademicRAG offers diverse methodological approaches tailored to ride-sharing. [...]</p> <p>Empowerment: "Winner": "Answer AcademicRAG"</p> <p>Reasoning: AcademicRAG provides actionable insights and a clear implementation roadmap. [...]</p> <p>Overall Winner: "Answer AcademicRAG"</p> <p>Reasoning: AcademicRAG excels in comprehensiveness, diversity, and empowerment. [...]</p>

organizational coherence and conceptual depth. Specifically, LightRAG’s response, though informative with its five-point structure covering regression models, clustering techniques, combined approaches, evaluation metrics, and data transformation, lacks the integrated theoretical framework evident in AcademicRAG’s presentation. AcademicRAG’s response is distinguished by its systematic progression from foundational hybrid models to more sophisticated techniques like hierarchical clustering and ensemble methods, coupled with a concrete practical example that delineates implementation steps. The LLM Decision section further validates this qualitative assessment, highlighting AcademicRAG’s superiority across all evaluation dimensions—comprehensiveness, diversity, and empowerment—thereby reinforcing the framework’s enhanced capability to generate academically rigorous and practically applicable responses in specialized domains such as transportation analytics and user behavior modeling.

E Entity Types Design for Course Syllabi

Table E1: Course Syllabi Knowledge Graph Entity Type Definition

Entity Type	Definition and Function	Example
Course Name	Official identification of the course, including course code and title, as the core node of the Knowledge Graph	DT1175 Sound
Credits	Credit value of the course, reflecting the amount of learning and relative importance of the course	7.5 hp
Knowledge Concept	Core subject concepts and theories covered by the course, which form the basic units of knowledge of the course content	Acoustic communication, fluctuation theory, digital audio technology
Skill	Specific competencies and practical applications developed by the course, extracted from the Learning Outcomes section	Ability to describe the different levels of representation of acoustic communication, and ability to calculate sound intensity and decibel levels
Person	Personnel roles associated with the course such as instructor, teaching assistants, etc.	Kjetil Falkenberg
Assessment	Assessment and evaluation methods used in the course	Written exams, lab work, project reports
Learning Material	Textbooks, references, and learning resources recommended for the course	Sound as an information carrier (handouts)
Learning Tool	Technical tools such as software, lab equipment used in the course	Mixing consoles, audio editing software
Grading Scale	Grading criteria and grading system used in the course	A-F, pass/fail

F Detailed Evaluation Results for Application

Table F1: Learning Path Planning for Audio Engineering

Query	<p>Background: I have a solid foundation in mathematics... My coursework included subjects such as wave physics... I am proficient in MATLAB...I'm interested in audio engineering...</p> <p>Question: Which courses should I take, in what order, to build expertise in this area while meeting all prerequisites?</p>
Answer	<p>1. Foundation in Sound and Acoustics Course: DT1175 Sound Description: This course provides foundational knowledge in acoustics, sound technology, and audio engineering. It covers wave physics, human hearing, room acoustics, and basic signal processing. Why take this first? It builds the essential theoretical and practical groundwork for understanding sound and its applications. Prerequisites: Basic wave theory and signal processing (already in your background).</p> <p>2. Intermediate Acoustics and Signal Processing Course: EQ2321 Speech and Audio Processing Description: Focuses on advanced signal processing techniques for speech and audio signals, including coding, noise reduction, and psychoacoustics. Why take this next? It bridges foundational acoustics with practical audio engineering by introducing signal analysis, coding, and enhancement techniques.</p> <p>3. Advanced Acoustics and Engineering Course: SD2111 Engineering Acoustics Description: Covers the fundamentals of elastic wave propagation, human response to sound and vibration, and advanced measurement techniques.</p> <p>4. Specialization in Music Acoustics Course: DT2212 Music Acoustics Description: Focuses on the acoustics of musical instruments and sound synthesis.</p> <p>5. Advanced Spatial Audio and Room Acoustics Course: FDT3318 Room Acoustics and Spatial Audio Description: Focuses on advanced topics in room acoustics and spatial audio reproduction, including virtual environments and VR applications.</p> <p>Skills and Tools You Will Gain:</p> <ul style="list-style-type: none"> • Acoustic measurement and analysis • Programming tools like MATLAB for audio signal processing • Simulation software like COMSOL for room acoustics • Designing audio systems for rooms and virtual environments • Enhancing speech and audio signals in noisy environments • Analyzing and synthesizing musical instruments

Table F2: Prerequisite Identification Query

Query	<p>Background: I have a solid foundation in mathematics... My coursework included subjects such as wave physics... I am proficient in MATLAB...I'm interested in audio engineering...</p> <p>Question: What prerequisites courses and knowledge do I need before enrolling in DT2300 Sound in Interaction?</p>
Answer	<p>Mandatory Prerequisites: • Programming Knowledge: You must have completed at least 6 credits in programming, equivalent to courses like DD1337/DD1310-DD1318/DD1321/DD1331/DD100N/ID1018. Proficiency in programming is essential for designing sound-based interactive applications. • Human-Computer Interaction (HCI): You must have completed at least 6 credits in human-computer interaction, equivalent to courses like DH1620/DH1622. This foundational knowledge is crucial for understanding interaction design principles. • English Language Proficiency: You must have a basic level of English proficiency, equivalent to the Swedish upper secondary course English B/6. This is necessary to understand course materials and participate in discussions.</p> <p>Recommended Background: • Knowledge in Human Perception: Equivalent to course DM2350 (formerly DT2350). Understanding human perception will help you analyze soundscapes and design inclusive sound-based applications. • Multimodal Interaction: Equivalent to course DT2140. This knowledge will provide a broader context for sound in interactive systems.</p> <p>Course Content and Expectations The course focuses on:</p> <ul style="list-style-type: none"> • Sound Perception: Principles of how sound is experienced and perceived in interactive systems. • Interaction Design: Using sound as feedback, sonification of movements, and designing soundscapes. • Practical Design: Using free, state-of-the-art tools and sensors (e.g., Arduino, Raspberry Pi) to design new sound-based interactive applications.

Table F3: Course Recommendation for Laboratory-Based Acoustics

Query	<p>Background: I have a solid foundation in mathematics... My coursework included subjects such as wave physics... I am proficient in MATLAB...I'm interested in audio engineering...</p> <p>Question: Which courses cover both acoustic theory and signal processing with practical laboratory components that would complement my current knowledge of basic wave physics?</p>
Answer	<p>1. DT2212 Music Acoustics Course Overview: • Combines theoretical concepts such as wave equations, Fourier transforms, and acoustics of musical instruments with practical lab work. • Topics include signal analysis, physical modeling of musical instruments, and numerical synthesis methods. • Laboratories involve implementing instrument models in MATLAB and analyzing sound radiation.</p> <p>2. SD2111 Engineering Acoustics Course Overview: • Focuses on elastic wave phenomena in fluids and solids, signal analysis techniques, and human response to sound and vibration. • Includes laboratory exercises for measuring sound and vibration, and analyzing wave propagation.</p> <p>3. FDT3318 Room Acoustics and Spatial Audio Course Overview: • Focuses on room acoustics, spatial audio reproduction, and virtual acoustics. • Includes laboratory sessions for measuring reverberation times, simulating spatial audio, and using tools like COMSOL for room acoustics modeling.</p> <p>4. EQ2321 Speech and Audio Processing Course Overview: • Focuses on advanced signal processing methods for speech and audio, including psychoacoustic principles, coding, and enhancement techniques. • Includes programming exercises in MATLAB or Python for analyzing and processing audio signals.</p> <p>5. DT1175 Sound Course Overview: • Introduces the fundamentals of sound communication, including wave theory, room acoustics, and audio technology. • Laboratories involve measuring hearing thresholds, analyzing signal distortion, and designing experiments.</p>

Table F4: Course Classification by Term and Academic Level

	Autumn Term	Spring Term
Bachelor's Level	SD1116: Design of Silent and Vibration-free Products (6.0 credits) DT1175: Sound (7.5 credits) LD1016: Learning to learn with AI (2.0 credits)	
Master's Level	SD2111: Engineering Acoustics (6.0 credits) SD2180: Non-linear Acoustics (6.0 credits) SD2190/SD2191: Vehicle Acoustics and Vibration (6.0 credits) SD2102: Individual Project Work on Sound and Vibration (12.0 credits) SD2103: Individual Project Work on Sound and Vibration (16.0 credits) AF2404/AF244V: Building Acoustics (7.5 credits) SD2166: Building Acoustics and Community Noise (7.5 credits) SD2165: Acoustical Measurements (8.0 credits) FDT3318: Room Acoustics and Spatial Audio (7.5 credits) AL2511: Resilience Thinking in Sustainable Planning (7.5 credits)	SD211X: Degree Project in Technical Acoustics (30.0 credits) SD2101: Individual Project Work on Sound and Vibration (8.0 credits) FSD3110: Structure Borne Sound (7.5 credits) SD2155: Flow Acoustics (6.0 credits) SD2175: Numerical Methods for Acoustics and Vibration (9.0 credits) DT2212: Music Acoustics (7.5 credits) DT2300: Sound in Interaction (7.5 credits) EQ2321: Speech and Audio Processing (7.5 credits) CH2012: Evaluation and Measures of the Acoustic Work Environment and Vibrations (7.5 credits)

Table F5: Analysis of Common Intended Learning Outcomes Between Course Pairs (Part 1)

Query	How many intended learning outcomes are common between each pair of courses?
Answer	<p>1. EQ2321 Speech and Audio Processing vs. DT2212 Music Acoustics Common Learning Outcomes: • Extraction and presentation of scientific content from acoustics research. • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 2</p> <p>2. EQ2321 Speech and Audio Processing vs. SD2116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>3. EQ2321 Speech and Audio Processing vs. SD1116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>4. EQ2321 Speech and Audio Processing vs. DT2300 Sound in Interaction Common Learning Outcomes: • Application of principles of sound perception and interaction in designing sound-based systems. Total Common Outcomes: 1</p> <p>5. DT2212 Music Acoustics vs. SD2116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>6. DT2212 Music Acoustics vs. SD1116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>7. DT2212 Music Acoustics vs. DT2300 Sound in Interaction Common Learning Outcomes: • Application of principles of sound perception and interaction in designing sound-based systems. Total Common Outcomes: 1</p>

Table F6: Analysis of Common Intended Learning Outcomes Between Course Pairs
(Part 2)

Query	How many intended learning outcomes are common between each pair of courses? (continued)
Answer	<p>8. SD2116 Design of Silent and Vibration-Free Products vs. SD2165 Acoustical Measurements Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>9. SD2116 Design of Silent and Vibration-Free Products vs. SD1116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>10. SD2116 Design of Silent and Vibration-Free Products vs. DT2300 Sound in Interaction Common Learning Outcomes: • Application of principles of sound perception and interaction in designing sound-based systems. Total Common Outcomes: 1</p> <p>11. SD2165 Acoustical Measurements vs. SD1116 Design of Silent and Vibration-Free Products Common Learning Outcomes: • Application of mathematical models and methods to describe sound and vibration quantities. Total Common Outcomes: 1</p> <p>12. SD2165 Acoustical Measurements vs. DT2300 Sound in Interaction Common Learning Outcomes: • Application of principles of sound perception and interaction in designing sound-based systems. Total Common Outcomes: 1</p> <p>13. SD1116 Design of Silent and Vibration-Free Products vs. DT2300 Sound in Interaction Common Learning Outcomes: • Application of principles of sound perception and interaction in designing sound-based systems. Total Common Outcomes: 1</p> <p>14. Summary of Most Common Learning Outcomes: • "Application of mathematical models and methods to describe sound and vibration quantities" appears in 8 course pairs. • "Application of principles of sound perception and interaction in designing sound-based systems" appears in 5 course pairs.</p>

Table F7: Course Integration Recommendations

Query	Based on the answers to questions about education level, teaching term, and common learning outcomes, can you suggest viable and feasible ways to integrate/merge courses with high overlapping relevance?
Answer	<p>Viable and Feasible Course Integration Recommendations</p> <p>Based on analysis of academic level, teaching term, and common intended learning outcomes, 5 potential course integrations are recommended:</p> <ol style="list-style-type: none"> 1. DT2212 (Music Acoustics) and DT1175 (Sound) • <i>Rationale</i>: Share common learning outcome related to describing sound behavior in spaces • <i>Integration Method</i>: Create a multi-term sequence where DT1175 serves as foundation, and DT2212 builds advanced skills 2. SD2175 (Numerical Methods) and SD1116 (Design of Silent Products) • <i>Rationale</i>: Share learning outcome related to mathematical models for sound/vibration • <i>Integration Method</i>: Combine into a single advanced course integrating numerical methods with product design 3. SD2111 (Engineering Acoustics) and SD2180 (Non-linear Acoustics) • <i>Rationale</i>: Overlapping content related to sound propagation • <i>Integration Method</i>: Create two-module course with linear acoustics first, then non-linear phenomena 4. DT2300 (Sound in Interaction) and EQ2321 (Speech and Audio Processing) • <i>Rationale</i>: Share learning outcome related to sound perception in system design • <i>Integration Method</i>: Combine into single course covering interaction and speech processing 5. FDT3318 (Room Acoustics) and DT2217 (Advanced Music Acoustics) • <i>Rationale</i>: Overlap in sound measurement and simulation skills • <i>Integration Method</i>: Merge into comprehensive course on room acoustics and spatial audio <p>Implementation Recommendations:</p> <ul style="list-style-type: none"> • Conduct curriculum review to identify redundant content • Gather student feedback to ensure alignment with learning needs • Encourage faculty collaboration on integrated course materials • Start with pilot programs before full-scale implementation