

A Hybrid GraphRAG Approach to Generating Explainable Recommendations

Francisco-Javier Intriago-Usca
0000-0003-4685-3909
Universidad Técnica Particular de Loja,
Loja, Ecuador
Email: fjintriago1@utpl.edu.ec

Janneth Chicaiza
0000-0003-3439-3618
Universidad Técnica Particular de Loja,
Loja, Ecuador
Email: jachicaiza@utpl.edu.ec

Abstract—This paper proposes an architecture for hybrid recommender systems that integrates knowledge graphs and semantic representations of textual information. The architecture is based on the GraphRAG framework and has been designed with the aim of improving the relevance and explainability of recommendations by integrating knowledge graphs and large-scale language models. To validate the proposal, a prototype was implemented to provide recommendations related to the research of a specific topic. The functional prototype empirically validated the viability of the solution through a multidimensional evaluation, obtaining high levels of accuracy, relevance, and fidelity. The results showed that the integration of graphs and generative models significantly improves the quality, transparency, and reliability of recommendations, making a relevant contribution to the development of explainable recommendation systems.

Index Terms—GraphRAG, Knowledge Graphs, LLMs, Recommendation Systems, Explainability.

I. INTRODUCTION

THE exponential growth of information has highlighted the limitations of traditional recommender systems, which mainly suffer from problems of cold start, scalability, and a lack of interpretability [1], [2], [3]. To face these challenges, this paper presents the design and implementation of a hybrid recommender system that integrates knowledge graphs and textual information to generate explainable recommendations through large language models (LLMs).

On the one hand, Knowledge Graphs (KGs) have become established as essential data structures, capable of modeling complex semantic entities and their relationships [4]. By incorporating domain-specific knowledge and leveraging relationships between users and items, KG-based systems overcome the limitations of traditional approaches, offering more precise and contextually rich recommendations.

On the other hand, textual information is one of the most abundant types of data in organizations and on the web, including users' reviews and item descriptions. Nowadays, the most popular approach for analyzing text is the use of large language models. LLMs have proven to be effective in generating texts, even with few or no prior examples of the task to be performed [5], [6], [7].

In recent years, LLMs have been successfully integrated into the generation of explainable recommendations, thanks to their ability to process natural language and model complex

preferences. However, their reliance on parametric knowledge induces hallucinations that compromise the veracity of the suggestions. To mitigate this limitation, the Retrieval-Augmented Generation (RAG) paradigm has emerged because it integrates external contextual knowledge, thus reducing hallucinations. Despite this advance, conventional RAG is vulnerable to noise and is not able to acquire up-to-date knowledge. To overcome these challenges, this paper proposes a hybrid recommendation framework that identifies the user's search intent and, based on its classification, the system decides the most suitable retrieval approach according to the query type, thereby ensuring the recommendation of items in a more reliable and efficient manner.

The proposed approach tries to leverage both structured and unstructured information [8] and generate recommendations in a particular domain. Knowledge graphs provide a rich semantic picture of the domain, linking entities and relationships that add context and help to mitigate the lack of data often seen in cold-start situations. At the same time, LLMs improve the personalization and interpretability of the recommendations. In this context, the goal of this paper is to present an architecture that addresses the growing need to improve recommendation systems to generate explainable, accurate, and scalable recommendations.

This paper is organized as follows: Section II presents the research background and related work. Section III presents the recommendation system architecture. Section IV describes the implementation of a web prototype designed to allow users to search for and receive recommendations on topics and scientific documents related to the application of AI in health in Latin American countries. Section V explains the results of the evaluation. Finally, the conclusion is presented in the last section.

II. BACKGROUND

A. Recommender Systems based on Knowledge Graphs

In general, a recommendation system built on knowledge graphs includes three main components: the KG itself, a recommendation module, and a connection module [9]. The KG holds semantic details about different entities, while the recommendation module computes user-item relationships. Meanwhile, the connection module transforms that semantic

information from the KG into a low-dimensional vector and then blends it with the recommendation module to work out the overlap between users and items.

In real-world settings, graph-based recommendation systems have shown they work well across areas like education, shopping, finance, social networks, healthcare, and scientific research [10], [11]. These systems offer easy-to-understand recommendations while also safeguarding user data and meeting regulatory compliance.

B. Recommendations based on LLMs

LLMs have brought traceability and confidence to the field of recommendation systems. This statement is evidenced by the growing and real-time use of users with chat services that allow them to obtain not only answers to their questions but also offer them precise recommendations. Additionally, multimodal LLMs are capable of managing different types of data—such as text, images, videos, and audio—thus creating rich, flexible, and relevant recommendations.

Likewise, LLMs address key challenges in recommendation systems, such as the "cold start" [12], which is the difficulty in making recommendations to new users or those with limited data, the lack of diversity [13], and the limited explainability of classical methods [14]. Some methods combine traditional models to select candidates and then use LLMs to reorder and improve the quality of recommendations [15].

C. Explainable Recommendation Systems

Today, explainable recommendation systems (XRSs) have become an important tool because they try to deliver procedures more transparently, build users' confidence, and improve satisfaction with the recommendations they receive. When a system is able to clearly explain why a particular item was suggested, users can easily understand why an item was recommended; it promotes more meaningful interactions. Also, XRSs are easy to detect and correct errors or hidden biases within their recommendation models [16].

A common strategy for adding explainability to recommendation systems is the post-hoc approach. This solution gives users a clearer view of the factors that influence suggestions, which helps people better understand them [17]. Another strategy is the integration of visual explanations. Visualizing the reasons behind the recommendation significantly improves user understanding and interaction with the system [18]. Furthermore, approaches using KG and LLM have been proposed to enrich explanations, allowing complex decisions to be explained simply for the user [19].

D. Related Work

The convergence between the structural capacity of KGs and the generative power of LLMs gives rise to GraphRAG (Graph Retrieval-Augmented Generation). This architecture uses a specialized retrieval component that extracts the most relevant pieces of information from a graph database to feed the language model and generate contextual, accurate, and explainable responses. Recent literature has begun to explore the

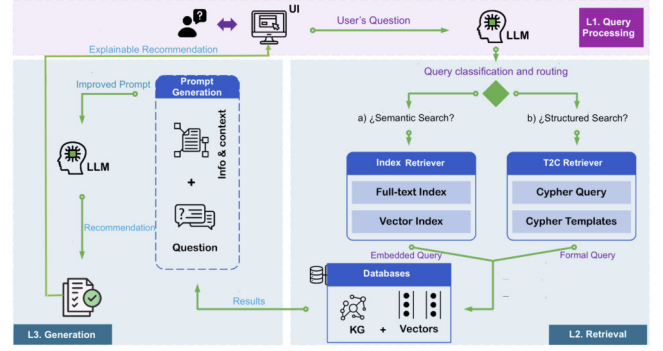


Fig. 1. Hybrid recommendation system architecture

potential of GraphRAG architectures to mitigate hallucinations and loss of context in complex recommendation systems.

To date, five GraphRAG proposals for recommendation have been found in Scopus. In the educational field, two proposals [20], [21] have been found aimed at optimizing teaching and learning processes using this type of advanced RAG architecture. On the other hand, in regulatory compliance domains, such as FinTech, the BondReco system, presented at [22], demonstrates how the integration of graph databases and GPT-4 models allows for the automation of regulatory compliance analysis with greater accuracy than plain text retrieval methods. In the field of smart tourism, [23] presents the use of GraphRAG-based frameworks to improve visitor interaction through personalized recommendations of routes and local products by facilitating real-time interactions that promote lesser-known destinations and support the local economy.

In addition, in the industrial sector, [24] proposes a recommendation system for customer service in the electricity sector that uses GraphRAG to integrate relational databases, guaranteeing the traceability of results and the security of private information. Finally, [25] introduces an analytics platform that uses ontologies and GraphRAG to turn complicated technical advice into easy-to-understand explanations for non-experts in IoT manufacturing settings.

The above works highlight the ability of graph-based architectures to provide recommendation systems with explanatory logic and data robustness superior to LLMs. The results confirm that knowledge graphs are effective in making information easier to access and more personalized in different recommendation settings.

III. HYBRID RECOMMENDATION SYSTEM ARCHITECTURE

This section describes the proposed architecture for the hybrid recommendation system. Figure 1 shows how the system is organized into three clearly differentiated functional layers, which operate sequentially and in a coordinated flow.

The architecture is based on the GraphRAG paradigm; that is, the architecture leverages the topological structure of the graph to enrich both the retrieval and the generation of explanations for the scientific recommendations. The relevant aspects of each layer are presented below.

Layer 1: Query Processing. This layer acts as the system's entrance and takes care of handling questions that users ask in natural language. Its main goal is to figure out what the user really needs and then decide whether the request calls for 1) a closer look at the text-based information, 2) turning it into a structured query to pull out specific facts from the graph, or 3) combining both strategies.

The system identifies the most appropriate retrieval approach to answer the user's query according to the query classification. In this work, we propose using an LLM to analyze the semantic load of the query and determine the most suitable retrieval strategy. This strategy reduces unnecessary searches and optimizes the use of computational resources; in this way, it is a key component in the overall efficiency of the system.

Layer 2: Retrieval. The second layer is the retrieval engine, which is responsible for accessing the knowledge stored in the databases: vector (created from unstructured content) and graph (created from structured data). Therefore, this layer operates under two complementary strategies:

- 2.1 Index Retriever: Combines vector indexes and full-text indexes to identify relevant documents when the query relates to concepts. A full-text index is based on traditional information retrieval techniques focused on lexical keyword matching. On the other hand, a vector index is built from embeddings generated using a language model and facilitates the identification of conceptually related documents.
- 2.2 T2C Retriever: An LLM acts as a natural language translator to Cypher queries, allowing direct access to graph relationships to answer factual queries. These are questions that seek specific and verifiable information within the graph and do not require extensive semantic inference, but rather direct access to nodes and relationships defined in the data model.

Furthermore, the two retrieval strategies can be combined; in this way, the system dynamically adapts its behavior to the type of query, overcoming the limitations of monolithic retrieval approaches.

Layer 3: Generation In this layer, an LLM creates the final recommendation for users using only the context retrieved from the second layer to provide an explicit justification for the recommendation.

To illustrate the feasibility and usefulness of the proposed architecture, a prototype has been implemented to generate explainable recommendations about knowledge and entities of scholarly publications.

IV. PROTOTYPE IMPLEMENTATION

To validate the proposed recommendation approach, a prototype was implemented over a graph of academic publications. Scholarly production is a knowledge-intensive domain that requires precise, traceable, and easy-to-understand information services.

The prototype's implementation follows the general architecture shown in Figure 1, but before explaining the relevant

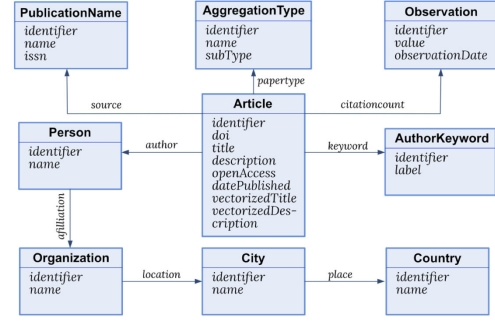


Fig. 2. Data model for metadata of scientific publications

technical details, the process of creating the knowledge graph, a core database for the recommender system, is described.

A. KG Creation for Scholarly Publications

The KG creation begins with defining the scope of application and choosing a scientific database. For this project, the scope was defined as "artificial intelligence applied to health in Latin American countries," and the Scopus database was chosen due to its being the most comprehensive scientific database with the broadest disciplinary coverage.

To collect documents' metadata, the Elsevier Scopus APIs were used by retrieving, on November 3, 2025, a total of 1,403 records, including detailed information for each article and its related entities, such as authors, organizations, keywords, etc. The results were stored in plain text files and then processed.

1) *Definition of the Data Schema:* Based on the extracted metadata, a conceptual graph schema was defined to ensure structural consistency, interoperability, and reusability of the model across different bibliographic sources. To achieve this, a set of uniqueness constraints was established for the primary identifiers of each entity to prevent duplication and ensure the integrity of the nodes and their relationships. Figure 2 shows the schema, which integrates nodes for articles, authors, organizations, keywords, publication types, and geographical locations, along with the relationships that describe semantic connections such as affiliation, provenance, and citation.

The data design made it possible to explicitly model the network of relationships inherent in scientific production, subsequently facilitating multirelational analyses, centrality measures, semantic searches, and connectivity-based recommendations.

2) *Preprocessing Data:* From the collected data, some preprocessing tasks were performed, such as term normalization. One of the main challenges was keyword processing, which was consolidated using similarity metrics and semantic expansion. In general, these terms exhibit lexical variations, abbreviations, semantic synonyms, and redundant terminology. The quality of this stage is fundamental, as keywords constitute central semantic nodes in the graph.

For the prototype, a multi-phase normalization process was developed using the `RapidFuzz` library, which allows for

measuring string similarity through fuzzy comparison algorithms. In this way, terms such as "Artificial intelligence," "Artificial Intelligence (AI)," and "artificial intelligence" were integrated into a single canonical form: "Artificial intelligence."

3) *Transformation of the Data into a Graph Format:* This step has two inputs: the data schema (step 1) and preprocessed data (step 2). The data was then reorganized into several plain text files (.CSV), each designed to independently represent entities and relationships, to build a standardized schema compatible with Neo4j and extensible to other document sources. In short, as many CSV files were created as there are nodes and relationships in the defined schema.

4) *Loading Data into the Storage and Embedding Creation:* For this project, Neo4j Desktop 2 was used to store the graph. This platform allows for managing local instances, running Cypher queries, and enabling advanced features through specialized plugins.

From the CSV files, and using Cypher commands, the graph's nodes, relationships, and constraints were created. In addition to data loading, the GenAI plugin was configured. This plugin integrates generative modeling and embedding capabilities within Neo4j, thus facilitating the vectorization of textual information from the nodes using external models, including those provided by OpenAI.

Once the knowledge graph was created from the publication metadata, the nodes were enriched with vectorized semantic representations (embeddings). Embeddings were created using OpenAI's `text-embedding-3-small` model and applied to publication titles and abstracts. This model was chosen for its balance between efficiency, reduced dimensionality, and semantic quality.

The last step generated an additional property for each article, containing a high-dimensional numerical vector that represents the semantic content of the textual metadata. These vectors allow for a) similarity searches, b) semantic distance calculations, c) enrichment of the recommendation system, and d) integration of hybrid techniques combining content-based and graph-based methods.

After vectorizing abstracts and titles, all Article nodes in the graph were enriched with dense representations. Table II presents a summary of the information contained in the graph.

The resulting graph constitutes the database core of the proposed hybrid recommender system, enabling the modeling of entities, semantic relationships, and enriched properties using generative artificial intelligence techniques.

5) *Creation of Search Indexes:* To improve retrieval strategies, advanced indexing mechanisms were created to efficiently exploit both the structural information and the textual content associated with the nodes. In this context, two complementary types of indexes were created: vector indexes and full-text indexes. The incorporation of these indexes is a key component of the proposed hybrid approach, as it enables similarity-based semantic queries, efficient text searches, and information retrieval mechanisms explainable within the graph.

- Vector indexes are specialized structures designed to store and query high-dimensional numerical representations (embeddings). In the context of this research, the embeddings represent the semantic content of article titles and abstracts, created using the `text-embedding-3-small` model. Unlike traditional indexes, vector indexes allow for searches based on semantic similarity, calculating the closeness between vectors using metrics such as cosine similarity. This type of indexing is particularly well-suited for content-based recommendation systems, where the goal is to identify conceptually related documents (such as related works), even when they do not share the same terms.
- Full-text indexes are designed to optimize the search for text strings based on alphanumeric properties. This feature simplifies partial matches and allows for queries with logical operators, retrieving relevant results based on textual similarity. They also complement vector indexes, facilitating fast and precise lexical searches on key entities within the graph. These indexes allow, for example, the rapid location of specific authors or research groups, the retrieval of institutions by partial match or name variations, or the exploration of the graph using thematic concepts expressed as keywords. Thanks to these characteristics, this type of index is ideal in scenarios where the user employs specific terms or names in their query.

This dual-index architecture forms the retrieval backbone of the hybrid recommendation system, enabling both conceptual similarity searching and precise entity-based searching within a unified graph structure.

B. Conversational Interface Development

User interaction is facilitated through a Streamlit-based conversational interface that receives natural language queries, maintains session history, and optionally displays a visual representation of the underlying graph substructure for each recommendation. As part of the output, the interface presents source traceability information, reinforcing the system's transparency goals.

1) *Layer 1: Query Processing:* When the system receives a user query, the LLM-based analysis component performs two simultaneous functions: 1) it standardizes the linguistic form of the input and 2) determines the most appropriate retrieval strategy for the query. The LLM operates under a zero-shot prompting scheme in which a system request, anchored to the domain, guides the model toward one of two retrieval strategies. The LLM's output is generated in a structured JSON format, facilitating its direct integration with the control logic of the system implemented in Python.

The logic for classifying the query, and the subsequent decision on the best retrieval approach, is detailed in algorithm 1.

Then, depending on the type and complexity of the query, the LLM, based on the logic of the above algorithm, defines what the most efficient retrieval path is.

TABLE I
GRAPH ENTITIES (E) AND RELATIONSHIPS (R)

Label	Type	Description	Unique Number of Items
AggregationType	E	Publication Type. Metadata: name, subType.	16
Article	E	Identify an academic publication. Metadata: identifier, doi, title, description, openAccess, datePublished, vectorizedTitle, vectorizedDescription	1403
AuthorKeyword	E	Keywords defined by the authors. Metadata: name	3226
City	E	City of an organization. Metadata: name	1311
Country	E	Country of an organization. Metadata: name	124
Observation	E	Number of citations of a paper. Metadata: Value, observationDate	87
Organization	E	Authors' Institutional Affiliations. Metadata: identifier, name	3532
Person	E	Author of a publication. Metadata: identifier, name	8331
Publication	E	Journal and/or Proceedings of a conference. Metadata: identifier, name, issn	787
AFFILIATION	R	Person $\rightarrow$ Organization	10014
AUTHOR	R	Article $\rightarrow$ Person	9473
CITATIONCOUNT	R	Article $\rightarrow$ Observation	1403
KEYWORD	R	Article $\rightarrow$ AuthorKeyword	6523
LOCATION	R	Organization $\rightarrow$ City	3532
PLACE	R	City $\rightarrow$ Country	1311
SOURCE	R	Article $\rightarrow$ Publication	1403
TYPE	R	Article $\rightarrow$ AggregationType	1403

Algorithm 1 Hybrid Retrieval Strategy Selection

```

1: Input: User query  $Q$ 
2: Output: Retrieved context  $C_{final}$ 
3:  $(S, E) \leftarrow \text{ClassifyQuery}(Q)$   $\{S$ : Semantic,  $E$ : Entities $\}$ 
4: if  $S = \text{High}$  and  $E = \text{Low}$  then
5:   {Strategy 1 (INDEX_RETRIEVER): Vector-based}
6:    $C_{final} \leftarrow \text{VectorRAG}(Q)$ 
7: else if  $S = \text{Low}$  and  $E = \text{High}$  then
8:   {Strategy 2 (T2C_RETRIEVER): Graph-based}
9:    $C_{final} \leftarrow \text{GraphRAG}(Q)$ 
10: else
11:   {Strategy 3 (HYBRID): Vector-based + Graph-based}
12:    $C_v \leftarrow \text{VectorRAG}(Q)$ 
13:    $C_g \leftarrow \text{GraphRAG}(Q)$ 
14:    $C_{final} \leftarrow \text{Combine}(C_v, C_g)$ 
15: end if
16: return  $C_{final}$ 

```

- Index Retrieval (VectorRAG): Activated with high S and low E . Ideal for queries whose information needs are based on concepts, research topics, or subject areas, without explicit reference to graph entities, are assigned to the semantic search path.
- T2C Retrieval (GraphRAG): It is activated with low S and high E . It is used when the user's query makes reference to specific entities present in the knowledge graph, such as author names, institutional affiliations, countries, or publication years.
- Hybrid Retrieval: It is activated when both indicators are high. It combines both methods to resolve highly complex queries that require both thematic context and

relational precision.

Before describing the next layer, it's important to highlight that intent detection influences both the retrieval and the nature of the explanation offered to the user. While index searches (or semantic) generate explanations based on conceptual similarity, T2C searches produce answers grounded in explicit graph relationships, strengthening the system's transparency.

2) *Layer 2: Retrieval*: The retrieval engine is responsible for accessing the knowledge stored in Neo4j. It constitutes the operational core of the proposed system, acting as the bridge between the user's expressed intent and the knowledge stored in the graph. Its function is to identify, filter, and enrich the most relevant pieces of information that will serve as evidence for the generation phase.

The retrieval layer was designed using a hybrid and adaptive approach, allowing for the combination of semantic and structured search strategies based on the intent identified and classified in the previous layer. Therefore, this layer operates under two complementary strategies:

- Index retrieval combines vector indexes and full-text indexes to identify scientifically relevant articles when the query is based on concepts or subject areas.
- T2C retrieval: The LLM acts as a natural language translator for Cypher queries, allowing direct exploitation of graph relationships to answer factual queries; that is, questions seeking specific and verifiable information within the Cypher query that do not require extensive semantic inference but rather direct access to nodes and relationships defined in the data model.

The coexistence of these strategies allows the system to dynamically adapt its behavior to the type of query, overcoming the limitations of monolithic retrieval approaches.

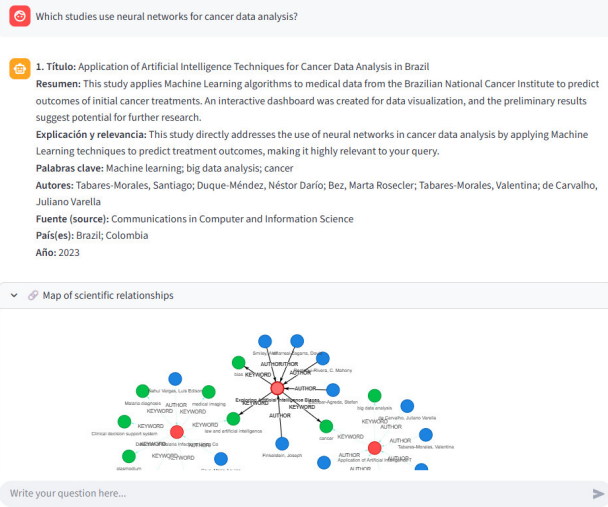


Fig. 3. Output generated by the system in response to a given query

3) *Layer 3: Generation*: The final layer of the process produces the explainable recommendation that is returned to the user. The generative model synthesizes the response solely from the context provided by the retriever in the previous layer. Each recommendation is structured around four traceable components: 1) A contextualized introduction generated by the model, based on the query framework; 2) An executive summary derived directly from the article's description attribute in the knowledge graph; 3) A relevance justification inferred by the model from the correspondence between the query and the retrieved evidence; and 4) Metadata explicitly extracted from the graph's edges.

The OpenAI GPT-4o model was used to generate the user's output. It was selected for its balance between reasoning ability, strict instruction following, and low latency. As part of the configuration, a temperature of 0 was set to ensure deterministic and reproducible responses.

The orchestration between the generative model and the retrieval modules is performed using the Python library `neo4j-graphrag`. This approach integrates the retrieved context, the generation template, and the original user query, all within a controlled flow.

The system's explainability is achieved through the design of generation templates (RAGTemplates), which act as a cognitive control mechanism, restricting the model's behavior and guiding it to produce relevant responses. Figure 3 illustrates a system's response for a specific user query; the output shows clear, justifiable, and traceable explanations.

The natural language explanations about the reasoning behind each recommendation are generated thanks to prompts that are dynamically created according to the semantic structure of the graph and the context provided by the retriever.

V. EVALUATION

To empirically validate the proposed system, a multidimensional evaluation framework was designed and applied to

TABLE II
MULTIDIMENSIONAL EVALUATION FRAMEWORK

Dimension	Metric	Description
Response quality (RAG Triad)	Faithfulness	Degree to which the generated response is grounded in the retrieved context, without unsupported content
	Answer Relevance	Degree to which the response directly addresses the user's query intent
	Context Relevance	Degree to which the retrieved context was pertinent and necessary for generating the response
Graph-specific effectiveness	Cypher Structural Correctness	Syntactic validity and structural adequacy of LLM-generated Cypher queries
Ranking	Precision@5	Proportion of relevant articles within the top-5 ranked recommendations
	MRR (Mean Reciprocal Rank)	Average reciprocal rank of the first relevant result, measuring ranking prioritization
Operational performance	Intent Classification Accuracy	Proportion of queries correctly routed to the appropriate retrieval strategy
	Response Latency	End-to-end time elapsed from query submission to final response delivery

a controlled set of queries. Given the hybrid nature of the architecture, the evaluation addresses several dimensions, in line with recent recommendations in the literature on RAGs and hybrid recommendation systems [26], [27].

A. Evaluation Setting

A test set of 100 natural language queries was constructed to reflect realistic use cases within the corpus domain. The queries were distributed into three categories aligned with the system's intent classification strategies. The categories were: Q1) Queries focus on general concepts or themes, i.e., queries without named entities. Q2) Queries referencing explicit entities (authors, institutions, countries, years). Q3) Queries outside the corpus domain or non-informative.

Also, the evaluation environment was organized around four dimensions, each targeting a distinct functional stage of the GraphRAG workflow. Table II presents each evaluation dimension with its metrics.

The RAG triad metrics were assessed using LLM-assisted evaluation, where a judge model rated each response on a scale of 0 to 1 based on the three criteria. Cypher's structural correctness was assessed by running the generated queries on the live Neo4j instance and inspecting both syntactic validity and the consistency of the result. Precision@5 and MRR were calculated by comparing them to relevance judgments manually annotated by domain experts. Intent classification accuracy was determined by comparing the classifier's routing decision with the reference strategy assigned during query set construction.

B. Results

Table III consolidates the average scores for each evaluation dimension, providing an overall view of system performance at all stages of the pipeline.

TABLE III
AGGREGATE EVALUATION RESULTS BY DIMENSION

Evaluation Dimension	Metrics	Average Score
Response quality (RAG Triad)	Faithfulness, Answer Relevance, Context Relevance	95.66 %
Graph-specific effectiveness	Cypher Structural Correctness	96.00 %
Recommendation ranking	Precision@5, MRR	94.00 %
Operational performance	Intent Classification Accuracy	93.00 %
Overall average		94.67 %

The four dimensions consistently reach 90%, reflecting balanced and stable integration throughout the process. However, it is important to note that the system operates under a sequential pipeline, where each stage depends directly on the output of the previous one. In this context, the correct selection of the query routing strategy is a critical decision point, as it determines the retrieval model to be used. When the classification is incorrect, the system activates a retrieval mechanism that is not aligned with the user’s actual needs, leading to the retrieval of partially relevant or even irrelevant context. Consequently, subsequent stages are affected, producing recommendations that, while internally consistent, do not meet the user’s expectations.

The cascading effect explains why, despite the high average values, slight decreases are observed in some metrics, demonstrating that the overall system performance is strongly dependent on the accuracy of the initial routing strategy. Therefore, to complement the quantitative results and avoid a positively biased interpretation, a systematic analysis of the failure cases was performed. Three representative failures were identified and classified into two types, as summarized in Table IV.

Case 1 reveals a systematic bias toward semantic signals when a query combines a named entity with thematic vocabulary. The model fails to detect the dominant structural anchor, causing the query to be incorrectly routed to the vector retriever. *Case 2* exposes a complementary limitation: when a structurally filterable attribute (publication year) is embedded in a semantically rich query, the classifier tends to prioritize the conceptual framework over the explicit time constraint. *Case 3* represents a distinct failure mode, confined to Cypher’s generation module. The LLM generated a query referencing an assumed attribute that was not present in the graph schema. This type of error stems from the model’s tendency to extrapolate plausible graph attributes from its parametric knowledge, rather than strictly adhering to the provided schema.

These findings suggest two concrete lines of improvement for the system: 1) replacing the binary intent classification with a multi-label or confidence-scoring classifier capable of handling hybrid intent queries, and 2) integrating a post-generation schema validation module for Cypher queries that allows intercepting attribute-level errors before graph execution.

VI. CONCLUSION

Recommendation approaches that integrate knowledge graphs and generative models represent a trend that attempts to overcome the structural limitations of traditional recommendation systems. However, the literature review revealed that, although various conceptual proposals exist, very few studies materialize complete and comprehensively evaluated architectures.

This paper presents a GraphRAG-based hybrid recommendation that integrates the retrieval of knowledge graphs and embedding-based indexes with the capabilities of generative language models to produce explainable recommendations in real-time. The system addresses three fundamental limitations of conventional recommendation approaches: 1) sensitivity to cold starts, 2) limited semantic expressiveness, and 3) opacity of the recommendation process by grounding each recommendation in explicit and traceable evidence extracted from a knowledge graph.

The architecture of the recommendation system is divided into three layers: L1) a query processing component that standardizes the input and selects the appropriate retrieval strategy; L2) a hybrid retrieval approach that is activated according to the type of the user’s query; and L3) the generation stage in which the LLM produces structured and traceable recommendations from a context.

Through the implementation of a functional prototype, created to produce recommendations for scientific literature, the technical feasibility of the proposed architecture was validated, demonstrating that it is possible to generate explainable recommendations in real time with high levels of accuracy, reliability, and operational performance. The results of the comprehensive evaluation showed scores exceeding 90% in all analyzed dimensions, along with a response time suitable for interactive environments.

However, the evaluation also identified inherent limitations of LLMs: incorrect strategy selection for activating the most appropriate retriever and incorrect translation of user queries into Cypher queries. These findings provide insights for future architectural improvements.

Future work will focus on optimizing the search intent classifier using a multi-label classification scheme. This will enable the system to handle queries with simultaneous structural and semantic intent, thereby reducing the classification error rate in hybrid queries. Additionally, a Cypher query validation module will be integrated to intercept references to invalid entities or properties, eliminating the primary cause of query failures identified in this study. Finally, it is important to scale the prototype to a production environment and also evaluate its behavior using real users, incorporating acceptance metrics and thus estimating the perceived usefulness in a real context.

GENERATIVE AI DISCLOSURE STATEMENT

During the preparation of this work, the authors used Grammarly and Gemini 3 Fast to proofread the text and correct linguistic errors, thereby improving its academic quality. After

TABLE IV
FAILURE CASES DURING QUERY CLASSIFICATION AND ROOT CAUSES

Case	Query (paraphrased)	Expected Strategy	Assigned Strategy	Failure Type	Root Cause
1	Recommend authors in the same research line as a <i>named author</i>	T2C_RETRIEVAL	INDEX_RETRIEVAL	Misclassification of intent	Overweighting of semantic signals over named entity
2	Recommend ML models for disease prediction published in 2025	T2C_RETRIEVAL	INDEX_RETRIEVAL	Misclassification of intent	Failure to detect temporal structural filter
3	Most cited papers by authors from Peru	T2C_RETRIEVAL	T2C_RETRIEVAL (correct)	Cypher generation error	LLM inferred a non-existent citation attribute

using these tools, the authors review and edit the content as needed and take full responsibility for this publication's content.

REFERENCES

- [1] J. Zhou, D. Shen, Y. Guo, Y. Zhao, and H. Zhang, "Enhancing knowledge graph recommendations through deep reinforcement learning," *Scientific Reports*, vol. 16, no. 1, p. 3076, Jan. 2026, doi: 10.1038/s41598-025-31109-8.
- [2] J. Chen and T. Zhu, "Personalized Learning Resource Recommendation Framework Based on Knowledge Graph and Large Language Model," in *Knowledge Science, Engineering and Management*, T. Zhu, W. Zhou, and C. Zhu, Eds. Singapore: Springer Nature Singapore, 2026, pp. 228–245.
- [3] J. Z. Pan, S. Razniewski, J.-C. Kalo, S. Singhanian, J. Chen, S. Dietze, H. Jabeen, J. Omeliyanenko, W. Zhang, M. Lissandrini, R. Biswas, G. de Melo, A. Bonifati, E. Vakaj, M. Dragoni, and D. Graux, "Large language models and knowledge graphs: Opportunities and challenges," 2023.
- [4] J. Chicaiza and P. Valdiviezo-Diaz, "A Comprehensive Survey of Knowledge Graph-Based Recommender Systems: Technologies, Development, and Contributions," *Information*, vol. 12, no. 6, 2021, doi: 10.3390/info12060232.
- [5] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.
- [6] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh, "AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, B. Webber, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 4222–4235, doi: 10.18653/v1/2020.emnlp-main.346.
- [7] T. Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, "Calibrate before use: Improving few-shot performance of language models," 2021.
- [8] E.-R. Luțan and C. Bădică, "Literature books recommender system using collaborative filtering and multi-source reviews," *Annals of Computer Science and Information Systems*, no. 2024, p. 225 – 230, 2024.
- [9] C. Zhang, X. Huang, J. An, and S. Zou, "Improving conversational recommender systems via multi-preference modelling and knowledge-enhanced," *Knowledge-Based Systems*, vol. 286, p. 111361, 2024, doi: 10.1016/j.knosys.2023.111361.
- [10] C. Peng, F. Xia, M. Naseriparsa, and F. Osborne, "Knowledge Graphs: Opportunities and Challenges," *Artificial Intelligence Review*, vol. 56, no. 11, pp. 13 071–13 102, Nov. 2023, doi: 10.1007/s10462-023-10465-9.
- [11] S. Raza, M. Rahman, S. Kamawal, A. Toroghi, A. Raval, F. Navah, and A. Kazemeini, "A comprehensive review of recommender systems: Transitioning from theory to practice," *Computer Science Review*, vol. 59, p. 100849, 2026, doi: 10.1016/j.cosrev.2025.100849.
- [12] C. Liu, D. Jiang, Y. Cai, and H. Li, "A review of deep learning and large language models for cold start problem in recommender systems," *WIREs Data Mining and Knowledge Discovery*, vol. 16, no. 1, p. e70068, 2026, doi: 10.1002/widm.70068.
- [13] J. Chen, C. Gao, S. Yuan, S. Liu, Q. Cai, and P. Jiang, "Dlrec: A novel approach for managing diversity in llm-based recommender systems," in *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, ser. WSDM '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 857–865, doi: 10.1145/3701551.3703572.
- [14] G. Zeng, G. Tian, G. Zhang, and J. Lu, "RoSiLC-RS: A Robust Similar Legal Case Recommender System Empowered by Large Language Model and Step-Back Prompting," *Neurocomput.*, vol. 648, no. C, Oct. 2025.
- [15] A. García-Martín, "Grandes modelos de lenguaje en sistemas de recomendación," 2024.
- [16] D. Afchar, A. B. Melchiorre, M. Schedl, R. Hennequin, E. V. Epure, and M. Moussallam, "Explainability in music recommender systems," *AI Magazine*, vol. 43, no. 2, p. 190–208, Jun. 2022, doi: 10.1002/aaai.12056.
- [17] J. Govea, R. Gutierrez, and W. Villegas-Ch, "Transparency and precision in the age of ai: evaluation of explainability-enhanced recommendation systems," *Frontiers in Artificial Intelligence*, vol. Volume 7 - 2024, 2024, doi: 10.3389/fraci.2024.1410790.
- [18] O. Adegunwa, A. Oduroye, A. Akanni, F. Ugbeshe, and O. Ajilore, "Explainable recommender system: a survey and future directions," in *Proceedings of the Caleb University Disruptive Innovation Conference (CUDIC)*, 2023, 11 2023.
- [19] G. Balloccu, L. Boratto, G. Fenu, and M. Marras, "Post processing recommender systems with knowledge graphs for recency, popularity, and diversity of explanations," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 646–656, doi: 10.1145/3477495.3532041.
- [20] X. Cheng, Z. Zhang, J. Wang, L. Fang, C. He, Q. Guan, S. Pan, and W. Luo, "Graphrag-induced dual knowledge structure graphs for personalized learning path recommendation," K. S., J. C., and T. M.E., Eds., vol. 40, no. 17. Association for the Advancement of Artificial Intelligence, 2026, pp. 14 610 – 14 620, doi: 10.1609/aaai.v40i17.38479.
- [21] Y. Li and L. Zheng, "EduRAG: A RAG Framework for Academic Expert Recommendation," *Lecture Notes in Computer Science*, vol. 15860 LNCS, pp. 173 – 182, 2025, doi: 10.1007/978-981-96-9815-8_15.
- [22] R. T. S. Babu, A. Soundara Pandian, P. Sawant, and S. Prakashan, "Bondreco - a responsible ai solution for 401(k) investments," S. S., P. A., and M. V.P., Eds. Institute of Electrical and Electronics Engineers Inc., 2025, pp. 183 – 190.
- [23] P. Krataithong, W. Buranasing, M. Buranarach, T. Wuthitasarn, P. Meeklai, and P. Tumsangthong, "Tourism chatbot framework: Enhancing visitor experience through graphrag and ai chatbot." Institute of Electrical and Electronics Engineers Inc., 2025.
- [24] S. Xu, L. Zhu, B. Peng, J. Ran, J. Fu, and C. Huang, "Generalization method of power customer service knowledge recommendation model based on language model+graphrag," D. R., C. S.-K., S. J., and G. P., Eds., vol. 13665. SPIE, 2025.
- [25] M. Niederhaus, N. Migenda, J. Weller, M. Kohlhasse, and W. Schenck, "Integrating graph retrieval-augmented generation into prescriptive recommender systems," *Big Data and Cognitive Computing*, vol. 9, no. 10, 2025.
- [26] A. Brown, M. Roman, and B. Devereux, "A systematic literature review of retrieval-augmented generation: Techniques, metrics, and challenges," *Big Data and Cognitive Computing*, vol. 9, no. 12, 2025. [Online]. Available: <https://www.mdpi.com/2504-2289/9/12/320>
- [27] A. Kimothi, *RAG Evaluation: Accuracy, Relevance, Faithfulness*. Manning Publications, 2025, p. 256, eBook.