Saad Dahlab University of Blida 1

Faculty of Sciences

Department of Computer Science

Master Thesis

Computer Science
Option: Intelligent Systems and Software Engineering

Theme: Improving Medical Information Retrieval via Knowledge Graph-Enhanced Graph Neural Networks

By: Miss. DJAIT Ikram and Miss. ZERARI Sarah

Defended on June 29 , 2025, in front of the jury members:

Dr. L.OUKID	President	University of Blida 1
Dr. BERRAMDANE	Examiner	University of Blida 1
Dr. M. MEZZI	Supervisor (Internal)	University of Blida 1
Dr. M. FAREH	Co-Supervisor (Internal)	University of Blida 1

Acknowledgments

First and foremost, we give thanks to Allah, the Almighty, who, through His mercy, granted us the health, strength, and will to carry out our efforts, and surrounded us with wonderful people whom we wish to warmly thank.

We express our deep gratitude to Mrs.MEZZI for her unwavering support, her trust in us, and her invaluable help throughout this journey. We also extend our thanks to Mrs. FAREH, our co-supervisor, for her availability, valuable advice, and guidance, which greatly contributed to the success of our project. We are honored to have worked with your team and to have benefited from your expertise.

We would like to sincerely thank all those who contributed, directly or indirectly, to the completion of this thesis.

Finally, we extend our heartfelt thanks to our families: our parents, brothers, sisters, and friends, who supported, encouraged, and stood by us tirelessly throughout the preparation of this thesis. Their presence was a great source of comfort.

Ikram & Sarah

Dedication

I dedicate this thesis:

*To **my dearest mother**, my first guide, my eternal source of inspiration, and the reason I keep going. No matter what I say or do, words will never be enough to express the depth of my gratitude. Your presence, love, and priceless advice have been my greatest treasure. Your light and unconditional love continue to guide me every step of the way. I love you, Mom, and I pray to the Almighty to bless you with health, happiness, and a long, peaceful life. This humble work is the fruit of all the sacrifices you've made for my education and growth. Thank you from the bottom of my heart.*

*To **my father**, No words could ever truly express the immense gratitude and deep love I hold for you. Your strength and quiet support have been pillars in my life.*

*To **my beloved grandmother** may Allah have mercy on her soul. She was the kindest and most selfless person in my life. Her love, wisdom, and warmth shaped who I am today. I carry her memory in my heart every single day. I love you always, Grandma.Allah yerhamha.*

*To you, **my big brother Sofiane**, I deeply appreciate your constant encouragement and unwavering support. You've always been there for me, correcting my mistakes and lifting me up. I feel truly lucky to have you in my life.*

*To you, **my dear sister Amina**, Your presence is a beautiful gift that brings love and joy into my life. Thank you for being such a light to my days.*

*To you, **my little brother Islam**, Every comforting word, every laugh we've shared is forever engraved in my heart. Your support means more to me than I can say.*

*To **my binôme Ikram**, Thank you for being a wonderful partner throughout this journey. Your determination, patience, and collaborative spirit made this experience richer and more rewarding. I am grateful for every challenge we faced together and every success we celebrated side by side.*

*I would also like to express my sincere gratitude to the amazing friends who made this journey brighter: **Imene**, **Lydia**, **Anfel**, **Kawther**, **Lina**, and **Nihed**. Your support, laughter, and kindness have brought warmth and motivation throughout these university years.*

*A heartfelt thanks to the entire **IL** class. We've shared a path filled with challenges and triumphs, and your camaraderie made it unforgettable.*

Thank you all for being part of this journey.

Sarah Zerari

Dedication

I dedicate this thesis:

*To **my beloved mother**, Your love, sacrifices, and strength are the reason I made it this far. Even though you're no longer with me, every success I achieve is a reflection of the light you left in my life. Thank you for everything. I carry your memory with me, always.*

*To **my father**, Thank you for your unwavering trust in me and in my abilities. Your quiet confidence gave me strength when I doubted myself, and your belief in my success helped me believe in it too.*

*To **my brother**, Thank you for being my constant support in all times — good and bad. Your presence, your strength, and your love have meant more than words can express. You were always there when I needed a hand to hold.*

*To **my grandfather**, Thank you for being my rock after my mother's passing. Your unwavering support, your pride in me, and your belief in my potential gave me the courage to move forward. You've been my greatest source of motivation.*

*To **his wife**, who became a second mother to me, Your kindness, care, and tenderness touched my soul in ways I will never forget. You protected me, loved me, and filled my life with warmth. You are truly one of the purest hearts I've ever known.*

*To **Meriem**, My cousin, my sister in spirit, the one who rejoices in my joy and shares my sorrow. You were my emotional anchor, my safe space, my biggest supporter. Your presence meant the world to me.*

*To **Nassiba**, My dearest friend and companion throughout this academic journey. Your kindness, loyalty, and constant encouragement made this path bearable and beautiful. You are more than a friend — you are a part of this success.*

This is not just my achievement – it's ours.

Djait Ikram

Abstract

Knowledge Graphs are employed to refer to entities and their relationships in a semantic, structured form that enhances information organization and retrieval. However, due to their nature and size, traditional Information Retrieval techniques typically do not work for use against knowledge graphs.

This thesis proposes a cutting-edge Information Retrieval approach based on Knowledge Graph Embeddings and Relational Graph Convolutional Networks . Dense vector representations of entities are initially acquired using KGE models to encode semantic relations. These embeddings are afterwards fine-tuned using R-GCN to incorporate structural and relation knowledge from the graph. The learned embeddings are used for retrieving relevant entities or concepts by semantic similarity.

The system is evaluated across several scenarios, namely: the evaluation of embedding and similarity search techniques using standard metrics, the evaluation of queries using a semantic similarity module based on Bio_ClinicalBERT and the exploitation of the Knowledge Graph in a question-answering use case. Results indicate that adding KGE to R-GCN improves retrieval quality in a way it is possible to have better and more contextual search results.

Keywords: Information Retrieval, Knowledge Graph, Knowledge Graph Embeddings, Relational Graph Convolutional Network, Semantic Search, Medical Knowledge Graphs.

الملخص

تُستخدم الرسوم البيانية المعرفية لتمثيل الكيانات والعلاقات بينها بشكل دلالي ومنظم، مما يسهل عملية تنظيم المعلومات والبحث عنها. ومع ذلك، وبسبب تعقيدها وحجمها الكبير، فإن تقنيات البحث التقليدية غالبًا ما تكون غير مناسبة لهذا النوع من البيانات.

يقترح هذا العمل مقاربة متقدمة للبحث عن المعلومات، تعتمد على تضمينات الرسوم البيانية المعرفية (تضمين الكيانات والعلاقات في الفضاء الشعاعي) والشبكات العصبية الالتفافية العلائقية. في البداية، يتم توليد تمثيلات متجهية كثيفة للكيانات باستخدام نماذج تضمين الرسوم البيانية المعرفية بهدف التقاط العلاقات الدلالية. ثم يتم تحسين هذه التمثيلات باستخدام شبكة عصبية التفافية علائقية لدمج المعلومات الهيكلية والدلالية الخاصة بالرسم البياني المعرفي. تسمح التضمينات النهائية بالبحث عن الكيانات ذات الصلة بناءً على التشابه الدلالي.

تم تقييم النظام عبر عدة سيناريوهات، من بينها: تقييم تقنيات التضمين والبحث عن التشابه باستخدام مقاييس معيارية، بالإضافة إلى استغلال الرسم Bio_ClinicalBERT، وتقييم الاستعلامات باستخدام وحدة تشابه دلالي تعتمد على نموذج البياني المعرفي في حالة استخدام من نوع سؤال جواب.

تشير النتائج إلى أن دمج نماذج التضمين مع الشبكة العصبية الالتفافية العلائقية يُحسن من جودة البحث، مما يسمح بالحصول على نتائج أكثر صلة وسياقية.

الكلمات المفتاحية: البحث عن المعلومات، الرسم البياني المعرفي، تضمينات الرسم البياني المعرفي، الشبكات العصبية الالتفافية العلائقية، البحث الدلالي، الرسم البياني الطبي

Résumé

Les graphes de connaissances permettent de représenter les entités et leurs relations sous une forme sémantique et structurée, facilitant ainsi l'organisation et la recherche d'informations. Cependant, en raison de leur complexité et de leur taille, les techniques traditionnelles de recherche d'information ne sont généralement pas adaptées à ce type de données.
Ce mémoire propose une approche avancée de la recherche d'information, basée sur les embeddings des graphes de connaissances et les réseaux de neurones convolutionnels relationnels. Nous générons d'abord des représentations vectorielles denses des entités à l'aide de modèles KGE afin de capturer les relations sémantiques. Ces représentations sont ensuite raffinées grâce à R-GCN, pour intégrer les informations structurelles et relationnelles du graphe. Les embeddings finaux permettent de rechercher des entités pertinentes par similarité sémantique.
Le système a été évalué à travers plusieurs scénarios, notamment : l'évaluation des techniques d'embedding et de recherche de similarité à l'aide de métriques standards, ainsi que l'évaluation des requêtes à l'aide d'un module de similarité sémantique basé sur Bio_ClinicalBERT,et l'exploitation du graphe de connaissances dans un cas d'usage de type question-réponse . Les résultats indiquent que l'ajout des modèles d'embedding au réseau R-GCN améliore la qualité de la recherche, permettant ainsi d'obtenir des résultats plus pertinents et contextuellement adaptés.

Mots clés : Recherche d'information, Graphe de connaissances, Embeddings de graphe de connaissances, R-GCN, Recherche sémantique, Graphe médical.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Approximate Nearest Neighbor
AP	Average Precision
BERT	Bidirectional Encoder Representations from Transformers
Bio-ClinicalBERT	Biomedical and Clinical version of BERT
ComplEx	Complex-valued Embedding Model
ConvE	Convolution-based Embedding Model
CSV	Comma-Separated Values
DB	DrugBank
DistMult	Distance Multiplication Embedding Model
EMRs	Electronic Medical Records
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GNN	Graph Neural Network
Graphormer	Transformer-based Graph Neural Network model
HNSW	Hierarchical Navigable Small World
ICD-9	International Classification of Diseases, 9th Revision
ICD-9-CM	ICD-9 Clinical Modification
IR	Information Retrieval
KG	Knowledge Graph
KGAT	Knowledge Graph Attention Network
KGE	Knowledge Graph Embedding
LLM	Large Language Model
MAP	Mean Average Precision
MIMIC-III	Medical Information Mart for Intensive Care III
MIR	Medical Information Retrieval
MIRNA	Micro Ribonucleic Acid
MMGAT	Multi-Modal Graph Attention Network
MRR	Mean Reciprocal Rank

nDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing
P@k	Precision at rank k
PDD	Patient-Disease-Drug (knowledge base)
R@k	Recall at rank k
RDF	Resource Description Framework
RDF2Vec	RDF Embedding model based on Word2Vec
R-GCN	Relational Graph Convolutional Network
RotatE	Rotational Embedding Model
TF-IDF	Term Frequency – Inverse Document Frequency
TransE	Translation-based Embedding Model
XML	Extensible Markup Language

General Introduction

General Context

In the last few years, the volume of medical data ranging from clinical reports to biomedical research has grown exponentially, which poses an acute need for effective and wise Medical Information Retrieval Systems. Traditional IR models are often unable to capture the fine semantic and relational characteristics of Medical Knowledge. Of particular concern in medicine, where the access to correct information directly affects diagnosis and treatment quality. To this end, Knowledge Graphs have emerged as a useful representation framework for capturing the intricate interrelations between diseases, drugs, and procedures. Meanwhile, Graph Neural Networks, especially Relational Graph Convolutional Networks, have shown extremely promising in leveraging such structured information to learn informative, context-aware embeddings.

Research Problem

The fundamental problem that this study aims to address is the inadequate representation of objects and relationships in Medical Knowledge Graphs, which significantly degrades the performance of current medical Information Retrieval Systems. Most classical Knowledge Graph Embedding models cannot adequately capture the rich contextual, structural, and relational information inherent in biomedical data. This results in Retrieval Systems that are too rigid, too general, or just plain unable to retrieve the correct information based on advanced clinical inquiry.

Existing KGE models typically lack to be independent and do not have any way of leveraging graph topology or intra-node relations that are central in a healthcare setting. Therefore, such models struggle to understand implicit relations and context-dependent semantics, both of which are fundamental in enabling decision-making in healthcare.

This study seeks to overcome these weaknesses by proposing a hybrid retrieval model combining Graph Neural Networks and Knowledge Graph Embeddings. The aim is to enhance entity and relation representation through structured, context-sensitive learning processes to support improved and more scalable retrieval over complex medical data.

Research Objectives

The primary objective of this study is to create and implement an augmented Information Retrieval System for the healthcare industry by integrating Knowledge Graph Embeddings and Graph Neural Networks. The purpose is to improve the semantic understanding of entities and relations in medical Knowledge Graphs, thereby making the retrieval systems more precise, appropriate, and adaptive. Specifically, this study strives to:

- Design a hybrid model that combines GNNs with KGE to learn structural and semantic information from medical Knowledge Graphs.
- Test the model on a benchmark medical data to compare its performance in retrieving relevant information based on biomedical or clinical queries.

Thesis Structure

This dissertation is structured into three main chapters, each addressing a key aspect of the research project.

- **Chapter1 :**Overview about Medical Information Retrieval and Knowledge Graphs
  This chapter offers the theoretical background necessary to comprehend Knowledge Graphs and their application in medical Information Retrieval. It tackles Knowledge Graph Embeddings, their advantages in representing complex biomedical relations, and their limitations in retrieval contexts. The chapter then addresses how Graph Neural Networks, more particularly Relational Graph Convolutional Networks, can be utilized to enhance the expressiveness of KGs. In addition, a thorough examination of recent studies, their approach, datasets, limitations, and results is given comparison tables capture these works. The chapter concludes by outlining the primary gaps in current solutions, motivating the need for more scalable, accurate, and semantically aware systems in the healthcare industry.
- **Chapter2 :** Design of the Proposed Medical Information Retrieval System
  This chapter presents the proposed system architecture and components. The chapter describes building and enriching a medical Knowledge Graph from the PDD graph, entity conversion to Embeddings using RDF2Vec, and refinement using R-GCN. The chapter also describes preprocessing of user Queries, Embedding, and Matching with corresponding Knowledge Graph entities through Similarity Search Algorithms such as HNSW. Finally, the entire Retrieval and Ranking pipeline is explained.
- **Chapter3 :**TESTS AND EVALUATION OF THE PROPOSED SOLUTION
  This chapter introduces the technical implementation of the system and provides an in-depth analysis. It describes the environment used for development, utilized Software

Libraries, and the Hardware setup used. Then the detailed results of our experiments are presented together with a critical analysis. Finally, the chapter concludes with the presentation of our Knowledge Graph Querying System to evaluate the accuracy of the obtained results.

The thesis concludes with a General Conclusion that summarizes the contributions of this thesis and suggests avenues for future research to advance medical Information Retrieval with more profound graph learning techniques.

Chapter 1

Overview about Medical Information Retrieval and Knowledge Graphs

1.1 Introduction

This chapter provides an overview of traditional retrieval challenges, explains the principles of GNNs and knowledge Embeddings, and discusses their application in Medical Information Retrieval. Additionally, the integration of knowledge Embeddings into Graph Neural Networks is explored as a transformative approach to enhancing medical Information Retrieval. Traditional retrieval methods often struggle with the complexity, variability, and interconnected nature of medical data. Furthermore, by leveraging Knowledge Graphs and Embedding techniques, GNNs can effectively model relationships between medical concepts such as diseases, treatments, and research literature. Moreover, this allows for more accurate, context-aware retrieval, ultimately enhancing decision-making and providing valuable insights for healthcare professionals.

1.2 Information Retrieval

Before exploring advanced techniques, it is essential to understand the fundamental role of Information Retrieval in managing and accessing large data. IR enhances data accessibility and relevance, making efficient retrieval crucial in fields like healthcare, finance, and academia.

1.2.1 Origins of Information Retrieval

Information Retrieval is a branch of computer science that deals with document processing, containing free text, to enable their fast retrieval based on keywords specified in a user query. IR as a concept began in the 1950s, basically driven by the need to efficiently find useful information in large collections of documents. This was to be the beginning of serious scientific work in making information management and retrieval effective.

Over the last sixty years, this area has grown tremendously due to technological development and the exponential growth in information volume. This growth has brought about state-of-the-art retrieval methods and systems that inculcate probabilistic models, vector-space approaches such as the TF-IDF model, and machine learning algorithms [19].

1.2.2 Pipeline of Information Retrieval

Hereafter is a breakdown of the key components typically involved in an IR pipeline illustrated in figure 1.1 [1]:

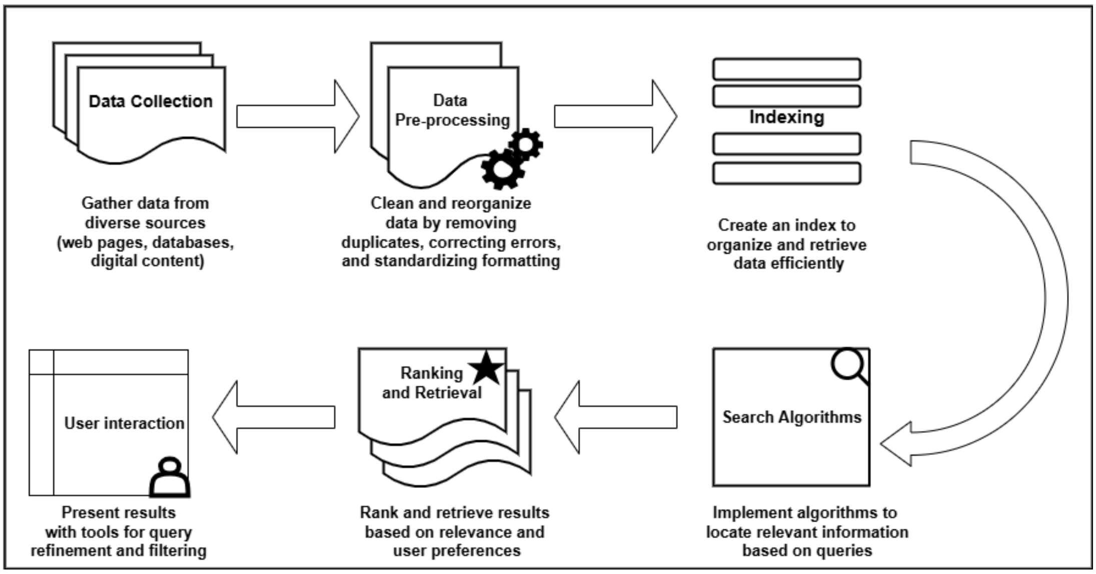


Figure 1: Pipeline of Information Retrieval

1.2.3 Classical Techniques of Information Retrieval

Classical Information Retrieval techniques focus on the effective extraction and retrieval of information from large collections of textual documents. Here are some key techniques and algorithms commonly associated with classical IR [14]:

- **Boolean Retrieval Model:** This model uses Boolean logic (AND, OR, NOT) to combine search terms. Documents are retrieved based on whether they satisfy the Boolean expression formed by the query. This method is straightforward but can be limited in handling complex queries.

- **Vector Space Model:** In this model, documents and queries are represented as vectors in a multidimensional space. The similarity between a document and a query is calculated using measures such as cosine similarity. This allows for ranking documents based on their relevance to the query.

- **Term Frequency-Inverse Document Frequency :** This is a statistical measure used to evaluate the importance of a word in a document relative to a collection (corpus). The TF component measures how frequently a term appears in a document, while the IDF component measures how important a term is across the entire corpus. The product of these two gives a weight that helps in ranking documents.

- **Probabilistic Models:** These models, such as the Binary Independence Model, estimate the probability that a document is relevant to a given query. They use statistical methods to predict relevance based on the presence or absence of terms in documents.

1.2.4 Challenges of Information Retrieval in the Medical Domain

Information Retrieval in the medical domain is an important field of research that aims at assisting clinicians and researchers in efficiently sifting through large volumes of medical data.

Medical IR systems should not only be able to retrieve relevant information but also guarantee that the results are accurate, contextually meaningful, and actionable considering the potential consequences on patient care and medical decision making. Some of the following points bring forth the challenges associated with IR in this field [10]:

- **Difficulty in Defining Research Questions:**

  One of the initial challenges is defining what the researcher wants to know. This step is crucial as it sets the direction for the entire information search process. Without a clear question, it becomes difficult to formulate an effective search strategy .

- **Choosing the Right Information Sources:**

  There is an overwhelming choice of information sources. The way in which information sources have been classified has changed and it is necessary to distinguish between conventional literature and gray literature for effective research.

- **Information Overloads:**

  With the exponential growth of digital content, IR systems must sort through vast amounts of data to find relevant results, often resulting in information fatigue for users.

- **User Behavior Variability:**

  Different users express their queries in unique ways, and their preferences may vary widely, making it difficult to provide uniformly satisfying results.

- **Time Constraints:** Health professionals are always bound by time constraints which may limit their ability to carry out exhaustive searches. This factor is best illustrated among medical students whose use of information technologies tends to wane in the final years due to pressures and time limits imposed by educators.

These challenges highlight the importance of developing effective strategies and skills for seeking health information, which can ultimately enhance both academic and clinical outcomes.

1.2.5 From Traditional Information Retrieval to Graph Neural Network-Based Information Retrieval

Traditional Information Retrieval systems rely on classic Neural Architectures, such as dense retrieval models and deep learning-based ranking mechanisms. While these approaches can improve search relevance, they often struggle to capture complex relationships between medical concepts. Graph-based models, particularly Graph Neural Networks, provide a powerful alternative by structuring information in a graph format, allowing for richer contextual understanding. By leveraging knowledge Embeddings, GNNs enhance retrieval by learning meaningful relationships between entities, improving ranking and recommendation in medical search systems.

1.3 Graph Neural Networks

Graph Neural Network is a deep learning architecture specifically designed to process graph-structured data. In contrast to the traditional neural networks, which typically process grid-like data (e.g., images), They are designed to process complex relationships and interactions in data and are therefore versatile enough for a wide range of applications. GNNs have been applied in fields ranging from social networks, molecular compounds, to computer vision, and have demonstrated their versatility and effectiveness in addressing real-world issues [24].

1.3.1 Types of Graph Neural Networks

Graph Neural Networks have developed into multiple forms, each of which aims to solve distinct challenges and uses in graph-structured data. Based on the analysis of several relevant research papers, Graph Neural Networks can be classified into distinct categories. Some of these categories are summarized through an analysis of the work described in figure 2[23].

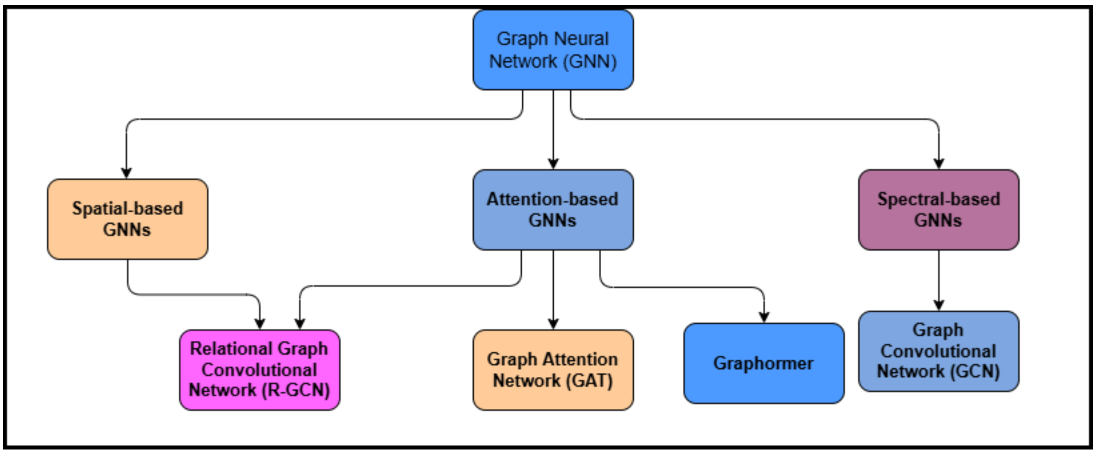


Figure 2: Types of GNNs

In the context of the work, Particular focus is placed on Graph Convolutional Networks and Relational Graph Convolutional Networks, their ability to handle graph-structured data effectively, making them highly relevant to the objectives of this study.

1.3.2 Graph Convolutional Networks

Graph Convolutional Networks are powerful tools for analyzing graph-structured data, effectively capturing relationships between nodes. By learning node representations through neighboring connections, GCNs improve tasks such as classification and recommendation, making them highly applicable in domains with complex, interconnected data [2].

1.3.2.1 Definition of Graph Convolutional Networks

Graph Convolutional Networks are a class of neural networks known as Graph Neural Networks that apply convolution operations on graph structures. Among the different approaches to extract features from graphs, GCNs are mainly divided into two categories: spectral domain-based models and spatial domain-based models [2].
Figure 3 illustrates an Example of Graph Convolutional Networks :

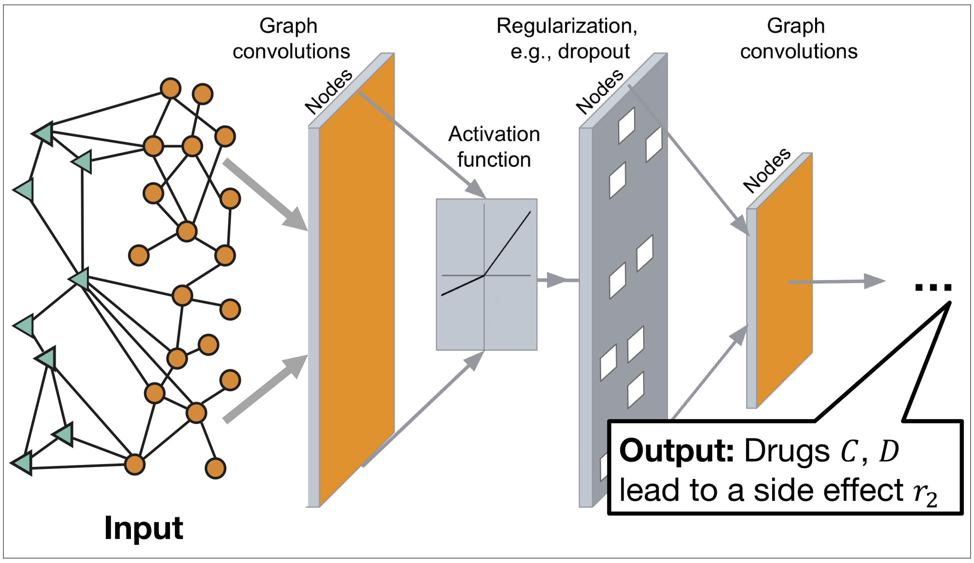


Figure 3: Example of Graph Convolutional Networks [30]

1.3.2.2 Architecture Of Graph Convolutional Networks

The architecture of Graph Convolutional Networks is structured to effectively learn from graph-structured data [29]. Figure 4 illustrates the architecture of the Graph Convolutional Network:

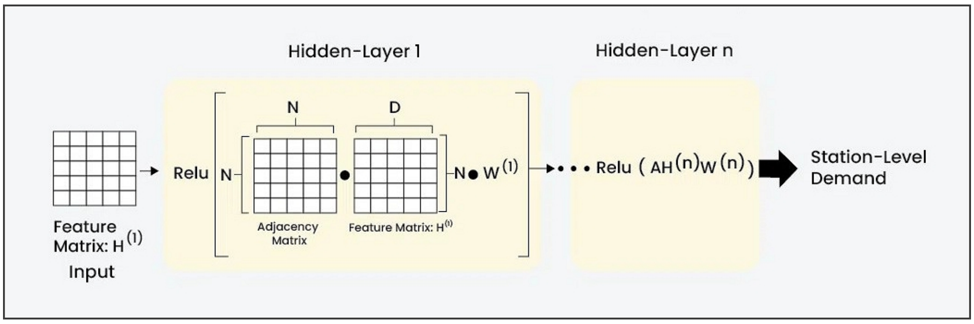


Figure 4: Architecture of Graph Convolutional Networks [28]

- **Input Representation :**

  The input to a GCN consists of a graph represented by its adjacency matrix and a feature matrix. The adjacency matrix captures the connections between nodes, while the feature matrix contains the attributes of each node.

- **Layer-Wise Propagation:** GCN employs a layer-wise propagation rule that aggregates information from neighboring nodes. Each layer updates the node representations by combining the features of a node with those of its neighbors.

- **Normalization of Adjacency Matrix:**

  The adjacency matrix is normalized to ensure that the node features are scaled appropriately. This normalization helps in stabilizing the learning process and allows the GCN to handle varying node degrees effectively.

- **Stacking Layers:** The GCN architecture consists of multiple layers stacked on top of each other. Each layer learns a new representation of the nodes by aggregating features from the previous layer. This stacking allows the model to capture higher-order relationships in the graph, enhancing its expressive power.

- **Output Layer:** The final layer of the GCN typically outputs class probabilities for each node, especially in semi-supervised learning tasks. This is achieved by applying a softmax function to the output of the last layer, which provides a probability distribution over the classes for each node.

- **Training Process:**

  The GCN is trained using a loss function that measures the difference between the predicted and actual labels of the nodes. The training process involves backpropagation through the layers, updating the weights to minimize the loss.

- **Scalability:** The architecture is designed to scale linearly with the number of edges in the graph, making it efficient for large datasets. This efficiency is crucial for real-world applications where graphs can be very large and complex.

1.3.3 Relational Graph Convolutional Network

Relational Graph Convolutional Networks are an extension of traditional Graph Convolutional Networks designed to manage heterogeneous graphs, where nodes and edges can have multiple types.

1.3.3.1 Definition of Relational Graph Convolutional Networks

Relational Graph Convolutional Networks are one of the specialized neural networks that is particularly applied to operate on multi-relational Knowledge Graph data in order to facilitate operations such as link prediction and entity classification. R-GCNs aggregate information between neighboring nodes based on relation-specific transformations, basically learning the complex relationships that exist in knowledge bases [21].

1.3.3.2 Architecture of Relational Graph Convolutional Networks

The architecture of Relational Graph Convolutional Networks is particularly optimized for handling complex graphs with multi-type and heterogeneous relations between entities. The R-GCNs are an improvement over the traditional Graph Convolutional Networks, as they employ mechanisms that can handle multiple relations in graphs efficiently [21]. The components and features of the R-GCN architecture are discussed below:

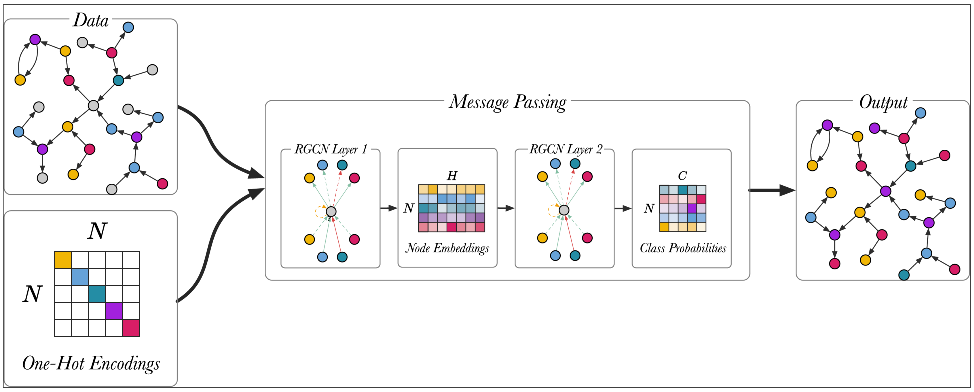


Figure 5: Architecture of Relational Graph Convolutional Networks [26]

1. **Input to R-GCN**
   The data fed into an R-GCN are:

   - **Graph Structure:** The graph is represented in terms of nodes (entities) and edges (relationships). A node represents an entity, and an edge represents a relationship between two entities. The relationships tend to be labeled, describing the type of connection between the nodes (e.g., "is_a", "located_in", etc.).

- **Node Features:** For each node, associated features (e.g., numeric, categorical data) specify the characteristics of the entity. These features act as input to the R-GCN, allowing the model to learn meaningful node embeddings.
- **Edge Types:** Edges connecting the nodes are not homogeneous but vary based on the different kinds of relationships between entities. These edge types are essential for enabling relation-specific transformations during message passing.

2. **Message Passing and Aggregation :**

   The underlying working principle of R-GCNs is the message-passing mechanism, through which the node representations are updated by collecting information from neighboring nodes depending on the relation type (edge type) between the nodes, Hereafter is how it works:

   - **Layer-Wise Updates:** R-GCN consists of multiple layers, and each layer updates node representations. In each layer, for each node, the network aggregates information from its neighboring nodes. The aggregated information is weighted based on the relation type between nodes. This allows the R-GCN to learn multi-hop, complex relations in the graph.
   - **Relation-Specific Transformations:** R-GCNs, unlike standard GCNs, use relation-specific transformations for aggregation. That is, every edge type (relationship) has its own set of parameters (weights) to update node features. This allows the R-GCN to capture the different types of interactions in the data accurately.
   - **Self-Connections:** Nodes may retain information from their previous state through self-connections (or identity mappings). This is necessary to prevent nodes losing their original information when they add new information from neighbors.
   - **Output:**
     - **Node Representations:** The result of an R-GCN are learned representations for each node (entity) in the graph. The representations contain the node is features based on its connectivity and the type of relationship with other nodes. The result for each node is a vector summarizing information about it in the graph structure context.
     - **Entity Classification:** For entity classification tasks, the R-GCN produces a probability distribution of possible labels for each node. It does this via a softmax on the final layer output, which converts the node representations to probabilities for each class label. The model is trained to minimize the cross-entropy loss between the predicted probabilities and actual node labels.
     - **Link Prediction Scores:** In link prediction, the R-GCN assigns scores to potential edges (relations) between nodes. The scores are the chance that a given

relationship exists between two entities. The R-GCN uses an encoder-decoder framework where the encoder generates node representations, and a scoring function (decoder) calculates these representations to predict novel links.

1.3.4 Comparison of R-GCN with GCN

The following table provides a concise comparison between GCN and R-GCN, highlighting their key differences in terms of architecture, scalability, and application domains [15, 21].

Aspect	**GCN**	**R-GCN**
Graph Type	Homogeneous (single edge type).	Heterogeneous (multi-relational).
Weight Matrix	Single matrix for all edges.	Multiple matrices, one per relation type.
Message Passing	Uniform aggregation .	Relation-specific aggregation .
Scalability	Efficient for homogeneous graphs .	Regularization improves scalability.
Application Domains	Social networks, citation graphs.	multi-relational learning, semantic representation.
Training Complexity	Lower (single matrix) .	Higher (multiple matrices, regularization).
Use Cases	Node classification, link prediction .	Entity classification, multi-relational link prediction .
Interpretability	Easier (single transformation) .	Complex (relation-specific transformations) .

Table 2: Comparison between GCN and R-GCN

As shown in the table above, while GCN is well-suited for simple, homogeneous graphs, R-GCN extends this capability to handle more complex, multi-relational structures, making it particularly effective for knowledge representation tasks in domains like medical information retrieval.

1.3.5 Inputs of Graph Neural Network

In this project, Knowledge Graphs (KGs) serve as a foundational data source from which essential inputs for Graph Neural Networks (GNNs) are derived—specifically, the node feature matrix and the adjacency matrix. KGs offer rich semantic information that enhances feature representation and defines meaningful relationships between nodes. This integration is crucial, as it provides contextual knowledge that helps structure the graph more effectively and improves the GNN's performance on tasks such as node classification and link prediction.

Utilizing Knowledge Graphs to build these input components greatly enhances the GNN's capacity to interpret and process graph-structured data, thereby contributing to improved results aligned with the goals of this project. To fully appreciate the role of Knowledge Graphs in this context, it is essential to first provide a clear overview of their definition and purpose.

1.3.6 Definition of Knowledge Graphs

Knowledge Graphs are powerful representations of structured information in artificial intelligence. Which capture a wide range of real-world facts. They consist of nodes, which represent entities such as people, places, and events, and edges that define the relationships between these entities. Figure 6 illustrates A toy example for Medical Knowledge Graph [11]:

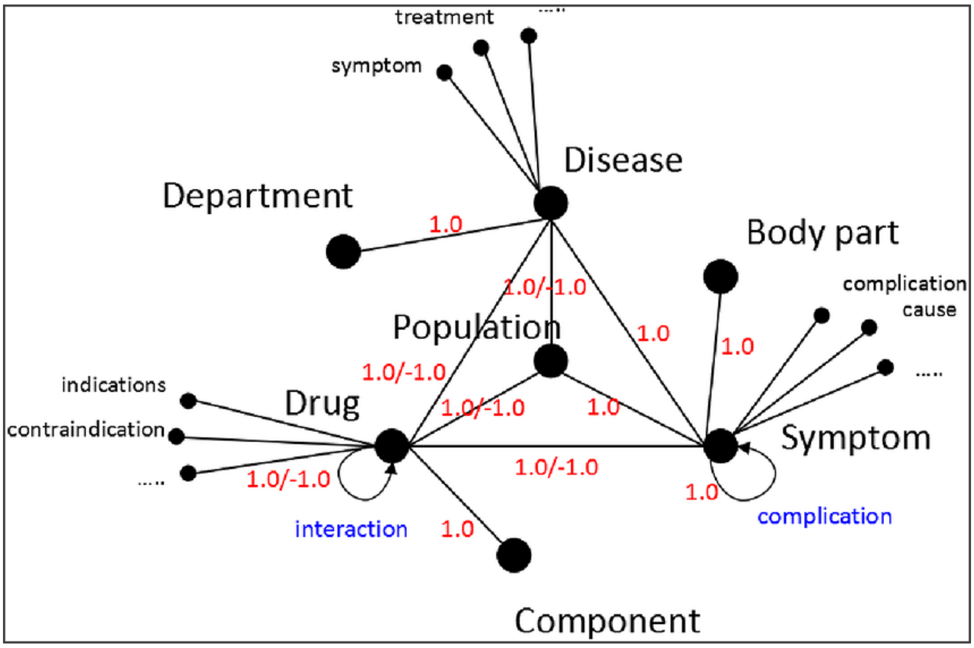


Figure 6: Medical Knowledge Graph [13]

1.3.7 History and Evolution of Knowledge Graphs

The evolution of Knowledge Graphs is a journey marked by significant advancements in artificial intelligence. This process has been summarized in the following figure [16]:

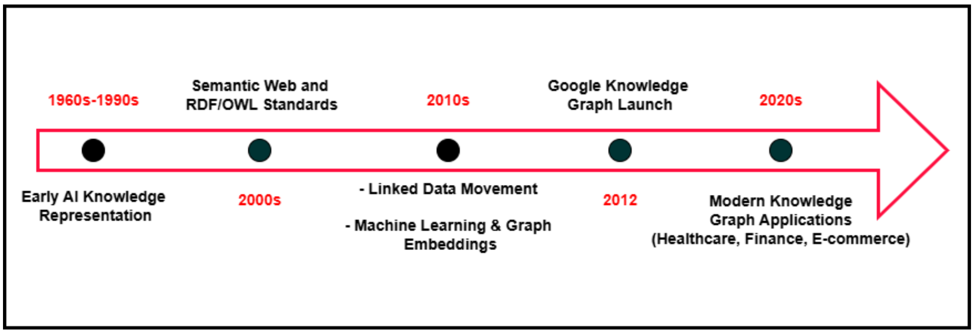


Figure 7: History and Evolution of Knowledge Graphs

1.3.8 Advantages of Knowledge Graphs

Knowledge Graphs offer significant advantages in various domains, particularly in healthcare, where managing and interpreting complex data is essential. By structuring information and uncovering relationships, they enable more precise recommendations, streamline information extraction, and enhance the interpretability of data, ultimately improving decision-making processes [7].

- **Improved Recommendations:**

Knowledge Graphs enhance the structuring of medical information, facilitating more accurate recommendations for medical departments based on patients' symptoms and medical history.

- **Automatic Information Extraction:**

Utilizing natural language processing techniques, Knowledge Graphs can automatically extract relevant information from medical documents, reducing the need for manual rule definition.

- **Interpretability:**

Knowledge Graphs illustrate the relationships between patients conditions, diseases, and departments, helping to understand the recommendations provided.

1.3.9 Transitioning to Knowledge Graph Embeddings

As the discussion is moved from Knowledge Graphs to Knowledge Graph Embeddings, it is important to note that KGE techniques are crucial for converting the discrete data in a KG into continuous vector representations. This step enhances the ability of GNN to process and learn from KGs more efficiently.

The transition between KGs and KGEs marks a shift from explicit graph-based structures to latent vector spaces, which can be more easily manipulated by machine learning algorithms, including GCNs and RGCNs.

1.3.10 Definition of Knowledge Graph Embeddings

Knowledge Graph embedding is a technique used to represent entities and relations from a Knowledge Graph in a continuous, low-dimensional vector space. This representation aims to preserve the structural and semantic information of the graph, facilitating various computational tasks [8].

1.3.11 Modeles of Knowledge Graph Embeddings

The document discusses various models of Knowledge Graph Embedding, which can be broadly classified into two main categories: translation-based models and semantic matching models [6].

- Translation-Based Models:

These models represent relationships as translations in the vector space. The most notable example is the *TransE* model, which uses vector addition to represent relationships between entities. This approach has been foundational in the development of KGE methods.
The figure 8 illustrates the working principle of the TransE model, in which the relationship between two entities is modeled as a translation operation in the vector space:

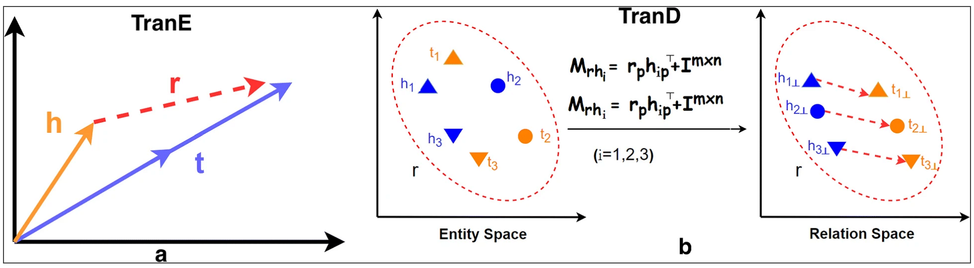


Figure 8: Example of Transalations-Based Models [4]

- Semantic Matching Models:

These models utilize multiplicative interactions to capture more complex relationships. They are designed to handle various types of relations, such as symmetric, anti-symmetric, and transitive relations. Examples include:

***ComplEx*:** This model extends the idea of embeddings to handle anti-symmetric relations by using complex-valued embeddings.

***DistMult*:** This model uses a bilinear product to represent relationships, focusing on symmetric relations. The figure 9 illustrates the DistMult model:

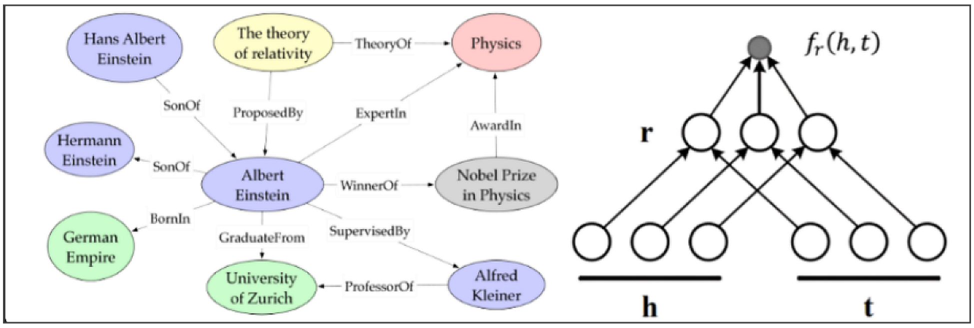


Figure 9: Example of Semantique Matching Models: DistMult [25]

- Random Walk-Based Models:

Random walk-based models like *RDF2Vec* work by generating multiple random walks from each node in a Knowledge Graph, treating these walks as sentences for training language models such as *Word2Vec*. Different walk strategies such as classic walks, entity-focused walks, and predicate focused walks are used to capture diverse structural and semantic relationships. These walks can also be biased using edge weighting to enhance their relevance. The resulting sequences are then encoded into embeddings, typically using frameworks like *RDF2Vec*, for use in downstream tasks [20].

1.4 Related Works

Graph Neural Networks are applied in medical Information Retrieval (MIR) to analyze complex data represented as graphs, where medical entities are connected through relationships. They enhance tasks such as treatment recommendations, disease-gene association prediction, and medical image analysis. By utilizing Knowledge Graphs, GNNs capture contextual dependencies, improving the accuracy of MIR systems. This approach opens up new opportunities for more intelligent, personalized clinical applications.

In this section, related works in the field of medical Information Retrieval using Graph Neural Networks are presented:

a) **Multilevel Classification of Users Needs in Chinese Online Medical and Health Communities:**
   This study [5] focuses on developing a multilevel classification framework to categorize users' needs in Chinese online medical and health communities, enhancing Information Retrieval accuracy and improving user experiences. Using a Graph Convolutional Network model, the framework combines manual coding of disease

types with clustering techniques to identify users information needs, enabling automated classification of medical queries. The findings highlight the importance of understanding diverse user needs across diseases to provide targeted and effective health services. The study also suggests that the framework can be applied to broader disease categories, contributing to better user satisfaction and health literacy in online medical platforms.

b) **MedGCN: Graph Convolutional Networks for Multiple Medical Tasks:**
This research [18] introduces MedGCN, an intelligent model designed to improve clinical practices by simultaneously addressing medication recommendation and lab test imputation. Using a specialized heterogeneous graph called MedGraph, the model captures complex relationships among medical entities like patients, encounters, lab tests, and medications. MedGCN extends traditional Graph Convolutional Networks to handle heterogeneous graphs and missing features, learning distributed representations from diverse medical data. A cross-regularization technique further enhances performance by leveraging losses from one task to improve the other, reducing overfitting. Experimental results show MedGCN outperforms state-of-the-art models in both tasks, demonstrating its effectiveness in handling complex and incomplete clinical data.

c) **ReHoGCNES-MDA: Prediction of MiRNA Disease Associations Using Homogenous Graph Convolutional Networks Based on Regular Graph with Random Edge Sampler:**
This study [3] introduces "ReHoGCNES-MDA," a novel model for predicting microRNA-disease associations using homogeneous Graph Convolutional Networks and a random edge sampler. By leveraging a regular graph structure, the model enhances connectivity and scalability, crucial for accurate predictions. The innovative ReHoGCNES model effectively aggregates neighborhood data while addressing challenges like neighbor explosion in GCNs. Experiments show the method outperforms traditional approaches in prediction accuracy. The model is ability to predict previously unknown miRNA-disease associations highlights its potential applications in drug discovery and understanding disease mechanisms. Future work aims to expand its utility in biomedical research and therapeutic development.

d) **Advancing Medical Imaging: Detecting Polypharmacy and Adverse Drug Effects with Graph Convolutional Networks:**
This study [9] addresses the critical issue of polypharmacy, where the use of multiple medications can lead to adverse drug reactions and interactions. It proposes a novel Graph Convolutional Network approach to predict and mitigate these effects by modeling drug-drug interactions in a medicine interaction graph. Nodes represent drugs, and edges capture interactions based on pharmacological properties. The research

integrates data from diverse sources, such as electronic health records and drug interaction databases, ensuring high-quality preprocessing to standardize and categorize medications. Future directions include incorporating genomics data for personalized predictions, developing explainable AI for interpretability, and exploring real-time implementation as a decision support system to improve patient care.

e) **KindMed: Knowledge-Induced Medicine Prescribing Network:**
This paper [17]proposes a novel paradigm for personalized drug suggestion through multi-source medical knowledge integration via Relational Graph Convolutional Networks. The model constructs a heterogeneous medical Knowledge Graph consisting of multiple clinical entities such as diseases, symptoms, and drugs along with their semantic and relational dependencies. By R-GCN, the model learns expressive rich embeddings capturing the complex interrelation among medical entities that enable accurate and personalized prescription prediction. Utilized with PyTorch Geometric, KindMed demonstrates the applicability of graph learning to clinical decision support. Although medication suggestion is the primary focus, the approach of using structured Knowledge Graphs and relational GNNs is especially applicable to a broad range of medical Information Retrieval tasks.

f) **Modeling Relational Data with Graph Convolutional Networks:**
This study [22] presents an elementary framework for reasoning over multi-relational Knowledge Graphs through the introduction of Relational Graph Convolutional Networks. The authors present a neural model that generalizes to enhance Graph Convolutional Networks with relation-specific transformations to learn directed, labeled edges in knowledge bases. Their methodology is benchmarked on two main tasks entity classification and link prediction and shown to surpass strong baselines, including factorization models like DistMult, especially on challenging datasets like FB15k-237. The R-GCN is combined with a decoder to form an encoder-decoder model that accumulates structural evidence over the course of a graph. Although the method was originally designed for wide-domain knowledge bases like Freebase and DBpedia, it is extremely relevant to domain-specialized retrieval systems in biomedicine, whose Knowledge Graphs are especially suffering from incompleteness. The emphasis on graph-structured learning and entity embeddings makes it directly applicable to medical Information Retrieval tasks that require rich semantic modeling and reasoning over complex relationships.

1.4.1 Comparison of the Related Works

Based on the reviewed works, a comparison was conducted and is summarized in the following tables. The first part of this comparison is presented in Table 3 according to the following criteria:

- **Objective:**

The primary goal or purpose of the research, such as improving medical decision making, predicting risks, or classifying user needs.

- **Graph Structure:**

The method used to organize and represent data in graph form, which could be heterogeneous (different types of entities) or homogeneous (similar entities), with specific types like regular graphs or multiview graphs used for different tasks.

- **Application:**

The real-world impact of the model, describing where and how the developed system can be applied, like in clinical environments, biomedical research, drug safety, or online health platforms.

Work	**Objective**	**Graph Structure**	**Application**
Multilevel Classification "Cheng, Q. and Lin, Y. (2023)"	User needs classification	Multi-level labeled graph	Online health platforms
MedGCN "Mao and al. (2022)"	Recommendations + imputation	Heterogeneous MedGraph	Clinical decision-making
ReHoGCNES-MDA "Bi and al. (2024)"	miRNA-disease prediction	Homogeneous regular graph	Biomedical research
Polypharmacy Risks "Dara and al. (2024)"	Adverse drug effect detection	Heterogeneous Drug Interaction Graph	Drug safety management
KindMed "Li and al. (2023)"	Medication recommendation	Heterogeneous medical Knowledge Graph (diseases, symptoms, drugs)	Personalized clinical decision support
Modeling Relational Data with R-GCN "Schlichtkrull and al. (2018b)"	Entity classification and link prediction	Multi-relational Knowledge Graph	Knowledge base completion, question answering, Information Retrieval

Table 3: Overview of Medical Knowledge Graph Approaches

The second part of the comparison is presented in Table 4 Comparison of some Medical Knowledge Graph Applications: Datasets, Challenges, Results, and Insights and focuses on the following criteria:

- **Dataset:**

A dataset is a collection of data used to test and evaluate an algorithm.

- **Challenges and Considerations:**

The obstacles, complexities, or important factors that need to be addressed or kept in mind during the design, implementation, or evaluation of a project, research, or system.

- **Results:**

The main outcomes or performance metrics derived from an experiment or study that highlight the effectiveness, accuracy, or impact of the proposed method, model, or system.

Work	**Dataset**	**Challenges and Considerations**	**Results**
1. Multilevel Classification "Cheng, Q. and Lin, Y. (2023)"	Qiuyi online medical health community from the Cardiovascular Disease.	Balancing precision and recall in classification; handling sparse and subjective data.	Multilevel classification: Accuracy = 0.6265, Precision = 0.6328, Recall = 0.5788, F1-Score = 0.5912.
2. MedGCN "Mao and al. (2022)"	Electronic Health Record data from a tertiary care hospital - MedGraph structure data.	Addressing heterogeneous data types and missing values in medical records; maintaining generalization for new data.	Medication recommendation: NMEDW dataset $\rightarrow$ LRAP: 0.7588 ± 0.0028,MAP@2: 0.7558 ± 0.0035 MIMIC-III dataset $\rightarrow$ LRAP: 0.8349 ± 0.0008, MAP@2: 0.8069 ± 0.0022. Lab test imputation: NMEDW dataset $\rightarrow$ MSE: 0.0229 ± 0.0025, MIMIC-III dataset $\rightarrow$ MSE: 0.0140 ± 0.0002.

3. ReHoGCNES-MDA "Bi and al. (2024)"	HMDD v2.0, HMDD v3.0, Training Set, Tp Test Set, Td Test Set, Tm Test Set, Tn Test Set.	Difficulty in generalizing miRNA-disease associations for novel cases. Risk of over-smoothing with complex graphs.	AUC: 0.9996 (Tp), 0.9985 (Td), 0.9985 (Tm), 0.9940 (Tn).
4. Polypharmacy Risks "Dara and al. (2024)"	Electronic Health Records, Pharmacovigilance Databases, Clinical Databases, Diverse Patient Datasets.	Complexity of drug interactions and biases in medical data. Requires handling of imbalanced datasets.	Medication recommendation: LRAP: 0.7374. Lab test imputation: MSE: 0.0229 (outperformed baseline methods).
5.KindMed: Knowledge-Induced Medicine Prescribing Network "Li and al. (2023)"	MIMIC-III, MIMIC-IV electronic health records	Integrating heterogeneous medical Knowledge Graphs (diagnoses, drug-drug interactions, semantic relations); capturing temporal dependencies across clinical visits; balancing precision and safety in medication recommendation to reduce adverse drug interactions.	Jaccard similarity: 0.619, F1-score: 0.764, Precision: 0.786, Recall: 0.745; reduced drug-drug interaction rate by 7.8%.
6.Modeling Relational Data with Graph Convolutional Networks(R-GCN):"Schlichtkrull and al. (2018b)"	AIFB, MUTAG, BGS, AM (entity classification); FB15k-237, WN18 (link prediction)	Parameter explosion with many relations; overfitting on rare relations; ineffective aggregation for high-degree nodes; limited decoder expressiveness.	Entity classification: AIFB = 95.83%, AM = 89.29%. Link prediction (FB15k-237): MRR = 0.248, Hits@10 = 0.414 (29.8% gain over DistMult).

Table 4: Comparison of Medical Knowledge Graph Applications

1.4.2 Analysis of the Related Works

Each of these studies adopts a particular approach, depending on the kind of data, processing models, and objectives pursued. The study **[5]** classifies user queries in online medical forums using a mix of manual coding, K-means clustering and GCNs to organize questions around disease types and information needs. In **[18]**, the authors developed an intelligent system with EHRs that will provide predicted laboratory test value imputation and medication recommendations, using for this a heterogeneous medical graph to strengthen the decision. Similarly,**[3]** proposed

a homogeneous graph model with regular structure for the prediction of associations between miRNAs and diseases-a very important topic in biomedical research and drug development. At the same time, the work **[9]** introduces drug interaction analysis, building a graph of interactions between drugs whereby nodes correspond to medicines and edges correspond to pharmacological interactions. This helps predict adverse drug reactions, although the prediction is highly dependent on the completeness of the data set.
Moreover, we can clearly note that bias in healthcare data sets limits generalizability. In fact, the paper **[17]**employs Relational Graph Convolutional Networks on a richly constructed heterogeneous medical Knowledge Graph that integrates patient EHR information with external medical ontologies and drug interaction networks. By extracting both semantic relations and temporal dynamics of clinical events, it is able to effectively enhance personalized medication advice. While domain-specific in the clinical domain, KindMed demonstrates how state-of-the-art GNN architectures can be combined with multi-source structured knowledge to enhance healthcare decision support systems. However, challenges remain in balancing the complexity of models and interpretability, as well as scalability in the process of managing large-scale real-world EHR datasets, The foundational work**[22]** introduces R-GCNs as a general-purpose model for relational knowledge bases, focusing on link prediction and entity classification through multi-relational message passing. Though developed for generic graphs like Freebase or DBpedia, the core idea of encoding relation-aware context proves highly transferable to medical domains, where structured semantic relationships are abundant. The study highlights challenges such as overfitting on sparse relations and limited decoder flexibility, which are also prevalent in biomedical graphs. This reinforces the relevance of adapting Relational GCNs for improving Information Retrieval and inference within complex medical datasets.

Together, these works point toward the prospect of GNNs in medical AI while being effective for knowledge extraction, prediction accuracy, and decision support. However, data quality, model scalability, and interpretability remain the key limitations.

The solution builds upon these approaches by combining knowledge Embeddings with GNNs to develop a more accurate, scalable, and interpretable framework for medical Information Retrieval and decision-making. By integrating structured and unstructured data while improving model interpretability, this approach aims to overcome data bias, improve generalizability, and ensure robust performance across various medical applications.

1.5 Conclusion

This chapter explored the principles and challenges of Information Retrieval, particularly in the medical field, where large and complex datasets create unique difficulties. Graph Neural Networks were examined as a powerful tool to enhance Information Retrieval, focusing on their architecture and the role of Knowledge Graphs and embeddings. Various studies applying GNNs to medical Information Retrieval were reviewed, highlighting the improved accuracy when combined with Knowledge Graphs. The chapter sets the stage for the proposed solution, which aims to enhance efficiency in Medical Information Retrieval by incorporating dynamic embeddings and multitask learning frameworks.

Chapter 2

Design of the Proposed Medical Information Retrieval System

2.1 Introduction

The improvement of medical information retrieval relies on optimizing the process of retrieving relevant data from a vast knowledge base. In this context, an innovative architecture is proposed, leveraging knowledge Embeddings and Relational Graph Convolutional Networks to enhance information representation and optimize the accuracy of retrieved results. This approach enables better capture of complex relationships between medical entities and leverages the richness of Knowledge Graphs. By combining these techniques, the system is designed to provide a more efficient and relevant search system for users.

2.2 Features of the System

The system is designed to enhance medical Information Retrieval by integrating advanced techniques in knowledge representation. The following are the key characteristics:

a) **Knowledge Graph-Based Representation:**
The system uses a Knowledge Graph to build and organize medical data, providing storage of energy homage data in a rich and interconnected data landscape.

b) **knowledge Embeddings:**
By transforming graph data into vector representations, the approach captures complex relationships between medical entities, improving retrieval accuracy.

c) **Relational Graph Convolutional Networks :**
These networks enable efficient propagation of information through the graph, refine the semantic understanding of queries and documents, and enhance the input embeddings.

d) **Efficient Query Processing :**
User queries are converted to embeddings, allowing meaningful comparisons with indexed medical knowledge.

e) **Ranking and Retrieval Mechanism :**
The system ranks results based on semantic similarity, ensuring that the most relevant medical information is presented to the user.

2.3 Global Schema of the System

This approach is based on a sequential end-to-end pipeline to improve Medical Information Retrieval using Knowledge Graphs, knowledge Embeddings and Graph Convolutional Networks. The figure 10 presents the global architecture of the proposed system:

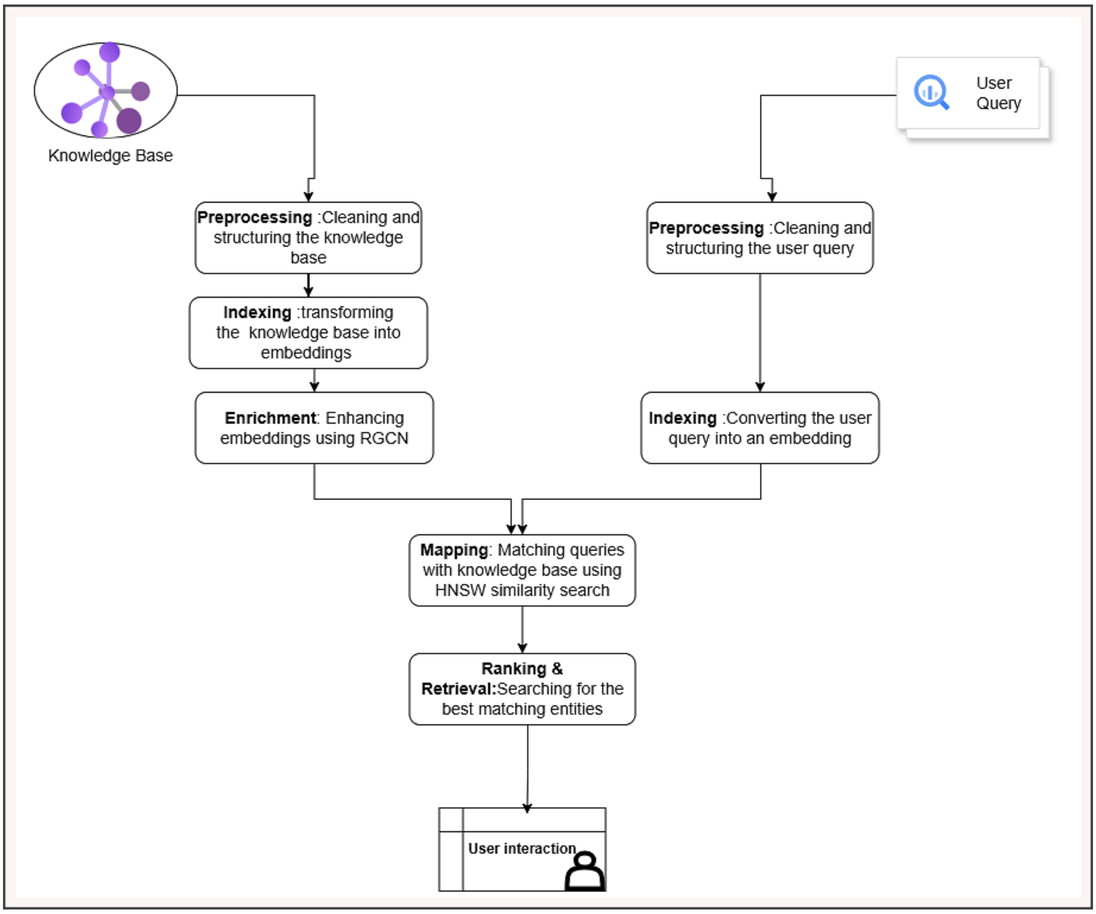


Figure 10: Global Schema of the System

The architecture is divided into two main components:

- Knowledge Base Processing (Prepares Medical knowledge for Retrieval).
- Query Processing (Handles User Queries and Ranks Relevant Results).

The details of each step within these components will be discussed in the subsequent parts of this chapter.

2.4 The Knowledge Base

In the context of healthcare, it is necessary to reflect accurately the complex relationships between patients, diseases, and drugs to make optimal clinical decisions. The goal is well addressed by the PDD Graph, which is designed to transform Electronic Medical Records into a heterogeneous Knowledge Graph dense with information. The graph models the advanced relations between patients, their diagnoses, and the medication prescribed to them. Built from data in MIMIC-III, the PDD Graph detects significant medical entities and combines them with existing Biomedical Knowledge Bases, such as the ICD-9 ontology and DrugBank, in a seamless manner to produce a high-quality, structured representation of medical knowledge .The figure 11 provides an example subgraph extracted from the PDD Graph, demonstrating the structured representation of relationships among patients, diagnoses, and prescribed drugs within the medical knowledge base [27]:

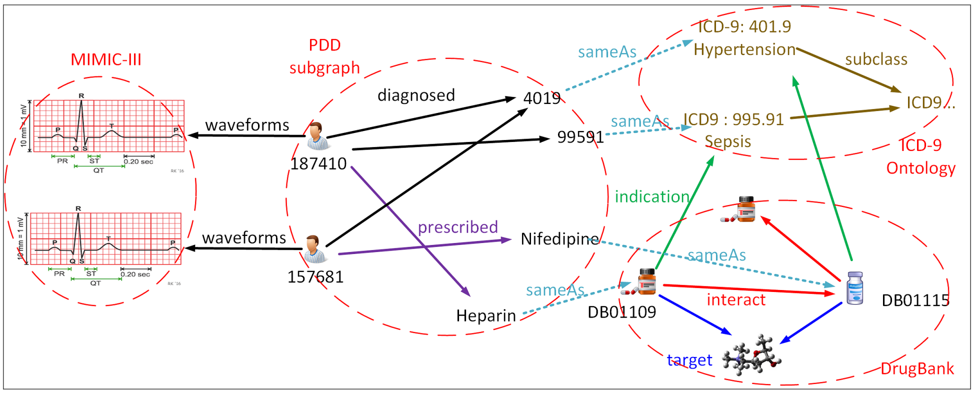


Figure 11: Subgraph of PDD [27]

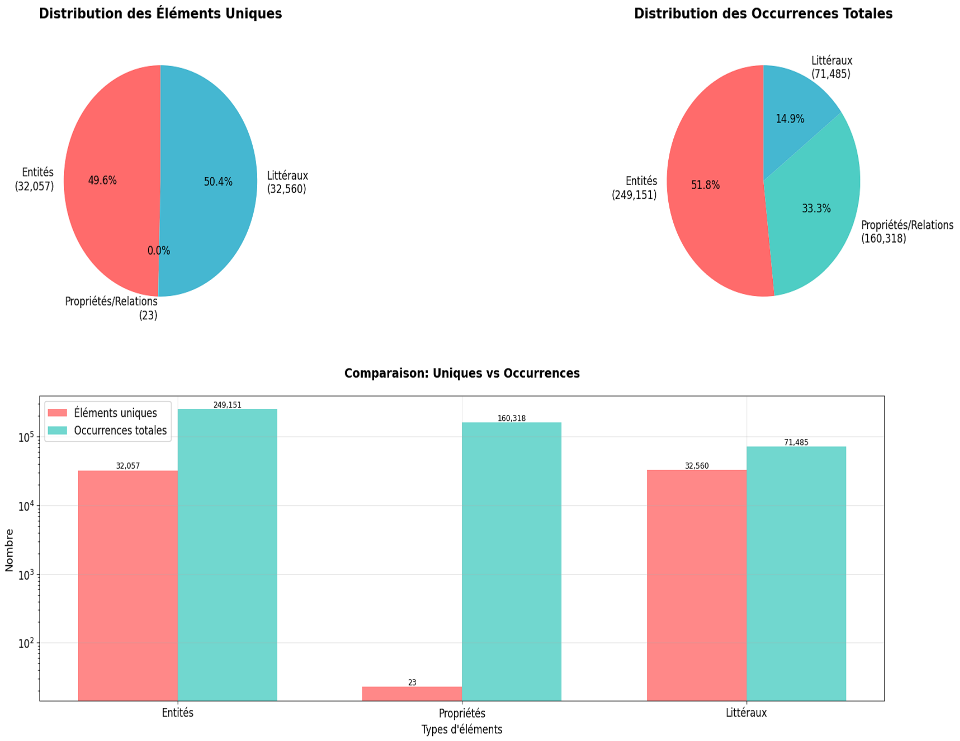
The data is structured as RDF triples of the form (subject, predicate, object). Each triple represents a relationship between two medical entities. The description of the predicates in the Knowledge Base is presented in Table 5.

Predicate	**Description**
patient-id	Unique identifier for each patient
bmi-max	Highest BMI recorded during admission
bmi-min	Lowest BMI recorded during admission
bmi-first	Patient's BMI at admission
interact	Drug-drug interaction relation
hospital-admission-id	Unique identifier for a hospital encounter
diagnoses-icd9	ICD-9 disease code assigned to the patient
age	Age of the patient
gender	Gender of the patient (e.g., M, F)
take-drugbank-id	DrugBank identifier of the prescribed drug
take-drug-name	Name of the drug prescribed to the patient

drug-type	Type or category of the prescribed drug
duration-days	Duration (in days) for which the drug was prescribed
start-date	Date on which the prescription started
end-date	Date on which the prescription ended
dose	Prescribed dosage of the drug
prescription-id	Unique identifier for each prescription
has-prescription	Indicates that a patient has an associated prescription

Table 5: Description of the Predicates in PDD knowledge Base

Figure 12 presents the comprehensive statistical analysis of the knowledge Base composition, revealing the distribution between unique elements and their total occurrences, where entities dominate both the unique count (32,057) and total occurrences (249,187), followed by literals and properties/relations.



Statistiques Détaillées du Knowledge Graph

Type	Éléments Uniques	Occurrences Totales
Entités uniques	32,057	249,151
Propriétés/Relations uniques	23	160,318
Littéraux uniques	32,560	71,485
TOTAL	64,640	160,318 triplets

Figure 12: Knowledge Base Statistics

2.5 Motivation for Choosing the PDD Knowledge Base

The PDD Graph Knowledge Base was selected for this project because it is a dense real-world Medical Knowledge Graph that is highly appropriate for Information Retrieval applications. It expresses valuable relationships between patients, diseases, and drugs, and is encoded in RDF, supporting semantic querying and integration. It is graphical nature also sets it up for the straightforward application of Knowledge Graph Embeddings and Graph Neural Networks in order to learn meaningful representations of medical entities. This works directly towards allowing the intelligent retrieval system to match patient profiles with applicable medical knowledge more effectively

2.6 Description of the Proposed Solution

2.6.1 Knowledge Graph Preprocessing

a) **Loading RDF Triples** RDF data provides a standard pattern for modeling structured information on the web. It allows data to be expressed in the form of subject-predicate-object triples, and then forms a linked data graph.

The flexible form graph model provides an efficient way of associating data coming from different sources.

At this phase, triples were extracted from the PDD graph. First, The process began by loading the Knowledge Base. followed by traversing the Knowledge base in order to identify and obtain all the triples present. Figure 13 shows a few examples of these triples. The first column represents the subjects, while the second column refers to the predicates, and the third column represents the objects.

```
Property                                                  Resource                                  Value
--------------------------------------------------------------------------------------------------------------------------
<http://pdd.wangmengsd.com/property/hospital_admission_id> <http://pdd.wangmengsd.com/resource/100001>  100001
<http://pdd.wangmengsd.com/property/take_drug_name>        <http://pdd.wangmengsd.com/resource/100001>  Aluminum-Magnesium Hydroxide-Simethicone
<http://pdd.wangmengsd.com/property/take_drugbank_id>      <http://pdd.wangmengsd.com/resource/100001>  <http://bio2rdf.org/drugbank:DB00030>
<http://pdd.wangmengsd.com/property/age>                   <http://pdd.wangmengsd.com/resource/100001>  35^^xsd:integer
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>          <http://pdd.wangmengsd.com/resource/100001>  <http://pdd.wangmengsd.com/vocabulary/Adm
<http://pdd.wangmengsd.com/property/bmi_max>               <http://pdd.wangmengsd.com/resource/100009>  34.38^^xsd:float
<http://pdd.wangmengsd.com/property/interact>              <http://bio2rdf.org/drugbank:DB00030>        <http://bio2rdf.org/drugbank:DB00104>
<http://pdd.wangmengsd.com/property/bmi_min>               <http://pdd.wangmengsd.com/resource/100009>  34.38^^xsd:float
<http://pdd.wangmengsd.com/property/bmi_first>             <http://pdd.wangmengsd.com/resource/100009>  34.38^^xsd:float
<http://pdd.wangmengsd.com/property/gender>                <http://pdd.wangmengsd.com/resource/100001>  F
```

Figure 13: Rdf Data Extract From PDD

b) **Elimination of Triples with Empty Elements**

As a preliminary step, triples containing empty or null elements are identified and removed. This includes triples where the subject, predicate, or object is None, an empty string, or contains only whitespace characters. This validation ensures that all remaining triples have complete structural integrity before further processing.

The following algorithm outlines the step-by-step process of eliminating triples with empty elements:

Algorithm 1 Algorithm for Eliminating Triples with Empty Elements

Input: raw_triples: list of raw RDF triples
Output: non_empty_triples: list of triples without empty elements
begin
    non_empty_triples ← []

    for *each triple (s, p, o) in raw_triples* **do**
        if *IsNotEmpty(s) AND IsNotEmpty(p) AND IsNotEmpty(o)* **then**
            AddToList(non_empty_triples, (s, p, o))
        end
    end
    return non_empty_triples
end

c) **Cleaning Data by Eliminating Malformed Triples**

In this step, a cleaning process is conducted to identify and remove malformed triples. These are triples that do not conform to the expected RDF structure, such as those with incorrect data types, or invalid URIs. Cleaning the Knowledge Base at this stage helps prevent errors and ensures reliable results in subsequent processing steps. The algorithm proposed for this purpose is presented below:

Algorithm 2 RDF-Malformed Triples Removal

```
Input: valid_triples: list of complete RDF triples
Output: well_formed_triples: list of RDF-compliant triples
begin
    well_formed_triples ← []
    for each triple in valid_triples do
        (s, p, o) ← triple
        if IsLiteral(s) then
            // Subject cannot be literal in RDF
            Continue
        end
        AddToList(well_formed_triples, triple)
    end
    return well_formed_triples
end
```

d) **Removal of Duplicate Triples**

In addition, duplicate triples that are not relevant to the analysis are removed to reduce redundancy and enhance the efficiency of the system. This step is carried out using the following algorithm:

Algorithm 3 Duplicate Triples Removal

```
Input: well_formed_triples: list of RDF-compliant triples
Output: unique_triples: list of unique RDF triples
begin
    unique_triples ← RemoveDuplicates(well_formed_triples)
    return unique_triples
end
```

After completing the preprocessing steps including the elimination of Empty and Incomplete Triples, removal of malformed and duplicate triples the Knowledge Graph has been cleaned and structured for further analysis.

Figure 14 presents an illustrative excerpt of the preprocessed Knowledge Graph, showcasing a subset of DrugBank entity IDs and their interactions:

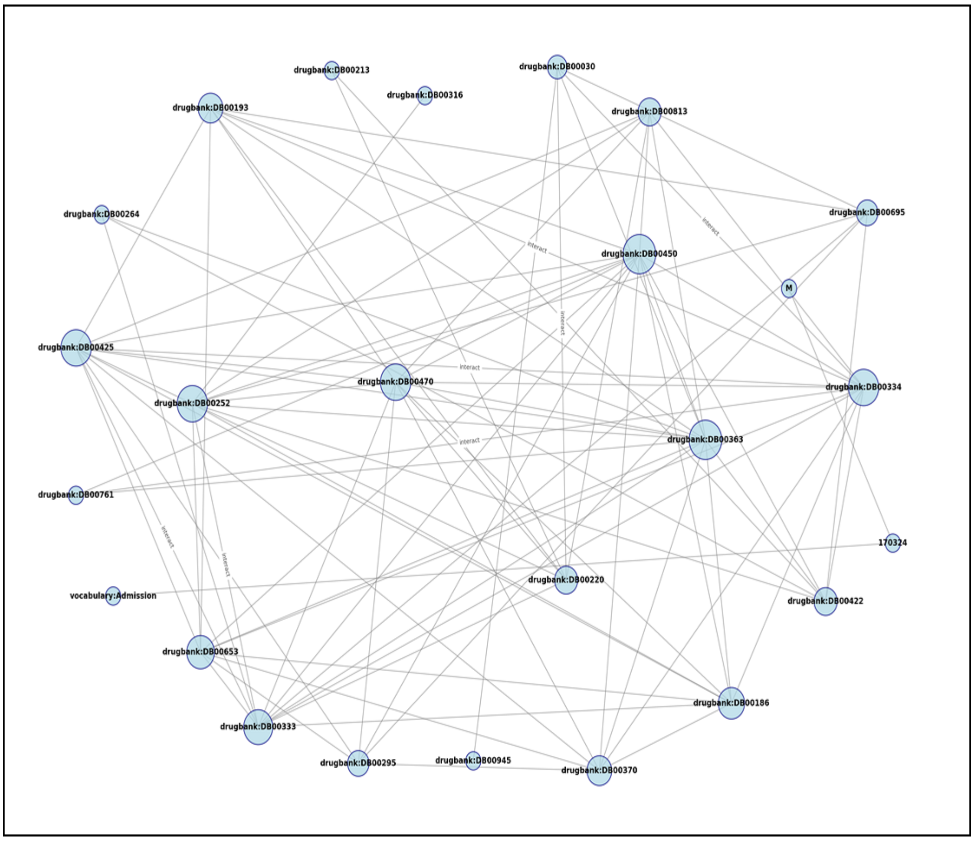


Figure 14: Sample Preprocessed Knowledge Graph

2.6.2 Transforming the Knowledge Base into Embeddings

This approach leverages the RDF2Vec embedding model to transform the Knowledge Graph into dense vector representations that can be exploited by deep learning models.The figure 15 presents the RDF2Vec pipeline:

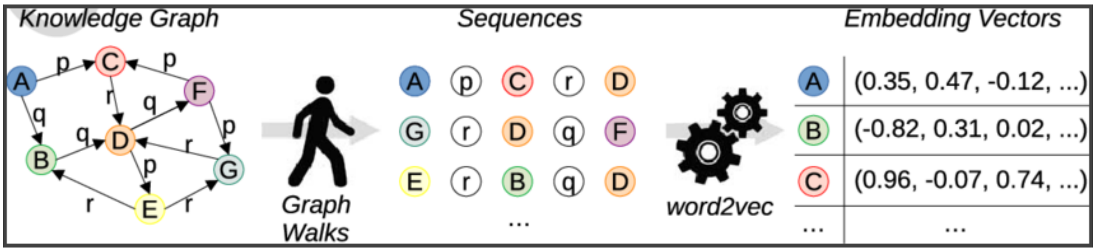


Figure 15: Workflow of RDF2Vec [20]

- The RDF triples are initially transformed into a directed graph, where nodes represent entities and edges are labeled with the corresponding relations.
- Random walks are then performed starting from each entity in this graph representation. At each step of a walk, a neighboring node is randomly selected, and the relation connecting the current entity to the next is recorded. These walks generate sequences that resemble sentences, alternating between entities and relations.

- Once a sufficiently large number of random walks have been collected, a Word2Vec model is trained on the resulting sequences to learn vector representations of entities and relations.

  Word2Vec is a two-layer neural network with a single hidden layer, trained to learn word representations from the context of words in sentences. In this case, entities and relations are treated as words, and random walks as sentences. The model learns embeddings by predicting the context tokens (surrounding) given a target token, in the style of Skip-gram. The intuition is that entities that appear in similar structural contexts in the graph will have similar embeddings. This training process allows Word2Vec to learn both topological proximity and semantic similarity between entities.

As a result, all entities are embedded into a dense vector space with similar entities mapped closer to each other. These embeddings are then used as input features for further Graph Neural Network training, Figure 17 illustrates examples of these embeddings.

```
some entity embeddings generated by (RDF2vec)
1040: [-0.01763148 -0.02975791 -0.01378354 -0.00444753 -0.00490284]... (dim: 128)
137280: [-0.03916646 -0.04862752 -0.02127676  0.00188328 -0.00785878]... (dim: 128)
24.42: [-0.16504018 -0.11829975 -0.03126879 -0.0266816  -0.01816801]... (dim: 128)
178521: [-0.03818869 -0.04342987 -0.01273129  0.00038949 -0.0074637 ]... (dim: 128)
6596: [-0.02616327 -0.03436745 -0.00371496  0.00868658 -0.01633453]... (dim: 128)
```

Figure 16: Examples of Entity Embeddings Generated with RDF2Vec

2.6.3 Enrichment of Embeddings Using R-GCN

2.6.3.1 Preparing Input Data for RGCN

At this stage, RDF2Vec embeddings have been generated for each entity. The graph structure required by the R-GCN is then prepared:

- **Node Features**: Each node is initialized with its precomputed RDF2Vec embedding as a feature vector (data.x). These embeddings are organized into a torch. tensor suitable for input to the R-GCN model.

- **Entity Indexing (entity-to-idx)**: PyTorch Geometric (torch-geometric) needs nodes to be referred to by numbers (0, 1, 2, 3...) So each RDF entity (which is a text like "http://pdd.wangmengsd.com/resource/12345") is mapped to a unique integer ID.
  The following table shows an example of entity indexing, where each RDF entity is assigned a unique integer identifier:

RDF entity	Index
http://pdd.wangmengsd.com/resource/12345	0
http://pdd.wangmengsd.com/resource/67890	1
http://pdd.wangmengsd.com/resource/24680	2
http://pdd.wangmengsd.com/resource/13579	3

- **Relation Indexing (Relation-To-Idx)**: Each relation (predicate) also must be represented by an integer ID, because RGCNConv expects edges to carry a type (relation type).
  The following table presents an example of relation indexing, where each predicate is assigned a unique integer ID to represent relation types required by the RGCNConv layer

Relation (predicate)	Index
Gender	0
Age	1
Name	2
takeDrug	3

- **Build Edge-Index And Edge-Type**:

  edge-index = list of pairs (source-id, target-id) based on the triples.

  edge-type = for each pair, what relation connects them.

$$\text{edge\_index} = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 1 & 2 & 3 & 1 \end{bmatrix} \tag{1}$$

$$\text{edge\_type} = \begin{bmatrix} 2 & 3 & 0 & 1 \end{bmatrix} \tag{2}$$

2.6.3.2 Configuration of the R-GCN Model

A Relational Graph Convolutional Network is constructed to further refine the RDF2Vec embeddings.

- **Layer 1**: A RGCNConv layer aggregates neighbor information depending on relation types.

- **Attention Layer**: Apply a multi-head attention mechanism to dynamically weigh neighbors.

- **Layer 2**: Another RGCNConv layer to deepen feature extraction across the graph.

- **Final Linear Layer**: A Linear layer projects the output embeddings back to the RDF2Vec embedding size.

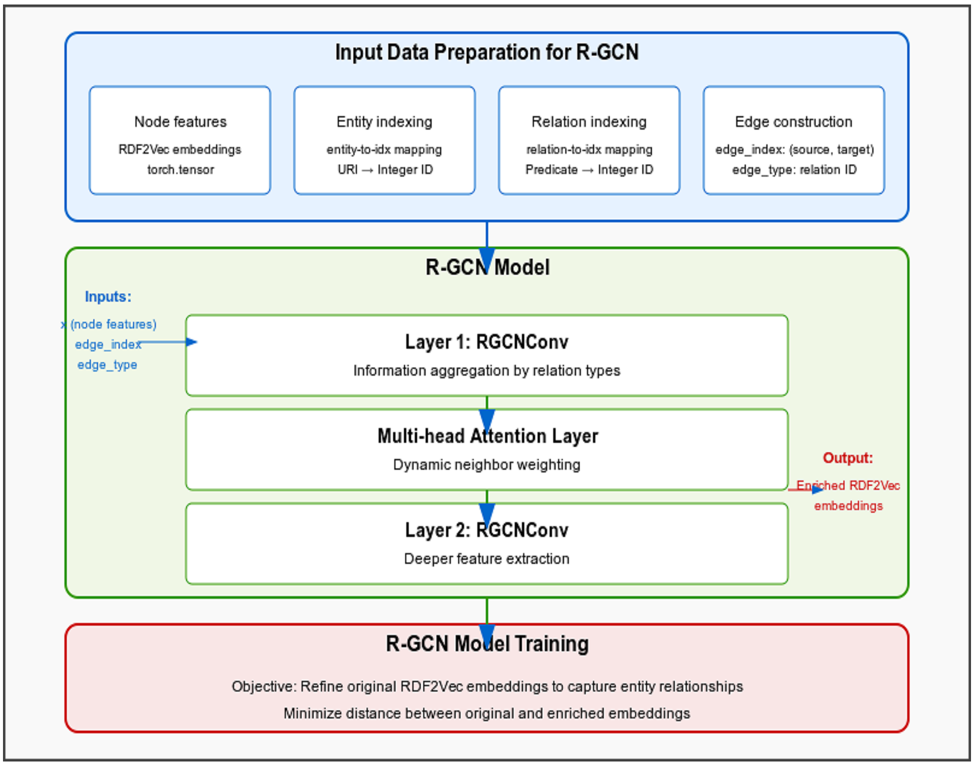


Figure 17: R-GCN Enrichment Architecture for RDF2Vec Embeddings

2.6.3.3 Training the RGCN Model

After preparing the graph data and defining the RGCN model, the next step is to train the model in order to refine the initial RDF2Vec embeddings. The goal of this training is to make the RGCN learn embeddings that are closer to the original RDF2Vec embeddings, A simplified overview of the training procedure is presented below:

Algorithm 4 Training an RGCN with RDF2Vec Initial Embeddings

Input: Graph data $(x, edge\_index, edge\_type)$,
RGCN model (initialized),
RDF2Vec initial embeddings (target),
Parameters: Optimizer (Adam, lr=0.01), Loss (MSE), Epochs (N_{epochs})
Output: Trained RGCN model, Updated node embeddings
begin
  1. Initialize the RGCN model in training mode **for** *epoch* $= 1$ *to* N_{epochs} **do**
    2. Forward and Backward Pass:
      a. optimizer.zero_grad() `// Clear previous gradients`
    b. output ← RGCN(data) `// Predict node embeddings`
    c. loss ← MSE(output, x) `// Compute reconstruction loss`
    d. loss.backward() `// Backpropagation`
    e. optimizer.step() `// Update model weights`
    3. Monitor Training:
    if *epoch % 5 == 0* **then**
      Print "Epoch:", epoch, "Loss:", loss.item()
    end
  end
  4. Finalize:
  a. Set the model to evaluation mode
  b. Compute final embeddings:
  updated_embeddings ← RGCN(data)
  5. Return trained RGCN model, updated_embeddings.
end

2.6.4 User Query Preprocessing

This subsection explains the preprocessing pipeline used to handle natural language questions in the medical scenario. The approach translates unstructured user queries into structured, semantically annotated query representations. The pipeline represented in Figure 18, ensures that queries are appropriately processed for proper Information Retrieval.The figure 18 illustrates the overall architecture of the user query preprocessing pipeline:

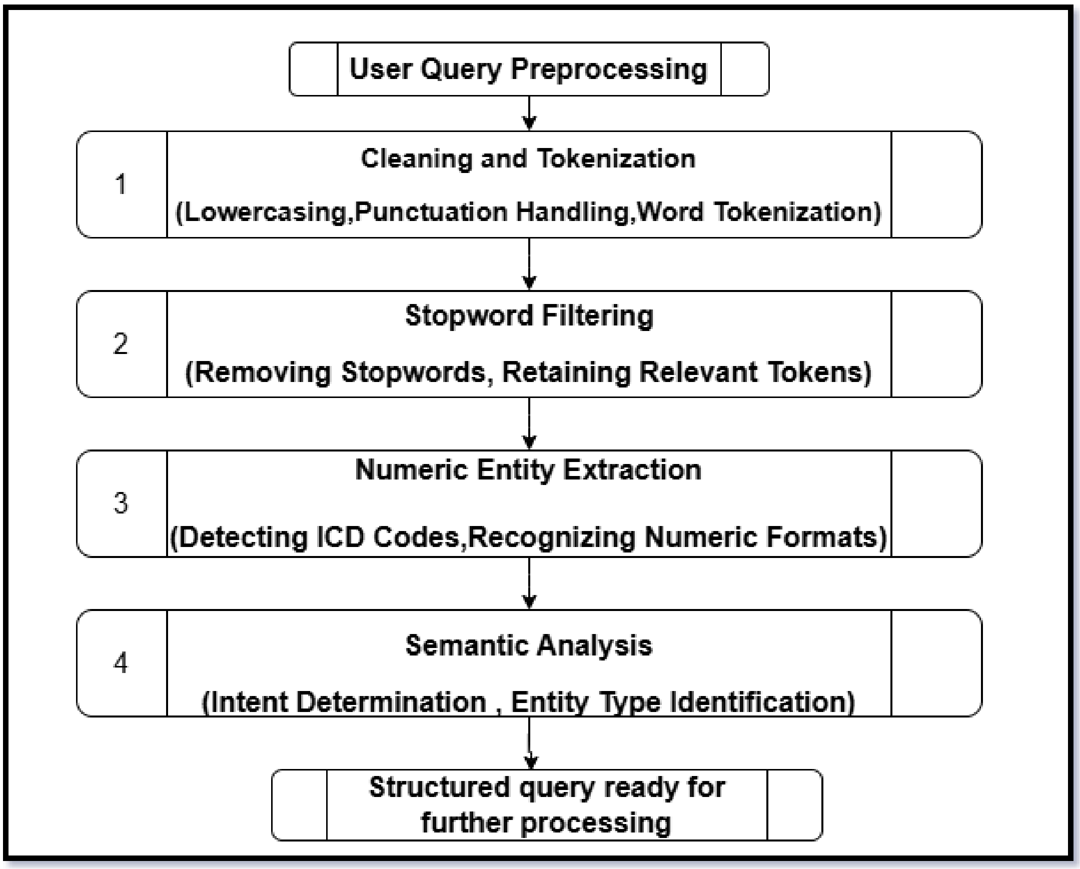


Figure 18: General Architecture of User Query Processing

Each stage in this pipeline transforms the query to progressively extract more structured information.

a) **Cleaning and Tokenization**

To begin, the user query is first cleaned and tokenized. This step is essential for ensuring that the text is in a standard format for further processing. Algorithm 5 presents the implementation of this step:

Algorithm 5 Cleaning and Tokenization(query)

Input: query: Raw user query (string)
Output: all_tokens: List of cleaned and tokenized words
begin
  clean_query ←**LOWERCASE(query)**
  clean_query←**REMOVE_NON_DECIMAL_PERIODS(clean_query)**
  clean_query←**REMOVE_PUNCTUATION(clean_query)**
  clean_query←**NORMALIZE_SPACES(clean_query)**
  all_tokens ←**TOKENIZE(clean_query)**
  return all_tokens
end

---

example:

- Input Query: "Does METHIMAZOLE interact with ROTAVIRUS vaccine ? "
- Lowercased: "does methimazole interact with rotavirus vaccine ?"
- Punctuation Removed: "does methimazole interact with rotavirus vaccine"
- Tokenized: ['does', 'methimazole', 'interact', 'with', 'rotavirus', 'vaccine']

b) **Stopword Filtering** In this step, common words that do not carry significant meaning (stopwords) are removed. This helps focus on the important terms of the query. The algorithm below provides a detailed overview of this step:

Algorithm 6 Filter Meaningful Tokens

Input: all_tokens: List of all tokens
Output: tokens: Dictionary containing all and meaningful tokens

```
begin
    meaningful_tokens ← []
    foreach token in all_tokens do
        if token not in stop_words then
            APPEND token to meaningful_tokens
        end
    end
    tokens ← {
     "all_tokens": all_tokens,
     "meaningful_tokens": meaningful_tokens
     }
    return tokens
end
```

example:

- Meaningful Tokens: ['methimazole', 'interact', 'rotavirus', 'vaccine']

c) **Numeric Entity Extraction** This step focuses on identifying and extracting numeric entities such as medical codes or numeric values that may be relevant. Algorithm 7 summarizes the overall process:

Algorithm 7 Extraction of Numeric Entities

Input: all_tokens: List of tokens
Output: numeric_entities: List of numeric entities

```
begin
    numeric_entities ← [ ]
    for each token in all_tokens do
        if token matches REGEX('\d+(\.\d+)?') then
            Add token to numeric_entities
        end
    end
end
```

example:

- Detected Numeric Entity: None.
- Medical Code Detection: None.

d) **Semantic Analysis** Here, the query is analyzed for its intent and the types of entities it contains. This step helps to understand what the user is looking for. Algorithm 8 presents the implementation of this step:

Algorithm 8 Intent Detection and Entity Identification

Input: all_tokens: List of tokens, semantic_keywords: Dictionary of intent and entity keywords
Output: top_intent: Detected intent, mentioned_entities: Identified entities

```
begin
    intent_scores ← {intent: 0 for each intent in semantic_keywords["intent"]}
    for each token in all_tokens do
        for each (intent, keywords) in semantic_keywords["intent"] do
            if token is in keywords then
                intent_scores[intent] ← intent_scores[intent] + 1
            end
        end
    end
    top_intent ← MAX_KEY(intent_scores)
    if intent_scores[top_intent] = 0 then
        top_intent ← "query" // Default intent if no match
    end
    entity_scores ← {entity: 0 for each entity in semantic_keywords["entities"]}
    for each token in all_tokens do
        for each (entity, keywords) in semantic_keywords["entities"] do
            if token is in keywords then
                entity_scores[entity] ← entity_scores[entity] + 1
            end
        end
    end
    mentioned_entities ← {entity: score for (entity, score) in entity_scores if score > 0}
end
```

example:

- Intent Determination: Interaction
- Entity Types Identified: ['vaccine']

2.6.5 Indexing the User Query

This solution is inspired by methods proposed in the work of Ebeid (2022), presented in the MedGraph framework for biomedical Information Retrieval. Here is a detailed breakdown of this solution[12]:

- The core of the entity lookup strategy is to find matching query entities with their counterparts in the embedding dictionary of the Knowledge Graph.

- Upon finding potential entity matches, their embeddings are retrieved from the embedding dictionary.

- The retrieved embeddings are stored in a results dictionary that maintains the entity identifier-to-vector representation mapping.
  The figure 19 outlines the process of indexing a user query by matching entities to their corresponding embeddings within the Knowledge Graph:

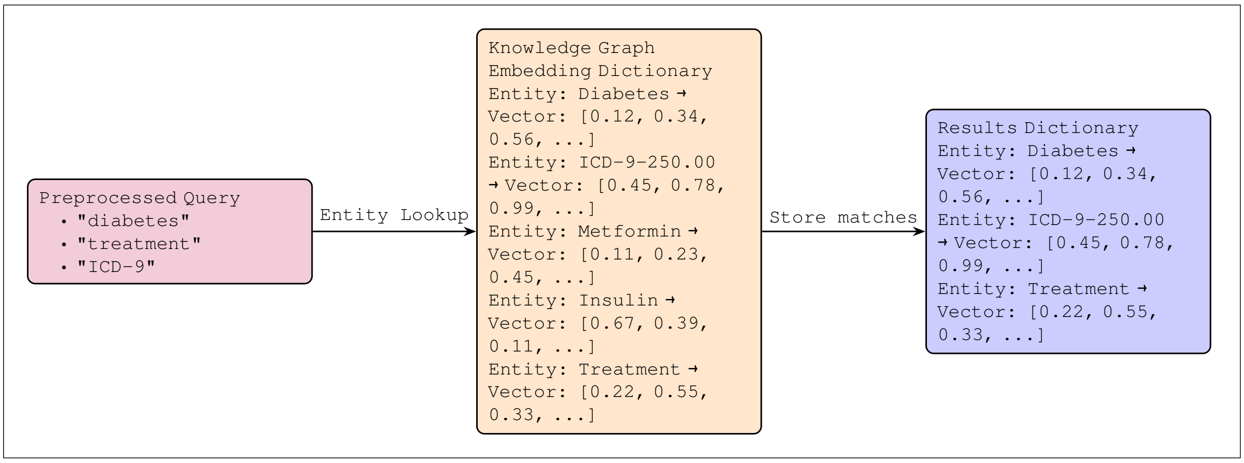


Figure 19: Entity Lookup with Bridging Semantic and Structural Spaces.

2.6.6 Mapping: Similarity Search Using HNSW

After enhancing the entity embeddings using the RGCN model, an entity similarity search is performed, this is a crucial step in measuring how well the graph-aware embeddings capture semantic proximity between nodes.

In this work, HNSW is employed, a graph-based indexing algorithm designed for efficient retrieval in high-dimensional spaces. The implementation is adapted to operate using cosine similarity, which is more suitable for the semantic structure of RDF2Vec and R-GCN embeddings.
The figure 20 illustrates the architecture of the HNSW algorithm, used in this work to efficiently perform similarity search over graph-based embeddings.

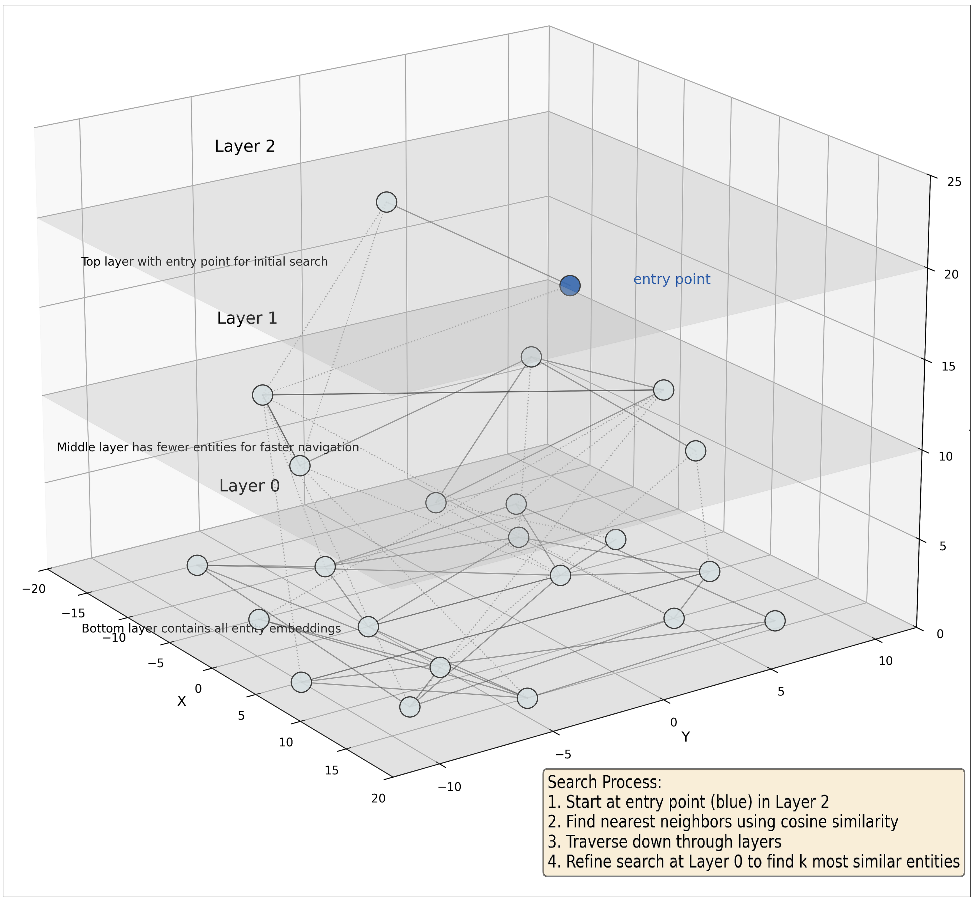


Figure 20: HNSW Diagram

The HNSW index builds a multi-layer graph structure, where each layer corresponds to different levels of granularity. The upper layers are used as a coarse navigation map and contain only a subset of the entities, allowing long-range jumps during query time. The lower layers contain more entities and allow fine-grained local exploration.

- Once the entity embeddings are optimized by the R-GCN model, they are then indexed using HNSW with the space parameter being set to 'cosine' as the distance metric instead of the default 'L2'.
- At retrieval time, search starts at an entry point in the top layer and greedily approaches neighbors with lower cosine similarity traversing the graph layer by layer, progressing down to the most detailed level.
- Finally, a refinement step at the bottom layer retrieves the top-k most semantically similar entities:

This layer-based traversal along with the angular measure of similarity renders the retrieval process efficient as well as semantically congruent with the RDF2Vec + R-GCN vector space. The following algorithm outlines the complete procedure for performing similarity search over medical entities.

Algorithm 9 Entity Similarity Search using HNSW

Input: entity_to_query, target entity URI
  `rgcn_model` $\leftarrow$ trained R-GCN model
  `data` $\leftarrow$ graph data (features, edges)
  `entity_to_idx` $\leftarrow$ entity-to-index mapping
  `top_k` $\leftarrow$ number of similar entities
  `ef_search` $\leftarrow$ HNSW search param (default: 200)

Output: Top-k most similar entities with similarity scores

begin

  Step 1: Embedding Extraction
  $E \leftarrow$ `rgcn_model(data)`; `// Forward pass to get embeddings`
  $E_{np} \leftarrow$ convert E to numpy array

  Step 2: HNSW Index Construction
  $index \leftarrow$ HNSW index with metric = `cosine`, dimension = $dim(E)$
  Initialize with $ef\_construction = 300$, $M = 24$
  Add E_{np} to index

  Step 3: Similarity Search
  $idx \leftarrow entity\_to\_idx[entity\_to\_query]$
  $results \leftarrow$ query HNSW for $top\_k + 1$ neighbors of $E_{np}[idx]$
  Exclude query entity if present in $results$

  Step 4:cosine similarity Computation
  foreach *neighbor in* $results$ **do**
    $cosine\_similarity \leftarrow 1 - cosine\_distance$
    Add $(entity, cosine\_similarity)$ to final list
  end

  return *Top-k most similar entities and cosine_similarities*

end

In this retrieval setup, cosine similarity is used instead of Euclidean distance because it better captures the angular closeness between vector representations a vital aspect when dealing with semantic embeddings.

Cosine similarity measures the cosine of the angle between two vectors of high dimensions. When the similarity is 1, it indicates that the vectors are in the same direction (semantically the same), and 0 indicates that they are orthogonal (completely unrelated).

The cosine similarity between two vectors A and B is given by the formula:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\| \times \|B\|}$$

where:

- $A \cdot B$ is the dot product of the vectors
- $\|A\|$, $\|B\|$ are their Euclidean norms (magnitudes).

2.6.7 Ranking and Retrieval

Following the mapping stage, which acquires an initial set of candidate entities through approximate nearest neighbor search (using HNSW), the system eliminates these candidates through a multi-criteria ranking process optimized for precision and relevance.
The figure 21 illustrates the process includes several key :

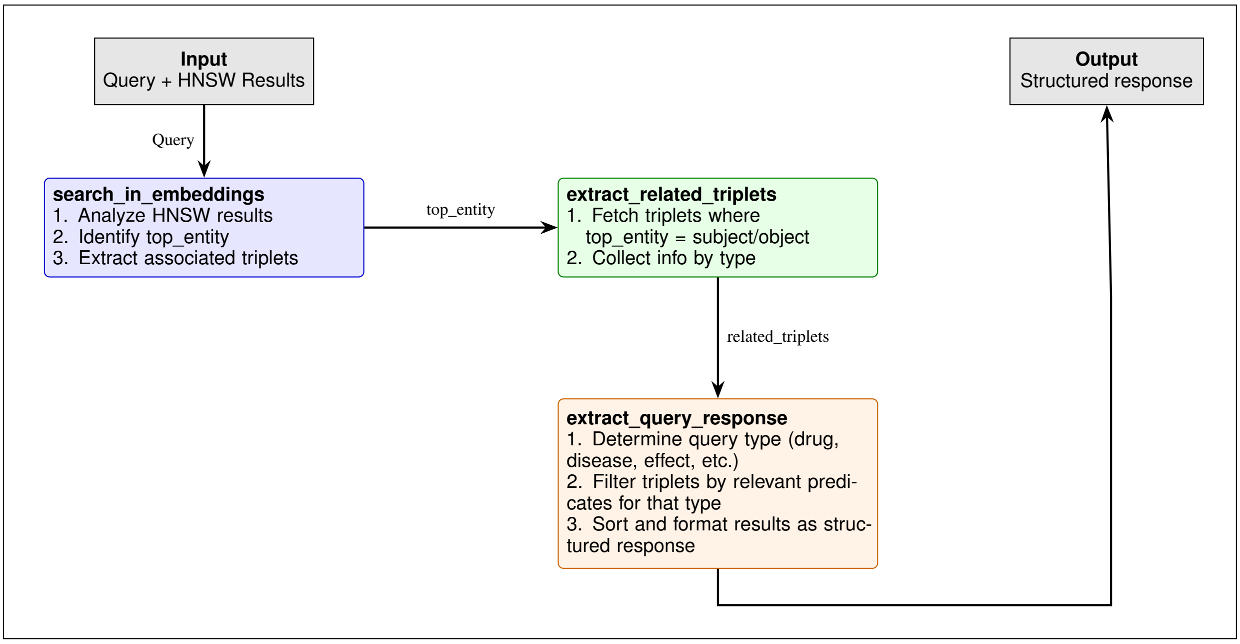


Figure 21: Ranking & Retrieval Process Pipeline

2.7 Conclusion

In this chapter,the pipeline of Information Retrieval system over a formal Medical Knowledge Base is prensented. The approach consists of several key components: generating semantic embeddings through RDF2Vec, refining the embeddings through a Relational Graph Convolutional network, and performing efficient similarity-based retrieval through HNSW indexing with cosine similarity.

In the next chapter, the implementation and application of the approach are presented so far, including testing different embedding models, neural network architectures and evaluating the results to validate the effectiveness of the solution.

Chapter 3

TESTS AND EVALUATION OF THE PROPOSED SOLUTION

3.1 Introduction

After finishing the design phase, this chapter presents the implementation of the system, followed by its evaluation to assess performance and real-world applicability.

The chapter begins by presenting the development environment and tools used throughout the implementation process. It then moves on to the evaluation phase, where the system is tested on a benchmark medical dataset across several experimental scenarios. These include the assessment of embedding models, similarity search methods, and semantic relevance using Bio_ClinicalBERT. The results are analyzed to quantify system performance and effectiveness. Following the experimentation, the chapter introduces the querying interface through a visual walkthrough of key system components, highlighting how users can interact with the system to retrieve semantically relevant medical information.

3.2 Hardware Environment

The work and tests presented in this thesis were carried out on a high-performance machine with the following specifications:

CPU	**GPU**	**RAM**	**Operating System**
11th Gen Intel(R) Core(TM) i7-1165G7	Intel(R) Iris(R) Xe Graphics	16 GB	windows 11

Table 6: Hardware Configuration

3.3 Software Environment

In this section, the software resources used during the implementation phase, including the programming language, frameworks, and libraries are presented in Table 7.

Software	Description
PyCharm[1]	is an integrated development environment (IDE) developed by JetBrains. designed for Python programming and offers features such as intelligent code completion, code inspections, debugging tools, and seamless integration with version control systems.
Python[2]	is a general-purpose, high-level language that is typically used for a wide variety of tasks and applications. Python is easy and readable, with a minimalist and clean syntax that makes programs easier to read and maintain. Python also boasts a rich standard library as well as an incredible amount of modules and frameworks.
Google Colab[3]	is a cloud-based Jupyter Notebook service that requires no configuration and provides free access to computing resources such as GPUs and TPUs. It is particularly well-suited for machine learning, data science, and educational purposes.
Numpy[4]	is a Python programming library designed to manipulate matrices and multidimensional arrays, as well as mathematical functions operating on these arrays.
Pandas[5]	is an open-source Python library for data manipulation and analysis, offering data structures such as DataFrame and Series to efficiently handle structured data. It is widely used in data science for tasks such as data cleaning and data aggregation.

[1] https://www.jetbrains.com/pycharm
[2] https://www.python.org/
[3] https://colab.research.google.com/
[4] https://numpy.org/
[5] https://pandas.pydata.org/

PyTorch[6]	is an open-source machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing.
HNSWlib[7]	is a fast library for finding similar items in large datasets. It uses a smart structure called "Hierarchical Navigable Small World graphs" to make similarity search efficient.
PyKEEN[8]	(Python Knowledge Embedding Engine) is used to create and train models that learn vector representations (embeddings) of entities and relations in a Knowledge Graph. Numerous state-of-the-art algorithms are included, allowing for easy experimentation with Knowledge Graph Embeddings.
PyTorch Geometric[9]	It is a powerful library that is used to easily build and train neural networks on graph structured data. When the data is organized as a network such as in social networks or Knowledge Graphs deep learning can be applied using models such as GCN and GAT through the PyTorch framework.
RDFLib[10]	is a Python library for working with RDF (Resource Description Framework) data, allowing parsing, querying, and serializing of RDF graphs.
Gensim[11]	is an open-source Python library designed for unsupervised topic modeling and natural language processing, particularly known for its efficient implementation of word embedding algorithms like Word2Vec.
Draw.io[12]	is a popular web-based application for creating diagrams and visualizations. It is very simple to use and contains a vast library of graphical objects that can be drag-and-dropped to form flowcharts, schemas, mockups, and other visualizations.

Table 7: Software Resources

[6] https://pytorch.org/
[7] https://github.com/nmslib/hnswlib
[8] https://github.com/pykeen/pykeen
[9] https://pytorch-geometric.readthedocs.io/
[10] https://rdflib.readthedocs.io/
[11] https://radimrehurek.com/gensim/
[12] https://app.diagrams.net/

3.4 Testing Procedure

To properly determine the efficiency of the Graph knowledge-based Information Retrieval system, several experiments are conducted on some of the most important pipeline stages.

3.4.1 STEP 1: Evaluation Using Metrics

Initially, various Embedding Models were compared and evaluated. The goal during this step was to determine the semantic encoding best suited for the kind of the PDD Medical Knowledge Base.

Then, different architectures of Neural Networks were experimented to further fine-tune these embeddings, and analyze their performance using several metrics according to the following procedure:

3.4.1.1 Embedding and GNN Combinations

A variety of Knowledge Graph embedding models were initially tested each aiming to encode entities and relations as vector space representations. The selected models, listed in table 8 , are efficient in modeling structured knowledge:

Model	Core Idea
TransE	Represents relations as translations in vector space; aims to make $h + r \approx t$ for each triple (h, r, t)
RotatE	Models relations as rotations in the complex vector space
DistMult	Bilinear model using element-wise multiplication and dot product; symmetric by nature
ConvE	Applies 2D convolution to (h, r) embeddings and compares to t
ComplEx	Extends DistMult to complex numbers; supports asymmetric relations via complex conjugation
RDF2Vec	Learns embeddings from RDF graph paths using Word2Vec on random walks

Table 8: Summary of Knowledge Graph Embedding Models

Once the embeddings are obtained, they are further refined using different Graph Neural Networks. The GNN architectures evaluated in this study include: Graph Convolutional Network, Graph Attention Network, Graphormer, and Relational Graph Convolutional Network.

3.4.1.2 Evaluation Metrics

The table 9 presents the definition, formula, explanation, and purpose of the different evaluation metrics used to measure the retrieval performance.

Metric	Definition	Formula	Explanation	Goal
MRR	Average of the reciprocal ranks of the first relevant entity for each query.	MRR $= \frac{1}{\lvert Q\rvert}\sum_{i=1}^{\lvert Q\rvert}\frac{1}{\text{rank}_i}$	rank_i: rank position of the correct entity for query i, Q: number of queries.	Maximize
Precision@k=5	Proportion of relevant entities among the top-$k = 5$ retrieved entities.	Precision@k=5 $= \frac{\#\text{relevant in top } k=5}{k=5}$	Measures how many of the top-$k = 5$ retrieved entities are actually relevant.	Maximize
Recall@k=5	Proportion of all relevant entities found in the top-$k = 5$ retrieved.	Recall@k=5 $= \frac{\#\text{relevant in top } k=5}{\#\text{relevant in total}}$	Measures the coverage of relevant entities in the top-5.	Maximize
MAP	Mean of the average precision across all queries.	MAP $= \frac{1}{\lvert Q\rvert}\sum_{i=1}^{\lvert Q\rvert}\text{AP}(q_i)$	Considers ranking of all relevant entities, not just the first one.	Maximize

Table 9: Evaluation Metrics Used for Similarity Retrieval

The table 3.4.1.2 presents a comparative overview of the evaluation metrics results for the different combinations of embedding models and GNN techniques.

GNN	Metric	RDF2Vec	TransE	RotatE	ConvE	DistMult	ComplEx
GCN	MAP	**0.6691**	0.5846	0.5324	0.4741	0.5979	0.5207
	Recall@k=5	0.2732	0.5466	0.5510	0.4750	**0.5567**	0.0536
	Precision@k=5	**0.6776**	0.5466	0.5517	0.4700	0.5567	0.5354
	MRR	0.0004	0.0004	0.7199	0.5400	0.5400	**0.7284**
R-GCN	MAP	**0.8717**	0.7461	0.7717	0.6365	0.8041	0.6815
	Recall@k=5	**0,6200**	0.5212	0.5551	0.5314	0.5872	0.3447
	Precision@k=5	**0,8040**	0.5212	0.5551	0.5314	0.5872	0.3447
	MRR	**0.9696**	0.0001	0.0001	0.0001	0.0001	0.0001
GAT	MAP	**0.5804**	0.3667	0.3333	0.4167	0.5333	0.5533
	Recall@k=5	**0.2001**	0.0003	0.0001	0.0002	0.0001	0.0001
	Precision@k=5	0.5000	0.4000	0.2000	0.4000	**0.6000**	0.4000
	MRR	0.0003	0.0035	0.0038	0.0278	**0.0063**	0.0003
Graphormer	MAP	**0.6800**	0.534000	0.502000	0.474167	0.487000	0.497000
	Recall@k=5	0.3000	**0.534000**	0.502000	0.475000	0.487000	0.497000
	Precision@k=5	**0.6900**	0.534000	0.502000	0.047500	0.048700	0.049700
	MRR	0.0005	**0.534109**	0.502226	0.474447	0.487205	0.497429

Table 10: Evaluation of GNN and Embedding Combinations Using Multiple Metrics

- R-GCN clearly stands out as the best-performing GNN model, especially when combined with RDF2Vec :

— MAP = 0.8717, MRR = 0.9696, Precision@5 = 0.8040.

— This demonstrates how structural relation enrichment with R-GCN significantly improves performance.

- **GCN** performs quite well, particularly with DistMult and ComplEx (MRR = 0.5400 and 0.7284), but poorly with RDF2Vec (MRR = 0.0004).
- The worst-performing GNN is GAT, with very low MRR scores across the board (especially with RDF2Vec and TransE), which shows that attention alone is not sufficient for this medical knowledge retrieval task.
- A newer model, Graphormer, does well with TransE and RDF2Vec but still falls behind R-GCN.

Therefore, it can be concluded that the best approach for entity similarity and Knowledge Graph embedding is the implementation of RDF2Vec and R-GCN together.

To support this conclusion,The figure 22 visualizes the training loss progression across 100 epochs, confirming the convergence, stability, and performance improvement of the RDF2Vec + R-GCN model:

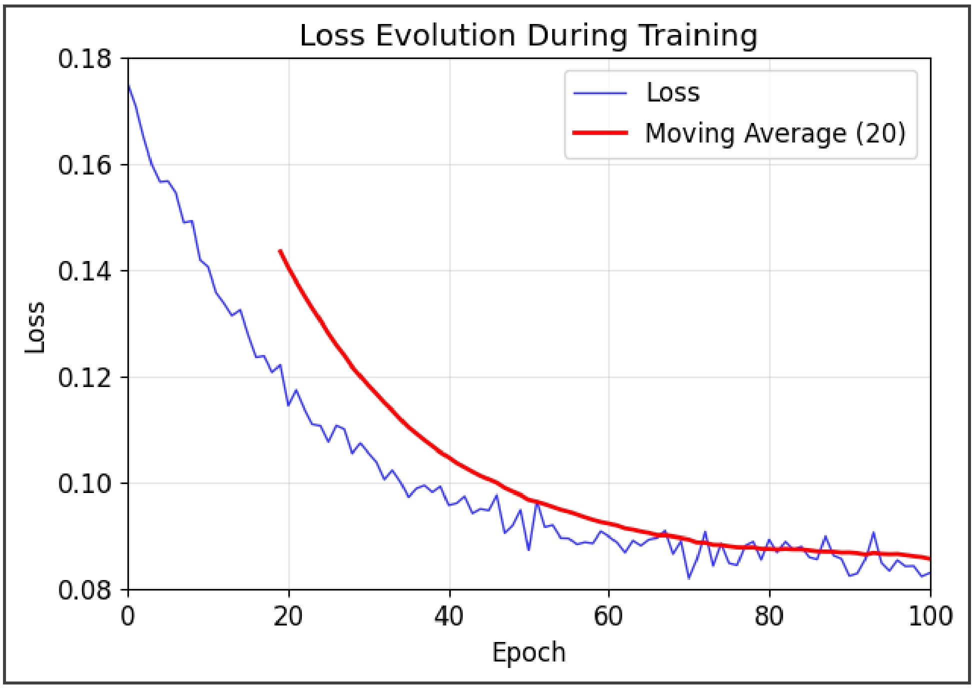


Figure 22: Evolution of Training Loss for The RDF2Vec + R-GCN Model

Analysis of the Full View of the Loss Over 100 Epochs:

— The curve starts at a high loss value (0.175) and drops rapidly in the first 20–30 epochs, showing fast initial learning.

— The loss gradually plateaus after 60 epochs, indicating diminishing improvements and convergence.

— The moving average line (red) confirms the stability and smoothing of fluctuations.

This demonstrates that the RDF2Vec + R-GCN model successfully minimized the loss and is not overfitting, as there's no sign of divergence or re-increase.

The figure 23 presents the evaluation metrics Precision@k, Recall@k, and nDCG@k used to assess the retrieval performance of the RDF2Vec + R-GCN model across various cutoff values:

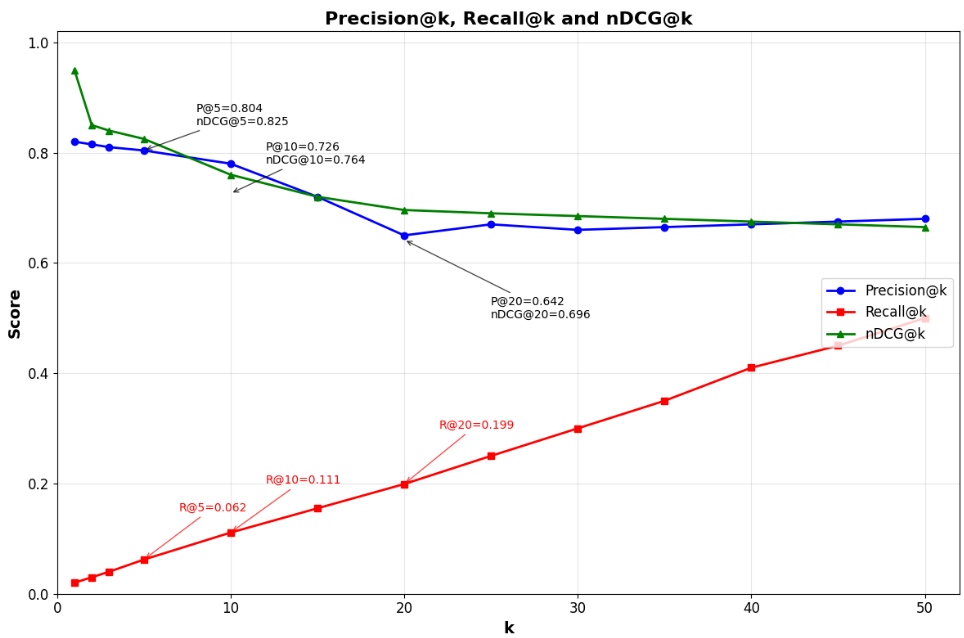


Figure 23: Precision@k, Recall@k and nDCG@k for RDF2Vec + R-GCN Retrieval

Additionally, the Precision@k, Recall@k, and nDCG@k plots offer an overall picture of retrieval performance for different k values, the model demonstrates with extremely good performance being exhibited at P@5 (0.804), nDCG@5 (0.825), and a continually increasing Recall@k confirms the effectiveness of the model in retrieving semantically relevant entities from the medical knowledge base.

3.4.2 STEP 2: Evaluation Using User Query

To further evaluate the effectiveness and practical value of the system, a second layer of testing is introduced focusing on realistic user queries. In this step, actual user interactions are simulated by formulating natural language questions and entity-based lookups much like a user or medical professional would when interacting with an intelligent Information Retrieval system.

This evaluation aims to verify not only the performance of the similarity search mechanisms, but also the ability of the pipeline, which combines RDF2Vec embeddings, Relational Graph Convolutional Networks, and Hierarchical Navigable Small World search, to return accurate, meaningful, and semantically relevant answers.

In the following subsection, first the similarity search techniques applied are detailed, before illustrating concrete examples of user queries and the system response.

3.4.2.1 Similarity Search Techniques

The immediate objective was to simulate semantic real-world searching: for every query medical entity, the system ought to respond with the most semantically relevant entities in the Knowledge Graph.
The following search methods for similarities had been tested:

a) **Cosine Similarity:** Cosine similarity measures the angle between two vectors in a multidimensional space. It is bounded between -1 and 1, where 1 indicates perfect alignment, and -1 indicates opposite directions.

$$\text{sim}_{\text{cos}}(h,t) = \frac{h \cdot t}{\|h\| \cdot \|t\|}$$

This was the primary metric for semantic similarity after training, especially in RDF2Vec and during Triplet Loss evaluation.

b) **k-Nearest Neighbors:** The k-Nearest Neighbors algorithm retrieves the top-k most similar vectors for a given query using a selected distance metric, this method is used with cosine distances.

c) **Hierarchical Navigable Small World:** HNSW is a graph-based Approximate Nearest Neighbor (ANN) search algorithm, optimized for high-performance vector retrieval in large-scale datasets. It constructs a hierarchical proximity graph that accelerates query search time. HNSW is evaluated with Cosine Distance measures.

Here is a comparative table and a graph that shows the results obtained using the three methods explained above:

Method	Top-1 Similarity	Overall Ranking Quality	Notable Features / Notes
Cosine Similarity	**0.9756**	High	Best precision, exact neighbor match on predicates
k-NN	0.6771	Moderate	Good generalization but slower, based on pairwise distances in embedding space
HNSW	**1.0000**	High (Top-5 > 0.9996)	Very fast and scalable ANN method, maintains high accuracy on dense and large embedding spaces

Table 11: Comparative Results of Search Methods for Similarities

The figure 24 presents a comparison of the Top-5 similarity scores obtained using various methods:

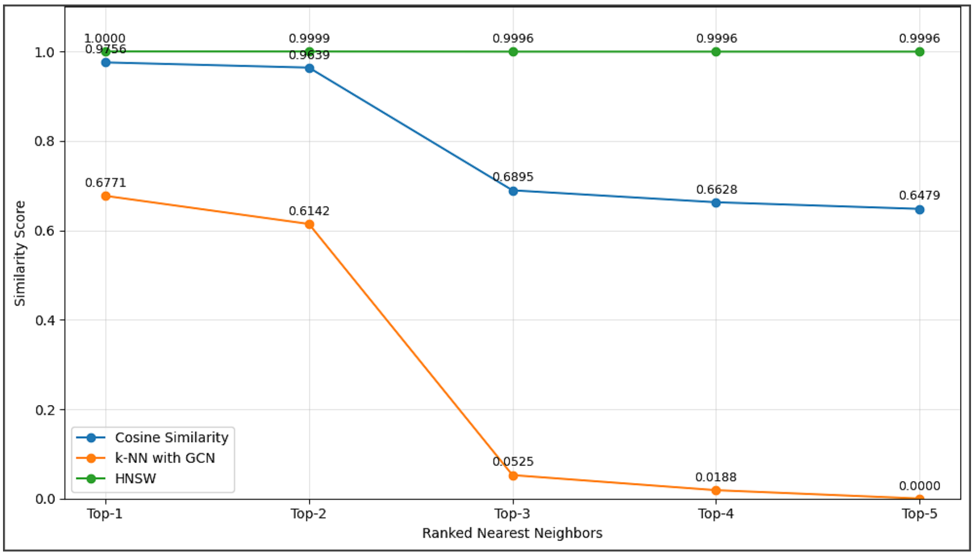


Figure 24: Top-5 Similarity Scores by Method with Values

These similarity scores were obtained on the medical entity subset of the knowledge Base using RDF2Vec and RGCN-enhanced embeddings, HNSW proves to be the most scalable and performant method for entity similarity search in enriched medical Knowledge Graphs. While pairwise cosine similarity is still valuable for evaluation and k-NN aids in embedding inspection, HNSW is the best suited for real-time and large-scale retrieval scenarios in this project.
Following the comparative evaluation of similarity methods, a realistic query-based test was conducted to simulate the use of the system in real-world scenarios. This was done by developing a sample collection of structured or textual queries representing common information needs in the healthcare field.

In order to effectively recover the most similar entities per query, the highest-performing embedding method and similarity method: RDF2Vec embeddings passed through a Relational Graph Convolutional network was used, then the Approximate Nearest Neighbor search was performed with HNSW.

The figure 25 is an example of one such request:

```
Test query: 'what is the category of disease of the icd code 432.1 ?'

Response:
For your search concerning the disease category of ICD code 432.1, I found the following information:
- CEREBROVASCULAR DISEASE
```

Figure 25: Request Example 1: ICD Code Category

This request demonstrates how the system can interpret a natural language query, map it to the corresponding concept in the medical Knowledge Graph (e.g., using RDF2Vec + R-GCN embeddings), and retrieve the most relevant class or category using HNSW-based similarity search. The retrieved answer is semantically and medically correct, showing that the system successfully identified Cerebrovascular Disease as the parent category of ICD code 432.1, which corresponds to Subarachnoid Hemorrhage. The figure 26 illustrates a second query example, where the system processes a drug identifier to retrieve its corresponding name and medical description, demonstrating its ability to handle structured entity-based queries:

```
Query test: 'give me the description of this ID DB00530?'

Found:
• http://example.org/name -> Erlotinib
• http://example.org/description -> Erlotinib is an inhibitor of the epidermal growth factor receptor (EGFR) tyrosine

DESCRIPTION:
--------------------------------------------------
Erlotinib is an inhibitor of the epidermal growth factor receptor (EGFR) tyrosine kinase that is used in the treatment of non-small

    cell lung cancer, pancreatic cancer and several other types of cancer.

It is typically marketed under the trade name Tarceva. Erlotinib binds to the epidermal growth factor receptor (EGFR) tyrosine
kinase in a reversible fashion at the adenosine triphosphate (ATP) binding site of the receptor. Recent studies demonstrate
that erlotinib is also a potent inhibitor of JAK2V617F, which is a mutant form of tyrosine kinase JAK2 found in most patients
with polycythemia vera (PV) and a substantial proportion of patients with idiopathic myelofibrosis or essential thrombocythemia.
This finding introduces the potential use of erlotinib in the treatment of JAK2V617F-positive PV and other myeloproliferative disorders.
```

Figure 26: Request Example 2: Drug Description Query

Another use case was tested involves querying the system with a drug identifier (ID) and retrieving its associated name and medical description. This simulates a scenario where a user or application inputs a known entity (e.g., from DrugBank) and expects structured output.

3.4.2.2 User Query Evaluation

To assess the quality of the results returned by the Information Retrieval system, a query evaluation module was included using semantic similarity between the query and the returned entities. The module employs Bio-ClinicalBERT, a language model pre-trained and fine-tuned in the biomedical domain. It represents both the query and all returned entities as a dense vector representations in one shared semantic space, upon which cosine similarity is computed.

The objective is to estimate the semantic utility of the entities discovered for the original question, regardless of the full lexical overlap. This contrasts with typical evaluation protocols founded on human relevance judgments since this approach provides automatic and accurate evaluation based on the underlying representations the model produces.

Among the ten queries evaluated using the semantic relevance module,a detailed analysis of a representative case is presented below:

Figure 27 demonstrates the Bio_ClinicalBERT evaluation framework in action, showing the automatic semantic similarity assessment between a drug query (DB00530) and the system response about Erlotinib

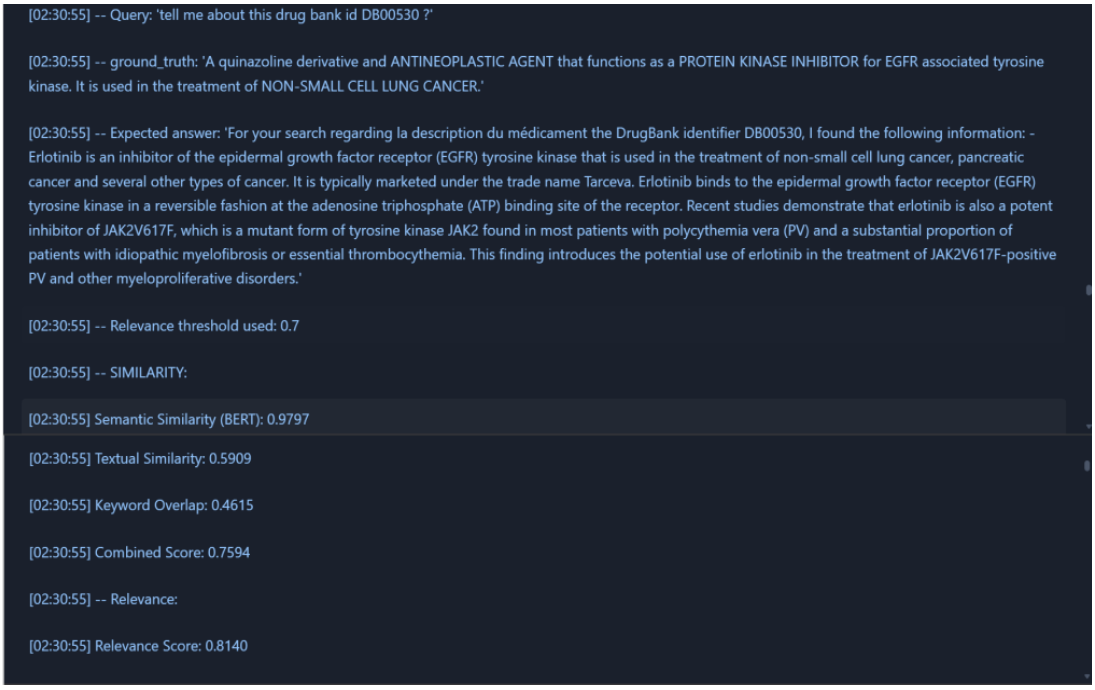


Figure 27: System Evaluation Output for a User Query Showing Semantic Similarity Analysis and Relevance Scoring Using Bio-ClinicalBERT Model

The following table presents a detailed evaluation of the response retrieved for the query. The analysis considers multiple dimensions including semantic similarity, textual overlap, and overall relevance, using metrics derived from domain-specific language models and textual analysis. Each metric is interpreted and classified to assess the system is performance in providing accurate and clinically appropriate responses.

Metric	Score	Performance	Interpretation
Semantic Similarity (BERT)	0.9758	Excellent	Near-perfect semantic understanding with Bio_ClinicalBERT.
Textual Similarity	0.5455	Expected	System provides detailed answers beyond surface-level matching.
Keyword Overlap	0.4615	Expected	Limited keyword match shows semantic rather than literal interpretation.
Combined Score	0.7438	Good	Balances semantic and lexical aspects well.
Relevance Score	0.8031	Excellent	System successfully detected high relevance.

Table 12: Query Evaluation Results Analysis

3.4.3 Knowledge Base Enrichment with ICD-9 and DrugBank Ontologies

To enhance the semantic depth and medical relevance of the knowledge Base, an enrichment step was performed using two popular medical ontologies: ICD-9-CM and DrugBank.
These ontologies were selected due to their prevalence in clinical diagnosis and pharmacological databases, respectively. The integration was done to map diseases and drugs in the graph to standardized ontology-based representations.

The enrichment was achievable because of the presence of shared identifiers between the core Knowledge Base and the external ontologies. Specifically, entities in the database were aligned with ICD-9 disease codes and DrugBank drug identifiers. The ICD-9 data was provided in XML/Turtle forma and DrugBank available as CSV data.

The pipeline begins with the merging of multiple N-Triples (.nt) files to construct a unified base graph (loading the PDD data). In the second phase, DrugBank enrichment was performed by parsing the CSV file to retrieve drug names and descriptions, which were then matched against existing drugbank:ID entries in the Knowledge Graph. Relevant descriptions were cleaned and added as literal triples, improving drug entity annotations.

In the third phase, ICD-9-CM enrichment was conducted by aligning diagnosis codes in the base graph with hierarchical disease entries from the ICD-9 ontology. Using string-based or range-based code matching, disease labels and subclass relationships were extracted from the Turtle file[13] and injected into the original diagnosis subgraph.

[13] A Turtle file (with the `.ttl` extension) is a plain text file used to represent RDF (Resource Description Framework) data in a compact, human-readable format. It is short for Terse RDF Triple Language.

The enrichment algorithm applied semantic alignment through shared identifiers, robust text preprocessing (escaping characters, removing references) and consistency checks. Although the process was limited by partial access to the full DrugBank RDF, integration still yielded a richer ontology-aware medical Knowledge Graph capable of supporting improved similarity computation and retrieval tasks. The figure 28 illustrates newly added triples resulting from the enrichment step:

```
<http://purl.bioontology.org/ontology/ICD9CM/753.0> <http://pdd.wangmengsd.com/property/hasDiseaseLabel> "CONGENITAL ANOMALIES"@en .
<http://bio2rdf.org/drugbank:DB00126> <http://example.org/name> "Ascorbic acid" .
<http://bio2rdf.org/drugbank:DB00694> <http://example.org/description> "A very toxic anthracycline aminoglycoside antineoplastic isolated
<http://pdd.wangmengsd.com/namedrug/Capecitabine> <http://pdd.wangmengsd.com/property/Rotavirus_Vaccine_Interactions> "The therapeutic ef
```

Figure 28: Example of Newly Added Triples after the Enrichment Step

3.4.4 Practical Demonstration of the Retrieval System Evaluation

In this section, the interfaces of the system are presented.

a) **Home Interface**

Figure 29 and 30 shows the home interface of the **MedSearch** system, a web-based platform dedicated to semantic search over medical Knowledge Graphs. This interface serves as the main entry point into the application, offering users a clear overview and quick access to the core functionality.

It consists of the following elements:

— **A top navigation menu:** Including three sections(Home, Data Sources, and Pipeline) enabling easy access to different stages of graph processing and project documentation.

— **A central search bar:** Allowing users to enter natural language queries (e.g. "What is the category of disease of the ICD code 432.1?") to explore entities and relationships within the Knowledge Graph.

— **A query examples section:** Providing guidance to users by showcasing sample questions related to ICD codes, drugs (DrugBank), or other entities in the graph.

— **A "Project Objective" box:** Which explains the aim of the system: to help healthcare professionals and researchers effectively query, explore, and understand medical knowledge using advanced embedding techniques and Relational Graph Convolutional Networks.

— **A "Log Console – Model Initialization" panel:** Located at the bottom of the interface, which displays system messages during model loading. For instance, connection errors such as "Erreur de connexion: Failed to fetch" help diagnose technical issues during initialization.

This interface is designed to combine ease of use with powerful exploration capabilities, supporting enhanced discovery of complex medical relationships across integrated Knowledge Graphs.
The figure 29 present the MedSearch system's home interface:

Figure 29: Search Interface

The Figure 30 displays the log console of the MedSearch interface, which provides real-time feedback during model initialization and system operations, aiding in error tracking and debugging:

Figure 30: Debug Console

b) **Data Sources Interface**

Figure 31 presents the Data Sources interface of the MedSearch system. This page introduces the foundational datasets integrated into the Medical Knowledge Base, offering users an overview of the origin and nature of the medical data used.

It is made up of three main data blocks:

— **PDD Graph :** This section describes the PDD Graph as a comprehensive Knowledge Graph that links Patient, drugs, and diseases. A link is provided to view the original data source.

— **DrugBank:** This block highlights DrugBank as a publicly accessible and detailed pharmaceutical database containing drug and drug target information. The system integrates multiple DrugBank sources to enrich the graph. Two links are available for further exploration: one to the Kaggle dataset and another to an XML example.

— **ICD-9-CM Ontology :** This section explains the ICD-9-CM ontology, which provides a hierarchical structure for classifying diseases, symptoms, and medical procedures. It helps ensure the accurate identification and categorization of medical conditions. A direct link to view the ontology is included.

The figure 31 illustrates the Data Sources interface of the MedSearch system, which provides users with detailed insight into the origin and reliability of the medical datasets integrated into the Knowledge Graph:

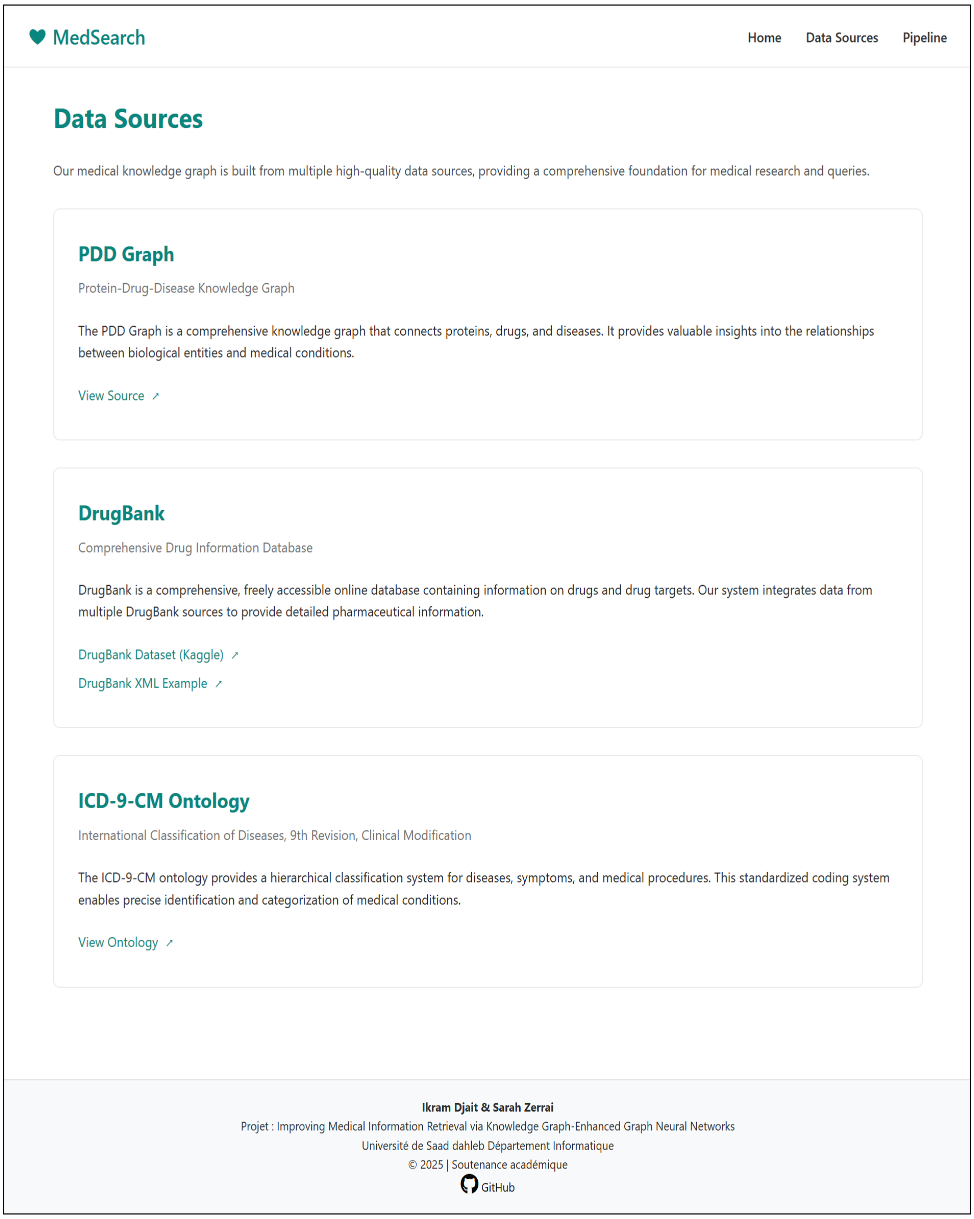


Figure 31: Data Sources Interface

c) **Pipeline Interface**

The figure 32 shows the Pipeline interface of the MedSearch application. This interactive page illustrates the key stages involved in transforming raw medical data and user queries into structured, query-able embeddings through a multi-step pipeline.

The pipeline is divided into two parallel paths: one for the knowledge base and one for the user query, eventually merging for final matching and retrieval.

Knowledge Base Path:

— **Preprocessing:**

This step involves cleaning and structuring the raw knowledge base data to prepare it for embedding generation.

— **Indexing:**

The preprocessed knowledge base is converted into vector embeddings to enable similarity-based search.

— Enrichment:

The initial embeddings are refined using a Relational Graph Convolutional Network, improving the quality of the semantic representation.

User Query Path:

— Preprocessing:

The user's textual input is cleaned and structured in preparation for embedding.

— Indexing:

The preprocessed query is transformed into a vector embedding compatible with the representation of the Knowledge Graph.

Merged Pipeline:

— Mapping:

This step involves matching the query embedding with the enriched knowledge base embeddings using HNSW similarity search.

— Ranking Retrieval:

The matched entities are ranked, and the most relevant ones are retrieved.

— User Interaction:

The results are presented to the user, allowing for interpretation and refinement of the output.

The figure 32 illustrates the Pipeline interface, where each stage of the system's workflow is interactively presented, allowing users to explore detailed explanations of the data transformation and query-processing steps.

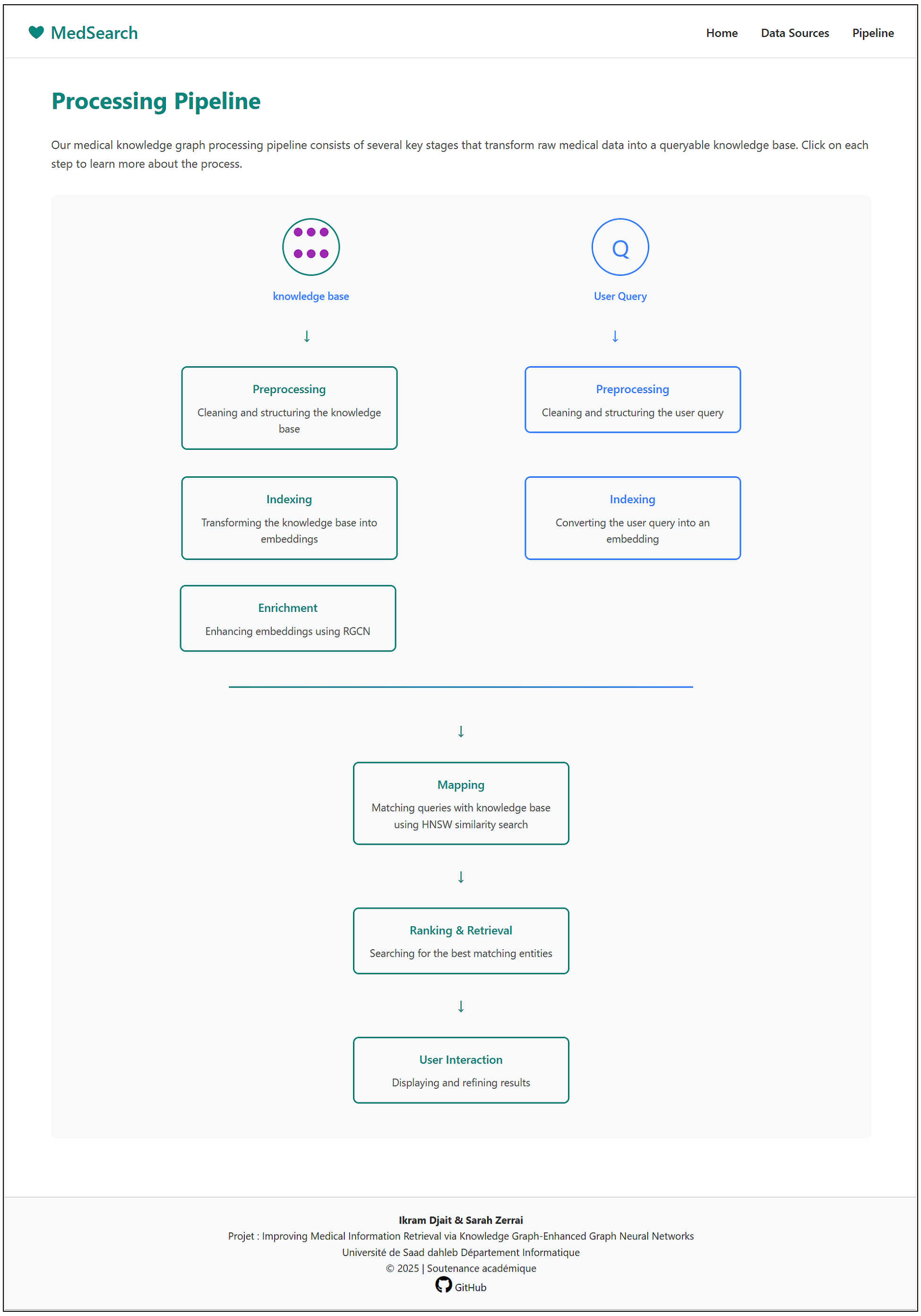


Figure 32: Pipeline Interface

User System Interaction:

The figure 33 illustrates the automated processing of a query related to a medication. A question containing the DrugBank identifier (DB09153) is submitted and then analyzed by the system. Relevant information is retrieved from the Knowledge Base, and a structured response is generated. In this example, the medication is correctly identified as sodium chloride, and its characteristics are provided.

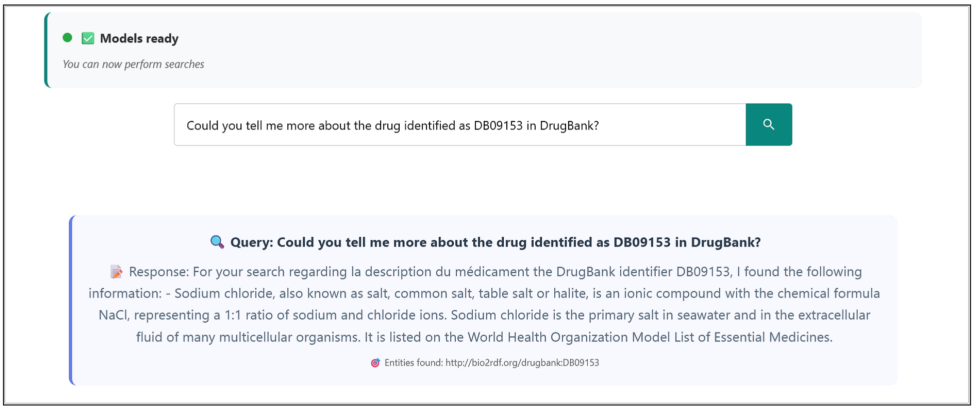


Figure 33: Example of User–System Interaction

3.5 Conclusion

In this chapter, the tools and environment used for the development of this system were presented. then was proceeded by the testing procedure to evaluate the system is performance. This was followed by an analysis of the results obtained through two evaluation steps: one based on performance metrics and the other based on user queries. Additionally, the enrichment of the Knowledge Base using the ICD-9 and DrugBank ontologies was discussed, which enhanced the system is semantic representation capabilities. Finally, the developed user interfaces were showcased, illustrating the structure and functionality of each component within the Medical Information Retrieval system.

General Conclusion

This project was motivated by the limitation of existing Medical Information Retrieval systems, their inability to effectively capture the complex structure and semantics of entities and relations in medical Knowledge Graphs.
Conventional Knowledge Graph embedding models are not likely to represent the rich relational and contextual information contained in biomedical data, and hence the accuracy and agility of retrieval systems in clinical applications are compromised.
In answer to this problem, a hybrid retrieval model was proposed, combining RDF2Vec embeddings and a Relational Graph Convolutional Network. This enables the combination of semantic and structural properties of medical entities and resulting in more informative and context-specific representations. The system's pipeline includes RDF graph transformation, embedding generation, GNN-based refinement, and efficient similarity search based on HNSW indexing. The model was validated under a question-answering setting, where it demonstrated high semantic relevance through evaluation using a semantic similarity module based on Bio_ClinicalBERT, which assesses the semantic alignment between user queries and retrieved entities, offering an automatic, consistent, and context-aware measure of result quality.
Therefore, this research successfully achieved its central objective of developing an augmented medical information retrieval system and validating it on a large benchmark medical dataset. By integrating Knowledge Graph Embeddings with Graph Neural Networks, the proposed framework demonstrates enhanced precision, adaptability, and semantic awareness in retrieving clinically relevant information.

To further enhance this work, several directions can be explored in the future:

- **Evaluation with domain experts:** Conducting large-scale qualitative and quantitative evaluations with healthcare professionals to validate system performance.
- **Dynamic Data Adaptation:** The system is currently constructed on top of a static copy of the Knowledge Graph. Extending the retrieval pipeline to accommodate dynamic or real time updates particularly in fast evolving fields like biomedical science is another important direction for future work.
- **Semantic Evaluation Enhancement:** The current evaluation relies on a semantic similarity module based on Bio_ClinicalBERT to assess the alignment between queries and retrieved

entities. While this provides a meaningful measure of contextual relevance, future work could extend this evaluation with human relevance judgments, user feedback, or interactive assessments to better reflect real world utility and user satisfaction in clinical settings.

- **User Behavior and Query Variability:** The system currently assumes a uniform query interpretation process. Addressing variations in user behavior, query formulation, and intent will be essential to improve personalization and relevance in retrieval results.

  These perspectives aim to push the boundaries of what intelligent, graph-powered medical Information Retrieval systems can achieve in practice.

Bibliography

[1] Amini, M.-R. and Gaussier, E. (2013). *Recherche d'information: Applications, modèles et algorithmes-Fouille de données, décisionnel et big data*. Editions Eyrolles.

[2] Bhatti, U. A., Tang, H., Wu, G., Marjan, S., and Hussain, A. (2023). Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence. *International Journal of Intelligent Systems*, 2023(1):8342104.

[3] Bi, X., Jiang, C., Yan, C., Zhao, K., Zhang, L., and Wang, J. (2024). Esgc-mda: Identifying mirna-disease associations using enhanced simple graph convolutional networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

[4] Celebi, R., Uyar, H., Yasar, E., and Sezerman, A. (2019). Evaluation of knowledge graph embedding approaches for drug-drug interaction prediction in realistic settings. *BMC Bioinformatics*, 20:726.

[5] Cheng, Q. and Lin, Y. (2023). Multilevel classification of users' needs in chinese online medical and health communities: Model development and evaluation based on graph convolutional network. *JMIR Formative Research*, 7:e42297.

[6] Choudhary, S., Luthra, T., Mittal, A., and Singh, R. (2021). A survey of knowledge graph embedding and their applications. *arXiv preprint arXiv:2107.07842*.

[7] Cui, Z., Yuan, Z., Wu, Y., Sun, X., and Yu, K. (2023). Intelligent recommendation for departments based on medical knowledge graph. *IEEE Access*, 11:25372–25385.

[8] Dai, Y., Wang, S., Xiong, N. N., and Guo, W. (2020). A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5):750.

[9] Dara, O. N., Ibrahim, A. A., and Mohammed, T. A. (2024). Advancing medical imaging: detecting polypharmacy and adverse drug effects with graph convolutional networks (gcn). *BMC Medical Imaging*, 24(1):174.

[10] de Mello, L. M. and Galvão, M. C. B. (2022). Searching for health information for academic and clinical purposes: possibilities and challenges. *Research, Society and Development*, 11(3):e41011326726–e41011326726.

[11] Dong, H., Wang, P., Xiao, M., Ning, Z., Wang, P., and Zhou, Y. (2024). Temporal inductive path neural network for temporal knowledge graph reasoning. *Artificial Intelligence*, 329:104085.

[12] Ebeid, I. A. (2022). Medgraph: A semantic biomedical information retrieval framework using knowledge graph embedding for pubmed. *Frontiers in Big Data*, 5. Open Access.

[13] Huang, X., Zhang, J., Xu, Z., Ou, L., and Tong, J. (2021). A knowledge graph based question answering method for medical domain. *PeerJ Computer Science*, 7:e667.

[14] James, D. A. (1995). *The Application of Classical Information Retrieval Techniques to Spoken Documents*. Phd thesis, University of Cambridge, Cambridge, UK.

[15] Jiang, W. and Long, Y. (2025). A graph convolutional network face recognition algorithm based on clustering. *Proceedings of SPIE - The International Society for Optical Engineering*, 144.

[16] Kejriwal, M. (2022). Knowledge graphs: A practical review of the research landscape. *Information*, 13(4):161.

[17] Li, Y., Zhou, S., Yang, Z., Song, W., Song, X., and Zhang, Y. (2023). Kindmed: Knowledge-induced medicine prescribing network. *arXiv preprint arXiv:2310.14552*.

[18] Mao, C., Yao, L., and Luo, Y. (2022). Medgcn: Medication recommendation and lab test imputation via graph convolutional networks. *Journal of Biomedical Informatics*, 127:104000.

[19] Oludele, A. et al. (2012). Information retrieval: An overview. *International Journal of Advanced Research in Computer Science*, 3(5).

[20] Portisch, J. and Paulheim, H. (2024). The rdf2vec family of knowledge graph embedding methods: An experimental evaluation of rdf2vec variants and their capabilities. *Semantic Web*, 15(3):845–876.

[21] Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2018a). Modeling relational data with graph convolutional networks. In *The Semantic Web - ISWC 2018*, pages 593–607. Springer, Cham.

[22] Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2018b). Modeling relational data with graph convolutional networks. In *Proceedings of the European Semantic Web Conference (ESWC)*, pages 593–607. Springer.

[23] Silva, A. M. B. d., Ferreira, N. C. d. S., Braga, L. A. M., Mota, F. B., Maricato, V., and Alves, L. A. (2024). Graph neural networks: A bibliometric mapping of the research landscape and applications. *Information*, 15(10):626.

[24] Sirojiddin, S. and Marxabo, Z. (2023). Graph neural network and its applications. In *Advances in Systems Analysis, Software Engineering, and High Performance Computing Book Series*, pages 19–32.

[25] Superlinked (2024). Answering questions with knowledge graph embeddings. `https://superlinked.com/vectorhub/articles/answering-questions-knowledge-graph-embeddings`. Web article, published January 5, 2024.

[26] Thanapalasingam, T., van Berkel, L., Bloem, P., and Groth, P. (2022). Relational graph convolutional networks: A closer look. *PeerJ Computer Science*, 8:e1073.

[27] Wang, M., Zheng, J., Liu, J., Hu, W., Wang, S., Li, X., and Liu, W. (2017). Pdd graph: Bridging electronic medical records and biomedical knowledge graphs via entity linking. In *International Semantic Web Conference (ISWC)*, pages 219–227. Springer, Cham.

[28] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2020). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*. arXiv:1901.00596.

[29] Zhang, S., Tong, H., Xu, J., and Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23.

[30] Zitnik, M., Agrawal, M., and Leskovec, J. (2018). Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466.